DRP/BXP/RKP Series Computers Linux Software Manual

Version 3.0, January 2025

www.moxa.com/products

DRP/BXP/RKP Series Computers Linux Software Manual

The software described in this manual is furnished under a license agreement and may be used only in accordance with the terms of that agreement.

Copyright Notice

Trademarks

Disclaimer

- Information in this document is subject to change without notice and does not represent a commitment on the part of Moxa.
- Moxa provides this document as is, without warranty of any kind, either expressed or implied, including, but not limited to, its particular purpose. Moxa 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, Moxa assumes no responsibility for its use, or for any infringements on the rights of third parties that 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.

Technical Support Contact Information

www.moxa.com/support

Table of Contents

1. Introduction

This Moxa x86 Linux Software User Manual can help x86 Linux users to understand and navigate the usage of Moxa x86 Linux utilities and standard Linux operating system.

Below, we've provided comprehensive information for Getting Started, x86 Linux SDK wizard, Peripheral Interface Operations, Basic Linux Concepts, Troubleshooting and Appendix for x86 Linux user.

Applicable Series

- **BXP** Series
  - BXP-A100, BXP-C100, BXP-A101
- **DRP** Series
  - DRP-A100, DRP-C100
- **RKP** Series
  - RKP-A110, RKP-C110, RKP-C220

2. Getting Started

The Getting Started section will introduce the Linux OS distribution installation instructions.

Linux OS Installation Instructions

Prepare bootable USB drive

At first, prepare a USB storage drive, download the Rufus to create bootable USB drive. Download the ISO image file and restore ISO image into USB storage drive.

Current Supported Distributions

- **Debian**
  - **Debian 11 (bullseye), Linux kernel 5.10**
  - **Debian 12 (bookworm), Linux kernel 6.1**
  - Official Debian 11.8 netinst ISO download link
  - Official Debian 12.8 netinst ISO download link
  - Official Debian installation guide
- **Ubuntu**
  - **Ubuntu 20.04 LTS (Focal Fossa), Linux kernel 5.4 (20.04.1), Linux kernel 5.15 (20.04.5), HWE kernel 5.15 or later version**
  - **Ubuntu 22.04 LTS (Jammy Jellyfish), Linux kernel 5.15 (22.04.3), Linux kernel 6.5 (22.04.4), HWE kernel 6.5 or later version**
  - Official Ubuntu 22.04.03 LTS desktop ISO download link
  - Official Ubuntu 22.04.03 LTS server ISO download link
  - Official Ubuntu installation guide
- **RedHat**
  - **RedHat 9, Linux kernel 5.14**
    - ❐ Official RedHat 9 download link
    - ❐ Official RedHat 9 installation guide
- **CentOS 7**
  - **CentOS 7.9, Linux kernel 3.10**
    - ❐ CentOS-7-x86_64-DVD-2009.iso download link

How to Enter BIOS Menu

Boot up device and press **F2** key from keyboard to enter the BIOS menu, and select **boot from USB** from **UEFI mode**.

Then follow the distribution's official installation guide to finish OS installation procedure.

3. x86 Linux SDK Wizard

Basic Information

The **Moxa x86 Linux SDK** enables the easy deployment on the Moxa x86 IPC platform. The SDK contains components for peripheral drivers, peripheral control tools, and configuration files.

It also provides deployment features, such as build & installation log, dry-run, and self-test on target model. Users can download the Moxa x86 Linux SDK zip file from official product's website.

Below is the list of files:

- ***.tgz -** the tarball file of x86 Linux SDK Install Wizard
- **README.docx/README.md -** the user manual of x86 Linux SDK Install Wizard
- **sources_list -** the list of source code
- **build_info -** build information

NOTE

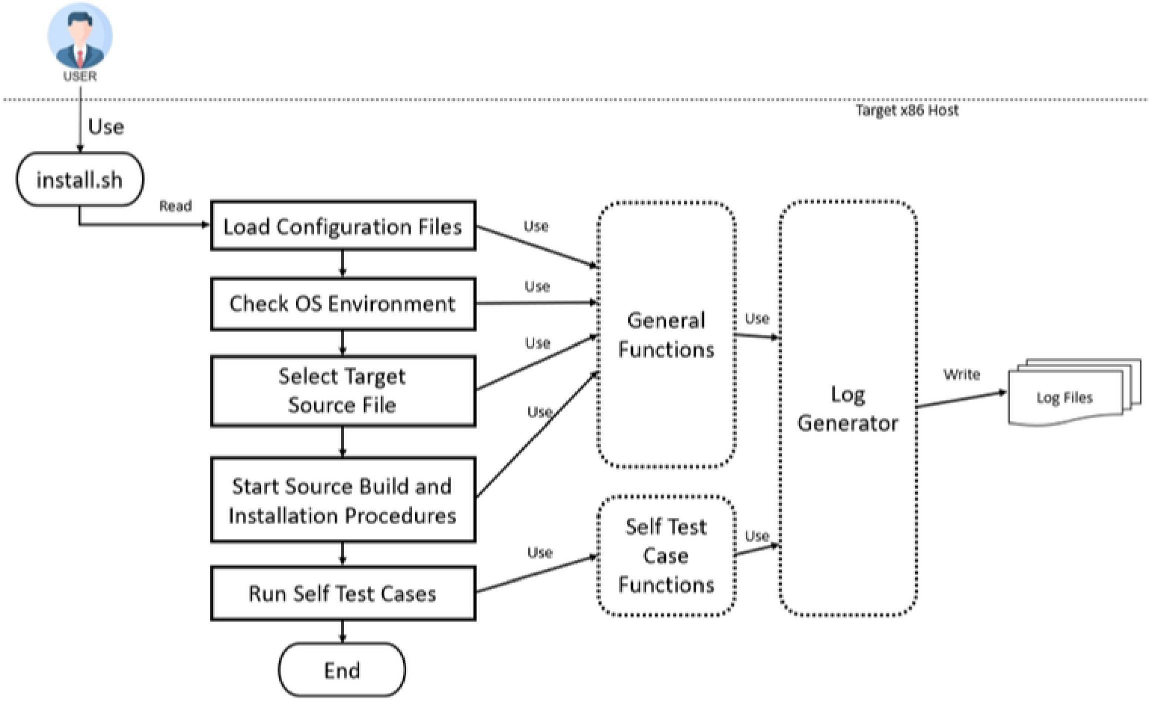
Please extract the **tgz** tarball file under Linux OS environment to avoid file permission issue.

Moxa x86 Linux SDK: Applicable Products

Series	Available SDK Version	Linux Distributions Supported
BXP-A100 BXP-C100	V1.2	Debian 11, Ubuntu 22.04.03 LTS (Jammy Jellyfish), RedHat 9, CentOS 7.9
DRP-A100 DRP-C100	V1.2	Debian 11, Ubuntu 22.04.03 LTS (Jammy Jellyfish), RedHat 9, CentOS 7.9
RKP-A110 RKP-C110	V1.2	Debian 11, Ubuntu 22.04.03 LTS (Jammy Jellyfish), RedHat 9, CentOS 7.9
BXP-A101	V1.2	Debian 12, Ubuntu 22.04 LTS (HWE), RedHat 9
RKP-C220	V1.2	Debian 12, Ubuntu 22.04 LTS (HWE), RedHat 9

Installing the SDK

Software Flow Diagram



Before Starting the Installation

- Please Configure your **network settings** before installation.
- Prepare a **USB storage drive**
  - ➢ Download the ISO image file and restore it to a USB storage drive.
  - ➢ Youn can use the Rufus tool to create a bootable USB drive.
- To extract the tgz tarball file under Linux environment (e.g. tar xvf *.tar.gz)
- Run the --dry-run option before installation, to check the target host device and environment are available.
- Run the --selftest option after installation, to check the status of drivers and tools.

User Interface for Installing the SDK

User Interface	Main Command	Sub Command	Option	Description
`install.sh`				Start to install all procedures (default)
			-y, --yes	Automatic yes to prompts
	-h, --help			Display the help menu
	-v, --version			Display the version information
	-s, --selftest			Run the self test cases
	--uninstall			Uninstall driver and tool
	--dry-run			It won't perform the installation, list available driver and tool only
			--force	Install driver and tool even if the version is the same or older (default is to install newer version)

4. Peripheral Interface Operations

This guide is introduced the usage of **Moxa peripheral interface control utility**. These utilities should be installed after the x86 Linux SDK Wizard installation procedure.

Users can check the status of utilities via running ./install.sh –selftest command.

Utilities

Utilities Supported

Series	Serial Port Utility	DIO Port Utility	PLED Utility	Relay Utility	Power Input Utility	USB Power Utility
BXP-A100 BXP-C100	O	O	X	X	X	X
DRP-A100 DRP-C100	O	X	X	X	X	X
RKP-A110 RKP-C110	O	O	X	X	X	X
BXP-A101	O	O	X	X	X	X
RKP-C220	O	O	X	X	X	X

Series	LTE (mPCIe slot) Module Utility	Scaler Utility	MCU Manager and upgrade tool	HSR/PRP Utility	IRIG-B Utility	MCIM wrapper	Disk Hotswap Daemon
BXP-A100 BXP-C100	X	X	X	X	X	O	X
DRP-A100 DRP-C100	X	X	X	X	X	O	X
RKP-A110 RKP-C110	X	X	X	X	X	O	X
BXP-A101	O	X	X	X	X	O	X
RKP-C220	X	X	X	X	X	O	X

Serial Ports

The serial ports support RS-232, RS-422, RS-485 2-wire, and RS-485 2-wire operation modes with flexible baudrate settings. The default operation mode is RS-232. You can use the **mx-uart-ctl** command to change the operation mode.

Syntax

```
mx-uart-ctl -p <port_number> [-m <uart_mode>]
```

- Port Number
  - 0, 1, 2, 3…
- UART Mode

Option	UART Mode
None	Set target port to UART mode
0	RS-232
1	RS-485 2-wire
2	RS-422
3	RS-485 4-wire

- Drivers Dependency
  - moxa-it87-gpio-driver
  - moxa-it87-serial-driver
  - moxa-mxuport-driver
- Libraries Dependency
  - libgpiod

Usage of UART mode control

```
Usage:
                mx-uart-ctl -p <port_number> [-m <uart_mode>]

OPTIONS:
                -p <port_number>
                                Set target port.
                -m <uart_mode>
                                Set target port to uart_mode
                                0 --> set to RS-232 mode
                                1 --> set to RS-485-2W mode
                                2 --> set to RS-422 mode
                                3 --> set to RS-485-4W mode

Example:
                Get mode from port 0
                # mx-uart-ctl -p 0

                Set port 1 to RS232 mode
                # mx-uart-ctl -p 1 -m 0Current uart mode is RS422/RS485-4W
interface.
```

Digital I/Os (DIOs)

Moxa DIO port control tool **mx-dio-ctl** is for getting DI/DO and setting DO ports status (low/high).

Syntax

```
mx-dio-ctl <-i|-o <#port number> [-s <#state>]>
```

- State
  - 0 Low
  - 1 High
- Drivers dependency
  - moxa-it87-gpio-driver
- Libraries dependency
  - libgpiod

For details,see **Installing Drivers for Interfaces** and **Installing Libraries.**

Usage of DIO state control

```
Usage:
        mx-dio-ctl <-i|-o <#port number> [-s <#state>]>

OPTIONS:
        -i <#DIN port number>
        -o <#DOUT port number>
        -s <#state>
                Set state for target DOUT port
                0 --> LOW
                1 --> HIGH

Example:
        Get value from DIN port 0
        # mx-dio-ctl -i 0
        Get value from DOUT port 0
        # mx-dio-ctl -o 0

        Set DOUT port 0 value to LOW
        # mx-dio-ctl -o 0 -s 0
        Set DOUT port 0 value to HIGH
        # mx-dio-ctl -o 0 -s 1
```

LTE (mPCIe slot) Module Control

Moxa LTE (mPCIe slot) module control tool **mx-module-ctl** is provided to control LTE power on/off state and SIM card select functions

- Drivers dependency
  - moxa-it87-gpio-driver
  - moxa-gpio-pca953x-driver
- Libraries dependency
  - libgpiod

Usage of LTE (mPCIe slot) module control tool

```
Usage:
        mx-module-ctl [Options]

Operations:
        -s, --slot <module_slot_id>
                Select module slot
        -p, --power [on|off]
                Get/Set power on/off module
        -r, --reset [on|off]
                Get/Set reset pin to high(on)/low(off) to slot
        -i, --sim 1|2
                Get/Set sim card slot

Example:
        Power on module 1
        # mx-module-ctl -s 1 -p on

        Set module 2 reset pin to high
        # mx-module-ctl -s 2 -r on

        Select SIM 2 for module 1
        # mx-module-ctl -s 1 -i 2

        Get power status of module 1
        # mx-module-ctl -s 1 -p

        Get current SIM slot of module 1
        # mx-module-ctl -s 1 -i
```

Programmable LED Control

Moxa LED control tool **mx-led-ctl** is provided to control programmable LEDs light on/off.

- Drivers dependency
    - moxa-it87-gpio-driver
    - moxa-gpio-pca953x-driver
- Libraries dependency
    - libgpiod

Usage of programmable LED control tool

```
Usage:
        mx-led-ctl -i <led_index> [on|off]

OPTIONS:
        -i <led_index>
                Set LED index.

Example:
        Get state from index 1
        # mx-led-ctl -i 1

        Set index 1 to on
        # mx-led-ctl -i 1 on
```

Relay Port State Control

Moxa relay port state control tool **mx-relay-ctl** is for getting and setting relay ports status (NO: Normal Open/NC: Normal Closed).

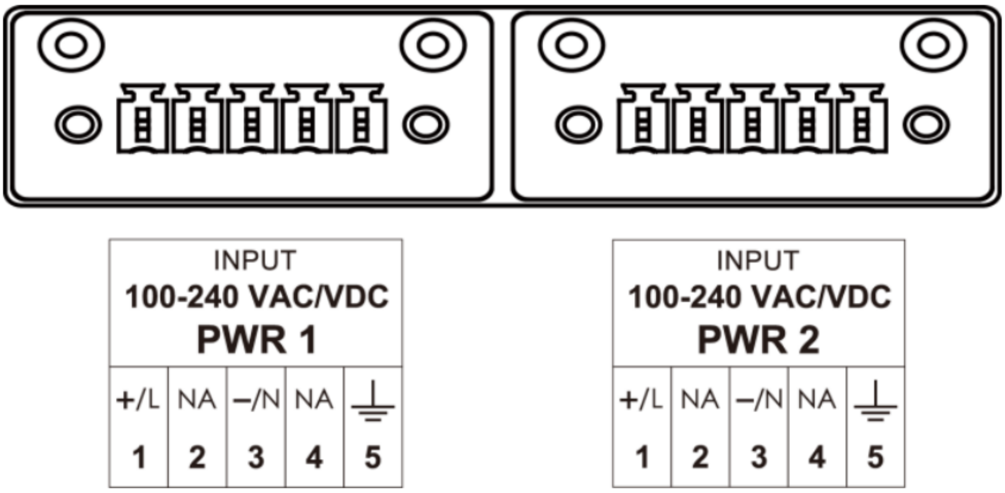
- Drivers dependency
    - moxa-it87-gpio-driver
- Libraries dependency
    - libgpiod

Usage of relay state control

```
Usage:
                mx-relay-ctl -p <port_number> [-m <relay_mode>]

OPTIONS:
                -p <port_number>
                                Set target port.
                -m <relay_mode>
                                Set target port to relay_mode
                                0 --> set to NC (Normal Closed) mode
                                1 --> set to NO (Normal Open) mode

Example:
                Get mode from port 0
                # mx-relay-ctl -p 0

                Set port 0 to mode NC
                # mx-relay-ctl -p 0 -m 0

                Set port 0 to mode NO
                # mx-relay-ctl -p 0 -m 1
```

Power Input Port State

Moxa power input port state tool **mx-input-power-state** is for getting power input ports status (connected/disconnected):



- Drivers dependency
  - oxa-it87-gpio-driver
- Libraries dependency
  - libgpiod

Usage of power input port state tool

```
USAGE:
        mx-input-power-state -i <power_port>

OPTIONS:
        -i <power_port>
                Get power input port state <connected/disconnected>

EXAMPLE:
        Get power input port 0 state
        mx-input-power-state -i 0
```

USB Port Power State Control

Moxa USB port power state control tool **mx-usb-power-ctl** is for setting/getting USB ports (front/rear/internal) power state (off/on) control:

- Drivers dependency
  - moxa-it87-gpio-driver
- Libraries dependency
  - libgpiod

Usage of USB power port state control tool

```
USAGE:
        mx-usb-power-ctl -i <usb_port> [-s <state>]

OPTIONS:
        -i <usb_port>
                Get USB port power state
                        0: front
                        1: rear
                        2: internal

        -s <state>
                Set USB port power state
                        0: off
                        1: on

EXAMPLE:
        Get USB front port power state
        mx-usb-power-ctl -i 0

        Get USB rear port power state
        mx-usb-power-ctl -i 1

        Set USB front port power state to off
        mx-usb-power-ctl -i 0 -s 0

        Set USB internal port power state to on
        mx-usb-power-ctl -i 2 -s 1
```

Scaler Utility

Moxa scaler utility is designed to configure basic settings of display devices, such as brightness, touch panel status, and OSD settings

Usage

```
Usage:
        mx-scaler-util [Options]...
Options:
        -v, --version
                Get scaler firmware version
        -s, --status
                Get system status
        -m, --model
                Get model name
        -b, --brightness [0~10]
                Set brightness, query without arg
        -p, --touch-panel [0~1]
                Set touch panel on(1)/off(0), query without arg
        -o, --osd-config [0~1]
                Set OSD on(1)/off(0), query without arg
```

MCU upgrade tool

The mx-lpc-mcu-upgrade-tool is a command-line utility designed for upgrading the firmware of MCU.

WARNING

Before using MCU firmware upgrade tool, please stop Moxa MCU related services to avoid communication conflict issue.

Usage

```
Usage:
        mx-lpc-mcu-upgrade-tool [Options]...
Options:
        -f, --file
                Start MCU upgrade from file
        -v, --version
                Get current MCU version
Example:
        mx-lpc-mcu-upgrade-tool -f FB_MCU_V3000_V1.00S03_22060219.bin
        mx-lpc-mcu-upgrade-tool -v
```

Moxa MCU Manager

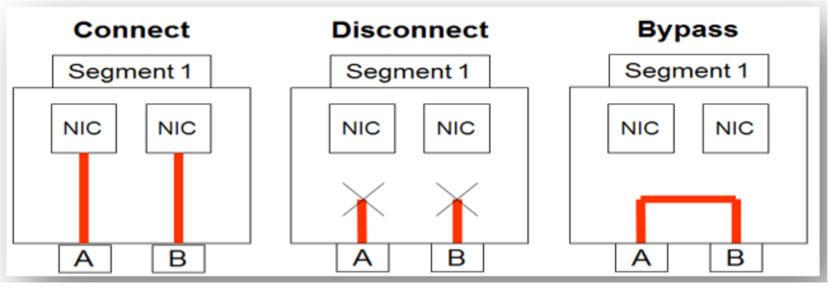
Moxa MCU Manager (MMM) is used to control MCU(microcontroller) on Moxa x86 computer products. Including the LAN bypass, panel display, panel programmable LEDs, and update MCU ROM firmware.

Please ensure that **mx-mcud** is running on background as daemon.

Usage

```
MOXA MCU Management Command-line Utility

 Usage:
    mx-mcu-mgmt [command]

 Flags:
    -h, --help       Prints help information
    -v, --version    Prints utility version

 Commands:
    mcu_version      Get MCU firmware version
    relay            Control relay mode
    wdt_reset        Control watchdog reset mode
    wdt_relay        Control watchdog relay mode
    poweroff_relay   Control power off (S5) relay mode
    app_wdt_reset    Control app watchdog reset mode
    app_wdt_relay    Control app watchdog relay mode
    app_wdt_timout   Control app watchdog timeout
```

LAN Bypass Modes



The LAN Bypass feature support the following three modes:

- **[Connect]**
  - ➢ LAN A and LAN B ports are connected to the NICs and is data transmitted through system normally.
- **[Disconnect]**
  - ➢ LAN A and LAN B ports are neither connected to the NICs nor to each other, which means that data packets are blocked.
- **[Bypass]**
  - ➢ LAN A and LAN B ports are connected to each other to keep transmitting data without interruption even when a system device crashes or encounters a cyber attack.

Options of LAN Bypass Modes Control Utility

Relay Mode

- Description
  - ➢ Directly to get or set LAN Bypass mode from relay states.
- Example
  - ➢ Get LAN Bypass Mode: **mx-mcu-mgmt relay get_mode**
  - ➢ Set LAN Bypass Mode:
    - ❒ Set **Connect** Mode: **mx-mcu-mgmt relay set_mode connect**
    - ❒ Set **Disconnect** Mode: **mx-mcu-mgmt relay set_mode disconnect**
    - ❒ Set **Bypass** Mode: **mx-mcu-mgmt relay set_mode bypass**

Watchdog Reset Mode

- Description
  - ➢ Use the Watchdog Reset Mode to indicate if the **system needs to be reset** after the MCU RTC watchdog (ds1374) is triggered
- Example
  - ➢ Get Watchdog Reset Mode: **mx-mcu-mgmt wdt_reset get_mode**
  - ➢ Set Watchdog Reset Mode
    - ❒ Watchdog Reset OFF: **mx-mcu-mgmt wdt_reset set_mode off**
    - ❒ Watchdog Reset ON: **mx-mcu-mgmt wdt_reset set_mode on**

Watchdog Relay Mode

- Description
  - ➢ Use this Watchdog Relay Mode to indicate the **relay mode** to switch to after the MCU RTC watchdog (ds1374) is triggered.
- Example
  - ➢ Get Watchdog Relay Mode: **mx-mcu-mgmt wdt_relay get_mode**
  - ➢ Set Watchdog Relay Mode
    - ❒ Set Watchdog Relay **Connect** Mode: **mx-mcu-mgmt wdt_relay set_mode connect**
    - ❒ Set Watchdog Relay **Disconnect** Mode: **mx-mcu-mgmt wdt_relay set_mode disconnect**
    - ❒ Set Watchdog Relay **Bypass** Mode: **mx-mcu-mgmt wdt_relay set_mode bypass**

Power Off Relay Mode

- Description
  - ➢ Use this Power Off Relay Mode to indicate the **relay mode** to switch to after the system is **powered off (S5 state)**.
- Example
  - ➢ Get Power Off Relay Mode: **mx-mcu-mgmt poweroff_relay get_mode**
  - ➢ Set Power Off Relay Mode
    - ❒ Set Power Off Relay as **Disconnect** Mode: **mx-mcu-mgmt poweroff_relay set_mode disconnect**
    - ❒ Set Power Off Relay as **Bypass** Mode: **mx-mcu-mgmt poweroff_relay set_mode bypass**

App Watchdog Modes Control Utility

The App Watchdog Modes Control Utility is to configure MCU's behavior.

It provides set MCU timeout value, timeout-reset function, timeout-relay mode, and kicking service and daemon.

APP WDT Mode

- Description
  - Use this mode to enable or disable the MCU watchdog application
  - Activating the watchdog function is key to creating a trigger to activate LAN bypass when your application encounters issues or is unresponsive.
- Example
  - Get APP WDT Mode and Timeout value: **mx-mcu-mgmt app_wdt_timout get_timeout**
  - Set APP WDT Mode and Timeout value:
    - Enable APP WDT Mode and set 10 sec timeout: **mx-mcu-mgmt app_wdt_timout set_timeout 10**
    - Disable APP WDT Mode: **mx-mcu-mgmt app_wdt_timout set_timeout 0**

APP WDT Reset Mode

- Description
  - Use this mode to indicate if the **system needs to be reset** after the MCU app watchdog is timeout triggered.
- Example
  - Get App Watchdog Reset Mode: **mx-mcu-mgmt app_wdt_reset get_mode**
  - Set App Watchdog Reset Mode when app watchdog is triggered
    - Disable the reset system function: **mx-mcu-mgmt app_wdt_reset set_mode off**
    - Enable the reset system function: **mx-mcu-mgmt app_wdt_reset set_mode on**

APP WDT Relay Mode

- Description
  - Use this mode to indicate the relay mode to switch to after the MCU app watchdog is timeout triggered.
- Example
  - Get App Watchdog Relay Mode: **mx-mcu-mgmt app_wdt_relay get_mode**
  - Set App Watchdog Relay Mode when app watchdog is timeout triggered
  - Set the App Relay mode
    - Set **Connect** Mode: **mx-mcu-mgmt app_wdt_relay set_mode connect**
    - Set **Disconnect** Mode: **mx-mcu-mgmt app_wdt_relay set_mode disconnect**
    - Set **Bypass** Mode: **mx-mcu-mgmt app_wdt_relay set_mode bypass**

Moxa Disk Hotswap Daemon

Moxa Disk Hotswap Daemon is used to monitor the disk plug/unplug status with push buttons and programmable LEDs. User can remove disk via pressed push button to unmount and remove disk safely.

This feature support on V3400 series.

- Libraries dependency
  - libgpiod

Add systemd service to use (if needed)

edit /lib/systemd/system/mx_disk_hotswapd.service

```
[Unit]
Description=Moxa disk hotswap daemon service

[Service]
Type=oneshot
ExecStart=/usr/sbin/mx-disk-hotswapd
RemainAfterExit=yes

[Install]
WantedBy=multi-user.target
```

then enable service.

```
systemctl enable mx_disk_hotswapd.service
```

If user pressed push button over 3 seconds, the programmable LED will blinks 3 times and turn off light, and daemon will start to unmount and remove target disk.

Alternatively, if pressed push button less than 3 seconds, the daemon will scan disk and mount target disk, the programmable LED will turn on to notify user the disk has been mounted, for example:

```
/dev/sda2 on /media/disk1p2 type squashfs (ro,relatime,errors=continue)
/dev/sda1 on /media/disk1p1 type vfat
(rw,relatime,fmask=0022,dmask=0022,codepage=437,iocharset=ascii,shortname=mixed
,utf8,errors=remount-ro)
/dev/sda3 on /media/disk1p3 type ext4 (rw,relatime)
```

HSR/PRP Utility

Moxa HSR/PRP card utility is based on SMBUS to query FPGA related register.

Usage

```
[root@localhost moxa]# mxhsrprpd -h
Usage:
        -h: Show this information.
        -B: Run daemon in the background
        -b: SMBUS device, default is /dev/i2c-0
        -t: HSR/PRP Status update period. Default is 3 second.
        -m: configure to prp or hsr mode, default is prp mode.
                The argurement is [index]:[mode]
                [index] range from 0~7.
                [mode] 0 is prp, mode 1 is hsr.
                Ex: Set card 0 to hsr mode, card 1 to prp mode.
                root@Moxa:~# mxhsrprpd -t 2 -m 0:1,1:0
        -s: configure fiber speed, default is auto detect mode.
                The argurement is [index]:[speed]
                [index] range from 0~7.
                [speed] 0 is 100M, 1 is 1000M. (default fiber speed is 1000M)
                Ex: Set card 0 fiber speed to 100M, card 1 fiber speed to
1000M.
                root@Moxa:~# mxhsrprpd -t 2 -s 0:0,1:1
```

Add systemd service to use (if needed)

edit /lib/systemd/system/mx_hsrprp.service

```
[Unit]
Description=Moxa HSR-PRP daemon service

[Service]
Type=oneshot
ExecStart=/usr/sbin/mx_hsrprp start
ExecStop=/usr/sbin/mx_hsrprp stop
RemainAfterExit=yes

[Install]
WantedBy=multi-user.target
```

then enable service.

```
systemctl enable mx_hsrprp.service
```

IRIG-B Utility

Utility for controlling DA-IRIG-B expansion module Compile and install the IRIG-B time sync daemon.

Usage

```
[root@localhost moxa]# ServiceSyncTime -h
Found the IRIG-B module, Hardware ID = 7
IRIG-B time sync daemon.
Usage: ServiceSyncTime -t [signal type] -I -i [Time sync interval] -s [Time
Source] -p [Parity check mode] -B
   -t - [signal type]
       0 - TTL
       1 - DIFF
       default value is 1
   -I - Inverse the input signal
   -s - [Time Source] The sync source from FREERUN(Internal RTC), Fiber or
IRIG-B port
       0 - FREERUN(Internal RTC) module
       1 - Fiber port
       2 - IRIG-B port
       default value is 2
   -i - [Time sync interval] The time interval in seconds to sync the IRIG-B
time into system time.
       1 ~ 86400 Time sync interval. Default is 10 second.
   -p - [Parity check mode] Set the parity bit
       0: EVEN
       1: ODD
       2: NONE
       default value is 0
   -B - Run daemon in the background
Usage example: Enable to sync time from IRIG-B Port 1, in TTL signal type every
10 seconds. The input signals is not inverse.
root@Moxa:~# ServiceSyncTime -t 0 -i 10
```

Use systemd service step by step

1. Disable NTP service

WARNING

NTP service affects IRIG-B service time syncing.

- Disable service

```
timedatectl set-ntp false
```

- Make sure NTP service is inactive

```
timedatectl status
               Local time: Mon 2023-02-13 02:27:54 PST
           Universal time: Mon 2023-02-13 10:27:54 UTC
                 RTC time: Mon 2023-02-13 10:27:54
                Time zone: America/Los_Angeles (PST, -0800)
System clock synchronized: yes
              NTP service: inactive
          RTC in local TZ: no
```

2. Config IRIG-B time sync service

- Edit /usr/sbin/mx_irigb.sh to config service options MX_IRIGB_SERVICESYNCTIME_OPTS.

INFOMATION

For more details about options, please refer to ServiceSyncTime -h

```
...
# The time sync daemon default configure wtih
#   -t 1 - Sync time in DIFF signal format
#   -i 10 - The time interval in 10 seconds to sync the IRIG-B time into
system time.
#   -B - Run daemon in the background
#
MX_IRIGB_SERVICESYNCTIME_OPTS="-t 1 -i 10 -B"
...
```

3. Start IRIG-B time sync service

- Create and edit systemd service file /lib/systemd/system/mx_irigb.service

```
[Unit]
Description=Moxa DA-IRIG-B daemon service

[Service]
Type=oneshot
ExecStart=/usr/sbin/mx_irigb.sh start
ExecStop=/usr/sbin/mx_irigb.sh stop
RemainAfterExit=yes

[Install]
WantedBy=multi-user.target
```

- Launch service

```
$ systemctl daemon-reload
$ systemctl enable mx_irigb.service
Created symlink /etc/systemd/system/multi-

user.target.wants/mx_irigb.service →

/lib/systemd/system/mx_irigb.service.
$ systemctl start mx_irigb.service
$ systemctl status mx_irigb.service
```

```
• mx_irigb.service - Moxa DA-IRIG-B daemon service
     Loaded: loaded (/lib/systemd/system/mx_irigb.service; enabled;
vendor preset: enabled)
     Active: active (exited) since Tue 2023-02-14 01:48:29 PST; 5s ago
    Process: 8322 ExecStart=/usr/sbin/mx_irigb.sh start (code=exited,
status=0/SUCCESS)
   Main PID: 8322 (code=exited, status=0/SUCCESS)
        CPU: 9ms

Feb 14 01:48:29 moxa systemd[1]: Starting Moxa DA-IRIG-B daemon
service...
Feb 14 01:48:29 moxa systemd[1]: Finished Moxa DA-IRIG-B daemon service.
```

MCIM wrapper

MCIM wrapper means Moxa Computer Interface Manager (MCIM) shell script based wrapper. It's provide users with commands similar to MCIM when operating peripherals.

Usage

```
 The Moxa Computer Interface Manager (MCIM) is a tool designed to simplify
 user control of peripherals. The design of MCIM aims to enhance
 operational efficiency, enabling users to conveniently handle tasks
 related to peripheral devices.
Usage:
  mx-interface-mgmt [command]

Available Commands:
  cellular     Manages the cellular modem
  dio          Manages digital inputs and outputs for external devices
  led          Manages LED indicators
  relay        Manages the relay mode
  serialport   Manages the serial port
  input_power  Manages the power input state
  usb_power    Manages the usb power state

Flags:
  -h, --help      help for mx-interface-mgmt

Use "mx-interface-mgmt [command] --help" for more information about a command.
```

Usage (cellular wrapper)

```
Usage:
  mx-interface-mgmt cellular <NAME> <COMMAND> [ARG]

Available Commands:
  Get the power state of a cellular
    $ mx-interface-mgmt cellular <cellular_name> get_power
  Set the power state of a cellular
    $ mx-interface-mgmt cellular <cellular_name> set_power <power_state>
  Get the SIM slot of a cellular
    $ mx-interface-mgmt cellular <cellular_name> get_sim_slot
  Set the SIM slot of a cellular
    $ mx-interface-mgmt cellular <cellular_name> set_sim_slot <sim_slot>

Arguments:
  cellular_name: The slot number of cellular (e.g. 1|2)
  power_state: on|off
  sim_slot: 1|2
```

Usage (dio wrapper)

```
Usage:
  mx-interface-mgmt dio <NAME> <COMMAND> [ARG]

Available Commands:
  Get the state of a dio
    $ mx-interface-mgmt dio <dio_name> get_state
  Set the state of a dio
    $ mx-interface-mgmt dio <dio_name> set_state <dio_state>

Arguments:
  dio_name: The name of dio (e.g. DI0、DO0)
  dio_state: 0(low)|1(high)
```

Usage (led wrapper)

```
Usage:
  mx-interface-mgmt led <NAME> <COMMAND> [ARG]

Available Commands:
  Get the state of a LED
    $ mx-interface-mgmt led <led_name> get_state
  Set the state of a LED
    $ mx-interface-mgmt led <led_name> set_state <led_state>

Arguments:
  led_name: The number of LED (e.g. 0, 1, 2, ......)
  led_state: on|off
```

Usage (relay wrapper)

```
Usage:
  mx-interface-mgmt relay <NAME> <COMMAND> [ARG]

Available Commands:
  Get the mode of a relay
    $ mx-interface-mgmt relay <relay_name> get_mode
  Set the mode of a relay
    $ mx-interface-mgmt relay <relay_name> set_mode <relay_mode>

Arguments:
  relay_name: The number of relay (e.g. 0, 1, 2, ......)
  relay_mode: 0|1
              0 --> set to NC (Normal Closed) mode
              1 --> set to NO (Normal Open) mode
```

Usage (input_power wrapper)

```
Usage:
  mx-interface-mgmt input_power <NAME> <COMMAND> [ARG]

Available Commands:
  Get the state of a input_power
    $ mx-interface-mgmt input_power <input_power_name> get_state

Arguments:
  input_power_name: The number of input_power (e.g. 0, 1, 2, ......)
```

Usage (usb_power wrapper)

```
Usage:
  mx-interface-mgmt usb_power <NAME> <COMMAND> [ARG]

Available Commands:
  Get the usb power state of a port
    $ mx-interface-mgmt usb_power <usb_port> get_state
  Set the usb power state of a port
    $ mx-interface-mgmt usb_power <usb_port> set_state <state>

Arguments:
  usb_port: Get USB port power state
                        0: front
                        1: rear
                        2: internal
  state: Set USB port power state
                        0: off
                        1: on
```

Usage (serialport wrapper)

```
Usage:
  mx-interface-mgmt serialport <NAME> <COMMAND> [ARG]

Available Commands:
  Get the interface of a serial port
    $ mx-interface-mgmt serialport <serialport_name> get_interface
  Set the interface of a serial port
    $ mx-interface-mgmt serialport <serialport_name> set_interface
<serial_interface>

Arguments:
  serialport_name: The number of serial port (e.g. 0, 1, 2, ......)
  serial_interface:
                0 --> set to RS-232 mode
                1 --> set to RS-485-2W mode
                2 --> set to RS-422 mode
                3 --> set to RS-485-4W mode
```

Drivers

Applicable Drivers

The x86 Linux SDK Install Wizard includes drivers for GPIO, RS-485 Automatic Direction Control (ADDC), Watchdog timer, and Moxa UPort.

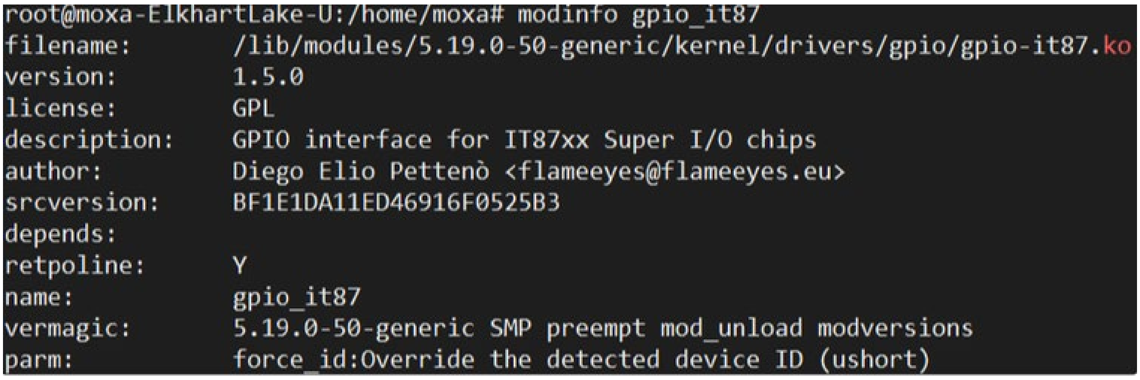
Available Models	it87_ gpio	it87_ serial	it87_ wdt	mxuport	mxu11x0	sdhci-pci	gpio-pca953x	hid-ft260	irigb	i915 (backport)
BXP-A100 BXP-C100	O	O	O	O	X	*[1]	X	X	X	X
DRP-A100 DRP-C100	O	O	O	O	X	X	X	X	X	X
RKP-A110 RKP-C110	O	O	O	O	X	X	X	X	X	X
BXP-A101	O	O	O	X	X	*[2]	X	X	X	X
RKP-C220	O	O	O	O	X	X	X	X	X	X

*[1]: *A100 on Debian 11
*[2]: Debian 11

moxa-it87-gpio-driver

The purpose of **moxa-it87-gpio-driver** is controlling GPIO interface for **IT87xx Super I/O** chips, based on the Linux kernel drivers/gpio/gpio-it87.c. removed label for Moxa utilities' compatibility and fix-up some issues.

Kernel module information



```
root@moxa-ElkhartLake-U:/home/moxa# modinfo gpio_it87
filename:       /lib/modules/5.19.0-50-generic/kernel/drivers/gpio/gpio-it87.ko
version:        1.5.0
license:        GPL
description:    GPIO interface for IT87xx Super I/O chips
author:         Diego Elio Pettenò <flameeyes@flameeyes.eu>
srcversion:     BF1E1DA11ED46916F0525B3
depends:
retpoline:      Y
name:           gpio_it87
vermagic:       5.19.0-50-generic SMP preempt mod_unload modversions
parm:           force_id:Override the detected device ID (ushort)
```

Once the **gpio_it87** driver has been probed, the gpiochip interfaces /sys/class/gpio/gpiochip* and /sys/class/gpio/gpio* are created by the driver.

Example

```
# cat /sys/class/gpio/gpiochip698/label
gpio_it87
# cat /sys/class/gpio/gpio699/value
0
```

Thus, by read/write the gpio value, user can get/set the super IO gpio value. For details, see drivers/gpio/gpio-it87.c.

NOTE

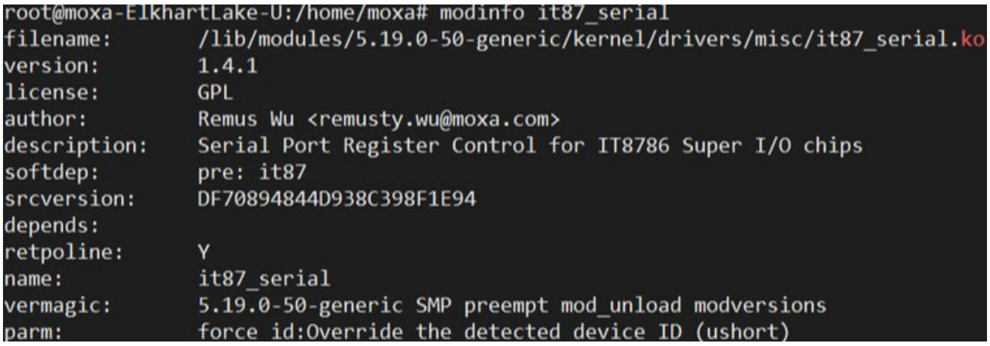
If the Linux kernel version ≥ 5.x, the **libgpiod** library is used by default to set/get the gpio value.

For Linux kernel version ≤ 3.x, the **sys class gpio** is used by default to set/get the gpio value.

moxa-it87-serial-driver

IT87xx Super I/O chips support six standard serial ports and **RS485 automatic direction control (ADDC)** mode. This driver provide an interface under misc device for controlling serial register.

Kernel module information



```
root@moxa-ElkhartLake-U:/home/moxa# modinfo it87_serial
filename:       /lib/modules/5.19.0-50-generic/kernel/drivers/misc/it87_serial.ko
version:        1.4.1
license:        GPL
author:         Remus Wu <remusty.wu@moxa.com>
description:    Serial Port Register Control for IT8786 Super I/O chips
softdep:        pre: it87
srcversion:     DF70894844D938C398F1E94
depends:
retpoline:      Y
name:           it87_serial
vermagic:       5.19.0-50-generic SMP preempt mod_unload modversions
parm:           force_id:Override the detected device ID (ushort)
```

Once the **it87_serial** driver has been probed, the /sys/class/misc/it87_serial/serial[p] interface is created by the driver.

Example

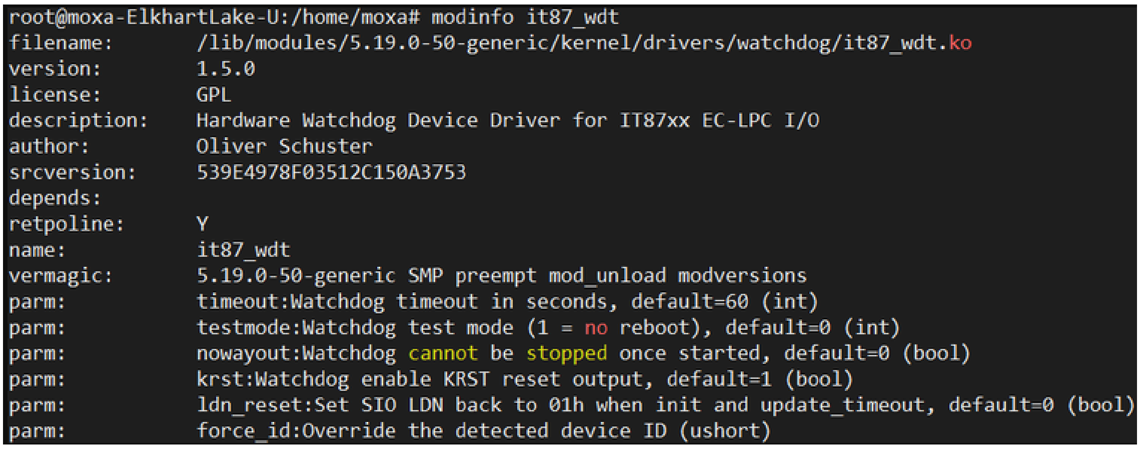
```
# cat /sys/class/misc/it87_serial/serial1/serial1_rs485
0
```

If 0 is returned, the RS-485 automatic direction control (ADDC) is disabled. If 1 is returned, the ADDC mode is enabled. The **UART RS-485 ADDC state** selection is imported into the **mx-uart-ctl** utility.

moxa-it87-wdt-driver

Watchdog timer driver for ITE IT87xx environment control. The moxa-it87-wdt-driver is based on Linux kernel drivers/watchdog/it87_wdt.c river, and add kernel parameters to support Moxa platform's hardware design.

Kernel module information



```
root@moxa-ElkhartLake-U:/home/moxa# modinfo it87_wdt
filename:       /lib/modules/5.19.0-50-generic/kernel/drivers/watchdog/it87_wdt.ko
version:        1.5.0
license:        GPL
description:    Hardware Watchdog Device Driver for IT87xx EC-LPC I/O
author:         Oliver Schuster
srcversion:     539E4978F03512C150A3753
depends:
retpoline:      Y
name:           it87_wdt
vermagic:       5.19.0-50-generic SMP preempt mod_unload modversions
parm:           timeout:Watchdog timeout in seconds, default=60 (int)
parm:           testmode:Watchdog test mode (1 = no reboot), default=0 (int)
parm:           nowayout:Watchdog cannot be stopped once started, default=0 (bool)
parm:           krst:Watchdog enable KRST reset output, default=1 (bool)
parm:           ldn_reset:Set SIO LDN back to 01h when init and update_timeout, default=0 (bool)
parm:           force_id:Override the detected device ID (ushort)
```

The watchdog device node /dev/watchdog0 is created by the it87_wdt driver.

The x86 Linux SDK Wizard will by default setup the watchdog daemon configuration file /etc/watchdog.conf and enable the service for specific Linux distributions.

The default timeout of watchdog device is 60 seconds (maximum is 65535 seconds). If you want to change timeout value, edit the watchdog daemon config file /etc/watchdog.conf.

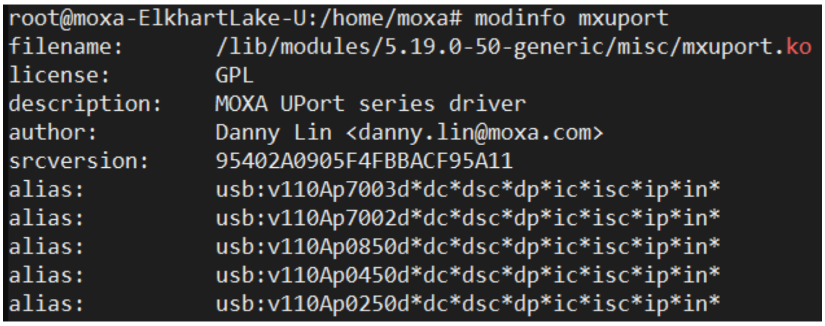
Example: watchdog timeout after 300 seconds:

```
watchdog-timeout = 300
```

moxa-mxuport-driver

The purpose of moxa-mxuport-driver is MOXA UPort series driver. This driver remains traditional serial device properties and only dial-in ports will be created.

Kernel module information



```
root@moxa-ElkhartLake-U:/home/moxa# modinfo mxuport
filename:       /lib/modules/5.19.0-50-generic/misc/mxuport.ko
license:        GPL
description:    MOXA UPort series driver
author:         Danny Lin <danny.lin@moxa.com>
srcversion:     95402A0905F4FBBACF95A11
alias:          usb:v110Ap7003d*dc*dsc*dp*ic*isc*ip*in*
alias:          usb:v110Ap7002d*dc*dsc*dp*ic*isc*ip*in*
alias:          usb:v110Ap0850d*dc*dsc*dp*ic*isc*ip*in*
alias:          usb:v110Ap0450d*dc*dsc*dp*ic*isc*ip*in*
alias:          usb:v110Ap0250d*dc*dsc*dp*ic*isc*ip*in*
```

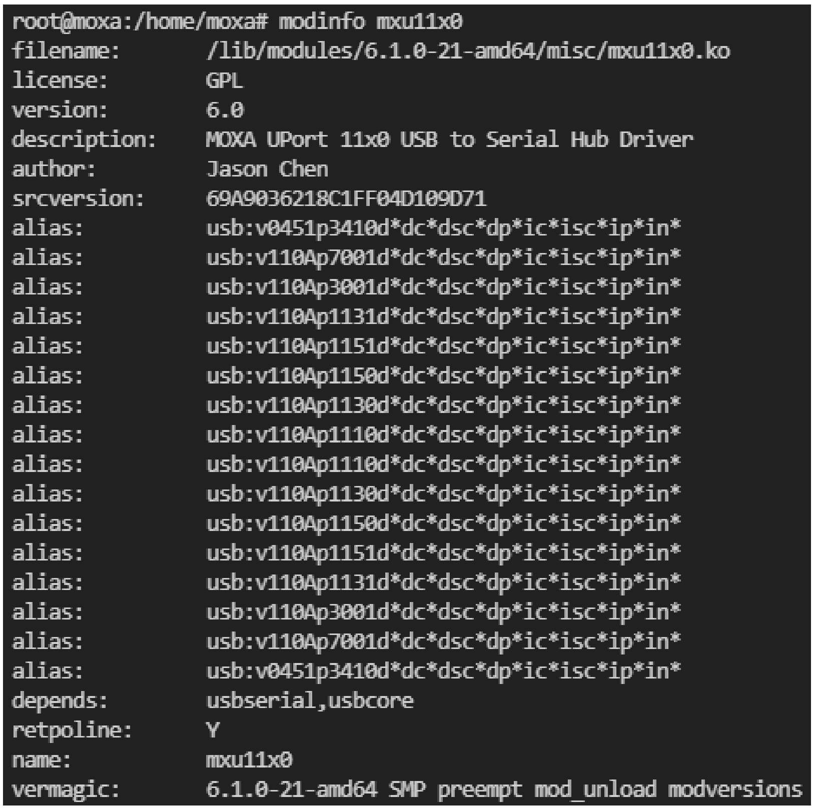
The device name for each serial port is /dev/ttyUSBxx, where xx is a sequence number maintained by the USB subsystem.

The mxuport UART mode selection has been imported into the **mx-uart-ctl** utility.

moxa-mxu11x0-driver

The purpose of moxa-mxu11x0-driver is Moxa UPort 11x0 USB to Serial Hub driver. The driver can be used in the Linux kernel with the usbcore and usbserial modules.

Kernel module information



```
root@moxa:/home/moxa# modinfo mxu11x0
filename:       /lib/modules/6.1.0-21-amd64/misc/mxu11x0.ko
license:        GPL
version:        6.0
description:    MOXA UPort 11x0 USB to Serial Hub Driver
author:         Jason Chen
srcversion:     69A9036218C1FF04D109D71
alias:          usb:v0451p3410d*dc*dsc*dp*ic*isc*ip*in*
alias:          usb:v110Ap7001d*dc*dsc*dp*ic*isc*ip*in*
alias:          usb:v110Ap3001d*dc*dsc*dp*ic*isc*ip*in*
alias:          usb:v110Ap1131d*dc*dsc*dp*ic*isc*ip*in*
alias:          usb:v110Ap1151d*dc*dsc*dp*ic*isc*ip*in*
alias:          usb:v110Ap1150d*dc*dsc*dp*ic*isc*ip*in*
alias:          usb:v110Ap1130d*dc*dsc*dp*ic*isc*ip*in*
alias:          usb:v110Ap1110d*dc*dsc*dp*ic*isc*ip*in*
alias:          usb:v110Ap1110d*dc*dsc*dp*ic*isc*ip*in*
alias:          usb:v110Ap1130d*dc*dsc*dp*ic*isc*ip*in*
alias:          usb:v110Ap1150d*dc*dsc*dp*ic*isc*ip*in*
alias:          usb:v110Ap1151d*dc*dsc*dp*ic*isc*ip*in*
alias:          usb:v110Ap1131d*dc*dsc*dp*ic*isc*ip*in*
alias:          usb:v110Ap3001d*dc*dsc*dp*ic*isc*ip*in*
alias:          usb:v110Ap7001d*dc*dsc*dp*ic*isc*ip*in*
alias:          usb:v0451p3410d*dc*dsc*dp*ic*isc*ip*in*
depends:        usbserial,usbcore
retpoline:      Y
name:           mxu11x0
vermagic:       6.1.0-21-amd64 SMP preempt mod_unload modversions
```

The device name for each serial port is /dev/ttyUSBxx which xx is a sequence number maintained by USB subsystem.

The mxu11x0 UART mode selection has been imported into **mx-uart-ctl** utility.

moxa-sdhci-pci-driver

NOTE

This driver is only available on BXP-A100 with Debian 11 for resolving the SD card detection issue.

The purpose of **moxa-sdhci-pci-driver** is SDHCI on PCI bus interface driver.

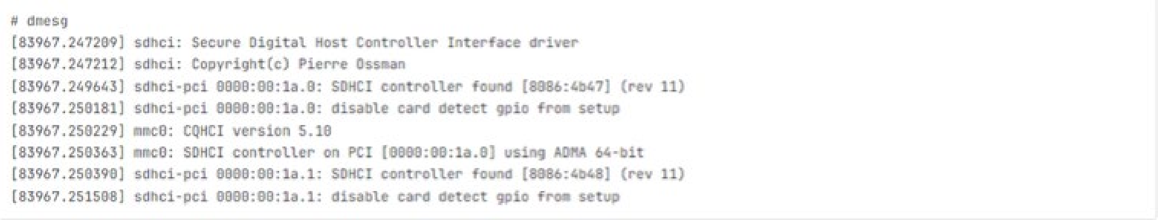
Due to the SD host controller communicates with the CPU via SDIO, it would not initialize successfully on **Debian 11**.

Thus to resolve this issue, this driver add module parameter (**`enable_probe_cd_gpio`**) to determine probe card detect gpio or not.

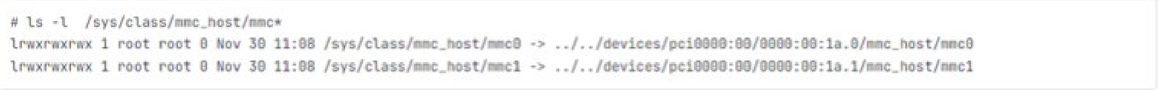
```
modprobe sdhci_pci enable_probe_cd_gpio=0
```

Or add modprobe configuration file: **`/lib/modprobe.d/sdhci-pci-option.conf`**

Kernel message and SD card interface:



```
# dmesg
[83967.247209] sdhci: Secure Digital Host Controller Interface driver
[83967.247212] sdhci: Copyright(c) Pierre Ossman
[83967.249643] sdhci-pci 0000:00:1a.0: SDHCI controller found [8086:4b47] (rev 11)
[83967.250181] sdhci-pci 0000:00:1a.0: disable card detect gpio from setup
[83967.250229] mmc0: CQHCI version 5.10
[83967.250363] mmc0: SDHCI controller on PCI [0000:00:1a.0] using ADMA 64-bit
[83967.250390] sdhci-pci 0000:00:1a.1: SDHCI controller found [8086:4b48] (rev 11)
[83967.251508] sdhci-pci 0000:00:1a.1: disable card detect gpio from setup
```

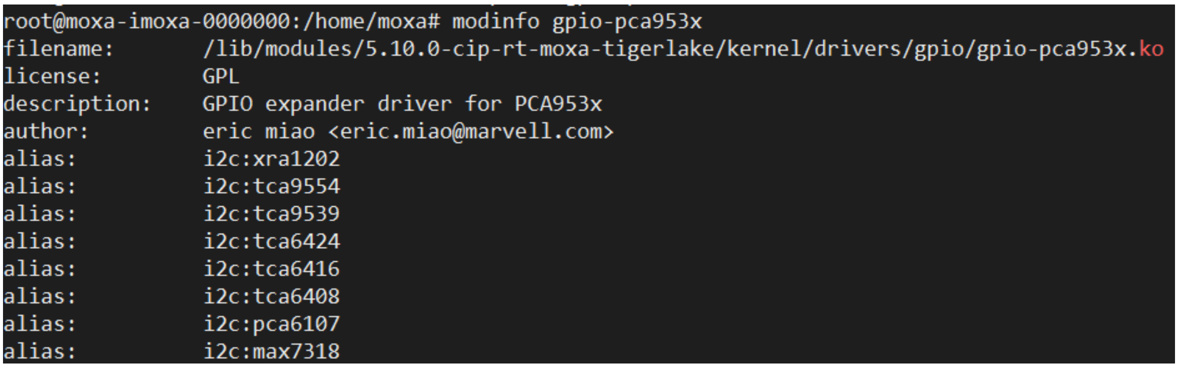


```
# ls -l  /sys/class/mmc_host/mmc*
lrwxrwxrwx 1 root root 0 Nov 30 11:08 /sys/class/mmc_host/mmc0 -> ../../devices/pci0000:00/0000:00:1a.0/mmc_host/mmc0
lrwxrwxrwx 1 root root 0 Nov 30 11:08 /sys/class/mmc_host/mmc1 -> ../../devices/pci0000:00/0000:00:1a.1/mmc_host/mmc1
```

moxa-gpio-pca953x-driver

This driver is for PCA953x 4/8/16/24/40 bit I/O ports control.

Kernel module information



```
root@moxa-imoxa-0000000:/home/moxa# modinfo gpio-pca953x
filename:       /lib/modules/5.10.0-cip-rt-moxa-tigerlake/kernel/drivers/gpio/gpio-pca953x.ko
license:        GPL
description:    GPIO expander driver for PCA953x
author:         eric miao <eric.miao@marvell.com>
alias:          i2c:xra1202
alias:          i2c:tca9554
alias:          i2c:tca9539
alias:          i2c:tca6424
alias:          i2c:tca6416
alias:          i2c:tca6408
alias:          i2c:pca6107
alias:          i2c:max7318
```

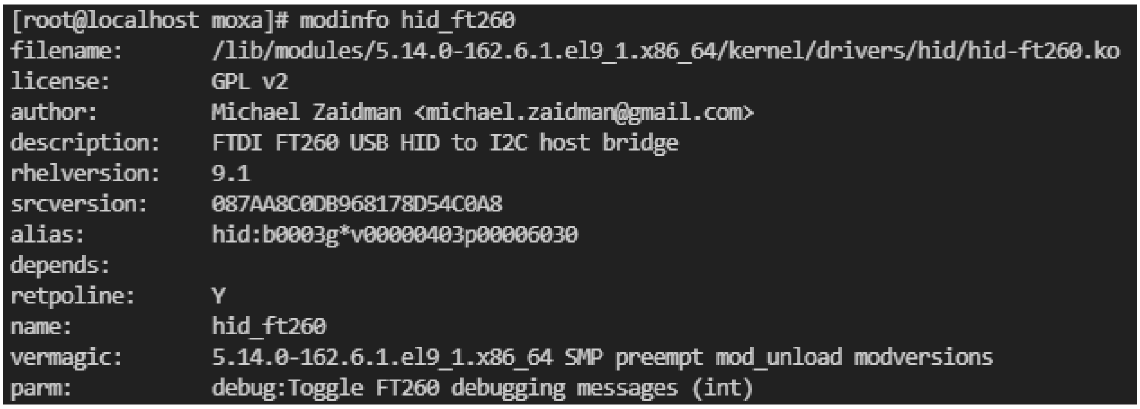
Once the gpio-pca953x driver has been probed, and bind with USB to i2c bridge (e.g. FT260 or CP2112), the gpiochip interface /sys/class/gpio/gpiochip* and /sys/class/gpio/gpio* are created by driver.

The example refers to **moxa-it87-gpio-driver** section.

moxa-hid-ft260-driver

This driver is for USB to SMBus master bridge driver on FT260.

Kernel module information



```
[root@localhost moxa]# modinfo hid_ft260
filename:       /lib/modules/5.14.0-162.6.1.el9_1.x86_64/kernel/drivers/hid/hid-ft260.ko
license:        GPL v2
author:         Michael Zaidman <michael.zaidman@gmail.com>
description:    FTDI FT260 USB HID to I2C host bridge
rhelversion:    9.1
srcversion:     087AA8C0DB968178D54C0A8
alias:          hid:b0003g*v00000403p00006030
depends:
retpoline:      Y
name:           hid_ft260
vermagic:       5.14.0-162.6.1.el9_1.x86_64 SMP preempt mod_unload modversions
parm:           debug:Toggle FT260 debugging messages (int)
```

Add Udev Rules to Rebind FT260 Device

To avoid the ft260 hid device is pre-bind to hid-generic subsystem, add udev rules to re-bind to ft260 driver.

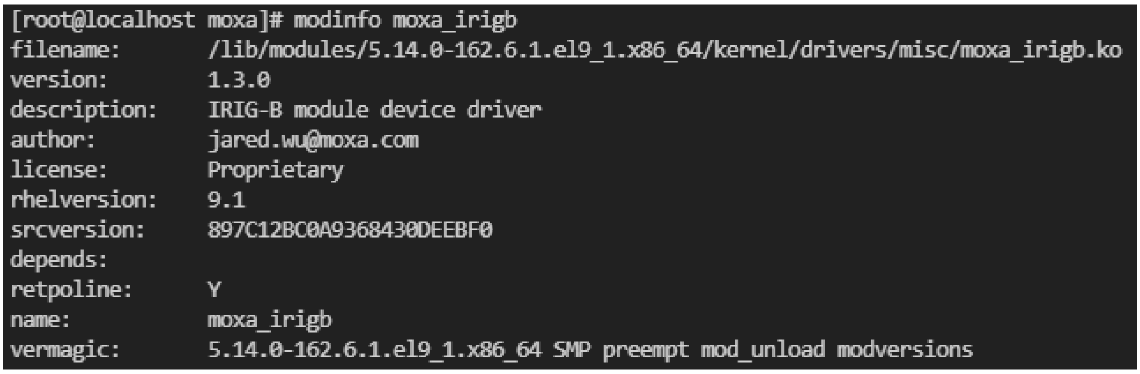
Edit /etc/udev/rules.d/11-ft260-pca9535.rules

```
ACTION=="add", KERNEL=="0003:0403:6030.*", SUBSYSTEM=="hid", DRIVERS=="hid-generic", \
RUN+="/bin/bash -c 'echo $kernel > /sys/bus/hid/drivers/hid-generic/unbind'", \
RUN+="/bin/bash -c 'echo $kernel > /sys/bus/hid/drivers/ft260/bind'"
```

moxa-irigb-driver

The IRIG-B driver is for Moxa embedded compute for controlling the IRIG-B device.

Kernel module information



```
[root@localhost moxa]# modinfo moxa_irigb
filename:       /lib/modules/5.14.0-162.6.1.el9_1.x86_64/kernel/drivers/misc/moxa_irigb.ko
version:        1.3.0
description:    IRIG-B module device driver
author:         jared.wu@moxa.com
license:        Proprietary
rhelversion:    9.1
srcversion:     897C12BC0A9368430DEEBF0
depends:
retpoline:      Y
name:           moxa_irigb
vermagic:       5.14.0-162.6.1.el9_1.x86_64 SMP preempt mod_unload modversions
```

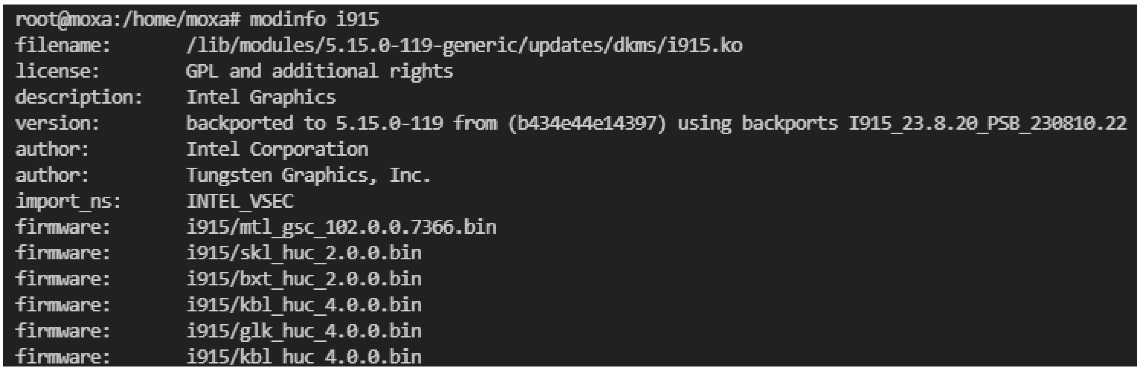
The IRIG-B driver is depends on IRIG-B Utility.

intel-gpu-i915-backports

Intel® Graphics Driver Backports for Linux® OS (intel-gpu-i915-backports)

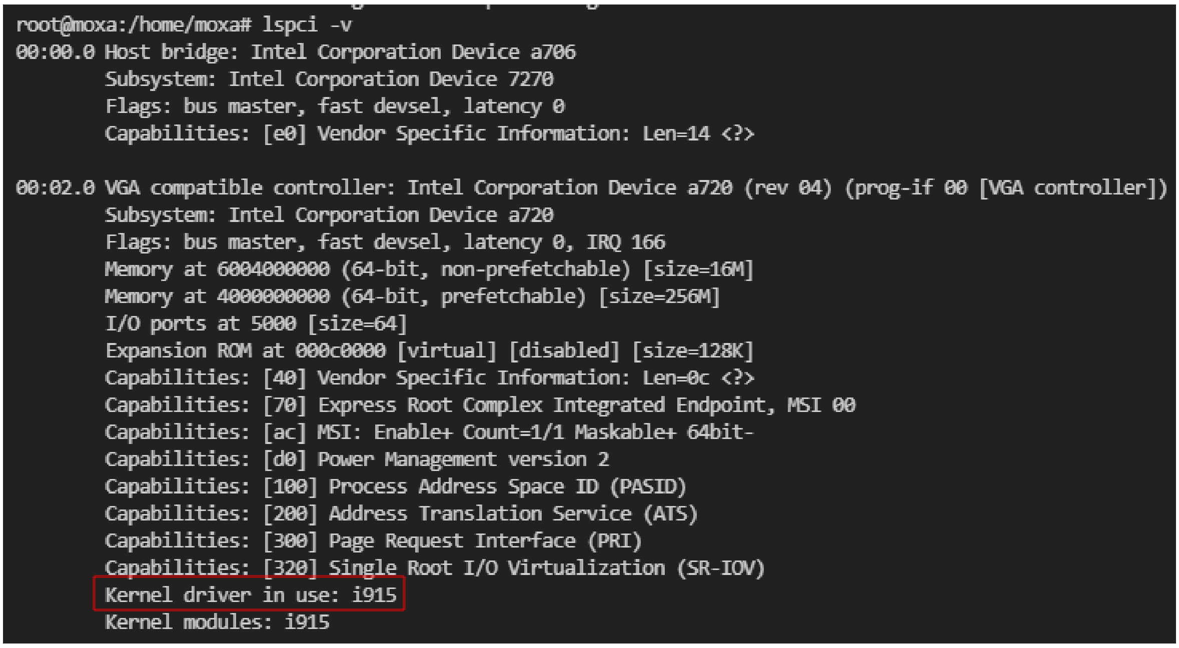
Contains the backported kernel module source code of intel GPUs on various OS distributions and LTS Kernels.

Kernel module information



```
root@moxa:/home/moxa# modinfo i915
filename:       /lib/modules/5.15.0-119-generic/updates/dkms/i915.ko
license:        GPL and additional rights
description:    Intel Graphics
version:        backported to 5.15.0-119 from (b434e44e14397) using backports I915_23.8.20_PSB_230810.22
author:         Intel Corporation
author:         Tungsten Graphics, Inc.
import_ns:      INTEL_VSEC
firmware:       i915/mtl_gsc_102.0.0.7366.bin
firmware:       i915/skl_huc_2.0.0.bin
firmware:       i915/bxt_huc_2.0.0.bin
firmware:       i915/kbl_huc_4.0.0.bin
firmware:       i915/glk_huc_4.0.0.bin
firmware:       i915/kbl_huc_4.0.0.bin
```

Use lspci -v to check i915 driver is in use



```
root@moxa:/home/moxa# lspci -v
00:00.0 Host bridge: Intel Corporation Device a706
        Subsystem: Intel Corporation Device 7270
        Flags: bus master, fast devsel, latency 0
        Capabilities: [e0] Vendor Specific Information: Len=14 <?>

00:02.0 VGA compatible controller: Intel Corporation Device a720 (rev 04) (prog-if 00 [VGA controller])
        Subsystem: Intel Corporation Device a720
        Flags: bus master, fast devsel, latency 0, IRQ 166
        Memory at 6004000000 (64-bit, non-prefetchable) [size=16M]
        Memory at 4000000000 (64-bit, prefetchable) [size=256M]
        I/O ports at 5000 [size=64]
        Expansion ROM at 000c0000 [virtual] [disabled] [size=128K]
        Capabilities: [40] Vendor Specific Information: Len=0c <?>
        Capabilities: [70] Express Root Complex Integrated Endpoint, MSI 00
        Capabilities: [ac] MSI: Enable+ Count=1/1 Maskable+ 64bit-
        Capabilities: [d0] Power Management version 2
        Capabilities: [100] Process Address Space ID (PASID)
        Capabilities: [200] Address Translation Service (ATS)
        Capabilities: [300] Page Request Interface (PRI)
        Capabilities: [320] Single Root I/O Virtualization (SR-IOV)
        Kernel driver in use: i915
        Kernel modules: i915
```

Library-libgpiod

The libgpiod - C library and tools are for interacting with the **Linux GPIO character device** (gpiod stands for GPIO device).

Because the GPIO sysfs interface is deprecated in the **Linux kernel 4.8,** user space should use the character device instead. This library encapsulates the ioctl calls and data structures behind a straightforward API.

The new character device interface guarantees all allocated resources are freed after closing the device file descriptor and adds several new features that are not present in the obsolete sysfs interface.

One device file per gpiochip

```
/dev/gpiochip0, /dev/gpiochip1, …, /dev/gpiochipX
```

Usage

```
There are currently six command-line tools available:

* gpiodetect - list all gpiochips present on the system, their names, labels
               and number of GPIO lines

* gpioinfo   - list all lines of specified gpiochips, their names, consumers,
               direction, active state and additional flags

* gpioget    - read values of specified GPIO lines

* gpioset    - set values of specified GPIO lines, potentially keep the lines
               exported and wait until timeout, user input or signal

* gpiofind   - find the gpiochip name and line offset given the line name

* gpiomon    - wait for events on GPIO lines, specify which events to watch,
               how many events to process before exiting or if the events
               should be reported to the console
```

Example

```
# Read the value of a single GPIO line.
$ gpioget gpiochip1 23
0

# Read two values at the same time. Set the active state of the lines to low.
$ gpioget --active-low gpiochip1 23 24
1 1

# Set the value of a single line, then exit immediately.
# This is useful for floating pins.
$ gpioset gpiochip1 23=1
```

5. Basic Linux Concepts

The section introduces basic Linux concepts, like x86 secure boot, IO interfaces, TPM2 module, SD card slot mounting, Linux PTP (IEEE 1588), etc.

To provide skills and basic information for newcomers to learn more about Linux.

Mounting SD Card Slot on BXP/DRP Series

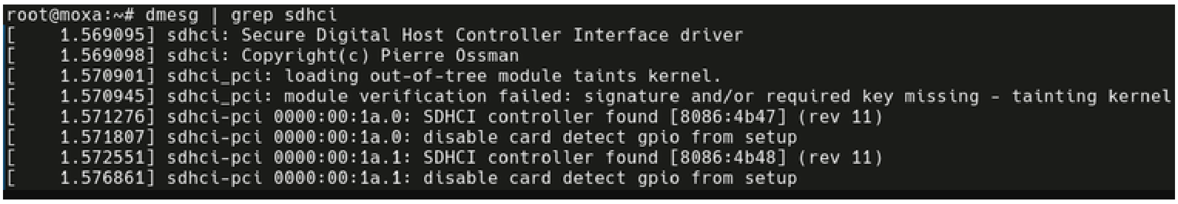
The **BXP** and **DRP** series support one SD card slot (SD 3.0 interface (SDHC/SDXC)).

The differences of hardware design between these model are:

- **BXP-A100:** SDIO interface
- **BXP-C100/DRP-A100/DRP-C100:** USB to SD Bridge IC (USB2244)

Make sure your SD card is inserted into the SD card slot on your computer, the kernel message should be shown:

For **BXP-A100**:



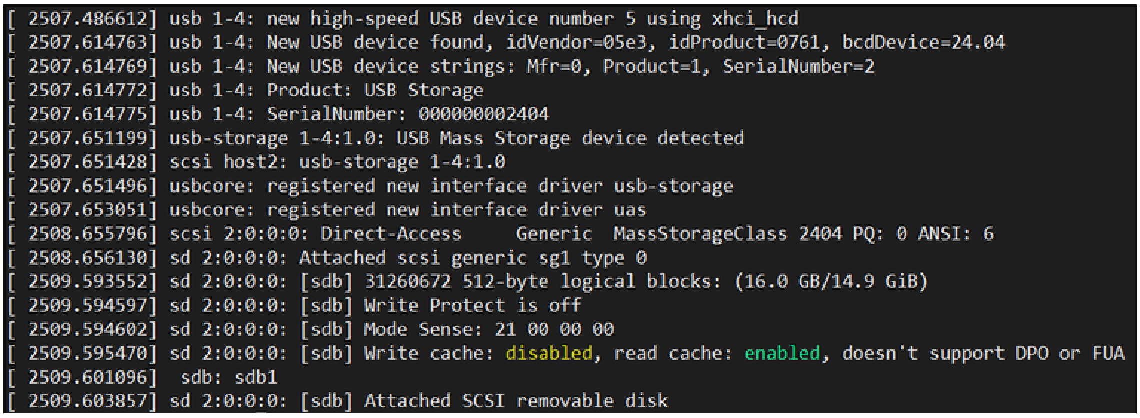
```
root@moxa:~# dmesg | grep sdhci
[    1.569095] sdhci: Secure Digital Host Controller Interface driver
[    1.569098] sdhci: Copyright(c) Pierre Ossman
[    1.570901] sdhci_pci: loading out-of-tree module taints kernel.
[    1.570945] sdhci_pci: module verification failed: signature and/or required key missing - tainting kernel
[    1.571276] sdhci-pci 0000:00:1a.0: SDHCI controller found [8086:4b47] (rev 11)
[    1.571807] sdhci-pci 0000:00:1a.0: disable card detect gpio from setup
[    1.572551] sdhci-pci 0000:00:1a.1: SDHCI controller found [8086:4b48] (rev 11)
[    1.576861] sdhci-pci 0000:00:1a.1: disable card detect gpio from setup
```

To **mount** the SD Card:

The block devices /dev/mmcblk1, the block device is created from sdhci driver.

Then, user can create a mount point on directory (e.g. /mnt): sudo mount /dev/mmcblk1p1 /mnt

For **BXP-C100/DRP-A100/DRP-C100**:



```
[ 2507.486612] usb 1-4: new high-speed USB device number 5 using xhci_hcd
[ 2507.614763] usb 1-4: New USB device found, idVendor=05e3, idProduct=0761, bcdDevice=24.04
[ 2507.614769] usb 1-4: New USB device strings: Mfr=0, Product=1, SerialNumber=2
[ 2507.614772] usb 1-4: Product: USB Storage
[ 2507.614775] usb 1-4: SerialNumber: 000000002404
[ 2507.651199] usb-storage 1-4:1.0: USB Mass Storage device detected
[ 2507.651428] scsi host2: usb-storage 1-4:1.0
[ 2507.651496] usbcore: registered new interface driver usb-storage
[ 2507.653051] usbcore: registered new interface driver uas
[ 2508.655796] scsi 2:0:0:0: Direct-Access     Generic  MassStorageClass 2404 PQ: 0 ANSI: 6
[ 2508.656130] sd 2:0:0:0: Attached scsi generic sg1 type 0
[ 2509.593552] sd 2:0:0:0: [sdb] 31260672 512-byte logical blocks: (16.0 GB/14.9 GiB)
[ 2509.594597] sd 2:0:0:0: [sdb] Write Protect is off
[ 2509.594602] sd 2:0:0:0: [sdb] Mode Sense: 21 00 00 00
[ 2509.595470] sd 2:0:0:0: [sdb] Write cache: disabled, read cache: enabled, doesn't support DPO or FUA
[ 2509.601096]  sdb: sdb1
[ 2509.603857] sd 2:0:0:0: [sdb] Attached SCSI removable disk
```

The block devices /dev/sdX, where "X" is a letter indicating the specific device (e.g., /dev/sdb, /dev/sdc, etc.).

Then, user can create a mount point on directory (e.g. /mnt): sudo mount /dev/sdX /mnt

Secure Boot

The **UEFI Secure Boot** is a security feature that has been widely adopted in modern computer systems, especially those running Windows and some Linux distributions.

Its primary purpose is to ensure the integrity and authenticity of the operating system and bootloader during the system boot process, protecting the system against boot-time malware and other unauthorized software.

Secure Boot Purpose

Secure Boot is designed to prevent the loading of malicious software, such as rootkits and bootkits, during the boot process.

It does this by ensuring that only trusted and digitally **signed** bootloaders and OS kernels are executed.

Thus, if user loads **unsigned** bootloaders and OS kernels on target Linux distributions when UEFI secure boot has been enabled on BIOS menu, the boot process or kernel modules should be failed due to unauthorized policy.

Operating System Support

User can be considered to refer to the following website links for more UEFI secure boot information.

- Debian Secure Boot
- Ubuntu Secure Boot
- RedHat Secure Boot

Linux PTP (IEEE 1588)

The **Precision Time Protocol (PTP)** is a protocol used to synchronize clocks throughout a computer network. PTP provides higher precision and faster synchronization than NTP even without hardware support. With hardware support, sub-microsecond accuracy can be expected.

Whereas NTP is intended for WAN use, PTP is designed for LAN environments and makes use of UDP multicast.

Available LAN chip

- Intel I210 (driver: ibg)
- Intel I219 (driver: e1000e)

Debian Linuxptp package

Linuxptp package is an implementation of the Precision Time Protocol (PTP) according to IEEE standard 1588 for Debian Linux. Features include:

1. Support for hardware and software time stamping via the Linux SO_TIMESTAMPING socket option.
2. Support for the Linux PTP Hardware Clock (PHC) subsystem by using the clock_gettime family of calls, including the new clock_adjtimex system call
3. Implementation of **Boundary Clock (BC)** and **Ordinary Clock (OC)**
4. Transport over UDP/IPv4, UDP/IPv6, and raw Ethernet (Layer 2)
5. Support for IEEE 802.1AS-2011 in the role of end station

Debian phc2sys program

`phc2sys` is a program which synchronizes two or more clocks in the system. Typically, it is used to synchronize the system clock to a PTP hardware clock (PHC), which itself is synchronized by the ptp4l(8) program. See manpage for more information.

- **Prerequisite**
  - Install **Debian 11** or later version
  - Install **Linuxptp** package: apt update && apt install linuxptp
  - Stop and disable systemd time sync daemon service to avoid some unexpected operations: systemctl stop systemd-timesyncd && systemctl disable systemd-timesyncd

Example for Linux PTP setting up

Ordinary Clock (OC) Mode

Set as **OC master** mode: Layer 2, P2P mode, peer delay mechanism

```
# Assume A side interface device is 'enp4s0'
ip link set dev enp4s0 up
ptp4l -m -2 -P -i enp4s0
```

Set as **OC slave** mode: Layer 2, P2P mode, peer delay mechanism

```
# Assume B side interface device is 'enp5s0'
ip link set dev enp5s0 up
ptp4l -m -2 -P -s -i enp5s0
# or with log: ptp4l -m -2 -s -P -i enp5s0 2>&1 | tee $(date +%Y%m%d%H%M%S.log)

# use phc2sys to sync sys clock for 10Hz
phc2sys -a -m -r -R 10
```

Boundary Clock (BC) Mode

Set as **BC mode** host

- clock_type Specifies the kind of PTP clock. Valid values are "OC" for ordinary clock, "BC" for boundary clock, "P2P_TC" for peer to peer transparent clock, and "E2E_TC" for end to end transparent clock. An multi-port ordinary clock will automatically be configured as a boundary clock. The default is "OC".
- boundary_clock_jbod When running as a **boundary clock** (that is, when more than one network interface is configured), ptp4l performs a sanity check to make sure that all of the ports share the same hardware clock device. This option allows ptp4l to work as a boundary clock using "just a bunch of devices" that are not synchronized to each other. For this mode, the collection of clocks must be synchronized by an external program, for example phc2sys(8) in "automatic" mode. The default is 0 (disabled).

Example for BC mode

```
# For example, edit config file 'bc.cfg'
# and assume 'enp12s0' and 'enp4s0' are connected network interface
[global]
sanity_freq_limit        0
step_threshold           0.000002
tx_timestamp_timeout     10
logMinPdelayReqInterval  0
logSyncInterval          0
logAnnounceInterval      0
announceReceiptTimeout   3
syncReceiptTimeout       2
twoStepFlag              1
summary_interval         0
clock_type               BC
priority1                128
priority2                127
delay_mechanism          P2P

[enp12s0]
boundary_clock_jbod          1
network_transport            L2
fault_reset_interval         0

[enp4s0]
boundary_clock_jbod          1
network_transport            L2
fault_reset_interval         0
```

```
# run the ptp4l procedure
ip link set dev enp12s0 up
ip link set dev enp4s0 up
ptp4l -m -f bc.cfg

# use phc2sys to sync sys clock for 10Hz
phc2sys -a -m -r -R 10
```

On OC **Grandmaster**

```
# assume interface is enp5s0
ip link set dev enp5s0 up
ptp4l -2 -m -P -i enp5s0
```

On OC **Slave**

```
# assume interface is enp4s0
ip link set dev enp4s0 up
ptp4l -2 -m -s -P -i enp4s0
# with log: ptp4l -2 -m -s -P -i enp4s0 2>&1 | tee $(date +%Y%m%d%H%M%S.log)
```

Transparent Clock (TC) Mode

Set TC mode host

```
# For example, edit config file 'tc.cfg'
# and assume 'enp12s0' and 'enp4s0' are connected network interface
[global]
priority1               254
priority2               253
free_running            1
freq_est_interval       3
tc_spanning_tree        1
clock_type              P2P_TC
network_transport       L2
delay_mechanism         P2P

[enp12s0]
egressLatency           0
ingressLatency          0
delay_mechanism         P2P
network_transport       L2

[enp4s0]
egressLatency           0
ingressLatency          0
delay_mechanism         P2P
network_transport       L2

# run the ptp4l procedure
ip link set dev enp12s0 up
ip link set dev enp4s0 up
ptp4l -m -f tc.cfg

# use phc2sys to sync sys clock between master & slave for 10Hz
# -c Specify the slave clock by device (e.g. /dev/ptp1) or interface (e.g.
eth1)
# -s Specify the master clock by device (e.g. /dev/ptp0) or interface (e.g.
eth0)
phc2sys -s enp12s0 -c enp4s0 -O 0 -R 10 -m
```

As OC Grandmaster

```
# assume interface is enp5s0
ip link set dev enp5s0 up
ptp4l -2 -m -P -i enp5s0
```

As OC Slave

```
# assume interface is enp4s0
ip link set dev enp4s0 up
ptp4l -2 -m -s -P -i enp4s0

# use phc2sys to sync sys clock for 10Hz on slve
phc2sys -a -m -r -R 10
```

6. Troubleshooting

The troubleshooting section provides fundamental skills for system logging, debugging, the debug of Moxa x86 SDK Wizard and issues tracing.

How to Print Kernel Message from Linux Environment

The dmesg command is used to display the kernel ring buffer, which contains messages related to the kernel and hardware events.

It's a useful tool for troubleshooting hardware-related issues, monitoring system-level events and diagnosing hardware issues.

To simply view the kernel ring buffer, run the following command: *dmesg*

You can save the output of dmesg to a file for further analysis. For instance, to save the log to a file named kernel.log, use the following command:

```
# save kernel message to log
dmesg >kernel.log

# or simply to save the error and warninglevel log:
dmesg --level=err,warn > kernel_err_warn.log
```

How to Collect Systems Logs from Linux Environment

The following procedure describes the collecting of log files. Log files in the /var/log directory.

Archive and compress all log files and put them in /tmp

```
tar czvf /tmp/varlog.tar.gz /var/log/*.log.*
```

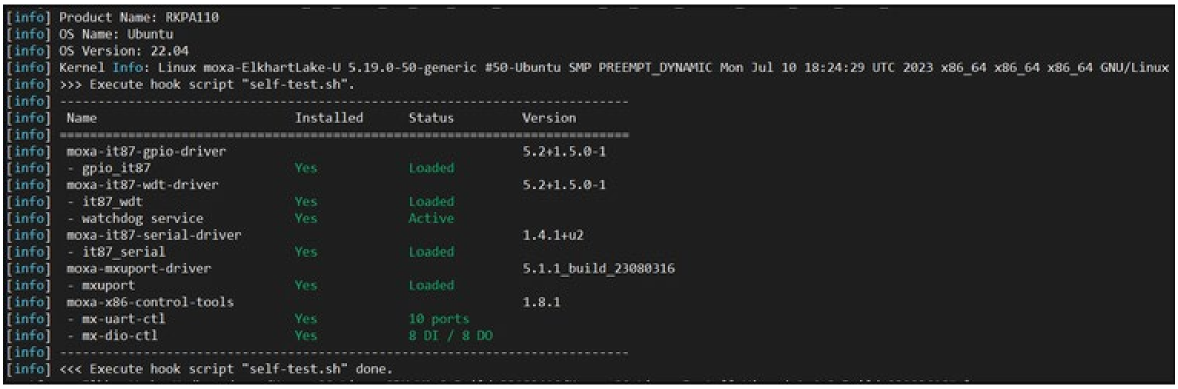
The output file **`/tmp/varlog.tar.gz`** can be transferred for debugging usage.

How to Get Installation Logs from Moxa x86 Linux SDK Install Wizard

Moxa x86 Linux SDK provides **self-test** for diagnosing the status of drivers and tools after installation. To simply see the log, run the following command:

```
./install.sh --selftest
```

Then the self test cases will check the SDK status and print on terminal, for example:



```
[info] Product Name: RKPA110
[info] OS Name: Ubuntu
[info] OS Version: 22.04
[info] Kernel Info: Linux moxa-ElkhartLake-U 5.19.0-50-generic #50-Ubuntu SMP PREEMPT_DYNAMIC Mon Jul 10 18:24:29 UTC 2023 x86_64 x86_64 x86_64 GNU/Linux
[info] >>> Execute hook script "self-test.sh".
[info] ---------------------------------------------------------------------------
[info]  Name                          Installed     Status        Version
[info] ===========================================================================
[info]  moxa-it87-gpio-driver                                     5.2+1.5.0-1
[info]  - gpio_it87                   Yes           Loaded
[info]  moxa-it87-wdt-driver                                      5.2+1.5.0-1
[info]  - it87_wdt                    Yes           Loaded
[info]  - watchdog service            Yes           Active
[info]  moxa-it87-serial-driver                                   1.4.1+u2
[info]  - it87_serial                 Yes           Loaded
[info]  moxa-mxuport-driver                                       5.1.1_build_23080316
[info]  - mxuport                     Yes           Loaded
[info]  moxa-x86-control-tools                                    1.8.1
[info]  - mx-uart-ctl                 Yes           10 ports
[info]  - mx-dio-ctl                  Yes           8 DI / 8 DO
[info] ---------------------------------------------------------------------------
[info] <<< Execute hook script "self-test.sh" done.
```

For further, the log of installation is also created on
Moxa_x86_Linux_Install_Wizard_<version>_Build_<build_date>/install.log

User can consider to view the log file and check issues.

How to Get Hardware Information on Host

IOS exports the hardware information on **DMI** (Desktop Management Interface) table.

Linux **dmidecode** is a tool for dumping a computer DMI (some say **SMBIOS**) table contents in a human-readable format. This table contains a description of the system's hardware components, as well as other useful pieces of information such as serial numbers and BIOS revision.

Install dmidecode Package

- Ubuntu/Debian: sudo apt-get install dmidecode
- RHEL: sudo yum install dmidecode

Example

[Get model name and hardware version]

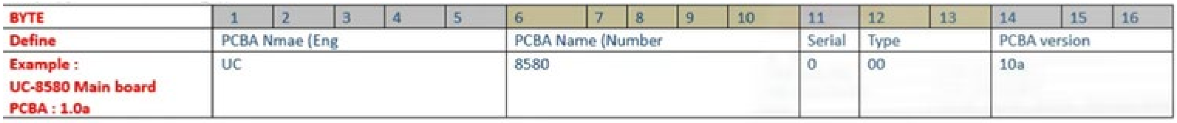
The Option 1 (or Option 2) displays the 16 bytes information, for example: RKP A110000091

RKP A110000091 means

- PCBA name = RKP
- PCBA number = A110
- PCBA serial = 0
- PCBA type = 00
- PCBA hw version = 091 (v0.91)

How to get information from dmitable

```
# dmidecode -t 12
Handle 0x0021, DMI type 12, 5 bytes
System Configuration Options
        Option 1:   RKP A110000091
        Option 2:
        Option 3:
...
```



BYTE	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Define	PCBA Nmae (Eng					PCBA Name (Number					Serial	Type		PCBA version		
Example : UC-8580 Main board PCBA : 1.0a	UC					8580					0	00		10a		

[Get current BIOS version]

```
# dmidecode -t bios
BIOS Information
        Vendor: INSYDE Corp.
        Version: V1.0.0S04
        Release Date: 05/15/2023
        Address: 0xE0000
        Runtime Size: 128 kB
        ROM Size: 10 MB
...
```

[Get memory and processor hardware information]

```
# dmidecode -t memory
Physical Memory Array
        Location: System Board Or Motherboard
        Use: System Memory
        Error Correction Type: None
        Maximum Capacity: 16 GB
        Error Information Handle: Not Provided
        Number Of Devices: 2
...
# sudo dmidecode -t processor
Processor Information
        Socket Designation: U3E1
        Type: Central Processor
        Family: Other
        Manufacturer: Intel(R) Corporation
        ID: 61 06 09 00 FF FB EB BF
        Version: Intel Atom(R) x6425E Processor @ 2.00GHz
        Voltage: 1.1 V
        External Clock: 100 MHz
...
```

7. Appendix

The License/Commercial-Use of Linux Distributions

A Linux distribution is a version of the Linux operating system that includes the Linux kernel, system utilities, libraries, and additional software and applications. Linux distributions are created by various organizations, communities, and individuals, each tailoring the operating system to meet specific needs and preferences.

Linux distribution include:

Debian

Debian is a free and open-source operating system, and its intellectual property rights policy is based on a commitment to free software principles. Debian adheres to a set of guidelines and policies outlined in the Debian Free Software Guidelines (DFSG). The DFSG defines the criteria that software must meet to be considered "free" in the context of Debian.

Commercial use:

Free redistribution.

The license of a Debian component **may not restrict** any party from selling or giving away the software as a component of an aggregate software distribution containing programs from several different sources. The license may not require a royalty or other fee for such sale.

https://wiki.debian.org/DebianFreeSoftwareGuidelines

https://www.debian.org/trademark

https://wiki.debian.org/ProposedTrademarkPolicy

Ubuntu

Ubuntu is built by Canonical and the Ubuntu community. We share access rights owned by Canonical with the Ubuntu community for the purposes of discussion, development and advocacy. We recognise that most of the open source discussion and development areas are for non-commercial purposes and we therefore allow the use of Canonical IP in this context, as long as there is no commercial use and that the Canonical IP is used in accordance with this IPRights Policy.

You can modify Ubuntu for **personal** or **internal commercial** use.

You can **redistribute** Ubuntu, but only where there has been **no modification** to it.

For more Canonical's intellectual property rights policy:

https://ubuntu.com/legal/intellectual-property-policy

Red Hat Enterprise Linux (RHEL)

Red Hat Enterprise Linux (RHEL) is a **commercial** Linux distribution provided by Red Hat, Inc. It is designed for enterprise environments and comes with a subscription-based pricing model.

https://www.redhat.com/en/store/linux-platforms

https://www.redhat.com/en/about/trademark-guidelines-and-policies

https://www.redhat.com/en/about/terms-use

CentOS

The CentOS Linux and CentOS Stream distributions are compilations of software packages. Each package is governed by its own license. The CentOS Linux and CentOS Stream compilation copyright is licensed under GPLv2. To the extent you hold any copyright in the selection, coordination, or arrangement of packages making up the CentOS Linux or CentOS Stream distributions, you license that copyright under GPLv2.

https://www.centos.org/legal/licensing-policy/