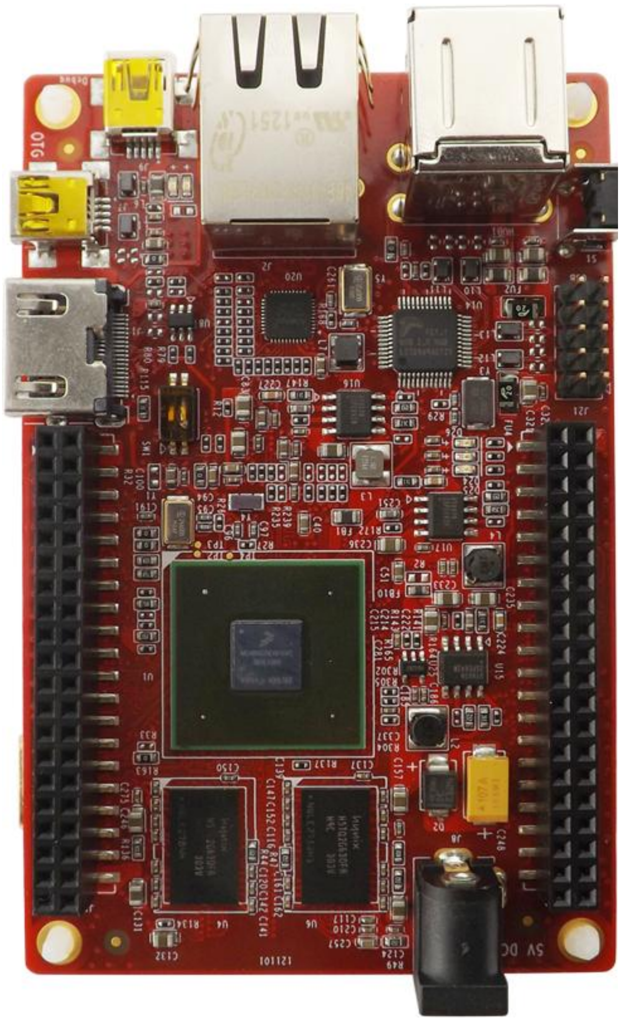
MarS Board



User Manual

Version 2.1---Dec. 27th, 2021

Copyright Statement:

Revision History:

Version	Date	Note
1.0	2013-3-29	Original Version
1.1	2013-5-11	Revision
1.2	2013-11-1	Revision
2.0	2014-8-25	Kernel upgrade to 3.0.35, Android upgrade to 4.3
2.1	2021-12-27	Upgrade MFGTOOLS to run under Win10

Table of Contents

Chapter 1 Product Overview

1.1 Introduction

MarS Board is an evaluation board designed by Embest Technology and based on Freescale's i.MX 6Dual processor. i.MX 6Dual integrates ARM Cortex™-A9 core of up to 1GHz, 2D and 3D graphics processors and 3D 1080p video processor. MarS Board is featured with abundant interfaces such as HDMI, LVDS, mini USB OTG, mini USB debug, RJ45, USB host, TF card and LCD display to help developers from different fields including netbooks, all-in-one PCs, high-end mobile Internet devices, handhold computers, portable media players, game consoles and portable navigation devices.

1.2 Packing List

- MarS Board
- Accessories package (option)
  - HDMI Cable
  - Mini USB Cable
  - 5V@4A power adapter
  - 4GB TF Card
  - Gigabit Ethernet line
- Other Options

1.3 Product Features

- **Product Parameters:**
  - Dimensions: 65mm x 102mm
  - Operation Temperature: 0 ~ 70℃
  - Operating Humidity: 20% ~ 90% (Non-condensing)

- Power Supply: 5V

- **Processor:**
  - i.MX 6Dual integrates ARM Cortex™-A9 core
  - 32 KByte L1 Instruction Cache
  - 32 KByte L1 Data Cache
  - Private Timer and Watchdog
  - Cortex-A9 NEON MPE (Media Processing Engine) Coprocessor
  - 2D/3D Graphics Processors

- **On-Board Memories:**
  - 4GByte eMMC
  - 4*256MB DDR3 SDRAM

- **On-Board Interfaces and Buttons:**
  - A HDMI Interface
  - A LVDS Interface
  - A LCD Interface
  - Two 480Mbps High-Speed USB2.0 Hub Interface
  - Two 480Mbps High-Speed USB2.0 Header Interface
  - A 480Mbps High-Speed USB2.0 OTG Interface
  - A COM-USB Debug (com2) Interface
  - A TF Card Interface
  - A 10/100M/1Gbps RJ45 Network Interface
  - A Boot Mode Interface
  - A Reset Button

- **Signals of On-Board Interfaces:**
  - An AUDMUX (Digital Audio Multiplexer) Signal
  - Two CAN Signals
  - A ECSPI2 (Enhanced Configurable SPI) Signal
  - Two I2C Signals
  - A camera/ Parallel signal, up to 16 bit

- A KPP (Keypad Port) Signal
- A PWM (Pulse Width Modulation) Signal
- A GPMI (General Purpose Memory Interface) Signal
- A SPDIF (Sony/Philips Digtal Interface) Signal
- Four UART Signals
- A USDHC1 (Ultra Secured Digital Host Controller) Signal
- A ESAI (Enhanced Serial Audio Interface) Signal

Note:

Pins of some interfaces listed above are multiplexed; please refer to data sheet of the processor and product schematic.

1.4 System Block Diagram

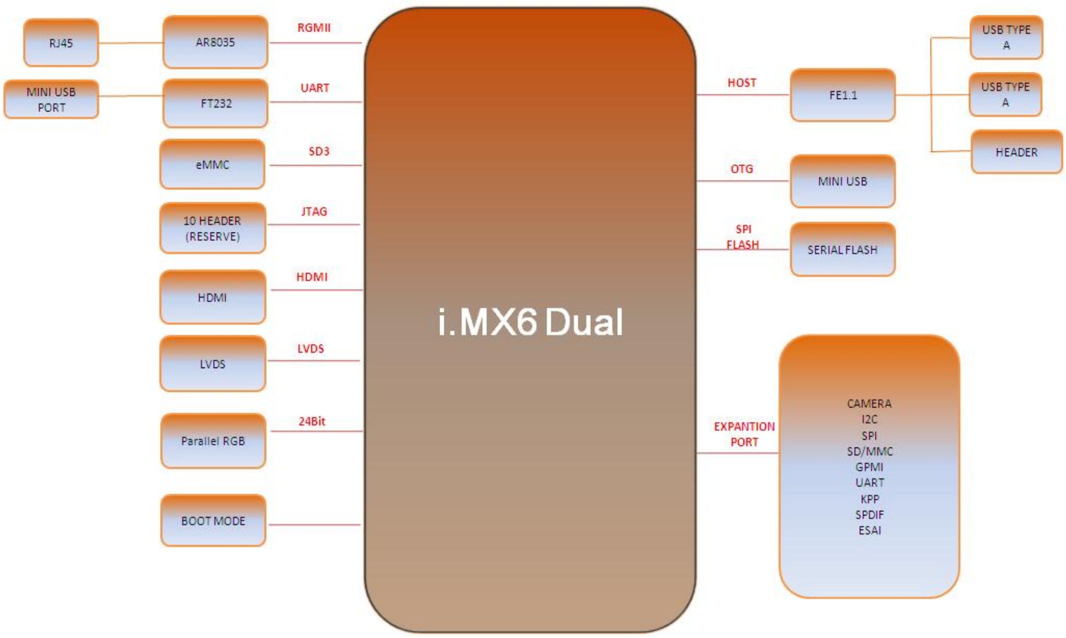


Figure 1-1 System block diagram of MarS Board

1.5 Hardware Dimensions (mm)

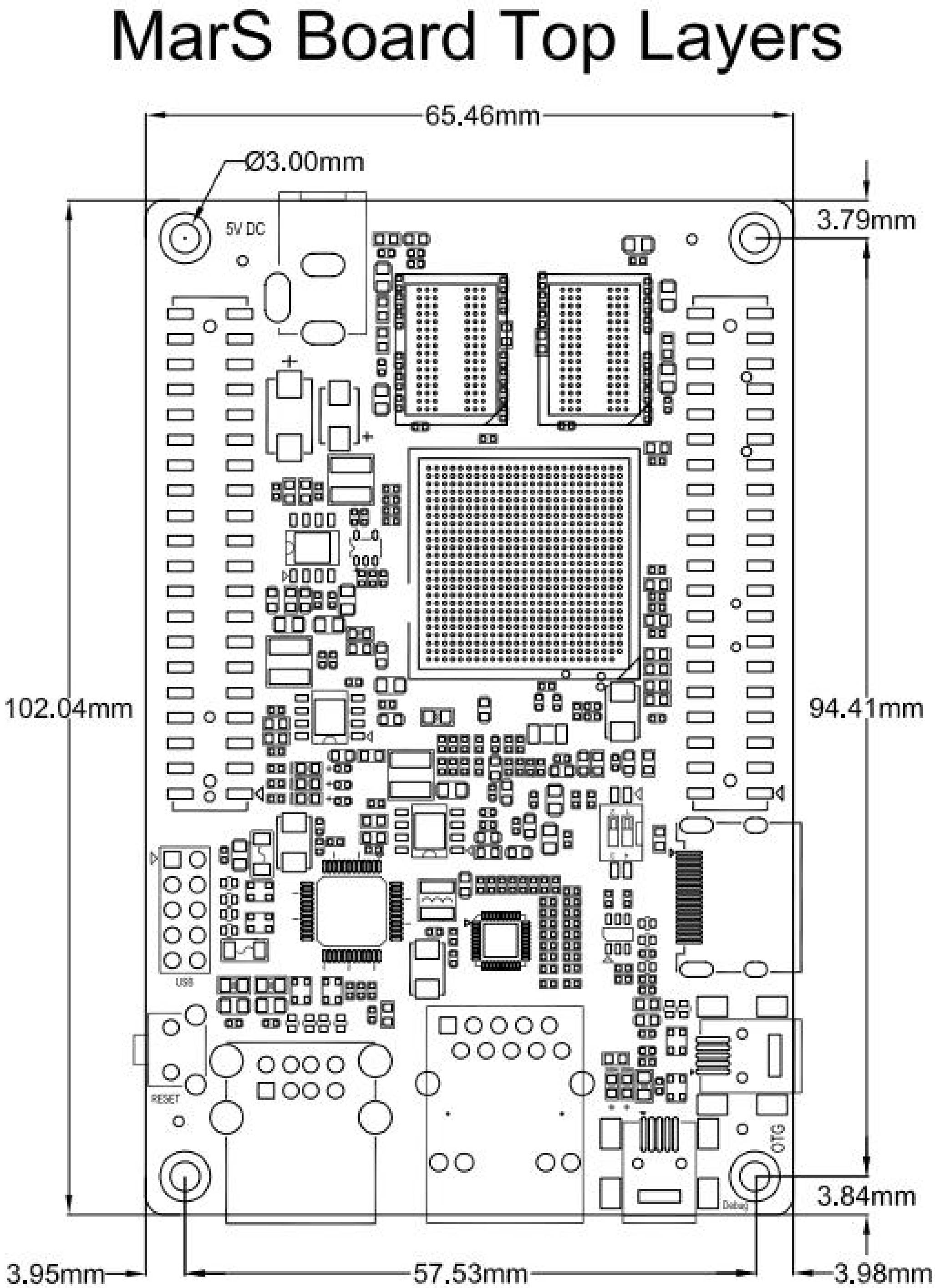


Figure 1-2 Dimensions of MarS Board

Chapter 2 Introduction to Hardware

This chapter will help you learn about the hardware composition of MarS Board by briefly introduce CPU, peripheral ICs and pin definition of various interfaces on the product.

2.1 CPU Introduction

i.MX 6Dual is an ARM™ Cortex-A9-based dual-core processor from Freescale. It runs at up to 1GHz, integrates 2D/3D graphics, 3D 1080p video processor and power management, and provides 64-bit DDR3/LVDDR3/LVDDR2-1066 interfaces as well as many other interfaces such as high-definition display and camera.

2.1.1 Clock Signals

The clock signals of i.MX 6Dual include a 32.768 KHz RTC clock and a 24 MHz external clock;

- **RTC Clock:** generated by an external crystal for low-frequency calculation;
- **External Clock:** used to generate main clock signal for PLL, CMM and other modules;

2.1.2 Reset Signal

Reset singal is determined by POR_B of CPU; low level validates reseting.

2.1.3 General Interfaces

General interfaces include 7 sets of GPIOs, each of which provides 32 dedicated GPIO pins (except GPIO7 which has 14 pins), and therefore the total pin number of GPIO can be up to 206.

2.1.4 Display Interface

- A parallel 24-bit RGB interface, supports 60Hz WUXGA output
- Two LVDS interfaces, support up to 165 Mpixels/sec output
- A HDMI 1.4 interface
- A MIPI/DSI interface with 1Gbps output rate

2.1.5 3D Graphics Acceleration System

i.MX 6Dual integrates GPU3Dv4 3D graphics processing unit which provides hardware acceleration for 3D graphics algorithms and allows desktop quality interactive graphics applicatios reach up to HD1080p resolution. The GPU3D supports OpenGL ES 2.0, including extensions, OpenGL ES 1.1, and OpenVG 1.1.
Additioally, i.MX 6Dual also has a GPUVGv2 vector graphics processing unit which provides hardware acceleration for 2D graphics algorithms.

2.2 Peripheral ICs around CPU

2.2.1 eMMC Flash NCEMBM11-04G

NCEMBM11-04G is an eMMC flash memory on MarS Board with 4GB memory space. The flash supports high-speed DDR data transfer at a clock frequency of up to 52MHz, as well as three widths of data line: 1-bit (default), 4-bit and 8-bit. The synchronous power managemet allows flash feature fast boot, automatical termination and sleep; meanwhile, NCEMBM11-04G supports hight-speed dual-data-transfer boot mode.

2.2.2 DDR H5TQ2G63DFR-H9C

H5TQ2G63DFR-H9C is a DDR3 SDRAM on MarS Board with 256MB memory space. It is suited for high-capacity and high-bandwidth applications and supports differential clock iput, differential data strobe, automatical refresh and asynchronous pin reset. MarS

Board has 4 chips of H5TQ2G63DFR-H9C summing up to 1GB.

2.2.3 AR8035 Ethernet PHY

AR8035 is a single port 10/100/1000 Mbps tri-speed Ethernet PHY feaured with low power and low cost. AR8035 supports MAC.TM RGMII interface and IEEE 802.3az-2010, Energy Efficient Ethernet (EEE) standard through proprietary SmartEEE technology, improving energy efficiency in systems using legacy MAC devices without 802.3az support. MarS Board can be either connected to a hub with a straight-through network cable, or to a PC with a cross-over network cable.

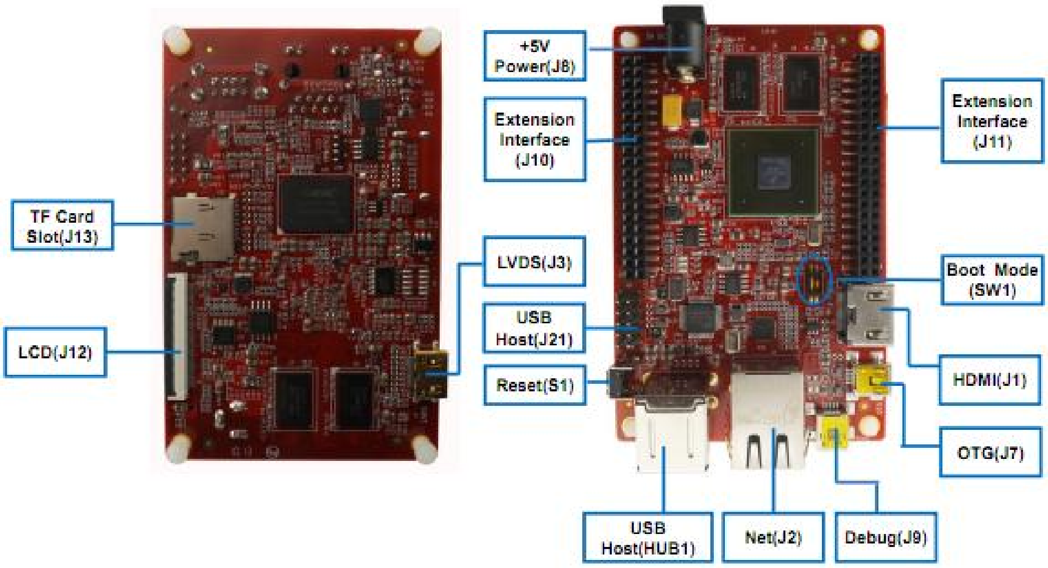
2.2.4 FE1.1 USB Hub

FE1.1 is a USB 2.0 high-speed 4-port hub solution. It uses USB3320 to provide 4 extended USB interface with support for high-speed (480MHz), full-speed (2MHz) and low-speed (1.5MHz) mode.

2.2.5 FT232RQ USB to UART Chip

FT232RQ is a USB-to-UART chip which realizes mini USB debug interface on MarS Board. It integrates a 1024-bit internal EEPROM and CBUS I/O configuration, and supports data transfer rates from 300 baud~3 Mbaud at TLL levels.

2.3 Hardware Interfaces on Mars Board



Hardware Interfaces on MarS Board

2.3.1 Power Jack (J8)

Table 2-1 Power Jack

Pins	Definitions	Descriptions
1	GND	GND
2	+5V	Power supply (+5V) 4A (Type)
3	+5V	Power supply (+5V) 4A (Type)

2.3.2 HDMI Interface (J1)

Table 2-2 HDMI Interface

Pins	Definitions	Descriptions
1	DAT2+	TMDS data 2+
2	DAT2_S	TMDS data 2 shield
3	DAT2-	TMDS data 2-
4	DAT1+	TMDS data 1+
5	DAT1_S	TMDS data 1 shield
6	DAT1-	TMDS data 1-
7	DAT0+	TMDS data 0+
8	DAT0_S	TMDS data 0 shield
9	DAT0-	TMDS data 0-

Pins	Definitions	Descriptions
10	CLK+	TMDS data clock+
11	CLK_S	TMDS data clock shield
12	CLK-	TMDS data clock-
13	NC	NC
14	NC	NC
15	SCL	IIC master serial clock
16	SDA	IIC serial bidirectional data
17	GND	GND
18	5V	5V
19	HPLG	Hot plug and play detect

2.3.3 LVDS Interface (J3)

Table 2-3 LVDS Interface

Pins	Definitions	Descriptions
1	3V3	+3.3V
2	LVDS_TX2_P	LVDS Data2+
3	LVDS_TX2_N	LVDS Data2-
4	GND	GND
5	LVDS_TX1_P	LVDS Data1+
6	LVDS_TX1_N	LVDS Data1-
7	GND	GND
8	LVDS_TX0_P	LVDS Data0+
9	LVDS_TX0_N	LVDS Data-
10	GND	GND
11	LVDS_CLK_P	LVDS_CLK+
12	LVDS_CLK_N	LVDS_CLK-
13	LCD_PWR_EN	Touch Reset Signal
14	Touch_Int	Touch Interrupt Signal
15	I2C_SCL	IIC Master Serial Clock
16	I2C_SDA	IIC Master Serial Data
17	LED_PWR_EN	Backlight Enable
18	5V	+5V
19	PWM	Pulse Width Modulation

2.3.4 USB OTG Interface (J7)

Table 2-4 USB OTG Interface

Pins	Definitions	Descriptions
1	VBUS	+5V
2	DN	USB Data-
3	DP	USB Data+
4	ID	USB ID
5	GND	GND

2.3.5 USB Debug Interface (J9)

Table 2-5 USB Debug Interface

Pins	Definitions	Descriptions
1	VBUS	+5V
2	DN	USB Debug Data-
3	DP	USB Debug Data+
4	NC	NC
5	GND	GND

2.3.6 Ethernet Interface (J2)

Table 2-6 Ethernet Interface

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

2.3.7 USB Hub Interface (Hub1)

Table 2-7 USB Hub Interface

Pins	Definitions	Descriptions
1	APV	5V power for HUB A
2	AD-	USB HUB A Data-
3	AD+	USB Debug Data+
4	GNDA	USB HUB A GND
5	BPV	5V power for HUB B
6	BD-	USB HUB B Data-
7	BD+	USB HUB B Data+
8	GNDB	USB HUB B GND

2.3.8 USB Hub Extension Interface (J21)

Table 2-8 USB HUB Extension Interface

Pins	Definitions	Descriptions
1	PWR2	5V power for HUB 2
2	PWR1	5V power for HUB 1
3	DM2	USB HUB 2 Data-
4	DM1	USB HUB 1 Data-
5	DP2	USB HUB 2 Data+
6	DP1	USB HUB 1 Data+
7	GND	GND
8	GND	GND
9	GND	GND
10	GND	GND

2.3.9 TF Card Interface (J13)

Table 2-9 TF Card Interface

Pins	Definitions	Descriptions
1	DAT2	Card data 2
2	DAT3	Card data 3
3	CMD	Command Signal
4	VDD	VDD
5	CLK	Clock

Pins	Definitions	Descriptions
6	VSS	VSS
7	DAT0	Card data 0
8	DAT1	Card data 1
9	CD	Card detect

2.3.10 LCD Interface (J12)

Table 2-10 LCD Interface

Pins	Definitions	Descriptions
1	B0	LCD Pixel data bit 0
2	B1	LCD Pixel data bit 1
3	B2	LCD Pixel data bit 2
4	B3	LCD Pixel data bit 3
5	B4	LCD Pixel data bit 4
6	B5	LCD Pixel data bit 5
7	B6	LCD Pixel data bit 6
8	B7	LCD Pixel data bit 7
9	GND1	GND
10	G0	LCD Pixel data bit 8
11	G1	LCD Pixel data bit 9
12	G2	LCD Pixel data bit 10
13	G3	LCD Pixel data bit 11
14	G4	LCD Pixel data bit 12
15	G5	LCD Pixel data bit 13
16	G6	LCD Pixel data bit 14
17	G7	LCD Pixel data bit 15
18	GND2	GND
19	R0	LCD Pixel data bit 16
20	R1	LCD Pixel data bit 17
21	R2	LCD Pixel data bit 18
22	R3	LCD Pixel data bit 19
23	R4	LCD Pixel data bit 20
24	R5	LCD Pixel data bit 21
25	R6	LCD Pixel data bit 22
26	R7	LCD Pixel data bit 23
27	GND3	GND
28	DEN	AC bias control (STN) or pixel data enable (TFT)
29	HSYNC	LCD Horizontal Synchronization

Pins	Definitions	Descriptions
30	VSYNC	LCD Vertical Synchronization
31	GND	GND
32	CLK	LCD Pixel Clock
33	GND4	GND
34	X+	X+ Position Input
35	X-	X- Position Input
36	Y+	Y+ Position Input
37	Y-	Y- Position Input
38	SPI_CLK	SPI serial clock
39	SPI_MOSI	SPI Master Output, Slave Input
40	SPI_MISO	SPI Master Input, Slave Output
41	SPI_CS	SPI Chip Select
42	IIC_CLK	IIC master serial clock
43	IIC_DAT	IIC serial bidirectional data
44	GND5	GND
45	VDD1	3.3V
46	VDD2	3.3V
47	VDD3	5V
48	VDD4	5V
49	RESET	Reset
50	PWREN	Backlight enable

Note:

📖 Please Do Not hot plug LCD flat cable.

2.3.11 AUDMUX (Digital Audio Multiplexer) Interface (J11)

Table 2-11 AUDMUX Interface

Pins	Definitions	Descriptions
31	AUD3_RXD	Receive audio data
25	AUD3_TXC	Audio transmission clock
27	AUD3_TXD	Transmit audio data
29	AUD3_TXFS	Transmit audio frame signal

2.3.12 CAN1 Interface (J11)

Table 2-12 CAN1 Interface

Pins	Definitions	Descriptions
33	RXCAN	Receive data
35	TXCAN	Transmit data

2.3.13 CAN2 Interface (J11)

Table 2-13 CAN2 Interface

Pins	Definitions	Descriptions
37	RXCAN	Receive data
39	TXCAN	Transmit data

2.3.14 ECSPI2 (Enhanced Configurable SPI) Interface (J10)

Table 2-14 ECSPI2 Interface

Pins	Definitions	Descriptions
21	MISO	Master Input Salve Output
19	MOSI	Master Output Salve Input
17	SCLK	Clock
15	SS0	Chip select

2.3.15 I2C1 Interface (J11)

Table 2-15 I2C1 Interface

Pins	Definitions	Descriptions
38	SCL	Master serial clock
40	SDA	Master serial data

2.3.16 I2C3 Interface (J11)

Table 2-16 I2C3 Interface

Pins	Definitions	Descriptions
3	SCL	Master serial clock
5	SDA	Master serial data

2.3.17 IPU1 (Image Processing Unit 1) Interface (J11)

Table 2-17 IPU1 Interface

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

2.3.18 KPP Keyboard Interface (J11)

Table 2-18 KPP Interface

Pins	Definitions	Descriptions
30	COL[0]	Keypad matrix column 0 output
34	COL[1]	Keypad matrix column 1 output
35	COL[2]	Keypad matrix column 2 output
28	ROW[0]	Keypad matrix row 0 input
32	ROW[1]	Keypad matrix row 1 input
37	ROW[2]	Keypad matrix row 1 input

2.3.19 PWM (Pulse Width Modulation) Interface (J10 & J11)

Table 2-19 PWM Interface

Pins	Definitions	Descriptions
26(J11)	PWM1	Pulse Width Modulation
13(J10)	PWM4	Pulse Width Modulation

2.3.20 GPMI (General Purpose Memory Interface) (J10)

Table 2-20 GPMI Interface

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

2.3.21 SPDIF (Sony/Philips Digital Interface) (J10)

Table 2-21 SPDIF Interface

Pins	Definitions	Descriptions
25	IN1	I2S data Input
23	OUT1	I2S data output
29	PLOCK	System master clock
27	SPDIF_EXTCLK	I2S frame clock
31	SRCLK	I2S bit clock

2.3.22 UART1 Interface (J11)

Table 2-22 UART1 Interface

Pins	Definitions	Descriptions
7	CTS	Clear To Send
9	RTS	Request To Send

Pins	Definitions	Descriptions
13	RXD_MUX	Receive data
11	TXD_MUX	Transmit data

2.3.23 UART3 Interface (J10)

Table 2-23 UART3 Interface

Pins	Definitions	Descriptions
33	CTS	Clear To Send
35	RTS	Request To Send
36	RXD_MUX	Receive data
38	TXD_MUX	Transmit data

2.3.24 UART4 Interface (J11)

Table 2-24 UART4 Interface

Pins	Definitions	Descriptions
28	RXD_MUX	Receive data
30	TXD_MUX	Transmit data

2.3.25 UART5 Interface (J11)

Table 2-25 UART5 Interface

Pins	Definitions	Descriptions
32	RXD_MUX	Receive data
34	TXD_MUX	Transmit data

2.3.26 USDHC1 (Ultra Secured Digital Host Controller) Interface (J10)

Table 2-26 USDHC1 Interface

Pins	Definitions	Descriptions
39	CD	Card detect
3	CLK	Card clock

Pins	Definitions	Descriptions
1	CMD	Command Signal
5	DAT0	Card data 0
7	DAT1	Card data 1
9	DAT2	Card data 2
11	DAT3	Card data 3

2.3.27 ESAI (Enhanced Serial Audio Interface) (J10 & J11)

Table 2-27 ESAI Interface

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

Chapter 3 Preparations

Before you start to use MarS Board, please read the following sections to get yourself familiar with the system images, driver code and tools which might be involved during development process.

3.1 Software Introduction

The table shown below lists the versions of Linux and Android systems that will be used later, as well as the device drivers.

Table 3-1 OS and Drivers

Types		Notes
OS	Linux	Version 3.0.35
	Android	Version 4.3
Device Drivers	Serial	Series driver
	RTC	Hardware clock driver
	Net	10/100/Gb IEEE1588 Ethernet
	Flash	Spi flash driver
	Display	Three display ports (RGB, LVDS, and HDMI 1.4a)
	mmc/sd	One SD 3.0/SDXC card slot & eMMC
	USB	3 High speed USB ports (2xHost, 1xOTG)
	Audio	Digital (HDMI) audio
	LED	User leds driver

3.2 Learning about Linux System

Please download the latest Linux image from http://www.embest-tech.com/shop/product/mars-board.html, the following tables list the specific images and storage patitions required to build a Linux system.

Table 3-2 Images Required by Linux

Images	Paths
u-boot image	u-boot.bin
kernel image	uImage

Ubuntu system image	oneiric.tgz

Table 3-3 Storage Partitions for Linux

Partition type/index	Name	Start Offset	Size	File System	Content
N/A	BOOT Loader	0	1MB	N/A	bootloader
N/A	Kernel	1M	9MB	N/A	uImage
Primary 1	Rootfs	10M	Total - Other	EXT3	oneiric.tgz

- **Partition type/index:** defined in MBR.
- **Name:** only meaningful in Android. You can ignore it when creating these partitions.
- **Start Offset:** shows where partition starts with unit in MB.

3.3 Learning about Android System

Download the latest Android image from http://www.embest-tech.com/shop/product/mars-board.html, the following tables list the specific images and storage patitions required to build an Android system.

Table 3-4 Images Required by Android

Images	Paths
u-boot image	u-boot.bin
boot image	boot.img
Android system root image	system.img
Recovery root image	recovery.img

Table 3-5 Storage Partitions for Android

Partition type/index	Name	Start Offset	Size	File System	Content
N/A	BOOT Loader	0	1MB	N/A	bootloader
Primary 1	Boot	8M	8MB	boot.img format, a kernel + ramdisk	boot.img
Primary 2	Recovery	Follow Boot	8MB	boot.img format, a kernel + ramdisk	recovery.img
Logic 4 (Extended 3)	DATA	follow Recovery	> 1024MB	EXT4 Mount at /data	Application data storage for system

					application.
Logic 5 (Extended 3)	SYSTEM	Follow DATA	512MB	EXT4. Mount as /system	Android system files under /system/ dir
Logic 6 (Extended 3)	CACHE	follow SYSTEM	512MB	EXT4. Mount as /cache	Android cache, for image store for OTA
Logic 7(Extended 3)	Device	follow CACHE	8MB	Ext4 Mount at /device	
Logic 8 (Extended 3)	Misc	Follow DATA	8M	N/A	For recovery store bootloader message, reserve.
Primary 4	MEDIA	Follow Misc	Total - Other images	VFAT	For internal media partition, in /mnt/sdcard/ dir.

- **SYSTEM Partition:** used to store Android system image.
- **DATA Partition:** used to store applications' unpacked data, system configuration database, etc.

Under normal mode, the root file system is mounted from uramdisk. Under recovery mode, the root file system is mounted from the RECOVERY partition.

3.4 Setting up HyperTerminal

Use a Mini USB cable to connect the debug port(J9) on Mars Board and the usb host on PC, and then select **Start** > **Programs** > **Accessories** > **Communications** > **HyperTerminal** to set up a new HyperTerminal according to the parameters as show below.

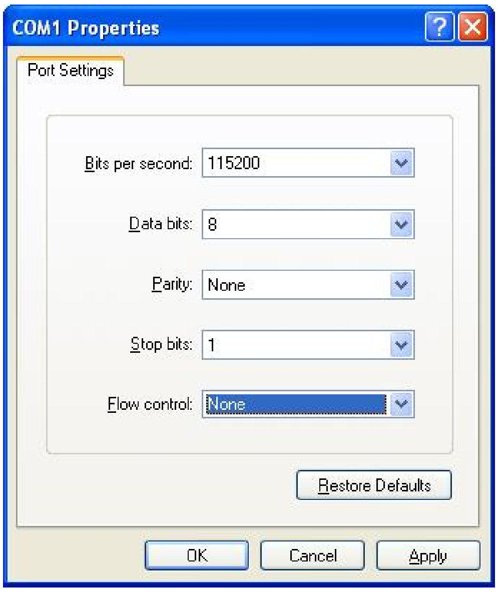


Figure 3-1 Set up HyperTerminal

Chapter 4 Downloading and Running System

Now you can download the existing system images to MarS Board and run it. The MFG tool will be used to download images.

4.1 Download and Run Linux or Android System

1） Download the MFG tool from http://www.embest-tech.com/shop/product/mars-board.html (assuming stored in C:\ tools directory).

2） Use a Mini USB cable to connect USB OTG interface（J7） on MarS Board to the USB Host on PC.

3） Set the boot switchs SW1 on MarS Board to MFG tool mode according to the configurations as shown in the following table;

Table 4-1 Boot Switch Configuration

Swtich	D1	D2
On/Off	OFF	ON

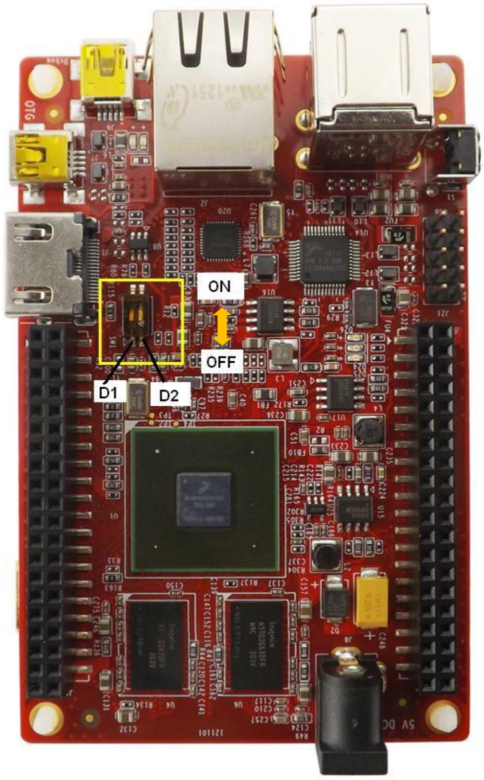


Figure 4-1 SW1 description

4） MarS Board support boot from eMMC or TF card, different mode use different image:

a) If you need to download the Linux system to the MarS board, please copy the linux image files u-boot.bin, uImag and oneiri c.tgz from the directory of “eMMC” or “TF” to C:\tools\MARSBOARD-MFGTOOLS-L4.1.15-1.0.0_ga\Profiles\Linux\OS Firmware \files\ and overwrite the files with the same names.

b) If you need to download the Android system to the MarS Board, please copy the android image files boot.img, recovery.img, system.img and u-boot.bin from the directory of “eMMC” or “TF” to C:\tools\MARSBOARD-MFGTOOLS-L4.1.15-1.0.0_ga\Profiles\Linux\OS Firmware\files\android\ to overwrite the files with the same names

Note:

📖 Default image on ship is for eMMC boot.

5） Open the configuration file [cfg.ini]() under MARSBOARD-MFGTOOLS-L4.1.15-1.0.0_ga directory with notepad application:

```
[LIST]
name = Ubuntu-Marsboard-SPI_NOR & eMMC
#name = Ubuntu-Marsboard-SPI_NOR & TF
#name = Android-Marsboard-SPI_NOR&eMMC
#name = Android-Marsboard-SPI_NOR&TF
```

Enable the corresponding item you want to program on board. For example, if you want to program Android system into eMMC:

```
[LIST]
#name = Ubuntu-Marsboard-SPI_NOR & eMMC
#name = Ubuntu-Marsboard-SPI_NOR & TF
name = Android-Marsboard-SPI_NOR&eMMC
#name = Android-Marsboard-SPI_NOR&TF
```

6） Run MFG tool under C:\tools\ on your PC and power on MarS Board; the software window is shown below; (PC will install HID driver automatically if it is the first time connecting i MX6-based product)

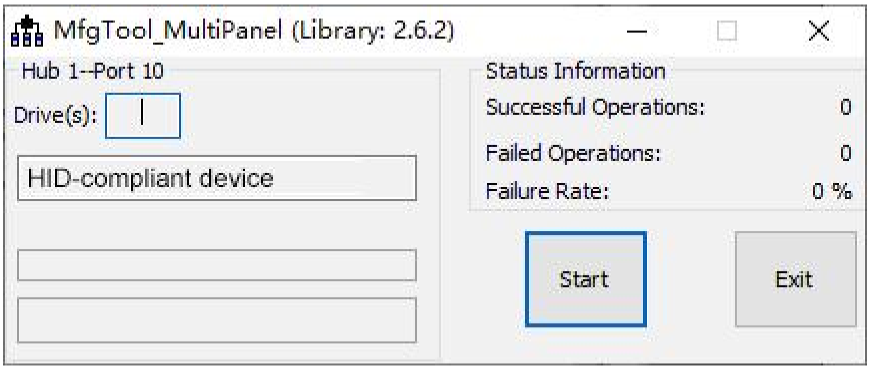


Figure 4-2 MFG tool window

7） Click **Start** in the above window; when download process is done, click **Stop** to finish.

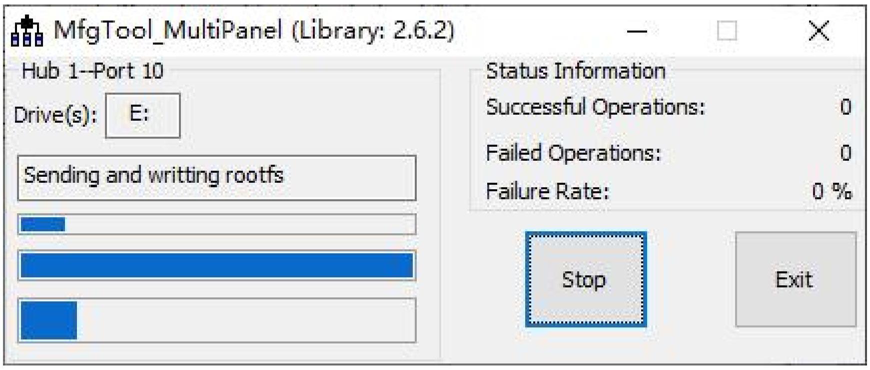


Figure 4-3 Click Start

8） Power off MarS Board and set the boot switches SW1 on it to SPI-NOR boot mode according to the configurations as shown In the following table;

Table 4-2 Boot Switch Configuration

Swtich	D1	D2
On/Off	OFF	OFF

After the switch is set, you can power on MarS Board to boot the system.

4.2 UcoS System Demonstration

UcoS system is used for demo, do not support the source code

The steps of how to download and run UcoS are as below:

1） Download the UcoS demo from http://www.embest-tech.cn/shop/product/mars-board.html, copy u-boot.bin and uImag files to C:/tools/Mfgtools-Rel-12.04.01_ER_MX6Q_UPDATER \Profiles\MX6Q Linux Update\OS Firmware\files\ to overwrite the files with the same names

2） Refer to the method of downloading linux image that described in chapter 4.1 to download UcoS system image.

3） Copy all of the files under ucos to TF card, insert the TF card into Mars Board, connect 7” LCD and HDMI TV, and then power on the board, the dual OS running concurrently. The UcoS system will display on 7” LCD, Ubuntu system will display on HDMI TV.

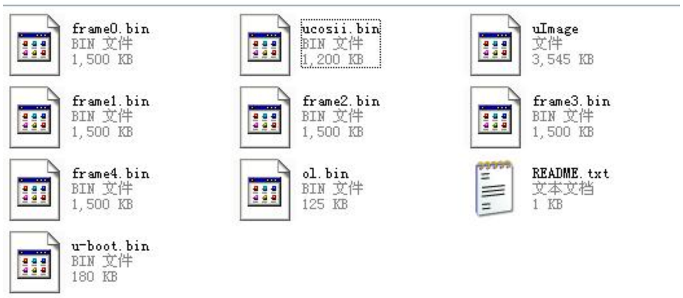


Figure 4-4 The required files for running UcoS demo

4.3 Display Mode Configurations of Linux&Android System

The system supports multiple display modes. Users can select an appropriate mode by cofiguring u-boot parameters.Please reboot the kit and press any key on your PC’s keyboard when the system prompts you with a countdown in seconds as shown below:

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

CPU: Freescale i.MX6 family TO0.0 at 792 MHz
Temperature:     51 C, calibration data 0x58150469
mx6q pll1: 792MHz
mx6q pll2: 528MHz
mx6q pll3: 480MHz
mx6q pll8: 50MHz
ipg clock       : 66000000Hz
ipg per clock : 66000000Hz
uart clock      : 80000000Hz
cspi clock      : 60000000Hz
ahb clock        : 132000000Hz
axi clock     : 264000000Hz
emi_slow clock: 29333333Hz
ddr clock        : 528000000Hz
usdhc1 clock   : 198000000Hz
usdhc2 clock   : 198000000Hz
usdhc3 clock   : 198000000Hz
usdhc4 clock   : 198000000Hz
nfc clock       : 24000000Hz
Board: MX6Q-MARSBOARD:[ POR]
Boot Device: I2C
I2C:     ready
DRAM:     1 GB
MMC:     FSL_USDHC: 0,FSL_USDHC: 1
JEDEC ID: 0xbf:0x25:0x41
Reading SPI NOR flash 0xc0000 [0x2000 bytes] -> ram 0x276009b8
SUCCESS

*** Warning - bad CRC, using default environment

In:      serial
Out:     serial
Err:     serial
Net:     got MAC address from IIM: 00:00:00:00:00:00
----enet_board_init: phy reset
FEC0 [PRIME]
Hit any key to stop autoboot:   0 ( press any key to enter u-boot command mode）
MX6Q MARSBOARD U-Boot >
```

1） Display with 4.3” LCD

Execute the following instructions in u-boot mode to configure for 4.3-inch display mode;

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

2） Display with 7” LCD

Execute the following instructions in u-boot mode to configure for 7-inch display mode;

- MX6Q MARSBOARD U-Boot > **setenv bootargs console=ttymxc1,115200 init=/init rw video=mxcfb0:dev=lcd,1024x768M@60 ,if=RGB24 fbmem=10M vmalloc=400M androidboot.console=ttymxc1 calibration androidboot.hardware=freescale**

3） Display with 9.7” LVDS

Execute the following instructions in u-boot mode to configure for 9.7-inch display mode;

- **MX6Q MARSBOARD U-Boot > setenv bootargs console=ttymxc1,115200 init=/init rw video=mxcfb0:dev=ldb,LDB-XGA,if=RGB666 fbmem=10M vmalloc=400M androidboot.console=ttymxc1 androidboot.hardware=freescale**

4） Display with VGA8000

Execute the following instructions in u-boot mode to configure for VGA8000 display mode;

- **MX6Q MARSBOARD U-Boot > setenv bootargs console=ttymxc1,115200 init=/init rw video=mxcfb0:dev=lcd,1024x768M@60,if=RGB24 fbmem=10M vmalloc=400M androidboot.console=ttymxc1 androidboot.hardware=freescale**

Note:

Usually, the default display resolution is 1024x768, yet users may set many other resolutions such as 800x600, 1440x900 and 1280×1024 that suited for their VGA displays.

To change VGA output resolution (assuming from 1024x768 to 800x600): Replace “video=mxcfb0:dev=lcd,1024x768M@60” with “video=mxcfb0:dev=lcd,800x600M@60”, in the instructions above and keep the rest unchanged.

5） Display with HDMI

Execute the following instructions in u-boot mode to configure for HDMI display mode;

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

Note:

The u-boot parameter is stored in SPI-NOR flash, if you want to revert to the default, please use the below command:

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

Chapter 5 Making Images

This Chapter will introduce how to make images by using the BSP. The BSP is a collection of binary, source code, and support files that can be used to create a u-boot bootloader, Linux kernel image, and Android file system for i.MX 6Dual Mars Board.

Note:

- The following instructions are all executed under Ubuntu system.
- Each instruction has been put a bullets "•" before it to prevent confusion caused by the long instructions that occupy more than one line in the context.

5.1 Making Images for Linux

Please strictly follow the steps listed below to make images for Linux system.

5.1.1 Getting Tools and Source Code

1） Execute the following instructions to get cross compiling toolchain;

- **$ cd ~**
- **$ git clone git://github.com/embest-tech/fsl-linaro-toolchain.git**

2） Execute the following instructions to get u-boot source code;

- **$ cd ~**
- **$ git clone git://github.com/embest-tech/u-boot-imx.git -b embest_imx_3.0.15_12.04.01**

3） Execute the following instructions to get kernel source code;

- **$ cd ~**
- **$ git clone git://github.com/embest-tech/linux-imx.git -b embest_imx_3.0.35_4.0.0**

5.1.2 Compiling System Images

1） Execute the following instructions to compile u-boot image;

a) For eMMC Boot:

- **$ cd ~ /uboot-imx**
- **$ export ARCH=arm**
- **$ export CROSS_COMPILE=~/fsl-linaro-toolchain/bin/arm-fsl-linux-gnueabi-**
- **$ make distclean**
- **$ make mx6q_marsboard_emmc_config**
- **$ make**

b) For TF Boot:

- **$ cd ~ /uboot-imx**
- **$ export ARCH=arm**
- **$export CROSS_COMPILE=~/fsl-linaro-toolchain/bin/arm-fsl-linux-gnueabi-**
- **$ make distclean**
- **$ make mx6q_marsboard_tf_config**
- **$ make**

After executing the instructions, a file u-boot.bin can be found in the current directory ;

2） Execute the following instructions to compile kernel image;

- **$export PATH=~/uboot-imx/tools:$PATH**
- **$ cd ~/linux-imx**
- **$ export ARCH=arm**
- **$export CROSS_COMPILE=~/ fsl-linaro-toolchain/bin/arm-fsl-linux-gnueabi-**
- **$ make imx6_defconfig**
- **$ make uImage**

After executing the instructions, a kernel image named uImage can be found under arch/arm/boot/.

Note:

🕮 The mkimage used to build kernel and ramfs images is automatically generated and saved under tools/ after compiling u-boot.bin, so please make sure uboot is compiled first before compiling kernel image.

Copy u-boot.bin and uImag files that are generated by compiling to C:\tools\MARSBOARD-MFGTOOLS-L4.1.15-1.0.0_ga\Profiles\Linux\OS Firmware\files\ to overwrite the files with the same names and then start over the operatios from step 2） in Chapter 4.1 so as to verify the Linux system you built.

5.2 Making Images for Android

Please strictly follow the steps listed below to make images for Android system.

5.2.1 Establishment of development environment

Before the android development, users have to establish an android cross development environment on PC. See http://source.android.com/source/initializing.html for detail

5.2.2 Getting Repo Source Code

1） Execute he following instructions to get repo source code;

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

2） Execute he following instructions to initialize repo source code;

- **$ mkdir ~/android-imx6-jb4.3-1.0.0**
- **$ cd ~/android-imx6-jb4.3-1.0.0**
- **$ repo init --repo-url=git://github.com/android/tools_repo.git -u git://github.com/embest-tech/imx-manifest.git -m embest_android_jb4.3_1.0.0**

3） Execute the following instructions to synchronize repo source code;

- **$ cd ~/android-imx6-jb4.3-1.0.0**
- **$ repo sync**

5.2.3 Compiling System Images

1） Boot mode select

MarS Board support boot from eMMC or TF card, and the startup mode can be selected by modifying the value of “TARGET_LOCATION” under “android-imx6-jb4.3-1.0.0/device/fsl/marsboard_6q/BoardConfig.mk”, the details as below:

eMMC Boot -- BUILD_TARGET_LOCATION ?= emmc

TF Boot -- BUILD_TARGET_LOCATION ?= sdmmc

2） Execute the following instructions to compile Android image;

- **$ cd ~/android-imx6-jb4.3-1.0.0**
- **$ source build/envsetup.sh**
- **$ lunch marsboard_6q-user**
- **$ make clean**
- **$ make**

After executing the instructions, the generated images can be found under android-imx6-jb4.3-1.0.0/out/target/product/marsboard_6q/; the table shown below lists all the images and directories after compilation is completed.

Table 5-1 Images and Directories

Images/Directories	Notes
root/	root file system, mounted at /
system/	Android system directory, mounted at /system
data/	Android data area. mounted at /data
recovery/	Root filesystem when booting in "recovery" mode, not used directly
boot.img	A composite image which includes the kernel zImage, ramdisk, and boot parameters
ramdisk.img	Ramdisk image generated from "root/", not directly used
system.img	EXT4 image generated from "system/". Can be written to "SYSTEM" partition of SD/eMMC card with "dd" command
userdata.img	EXT4 image generated from "data/"
recovery.img	EXT4 image generated from "recovery/". Can be

Images/Directories	Notes
	written to "RECOVERY" partition of SD/eMMC card with "dd" command
u-boot.bin	uboot image with padding

Note:

- Android image should be built in user mode; please visit http://source.android.com/source/building.html to learn more information.

3） Execute the following instructions to compile boot.img;

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

After executing the instructions, a boot.img image can be found under android-imx6-jb4.3-1.0.0/out/target/product/marsboard_6q/.

Note：

- The mkimage used to build kernel and ramfs images is automatically generated and saved under tools/ after compiling u-boot.bin, so please make sure uboot is compiled first before compiling kernel image.
- Copy boot.img, recovery.img, system.img and u-boot.bin files that are generated by compiling to C:\tools/MARSBOARD-MFGTOOLS-L4.1.15-1.0.0_ga\Profiles\Linux\OS Firmware\files\android\ to overwrite the files with the same names and then start over the operatios from step 2) in Chapter 4.1 so as to verify the Android system you built.

Appendix 1 – Installing Ubuntu Linux System

As we all know, an appropriate development environment is required for software development. The software development environment needs to be installed under Linux system. If you are working on a PC running Windows, you have to create a Linux system first, and then you can install the environment. Here we recommend using VirtualBox – a virtual machine software to accommodate Ubuntu Linux system under Windows. The following sections will introduce the installation processes of VirtualBox and Ubuntu system.

Installing VirtualBox

You can access http://www.virtualbox.org/wiki/Downloads to download the latest version of VirtualBox. VirtualBox requires 512MB memory space at least. A PC with memory space of more than 1GB would be preferred.

1） The installation process is simple and will not be introduced. Please start VirtualBox from the **Start** menu of Windows, and then click **New** in VirtualBox window. A pop-up window **Create New Virtual Machine** will be shown as below;

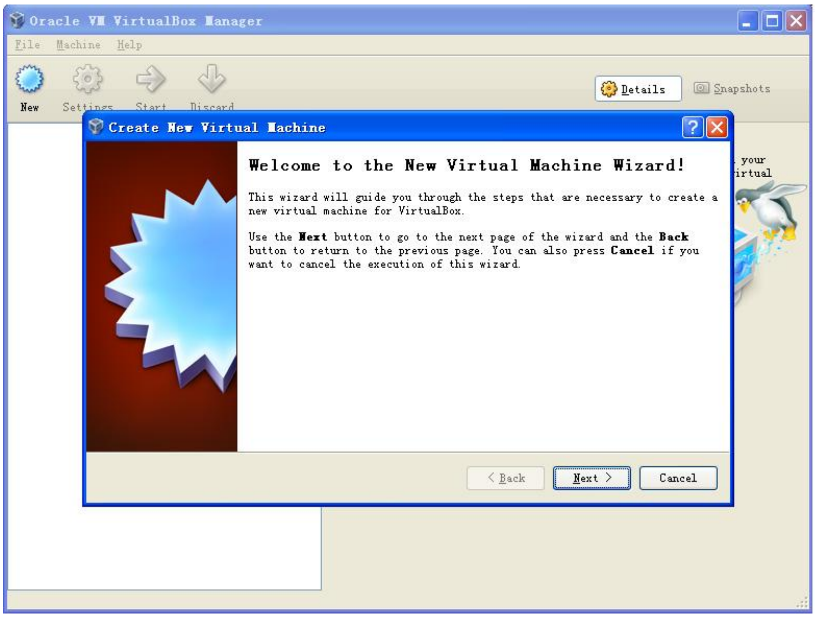


Figure 1 Create new virtual machine

Click **Next** to create a new virtual machine.

2） Enter a name for the new virtual machine and select operating system type as shown below;

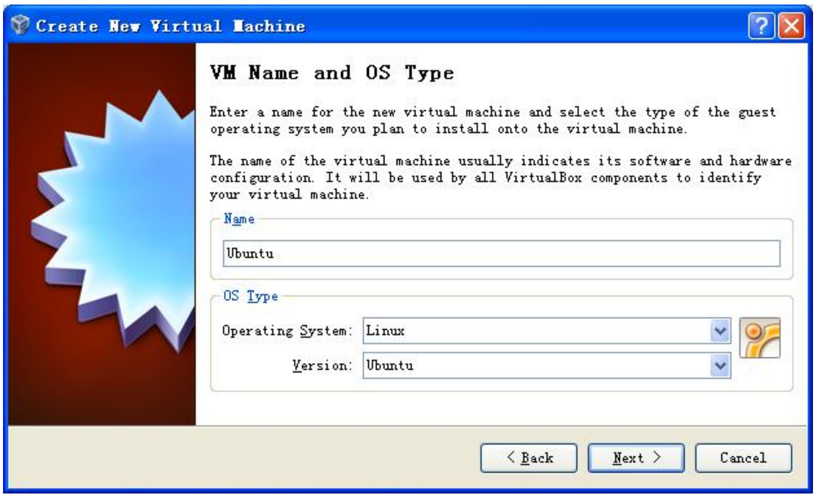


Figure 2 Name and OS type of virtual machine

Enter a name in the **Name** field, e.g. Ubuntu, and select **Linux** in the **Operating System** drop-down menu, and then click **Next**.

3) Allocate memory to virtual machine and then click **Next**;

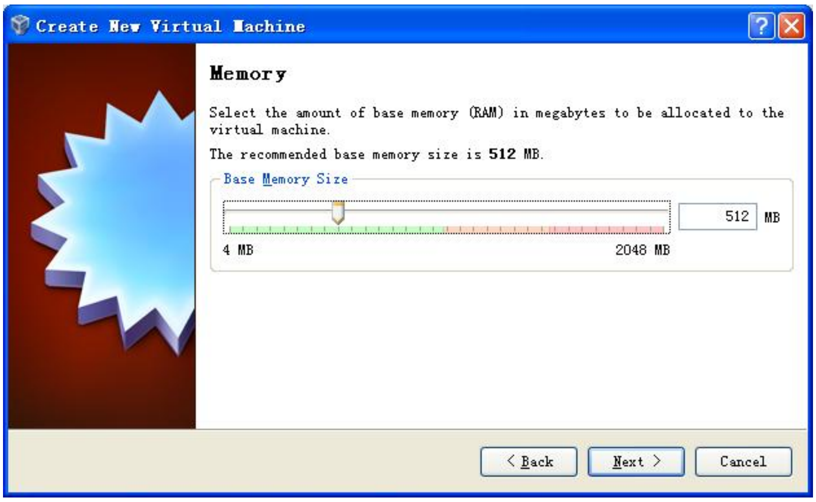


Figure 3 Memory allocation

Note:

- If the memory of your PC is only 1GB or lower, please keep the default setting;
- If the memory of your PC is higher than 1GB, you can allocate 1/4 or fewer to virtual machine, for example, 512MB out of 2GB memory could be allocated to virtual machine.

4) If this is the fist time you install VirtualBox, please select **Create new hard disk** in the following window, and then click **Next**;

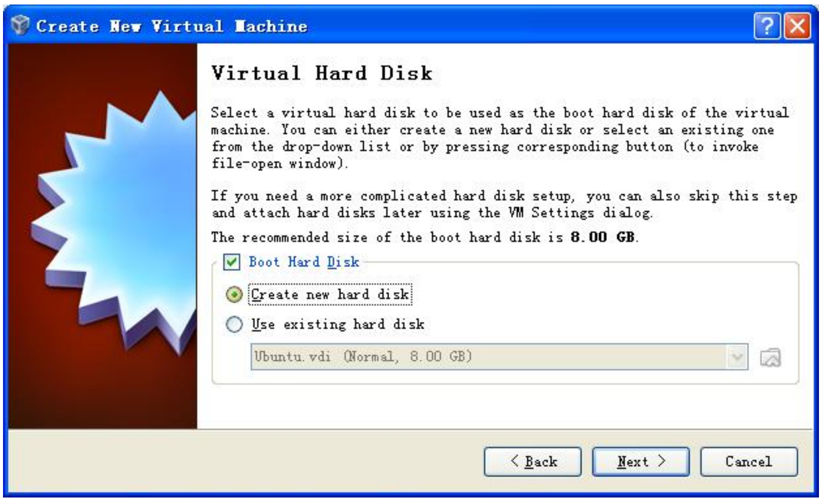


Figure 4 Create new hard disk

5） Click **Next** in the following window;

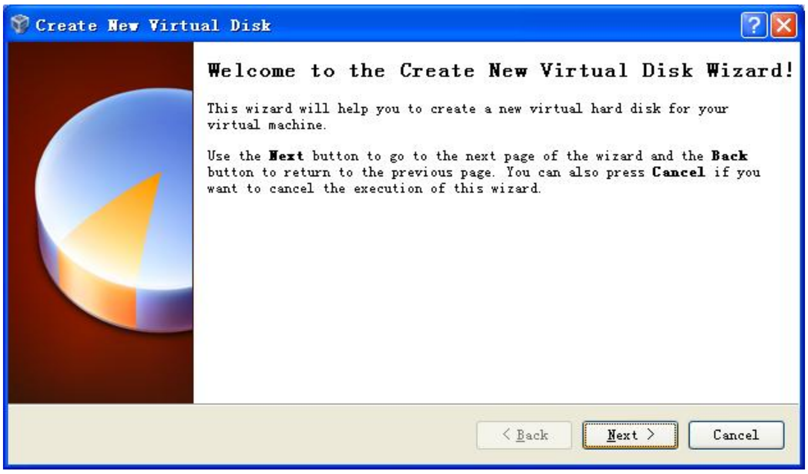


Figure 5 Wizard of new virtual disk creation

6） Selecting **Fixed-size storage** in the following window and click **Next**;

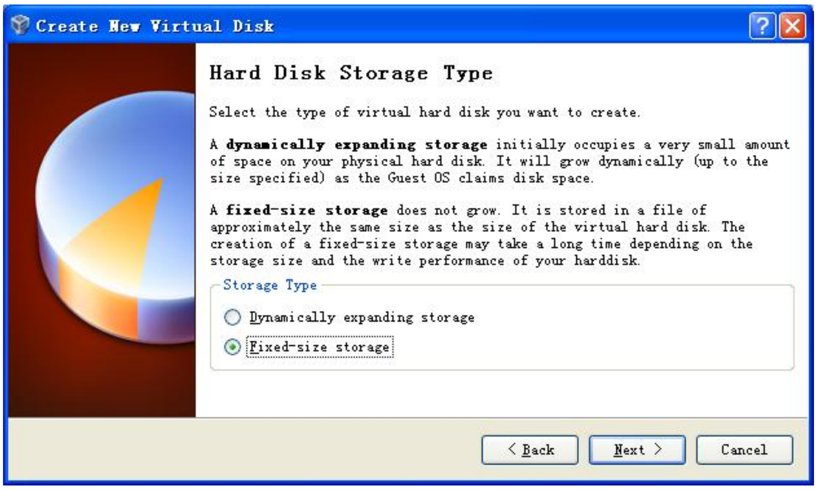


Figure 6 Select the second option

7） Define where the hard disk data is stored and the default space of the virtual disk (8G at least), and then click **Next**;

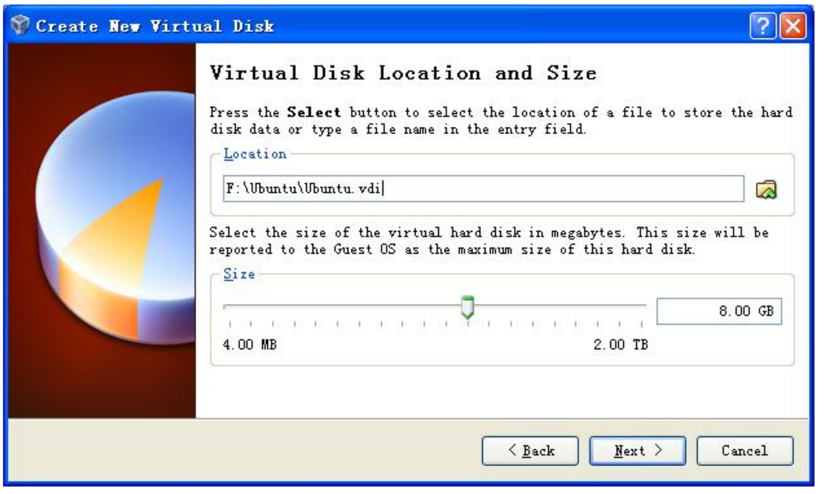


Figure 7 Virtual disk configuration

8） Click **Finish** in the following window;

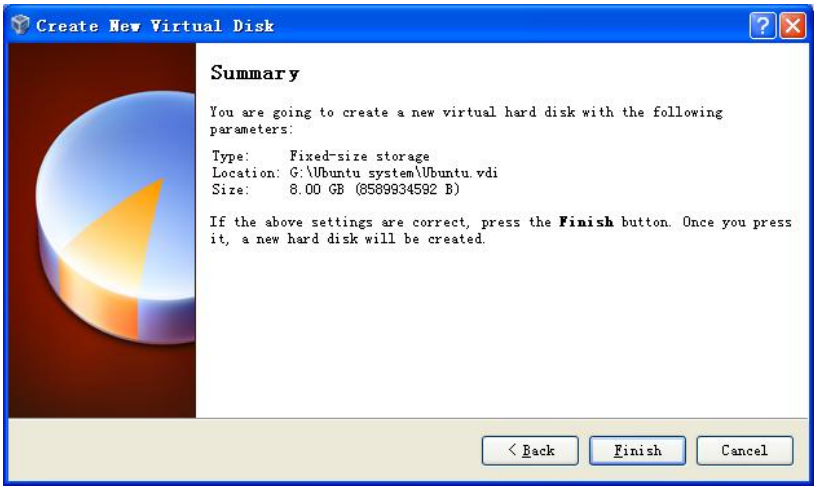


Figure 8 Virtual disk summary

9） PC is creating a new virtual disk;

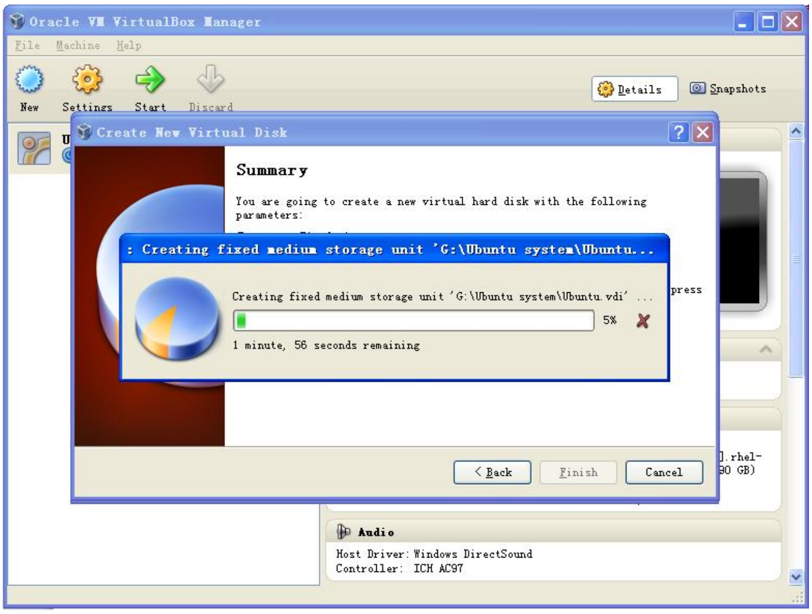


Figure 9 Virtual disk creation in process

10） A window with summary of the newly created virtual machine will be shown as below when the creation process is done. Please click **Finish** to complete the whole process.

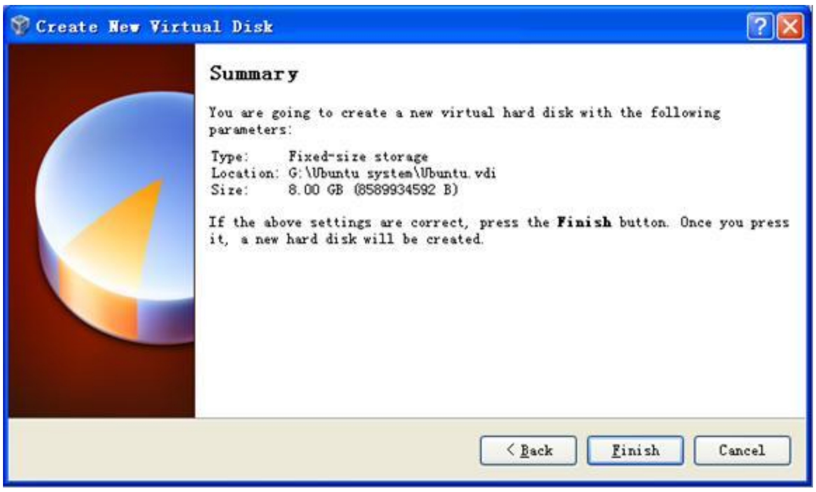


Figure 10 Virtual machine is ready

Installing Ubuntu Linux System

After virtualBox is installed, we can start the installation of Ubuntu Linux system now. Please access http://www.Ubuntu.com/download/Ubuntu/download to download the ISO image file of Ubuntu, and then follow the steps。

1） Start VirtualBox from the **Start** menu and click **Setting** on the VirtualBox window. A **Settings** window will be shown as below;

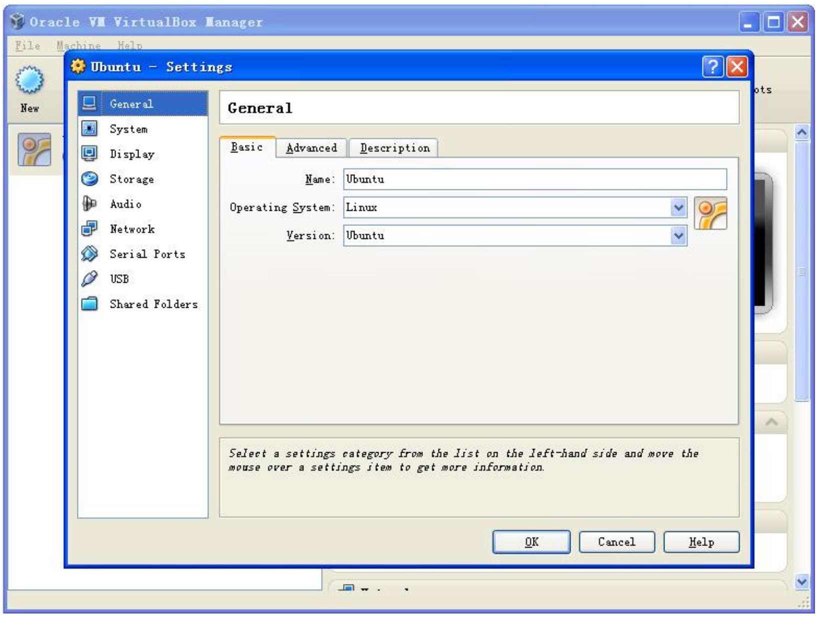


Figure 11 Setting window

2） Select **Storage** on the left in the **Settin**g window and click the CD-like icon next to the option **Empty** under IDC controller in the right part of the window, and then find the ISO file you downloaded;

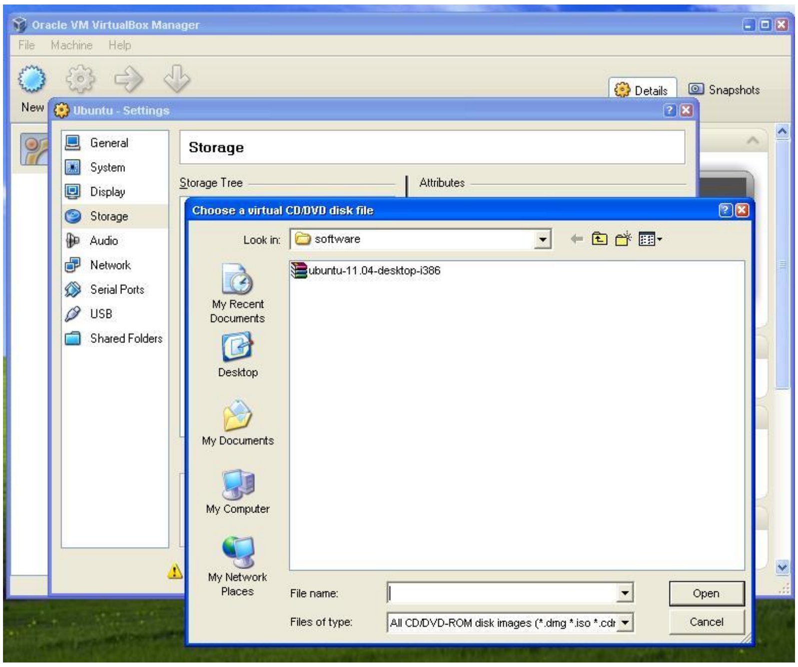


Figure 12 Find ISO file

3） Select the ISO file you added in and click **OK** as shown below;

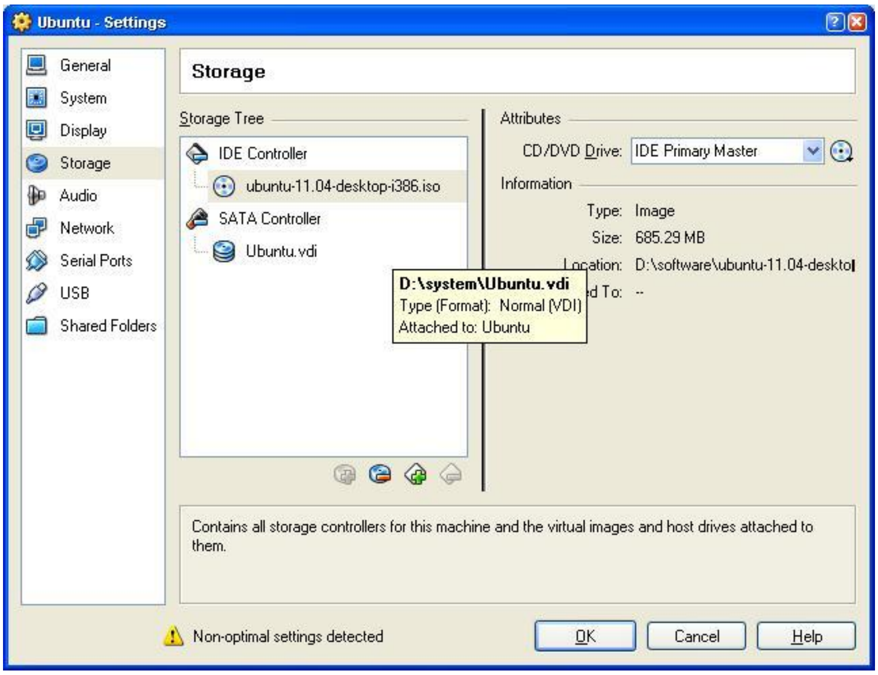


Figure 13 Select ISO file

4） Click **Start** on the VirtualBox window, the installation program of Ubuntu will be initiating as shown below;

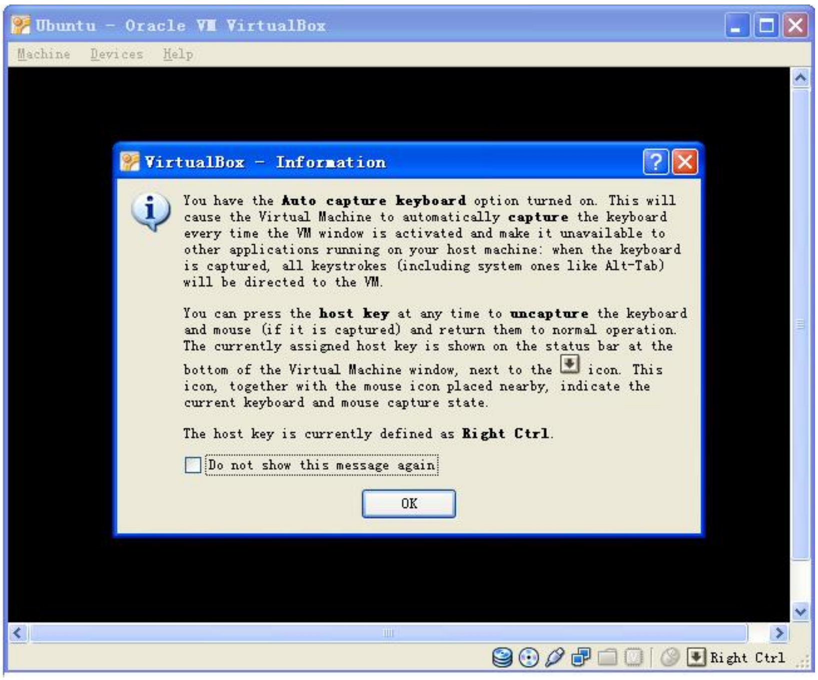


Figure 14 Ubuntu initiating window

Some prompt windows will interrupt in during the initiating process. You just need to click **OK** all the way to the end of the process.

5） Click **Install Ubuntu** to start installation when the following window appears;

Figure 15 Ubuntu installation window

6） Click **Forward** to continue the process;

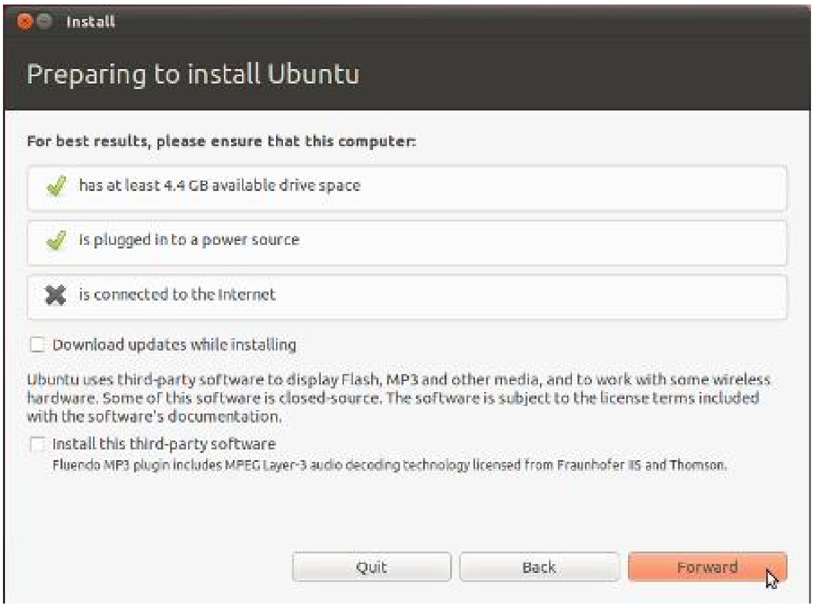


Figure 16 Information before installation

7） Select Erase disk and install Ubuntu and click Forward;

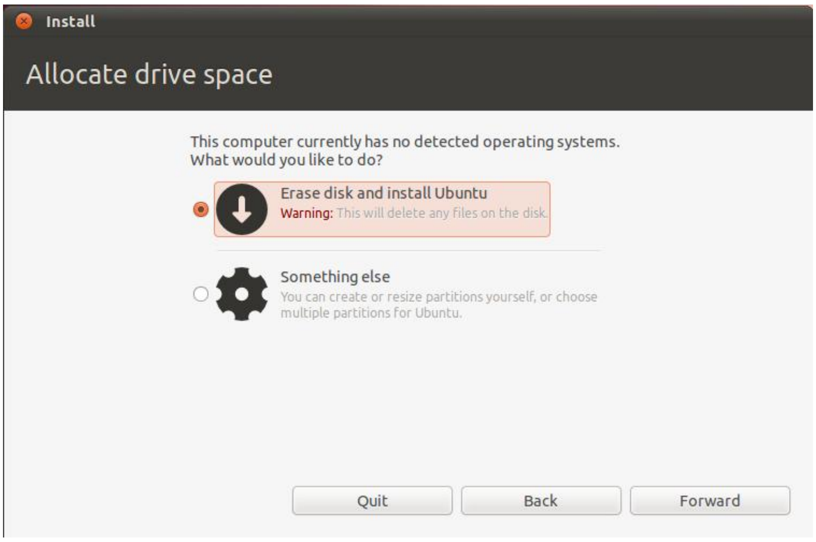


Figure 17 Options before installation

Note:

📖 Selecting this option will not lead to any content loss on your hard drive.

8） Click **Install Now** in the following window to start installation;

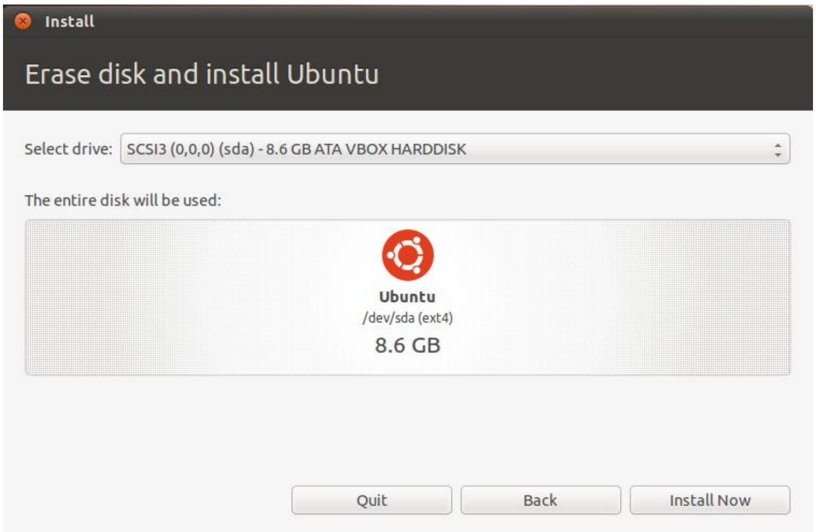


Figure 18 Confirm installation

9） Some simple questions need to be answered during the installation process. Please enter appropriate information and click **Forward**. The following window is the last question that will appear during the process;

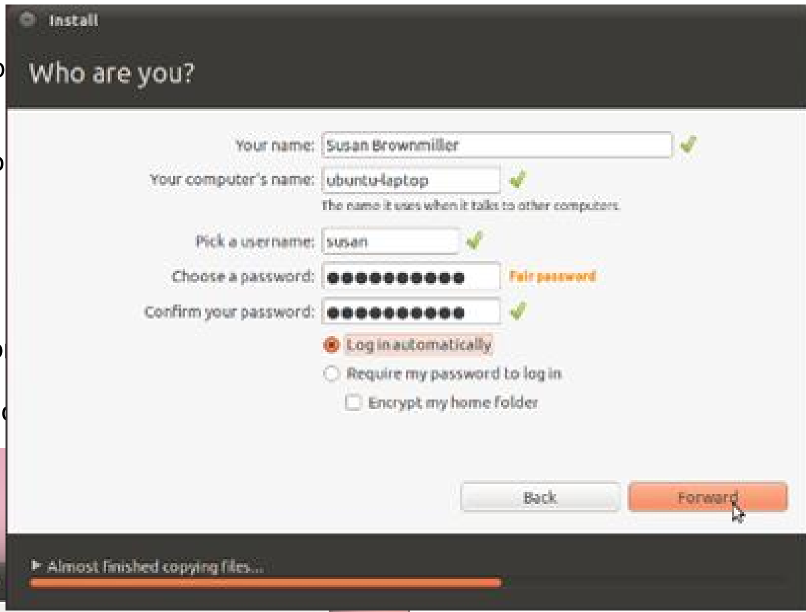


Figure 19 Enter ap

After all the required informatio

Log in automatically and click

10） The installation of Ubuntu may

on your PC's performance. A p

installation is done. Please sele

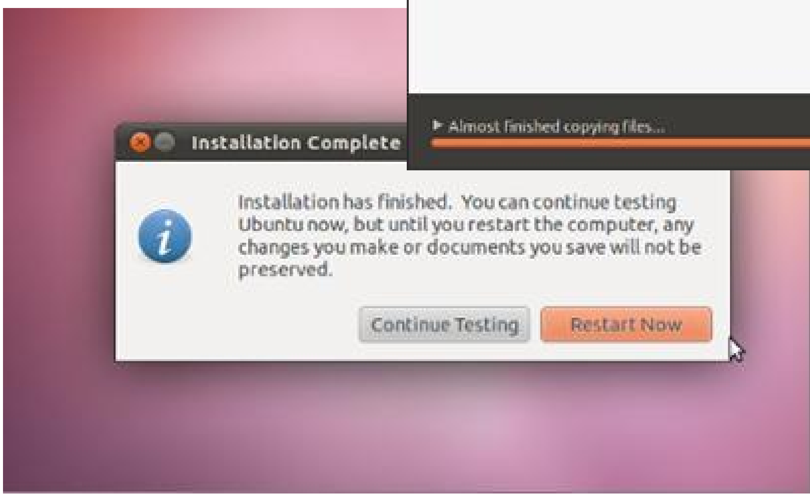


Figure 20 Restart Ubuntu

11) Ubuntu system is ready for use after restarting. Normally the ISO file shown in Figure 13 will be ejected automatically by VirtualBox after restarting Ubuntu. If it doesn't, you could eject the ISO file manually in the **Setting** window of VirtualBox. The following window shows how it looks after the ISO file is ejected.

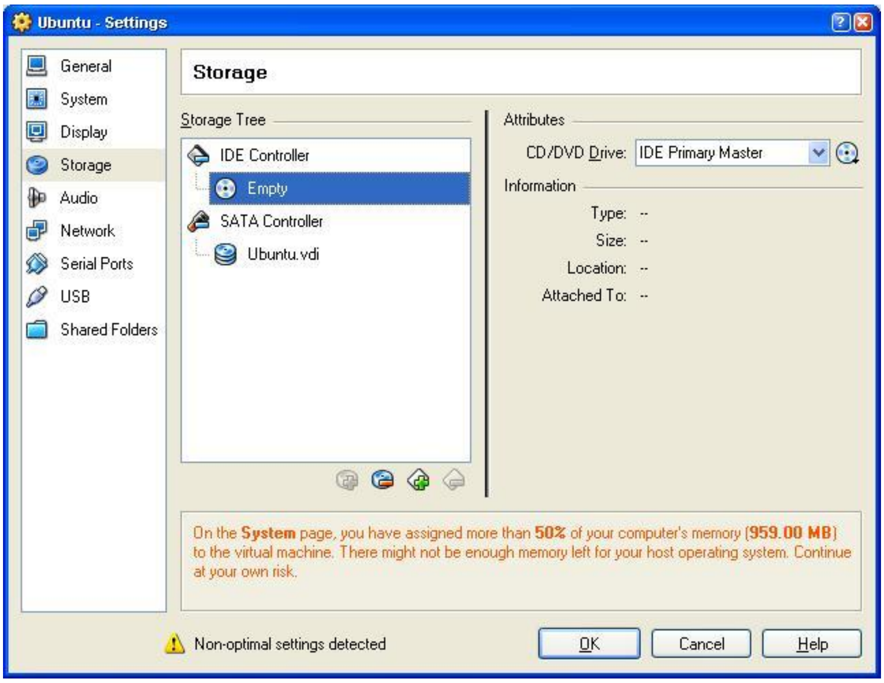


Figure 21 ISO file ejected

Technical Support and Warranty

Technical Support

Embest Technology provides its product with one-year free technical support including:

- Providing software and hardware resources related to the embedded products of Embest Technology;
- Helping customers properly compile and run the source code provided by Embest Technology;
- Providing technical support service if the embedded hardware products do not function properly under the circumstances that customers operate according to the instructions in the documents provided by Embest Technology;
- Helping customers troubleshoot the products.

The following conditions will not be covered by our technical support service. We will take appropriate measures accordingly:

- Customers encounter issues related to software or hardware during their development process;
- Customers encounter issues caused by any unauthorized alter to the embedded operating system;
- Customers encounter issues related to their own applications;
- Customers encounter issues caused by any unauthorized alter to the source code provided by Embest Technology;

Warranty Conditions

1） 12-month free warranty on the PCB under normal conditions of use since the sales of the product;

2） The following conditions are not covered by free services; Embest Technology will charge accordingly:

A. Customers fail to provide valid purchase vouchers or the product identification tag is damaged, unreadable, altered or inconsistent with the products.

B. Products are damaged caused by operations inconsistent with the user manual;

C. Products are damaged in appearance or function caused by natural disasters (flood, fire, earthquake, lightning strike or typhoon) or natural aging of components or other force majeure;

D. Products are damaged in appearance or function caused by power failure, external forces, water, animals or foreign materials;

E. Products malfunction caused by disassembly or alter of components by customers or, products disassembled or repaired by persons or organizations unauthorized by Embest Technology, or altered in factory specifications, or configured or expanded with the components that are not provided or recognized by Embest Technology and the resulted damage in appearance or function;

F. Product failures caused by the software or system installed by customers or inappropriate settings of software or computer viruses;

G. Products purchased from unauthorized sales;

H. Warranty (including verbal and written) that is not made by Embest Technology and not included in the scope of our warranty should be fulfilled by the party who committed. Embest Technology has no any responsibility;

3） Within the period of warranty, the freight for sending products from customers to Embest Technology should be paid by customers; the freight from Embest to customers should be paid by us. The freight in any direction occurs after warranty period should be paid by customers.

4） Please contact technical support if there is any repair request.

Note:

📖 Embest Technology will not take any responsibility on the products sent back without the permission of the company.

Contact Information

Technical Support

Tel: +86-755-25635626-872/875/897

Email: support@embest-tech.com

Sales Information

Tel: +86-755-25635626-863/865/866/867/868

Fax: +86-755-25616057

Email: globalsales@embest-tech.com

Company Information

Website: http://www.embest-tech.com

Address: Tower B 4/F, Shanshui Building, Nanshan Yungu Innovation Industry Park, Liuxian Ave. No. 1183, Nanshan District, Shenzhen, Guangdong, China (518055)