Edge AI Applied Computing

SINTRONES

IBOX-602P-IP66

User Manual

Version 1.3

SINTRONES

Revision History

Version	Date	Description of Changes
1.0	2025-09-02	Initial release.
1.1	2025-11-05	Updated the power adapter description and added a Supported Camera List in Section 1.1, *Product Information*.
1.2	2025-12-12	Added Section 2.4, *Default Login Credentials*
1.3	2026-03-02	Added a note in Section 5.1.1, *Setting the GMSL Camera*, reminding users that different camera types must not be mixed.

Copyright

Disclaimer

SINTRONES® Technology Corp. shall not be liable for any incidental or consequential damages resulting from the performance or use of this product.

SINTRONES® Technology Corp. makes no representation or warranty regarding the content of this manual. Information in this manual has been carefully checked for accuracy; however, no guarantee is given to the correctness of the contents. For continuing product improvement, SINTRONES® Technology Corp. reserves the right to revise the manual or make changes to the specifications of this product at any time without notice and obligation to any person or entity regarding such change. The information contained in this manual is provided for general use by customers.

Declaration of Conformity

The CE symbol on your product indicates that it complies with the European Union (EU) directives. A Certificate of Compliance is available by contacting Technical Support. This product has passed the CE test for environmental specifications when shielded cables are used for external wiring. We recommend the use of shielded cables.

This product has been tested and found to comply with the limits for a Class A device, according to Part 15 of the FCC Rules. These limits are designed to protect reasonably against harmful interference in a residential installation. This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used by the manufacturer’s instructions, may cause harmful interference to radio communications.

Trademarks

SINTRONES® is a registered trademark of SINTRONES® Technology Corp.

The terms HDMI, HDMI High-Definition Multimedia Interface, HDMI Trade dress and the HDMI Logos are trademarks or registered trademarks of HDMI Licensing Administrator, Inc.

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

Safety Information

Read the following precautions before setting up a SINTRONES® Product.

Electrical Safety

- Disconnect the power cable from the electrical outlet to prevent shock hazards before relocating the system.
- When adding or removing devices to or from the system, ensure that the power cables for the devices are unplugged before the signal cables are connected. Disrupt all power cables from the existing system before adding a device.
- Seek professional assistance before using an adapter or extension cord. These devices could interrupt the grounding circuit.
- Before connecting or removing signal cables from the motherboard, ensure all power cables are unplugged.
- Ensure your power supply is set to the correct voltage in your area. If you are unsure of the voltage of your current electrical outlets, contact your local power company.
- If the power supply is broken, do not fix it by yourself. Contact a qualified service technician or your retailer.

Operation Safety

- Before installing the motherboard and adding devices, carefully read all the manuals in the package.
- Before using the product, ensure all cables are correctly connected and the power cables are not damaged. If you detect any damage, contact your dealer immediately.
- Keep paper clips, screws, and staples away from connectors, slots, sockets, and circuitry to avoid short circuits.
- Avoid dust, humidity, and temperature extremes. Please do not place the product in any area that may become wet.
- Place the product on a stable surface.
- Contact a qualified service technician or retailer if you encounter technical problems with the product.

Environmental Safety

- Use this product in environments with ambient temperatures between -25˚C and 65ºC.
- Do not leave this product in an environment where the storage temperature may be below -40˚C or above 80˚C. To prevent damage, the product must be used in a controlled environment.

CAUTION:
Incorrectly replacing the battery may damage this computer. Replace only with the same or equivalent recommended by SINTRONES® Technology Corp. Dispose of the used battery according to the manufacturer's instructions.

Technical Support

Please call or e-mail our customer service when you cannot fix the problems.

- TEL: +886-2-8228-0101
- FAX: +886-2-8228-0100
- E-mail: sales@sintrones.com

SINTRONES

Contents

1. Introduction

The IBOX-602P-IP66 is a rugged edge-based vision AI solution engineered for mobility and deployment in harsh environments. Its compact form factor enables easy installation and seamless integration into existing systems. Featuring GMSL-2 technology and NVIDIA® Jetson Orin™ NX platform, this edge AI computer supports uncompressed videos transmission and delivers exceptional AI compute capability for real-time image processing and data-driven decision-making at the edge. This makes it an ideal choice for machine vision applications across automotive and industrial domains, including autonomous vehicles, precision griculture, smart traffic systems, medical imaging, security inspection, and factory automation.

This chapter introduces SINTRONES® IBOX-602P-IP66 and gives an overview of its product details.

Topics in this chapter include:

1.1. Product Information

Table 1-1 Specifications

System

Module

- NVIDIA® Jetson Orin™ NX 16GB (1024 CUDA cores + 8-core ARM Cortex-A78AE CPU + 16 GB LPDDR5)
- NVIDIA® Jetson Orin™ NX 8GB (1024 CUDA cores + 6-core ARM Cortex-A78AE CPU + 8 GB LPDDR5)

Security

- Platform Security Controller (PSC), Security Engine (SE)

Watchdog

- Automatic reset for unresponsive system

Interface

Video

- 1 x HDMI® (with a cover)

Audio

- 1 x HD audio from the HDMI®

Ethernet

- 1 x 2.5GbE (Intel® I226-IT) via M12 X-code
- 1 x GbE (SoM) via M12 X-code
- PoE supports 15.4W per port (PSE Class 3) with total power budget: 32W

Camera

- 2 x GMSL-2 via FAKRA

CAN

- 1 x CAN FD via M8

USB

- 1 x USB 3.2 (with a cover)

DIO

- 3 x DI, 4 x DO (12VDC/100mA) via M8

COM

- 1 x RS-232/422/485 via M8

Mgmt. Port

- 1 x USB Type-C for system recovery (device only)

Internal Expansion

M.2

- 1 x M.2 3042/3052 Key B for WWAN w/ dual Nano SIM support
- 1 x M.2 2230 Key E for Wi-Fi/BT

Storage

Type

- 1 x M.2 2280 Key M for NVMe SSD (Pre-installed system BSP)

Power

Power Input

- DC 9-60V (nominal power input DC 12V/24V/48V) via M12 K-code

Power Protection

- OCP, surge protection, and reversed polarity protection

Power Management

- Supports Smart Power Management

RTC Battery

- High-capacity coin cell battery for RTC

Battey Backup Unit (BBU)

- Optional battery kit for up to 5 ~ 10 minutes of emergency backup time

*BBU backup time varies depending on actual overall system power consumption and battery is required to be charged before being used for system power backup.

Patent No.: M447854 (Built-in battery)

Software

Operating System

- NVIDIA® JetPack 6.2 or above (Jetson Linux and NVIDIA® development tools included)

Environmental

Operating Temp.

With built-in low-temperature heater enabled:

- Startup (default power mode, no stress testing): -40°C ~ 65°C (-40°F ~ 149°F) with 0.6m/s airflow
- Operating (2× GMSL-2 enabled, no stress testing): -35°C ~ 65°C (-31°F ~ 149°F) with 0.6m/s airflow
- Operating (stress testing, 2×GMSL-2 + 2×PoE enabled): -35°C ~ 60°C (-31°F ~ 140°F) with 0.6m/s airflow

Powered by built-in BBU (no heater):

- Operating (2× GMSL-2 enabled, 10 min, no stress testing): -10°C ~ 55°C (14°F ~131°F) with 0.6m/s airflow

Storage Temp.

- -40°C ~ 80°C (-40°F ~ 176°F)

Relative Humidity

- 10% ~ 90% RH (non-condensing)

Vibration

- IEC60068-2-64, random, 2.5G@5~500Hz, 1hr/axis
- MIL-STD-810G, Method 514.6, Procedure I, Category 4

Shock

- MIL-STD-810G, Method 516.6, Procedure I, Trucks and semi-trailers = 15G (11ms)

Certification / Compliance

- CE, FCC Class A, UKCA, E-mark, EN50155, EN45545-2 (R25)

Mechanical

Construction

- Aluminum alloy

Antenna

- 6 x SMA connector mounting hole

Mounting

- Wall mounting

Weight

- 1.78 kg (3.92 lb)

Dimensions (L x W x H)

- 150 x 138 x 66 mm (6.0 x 5.5 x 5.6 in.)

Ingress Protection

- IP66

Table 1-2 Ordering Information

Model Number	IBOX-602P-IP66-ONX16	NVIDIA® Jetson Orin™ NX 16GB / 2 x PoE / 2 x GMSL-2
	IBOX-602P-IP66-ONX8	NVIDIA® Jetson Orin™ NX 8GB / 2 x PoE / 2 x GMSL-2
Description	NVIDIA® Jetson Orin™ NX SoM w/ 2 x LAN / 2x PoE / 2x GMSL-2 / 1 x CAN FD / DC 9-60V / IP66 Edge AI Computer	

Table 1-3 Optional Accessories

Storage	M.2 2280 Key M NVMe SSD **Note:** The NVMe SSD is a mandatory accessory for this product.
Wi-Fi	M.2 2230 Key A-E Wi-Fi module
WWAN	M.2 3042/3052 WWAN modem
GNSS	M.2 2242 Key B GNSS module
Battery (BBU)	BAT-2300v2 battery kit, charging Temp. 0°C ~ 45°C, discharging Temp. -10°C ~ 60°C
Power Adapter	AC/DC 100-240V/24V 160W C14 DC plug power adapter

Note:
All items listed in the Optional Accessories table are sold separately.

Table 1-4 Supported Camera List (IBOX-602P-IP66-01 Only)

Brand	Camera Part Number	Max. No. of Camera Connections
oToBrite	oToCAM264ISP Series (2MP, Sony IMX390)	2
	oToCAM260ISP Series (5.36MP, Sony IMX490)	2
	oToCAM223 Series (2.95MP, Sony ISX031	2
e-con Systems	STURDeCAM25 (2MP, Omsemi AR0234CS)	2
	STURDeCAM31 (3MP, Sony ISX031)	2
	STURDeCAM81 (8MP, Omsemi AR0821)	2
StereoLabs	ZED X Stereo (Dual 2MP)	2
	ZED X One 4K (8.2MP)	2

*BSP versions may vary depending on supported GMSL cameras.

**All brand and product names mentioned are the property of their respective owners.

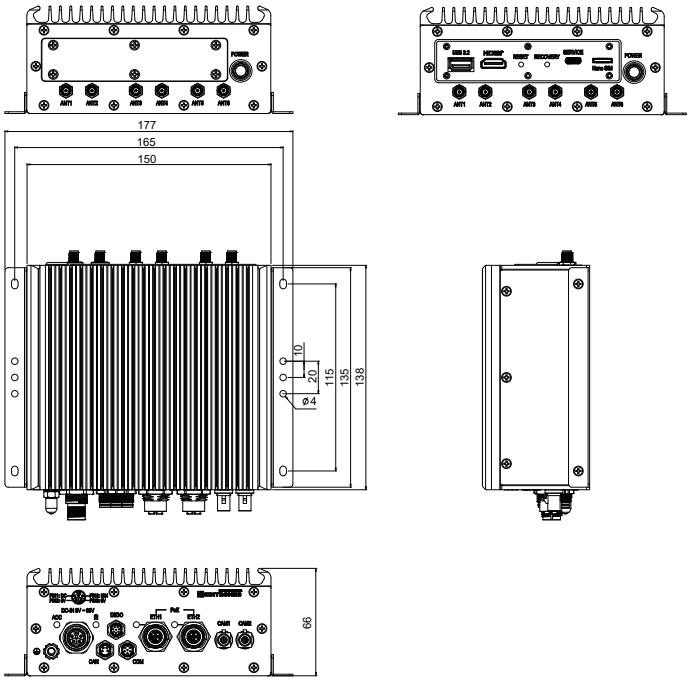
1.2. Product Photos

Figure 1-1 Front View of IBOX-602P-IP66

Figure 1-2 Rear View of IBOX-602P-IP66

1.3. Mechanical Drawings

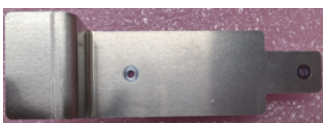
Unit : mm



1.4. Package Contents

See the following list to check if it matches your product package contents. Please contact SINTRONES® sales representatives or our sales partners if any of the items is missing.

Table 1-5 Package Contents for IBOX-602P-IP66

Item	Photo	Quantity	Description
IBOX-602P-IP66	See Product Photos *(on page 11)*	1	The edge AI computer
Power cable for IBOX-602P-IP66		1	Used to connect the computer with a M12 K-coded connector to a DC power supply
M8 4P(F) to DB9(M) cable (300 mm)		1	Used to connect the computer to a device with DB9 port for CAN FD data transfer
M8 6P(F) to DB9(M) cable (300 mm)		1	Used to connect the computer to a device with DB9 port for serial communication
M8 8P(F) to DB9(F) cable (300 mm)		1	Used to connect the computer to a device with DB9 port for DIO communication
Accessory Package			
M.2 WWAN heatsink Type 12		1	Used to transfer heat from an installed M.2 3042 or 3052 module

Item	Photo	Quantity	Description
Thermal pad (18x30x1.0T mm)		1	Used to transfer heat from an installed M.2 3042 or 3052 module
Thermal pad (30x34x1.0T mm)		1	Used to transfer heat from an installed M.2 3042 or 3052 module
Standoff screw (H75D50)		2	Served as the mounting screw-holes to secure the screws used to fasten a heatsink for an M.2 3042 or 3052 module
Screw I (Type M2.5x5L)		1	Used to fasten an M.2 module
Screw I (Type M2.5x10L)		2	Used to fasten a heatsink for an M.2 3042 or 3052 module.

1.5. Power Consumption

See the following table as the power consumption of IBOX-602P-IP66.

Table 1-6 IBOX-602P-IP66-ONX16

Mode	Input Voltage					
	12V		24V		48V	
40W (SoM Power Mode)	8.23A	98.76W	3.98A	95.52W	2.01A	96.48W
Idle	1.62A	19.44W	0.94A	22.56W	0.47A	22.56W
Standby (Ignition on)	0.74A	8.88W	0.39A	9.36W	0.24A	11.52W
Standby (Ignition off)	0.02A	0.24W	0.01A	0.24W	0.01A	0.48W

Table 1-7 IBOX-602P-IP66-ONX8

Mode	Input Voltage					
	12V		24V		48V	
40W (SoM Power Mode)	8.16A	98.76W	4.02A	96.48W	2.00A	96W
Idle	1.71A	19.44W	0.92A	22.08W	0.48A	23.04W
Standby (Ignition on)	0.68A	8.88W	0.39A	9.36W	0.22A	10.56W
Standby (Ignition off)	0.02A	0.24W	0.01A	0.24W	0.01A	0.48W

2. Getting Started

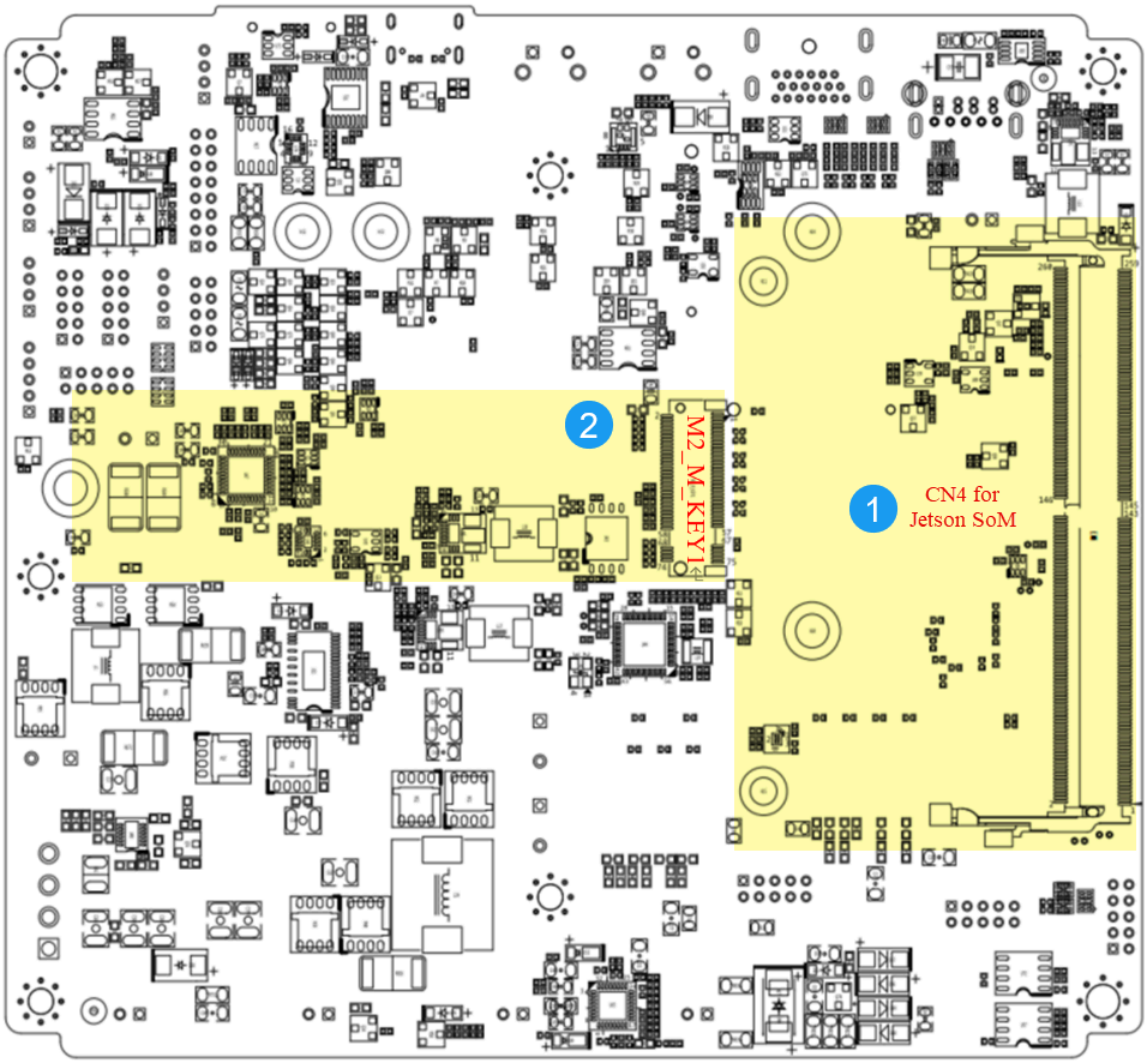
Topics in this chapter include:

2.1. SoM and SSD Info

The pre-installed NVIDIA® Jetson system on modules (SoM) and NVMe SSD are located on the bottom side of the mainboard. It is suggested to consult SINTRONES technical support for expansion requirements for Jetson SoM or an NVMe SSD.

Important:
DO NOT remove the pre-installed SoM or SSD, or install an empty one without preparing any backup image in advance.

Figure 2-1 Bottom View of Mainboard



Item	Internal Connector	Description	Specification
1	CN4 for Jetson Module	NVIDIA® Jetson module integrates CPU, GPU, and memory with pre-installed JetPack developer tools.	Information about Jetson Modules *(on page 18)*
2	M2M1	M.2 2280 Key M slot used for installing an NVMe SSD	M.2 Key M Slot *(on page 18)*

2.1.1. Information about Jetson Modules

Table 2-1 NVIDIA® Jetson Orin™ NX Modules Pre-installed in IBOX-602P-IP66

Series	Jetson Orin NX Series	
Model	Jetson Orin NX 16GB	Jetson Orin NX 8GB
GPU	1024-core NVIDIA® Ampere architecture GPU with 32 Tensor Cores	
CPU Frequency	2.0 GHz	
Power Consumption	25W	20W
CPU	8-core Arm® Cortex® - A78AE	6-core Arm® Cortex® - A78AE
DL Accelerator	2 x NVDLA v2	1 x NVDLA v2
Vision Accelerator	1 x PVA v2	
Memory	16GB 128-bit LPDDR5	8GB 128-bit LPDDR5
Storage	Supports external NVMe SSD (Pre-installed in system with BSP)	

2.1.2. M.2 Key M Slot

M.2 Key M Slot	Description
Size	NGFF 2280 / 75 Pin
Type	M.2 Key M H:8.5mm
Location	See SoM and SSD Info *(on page 17)* for the information.
Drawing	

Table 2-2 Pin Definition of M.2 Key M Slot

Pin	Signal	Pin	Signal
1	GND	2	V3P3_M2M
3	GND	4	V3P3_M2M

Pin	Signal	Pin	Signal
5	PCIE0_RX3_N	6	NC
7	PCIE0_RX3_P	8	NC
9	GND	10	NC
11	PCIE0_TX3_N	12	V3P3_M2M
13	PCIE0_TX3_P	14	V3P3_M2M
15	GND	16	V3P3_M2M
17	PCIE0_RX2_N	18	V3P3_M2M
19	PCIE0_RX2_P	20	NC
21	GND	22	NC
23	PCIE0_TX2_N	24	NC
25	PCIE0_TX2_P	26	NC
27	GND	28	NC
29	PCIE0_RX1_N	30	NC
31	PCIE0_RX1_P	32	NC
33	GND	34	NC
35	PCIE0_TX1_N	36	NC
37	PCIE0_TX1_P	38	NC
39	GND	40	JS_I2C2_SCL
41	PCIE0_RX0_N	42	JS_I2C2_SDA
43	PCIE0_RX0_P	44	M2M_ALERT
45	GND	46	NC
47	PCIE0_TX0_N	48	NC
49	PCIE0_TX0_P	50	PCIE0_RST
51	GND	52	PCIE0_CLKREQ
53	PCIE0_CLK_N	54	PCIE_WAKE
55	PCIE0_CLK_P	56	NC
57	GND	58	NC
59	KEY	60	KEY
61	KEY	62	KEY
63	KEY	64	KEY
65	KEY	66	KEY
67	NC	68	NC
69	NGFF3_PEDET	70	V3P3_M2M

Pin	Signal	Pin	Signal
71	GND	72	V3P3_M2M
73	GND	74	V3P3_M2M
75	GND		

2.2. System Setup

Before you start the installation, check the following safety instructions:

Important:

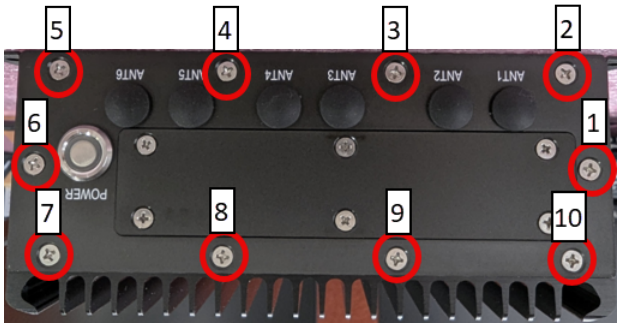
- Ensure the device is not connected to any power source such as a power adapter or a battery.
- Prior to installing any modules on the mainboard, always touch an unpainted and grounded metal object or wear a grounded anti-static wrist strap to prevent electrostatic discharge (ESD).

See the following steps to start the installation:

2.2.1. Removing the Bottom Cover

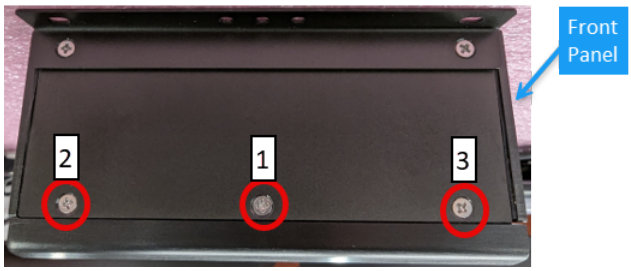
The bottom cover must be removed prior to installing expansion modules or an optional backup battery.

1. Turn the system upside down. Remove the 10 screws on the rear panel in the order indicated in the following figure.



2. Remove the 10 screws on the front panel in the order indicated in the following figure.

3. On the side panel, remove the three screws in the order indicated in the following figure.



4. On the other side panel, remove the three screws in the order indicated in the following figure.

5. After removing all the specified screws, gently lift the bottom cover and place it carefully.

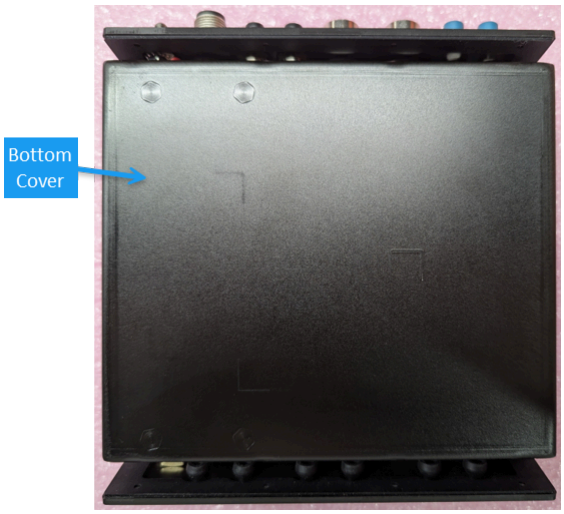


Figure 2-2 Mainboard of IBOX-602P-IP66

Important:

- Reassemble the system by following the steps above in reverse order.
- To ensure IP66 protection, fasten the screws in the order indicated in the figures above, with a tightening torque of 4 kgf-cm.

2.2.2. Installing Expansion Modules

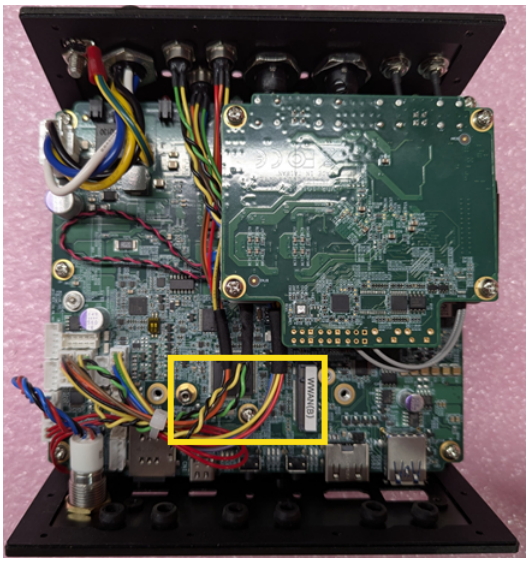
This section provides step-by-step instructions on how to install M.2 3042/3052 and M.2 2230 expansion modules. You can also refer to Expansion *(on page 42)* for more details such as pin definitions about the internal slots or connectors used for expansion purpose.

You need to remove the bottom cover to install expansion modules. See Removing the Bottom Cover *(on page 21)* for the instructions.

2.2.2.1. Installing an M.2 3042 or 3052 Module

1. **Installing an M.2 3042 Key B LTE WWAN card**:

   a. Locate the M.2 3042 Key B slot on the mainboard.

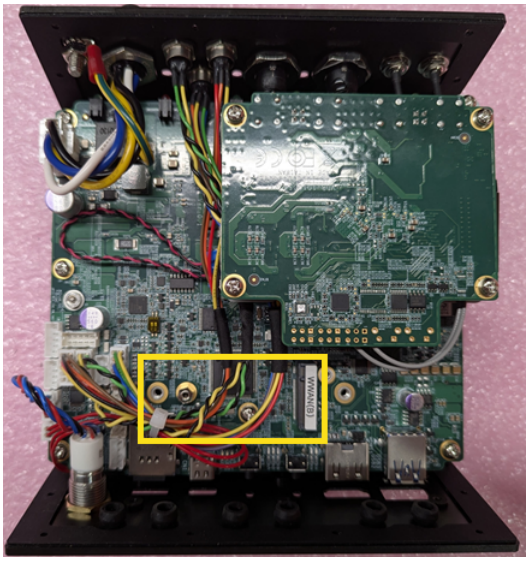
   b. Align the notch on the M.2 3042 WWAN card with the tab in the slot and gently insert the WWAN card at a 30 degree angle until it is fully embedded, and then press it down.

   c. Secure the expansion module to the mainboard with the M2.5x5L screw provided in the package (see Package Contents *(on page 13)*).

   d. To install a heatsink for the installed M.2 3042 module, go to Installing a Heatsink for the M.2 3042 or 3052 Module *(on page 24)* for the instructions.

2. **Installing an M.2 3052 Key B 5G WWAN card**:

   a. Locate the M.2 3052 Key B slot on the mainboard.

   b. Move the standoff screw (H73D50) from location H8 to H9 as shown below:

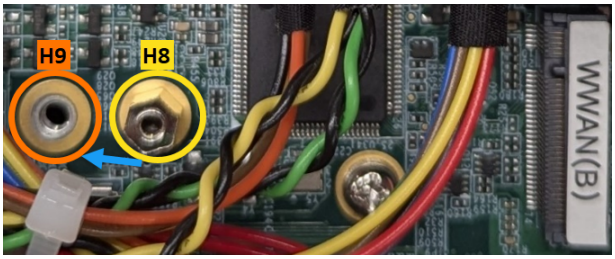


Figure 2-3 Standoff Screw (H73D50)

c. Align the notch on the M.2 3052 WWAN card with the tab in the slot and gently insert the WWAN card at a 30 degree angle until it is fully embedded, and then press it down.

d. To install a heatsink for the installed M.2 3052 module, go to Installing a Heatsink for the M.2 3042 or 3052 Module *(on page 24)* for the instructions.

2.2.2.1.1. Installing a Heatsink for the M.2 3042 or 3052 Module

1. Before you begin, check the Package Contents *(on page 13)* and get the items below:
   - 2 x standoff screw (H75D50)
   - 2 x M2.5x10L screw
   - 1 x M.2 WWAN heatsink Type 12
   - 1 x thermal pad (18x30x1.0T mm)
   - 1 x thermal pad (30x34x1.0T mm)
2. Attach the thermal pads to the heatsink as shown below.

   a. Figure 2-4 Thermal pad (30x34x1.0T mm) attached to the heatsink

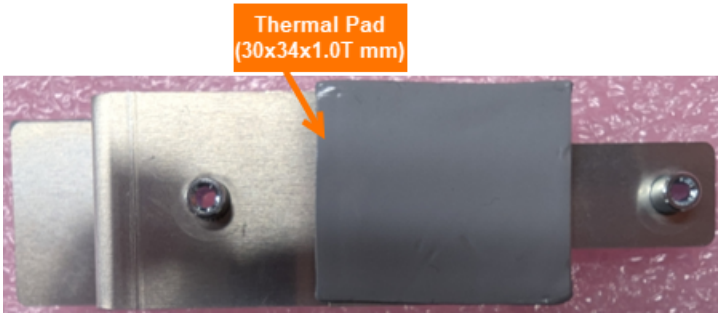
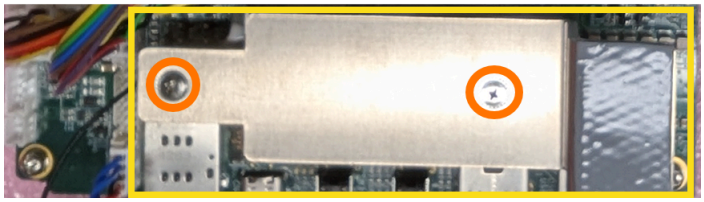


Figure 2-5 Thermal pad (18x30x1.0T mm) attached to the heatsink

3. Fasten the standoff screws (H75D50) to the mainboard as indicated in the figure below.

4. Align the two mounting holes on the heatsink with the mounting screw-holes on the mainboard, and then place the heatsink above the installed M.2 3042 or 3052 module.
5. Fasten the heatsink to the mainboard with the M2.5x10L screws.

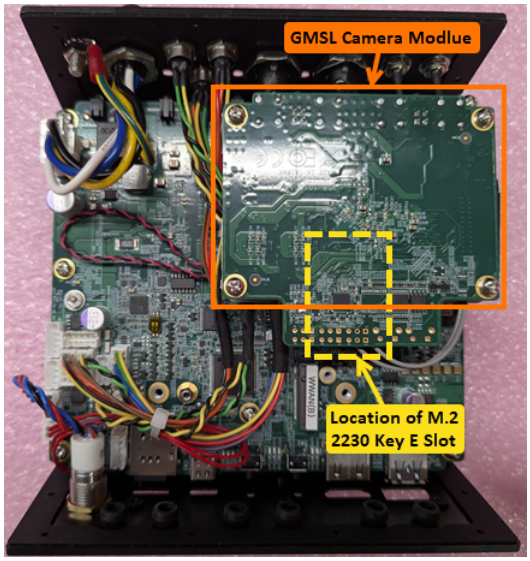


Note:
One of the M2.5x10L screw is also used to secure the M.2 3052 module to the mainboard.

6. After completing the installation of the M.2 3042 LTE or 3052 5G module, go to Installing Dual Nano SIM Cards *(on page 29)* for how to install SIM cards.

2.2.2.2. Installing an M.2 2230 Module

1. The M.2 2230 Key E slot is located beneath the GMSL camera module. The camera module must be removed prior to installing an M.2 2230 module.

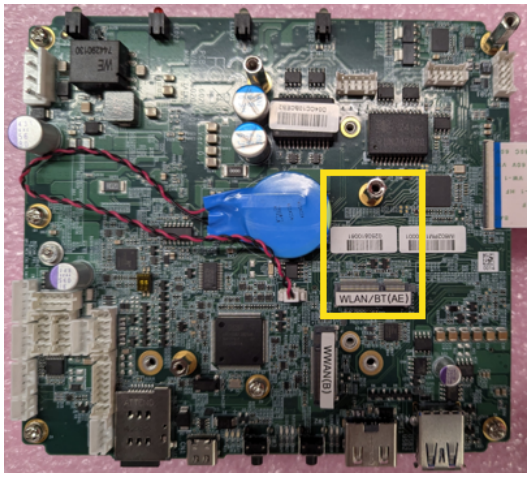


2. Remove the four screws to detach the GMSL camera module from its bracket.

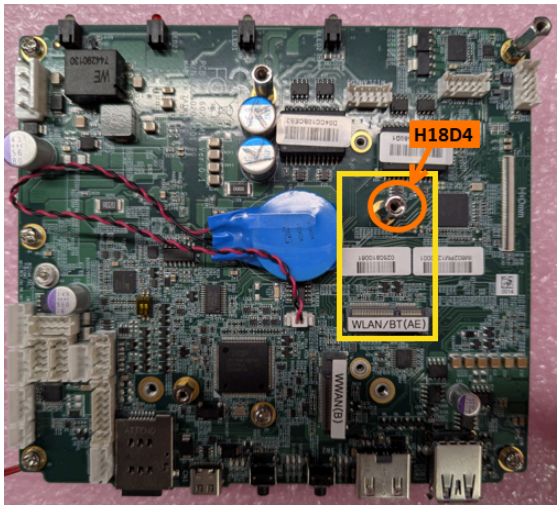
3. Carefully lift the camera module upward and gently place it aside, ensuring the attached cables remain connected and undisturbed.
4. Remove the three screws to detach the camera module bracket from the mainboard.

Important:
To prevent irreversible damage, avoid pulling or bending the cables and connectors during removal of the camera module and bracket.

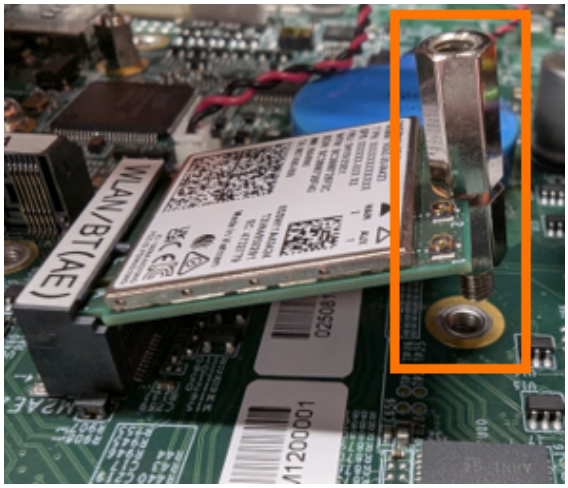
5. Locate the M.2 2230 Key E slot on the mainboard.



6. Remove the copper pillar (H18D4) and keep it for later use.



7. Align the notch on the expansion module with the tab in the slot and gently insert the module at a 30 degree angle until it is fully embedded.
8. Align the semicircular mounting hole at the center of the opposite edge on the module with the copper pillar (H18D4), and then press the module down.



9. Secure the expansion module to the mainboard with the copper pillar (H18D4).
10. Refer to **Step 4** and **Step 2** to reinstall the bracket and camera module.

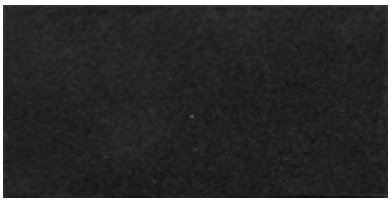
2.2.3. Installing a Backup Battery (BBU)

1. Before you begin, check the optional backup battery kit purchased from SINTRONES®, which includes:
    - 1 x battery cover

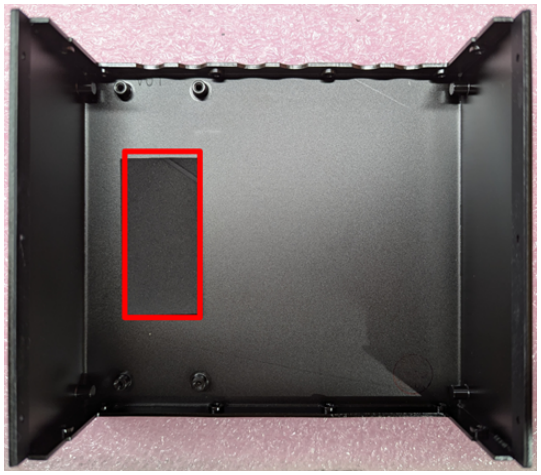
      
    - 1 x battery with a power cable

      
    - 4 x screw P3*6L

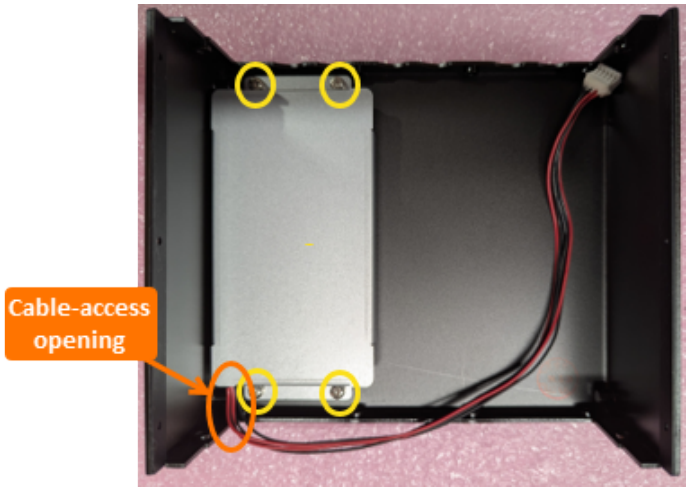
      
    - 1 x sponge

      
2. On both sides of the battery, peel off the release liners of the double-sided tapes.
3. Turn over the battery and adhere it to the battery cover with the information label face up.

   
4. Remove the bottom cover of the system to install the backup battery. See Removing the Bottom Cover *(on page 21)* for the instructions.
5. On both sides of the sponge, peel off the release liners of the double-sided tape.
6. Attach the sponge to the area on the back of the bottom cover.



7. Turn over the battery cover and fasten it to the bottom cover with the P3*6L srews. Get the power cable out from the cable-access opening as shown below.

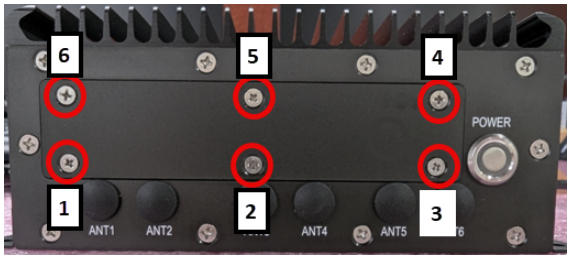


8. Connect the other end of the power cable to the UPS1 position on the mainboard as shown below.

2.2.4. Installing Dual Nano SIM Cards

The system provides a dual Nano SIM card tray within a single slot for dual SIM installation. See the following instructions for how to install two Nano SIM cards into the system.

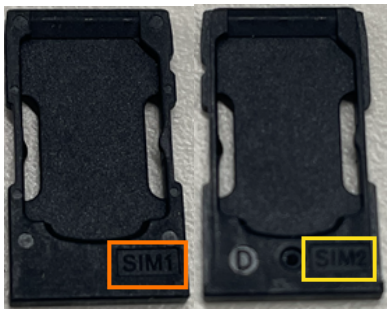
1. Ensure the IBOX-602P-IP66 is powered off.
2. Remove the 6 screws in the order indicated in the figure below to remove the protection cover on the rear panel.



3. Locate the **Nano SIM CARD** slot on the panel.

4. Use your fingernail or a small paperclip to push the dual SIM tray until it pops out.
5. The dual SIM tray has two opposing surfaces. Check the printed indicators for SIM1 and SIM2 as shown below.

Figure 2-6 SIM1 and SIM2



6. With the gold or bronze chip sides facing outward, place each SIM card onto its designated surface, and then insert the tray into the **Nano SIM CARD** slot.

Note:
By default, SIM1 is selected for use. See SIM Card Switch Commands *(on page 66)* for how to switch between the installed SIM cards.

7. Use your fingernail or a small paperclip to press the dual SIM tray inwards until it locks in place.
8. Fasten the protection cover back onto the rear panel by the specified order given in the figure in **Step 2** *(on page 30)*, with a tightening torque of 4 kgf-cm.

Note:
To remove the SIM card, follow the steps above.

2.3. Booting the System

This section describes how to boot the system via an ignition switch or the power button.

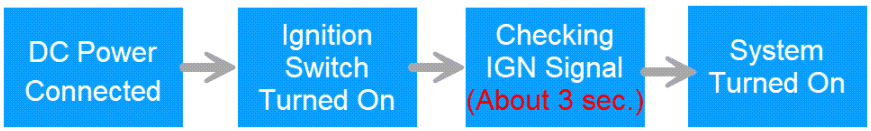
2.3.1. Turning On System By Ignition Switch

1. You can use the power cable that came with the package directly in fields without an ignition system.
2. Plug one end of the power cable into the DC-IN port of the computer and then plug the other end to a power source.
3. When the power is ready, turn on the ignition switch.
4. The system starts checking if there's any incoming ignition signal. It takes about **3 seconds** for the system to run the ignition control process.

   Note:
   SINTRONES® provides ignition power management that monitors the ignition signal and controls the system boot process to avoid potential risk of downtime or damages to the connected devices and system.

5. After the ignition control process is complete, the system will boot up.

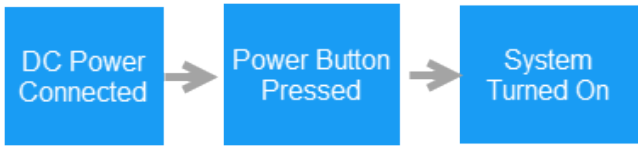
   Figure 2-7 Turning On System By Ignition Switch



2.3.2. Turning On System By Power Button

1. When the system is connected to a suitable DC power source, press the power button.
2. The **Power Button** turns to blue light when the system boots up.

   Figure 2-8 Turning On System By Power Button



2.4. Default Login Credentials

To get started after booting the system, log in using the default credentials:

- **Username**: `sintrones` (enter in lowercase)
- **Password**: `24332747`

  Important:
  For security reasons, change both the username and password immediately after your first login.

3. External I/O Ports

Topics in this chapter include:

3.1. Front Panel

This section provides the front panel view and the description of the associated I/O interface.

Figure 3-1 Front Panel of IBOX-602P-IP66

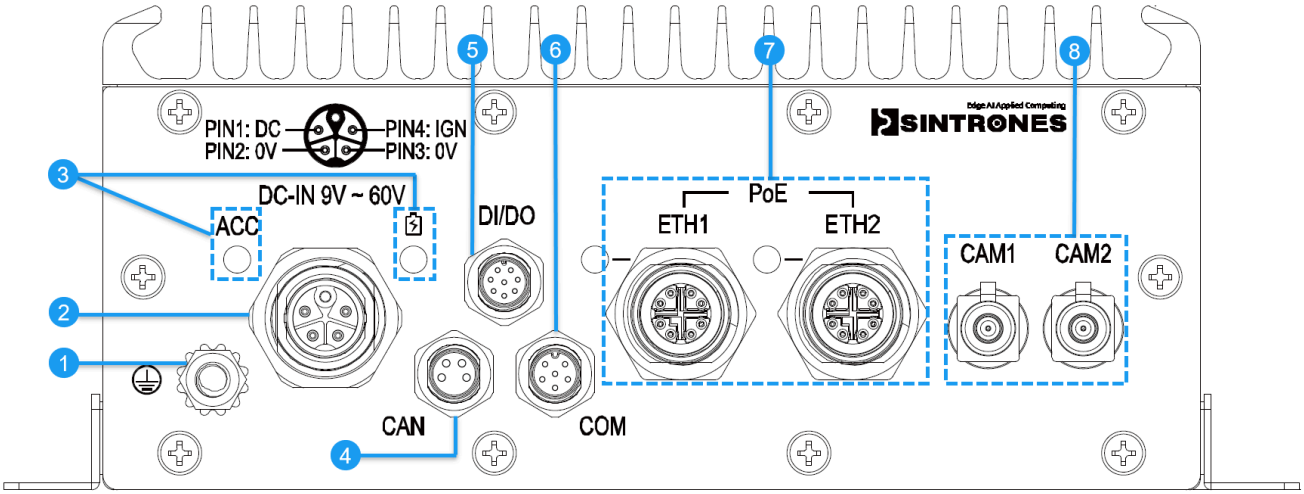


Table 3-1 I/O Interface on Front Panel

Item	I/O Interface	Description	Specification
1	Grounding Terminal	M5 bolt with nut for chassis grounding	-
2	DC Input	• Input voltage range: DC 9–60V • M12 K-coded connector	DC-IN Port (M12 K-coded Connector) *(on page 37)*
3	LED Indicators	ACC: • ON: Ignition enabled • OFF: Ignition disabled	-
		(BBU): • ON: Internal backup battery enabled • OFF: Power supplied from external power source or no backup battery installed	-
4	CAN FD Port	• Supports CAN FD protocol • Backward compatible with CAN bus 2.0 • M8 A-coded connector	CAN FD Port (M8 A-coded Connector) *(on page 37)*
5	DIO Port	• 3 x DI (DC 5-60V), 4 x DO (12V / 100mA) • M8 A-coded connector	DI/DO Port (M8 A-coded Connector) *(on page 38)*

Item	I/O Interface	Description	Specification
6	COM Port	• Supports RS-232/422/485 interfaces • Programmable via software configuration • M12 A-coded connector	COM Port (M8 A-coded Connector) *(on page 38)*
7	LAN Ports with PoE	• ETH1: 2.5 GbE (Intel® I226-IT) • ETH2: GbE (NVIDIA® SoM) • PoE supports 15.4W per port (PSE Class 3) with total power budget: 32W • M12 X-coded connectors	LAN Port (M12 X-coded Connector) *(on page 39)*
8	Camera Ports	FAKRA-Z connectors for GMSL-2 cameras	-

3.2. Rear Panel

This section provides the rear panel view and the description of the associated I/O interface.

Figure 3-2 Rear Panel of IBOX-602P-IP66

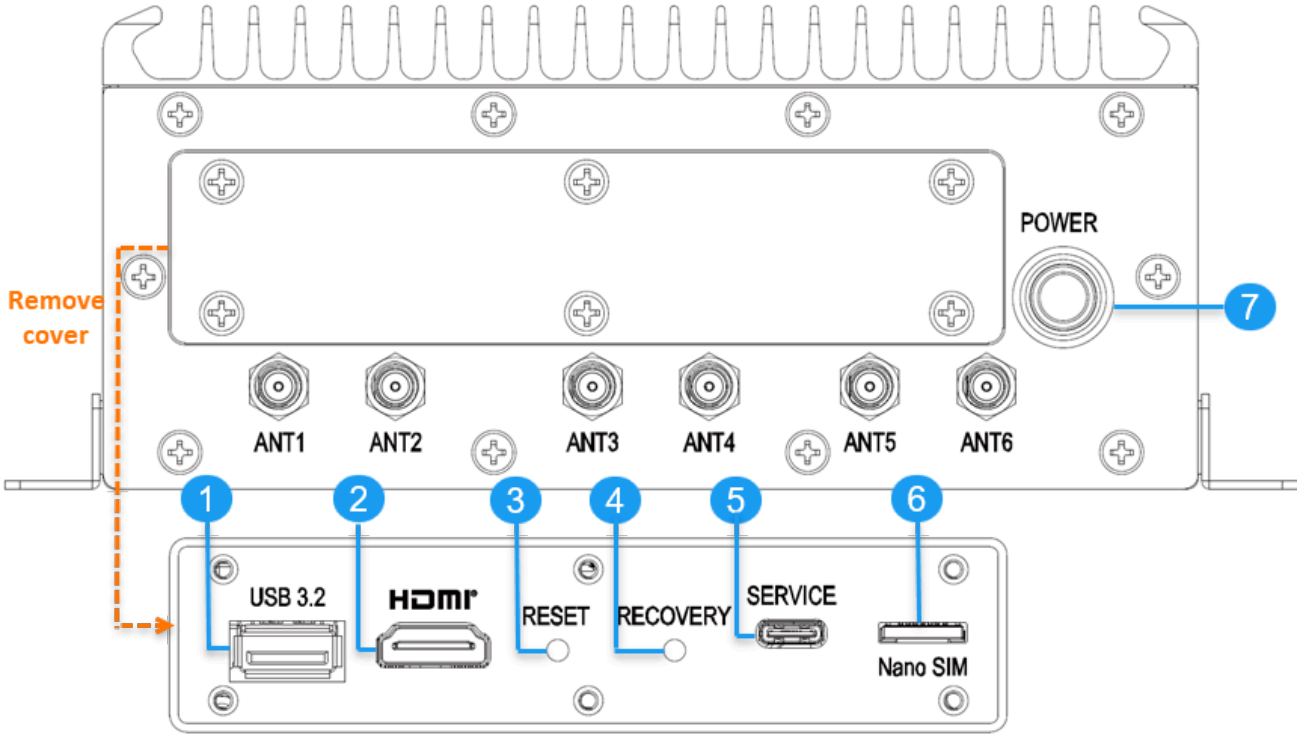


Table 3-2 I/O Interface on Rear Panel

Item	I/O Interface	Description	Specification
1	USB 3.2 Port	• USB 3.2, 5V / 900mA • Type-A	USB 3.2 Port *(on page 39)*
2	HDMI® Port	Supports HD audio output	HDMI® Port *(on page 40)*
3	Reset Button	Used for system reboot	-
4	Recovery Button	Used for system recovery **Note:** The recovery button works only when the USB Type-C (Service) port (as described below) is connected to a host computer.	-
5	USB (Service) Port	Used for system recovery when connected to a host computer containing certain JetPack BSP image via a USB Type-C cable. See System Recovery *(on page 68)* for the detailed instructions.	USB Type-C Port *(on page 41)*

Item	I/O Interface	Description	Specification
		Note: This USB port is used only for system recovery. It does not support power or other kinds of data transfer.	
6	SIM Card Slot (Dual SIM Tray)	• Supports two nano SIM cards • See Installing Dual Nano SIM Cards *(on page 29)* for how to install two Nano SIM cards into the system. • See SIM Card Switch Commands *(on page 66)* for how to switch between the installed SIM cards. **Note:** Ensure you power off the system before installing or removing the SIM card(s).	-
7	Power Button	System power status: • Red light: Standby mode • Blue light: System turned on	Power Button (LED Light Status) *(on page 41)*

3.3. Specifications of External I/O Ports

This section provides drawings and pin definitions of the external I/O ports.

3.3.1. DC-IN Port (M12 K-coded Connector)

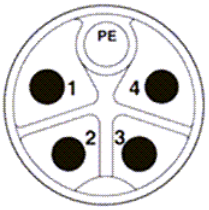


Table 3-3 Pin Definition of DC-IN Port (M12 K-coded Connector)

Pin	Signal
1	9 ~ 60 VDC
2	0V
3	0V
4	IGN

3.3.2. CAN FD Port (M8 A-coded Connector)

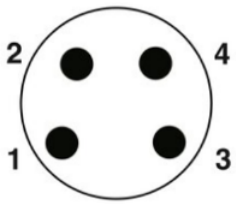


Table 3-4 Pin Definition of CAN Bus Port

Pin	M8 A-coded
	CAN Bus
1	CAN-L
2	GND
3	CAN-H
4	GND

3.3.3. DI/DO Port (M8 A-coded Connector)

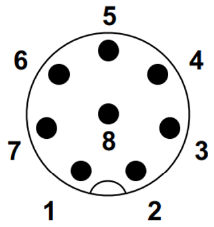


Table 3-5 Pin Definition of DI/DO Port

Pin	Signal	Pin	Signal
1	DO_2	2	DO_3
3	DO_4	4	DI_3 (DI_2)
5	DI_2 (DI_1)	6	DI_1 (DI_0)
7	DO_1	8	GND

3.3.4. COM Port (M8 A-coded Connector)

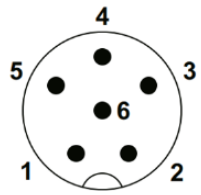


Table 3-6 Pin Definition of COM Port (M8 A-coded)

Pin	M8 A-coded		
	RS-232	RS-422	RS-485
1	NC	TXD-	Data-
2	RXD	TXD+	Data+
3	TXD	RXD+	NC
4	NC	RXD-	NC
5	GND	GND	GND
6	NC	NC	NC

3.3.5. LAN Port (M12 X-coded Connector)

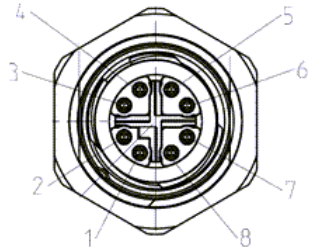


Table 3-7 Pin Definition of LAN Port (M12 X-coded Connector)

Pin	Signal
1	LAN_MDI0P
2	LAN_MDI0N
3	LAN_MDI1P
4	LAN_MDI1N
5	LAN_MDI3P
6	LAN_MDI3N
7	LAN_MDI2N
8	LAN_MDI2P

3.3.6. USB 3.2 Port

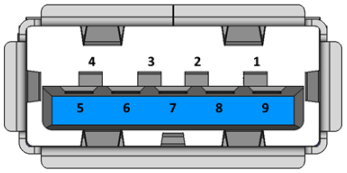


Table 3-8 Pin Definition of USB 3.2 Port

Pin	Signal
1	VBUS
2	D-
3	D+
4	GND
5	StdA_SSRX-
6	StdA_SSRX+
7	GND_DRAIN
8	StdA_SSTX-
9	StdA_SSTX+

3.3.7. HDMI® Port

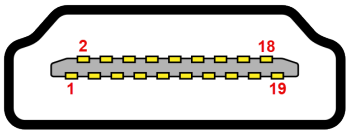


Table 3-9 Pin Definition of HDMI® Port

Pin	Signal	Pin	Signal
1	HDMI_DATA2_P	2	GND
3	HDMI_DATA2_N	4	HDMI_DATA1_P
5	GND	6	HDMI_DATA1_N
7	HDMI_DATA0_P	8	GND
9	HDMI_DATA0_N	10	HDMI_CLK_P
11	GND	12	HDMI_CLK_N
13	CEC	14	NC
15	HDMI_SCL	16	HDMI_SDA
17	GND	18	V5P_S_HDMI
19	HDMI_HPDET		

Note:

The pin definition of this connector conforms to the HDMI specification; however, the CEC function has not been validated.

3.3.8. USB Type-C Port

Table 3-10 Pin Definition of USB Type-C Port

Pin	Signal	Pin	Signal
A1	GND	B12	GND
A2	NC	B11	NC
A3	NC	B10	NC
A4	NC	B9	NC
A5	NC	B8	NC
A6	Dp1	B7	Dn2
A7	Dn1	B6	Dp2
A8	NC	B5	NC
A9	NC	B4	NC
A10	NC	B3	NC
A11	NC	B2	NC
A12	GND	B1	GND

3.3.9. Power Button (LED Light Status)

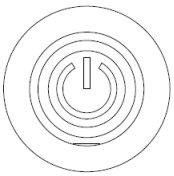


Table 3-11 Pin Definition of Power Button (LED Light Status)

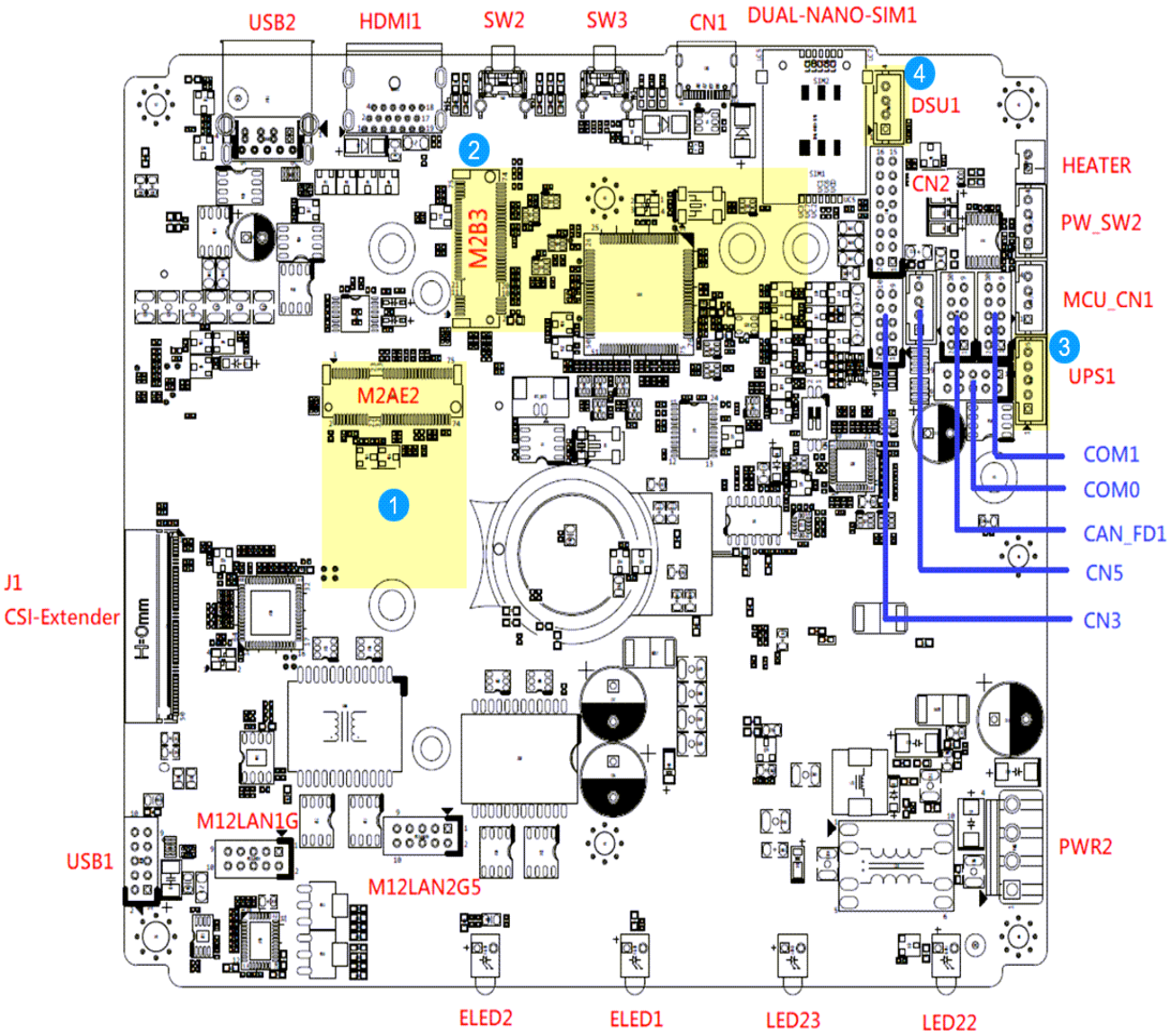
Pin	Signal	Pin	Signal
LED			
A1	POWER_LED+	C1	POWER_LED-
Switch			
1	GND	2	BUTTON_POWER_ON

4. Expansion

This chapter provides more details about the internal slots or connectors used for expansion purpose.

Topics in this chapter include:

4.1. Top View of Mainboard



Item	Internal Connector	Description	Specification
1	M.2_AE_KEY	M.2 2230 Key E slot used for installing an expansion module such as a WLAN or Bluetooth module	M.2 Key E Slot *(on page 44)*
2	M2B3	M.2 3042/3052 Key B slot used for installing an expansion module such as a WWAN module	M.2 Key B Slot *(on page 45)*
3	UPS1	Used for installing the backup battery	UPS1 (BBU) JST Connector *(on page 47)*
4	DSU1	Used for monitoring system status and collecting debug logs	DSU JST Connector *(on page 48)*

4.2. Specifications of Expansion Slots/Connectors

This section provides drawings and pin definitions about the slots or connectors used to install expansion modules.

4.2.1. M.2 Key E Slot

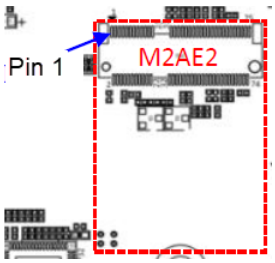
M.2 Key E Slot	Description
Size	NGFF 2230 / 75 Pin
Type	M.2 Key E H:8.5mm
Location	See Top View of Mainboard *(on page 43)* for the information.
Drawing	

Table 4-1 Pin Definition of M.2 Key E Slot

Pin	Signal	Pin	Signal
1	GND	2	V3P3_A
3	HUB3_DP_DN4	4	V3P3_A
5	HUB3_DM_DN4	6	NC
7	GND	8	NC
9	NC	10	NC
11	NC	12	NC
13	NC	14	NC
15	NC	16	NC
17	NC	18	GND
19	NC	20	UART WAKE (Reserve)
21	NC	22	UART0_RXD
23	NC	24	KEY
25	KEY	26	KEY
27	KEY	28	KEY
29	KEY	30	KEY
31	KEY	32	UART0_TXD

Pin	Signal	Pin	Signal
33	GND	34	UART0_CTS
35	PCIE1_TX0_P	36	UART0_RTS
37	PCIE1_TX0_N	38	NC
39	GND	40	NC
41	PCIE1_RX0_P	42	NC
43	PCIE1_RX0_N	44	NC
45	GND	46	NC
47	PCIE1_CLK_P	48	NC
49	PCIE1_CLK_N	50	BT_OSC_32KHZ
51	GND	52	PCIE1_RST
53	PCIE1_CLKREQ	54	M2E_WIFI_DIS2(BT)
55	PCIE_WAKE	56	M2E_WIFI_DIS1(WIFI)
57	GND	58	JS_I2C2_SDA (Reserve)
59	NC	60	JS_I2C2_SCL (Reserve)
61	NC	62	M2E_ALERT
63	GND	64	NC
65	NC	66	NC
67	NC	68	NC
69	GND	70	WIFI_WAKE (Reserve)
71	NC	72	V3P3_A
73	NC	74	V3P3_A
75	GND		

4.2.2. M.2 Key B Slot

M.2 Key B Slot	Description
Size	NGFF 3042 / 3052 / 75 Pin
Type	M.2 Key B H:8.5mm
Location	See Top View of Mainboard *(on page 43)* for the information.

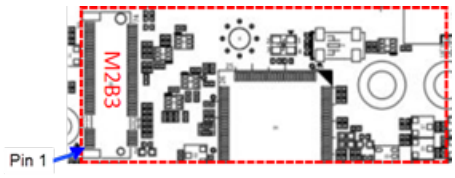
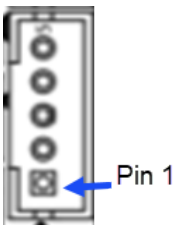
M.2 Key B Slot	Description
Drawing	

Table 4-2 Pin Definition of M.2 Key B Slot

Pin	Signal	Pin	Signal
1	NC	2	V3P3A_M2B1
3	GND	4	V3P3A_M2B1
5	GND	6	PWR_OFF (Pull-H_1.8V)
7	HUB3_DP_DN3	8	W_DIS1# (Pull-H_3.3V)
9	HUB3_DM_DN3	10	LED# (Pull-H_3.3V)
11	GND	12	KEY
13	KEY	14	KEY
15	KEY	16	KEY
17	KEY	18	KEY
19	KEY	20	NC
21	NC	22	NC
23	WWAN_WAKE	24	NC
25	M2B1DPR	26	WDIS2# (Pull-H_1.8V)
27	GND	28	NC
29	HUB3_SSRXM_DN3	30	M2UIM_RST_A
31	HUB3_SSRXP_DN3	32	M2UIM_CLK_A
33	GND	34	M2UIM_DAT_A
35	HUB3_SSTXM_DN3	36	M2UIM_PWR_A
37	HUB3_SSTXP_DN3	38	NC
39	GND	40	NC
41	NC	42	NC
43	NC	44	M2B_ALERN (Reserve)
45	GND	46	NC
47	NC	48	NC
49	NC	50	DEV_3V3RST# (Reserve)
51	GND	52	CLKREQ# (Reserve)
53	NC	54	WWAN_WAKE (Reserve)

Pin	Signal	Pin	Signal
55	NC	56	NC
57	GND	58	NC
59	NC	60	NC
61	NC	62	NC
63	NC	64	NC
65	NC	66	NC
67	M2B1RST2	68	NC
69	CONFIG_1	70	V3P3A_M2B1
71	GND	72	V3P3A_M2B1
73	GND	74	V3P3A_M2B1
75	NC		

4.2.3. UPS1 (BBU) JST Connector



See Top View of Mainboard *(on page 43)* for the information.

Table 4-3 Pin Definition

Pin	Signal
1	V12P_A_UPS
2	V12P_A_UPS
3	NC
4	GND
5	GND

4.2.4. DSU JST Connector

DSU JST Connector	Description
Size	1 x 4 / 4 Pin
Type	JST-2.0mm-M-180
Location	See Top View of Mainboard *(on page 43)* for the information.
Drawing	

Table 4-4 Pin Definition

Pin	Signal
1	Power (Reserve)
2	JETSON_UART2_TX
3	JETSON_UART2_RX
4	GND

The DSU JST connector is used to monitor the system status and collect debug logs via UART protocol by advanced developers. There's no need to use this connector under normal operation.

To retrieve UART logs, connect a host computert to the system with a USB-UART or RS232-UART converter cable (not included in the package), and then configure the UART settings on the host computer as shown below:

- Baud rate: 115,200bps
- Voltage: DC 3.3V
- Data bits: 8
- Parity bit: 0
- Flow control: No

5. Software

Topics in this chapter include:

5.1. System Configuration

This section provides instructions on how to initialize GMSL cameras as well as summarizes commands available for configuring smart power management, thresholds for low-temperature heater, COM (RS-232/422/485) ports, CAN FD, DI/DO interfaces, SIM card switch, and PoE function.

5.1.1. Setting the GMSL Camera

This section provides instructions on how to initialize and configure the connected GMSL cameras.

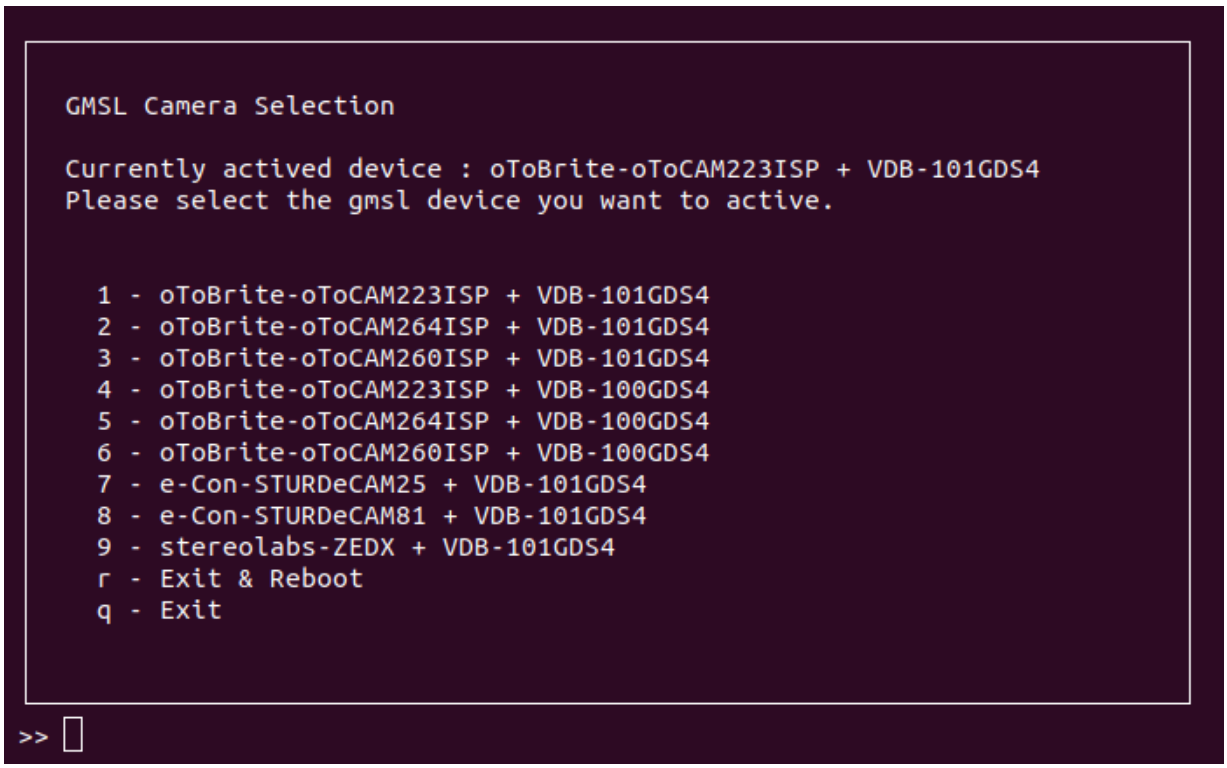
See the following steps:

1. Before connecting GMSL cameras, ensure the system is powered off.
2. Connect the GMSL cameras to the camara ports via the FAKRA-Z cables.

   Important:
   Connect only one type of cameras at a time. Different camera types cannot be mixed because the system does not support running multiple camera drivers simultaneously.

3. Connect the system to an HDMI monitor, a keyboard, and a mouse.
4. Boot the system.
5. Open the terminal window.
6. Enter the following command to pull up a list of the available GMSL camera modules.
   `sudo gmsl-camera-selection`
7. Select the GMSL camera module you want to initialize from the list.

   Figure 5-1 An Example of GMSL Camera Module List Available for Selection



Note:
Ensure you select the module that matches the connected cameras. Contact our sales representatives for assistance if needed.

8. Enter `r` to reboot the system.
9. Continue to the next sections for vendor-specific camera configuration instructions.

5.1.1.1. oToBrite GMSL Camera Settings

1. After the system reboots, enter the following command to verify whether the connected oToBrite GMSL cameras have been successfully initialized.
   `ls /dev/vi*`
2. For example, if you connect two oToBrite GMSL cameras to the system via CAM1 and CAM2 ports, the corresponding device names shall appear in the command output.

   Port names and corresponding device names shown in CLI:

   - CAM1: video1
   - CAM2: video2
3. To check a live video feed streaming from a specified camera, enter the following command:
   `gst-launch-1.0 v4l2src device="/dev/<video?>" ! videoconvert ! xvimagesink &`

   Note:
   Replace the texts within the brackets with the actual device name. For example, if the device name is `video1`, namely the camera connected via CAM1 port, enter

   `gst-launch-1.0 v4l2src device="/dev/video1" ! videoconvert ! xvimagesink &`
4. (Optional) If you want to add text overlay to the video streaming from CAM1, enter the following command:
   `gst-launch-1.0 v4l2src device="/dev/video1" ! textoverlay text="VIDEO 1" valignment=top halignment=left font-desc="Sans, 18" ! clockoverlay valignment=top halignment=right time-format="%D %H:%M:%S" ! videoconvert ! xvimagesink`

   Note:
   Likewise, replace `video1` with the actual device name to add text overlay to the video streaming from the specified camera.

   Important:
   To avoid unintended line breaks that may cause the command to fail, **copy the command into a plain text file to ensure it remains a single, continuous line** before pasting it into the terminal.

5.1.1.2. e-con Systems and Stereolabs GMSL Camera Settings

1. Connect the system to the Internet.
2. Download the GMSL camera streaming tool from the e-con Systems or Stereolabs website.
3. Follow the instructions on the e-con Systems or Stereolabs website to complete the respective camera settings.

5.1.2. Smart Power Management Settings

This section summarizes the commands available for configuring the Smart Power Management function.

5.1.2.1. Overview of Smart Power Management Commands

The general formats of the Smart Power Management commands are as follows:

`i2cget -y -f <i2c_num> <device_addr> <reg_addr>`: Used to **check** the current state of certain functionality.

`i2cset -f -y <i2c_num> <device_addr> <reg_addr> <value>`: Used to **change or save** the settings of certain functionality.

5.1.2.2. Smart Power Management Commands

See the following tables as the available commands for Smart Power Management.

Table 5-1 Checking IGN/UPS Status

Command	Value	Description	Outcome
`i2cget -y -f 1 0x4a 0x12`	-	Check the ignition (IGN) status	• `0x00`: Ignition turned off • `0x01`: Ignition turned on
`i2cget -y -f 1 0x4a 0x10`	-	Check the UPS backup battery status	• `0x07`: UPS mode

Table 5-2 Checking and Setting Ignition Power On Delay Time

Command	Value	Description	Outcome
`i2cget -y -f 1 0x4a 0x60`	-	Check the ignition power on delay time by seconds	If the delay time is set as 3 seconds, the output will be `0x03`
`i2cset -f -y 1 0x4a 0x60`	For example: `0x03`	Set the ignition power on delay time by seconds	For example, if you enter `i2cset -f -y 1 0x4a 0x60 0x03`, and then save the setting *(on page 54)*, the system will wait for 3 seconds to power on after the ignition is turned on.
`i2cget -y -f 1 0x4a 0x61`	-	Check the ignition power on delay time by minutes	If the delay time is set as 2 minutes, the output will be `0x02`
`i2cset -f -y 1 0x4a 0x61`	For example: `0x02`	Set the ignition power on delay time by minutes	For example, if you enter `i2cset -f -y 1 0x4a 0x61 0x02`, and then save the setting *(on page 54)*, the system will wait for 2 minutes to power on after the ignition is turned on.
`i2cget -y -f 1 0x4a 0x62`	-	Check the ignition power on delay time by hours	If the delay time is set as 1 hour, the output will be `0x01`

Command	Value	Description	Outcome
`i2cset -f -y 1 0x4a 0x62`	For example: `0x01`	Set the ignition power on delay time by hours	For example, if you enter `i2cset -f -y 1 0x4a 0x62 0x01`, and then save the setting *(on page 54)*, the system will wait for 1 hour to power on after the ignition is turned on.

Table 5-3 Checking and Setting Ignition Power Off Delay Time

Command	Value	Description	Outcome
`i2cget -y -f 1 0x4a 0x66`	-	Check the ignition power off delay time by seconds	If the delay time is set as 3 seconds, the output will be `0x03`
`i2cset -f -y 1 0x4a 0x66`	For example: `0x03`	Set the ignition power off delay time by seconds	For example, if you enter `i2cset -f -y 1 0x4a 0x66 0x03`, and then save the setting *(on page 54)*, the system will wait for 3 seconds to power off after the ignition is turned off.
`i2cget -y -f 1 0x4a 0x67`	-	Check the ignition power off delay time by minutes	If the delay time is set as 2 minutes, the output will be `0x02`
`i2cset -f -y 1 0x4a 0x67`	For example: `0x02`	Set the ignition power off delay time by minutes	For example, if you enter `i2cset -f -y 1 0x4a 0x67 0x02`, and then save the setting *(on page 54)*, the system will wait for 2 minutes to power off after the ignition is turned off.
`i2cget -y -f 1 0x4a 0x68`	-	Check the ignition power off delay time by hours	If the delay time is set as 1 hour, the output will be `0x01`
`i2cset -f -y 1 0x4a 0x68`	For example: `0x01`	Set the ignition power off delay time by hours	For example, if you enter `i2cset -f -y 1 0x4a 0x68 0x01`, and then save the setting *(on page 54)*, the system will wait for 1 hour to power off after the ignition is turned off.

Table 5-4 Saving the Setting

Command	Description	Outcome
`i2cset -f -y 1 0x4a 0x28 0x02`	Save the setting.	The specified setting will be saved. **Important:** Ensure you enter `i2cset -f -y 1 0x4a 0x28 0x02` to save and allow the system to excute the specified setting.

5.1.3. Configuring Thresholds for Low-Temperature Heater

To ensure safe system booting and reliable operation in low-temperature environments, a built-in heater activates under either of the following conditions:

- During system startup, when temperatures ranges from -40°C to -25°C
- After system startup, when temperatures falls below -10°C

This section summarizes the commands used to configure thresholds for the low-temperature heater.

5.1.3.1. Overview of Commands for Setting Low-Temperature Heater

The following table outlines three I²C commands used to configure temperature thresholds for the built-in low-temperature heater. Each command sets a specific condition by writing a value to register `0xDC`, `0xDE`, or `0xDF` at device address `0x4A`.

Command	Description	Default Setting	Setting Range
`I2cset -y 1 0x4A 0xDC <value>`	Used to set the system startup temperature.	-25°C.	-25°C to -20°C
`I2cset -y 1 0x4A 0xDE <value>`	Used to set the temperature below which the heater is enabled.	Below -10°C.	-25°C to 0°C
`I2cset -y 1 0x4A 0xDF <value>`	Used to set the temperature above which the heater is disabled.	Above -10°C.	No restriction

5.1.3.2. Example Commands and Conversion Table for Setting Low-Temperature Heater

See the following tables as the examples commands and conversion table for setting thresholds for the low-temperature heater.

Table 5-5 Example Commands

Command	Value	Description	Outcome
`I2cset -y 1 0x4A 0xDC`	0xE7	Set the system startup temperature as -25°C	When the temperature falls between -40°C and -25°C during startup, the built-in heater activates to raise the temperature to a safe threshold. Once the temperature **reaches -25°C**, the system proceeds with booting.
`I2cset -y 1 0x4A 0xDE`	0xF6	Set the temperature at -10°C below which the heater is enabled	After the system boots up, when the temperature is **below -10°C**, the heater will be enabled.
`I2cset -y 1 0x4A 0xDF`	0xF6	Set the temperature at -10°C above which the heater is disabled	After the system boots up, when the temperature is **above -10°C**, the heater will be disabled.

Table 5-6 Conversion Table: Negative Numbers to 8-Bit Hexadecimal Format

Neg. No.	8-Bit Hex	Neg. No.	8-Bit Hex	Neg. No.	8-Bit Hex
0	0x00	-10	0xF6	-20	0xEC
-1	0xFF	-11	0xF5	-21	0xEB
-2	0xFE	-12	0xF4	-22	0xEA
-3	0xFD	-13	0xF3	-23	0xE9
-4	0xFC	-14	0xF2	-24	0xE8
-5	0xFB	-15	0xF1	-25	0xE7
-6	0xFA	-16	0xF0	-26	0xE6
-7	0xF9	-17	0xEF	-27	0xE5
-8	0xF8	-18	0xEE	-28	0xE4
-9	0xF7	-19	0xED	-29	0xE3

5.1.4. COM Port (RS-232/422/485) Configuration

This section provides commands, sample code, and test programs to configure the COM port transceiver mode and validate serial communication.

The name of the COM port: ttyTHS0

See the following tables as the command and relevant information for checking the COM port status.

Table 5-7 Checking the Status

Command	Outcome	Description
`i2cget -f -y 1 0x4a 0x3c`	• 0bxxx01xxx: RS232 • 0bxxx11xxx: RS422 • 0bxxx10xxx: RS485	Check which mode (RS-232/422/485) is enabled.

See the following table as the command and relevant information for configuring transceiver mode by different communication protocols. You can also refer to Sample Code *(on page 58)* and use the keyword **configTransceiverMode** to locate the corresponding configuration snippet.

Important:
Only **bits [4:3]** are used to select the UART transceiver mode. All other bits must remain unchanged to avoid unintended behavior.

Table 5-8 Transceiver Mode Configuration

Command	Value	Description	Outcome
`i2cset -f -y 1 0x4a 0x3c <value>`	`0bxxx00xxx`	Invalid	UART transceiver mode configured
	`0bxxx01xxx`	RS232	
	`0bxxx11xxx`	RS422	
	`0bxxx10xxx`	RS485	

5.1.4.1. Setting Data Transceiver Direction for COM port

This section describes how to control the data transceiver direction for RS-232, RS-422, and RS-485 communication protocols. You can also refer to Sample Code *(on page 58)* for more information.

- RS-232:

  No direction control is needed.

- RS-422:

  Initial state: Set transceiver to **Transmit Mode** by setting `SYSFS_UART0_RTS = H`

  No further changes are required during communication, as RS-422 supports full-duplex transmission.

- RS-485:

Initial state: Set transceiver to **Receive Mode** by setting `SYSFS_UART0_RTS = L`

Data transmit procedure:

1. Switch to **Transmit Mode** by setting `SYSFS_UART0_RTS = H`
2. **Send data**: `write(fd_sp0, &write_buf, strlen(write_buf));`
3. **Hold Transmit Mode** until transmission completes: `usleep(TRANSCEIVER_HOLDING_TIME);`

   Note:

   - TRANSCEIVER_HOLDING_TIME = 10 (unit: milliseconds)
   - Holding time calculation (approximate):

     {(64 bytes * 10 bits) / 115200 BPS} * 2 ≈ 10 ms

     - Assumes 64 bytes of data
     - "8n1" framing = 10 bits per byte
     - BPS = bit per second
     - * 2 = +50% timing tolerance

4. Switch back to **Receive Mode** by setting `SYSFS_UART0_RTS = L`

5.1.4.2. Sample Code

This section provides the sample code (`UART_common.h`) for configuring the COM port.

```
#include <err.h>
#include <errno.h>
#include <fcntl.h>
#include <stdbool.h>
#include <stdio.h>
#include <stdint.h>
#include <stdlib.h>
#include <string.h>
#include <pthread.h>
#include <termios.h>
#include <unistd.h>
#include <asm-generic/ioctls.h>        /* TIOCGRS485 + TIOCSRS485 ioctl definitions */
#include <linux/i2c.h>
#include <linux/i2c-dev.h>
#include <linux/ioctl.h>
#include <linux/serial.h>
#include <linux/types.h>
#include <sys/ioctl.h>
#include <sys/poll.h>

#define TIOCM_DTR  0x002
#define TIOCM_RTS  0x004
#define TIOCM_ST   0x008
#define TIOCM_SR   0x010
#define TIOCM_CTS  0x020
#define TIOCM_CAR  0x040
#define TIOCM_RNG  0x080
#define TIOCM_DSR  0x100
#define TIOCM_CD   TIOCM_CAR
#define TIOCM_RI   TIOCM_RNG

typedef enum {
         RS232, RS422, RS485
} UART_MODE_t;

/* Transceiver */
const char *MCU_I2C_DEVNODE_PATH = "/dev/i2c-1";
const uint8_t MCU_I2C_ADDR = 0x4a;
const uint8_t MCU_I2C_REG_NUM = 0x3c;

const uint8_t UART0_MODE_MASK = 0x18;         /* Valid bit[3:4] */
```

```
const uint8_t UART0_MODE_RS232_VAL = 0x08;    /* Valid bit[3:4] */
const uint8_t UART0_MODE_RS422_VAL = 0x18;    /* Valid bit[3:4] */
const uint8_t UART0_MODE_RS485_VAL = 0x10;    /* Valid bit[3:4] */

/* SYSFS */
#define SYSFS_UART0_RTS                       "/dev/uart0-rts"
#define SYSFS_UARTX_RTS_GPIO_VAL_H    "1"
#define SYSFS_UARTX_RTS_GPIO_VAL_L    "0"

const uint8_t TRANSCEIVER_HOLDING_TIME = 10;          /* Unit: mS, 64bytes @ 115.2kbps = 5.5mS */

int fd_sp0 = 0;
bool finish = false;                           /* Meet the conditions for exit */
char *UART_devnode;

void configTransceiverMode(const uint8_t MASK, const uint8_t VAL) {
          char i2cbuf[3] = {0};
          int fd_MCU;

          fd_MCU = open(MCU_I2C_DEVNODE_PATH, O_RDWR);
          if (fd_MCU < 0)
                   err(errno, "Tried to open '%s'", MCU_I2C_DEVNODE_PATH);

          if (0 > ioctl(fd_MCU, I2C_SLAVE, MCU_I2C_ADDR))
                   err(errno, "Tried to set device address '0x%02x'", MCU_I2C_ADDR);

          /* Get GPIO of MCU state */
          i2cbuf[0] = MCU_I2C_REG_NUM;
          if (write(fd_MCU, i2cbuf, 1) != 1) {
                   perror("Failed to write address of transceiver mode");
                   exit(-EIO);
          }

          if (read(fd_MCU, i2cbuf, 1) != 1) {
                   perror("Failed to read transceiver mode");
                   exit(-EIO);
          }

          /* Set mode of related transceiver */
          i2cbuf[1] = i2cbuf[0] & ~(MASK);
          i2cbuf[1] = i2cbuf[1] | (VAL & MASK);
          i2cbuf[0] = MCU_I2C_REG_NUM;
          if (write(fd_MCU, i2cbuf, 2) != 2) {
                   perror("Failed to write to the I2C device");
                   exit(-EIO);
          }
          /* Using I2C Read, equivalent of i2c_smbus_read_byte(file) */
          if (read(fd_MCU, i2cbuf, 1) != 1) {
                   perror("Failed to read to the I2C device");
                   exit(-EIO);
          }
}
void configUART_mode(void) {
         struct serial_rs485 rs485conf = {0};
         struct termios tio;

         memset(&tio, 0, sizeof(tio));
         tio.c_iflag = 0;
         tio.c_oflag = 0;
         tio.c_cflag = CS8 | CREAD | CLOCAL;
         tio.c_lflag = 0;
         /* Block mode */
         tio.c_cc[VMIN] = 1;
         tio.c_cc[VTIME] = 5;
         tio.c_cflag |= CRTSCTS;                      /* Use RTS/CTS */

         cfsetospeed(&tio, B115200);

         fd_sp0 = open(UART_devnode, O_RDWR);
         if (fd_sp0 < 0) {
                  printf("Error: Can't open: %s\n", UART_devnode);
#if 1
         /* Read the current state of the RS-485 options with ioctl. */
         if (ioctl(fd_sp0, TIOCGRS485, &rs485conf) < 0)
                  printf("Error: TIOCGRS485 ioctl not supported.\n");
         /*
          * On Linux RTS is set when a tty device is open. This is not good
          * when RTS is used for RS485. In this case RTS will prevent the
```

```
          * transceiver to receive any data. So here RTS will be cleared.
          */
         int rtsflag = TIOCM_RTS;
         int ret = ioctl(fd_sp0, TIOCMBIC, &rtsflag);
         if (0 > ret) {
                 printf("Can not set rts pint, ioctl failed %d, (errno: %d)\n", ret, errno);
                 exit(-EIO);
         }

         /* Enable RS485 mode: */
         rs485conf.flags |= SER_RS485_ENABLED;

         /* Set logical level for RTS pin equal to 1 when sending: */
         rs485conf.flags |= SER_RS485_RTS_ON_SEND;

         /* set logical level for RTS pin equal to 0 after sending: */
         rs485conf.flags &= ~(SER_RS485_RTS_AFTER_SEND);

         /* Set rts delay before send, if needed: */
         rs485conf.delay_rts_before_send = 0;

         /* Set rts delay after send, if needed: */
         rs485conf.delay_rts_after_send = 0;

         /* Set this flag if you want to receive data even whilst sending data */
         rs485conf.flags |= SER_RS485_RX_DURING_TX;

         /* Write the current state of the RS-485 options with ioctl. */
         if (ioctl(fd_sp0, TIOCSRS485, &rs485conf) < 0) {
                 //printf("Error: TIOCSRS485 ioctl not supported.\n");
         }
         /* set the termios so the rs485 settings are applied */
         tcsetattr(fd_sp0, TCSANOW, &tio);
         fcntl(fd_sp0, F_SETFL, 0);
#endif
}

void* UART_read_data(void *arg) {
         while (!finish) {
                 char read_buf[64] = {0};
                 int n = read(fd_sp0, &read_buf, sizeof(read_buf));

                 if (n < 0)
                         printf("Error reading from %s\n", UART_devnode);
                 else
                         printf("UART reading: %s\n", read_buf);
                 usleep(100 * 1000);
         }
         pthread_exit(NULL);
}

void UART_write_data_RTSctl(const UART_MODE_t UARTmode, const char *RTS_SYSFS) {
          int fd_RTS = 0;

          if (NULL != RTS_SYSFS) {
                  fd_RTS = open(RTS_SYSFS, O_RDWR);
                  if ( 0 > fd_RTS) {
                          printf("Error: Can't open: %s\n", RTS_SYSFS);
                          exit(-EINVAL);
                  }
          }
          switch (UARTmode) {
                  case RS232:
                          break;
                  case RS422:
                          if (0 > write(fd_RTS, SYSFS_UARTX_RTS_GPIO_VAL_H, sizeof(SYSFS_UARTX_RTS_GPIO_VAL_H))) {
                                  printf("Error: Can't write: %s\n", RTS_SYSFS);
                                  exit(-EINVAL);
                          }
                          break;
                  case RS485:
                          if (0 > write(fd_RTS, SYSFS_UARTX_RTS_GPIO_VAL_L, sizeof(SYSFS_UARTX_RTS_GPIO_VAL_L))) {
                                  printf("Error: Can't write: %s\n", RTS_SYSFS);
                                  exit(-EINVAL);
                          }
                          break;
                  default:
                          printf("Error: UART mode invalid: %d\n", UARTmode);
```

```
                        exit(-EINVAL);
        }
        printf("Please enter:\n");
        do {
                char write_buf[128] = {0};

                fgets(write_buf, sizeof(write_buf), stdin);          /* keyboard key-in */
                if(strcmp(write_buf, "q\n") == 0 || strcmp(write_buf, "Q\n") == 0)
                        finish = true;

                if (finish) {
                        usleep(1000 * 1000);                        /* Waiting reading thread finish */
                        break;                                      /* while (true) */
                }

                if (RS485 == UARTmode)
                        write(fd_RTS, SYSFS_UARTX_RTS_GPIO_VAL_H, sizeof(SYSFS_UARTX_RTS_GPIO_VAL_H));/* Assume error
 free */

                write(fd_sp0, &write_buf, strlen(write_buf));
                if (RS485 == UARTmode) {
                        /* Holding transceiver as transmitting mode, waiting data transmit finished */
                        usleep(TRANSCEIVER_HOLDING_TIME * 1000);
                        write(fd_RTS, SYSFS_UARTX_RTS_GPIO_VAL_L, sizeof(SYSFS_UARTX_RTS_GPIO_VAL_L));/* Assume error
 free */
                }
        } while (true);
        printf("Closing %s\n", UART_devnode);
        if (fd_sp0)
                if (close(fd_sp0) < 0)
                        printf("Error: Can't close: %s\n", UART_devnode);
        if (fd_RTS)
                if (close(fd_RTS) < 0)
                         printf("Error: Can't close: %s\n", RTS_SYSFS);
}
void UART_write_data(const UART_MODE_t UARTmode) {
UART_write_data_RTSctl(UARTmode, NULL);
}
```

5.1.4.3. RS-232 Test Program

This section provides a sample test program (`rs232-demo.c`) for validating RS-232 communication.

```
#include "UART_common.h"

int main(int argc, char* argv[]) {
        UART_devnode="/dev/ttyTHS0";

        configTransceiverMode(UART0_MODE_MASK, UART0_MODE_RS232_VAL);
        configUART_mode();

        pthread_t UART_read_thread;
        if (pthread_create(&UART_read_thread, NULL, UART_read_data, NULL) != 0)
                printf("Error: pthread create UART read thread\n");

        UART_write_data(RS232);

        return 0;
}
```

5.1.4.4. RS-422 Test Program

This section provides a sample test program (`rs422-demo.c`) for validating RS-422 communication.

```
#include "UART_common.h"

int main(int argc, char* argv[]) {
        UART_devnode="/dev/ttyTHS0";

        configTransceiverMode(UART0_MODE_MASK, UART0_MODE_RS422_VAL);
        configUART_mode();

        pthread_t UART_read_thread;
        if (pthread_create(&UART_read_thread, NULL, UART_read_data, NULL) != 0)
                printf("Error: pthread create UART read thread\n");
```

```
        UART_write_data_RTSctl(RS422, SYSFS_UART0_RTS);

        return 0;
}
```

5.1.4.5. RS-485 Test Program

This section provides a sample test program (`rs485-demo.c`) for validating RS-485 communication.

```
#include "UART_common.h"

int main(int argc, char* argv[]) {
        UART_devnode="/dev/ttyTHS0";

        configTransceiverMode(UART0_MODE_MASK, UART0_MODE_RS485_VAL);
        configUART_mode();

        pthread_t UART_read_thread;
        if (pthread_create(&UART_read_thread, NULL, UART_read_data, NULL) != 0)
                printf("Error: pthread create UART read thread\n");

        UART_write_data_RTSctl(RS485, SYSFS_UART0_RTS);

        return 0;
}
```

5.1.5. CAN FD Configuration

The name of the CAN FD port: can0

See the following table as the available commands list for the CAN FD port.

Table 5-9 Initialization & Setup

Command	Description	Outcome
`sudo can_set`	Enable the CAN bus mode	CAN Bus set completed
`sudo ip link set down can0`	Disable the transmission function of the CAN bus	N/A
`sudo ip link set can0 type can bitrate 1000000 dbitrate 2000000 fd on`	Set the standard bit rate for the CAN bus as 1 Mbps and the data phase bit rate for CAN FD (Flexible Data Rate) as 2 Mbps. Enable CAN FD for faster transmission rate and larger data payloads (up to 64 bytes per frame).	N/A
`sudo ip link set up can0`	Enable the transmission function of the CAN bus	N/A

Table 5-10 Data Reception

Command	Description	Outcome
`candump can0 &`	Receive data	N/A

Table 5-11 Data Transmission

Command	Description	Outcome
`cansend can0 123#abcdabcd`	Send data	N/A

5.1.6. DIO Configuration

IBOX-602P-IP66 supports 3 programmable digital inputs (DI) and 4 digital outputs (DO), which can be configured to communicate or exchange data with the connected peripheral devices.

Note:
The DIO pins will be reset to default settings after a cold boot.

The general formats of the DIO commands are as follows:

`i2cget -f -y <i2c_num> <device_addr> <reg_addr>`: Used to **check** the current state of certain DI or DO channel.

`i2cset -f -y <i2c_num> <device_addr> <reg_addr> <value>`: Used to **change or save** the settings of certain DI or DO channel.

See the following as the DI data register table:

Table 5-12 DI Data Register – 0x30

Bit	Pin	Value
2	DI_3	• Low: 0 • High: 1
1	DI_2	
0	DI_1	

See the following as the DO data register table:

Table 5-13 DO Data Register – 0x31

Bit	Pin	Value
3	DO_4	• Low: 0 • High: 1
2	DO_3	
1	DO_2	
0	DO_1	

5.1.6.1. DIO Commands

See the following table as the available commands list for the DIO port.

Table 5-14 Checking Digital Input (DI) Status

Command	Outcome	Description
`i2cget -f -y 1 0x4A 0x30`	Value of DI status	Get the status of all the digital input pins

Table 5-15 Examples of Configuring Digital Outputs (DO)

Command	Value	Description
`i2cset -f -y 1 0x4A 0x31 <value>`	`0xf`	Set DO1 ~ DO4 as High (1111)
	`0x0`	Set DO1 ~ DO4 as Low (0000)
	`0x1`	Set DO1 as High (0001: DO1 = 1)
	`0x2`	Set DO2 as High (0010: DO2 = 1)
	`0x4`	Set DO3 as High (0100: DO3 = 1)
	`0x8`	Set DO4 as High (1000: DO4 = 1)

5.1.7. SIM Card Switch Commands

The system provides a dual Nano SIM card tray within a single slot for dual SIM installation. See the following table as the commands to switch between the installed SIM cards.

Note:
By default, SIM1 is selected for use.

Command	Description	Outcome
`sim_selection 1`	Switch to SIM1	The SIM card placed on the tray surface marked **SIM1** will be selected for use.
`sim_selection 2`	Switch to SIM2	The SIM card placed on the tray surface marked **SIM2** will be selected for use.

5.1.8. PoE Configuration Commands

Power over Ethernet (PoE) is supported on both ETH1 and ETH2 Ethernet ports. See the following commands for how to configure the PoE settings.

Note:
By default, PoE is enabled on both ports.

Table 5-16 PoE Commands

Command	Description	Outcome
`enPOEPort 0`	Disable PoE on both ports	ETH1 and ETH2 will not supply power.
`enPOEPort 1`	Enable PoE on ETH1 and disable it on ETH2.	Only ETH1 will supply power.
`enPOEPort 2`	Enable PoE on ETH2 and disable it on ETH1.	Only ETH2 will supply power.
`enPOEPort 3`	Enable PoE on both ports	ETH1 and ETH2 will supply power.

Important:
Enter the following command to preserve the specified **PoE** or **SIM card switch** configuration. If this command is not executed, after the system reboots, the specified configuration will be restored to its default state.

Table 5-17 Saving the Setting

Command	Description	Outcome
`i2cset -f -y 1 0x4a 0x28 0x02`	Save the setting.	The specified configuration is persistent and will remain unchanged after system reboot.

5.2. System Recovery

This section describes how to recover the system when needed.

5.2.1. Hardware and System Requirements

A host computer and some accessories are required to perform a system recovery. Before you begin, check the following list as the hardware and system requirements for the host computer and accessories.

Host Computer:

- Memory size: 4GB or above
- Storage space: 256GB or above
- Recommended OS: Ubuntu 20.04/22.04

Accessories:

- A USB Type-C cable
- An HDMI cable
- A monitor supporting HDMI inputs

5.2.2. Configuring Host Computer

Follow the procedures below to configure the host computer.

1. Connect the host computer to the Internet.
2. Open the terminal window.
3. Enter the following commands to install the required dependencies.

```
$ sudo apt update
$ sudo apt install -y abootimg bzip2 libxml2-utils nfs-kernel-server sshpass
```

5.2.3. Downloading BSP Image & Checksum Files

You must install a board support package (BSP) image on the host computer before performing the system recovery. Follow the procedures below to download the BSP image.

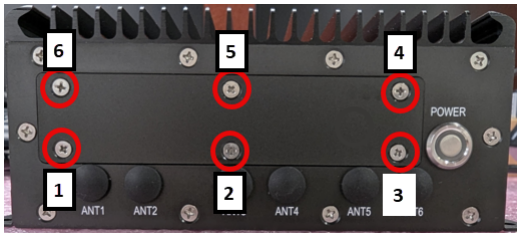
1. Download the BSP image and accompanied SHA256 checksum files from SINTRONES® website or contact our sales representative for the files.
2. Save the downloaded files in the host computer.

Note:
Do not extract the compressed `.tbz2` file.

5.2.4. Setting System in Recovery Mode

Follow the procedures to set the system in recovery mode and connect it to the host computer.

1. Ensure the system is powered off.
2. Remove the 6 screws in the order indicated in the figure below to remove the protection cover on the rear panel.



3. Locate the **SERVICE** Type-C port on the rear panel.

4. Plug one end of a USB Type-C cable into the port, and then plug the other end of the cable to the host computer.
5. Locate the **HDMI** port on the rear panel.

6. Connect the system to a monitor via an HDMI cable for later use (to configure the recovered system).
7. Locate the **RECOVERY** button on the rear panel.

8. Press and hold the **RECOVERY** button, at the same time, connect the system to a power source and enable ignition if any.

   Note:
   If the system is not connected with any ignition system, simply press the power button.

9. After powering on the system, hold the **RECOVERY** button for at least 10 seconds, and then release it.

5.2.5. Executing Recovery

Follow the steps below ro run the recovery commands on the host computer and flash the BSP image to the system.

1. On the host computer, enter the following command in the terminal to check if the system is set in recovery mode.

```
$ lsusb | grep NVIDIA
```

2. The following strings with the VID/PID info appear, indicating the system is in recovery mode. If these strings do not appear, repeat the steps in Setting System in Recovery Mode *(on page 68)*.

   Figure 5-2 An Example of Strings with the VID/PID Info

   Note:
   The VID/PID varies depending on different models.

   Important:
   Ensure the system is in recovery mode before you peform the flashing process.

3. Navigate to the folder that contains the downloaded BSP image and SHA256 checksum files.
4. Enter the following command to verify the integrity of the BSP image.

   ```
   $ sha256sum -c <ProjectName-KernelVersion-JetpackVersion-BuildSystem-BuildVersion-PackageVersion>.tbz2.sha256
   ```

   Note:
   Replace the texts within the brackets with the actual BSP file name, for example, `IBOX-602-5.15.148-6.2-ubuntu22.04-R1.00-00`

   Tip:
   After entering the first few characters of a command or file name, you can enter the **[tab]** key to auto-complete the command or filename.

5. Enter the following command to decompress the BSP image.

   ```
   $ sudo tar -jxvf <ProjectName-KernelVersion-JetpackVersion-BuildSystem-BuildVersion-PackageVersion>.tbz2
   ```

6. After the BSP image is decompressed, a folder will be automatically generated. Its name matches the BSP file name and follows this format:

   ```
   <ProjectName-KernelVersion-JetpackVersion-BuildSystem-BuildVersion-PackageVersion>
   ```

7. Enter the following command to navigate to the folder.

   ```
   $ cd <ProjectName-KernelVersion-JetpackVersion-BuildSystem-BuildVersion-PackageVersion>
   ```

8. Enter the following command to flash the BSP image to the system.

   ```
   $ sudo ./ibox-602_nx_flash.sh
   ```

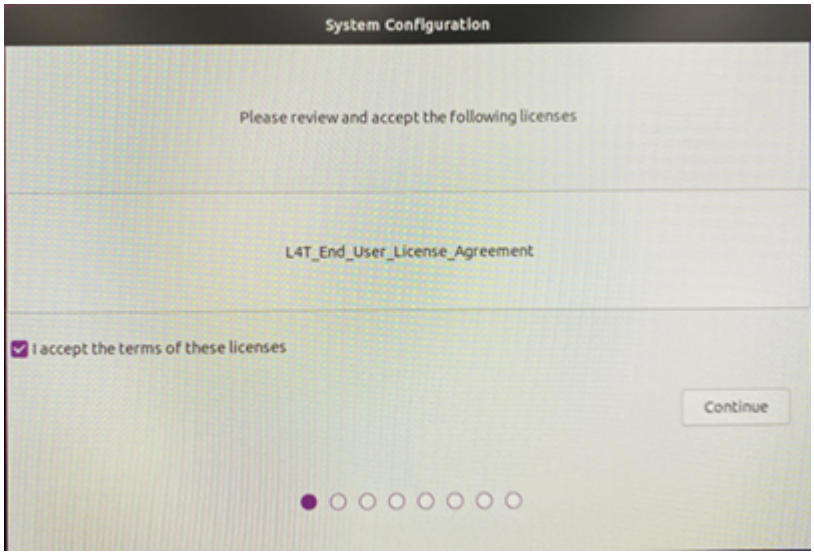
9. When the BSP image is successfully flashed to the system, the system will reboot and the configuration window will then appear on the connected monitor screen.

   Important:
   In order to maintain the system's IP66 waterproof funcationality, ensure you fasten the screws by the specified order given in the figure in **Setting the System in Recovery Mode: Step 2** *(on page 68)* when fastening the protection cover back onto the rear panel.

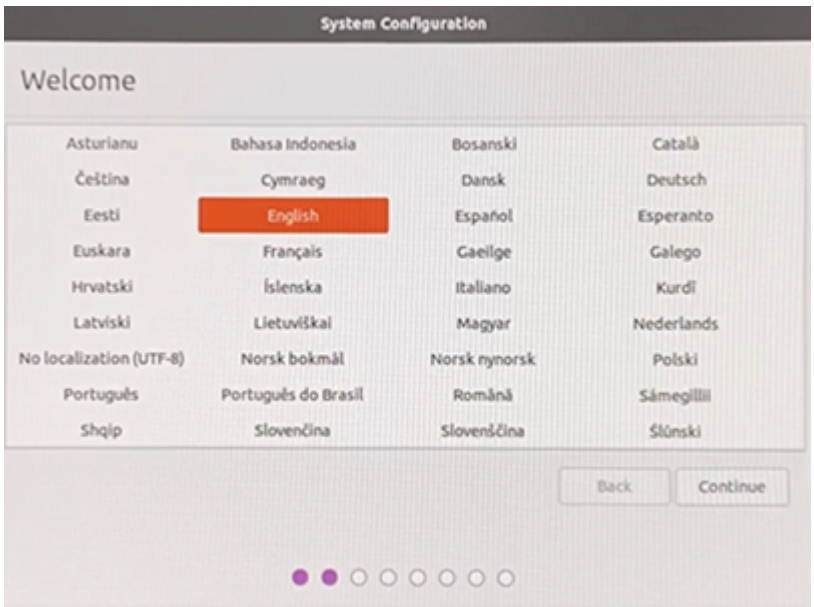
5.2.6. Configuring the Recovered System

After the system is recovered, initial settings must be configured prior to using the system. Follow the procedures below to start the configuration.

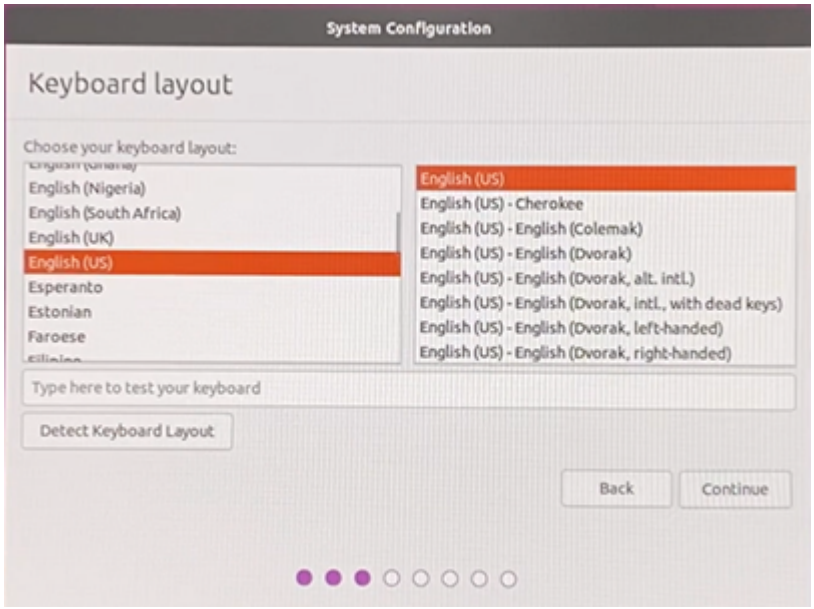
1. After the system reboot, the **System Configuration** wizard appears to guide users to complete the initial basic settings for the Linux for Tegra (LT4) platform.
2. Select **I accept the terms of these licenses** and then select **Continue**.



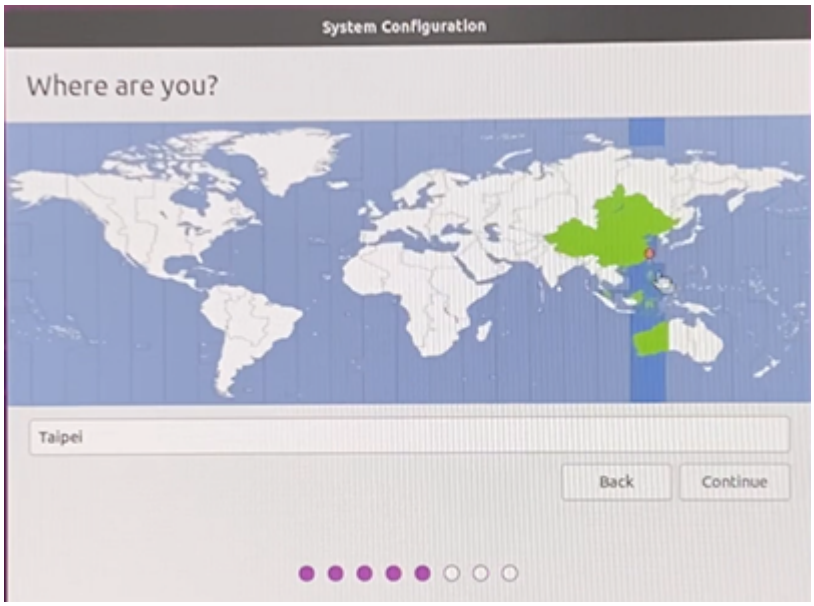
3. Select the preferred language setting and then select **Continue**.



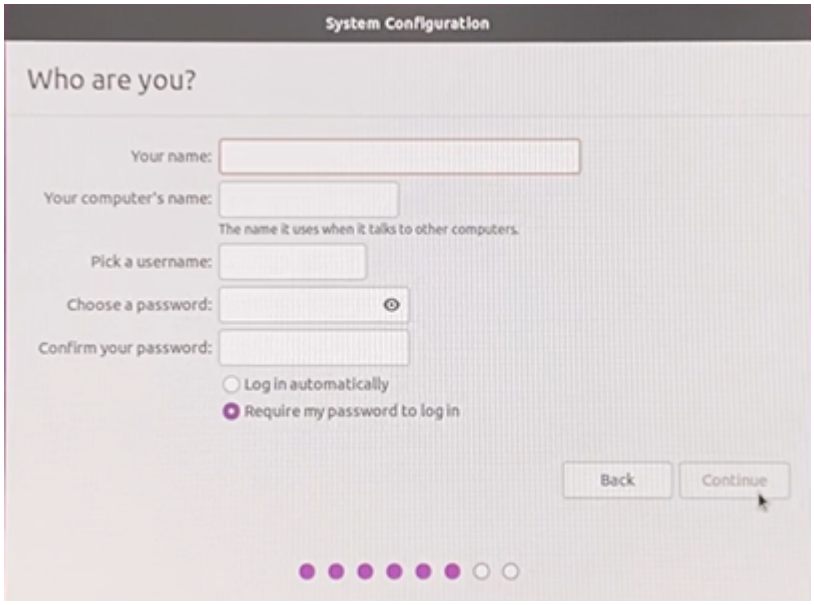
4. Select the preferred keyboard layout and then select **Continue**.



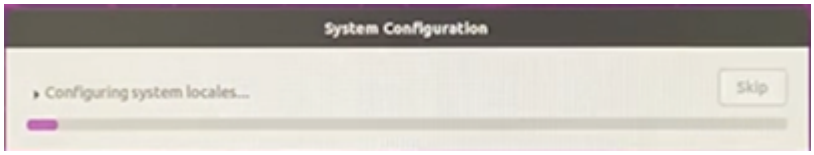
5. Select your location and then select **Continue**.

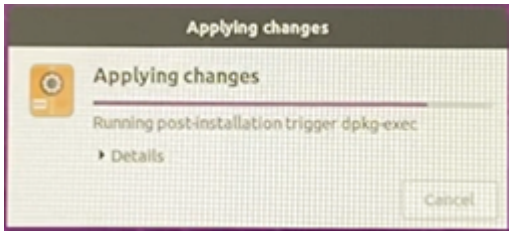


6. Specify the credentials such as a username and a password to create a user account. It is suggested to select **Require my password to log in** for security reasons. Select **Continue** to proceed to next step.



7. Choose **Install Chromium Browser** or **Do not install...** based on your needs, and then select **Continue**.
8. The system starts running the specified configuration and will reboot to complete the system configuration.





9. The **NVIDIA** logo appears and then the restored system will enter the welcome screen, ready to use now.

6. Appendix

Table 6-1 Decimal to Hexadecimal

Time	0	1	2	3	4	5	6	7	8	9
0	None	0x01	0x02	0x03	0x04	0x05	0x06	0x07	0x08	0x09
10	0x0a	0x0b	0x0c	0x0d	0x0e	0x0f	0x10	0x11	0x12	0x13
20	0x14	0x15	0x16	0x17	0x18	0x19	0x1a	0x1b	0x1c	0x1d
30	0x1e	0x1f	0x20	0x21	0x22	0x23	0x24	0x25	0x26	0x27
40	0x28	0x29	0x2a	0x2b	0x2c	0x2d	0x2e	0x2f	0x30	0x31
50	0x32	0x33	0x34	0x35	0x36	0x37	0x38	0x39	0x3a	0x3b