ISR101

Ruggedized Embedded System

with NXP ARM® Cortex®
A9 i.MX6 Dual-Lite SoC

User's Manual

Version 1.0
(Aug. 2018)

iBASE

Copyright

Disclaimer

IBASE reserves the right to make changes and improvements to the products described in this document without prior notice. Every effort has been made to ensure the information in the document is correct; however, IBASE does not guarantee this document is error-free. IBASE assumes no liability for incidental or consequential damages arising from misuse or inability to use the product or the information contained herein, and for any infringements of rights of third parties, which may result from its use.

Trademarks

All the trademarks, registrations and brands mentioned herein are used for identification purposes only and may be trademarks and/or registered trademarks of their respective owners.

Compliance

CE

The product described in this manual complies with all applicable European Union (CE) directives if it has a CE marking. For systems to remain CE compliant, only CE-compliant parts may be used. Maintaining CE compliance also requires proper cable and cabling techniques.

FC

This product has been tested and found to comply with the limits for a Class B 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 equipment generates, uses and can radiate radio frequency energy and, if not installed and used in accordance with manufacturer's instructions, may cause harmful interference to radio communications.

WEEE

This product must not be disposed of as normal household waste, in accordance with the EU directive of for waste electrical and electronic equipment (WEEE - 2012/19/EU). Instead, it should be disposed of by returning it to a municipal recycling collection point. Check local regulations for disposal of electronic products.

Green IBASE

This product complies with the current RoHS directives restricting the use of the following substances in concentrations not to exceed 0.1% by weight (1000 ppm) except for cadmium, limited to 0.01% by weight (100 ppm).

- Lead (Pb)
- Mercury (Hg)
- Cadmium (Cd)
- Hexavalent chromium (Cr6+)
- Polybrominated biphenyls (PBB)
- Polybrominated diphenyl ether (PBDE)

iBASE

Important Safety Information

Carefully read the following safety information before using this device.

Setting up your system:

- Put the device horizontally on a stable and solid surface.
- Do not use this product near water or any heated source.
- Leave plenty of space around the device and do not block the ventilation openings. Never drop or insert any objects of any kind into the openings.
- Use this product in environments with ambient temperatures between 0˚C and 60˚C.

Care during use:

- Do not place heavy objects on the top of the device.
- Make sure to connect the correct voltage to the device. Failure to supply the correct voltage could damage the unit.
- Do not walk on the power cord or allow anything to rest on it.
- If you use an extension cord, make sure the total ampere rating of all devices plugged into the extension cord does not cord's ampere rating.
- Do not spill water or any other liquids on your device.
- Always unplug the power cord from the wall outlet before cleaning the device.
- Only use neutral cleaning agents to clean the device.
- Vacuum dust and particles from the vents by using a computer vacuum cleaner.

Product Disassembly

Do not try to repair, disassemble, or make modifications to the device. Doing so will void the warranty and may result in damage to the product or personal injury.

CAUTION

Replace only with the same or equivalent type recommended by the manufacturer. Dispose of used batteries by observing local regulations.

Warranty Policy

- **IBASE standard products:**

  24-month (2-year) warranty from the date of shipment. If the date of shipment cannot be ascertained, the product serial numbers can be used to determine the approximate shipping date.

- **3rd-party parts:**

  12-month (1-year) warranty from delivery for 3rd-party parts that are not manufactured by IBASE, such as CPU, CPU cooler, memory, storage devices, power adaptor, display panel and touch screen.

* PRODUCTS, HOWEVER, THAT FAIL DUE TO MISUSE, ACCIDENT, IMPROPER INSTALLATION OR UNAUTHORIZED REPAIR SHALL BE TREATED AS OUT OF WARRANTY AND CUSTOMERS SHALL BE BILLED FOR REPAIR AND SHIPPING CHARGES.

Technical Support & Services

1. Visit the IBASE website at www.ibase.com.tw to find the latest information about the product.
2. If you encounter any technical problems and require assistance from your distributor or sales representative, please prepare and send the following information:
   - Product model name
   - Product serial number
   - Detailed description of the problem
   - Error messages in text or screenshots if any
   - The arrangement of the peripherals
   - Software used (such as OS and application software)
3. If repair service is required, please download the RMA form at http://www.ibase.com.tw/english/Supports/RMAService/.
   Fill out the form and contact your distributor or sales representative.

iBASE

Table of Contents

Chapter 1
General Information

The information provided in this chapter includes:

- Features
- Packing List
- Optional Accessories
- Specifications
- Overview
- Dimensions

1.1 Introduction

ISR101 is an ARM®-based embedded system with NXP Cortex® i.MX6 A9 processor. The device offers 2D, 3D graphics and multimedia accelerations while it also features numerous peripherals that are well suited for industrial applications, including RS-232/422/485, COM, GPIO, USB, USB OTG, LAN, HDMI for a full HD display and M.2 E2230 for wireless connectivity.

1.2 Features

- NXP ARM® Cortex® A9 i.MX6 Dual-Lite 1 GHz processor
- Video output through HDMI
- 1 GB DDR3, 4 GB eMMC and MicroSD socket for expansion
- Rich I/O interface with COM, GPIO, USB, USB-OTG, audio and Ethernet
- M.2 E2230 for wireless connectivity & 2 x UART headers
- OpenGL ES 2.0 for 3D BitBlt for 2D and OPENVG1.1 hardware accelerators

1.3 Packing List

Your product package should include the items listed below. If any of the item below is missing, contact the distributor or the dealer from whom you have purchased the product. Drivers and this user manual are downloadable from our website.

- ISR101

1.4 Optional Accessories

IBASE provides optional accessories as listed below. Please contact us or your dealer if you would like to order any item(s).

- Power Adapter
- WiFi & BT Combo (M.2 card)
- USB OTG Cable (USB-81)
- VESA Mounting Kit (75 x 75 / 100 x 100)
- DIN Rail Mounting Kit

1.5 Specifications

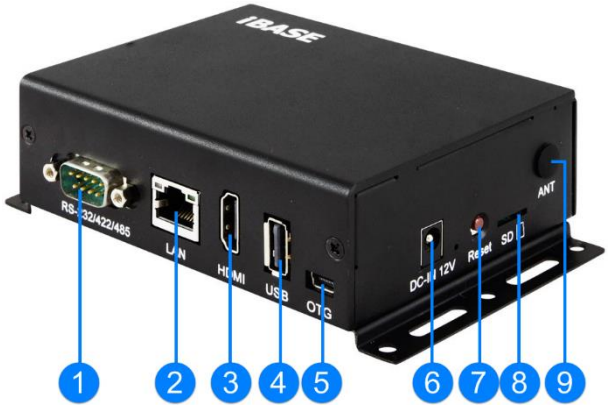
Product Name	ISR101
System Motherboard	IBR115 2.5” SBC
System	
Operating System	• Yocto 2.0 • Android 6.0
CPU Type	NXP Cortex™ A9 i.MX6 Dual-Lite
CPU Speed	1 GHz
Memory	• **System memory:** 1 GB DDR3 • **Data Memory:** 4 GB eMMC
Video Codec	• **Encoder:** 1080p, 30 fps (MPEG-4 SP, H.264 BP, H.263, MJPEG BP) • **Decoder:** 1080p, 30 fps (MPEG-4 ASP, H.264 HP, MPEG-2 MP, MJPEG BP)
Touch	USB headers for CPT
Audio	1 x 10-pin box header for Line-In & Line-Out
RTC	AnalogTEK AT8565S
Wireless	WiFi / BT module (Optional)
Power Supply	12V DC-In
Watchdog Timer	Yes (256 segments, 0, 1, 2…128 secs)
Dimensions	100 x 72 mm (4” x 2.8”)
RoHS	Yes
Certification	CE, FCC Class B
I/O Ports	
Display	1 x HDMI 1.4 (1080p at 60 Hz)
LAN	1 x RJ45 GbE LAN
USB	• 2 x USB 2.0 Type A • 1 x USB OTG via mini-USB Type B
Serial	1 x COM RS-232/422/485 port
MicroSD	1 x MicroSD socket (max.104 MB/s)
Digital IO	8 In / Out
Expansion Slots	1 x M.2 E-key (2230)

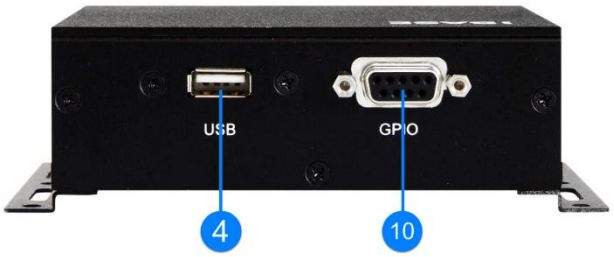
Environment	
Operating Temperature	0 ~ 60 °C (32 ~ 140 °F)
Relative Humidity	10 ~ 90 %, non-condensing

All specifications are subject to change without prior notice.

1.6 Overview

I/O View





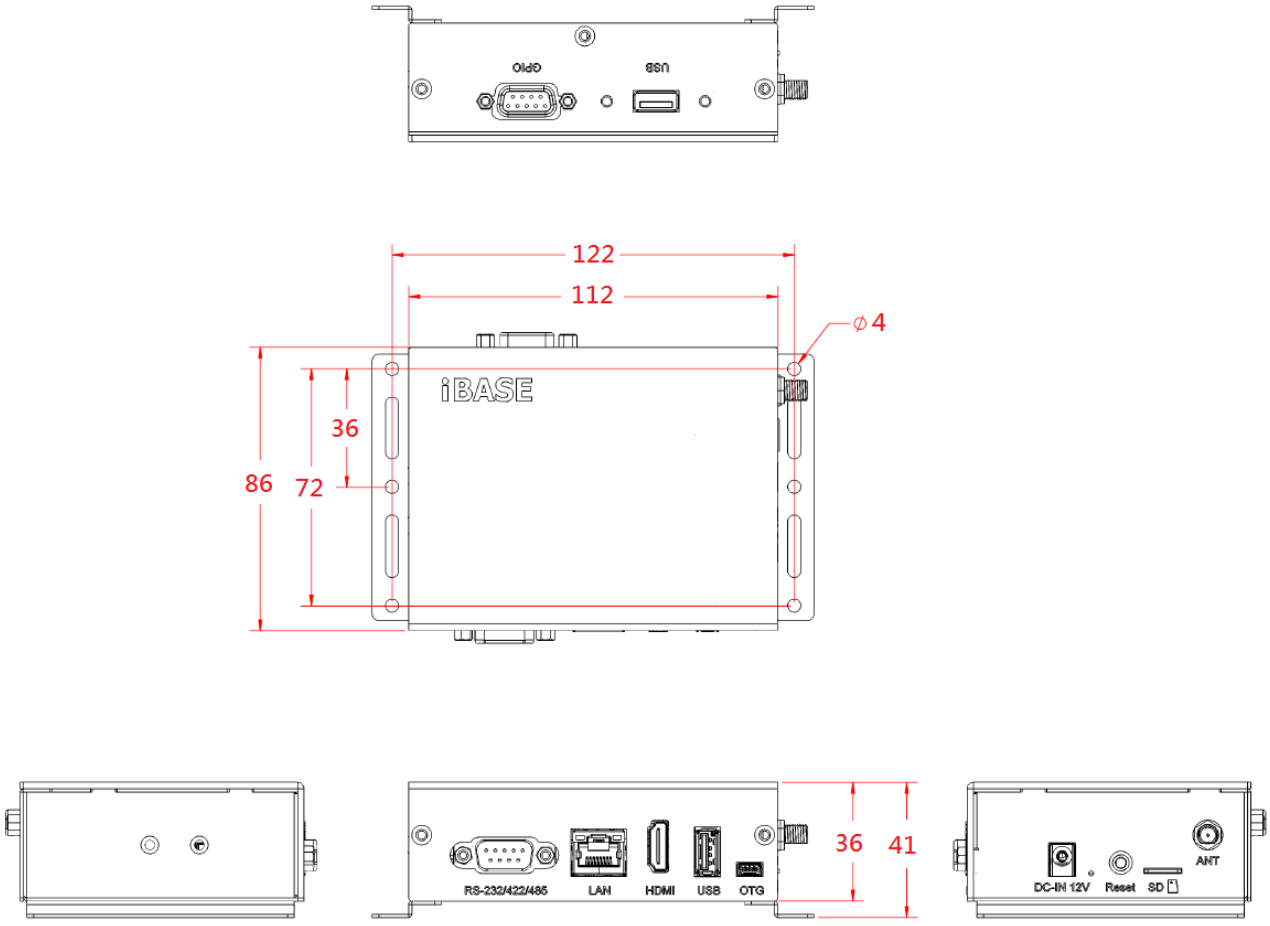
Top View

No.	Name	No.	Name
1	COM RS-232/422/485 Port	7	Reset Button
2	GbE LAN Port	8	MicroSD Card Slot
3	HDMI Port	9	Antenna Hole
4	USB 2.0 Ports	10	GPIO Port
5	USB OTG Port	11	DIN Rail Mounting Holes
6	DC Jack		

DIN Rail Mounting View (Optional)

1.7 Dimensions

Unit: mm



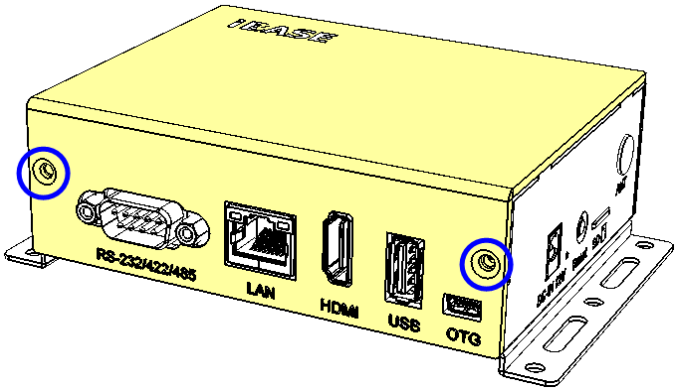
Chapter 2
Hardware Configuration

This section contains general information about:

- Installations
- Jumper and connectors

2.1 Installations

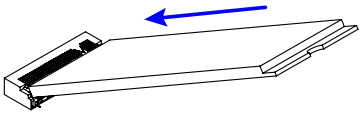
Before installing any card or module into the device, remove the 4 screws as shown in the picture below to take away the device cover.



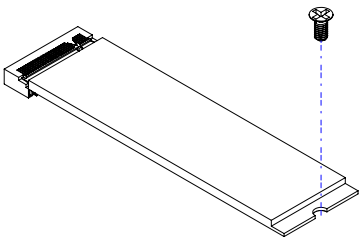
2.1.1 M.2 Cards Installation

To install the M.2 card, remove the device cover first as mentioned above, locate the slot inside the device, and perform the following steps. This is illustrated by the example of M.2 type 2280.

1. Align the keys of the M.2 card with that of the M.2 slot, and insert the card slantwise.



2. Push the M.2 card downwards as shown in the picture below, and fix it onto the brass standoff with a screw.



2.1.2 WiFi / 3G / 4G Antenna Installation

Insert the WiFi / 3G / 4G antenna extension cable through the antenna hole of the front I/O cover and fasten the antenna as shown below. Then apply adhesive around the hex nut behind the front I/O cover to prevent the extension cable from falling off if the cable becomes loose.

1. Fasten the hex nut and the washer. Then install the antenna.
2. Apply adhesive around here.

Info: The diameter of the nut is around 6.35 mm (0.25”-36UNC).

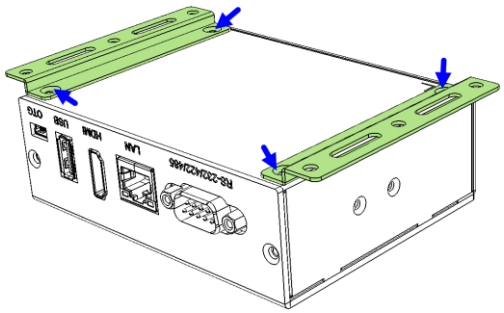
2.1.3 Mounting Installation

Requirements

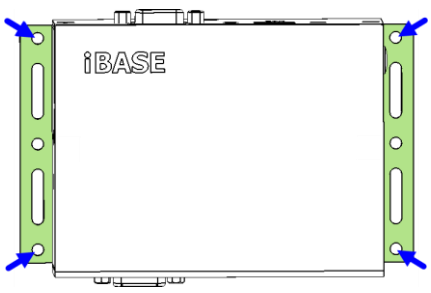
Before mounting the system, ensure that you have enough room for the power adaptor and signal cable routing, and have good ventilation for the power adaptor. The method of mounting must be able to support weight of the device plus the weight of the suspending cables attached to the system. Use the following methods for mounting your system:

Wall Mounting Installation

1. Turn your device upside down. Attach the wall-mount kit to the device and secure with the supplied 4 screws.



2. Prepare at least 4 screws (M3) to install the device on the wall.



2.1.4 COM RS-232/422/485 Port

Pin	Signal Name	Pin	Signal Name
1	DCD, Data carrier detect	6	DSR, Data set ready
2	RXD, Receive data	7	RTS, Request to send
3	TXD, Transmit data	8	CTS, Clear to send
4	DTR, Data terminal ready	9	NC
5	Ground		

Pin	Assignment		
	RS-232	RS-422	RS-485
1	DCD	TX-	DATA-
2	RX	TX+	DATA+
3	TX	RX+	NC
4	DTR	RX-	NC
5	Ground	Ground	Ground
6	DSR	NC	NC
7	RTS	NC	NC
8	CTS	NC	NC
9	NC	NC	NC

2.1.5 GPIO Port

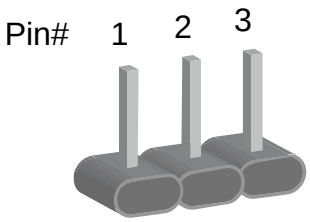
Pin	Signal Name	Pin	Signal Name
1	DIO1	6	DIO5
2	DIO2	7	DIO6
3	DIO3	8	DIO7
4	DIO4	9	DIO8
5	Ground		

2.2 Setting the Jumpers

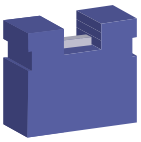
Configure your device by using jumpers to enable the features that you need based on your applications. Contact your supplier if you have doubts about the best configuration for your use.

2.2.1 How to Set Jumpers

Jumpers are short-length conductors consisting of several metal pins with a base mounted on the circuit board. Jumper caps are placed (or removed) on the pins to enable or disable functions or features. If a jumper has 3 pins, you can connect Pin 1 with Pin 2 or Pin 2 with Pin 3 by shorting the jumper.

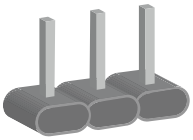
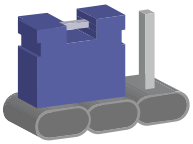
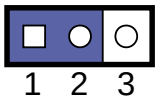
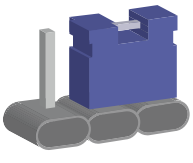
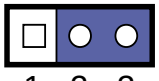


A 3-pin jumper



A jumper cap

Refer to the illustration below to set jumpers.

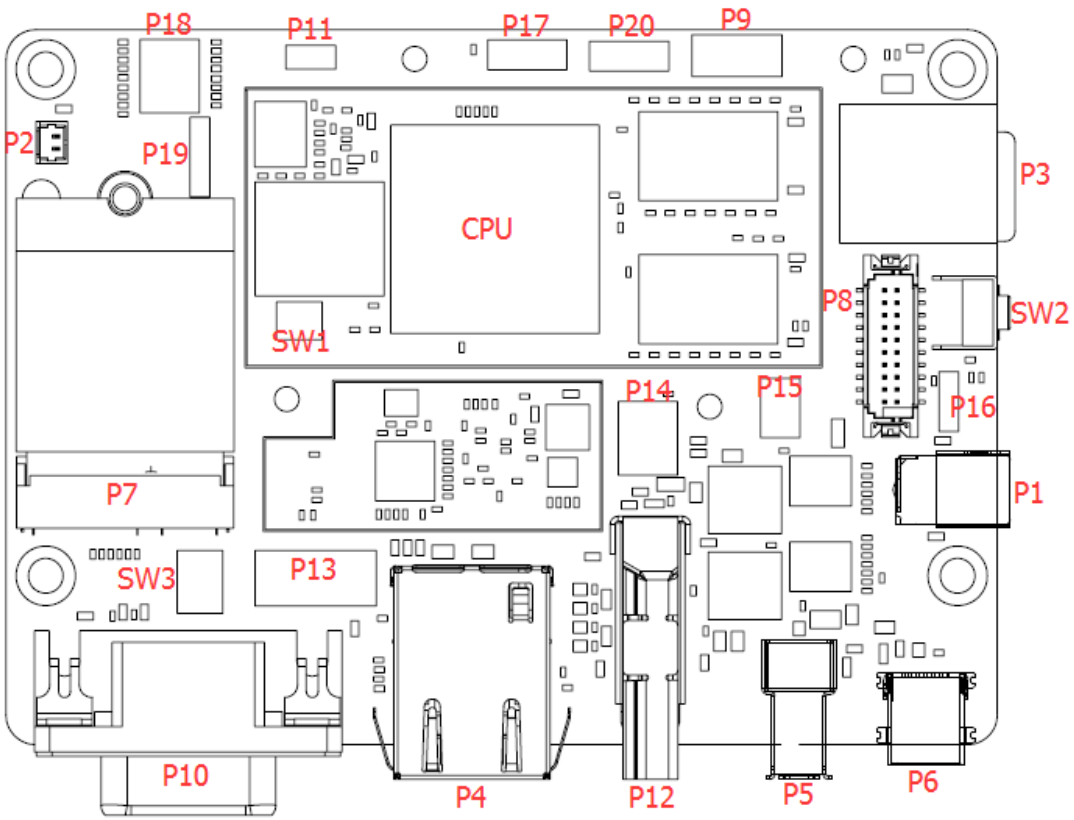
Pin closed	Oblique view	Illustration in the manual
Open		
1-2		
2-3		

When two pins of a jumper are encased in a jumper cap, this jumper is **closed**, i.e. turned **On**.

When a jumper cap is removed from two jumper pins, this jumper is **open**, i.e. turned **Off**.

2.3 Jumper & Connector Locations on Motherboard

Motherboard: IBR115



2.4 Jumper & Connectors Quick Reference

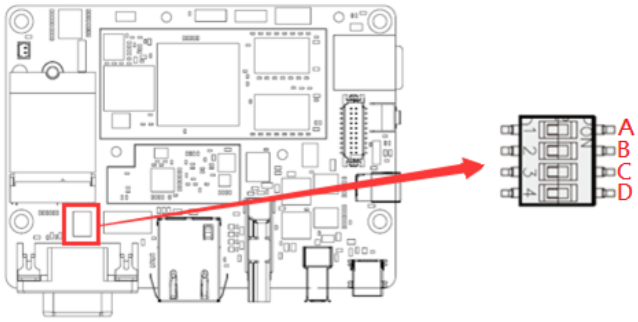
Connectors:

Function	Connector Name	Page
RS-232/422/485 Selection	SW3	14
RTC Lithium Cell Connector	P2	14
COM RS-232/422/485 Port	P10	15
USB Hub Connector	P13	16
Digital (GPIO) Connector	P18	16
LVDS Display Connector	P8	--
LVDS Backlight Control Connector	P9	--
Audio Connector	P14	--
2-Wire UART Connector	P17	--
I^2C Connector	P20	--
System Reset Button	SW2	--
DC Jack	P1	--
MicroSD Card Connector	P3	--
GbE LAN Port	P4	--
USB 2.0 Type A Port	P5	--
Mini-USB OTG Port	P6	--
NGFF M.2 Slot	P7	--
HDMI Port	P12	--
Factory Use Only	SW1, P11, P19	--

Jumpers:

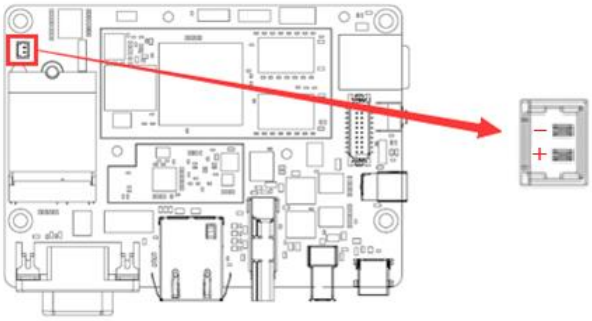
Function	Connector Name	Page
LVDS Power Setting	P16	--
LVDS Backlight Power Setting	P15	--

2.4.1 RS-232/422/485 Selection (SW3)



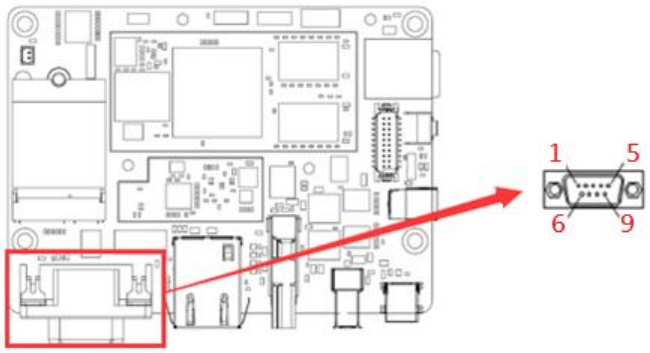
Panel Type	A	B	C	D
RS-422 Full Duplex	NC	ON	ON	ON
Pure RS232 (3T/5R)	NC	OFF	ON	ON
RS-485 Half Duplex (TX Low-Active)	NC	ON	OFF	ON
RS-485 Half Duplex (TX High-Active)	NC	OFF	OFF	ON
RS-422 Full Duplex	NC	ON	ON	OFF
RS-485 Half Duplex	NC	ON	OFF	OFF
Shutdown (Default)	NC	OFF	OFF	OFF

2.4.2 RTC Lithium Cell Connector (P2)



Pin	Signal Name	Pin	Signal Name
1	RTC_VCC	2	Ground

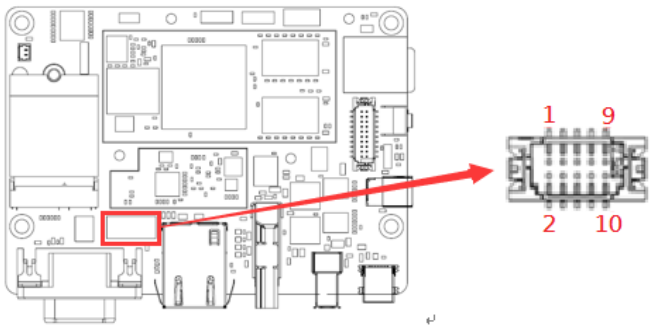
2.4.3 COM RS-232/422/485 Port (P10)



Pin	Signal Name	Pin	Signal Name
1	DCD, Data carrier detect	6	DSR, Data set ready
2	RXD, Receive data	7	RTS, Request to send
3	TXD, Transmit data	8	CTS, Clear to send
4	DTR, Data terminal ready	9	NC
5	Ground		

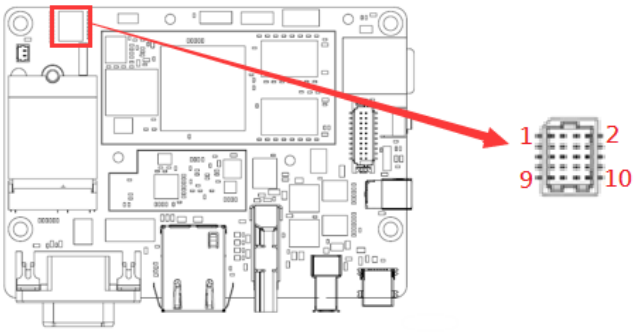
Pin	Assignment		
	RS-232	RS-422	RS-485
1	DCD	TX-	DATA-
2	RX	TX+	DATA+
3	TX	RX+	NC
4	DTR	RX-	NC
5	Ground	Ground	Ground
6	DSR	NC	NC
7	RTS	NC	NC
8	CTS	NC	NC
9	NC	NC	NC

2.4.4 USB Hub Connector (P13)



Pin	Signal Name	Pin	Signal Name
1	Ground	2	NC
3	USB1_DP	4	USB2_POWER
5	USB1_DM	6	USB2_DM
7	USB1_POWER	8	USB2_DP
9	NC	10	Ground

2.4.5 Digital (GPIO) Connector (P18)



Pin	Signal Name	Pin	Signal Name
1	3.3V	2	DIO5
3	DIO1	4	DIO6
5	DIO2	6	DIO7
7	DIO3	8	DIO8
9	DIO4	10	Ground

Chapter 3
Software Setup

This chapter introduces the following setup on the device:

- Make a recovery microSD card (for advanced users only)
- Display parameter setting in kernel

3.1 Make a Recovery MicroSD Card

Note: This is for advanced users who has IBASE standard image file only.

Basically, ISR101 is preloaded with O.S (Android / Linux) into eMMC by default. Connect the TFT-LCD with ISR101 (or HDMI), and 12V power directly.

This chapter guides you to make a recovery boot-up microSD card. IBASE optionally provides 7" /15" LVDS panel and HDMI display for you to prepare the software application pre-development easily under Linux platform.

3.1.1 Preparing the Recovery MicroSD card to Install Linux / Android image into eMMC

Note: All data in the eMMC will be erased.

1. System requirements: ubuntu12.04 or over; a microSD card: 4GB or greater in size.
2. Log in with root.
3. Decompress **ISR101_make_fs.tar.bz2**.

```
tar jxvf ISR101_make_fs.tar.bz2
```

4. Enter ISR101_make_fs directories

   cd ISR101_make_fs

5. Copy new firmware version (eg. 20150814-111827) into **ISR101_make_fs**.

```
cp $(direction to release)/20150814-111827 . –rf
```

6. Confirm the name of the microSD card device used for the upgrade.

   6-A. Check the partition of the current system.

```
#cat /proc/partitions
major minor  #blocks  name
    8        0   976762584 sda
    8        1   970554368 sda1
    8        2           1 sda2
    8        5     6205440 sda5
    8       16   488386584 sdb
    8       17    20988891 sdb1
    8       18           1 sdb2
    8       19   395428864 sdb3
    8       21    52436128 sdb5
    8       22    19530752 sdb6
    8       32   976762584 sdc
```

   6-B. Insert the microSD card and don't mount a microSD card device.

   6-C. Check the partition of the current system again.

```
#cat /proc/partitions
major minor  #blocks  name

    8        0   976762584 sda
    8        1   970554368 sda1
    8        2           1 sda2
    8        5     6205440 sda5
    8       16   488386584 sdb
    8       17    20988891 sdb1
    8       18           1 sdb2
    8       19   395428864 sdb3
    8       21    52436128 sdb5
    8       22    19530752 sdb6
    8       32   976762584 sdc
    8       48     3880960 sdd
    8       49      512000 sdd1
    8       50     3367936 sdd2
```

   6-D. Compare *6-A* and *6-C*, and get sdd.

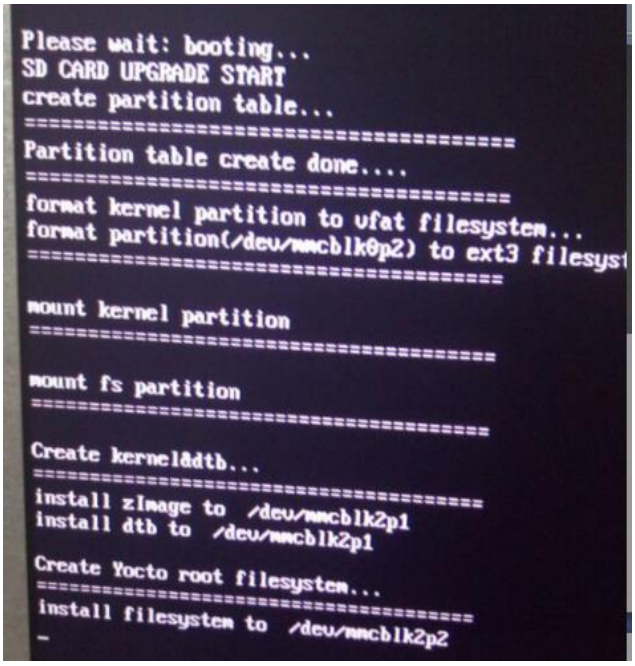
7. Run **make_card script**.

```
./make_emmc_sd.sh /dev/sdd 20150814-111827
```

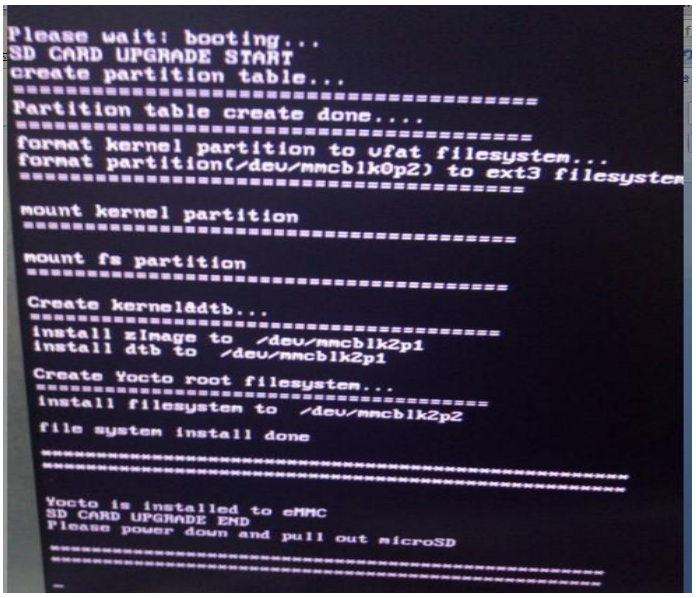
8. The string "file system create done" means success.
9. Pull out the microSD card.

3.1.2 Upgrade Firmware through the Recovery MicroSD Card

1. Insert the microSD card into ISR101.
2. Insert power to boot up and upgrade the firmware via the microSD card.



3. It takes about 15 minutes. After 10 minutes, the screen goes into sleep. When the upgrade is finished, the screen will automatically open. Do not power off.
4. The appearance of the sentence "Yocto is installed to eMMC SD CARD UPGRADE END Please power down and pull out microSD" indicates the installation is successfully completed.



5. Power down and pull out microSD.

Chapter 4
BSP Source Guide

This chapter is dedicated for advanced software engineers only to build BSP source. The topics covered in this chapter are as follows:

- Preparation
- Installing Toolchain
- Building U-Boot
- Building Kernel
- Building RAMdisk Image (Optional)
- Installing Linux to MicroSD Card
- Booting with your MicroSD Card

4.1 Building BSP Source

4.1.1 Preparation

The suggested Host platform is Ubuntu 12.04 and 14.04 in x86 and x64 versions.

1. Install necessary packages before building:

```
sudo apt-get install gawk wget Git-core diffstat unzip texinfo
sudo apt-get install gcc-multilib build-essential chrpath socat
sudo add-apt-repository ppa:git-core/ppa
sudo apt-get update
sudo apt-get install git
sudo apt-get install texinfo
```

2. Decompress the ISR101 source file **ISR101.tar.xz** into "/home/" folder.

4.1.2 Installing Toolchain

Decompress Toolchain **poky.tar** into directory "/opt".

4.1.3 Building U-Boot

```
cd /home/ISR101
cd    ISR101-x11/tmp/work/imx6dlsabresd-poky-linux-gnueabi/u-boot-imx/2016.03-r0/git
source /opt/poky/1.8/environment-setup-cortexa9hf-vfp-neon-poky-linux-gnueabi
make mx6dlsabresd_defconfig
make
```

4.1.4 Building Kernel

```
cd /home/ISR101
cd ISR101-x11/tmp/work/imx6dlsabresd-poky-linux-gnueabi/linux-imx/4.1.15-r0/git
source /opt/poky/1.8/environment-setup-cortexa9hf-vfp-neon-poky-linux-gnueabi
make imx_v7_defconfig
make zImage LOADADDR=0x10008000
make imx6dl-sabresd.dtb
```

4.1.5 Building RAMdisk Image (Optional)

1. Enter ISR101.

```
cd /home/ISR101
```

2. Running the script below.

```
./build.sh ISR101
```

3. See the image in the "release" directory.

```
ls release/ISR101-20170519-030011/
rootfs.tar.bz2  u-boot-imx6dlsabresd.imx  zImage  zImage-imx6dl-sabresd.dtb
```

4.1.6 Install Linux to MicroSD Card

1. Refer to the section *3.1 Make a Recovery MicroSD Card* in chapter 3.
2. Run the make card script below.

```
./make_sd.sh /dev/sdd 20150814-111827
```

3. The appearance of the sentence "SD card for booting create done" indicates the installation is successfully done.
4. Pull out the microSD card

4.1.7 Booting with Your MicroSD Card

Insert the microSD card to your board and insert special COM port dongle to boot from microSD. Connect a debug cable to debug port with serial port 115200/N/8/1 setting on your PC’s serial port program, such as hyperterminal/teraterm. Connect LVDS 15” display panel. Power on and you will see U-Boot prompt.

Appendix

This section provides the information of reference code.

A. How to Use GPIO in Linux

```
# Take gpio1_0    as example, export value:1*32+0=32

# GPIO example 1: Output
echo 32 > /sys/class/gpio/export
echo out > /sys/class/gpio/gpio40/direction
echo 0 > /sys/class/gpio/gpio40/value
echo 1 > /sys/class/gpio/gpio40/value

# GPIO example 2: Input
echo 32 > /sys/class/gpio/export
echo in > /sys/class/gpio/gpio40/direction
cat /sys/class/gpio/gpio40/value
```

B. How to Use Watchdog in Linux

```
// create fd
int fd;
//open watchdog device
fd = open("/dev/watchdog", O_WRONLY);
//get watchdog support
ioctl(fd, WDIOC_GETSUPPORT, &ident);
//get watchdog status
ioctl(fd, WDIOC_GETSTATUS, &status);
//get watchdog timeout
ioctl(fd, WDIOC_GETTIMEOUT, &timeout_val);
//set watchdog timeout
ioctl(fd, WDIOC_SETTIMEOUT, &timeout_val);
//feed dog
ioctl(fd, WDIOC_KEEPALIVE, &dummy);
```

C. eMMC Test

Note: This operation may damage the data stored in eMMC flash. Before starting the test, make sure there is no critical data in the eMMC flash being used.

- **Read, write, and check**

```
MOUNT_POINT_STR="/var"

#create data file
dd if=/dev/urandom of=/tmp/data1 bs=1024k count=10
#write data to emmc
dd if=/tmp/data1 of=$MOUNT_POINT_STR/data2 bs=1024k count=10
#read data2, and compare    with data1
cmp $MOUNT_POINT_STR/data2 /tmp/data1
```

- **eMMC speed test**

```
MOUNT_POINT_STR="/var"

#get    emmc write speed"
time dd if=/dev/urandom of=$MOUNT_POINT_STR/test bs=1024k count=10
# clean caches
echo 3 > /proc/sys/vm/drop_caches
#get    emmc read speed"
time dd if=$MOUNT_POINT_STR/test of=/dev/null bs=1024k count=10
```

D. USB (flash disk) Test

Insert the USB flash disk then assure it is in ISR101 device list.

Note: This operation may damage the data stored in the USB flash disk. Before starting the test, make sure there is no critical data in the eMMC flash being used.

- **Read, write, and check**

```
USB_DIR="/run/media/mmcblk1p1"
#create data file
dd if=/dev/urandom of=/var/data1 bs=1024k count=100
#write data to usb flash disk
dd if=/var/data1 of=$USB_DIR/data2 bs=1024k count=100
#read data2, and compare    with data1
cmp $USB_DIR/data2 /var/data1
```

- **USB speed test**

```
USB_DIR="/run/media/mmcblk1p1"
# usb write speed
dd if=/dev/zero of=$BASIC_DIR/$i/test bs=1M count=1000 oflag=nocache

# usb read speed
dd if=$BASIC_DIR/$i/test of=/dev/null bs=1M oflag=nocache
```

E. MicroSD Card Test

When ISR101 is booted from eMMC, microSD card is “/dev/mmcblk1” and able to see by “ls /dev/mmcblk1*” command:

```
/dev/mmcblk1    /dev/mmcblk1p2 /dev/mmcblk1p4 /dev/mmcblk1p5
/dev/mmcblk1p6
```

When booting from microSD card, replace test pattern “/dev/mmcblk1” to “/dev/mmcblk0”.

Note: This operation may damage the data stored the microSD card. Before starting the test, make sure there is no critical data in the eMMC flash being used.

- **Read, write, and check**

```
SD_DIR="/run/media/mmcblk1"
#create data file
dd if=/dev/urandom of=/var/data1 bs=1024k count=100
#write data to SD card
dd if=/var/data1 of=$ SD_DIR/data2 bs=1024k count=100
#read data2, and compare    with data1
cmp $SD_DIR/data2 /var/data1
```

- **MicroSD card speed test**

```
SD_DIR="/run/media/mmcblk1"

# SD write speed
dd if=/dev/zero of=$SD_DIR/test bs=1M count=1000 oflag=nocache

# SD read speed
dd if=$SD_DIR/test of=/dev/null bs=1M oflag=nocache
```

F. RS-232 Test

```
//open ttymxc1
fd    = open(/dev/ttymxc1,O_RDWR );

//set speed
tcgetattr(fd, &opt);
cfsetispeed(&opt, speed);
cfsetospeed(&opt, speed);
tcsetattr(fd, TCSANOW, &opt)

//get_speed
tcgetattr(fd, &opt);
speed = cfgetispeed(&opt);

//set_parity
// options.c_cflag
options.c_cflag &= ~CSIZE;
options.c_cflag &= ~CSIZE;
options.c_lflag    &= ~(ICANON | ECHO | ECHOE | ISIG);    /*Input*/
options.c_oflag    &= ~OPOST;     /*Output*/
//options.c_cc
options.c_cc[VTIME] = 150;
options.c_cc[VMIN] = 0;
#set parity
tcsetattr(fd, TCSANOW, &options)

//write ttymxc1
write(fd, write_buf, sizeof(write_buf));

//read ttymxc1
read(fd, read_buf, sizeof(read_buf)))
```

G. RS-485 Test

```
//open ttymxc1
fd    = open(/dev/ttymxc1,O_RDWR );

//set speed
tcgetattr(fd, &opt);
cfsetispeed(&opt, speed);
cfsetospeed(&opt, speed);
tcsetattr(fd, TCSANOW, &opt

//get_speed
tcgetattr(fd, &opt);
speed = cfgetispeed(&opt);

//set_parity
// options.c_cflag
options.c_cflag &= ~CSIZE;
options.c_cflag &= ~CSIZE;
options.c_lflag    &= ~(ICANON | ECHO | ECHOE | ISIG);    /*Input*/
options.c_oflag    &= ~OPOST;     /*Output*/
//options.c_cc
options.c_cc[VTIME] = 150;
options.c_cc[VMIN] = 0;
#set parity
tcsetattr(fd, TCSANOW, &options)

//write ttymxc1
write(fd, write_buf, sizeof(write_buf));

//read ttymxc1
read(fd, read_buf, sizeof(read_buf)))
```

H. Ethernet Test

- **Ethernet FTP test**

```
#intall ftp
sudo chmod 777 /home/root/testscript/ethernet/lftp_library/lftp
sudo cp /home/root/testscript/ethernet/lftp_library/lftp /usr/bin/
sudo mkdir /usr/lib/lftp
sudo mkdir /usr/lib/lftp/4.6.3a
sudo cp /home/root/testscript/ethernet/lftp_library/*.so /usr/lib/lftp/4.6.3a
sudo cp /home/root/testscript/ethernet/lftp_library/* /lib/

#lftp server 192.168.1.123
lftp 192.168.1.123 <<EOF
#upload data to server    192.168.1.123
put data1
#download data from server    192.168.1.123
get data1
#exit lftp
bye
```

- **Ethernet Ping test**

```
#ping server 192.168.1.123
ping -c 20 192.168.1.123 >/tmp/ethernet_ping.txt
```

- **Ethernet TCP test**

```
#server 192.168.1.123 run command    “iperf3 -s”
#communicate with server 192.168.1.123 in tcp mode by iperf3
iperf3 -c 192.168.1.123 -i 1 -t 20 -w 32M -P 4
```

- **Ethernet UDP test**

```
#server 192.168.1.123 run command    “iperf3 -s”
#communicate with server 192.168.1.123 in udp mode by iperf3
iperf3 -c $SERVER_IP -u -i 1 -b 200M
```

I. HDMI Test

- **HDMI display test**

```
//Open the file for reading and writing
framebuffer_fd = open("/dev/fb2", O_RDWR);

// Get fixed screen information
ioctl(framebuffer_fd, FBIOGET_FSCREENINFO, &finfo)

// Get variable screen information
ioctl(framebuffer_fd, FBIOGET_VSCREENINFO, &vinfo)

// Figure out the size of the screen in bytes
screensize = vinfo.xres * vinfo.yres * vinfo.bits_per_pixel / 8;

// Map the device to memory
fbp = (char *)mmap(0, screensize, PROT_READ | PROT_WRITE, MAP_SHARED, framebuffer_fd, 0);

// Figure out where in memory to put the pixel
memset(fbp, 0x00,screensize);

//draw point by fbp
long int location = 0;
location = (x+g_xoffset) * (g_bits_per_pixel/8) +
  (y+g_yoffset) * g_line_length;
*(fbp + location + 0) = color_b;
*(fbp + location + 1) = color_g;
*(fbp + location + 2) = color_r;

//close framebuffer fd
close(framebuffer_fd);
```

- **HDMI audio test**

```
#enable hdmi audio
echo 0 > /sys/class/graphics/fb2/blank
#play wav file by    hdmi audio
aplay /home/root/testscript/hdmi/1K.wav -D plughw:0,0
```