Atlas 200 Developer Kit

User Guide

Issue	01
Date	2020-05-30

HUAWEI TECHNOLOGIES CO., LTD.

Trademarks and Permissions

Notice

The purchased products, services and features are stipulated by the contract made between Huawei and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

Contents

1 Introduction

This document describes the preparations for using the Atlas 200 Developer Kit (Atlas 200 DK) to run AI applications, including preparing the SD card, managing the basic information about the developer board on Mind Studio, and configuring the cross compilation environment on Mind Studio.

2 Preparing Accessories and the Development Server

Preparing Accessories

Table 2-1 lists the accessories required for using the Atlas 200 DK. You need to purchase them in advance.

Table 2-1 Accessories

Name		Description	Recommended Model
SD card		Used to create the startup system for the Atlas 200 DK developer board	The following tested SD card models are recommended: ● Samsung UHS-I U3 Class 10 (64 GB) ● Kingston UHS-I U1 Class 10 (64 GB)
Card reader (recommended) or jumper cap	Card reader	For details about how to create an SD card using a card reader, see **3.5.1 Creating an SD Card When a Card Reader Is Available**.	Card reader supporting USB 3.0
	Jumper cap	For details about how to create an SD card by using a jumper cap, see **3.5.2 Creating an SD Card When a Card Reader Is Unavailable**.	2.54 mm jumper cap

Name	Description	Recommended Model
Type-C cable	Used to connect to USB port of the server where Mind Studio is located. For details, see **3.6 Connecting the Atlas 200 DK Developer Board to the Ubuntu Server**.	USB 3.0 Type-C cable
Network cable	Used to connect to the network port of the server where Mind Studio is located. For details, see **3.6 Connecting the Atlas 200 DK Developer Board to the Ubuntu Server**.	Common network cable with RJ45 connectors
Camera	Used to connect to the Atlas 200 DK for obtaining video streams. For details, see **3.4 Installing a Camera**.	The Atlas 200 DK is compatible with Raspberry Pi cameras, which require yellow flat ribbon cables.
(Optional) Camera support	Used to fix the camera. For details, see **3.4 Installing a Camera**.	Raspberry Pi transparent camera support
(Optional) Serial cable	Used to view the startup logs through the serial port when the Atlas 200 DK startup indicator is abnormal, or the SD card is successfully created but the Mind Studio installation server fails to be accessed. For details, see **8.5 What Do I Do If the Developer Board Cannot Connect to the UI Host?**.	USB-to-TTL serial cable with 3.3 V interface level

Preparing a Server

Prepare a server running the Ubuntu x86 OS. The server has the following functions:

- Enables you to connect the card reader or Atlas 200 DK to the Ubuntu server over a USB cable during SD card creation, helping prepare a system boot disk for the Atlas 200 DK. For details, see **3.5 Creating an SD Card**.
- Supports installation of the development tool. For details, see the *Ascend 310 Mind Studio Installation Guide (Ubuntu, x86)*.

The Ubuntu OS version must be 16.04.3. Download the corresponding version software from **http://old-releases.ubuntu.com/releases/16.04.3/** and install it. You can download the desktop version **ubuntu-16.04.3-desktop-amd64.iso** or server version **ubuntu-16.04.3-server-amd64.iso**.

3 Setting Up the Hardware Environment

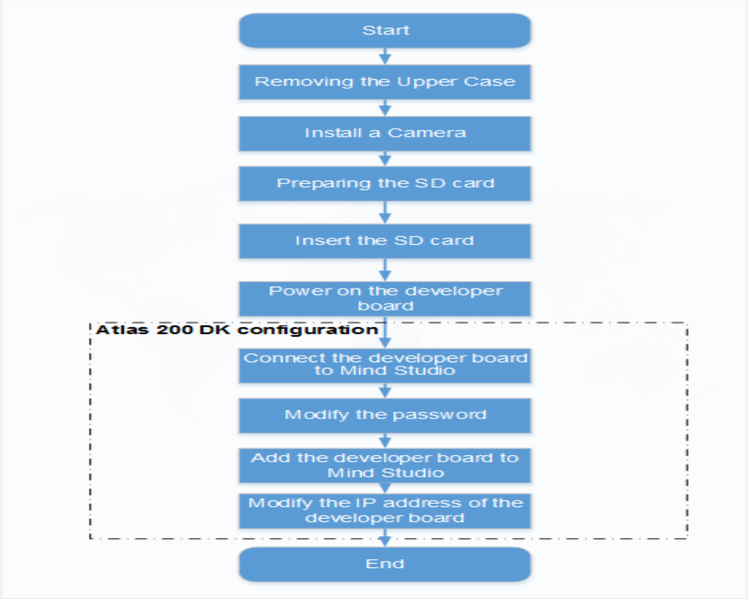
3.1 Precautions for Using the Board

- Before powering on the board, ensure that the Atlas 200 AI accelerator module is properly installed. Otherwise, the board cannot be powered on properly.
- If you need to use the camera or internal ports, remove the top cover when the power is off.
- When using a non-standard power adapter, ensure that the power supply range and power meet the board requirements.
- The signal level is 3.3 V. When using different ports, ensure that the levels of different ports match. Otherwise, the board may be damaged.
- The 40-pin extension header does not have strict ESD protection design. Protect the board from electrostatic discharge and do not insert or remove the board without disconnecting the power supply.

3.2 Usage Process

Figure 3-1 shows the process of using the Atlas 200 DK.

Figure 3-1 Atlas 200 DK usage process


Start
Removing the Upper Case
Install a Camera
Preparing the SD card
Insert the SD card
Power on the developer board
Atlas 200 DK configuration
Connect the developer board to Mind Studio
Modify the password
Add the developer board to Mind Studio
Modify the IP address of the developer board
End

NOTE

Use this process when no SD card comes with the board. If an SD card comes with the board, start the process from powering on the Atlas 200 DK developer board.

3.3 Removing the Cover

NOTE

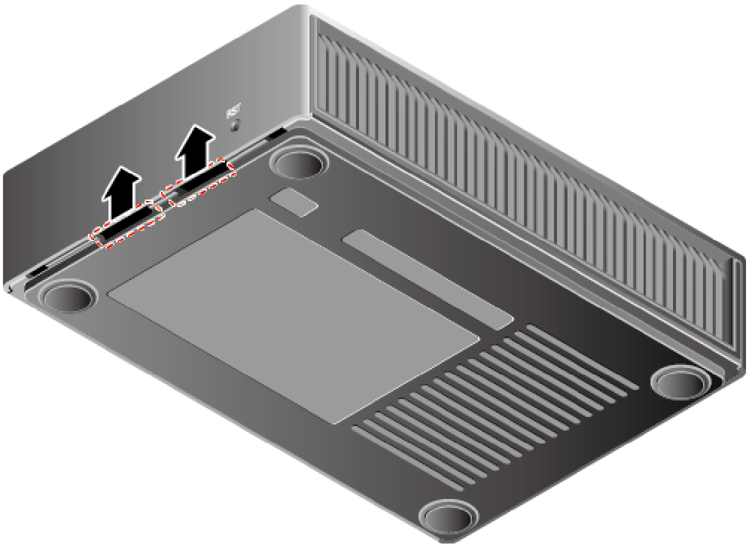
This operation needs to be performed when you need to use the camera or an internal interface.

Step 1 Check whether a camera cable is lead out from the Atlas 200 DK developer board.

- If yes, go to **Step 3**.
- If not, go to **Step 2**.

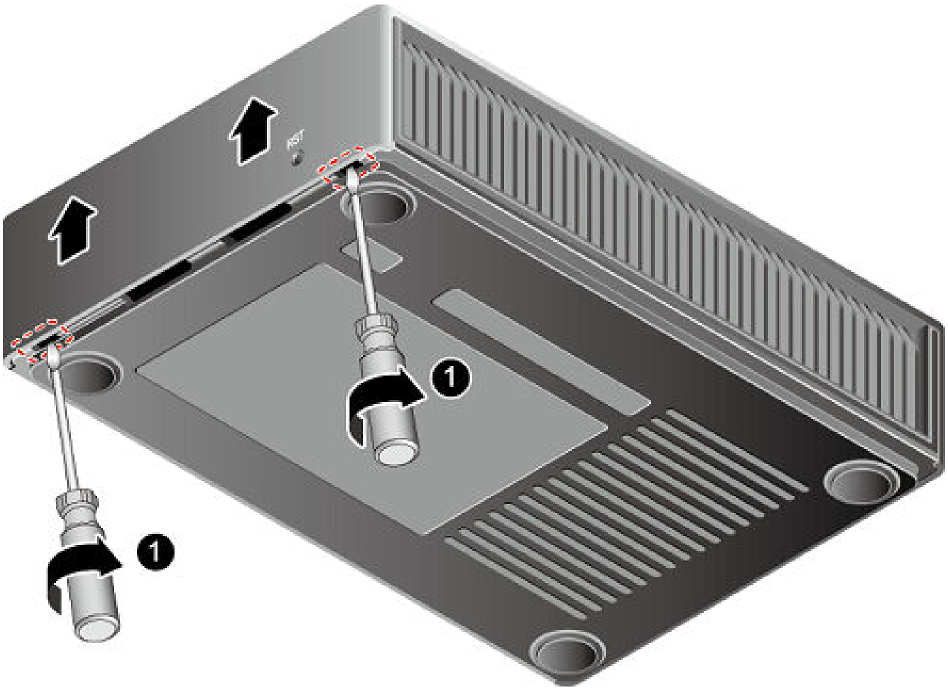
Step 2 If no camera cable is lead out from the Atlas 200 DK developer board, pull the plastic latch upwards to loosen the cover, as shown in **Figure 3-2**.

Figure 3-2 Removing the cover - 1



Step 3 If a camera cable is lead out from the Atlas 200 DK developer board, insert a flat-head screwdriver into the groove between the cover and the bottom plate, and rotate the screwdriver to pry off the cover, as shown in (1) in **Figure 3-3**.

Figure 3-3 Removing the cover - 2



Step 4 Remove the cover.

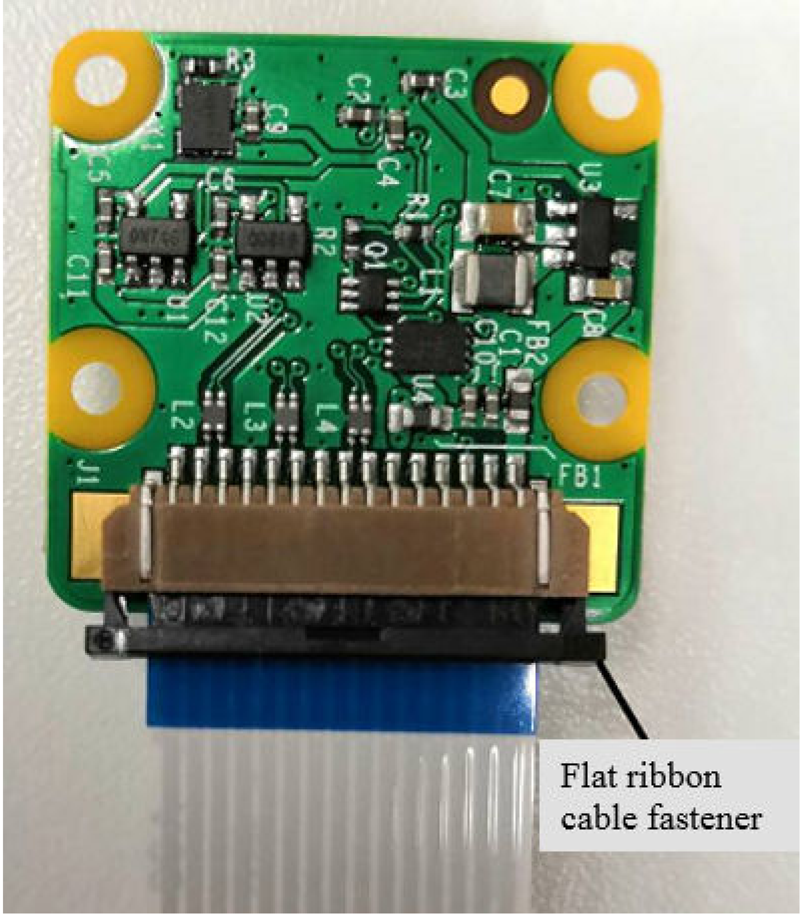
----End

3.4 Installing a Camera

Step 1 Replace the white flat ribbon cable of the Raspberry Pi camera with a yellow one.

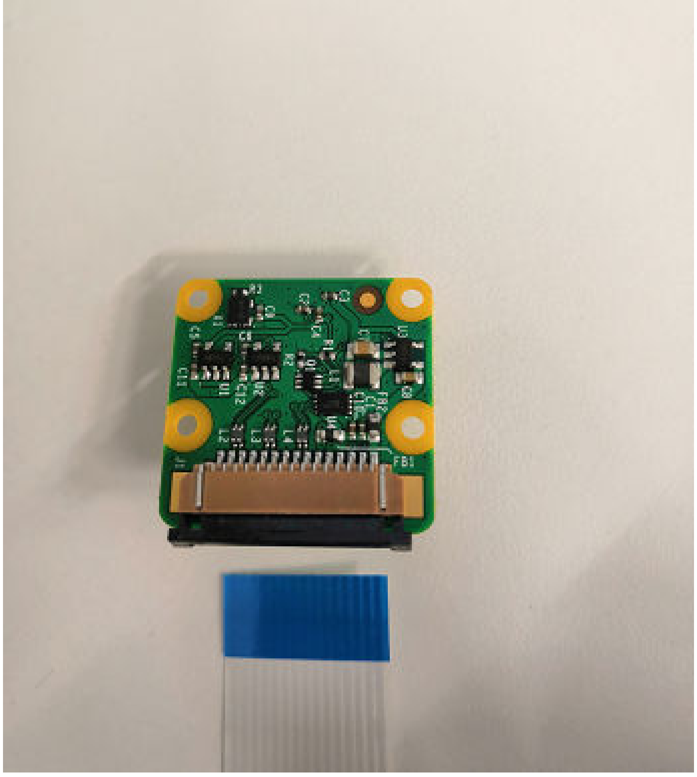
1. Remove the black flat ribbon cable fastener from the camera, as shown in **Figure 3-4**.

   Figure 3-4 Flat ribbon cable fastener

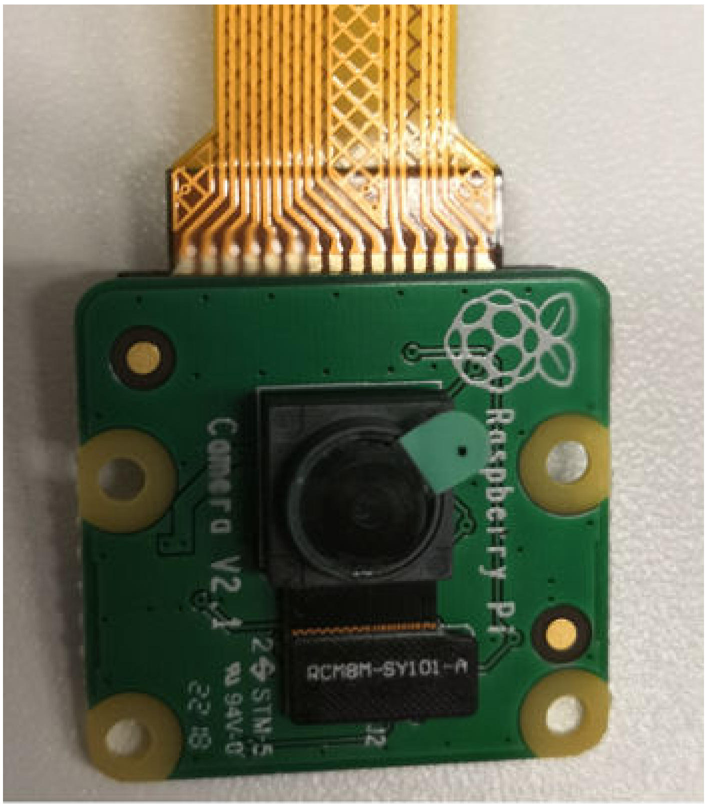
2. Take out the white flat ribbon cable, as shown in **Figure 3-5**.

Figure 3-5 White flat ribbon cable



3. Place the metal wire at the wider end of the yellow flat ribbon cable upwards and horizontally insert it into the cable slot of the camera until it is fastened, as shown in **Figure 3-6**.

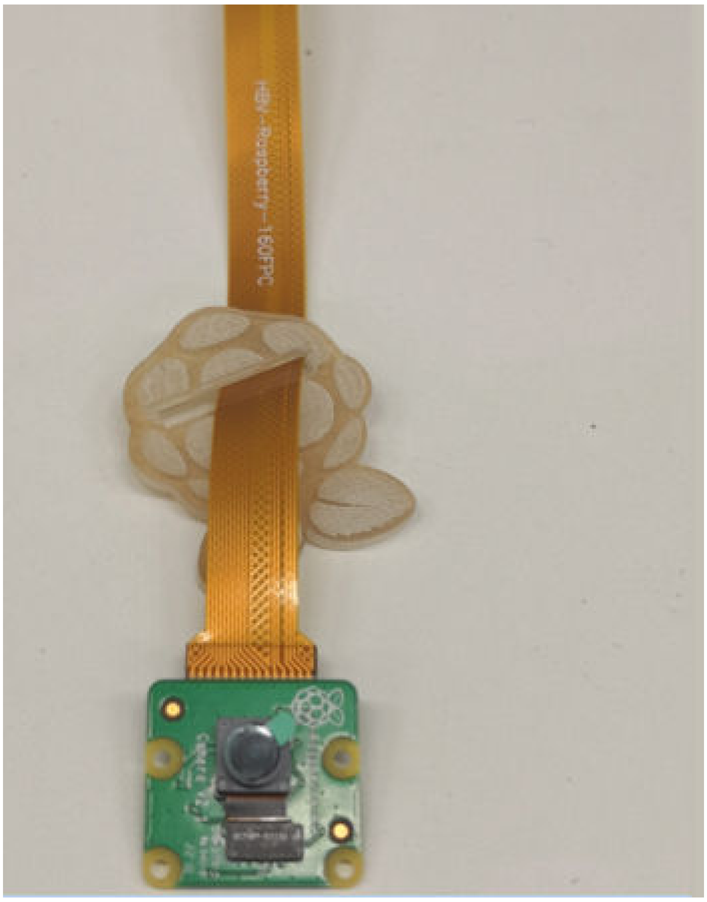
Figure 3-6 Connecting the camera



4. Secure the black flat ribbon cable fastener.

Step 2 Attach the fixing film to the yellow flat ribbon cable, as shown in **Figure 3-7**.

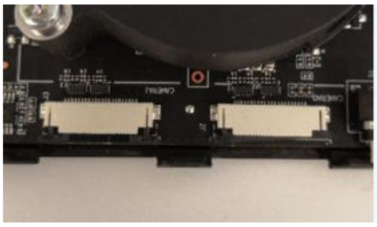
Figure 3-7 Installing the fixing film



Step 3 Connect the flat ribbon cable of the camera to the Atlas 200 DK developer board.

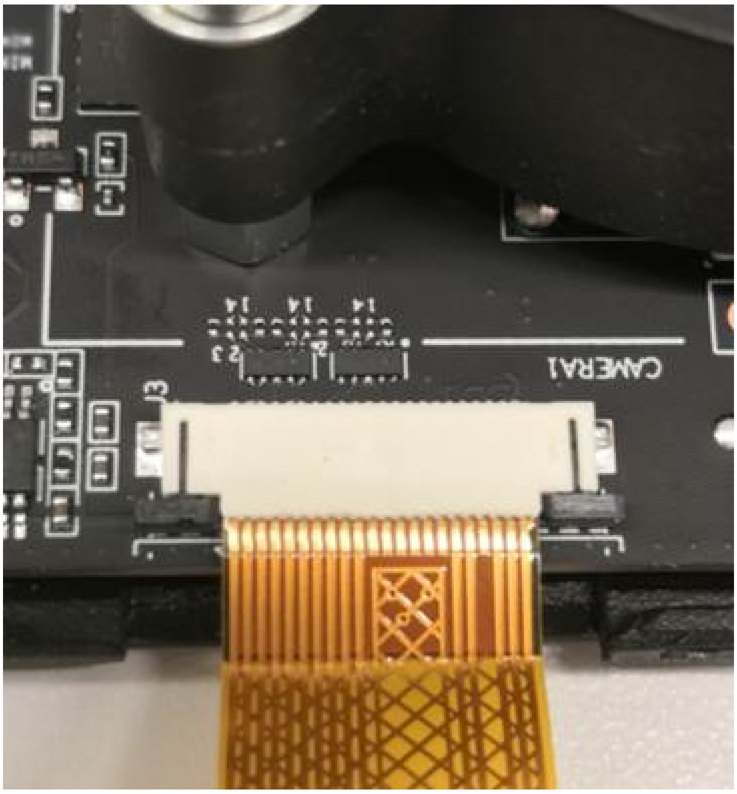
1. Remove the camera connector fastener from the Atlas 200 DK developer board, as shown in **Figure 3-8**.

   Figure 3-8 Removing the black fastener

2. Place the metal wire at the narrower end of the yellow flat ribbon cable upwards and horizontally insert it into the camera connector CAMERA0 or CAMERA1 on the Atlas 200 DK developer board until the cable is fastened. Insert the fastener, as shown in **Figure 3-9**.

Figure 3-9 Inserting the fastener

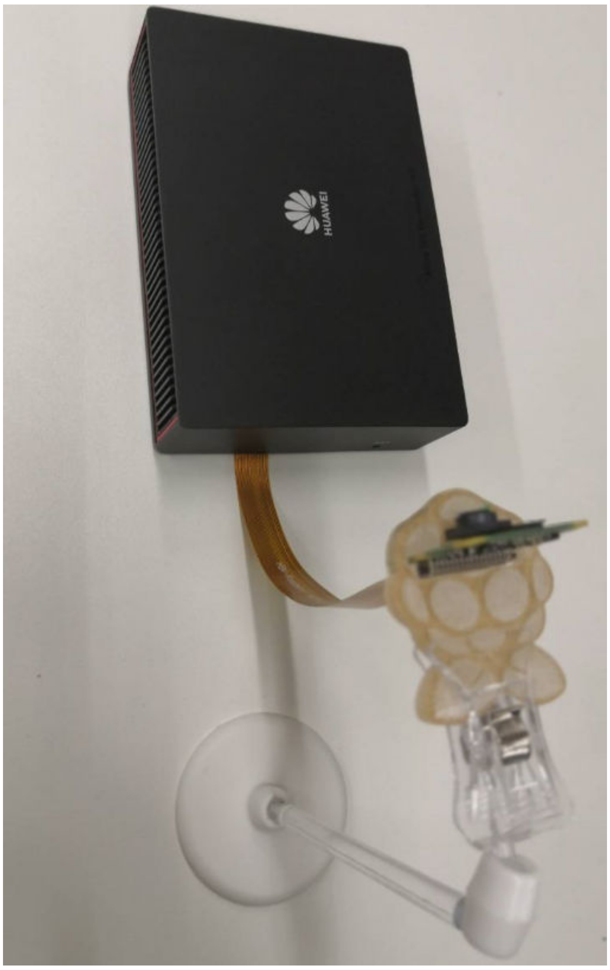


Step 4 Install the cover of the Atlas 200 DK developer board to the original position.

Step 5 Install the camera support.

1. Use the clip on the camera support to clamp the fixing film, as shown in **Figure 3-10**.

Figure 3-10 Installing the camera support



2. Place the camera on the camera support, as shown in the preceding figure. The base of the camera support has double-sided tape, which is a recommended method for securing the support on the desktop. This ensures that the camera is securely installed.

   NOTE

   Before using the camera, remove its protective film.

----End

3.5 Creating an SD Card

You can create a system boot disk for the Atlas 200 DK developer board by using the SD card creation function.

You can use either of the following methods to create an SD card:

- If a card reader is available, insert the SD card into the card reader, connect the card reader to the USB port on the PC where Ubuntu server is installed, and run the SD card creation scripts.
- If no card reader is available, insert the SD card into the card slot of the Atlas 200 DK developer board, use a jumper cap to connect the pins of the

developer board, connect the developer board to the USB port of the PC where Ubuntu server is installed, and run the SD card creation scripts.

NOTE

During SD card creation, the default user **HwHiAiUser** is automatically created for running applications.

3.5.1 Creating an SD Card When a Card Reader Is Available

Hardware Preparations

Prepare a 16 GB or larger SD card.

NOTE

The SD card will be formatted. Data must be backed up in advance.

Software Preparations

Obtain the **make_sd_card.py** and **make_ubuntu_sd.sh** SD card creation scripts, Atlas 200 DK running package, and Ubuntu package.

Table 3-1 shows the download packages.

Table 3-1 Software packages

Package	Name	Description
Entry script for creating an SD card	make_sd_card.py	Obtain the package from **https://gitee.com/HuaweiAscend/tools**.
Script for creating the SD card OS	make_ubuntu_sd.sh	Obtain the package from **https://gitee.com/HuaweiAscend/tools**.
Atlas 200 DK running package	mini_developerkit-xxx.rar The software integrity verification file is **mini_developerkit-xxx.rar.asc**.	Software package of the developer board. Verify the integrity of the downloaded software package by referring to **7.9 Verifying Software Integrity** .

Package	Name	Description
Ubuntu package	ubuntu-*xxx*-server-arm64.iso NOTE *xxx* indicates the version number.	The Ubuntu Linux version must be 16.04.3. Download the software of the required version from **http://old-releases.ubuntu.com/releases/16.04.3/** and install it. NOTE ● The Ubuntu package must be of the **server-arm64** version. ● The Ubuntu package version must be the same as the Ubuntu version of the device where Mind Studio is installed.

NOTE

Do not change the names of the downloaded files.

Procedure

Step 1 Insert the SD card into the card reader and connect the card reader to the USB port of the Ubuntu server.

Step 2 Run the following commands on the Ubuntu server to install the qemu-user-static, binfmt-support, YAML, and cross compiler:

su - root

Run the following command to update the source:

apt-get update

Run the following command to install the dependent libraries:

apt-get install qemu-user-static binfmt-support python3-yaml gcc-aarch64-linux-gnu g++-aarch64-linux-gnu

The versions of **gcc-aarch64-linux-gnu** and **g++-aarch64-linux-gnu** must be 5.4.0. You can use any versions of other dependent software packages. The default GCC version installed on Ubuntu 16.04.3 is 5.4.0.

Step 3 Upload **make_sd_card.py**, **make_ubuntu_sd.sh**, the Atlas 200 DK running package, and the Ubuntu package obtained from **Software Preparations** to any directory on the Ubuntu server as a common user, for example, */home/ascend/mksd*.

NOTE

The preceding scripts and software packages must be stored in the same directory.

The local creation of an SD card only allows software packages of the same version to be saved to the directory.

Step 4 Switch to the **root** user and go to the */home/ascend/mksd* directory where the SD card creation scripts are stored.

su - root

cd /home/ascend/mksd/

Step 5 (Optional) In the SD card making script, the default IP address of the USB NIC of the Atlas 200 DK developer board is **192.168.1.2**, and the default IP address of the NICDMA network card is **192.168.0.2**. If you want to change the default IP addresses, change them as follows:

Change the values of **NETWORK_CARD_DEFAULT_IP** and **USB_CARD_DEFAULT_IP** in the **make_sd_card.py** file.

- **NETWORK_CARD_DEFAULT_IP**: IP address of the NIC of the Atlas 200 DK developer board
- **USB_CARD_DEFAULT_IP**: IP address of USB NIC of the Atlas 200 DK developer board

Step 6 Run the SD card creation script.

1. Run the following command to query the name of the USB device where the SD card is located:

   fdisk -l

   The following uses the device name **/dev/sda** as an example:

2. Run the **make_sd_card.py** script.

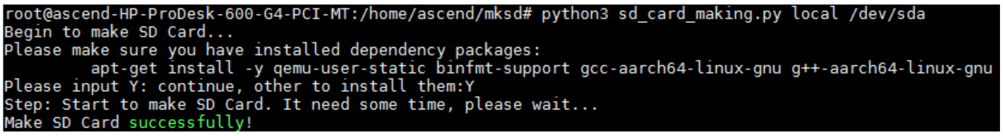
   python3 make_sd_card.py local /dev/sda

   – **local** indicates that the SD card is created locally.

   – **/dev/sda** is the name of the USB device where the SD card is located.

   If information in **Figure 3-11** is displayed, the card is successfully created.

Figure 3-11 Message indicating successful SD card creation



```
root@ascend-HP-ProDesk-600-G4-PCI-MT:/home/ascend/mksd# python3 sd_card_making.py local /dev/sda
Begin to make SD Card...
Please make sure you have installed dependency packages:
        apt-get install -y qemu-user-static binfmt-support gcc-aarch64-linux-gnu g++-aarch64-linux-gnu
Please input Y: continue, other to install them:Y
Step: Start to make SD Card. It need some time, please wait...
Make SD Card successfully!
```

NOTE

If the card creation fails, check the log files in the **sd_card_making_log** folder under the current directory.

Step 7 After the card is created, remove the SD card from the card reader and insert it into the card slot of the Atlas 200 DK developer board.

Step 8 Power on the Atlas 200 DK developer board.

NOTICE

Do not power off the Atlas 200 DK developer board during the first startup process. Otherwise, the developer board may be damaged. To re-power on the developer board, wait at least 2s after it is powered off.

For details about the method of powering on the Atlas 200 DK developer board and the description of the LED indicator status after power-on, see **7.1 Powering On the Atlas 200 DK Developer Board**.

----End

3.5.2 Creating an SD Card When a Card Reader Is Unavailable

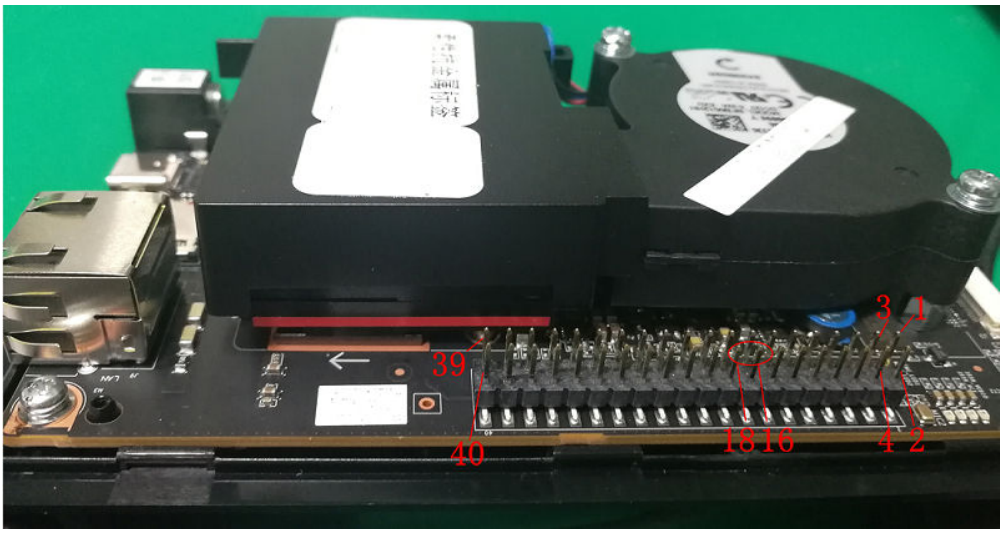
Hardware Preparations

1. Remove the cover by referring to **3.3 Removing the Cover**.
2. Place the jumper cap over pin 16 and pin 18 on the developer board, as shown in **Figure 3-12**.

NOTICE

- Perform this operation when the developer board is powered off. For details about the power-off requirements, see **7.2 Powering Off the Atlas 200 DK Developer Board**.
- Check the positions of pins carefully. If pins are misplaced, the Atlas 200 DK developer board will be severely damaged.
- The positions of pins 1, 2, and 40 are marked in white on the panel.

Figure 3-12 Pin arrangements of the developer board



3. Connect the Atlas 200 DK developer board to the USB port of the Ubuntu server.
4. Power on the Atlas 200 DK developer board. For details about the power-on operations, see **7.1 Powering On the Atlas 200 DK Developer Board**.

Software Preparations

Obtain the **make_sd_card.py** and **make_ubuntu_sd.sh** SD card creation scripts, Atlas 200 DK running package, and Ubuntu package.

Table 3-2 shows the download packages.

Table 3-2 Software packages

Package	Name	Description
Entry script for creating an SD card	make_sd_card.py	Obtain the package from **https://gitee.com/HuaweiAscend/tools**.
Script for creating the SD card OS	make_ubuntu_sd.sh	Obtain the package from **https://gitee.com/HuaweiAscend/tools**.
Atlas 200 DK running package	mini_developerkit-xxx.rar The software integrity verification file is **mini_developerkit-xxx.rar.asc**.	Software package of the developer board. Verify the integrity of the downloaded software package by referring to **7.9 Verifying Software Integrity** .
Ubuntu package	ubuntu-*xxx*-server-arm64.iso **NOTE** *xxx* indicates the version number.	The Ubuntu Linux version must be 16.04.3. Download the software of the required version from **http://old-releases.ubuntu.com/releases/16.04.3/** and install it. **NOTE** ● The Ubuntu package must be of the **server-arm64** version. ● The Ubuntu package version must be the same as the Ubuntu version of the device where Mind Studio is installed.

NOTE

Do not change the names of the downloaded files.

Procedure

Step 1 Run the following commands on the Ubuntu server to install the qemu-user-static, binfmt-support, YAML, and cross compiler:

su - root

Run the following command to update the source:

apt-get update

Run the following command to install the dependent libraries:

apt-get install qemu-user-static binfmt-support python3-yaml gcc-aarch64-linux-gnu g++-aarch64-linux-gnu

The versions of **gcc-aarch64-linux-gnu** and **g++-aarch64-linux-gnu** must be 5.4.0. You can use any versions of other dependent software packages. The default GCC version installed on Ubuntu 16.04.3 is 5.4.0.

Step 2 Upload **make_sd_card.py**, **make_ubuntu_sd.sh**, the Atlas 200 DK running package, and the Ubuntu package obtained from **Software Preparations** to any directory on the Ubuntu server as a common user, for example, */home/ascend/mksd*.

NOTE

The preceding scripts and software packages must be stored in the same directory.

The local creation of an SD card only allows software packages of the same version to be saved to the directory.

Step 3 Switch to the **root** user and go to the */home/ascend/mksd* directory where the SD card creation scripts are stored.

su - root

cd /home/ascend/mksd/

Step 4 (Optional) In the SD card making script, the default IP address of the USB NIC of the Atlas 200 DK developer board is **192.168.1.2**, and the default IP address of the NICDMA network card is **192.168.0.2**. If you want to change the default IP addresses, change them as follows:

Change the values of **NETWORK_CARD_DEFAULT_IP** and **USB_CARD_DEFAULT_IP** in the **make_sd_card.py** file.

- **NETWORK_CARD_DEFAULT_IP**: IP address of the NIC of the Atlas 200 DK developer board
- **USB_CARD_DEFAULT_IP**: IP address of USB NIC of the Atlas 200 DK developer board

Step 5 Run the SD card creation script.

1. Run the following command to query the name of the USB device where the SD card is located:

   fdisk -l

   The following uses the device name **/dev/sda** as an example:

2. Run the **make_sd_card.py** script.

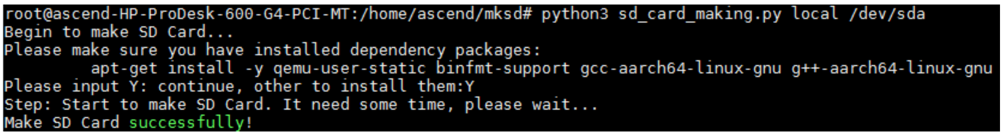
   python3 make_sd_card.py local /dev/sda

   – **local** indicates that the SD card is created locally.

   – **/dev/sda** is the name of the USB device where the SD card is located.

   If information in **Figure 3-13** is displayed, the card is successfully created.

Figure 3-13 Message indicating successful SD card creation



```
root@ascend-HP-ProDesk-600-G4-PCI-MT:/home/ascend/mksd# python3 sd_card_making.py local /dev/sda
Begin to make SD Card...
Please make sure you have installed dependency packages:
        apt-get install -y qemu-user-static binfmt-support gcc-aarch64-linux-gnu g++-aarch64-linux-gnu
Please input Y: continue, other to install them:Y
Step: Start to make SD Card. It need some time, please wait...
Make SD Card successfully!
```

NOTE

If the card creation fails, check the log files in the **sd_card_making_log** folder under the current directory.

Step 6 Remove the jumper cap.

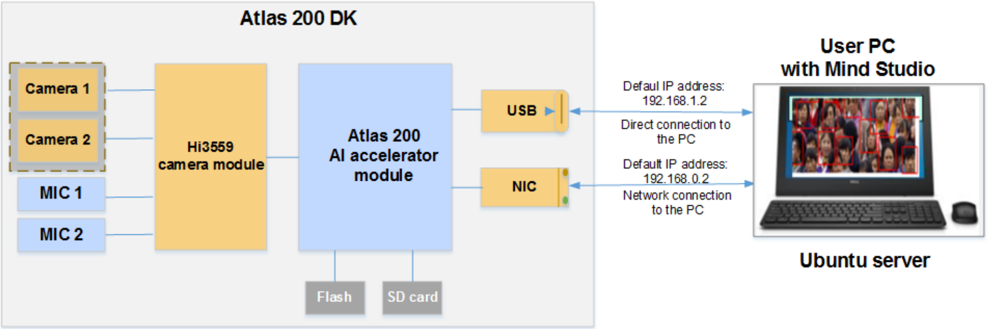
----End

3.6 Connecting the Atlas 200 DK Developer Board to the Ubuntu Server

Principle Description

You can connect the Atlas 200 DK developer board to Mind Studio using a USB cable or network cable, as shown in **Figure 3-14**.

Figure 3-14 Connection between the Atlas 200 DK developer board and Mind Studio



To implement communication between the Ubuntu server and the Atlas 200 DK, their IP addresses must be on the same network segment. For example, if the Atlas 200 DK developer board is connected to the Ubuntu server using a USB cable, set the IP address of the USB virtual NIC on the Ubuntu server to **192.168.1.***x* (such as **192.168.1.223**).

NOTE

If you have changed the IP address of the Atlas 200 DK developer board and that of the USB virtual NIC on the Ubuntu server to the same network segment during SD card creation, skip the following steps of changing the IP address of the USB virtual NIC on the Ubuntu server.

Procedure

1. Connect the Ubuntu server to the Atlas 200 DK developer board.

   There are two connection modes:

   – To connect to the Atlas 200 DK developer board over the USB port, go to **2**.

   – To connect to the Atlas 200 DK developer board through a network cable over a router or switch, go to **3**.

2. In USB connection mode, configure the IP address of the Ubuntu server.

   Change the IP address of the USB virtual NIC on the Ubuntu server to **192.168.1.*x*** (The value ranges of *x* is 0–1 and 3–255).

   NOTE

   In USB connection mode, the default address of the Atlas 200 DK developer board is **192.168.1.2**. USB 2.0 and USB 3.0 are both supported.

   When the Atlas 200 DK developer board is connected over a USB port, you need to configure a static USB IP address for the USB virtual NIC of the Ubuntu server either using a script or manually.

   – Configure the IP address by using a script.

     i. Download **configure_usb_ethernet.sh** from **https://gitee.com/HuaweiAscend/tools** to a directory on the Ubuntu server where Mind Studio is located, for example, **/home/ascend/config_usb_ip/**.

        NOTE

        You can configure the IP address of the USB virtual NIC for the first time using the script. If you want to change the IP address after it has been configured, manually change it by referring to **Configure the IP address manually**.

     ii. Go to the directory where the script for configuring the IP address of the USB virtual NIC is located as the **root** user, for example, **/home/ascend/config_usb_ip**.

     iii. Run the following command to configure the IP address of the USB virtual NIC :

        bash configure_usb_ethernet.sh -s *ip_address*

        ○ Change *ip_address* to the actual static IP address of the USB virtual NIC of the Ubuntu server. If **bash configure_usb_ethernet.sh** is run directly, the default IP address **192.168.1.166** is used.

        ○ If there are multiple USB NICs, run the **ifconfig** command to query the names of the USB NICs. Remove and insert the developer board to determine the USB virtual NIC of the developer board. The Ubuntu server identifies the Atlas 200 DK developer board as a USB virtual NIC. Then run the following command to configure the IP address of the specified virtual NIC:

          bash configure_usb_ethernet.sh -s *usb_nic_name ip_address*

          usb_nic_name: name of the USB virtual NIC

          ip_address: IP address to be configured

          For example, to set the IP address of the USB virtual NIC of the Ubuntu server to **192.168.1.223**, run the following command:

          bash configure_usb_ethernet.sh -s enp0s20f0u8 192.168.1.223

        After the configuration is complete, run the **ifconfig** command to check whether the IP address takes effect.

   – Configure the IP address manually.

     i. Log in to the server where Mind Studio is located as a common user and run the following command to switch to the **root** user:

```
su - root
```

ii. Obtain the name of the USB virtual NIC.

```
ifconfig -a
```

If the system has multiple USB NICs, you can reseat the developer board to identify the required virtual NIC.

iii. Add the static IP address of the USB virtual NIC to the **/etc/network/interfaces** file.

Run the following command to open the **interfaces** file:

```
vi /etc/network/interfaces
```

Configure the **interfaces** file as follows. Assume that the USB virtual NIC name is **enp0s20f0u4** and its static IP address is **192.168.1.223**.

```
auto enp0s20f0u4
iface enp0s20f0u4 inet static
address 192.168.1.223
netmask 255.255.255.0
```

iv. Modify the **NetworkManager.conf** file to prevent the network configuration from becoming invalid after restart.

If the Ubuntu server version is used, skip this step.

Run the following command to open the **NetworkManager.conf** file:

vi /etc/NetworkManager/NetworkManager.conf

Change **managed=false** to **managed=true** in the file.

v. Enable the static IP address.

Run the following commands:

```
ifdown enp0s20f0u4
ifup enp0s20f0u4
service NetworkManager restart     //If the Ubuntu server version is used, skip this step.
```

3. In network cable connection mode, configure the IP address of the Ubuntu server as follows:

If the Ubuntu server is connected to the Atlas 200 DK developer board through a router or switch, change the IP address of the Ubuntu server to **192.168.0.*x*** (the value of *x* can be 0 to 1 and 3 to 255).

NOTE

- The default IP address of the Atlas 200 DK developer board connected using network cables is **192.168.0.2**, with a 24-bit subnet mask.
- After the network port on the Atlas 200 DK is connected to a network cable, if the yellow ACT indicator blinks, it indicates that data is being transmitted. When the network port of the Atlas 200 DK accesses the GE network, the green link indicator is on. When the network port of the Atlas 200 DK accesses the 100M/10M Ethernet network, the link indicator is off, which is normal.

Perform the following steps:

– Log in to the Ubuntu server as a common user and run the following command to switch to the **root** user:

```
su - root
```

– Add the virtual static IP address to the **/etc/network/interfaces** file.

Run the following command to open the **interfaces** file:

```
vi /etc/network/interfaces
```

Configure the **interfaces** file, for example, by adding the virtual NIC **eth0:1** whose static IP address is **192.168.0.223**.

```
auto eth0:1
iface eth0:1 inet static
```

```
address 192.168.0.223
netmask 255.255.255.0
```

- Modify the **NetworkManager.conf** file to prevent the network configuration from becoming invalid after restart.

  If the Ubuntu server version is used, skip this step.

  Run the following command to open the **NetworkManager.conf** file:

  vi /etc/NetworkManager/NetworkManager.conf

  Change **managed=false** to **managed=true** in the file.

- Restart the network services.

```
service networking restart
service NetworkManager restart       //If the Ubuntu server version is used, skip this step.
```

Follow-up Operations

After the Atlas 200 DK developer board is connected to the Ubuntu server, you can determine whether to restart the Linux server on the Atlas 200 DK developer board or power off the Atlas 200 DK developer board based on the status of the Atlas 200 DK LED indicators. For details about the status of the LED indicators, see **Table 7-1**.

NOTICE

Exercise caution when restarting or powering off the Atlas 200 DK developer board, especially during its upgrade.

4 (Optional) Configuring the Cross Compilation Environment for the Mind Studio Installation Server

Before using the Atlas 200 DK to develop applications, you need to configure the cross compilation environment on the Linux server where Mind Studio is located.

NOTE

If the Ubuntu server used for SD card creation is the server where Mind Studio is located (connected to the Atlas 200 DK developer board), skip this section. During SD card creation, a cross compilation environment is automatically deployed for the Ubuntu server.

Prerequisites

Perform the following steps as the **root** user to check whether the cross compilation environment has been deployed:

Step 1 Run the following command to check whether the cross compiler is deployed.

aarch64-linux-gnu-gcc -v

- The command output is as follows:

```
Thread model: posix
gcc version 5.4.0 20160609 (Ubuntu/Linaro 5.4.0-6ubuntu1~16.04.9)
```

  If the preceding information is displayed, the GCC cross-compiler has been deployed in the current environment. In this situation, go to **Step 2**.

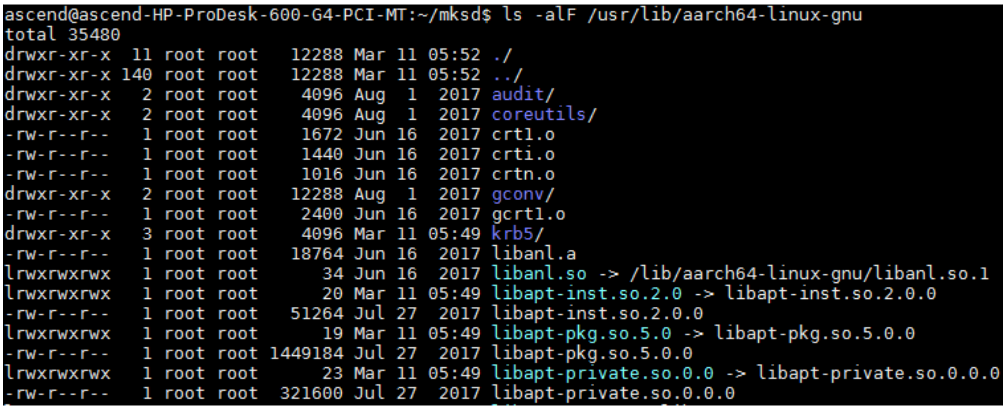
- If the command output is abnormal, the cross compiler is not deployed. In this situation, go to **Procedure**.

Step 2 Run the following command to check whether the compilation dependent libraries of the Atlas 200 DK developer board have been deployed:

ls -alF /usr/lib/aarch64-linux-gnu

- If information shown in **Figure 4-1** is displayed, the cross compilation environment has been configured for the server where Mind Studio is configured. In this situation, skip this section.

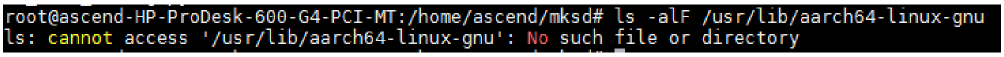
Figure 4-1 Message indicating that the cross compilation environment has been deployed



```
ascend@ascend-HP-ProDesk-600-G4-PCI-MT:~/mksd$ ls -alF /usr/lib/aarch64-linux-gnu
total 35480
drwxr-xr-x  11 root root   12288 Mar 11 05:52 ./
drwxr-xr-x 140 root root   12288 Mar 11 05:52 ../
drwxr-xr-x   2 root root    4096 Aug  1  2017 audit/
drwxr-xr-x   2 root root    4096 Aug  1  2017 coreutils/
-rw-r--r--   1 root root    1672 Jun 16  2017 crt1.o
-rw-r--r--   1 root root    1440 Jun 16  2017 crti.o
-rw-r--r--   1 root root    1016 Jun 16  2017 crtn.o
drwxr-xr-x   2 root root   12288 Aug  1  2017 gconv/
-rw-r--r--   1 root root    2400 Jun 16  2017 gcrt1.o
drwxr-xr-x   3 root root    4096 Mar 11 05:49 krb5/
-rw-r--r--   1 root root   18764 Jun 16  2017 libanl.a
lrwxrwxrwx   1 root root      34 Jun 16  2017 libanl.so -> /lib/aarch64-linux-gnu/libanl.so.1
lrwxrwxrwx   1 root root      20 Mar 11 05:49 libapt-inst.so.2.0 -> libapt-inst.so.2.0.0
-rw-r--r--   1 root root   51264 Jul 27  2017 libapt-inst.so.2.0.0
lrwxrwxrwx   1 root root      19 Mar 11 05:49 libapt-pkg.so.5.0 -> libapt-pkg.so.5.0.0
-rw-r--r--   1 root root 1449184 Jul 27  2017 libapt-pkg.so.5.0.0
lrwxrwxrwx   1 root root      23 Mar 11 05:49 libapt-private.so.0.0 -> libapt-private.so.0.0.0
-rw-r--r--   1 root root  321600 Jul 27  2017 libapt-private.so.0.0.0
```

- If the information shown in **Figure 4-2** is displayed, the cross compilation environment of the Mind Studio server is not configured. In this case, manually configure the cross compilation environment by following this section.

Figure 4-2 Message indicating that the cross compilation environment is not deployed



```
root@ascend-HP-ProDesk-600-G4-PCI-MT:/home/ascend/mksd# ls -alF /usr/lib/aarch64-linux-gnu
ls: cannot access '/usr/lib/aarch64-linux-gnu': No such file or directory
```

----End

Procedure

Step 1 Obtain the **make_ui_cross_env.py** script for configuring the cross compilation environment.

Obtain it from the **tools** repository at **https://gitee.com/HuaweiAscend/tools**.

Step 2 Upload the **make_ui_cross_env.py** script to a directory on the server where Mind Studio is located, for example, */home/ascend/mkuicross*.

Step 3 Switch to the **root** user and go to the */home/ascend/mkuicross* directory where the **make_ui_cross_env.py** script is located:

su - root

cd /home/ascend/mkuicross/

Step 4 Run the following command to install Pexpect for automatic SSH interaction:

pip3 install pexpect

Step 5 Run the script for configuring a cross compilation environment:

python3 make_ui_cross_env.py

The following information is displayed:

Atlas DK Development Board IP: IP address of the Atlas 200 DK developer board IP

Atlas DK Development Board SSH user: SSH login user name of the Atlas 200 DK developer board. Set this parameter to the common user. The default user name is **HwHiAiUser**.

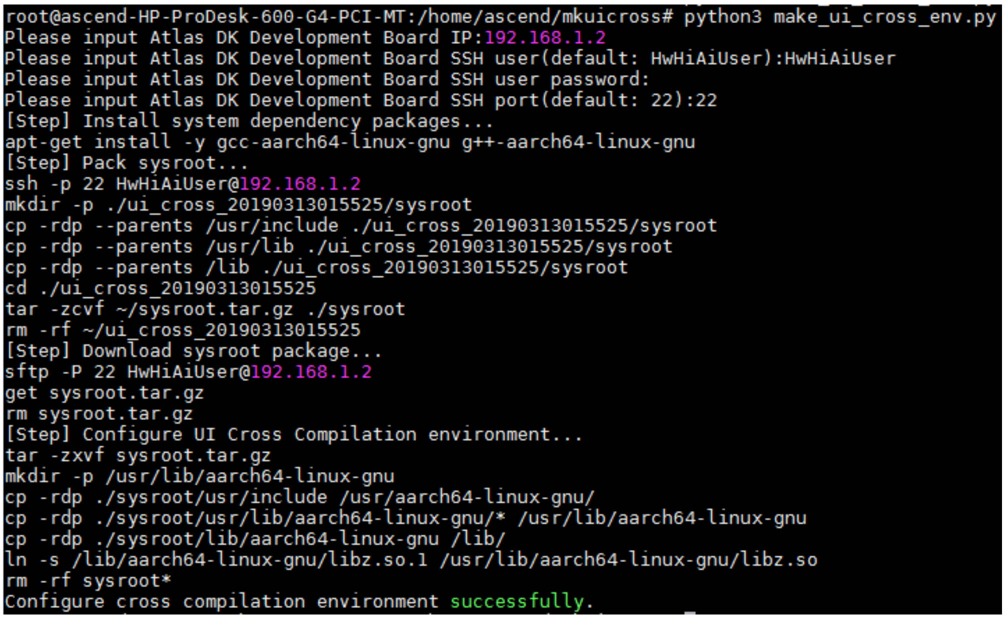
Atlas DK Development Board SSH user password: password of the SSH login user of the Atlas 200 DK developer board

Atlas DK Development Board SSH port: SSH port number of the Atlas 200 DK developer board. The default value is **22**.

- If the server where Mind Studio is located has been connected to the network:

  During the script execution, the cross-compilers gcc-aarch64-linux-gnu and g++-aarch64-linux-gnu are downloaded and installed on the Linux server where Mind Studio is located. **Figure 4-3** shows the execution result.

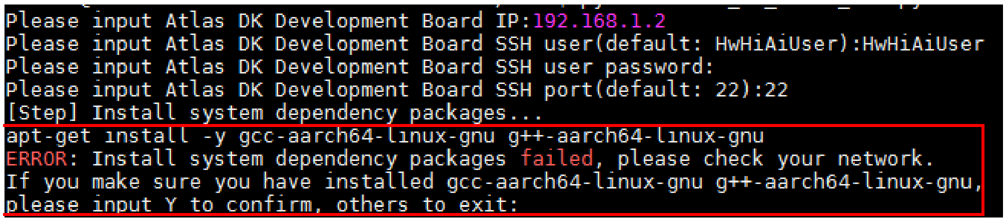
Figure 4-3 Configuring a cross compilation environment



```
root@ascend-HP-ProDesk-600-G4-PCI-MT:/home/ascend/mkuicross# python3 make_ui_cross_env.py
Please input Atlas DK Development Board IP:192.168.1.2
Please input Atlas DK Development Board SSH user(default: HwHiAiUser):HwHiAiUser
Please input Atlas DK Development Board SSH user password:
Please input Atlas DK Development Board SSH port(default: 22):22
[Step] Install system dependency packages...
apt-get install -y gcc-aarch64-linux-gnu g++-aarch64-linux-gnu
[Step] Pack sysroot...
ssh -p 22 HwHiAiUser@192.168.1.2
mkdir -p ./ui_cross_20190313015525/sysroot
cp -rdp --parents /usr/include ./ui_cross_20190313015525/sysroot
cp -rdp --parents /usr/lib ./ui_cross_20190313015525/sysroot
cp -rdp --parents /lib ./ui_cross_20190313015525/sysroot
cd ./ui_cross_20190313015525
tar -zcvf ~/sysroot.tar.gz ./sysroot
rm -rf ~/ui_cross_20190313015525
[Step] Download sysroot package...
sftp -P 22 HwHiAiUser@192.168.1.2
get sysroot.tar.gz
rm sysroot.tar.gz
[Step] Configure UI Cross Compilation environment...
tar -zxvf sysroot.tar.gz
mkdir -p /usr/lib/aarch64-linux-gnu
cp -rdp ./sysroot/usr/include /usr/aarch64-linux-gnu/
cp -rdp ./sysroot/usr/lib/aarch64-linux-gnu/* /usr/lib/aarch64-linux-gnu
cp -rdp ./sysroot/lib/aarch64-linux-gnu /lib/
ln -s /lib/aarch64-linux-gnu/libz.so.1 /usr/lib/aarch64-linux-gnu/libz.so
rm -rf sysroot*
Configure cross compilation environment successfully.
```

- If the Mind Studio server is not connected to the network, the message shown in **Figure 4-4** is displayed.

Figure 4-4 Message indicating a cross compiler download failure



```
Please input Atlas DK Development Board IP:192.168.1.2
Please input Atlas DK Development Board SSH user(default: HwHiAiUser):HwHiAiUser
Please input Atlas DK Development Board SSH user password:
Please input Atlas DK Development Board SSH port(default: 22):22
[Step] Install system dependency packages...
apt-get install -y gcc-aarch64-linux-gnu g++-aarch64-linux-gnu
ERROR: Install system dependency packages failed, please check your network.
If you make sure you have installed gcc-aarch64-linux-gnu g++-aarch64-linux-gnu,
please input Y to confirm, others to exit:
```

When a message is displayed indicating a failure to download the cross compiler:

  - If the cross compilers gcc-aarch64-linux-gnu and g++-aarch64-linux-gnu have been deployed in the system, input **Y** to continue the deployment of the cross compilation environment.

– If the cross compilers gcc-aarch64-linux-gnu and g++-aarch64-linux-gnu are not deployed in the system, input any other keys to exit the script execution, manually download and install the cross compilers, and then run the **python3 make_ui_cross_env.py** command.

  NOTE

  Download the 5.4.0 version of gcc-aarch64-linux-gnu and g++-aarch64-linux-gnu.

----End

5 Configuring and Managing the Atlas 200 DK

This section describes the basic configuration and management operations of the Atlas 200 DK, including changing the password and IP address.

5.1 Changing the Password

Before using the Atlas 200 DK developer board, you need to change the initial user password.

Changing the Password of a Common User

HwHiAiUser is the default user created on Mind Studio during SD card creation. The default password is **Mind@123**. After the Atlas 200 DK developer board is successfully connected to Mind Studio, you need to change the initial password of the **HwHiAiUser** user.

Step 1 On the Mind Studio server, log in to the Atlas 200 DK developer board as the **HwHiAiUser** user in SSH mode.

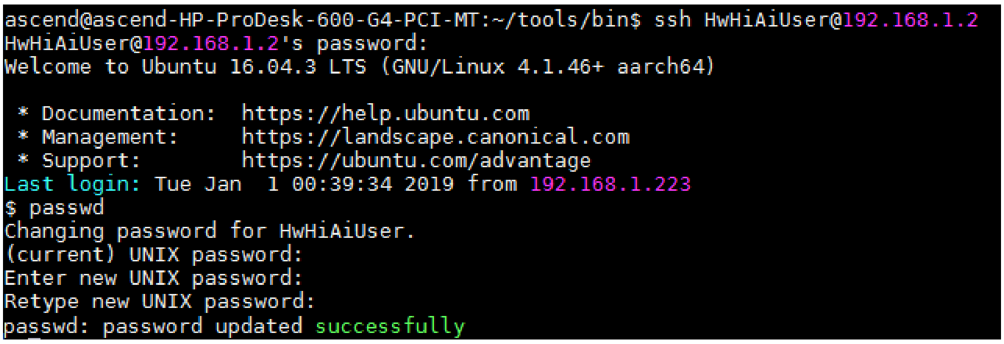
NOTE

- The default login password of the **HwHiAiUser** user is **Mind@123**.
- If the trust relationship fails to be established when you log in to the Atlas 200 DK developer board in SSH mode, see **8.4 What Do I Do If the Trust Relationship with the Developer Board Fails to Be Established by Using UI Host?**.

Step 2 Run the **passwd** command to change the password of the **HwHiAiUser** user.

See **Figure 5-1**.

Figure 5-1 Changing the password of the HwHiAiUser user



```
ascend@ascend-HP-ProDesk-600-G4-PCI-MT:~/tools/bin$ ssh HwHiAiUser@192.168.1.2
HwHiAiUser@192.168.1.2's password:
Welcome to Ubuntu 16.04.3 LTS (GNU/Linux 4.1.46+ aarch64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage
Last login: Tue Jan  1 00:39:34 2019 from 192.168.1.223
$ passwd
Changing password for HwHiAiUser.
(current) UNIX password:
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
```

----End

Modifying the Password of the Root User

Step 1 On the Mind Studio server, log in to the Atlas 200 DK developer board as the **HwHiAiUser** user in SSH mode.

NOTE

The default login password of the **HwHiAiUser** user is **Mind@123**.

Step 2 Run the following command to switch to the **root** user:

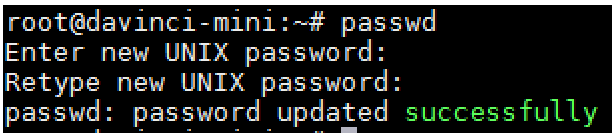
su - root

NOTE

The default login password of the **root** user is **Mind@123**.

Step 3 Run the **passwd** command to change the password of the **root** user, as shown in Figure 5-2.

Figure 5-2 Changing the password of the root user



```
root@davinci-mini:~# passwd
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
```

----End

5.2 Adding the Atlas 200 DK Developer Board to Mind Studio

You can use the Atlas 200 DK developer board configuration management function provided by Mind Studio to detect the connection status of the Atlas 200 DK developer board and modify the device information.

NOTE

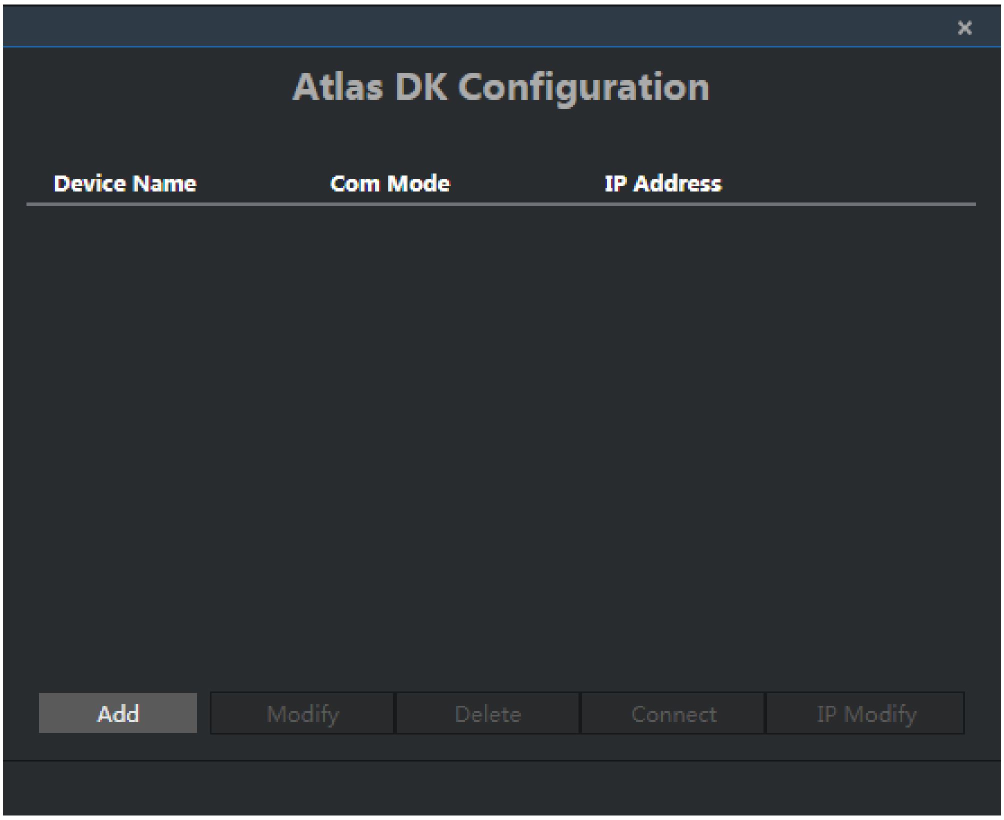
The current configuration management function supports only single-user and single-task scenarios. Multi-user and multi-task scenarios are not supported

Adding a Device

Step 1 On the menu bar of Mind Studio, choose **Tools > Atlas DK Configuration**.

The **Atlas DK Configuration** dialog box is displayed, as shown in **Figure 5-3**.

Figure 5-3 Atlas DK Configuration



Atlas DK Configuration

Device Name	Com Mode	IP Address

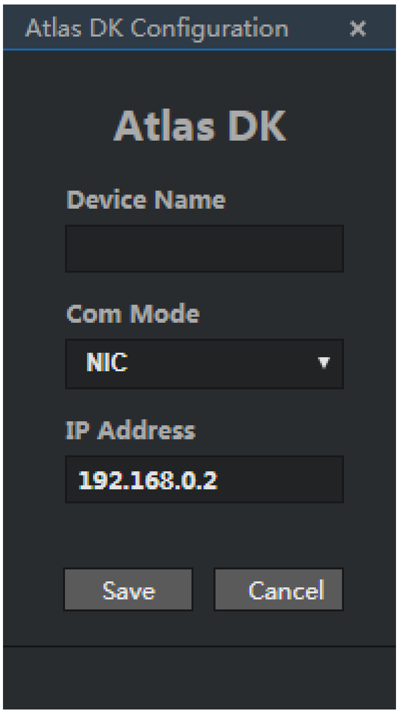
Step 2 Click **Add**.

The device information dialog box is displayed.

Step 3 Configure the basic information about the Atlas 200 DK to be added.

See **Figure 5-4**.

Figure 5-4 Atlas DK Configuration > Add



Atlas DK Configuration ×

Atlas DK

Device Name:

Com Mode: NIC

IP Address: 192.168.0.2

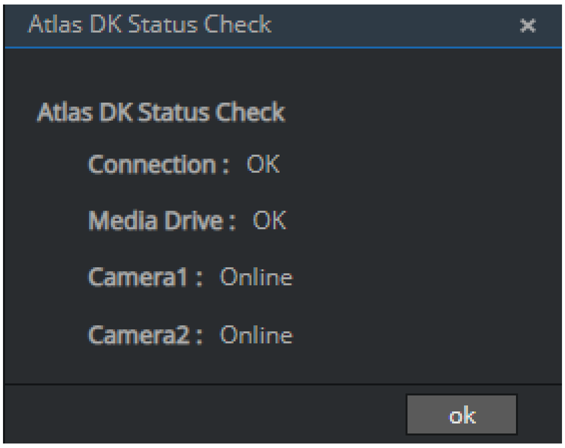
The configuration parameters are described as follows:

- **Device Name**: user-defined device name
- **Com Mode**: communication mode, which can be **NIC** or **USB**
  - **NIC** indicates using general network communication.
  - **USB** indicates using a USB virtual NIC.
- **IP Address**: IP address of the Atlas 200 DK device

Step 4 Click **Save**.

After the configuration is complete, the **Atlas 200 DK Status Check** dialog box as shown in **Figure 5-5** is displayed, showing the status of each module.

Figure 5-5 Atlas DK Status Check dialog box



Atlas DK Status Check ×

Atlas DK Status Check

Connection : OK

Media Drive : OK

Camera1 : Online

Camera2 : Online

The parameters are described as follows:

- **Connection**: communication status between Mind Studio and the Atlas 200 DK
- **Media Drive**: status of the audio and video processing module (Hi3559)
- **Camera1**: whether channel 1 is connected to a camera
- **Camera2**: whether channel 2 is connected to a camera

NOTE

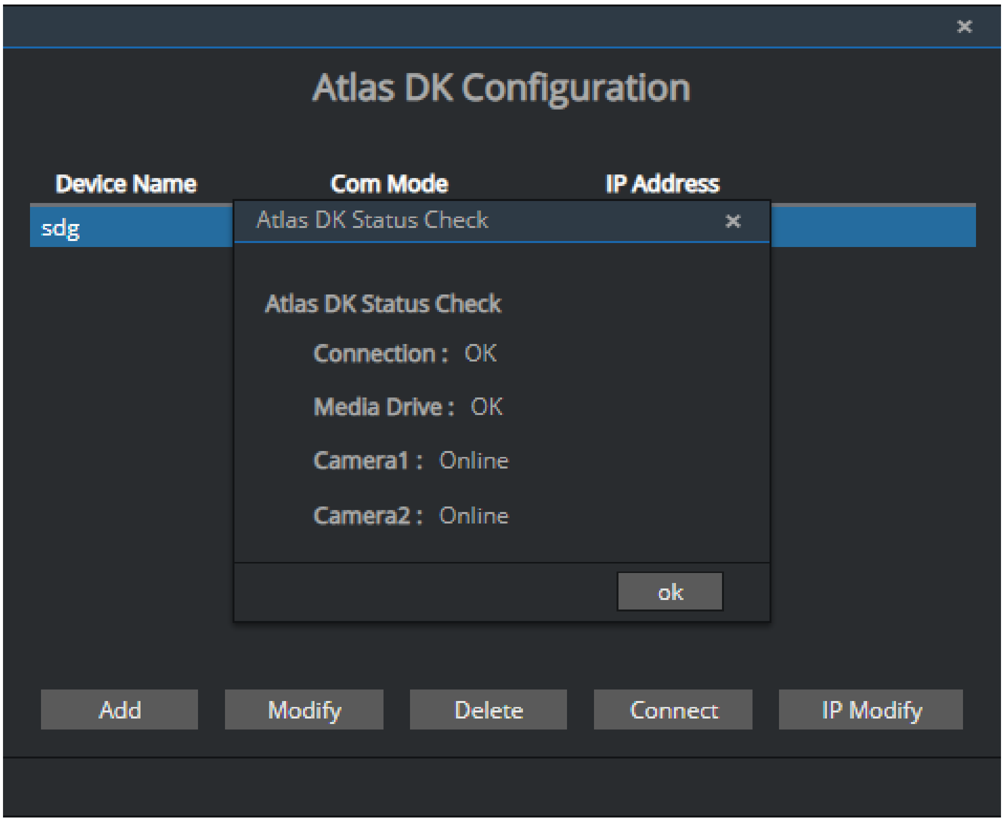
- During device addition, the network connection between the PC and the Atlas 200 DK device is checked. If the network connection is normal, the configuration information of the Atlas 200 DK device is added to the **Atlas DK Configuration** device list. Otherwise, a message is displayed indicating that the connection fails and the device configuration information is not added to the **Atlas DK Configuration** device list.
- If a USB virtual NIC is used for communication with the Atlas 200 DK device, the IP address of Mind Studio must be the default setting **192.168.1.2**, and the IP address of the USB virtual NIC on the PC must be on the same network segment as **192.168.1.2**. The default subnet mask is **255.255.255.0**. You can also use terminal software (such as Xshell) to access the system terminal of the Atlas 200 DK device over a NIC or USB virtual NIC.

----End

Connecting a Device

Select the device to be operated from the **Atlas DK Configuration** device list, and click **Connect**, as shown in **Figure 5-6**.

Figure 5-6 Atlas DK Configuration > Connect



Atlas DK Configuration

Device Name	Com Mode	IP Address
sdg		

Atlas DK Status Check

Atlas DK Status Check
Connection : OK
Media Drive : OK
Camera1 : Online
Camera2 : Online

For details about the description of the parameters, see **Adding a Device**.

5.3 Changing the IP Address of the Atlas 200 DK Developer Board

If multiple Atlas 200 DK developer boards are configured, using the default IP address my cause conflicts. You can change their IP addresses on Mind Studio.

NOTE

This section describes how to modify the IP address of an Atlas 200 DK developer board on Mind Studio in the network cable connection mode.

For the USB connection mode, you need to modify the background **/etc/network/interfaces** file, and then run the **service networking restart** command to restart the network service.

Procedure

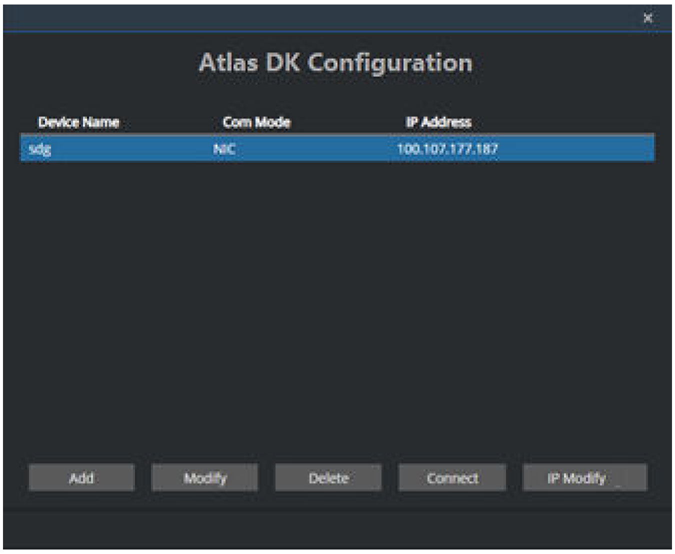
Step 1 On the menu bar of Mind Studio, choose **Tools > Atlas DK Configuration**.

The **Atlas DK Configuration** dialog box is displayed.

Step 2 In the displayed **Atlas DK Configuration** dialog box, select a device and click **IP Modify**.

See **Figure 5-7**.

Figure 5-7 Atlas DK Configuration device list



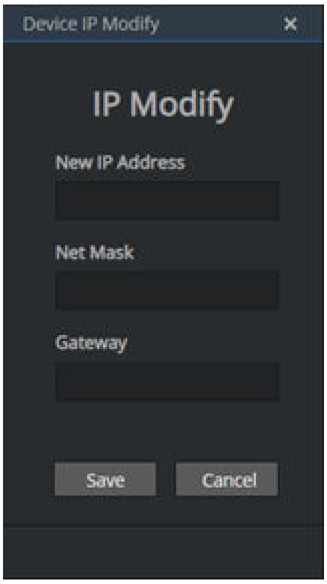
Atlas DK Configuration

Device Name	Com Mode	IP Address
sdg	NIC	100.107.177.187

Step 3 In the displayed **Device IP Modify**, configure the IP address of the Atlas 200 DK developer board.

See **Figure 5-8**.

Figure 5-8 Changing the IP address of the Atlas 200 DK developer board



Device IP Modify ×

IP Modify

New IP Address

Net Mask

Gateway

Enter the new IP address, subnet mask, and gateway of the device, and click **Save**.

NOTE

- Currently, this method supports only the NIC communication mode and does not support the USB NIC.

----End

6 Upgrading the Atlas 200 DK

Upgrade Using Mind Studio

You can upgrade the system of the Atlas 200 DK developer board online by using Mind Studio. For details, see "Version Upgrade" in Ascend 310 Mind Studio Installation Guide (Ubuntu, x86).

NOTE

Do not power off the Atlas 200 DK developer board during the upgrade. The upgrade takes about 15 minutes.

Manual Upgrade

Step 1 Upload the upgrade package **mini_developerkit-xxx.rar** of the Atlas 200 DK developer board to any directory on the Mind Studio server as the Mind Studio installation user.

Step 2 Go to the directory where the developer board upgrade package is stored as the Mind Studio installation user, log in to the developer board as the **HwHiAiUser** user in SSH mode, and switch to the **root** user.

ssh HwHiAiUser@*192.168.1.2*

su - root

NOTE

If the trust relationship fails to be established when you log in to the Atlas 200 DK developer board in SSH mode, see **8.4 What Do I Do If the Trust Relationship with the Developer Board Fails to Be Established by Using UI Host?**.

Step 3 Copy the upgrade package **mini_developerkit-xxx.rar** to the **/opt/mini** directory of the developer board.

cd /opt/mini

sftp *username@192.168.1.223*

NOTE

Replace *username* with the name of the Mind Studio installation user.

Replace *192.168.1.223* with the IP address of the Mind Studio server that is on the same network segment as the developer board.

sftp>cd */home/ubuntu/software*

NOTE

Replace */home/ubuntu/software* with the actual path for storing the developer board upgrade package on the Mind Studio server.

sftp> get mini_developerkit-xxx.rar

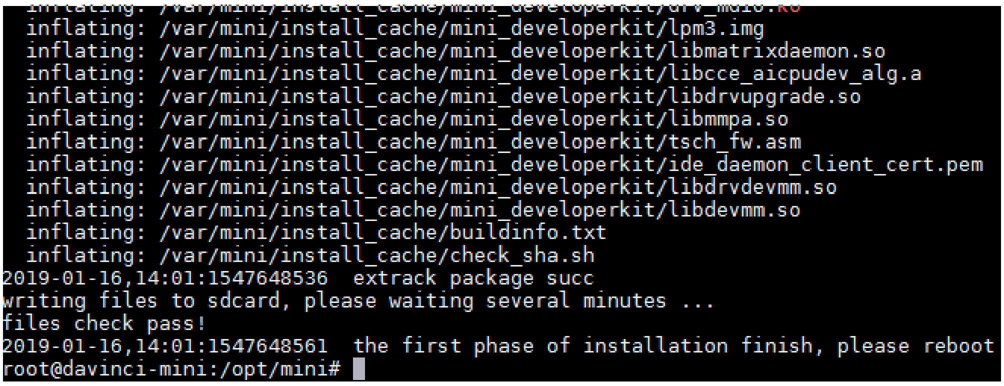
sftp> exit

Step 4 Run the following command to prepare for the upgrade.

./minirc_install_phase1.sh

Information as shown in **Figure 6-1** is displayed.

Figure 6-1 Running the upgrade script



```
  inflating: /var/mini/install_cache/mini_developerkit/lpm3.img
  inflating: /var/mini/install_cache/mini_developerkit/libmatrixdaemon.so
  inflating: /var/mini/install_cache/mini_developerkit/libcce_aicpudev_alg.a
  inflating: /var/mini/install_cache/mini_developerkit/libdrvupgrade.so
  inflating: /var/mini/install_cache/mini_developerkit/libmmpa.so
  inflating: /var/mini/install_cache/mini_developerkit/tsch_fw.asm
  inflating: /var/mini/install_cache/mini_developerkit/ide_daemon_client_cert.pem
  inflating: /var/mini/install_cache/mini_developerkit/libdrvdevmm.so
  inflating: /var/mini/install_cache/mini_developerkit/libdevmm.so
  inflating: /var/mini/install_cache/buildinfo.txt
  inflating: /var/mini/install_cache/check_sha.sh
2019-01-16,14:01:1547648536  extrack package succ
writing files to sdcard, please waiting several minutes ...
files check pass!
2019-01-16,14:01:1547648561  the first phase of installation finish, please reboot
root@davinci-mini:/opt/mini#
```

Step 5 Restart the developer board. The upgrade is complete.

reboot

NOTICE

Do not power off the Atlas 200 DK developer board during the upgrade. The upgrade takes about 15 minutes.

----End

Upgrade Verification

Step 1 On the Mind Studio server, log in to the Atlas 200 DK developer board as the **HwHiAiUser** user in SSH mode.

ssh HwHiAiUser@*192.168.1.2*

NOTE

Replace *192.168.1.2* with the actual IP address of the developer board.

Step 2 Run the following command to check the Ascend 310 version number of the developer board after the upgrade.

cat /etc/sys_version.conf

Step 3 Run the following command to query the firmware version and the versions of valid components:

1. Run the following command to switch to the **root** user:

   su - root

2. Go to the **firmware** directory.

   cd /var/davinci/firmware/

3. Run the following command to query the system version of the firmware:

   ./upgrade-tool --device_index -1 --system_version

4. Run the following command to query the component version of the firmware:

   ./upgrade-tool --device_index -1 --component -1 --version

Step 4 Run the following commands to query the upgrade logs:

cd /var/davinci/log

cat upgrade.log

cat firmware_upgrade_progress.log

Check whether the upgrade logs contain error information. If no error information is displayed, the upgrade is successful.

If error information is displayed, input the IP address of the desired developer board on the Mind Studio GUI to view logs and locate the fault. For details about how to use the log tool, see the *Ascend 310 Mind Studio Auxiliary Tools*.

----End

7 Common Operations

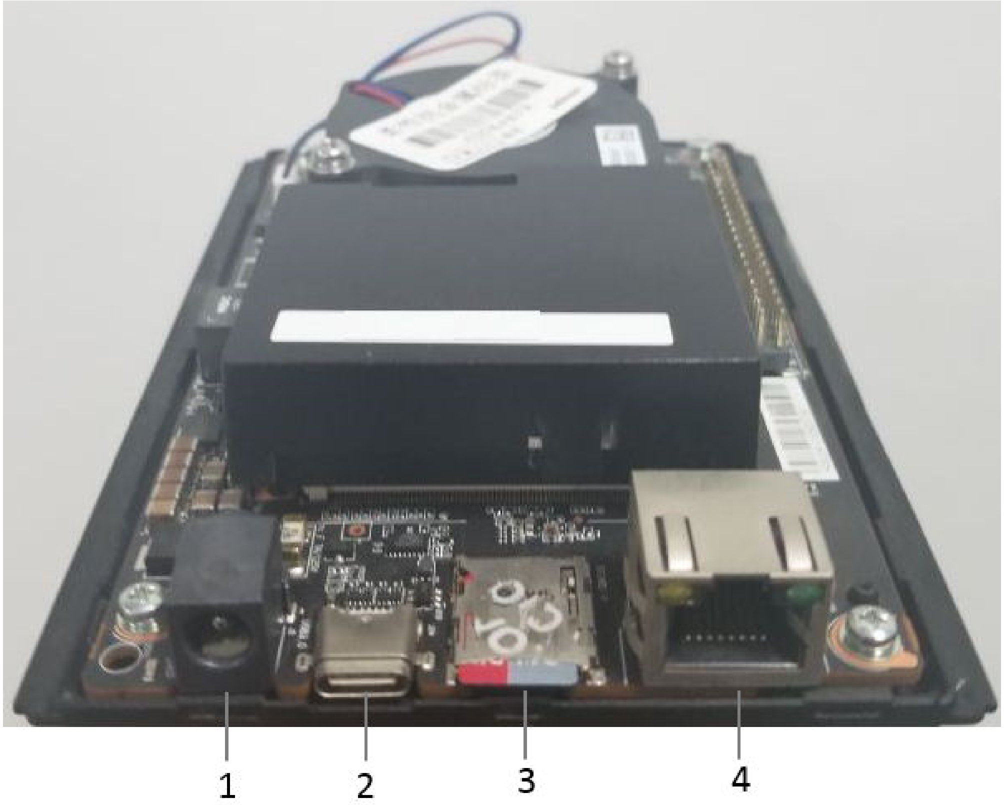
7.1 Powering On the Atlas 200 DK Developer Board

NOTICE

Do not power off the Atlas 200 DK developer board during the first startup process or upgrade. Otherwise, the developer board may be damaged. Before powering on the developer board again, wait at least 2s after it is powered off.

Step 1 Connect the power module to the external power supply. **Figure 7-1** shows the power port on the Atlas 200 DK developer board. After connected to the power supply, the Atlas 200 DK developer board automatically boots.

Figure 7-1 Port description



1	Power port	2	USB
3	SD card	4	Network port

Step 2 Check the status of the indicator to ensure that the Atlas 200 DK developer board is powered on properly.

Table 7-1 and **Table 7-2** describe the status of the indicators after power-on.

Table 7-1 Status description of LED1 and LED2

LED1	LED2	Status of the Atlas 200 DK Developer Board	Precautions
Off	Off	The Atlas 200 DK developer board is powered on.	You can power off or restart the Atlas 200 DK developer board.
Off	On	The Ascend 310 is being powered on.	You can power off or restart the Atlas 200 DK developer board except during version upgrade.

LED1	LED2	Status of the Atlas 200 DK Developer Board	Precautions
Blinking	Blinking	The firmware is being upgraded.	● Do not power off or restart the Atlas 200 DK developer board. Otherwise, the firmware upgrade may be incomplete and the board may be damaged. ● The firmware upgrade process is a part of the version upgrade. The upgrade takes about 15 minutes.
On	On	The power-on process of the Atlas 200 DK developer board is complete.	You can power off or restart the Atlas 200 DK developer board.

Table 7-2 Status description of LED3 and LED4

LED3	LED4	Status of the Atlas 200 DK Developer Board	Precautions
Off	Off	The Hi3559C system is not started.	None
Off	On	The Hi3559C system is being started.	None
On	On	The startup process of the Hi3559C system is complete.	None

----End

7.2 Powering Off the Atlas 200 DK Developer Board

Precautions

Determine whether the Atlas 200 DK developer board can be powered off based on the description in **Step 2**.

Procedure

Step 1 Disconnect the power cable from the power port to power off the Atlas 200 DK developer board.

NOTICE

The Atlas 200 DK cannot be shut down by using the OS-level shutdown command.

----End

7.3 Connecting to the Atlas 200 DK Developer Board over a Serial Port

Connecting to the Atlas 200 AI Accelerator Module over a Serial Port

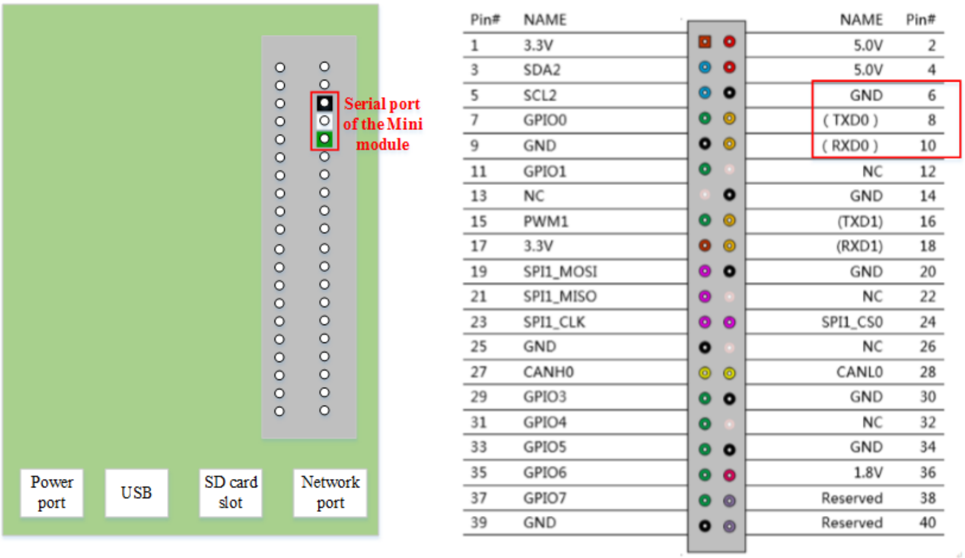
You can view the startup information about the Atlas 200 AI accelerator module on the Atlas 200 DK developer board over a serial port.

NOTE

This serial port is used only for viewing startup information. After the Atlas 200 AI accelerator module is started, its serial port is disabled, and the Atlas 200 AI accelerator module cannot be logged in to.

Figure 7-2 shows how to connect to the Atlas 200 AI accelerator module by using a serial cable.

Figure 7-2 Serial port connection of the Atlas 200 AI accelerator module



Pin#	NAME	NAME	Pin#
1	3.3V	5.0V	2
3	SDA2	5.0V	4
5	SCL2	GND	6
7	GPIO0	(TXD0)	8
9	GND	(RXD0)	10
11	GPIO1	NC	12
13	NC	GND	14
15	PWM1	(TXD1)	16
17	3.3V	(RXD1)	18
19	SPI1_MOSI	GND	20
21	SPI1_MISO	NC	22
23	SPI1_CLK	SPI1_CS0	24
25	GND	NC	26
27	CANH0	CANL0	28
29	GPIO3	GND	30
31	GPIO4	NC	32
33	GPIO5	GND	34
35	GPIO6	1.8V	36
37	GPIO7	Reserved	38
39	GND	Reserved	40

Serial port on the Atlas 200 AI accelerator module: Connect a cable to the serial port according to the colors specified in **Figure 7-2**.

Requirement for the serial cable: USB-to-serial cable (3.3 V)

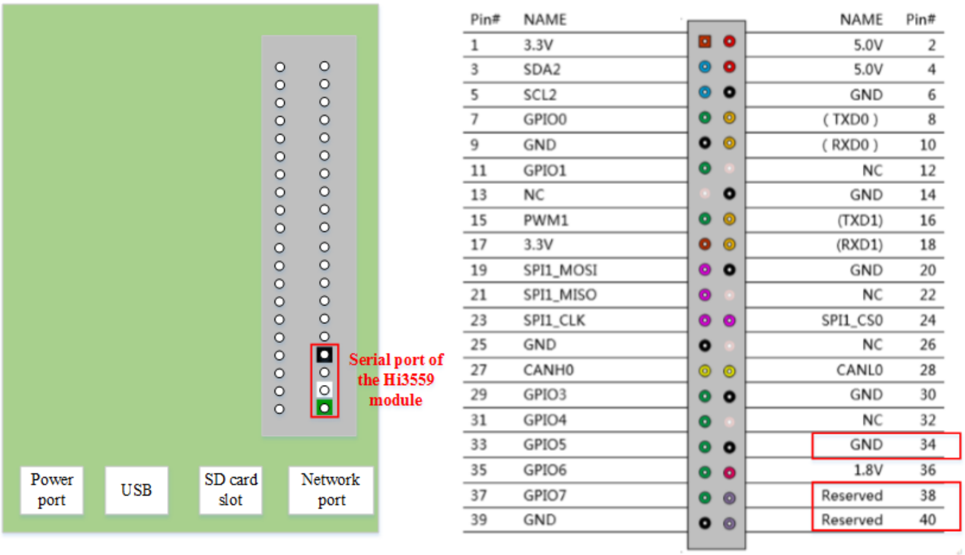
Connecting to the Hi3559 Module over a Serial Port

The Atlas 200 DK developer board provides a serial port for connecting to the Hi3559 module. **Figure 7-3** shows the serial port connection diagram.

NOTE

This serial port is used only for viewing startup information. After the Hi3559 module is started, its serial port is disabled, and the Hi3559 module cannot be logged in to.

Figure 7-3 Hi3559 serial port connection



Pin#	NAME	NAME	Pin#
1	3.3V	5.0V	2
3	SDA2	5.0V	4
5	SCL2	GND	6
7	GPIO0	(TXD0)	8
9	GND	(RXD0)	10
11	GPIO1	NC	12
13	NC	GND	14
15	PWM1	(TXD1)	16
17	3.3V	(RXD1)	18
19	SPI1_MOSI	GND	20
21	SPI1_MISO	NC	22
23	SPI1_CLK	SPI1_CS0	24
25	GND	NC	26
27	CANH0	CANL0	28
29	GPIO3	GND	30
31	GPIO4	NC	32
33	GPIO5	GND	34
35	GPIO6	1.8V	36
37	GPIO7	Reserved	38
39	GND	Reserved	40

Connect the serial cable to the Hi3559 serial port according to the colors specified in **Figure 7-3**.

Requirement for the serial cable: USB-to-serial cable (3.3 V)

7.4 Checking the Version of the Motherboard of the Developer Board

You can obtain the version number of the developer board by checking the PCB version number of the developer board.

Step 1 Create a code file for querying the version number of the developer board.

Create the **i2c_tool_atlas200dk.c** file in any directory on the Mind Studio server as the Mind Studio installation user.

touch i2c_tool_atlas200dk.c

Copy the following code to the **i2c_tool_atlas200dk.c** file:

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/ioctl.h>
#include <sys/types.h>
```

```
#include <sys/stat.h>
#include <fcntl.h>
#include <sys/select.h>
#include <sys/time.h>
#include <errno.h>
#include <string.h>
#define I2C0_DEV_NAME  "/dev/i2c-0"
#define I2C1_DEV_NAME  "/dev/i2c-1"
#define I2C2_DEV_NAME  "/dev/i2c-2"
#define I2C3_DEV_NAME  "/dev/i2c-3"
#define I2C_RETRIES    0x0701
#define I2C_TIMEOUT    0x0702
#define I2C_SLAVE      0x21
#define I2C_RDWR       0x0707
#define I2C_BUS_MODE   0x0780
#define I2C_M_RD       0x01
#define PCB_ID_VER_A   0x1
#define PCB_ID_VER_B   0x2
#define PCB_ID_VER_C   0x3
#define PCB_ID_VER_D   0x4
#define I2C_SLAVE_PCA9555_BOARDINFO (0x20)
#define BOARD_ID_DEVELOP_C (0xCE)
#define DEVELOP_A_BOM_PCB_MASK (0xF)
#define DEVELOP_C_BOM_PCB_MASK (0x7)

typedef unsigned char uint8;
typedef unsigned short uint16;
struct i2c_msg
{
    uint16 addr; /* slave address */
    uint16 flags;
    uint16 len;
    uint8 *buf; /*message data pointer*/
};
struct i2c_rdwr_ioctl_data
{
    struct i2c_msg *msgs; /*i2c_msg[] pointer*/
    int nmsgs; /*i2c_msg Nums*/
};
static uint8 i2c_init(char *i2cdev_name);
static uint8 i2c_read(uint8 slave, unsigned char reg,unsigned char *buf);
int fd = 0;

static uint8 i2c_read(unsigned char slave, unsigned char reg,unsigned char *buf)
{
    int ret;
    struct i2c_rdwr_ioctl_data ssm_msg;
    unsigned char regs[2] = {0};

    regs[0] = reg;
    regs[1] = reg;
    ssm_msg.nmsgs=2;
    ssm_msg.msgs=(struct i2c_msg*)malloc(ssm_msg.nmsgs*sizeof(struct i2c_msg));

    if(!ssm_msg.msgs)
    {
        printf("Memory alloc error!\n");
        return -1;
    }
    (ssm_msg.msgs[0]).flags=0;
    (ssm_msg.msgs[0]).addr=slave;
    (ssm_msg.msgs[0]).buf= regs;
    (ssm_msg.msgs[0]).len=1;

    (ssm_msg.msgs[1]).flags=I2C_M_RD;
    (ssm_msg.msgs[1]).addr=slave;
    (ssm_msg.msgs[1]).buf=buf;
    (ssm_msg.msgs[1]).len=2;
    ret=ioctl(fd, I2C_RDWR, &ssm_msg);
    if(ret<0)
    {
        printf("read data error,ret=%#x, errorno=%#x, %s!\n",ret, errno, strerror(errno));
        free(ssm_msg.msgs);
        return -1;
    }

    free(ssm_msg.msgs);
    return 0;
}

static uint8 i2c_init(char *i2cdev_name)
{
    fd = open(i2cdev_name, O_RDWR);
    if(fd < 0)
    {
        printf("Can't open %s!\n", i2cdev_name);
        return -1;
    }

    if(ioctl(fd, I2C_RETRIES, 1)<0)
    {
        printf("set i2c retry fail!\n");
        return -1;
    }

    if(ioctl(fd, I2C_TIMEOUT, 1)<0)
    {
        printf("set i2c timeout fail!\n");
        return -1;
    }
    return 0;
}

int main(int argc, char *argv[])
{
    char *dev_name = I2C0_DEV_NAME;
    uint8 board_id;
    uint8 pcb_id;
    uint8 buff[2] = {0};
    uint8 ret;

    if (i2c_init(dev_name))
    {
        printf("i2c init fail!\n");
        close(fd);
        return -1;
    }
    usleep(1000*100);

    ret = i2c_read(I2C_SLAVE_PCA9555_BOARDINFO, 0x0, buff);
    if (ret != 0)
    {
        printf("read %s %#x fail, ret %d\n", dev_name, I2C_SLAVE_PCA9555_BOARDINFO,  ret);
    }

    close(fd);

    board_id = buff[0];
    if (board_id == BOARD_ID_DEVELOP_C)
    {
        pcb_id = (buff[1]>>3)&DEVELOP_C_BOM_PCB_MASK;
    }
    else
    {
        pcb_id = (buff[1]>>4)&DEVELOP_A_BOM_PCB_MASK;
    }
    // show PCB ID;
    switch (pcb_id)
    {
    case PCB_ID_VER_A:
        printf("PCB version is: Ver.A !\n");
        break;
    case PCB_ID_VER_B:
        printf("PCB version is: Ver.B !\n");
        break;
    case PCB_ID_VER_C:
        printf("PCB version is: Ver.C !\n");
        break;
    case PCB_ID_VER_D:
        printf("PCB version is: Ver.D !\n");
        break;
    default:
        break;
    }
    return 0;
}
```

Step 2 Compile the file to obtain an executable file for obtaining the PCB version number.

Run the following command to compile the **i2c_tool_atlas200dk.c** file into a file that can be executed on the developer board:

aarch64-linux-gnu-gcc *i2c_tool_atlas200dk.c* **-o** *atlas200dk_version_tool*

atlas200dk_version_tool indicates the name of the executable file.

Step 3 Upload the executable file generated in **Step 2** to the developer board.

For example, upload the file to the home directory of the **HwHiAiUser** user of the developer board.

scp *atlas200dk_version_tool* **HwHiAiUser@***192.168.1.2***:/home/HwHiAiUser**

Step 4 Log in to the developer board as the **HwHiAiUser** user in SSH mode and query the PCB version number of the developer board.

ssh HwHiAiUser@*192.168.1.2*

Switch to the **root** user and run the query script.

su root

./atlas200dk_version_tool

The following information is displayed, indicating that the developer board is a VB version.

```
root@davinci-mini:/home/HwHiAiUser# ./atlas200dk_version_tool
PCB version is: Ver.B !
```

----End

7.5 Checking the Version of the Atlas 200 AI Accelerator Module

Method 1 (CLI Mode)

Step 1 Log in to the Atlas 200 DK developer board as the **HwHiAiUser** user in SSH mode.

Step 2 Run the following command to check the version of the Atlas 200 AI accelerator module:

cat /proc/cmdline

```
console=ttyAMA0,115200 root=/dev/mmcblk1p1 rw rootdelay=1 syslog no_console_suspend
earlycon=pl011,mmio32,0x10cf80000 initrd=0x880004000,200M cma=256M@0x1FC00000
log_redirect=0x1fc000@0x6fe04000 default_hugepagesz=2M reboot_reason=AP_S_COLDBOOT
himntn=1110001000000000000000000000000000000000000000000000000000000000
kmemdump=0x7C00020 slotid=00 boardid=000 nr_hugepages=25
```

Determine the version of the Atlas 200 AI accelerator module based on the value of **boardid**.

- **boardid** = **000**: Indicates that the Atlas 200 AI accelerator module is a VC or earlier version.
- **boardid** = **004**: Indicates that the Atlas 200 AI accelerator module is a VD version.

----End

Method 2 (Checking the PCB Version of the Atlas 200 AI Acceleration Module)

Step 1 Create a code file for querying the version number of the developer board.

Create the **i2c_tool_mini.c** file in any directory on the Mind Studio server as the Mind Studio installation user.

touch i2c_tool_mini.c

Copy the following code to the **i2c_tool_mini.c** file:

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/ioctl.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <sys/select.h>
#include <sys/time.h>
#include <errno.h>
#include <string.h>
#define I2C0_DEV_NAME  "/dev/i2c-0"
#define I2C1_DEV_NAME  "/dev/i2c-1"
#define I2C2_DEV_NAME  "/dev/i2c-2"
#define I2C3_DEV_NAME  "/dev/i2c-3"
#define I2C_RETRIES    0x0701
#define I2C_TIMEOUT    0x0702
#define I2C_SLAVE      0x21
#define I2C_RDWR       0x0707
```

```
#define I2C_BUS_MODE   0x0780
#define I2C_M_RD       0x01
#define PCB_ID_VER_A   0x10
#define PCB_ID_VER_B   0x20
#define PCB_ID_VER_C   0x30
#define PCB_ID_VER_D   0x40
typedef unsigned char uint8;
typedef unsigned short uint16;
struct i2c_msg
{
    uint16 addr; /* slave address */
    uint16 flags;
    uint16 len;
    uint8 *buf; /*message data pointer*/
};
struct i2c_rdwr_ioctl_data
{
    struct i2c_msg *msgs; /*i2c_msg[] pointer*/
    int nmsgs; /*i2c_msg Nums*/
};
static uint8 i2c_init(char *i2cdev_name);
static uint8 i2c_write(uint8 slave, unsigned char reg, unsigned char value);
static uint8 i2c_read(uint8 slave, unsigned char reg,unsigned char *buf);
int fd = 0;
static uint8 i2c_write(uint8 slave, unsigned char reg, unsigned char value)
{
    int ret;
    struct i2c_rdwr_ioctl_data ssm_msg;
    unsigned char buf[2]={0};

    ssm_msg.nmsgs=1;
    ssm_msg.msgs=(struct i2c_msg*)malloc(ssm_msg.nmsgs*sizeof(struct i2c_msg));
    if(!ssm_msg.msgs)
    {
        printf("Memory alloc error!\n");
        return -1;
    }

    buf[0] = reg;
    buf[1] = value;

    (ssm_msg.msgs[0]).flags=0;
    (ssm_msg.msgs[0]).addr=(uint16)slave;
    (ssm_msg.msgs[0]).buf=buf;
    (ssm_msg.msgs[0]).len=2;

    ret=ioctl(fd, I2C_RDWR, &ssm_msg);
    if(ret<0)
    {
        printf("write error, ret=%#x, errorno=%#x, %s!\n",ret, errno, strerror(errno));
        free(ssm_msg.msgs);
        return -1;
    }

    free(ssm_msg.msgs);
    return 0;
}
static uint8 i2c_read(unsigned char slave, unsigned char reg,unsigned char *buf)
{
    int ret;
    struct i2c_rdwr_ioctl_data ssm_msg;
    unsigned char regs[2] = {0};

    regs[0] = reg;
    regs[1] = reg;
    ssm_msg.nmsgs=2;
    ssm_msg.msgs=(struct i2c_msg*)malloc(ssm_msg.nmsgs*sizeof(struct i2c_msg));

    if(!ssm_msg.msgs)
    {
        printf("Memory alloc error!\n");
        return -1;
    }
    (ssm_msg.msgs[0]).flags=0;
    (ssm_msg.msgs[0]).addr=slave;
    (ssm_msg.msgs[0]).buf= regs;
    (ssm_msg.msgs[0]).len=1;

    (ssm_msg.msgs[1]).flags=I2C_M_RD;
    (ssm_msg.msgs[1]).addr=slave;
    (ssm_msg.msgs[1]).buf=buf;
    (ssm_msg.msgs[1]).len=1;

    ret=ioctl(fd, I2C_RDWR, &ssm_msg);
    if(ret<0)
    {
        printf("read data error,ret=%#x, errorno=%#x, %s!\n",ret, errno, strerror(errno));
        free(ssm_msg.msgs);
        return -1;
    }

    free(ssm_msg.msgs);
    return 0;
}
static uint8 i2c_init(char *i2cdev_name)
{
    fd = open(i2cdev_name, O_RDWR);
    if(fd < 0)
    {
        printf("Can't open %s!\n", i2cdev_name);
        return -1;
    }

    if(ioctl(fd, I2C_RETRIES, 1)<0)
    {
        printf("set i2c retry fail!\n");
        return -1;
    }

    if(ioctl(fd, I2C_TIMEOUT, 1)<0)
    {
        printf("set i2c timeout fail!\n");
        return -1;
    }
    return 0;
}
int main(int argc, char *argv[])
{
    char *dev_name = I2C0_DEV_NAME;
    uint8 slave;
    uint8 reg;
    uint8 data;
    int ret;

    if (i2c_init(dev_name))
    {
        printf("i2c init fail!\n");
        close(fd);
        return -1;
    }
    usleep(1000*100);

    // Read PCB ID
    slave = I2C_SLAVE;
    reg = 0x07;
    data = 0x5A;

    ret = i2c_read(slave, reg, &data);
    if (ret != 0)
    {
        printf("read %s %#x %#x to %#x fail!\n", dev_name, slave, data, reg);
    }

    slave = I2C_SLAVE;
    reg = 0x07;
    data = data|0xF0;

    ret = i2c_write(slave, reg, data);
    if (ret != 0)
    {
        printf("write %s %#x %#x to %#x fail!\n", dev_name, slave, data, reg);
    }

    slave = I2C_SLAVE;
    reg = 0x01;
    data = 0x5A;

    ret = i2c_read(slave, reg, &data);
    if (ret != 0)
    {
        printf("read %s %#x %#x to %#x fail!\n", dev_name, slave, data, reg);
    }

    close(fd);

    // show PCB ID;
    switch (data & 0xF0)
    {
    case PCB_ID_VER_A:
        printf("PCB version is: Ver.A !\n");
        break;
    case PCB_ID_VER_B:
        printf("PCB version is: Ver.B !\n");
        break;
    case PCB_ID_VER_C:
        printf("PCB version is: Ver.C !\n");
        break;
    case PCB_ID_VER_D:
        printf("PCB version is: Ver.D !\n");
        break;
    default:
        break;
    }
    return 0;
}
```

Step 2 Compile the file to obtain an executable file for obtaining the PCB version number.

Run the following command to compile the **i2c_tool_mini.c** file to a file that can be executed on the developer board:

aarch64-linux-gnu-gcc *i2c_tool_mini.c* **-o** *mini_version_tool*

mini_version_tool indicates the name of the executable file.

Step 3 Upload the executable file generated in **Step 2** to the developer board.

For example, upload the file to the home directory of the **HwHiAiUser** user of the developer board.

scp *mini_version_tool* **HwHiAiUser@***192.168.1.2***:/home/HwHiAiUser**

Step 4 Log in to the developer board as the **HwHiAiUser** user in SSH mode and query the PCB version number of the developer board.

ssh HwHiAiUser@*192.168.1.2*

Switch to the **root** user and run the query script.

su root

./mini_version_tool

The following information is displayed, indicating the Atlas 200 AI accelerator module is a VC version.

```
root@davinci-mini:/home/HwHiAiUser# ./mini_version_tool
PCB version is: Ver.C !
```

----End

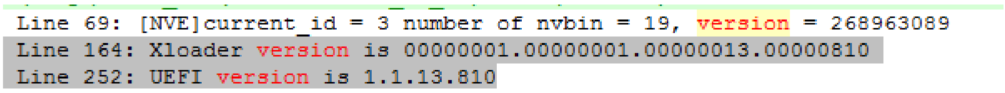
7.6 Checking the Firmware Version of the Developer Board

If you can log in to the OS of the Atlas 200 DK developer board, check the firmware version by referring to **6 Upgrading the Atlas 200 DK**. If you cannot log in to the OS of the Atlas 200 DK developer board, check the firmware version by viewing the startup logs of the Atlas 200 DK serial port as follows:

Step 1 Connect to the Atlas 200 AI accelerator module of the Atlas 200 DK over a serial port. For details, see **7.3 Connecting to the Atlas 200 DK Developer Board over a Serial Port**.

Step 2 Power on the Atlas 200 DK developer board and view the log on the serial port terminal. The firmware version is printed, as shown in the following figure.

Figure 7-4 Querying the firmware version



```
Line 69: [NVE]current_id = 3 number of nvbin = 19, version = 268963089
Line 164: Xloader version is 00000001.00000001.00000013.00000810
Line 252: UEFI version is 1.1.13.810
```

- The Xloader version number is **1.1.T13.B810**.
- The UEFT version number is **1.1.T13.B810**.

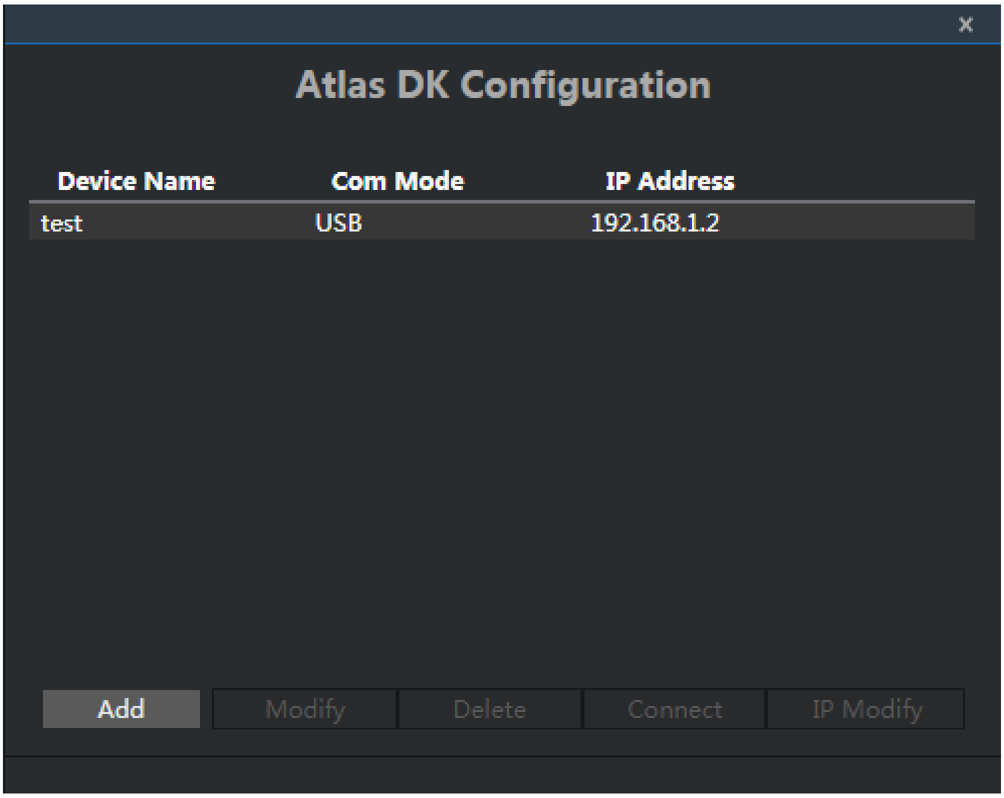
----End

7.7 Viewing the Channel to Which a Camera Belongs

To query the channel to which a camera belongs, perform the following steps:

Step 1 On Mind Studio, choose **Tools > Atlas DK Configuration** from the main menu. The **Atlas DK Configuration** interface is displayed, as shown in **Figure 7-5**.

Figure 7-5 Atlas 200 DK Configuration dialog box



Atlas DK Configuration

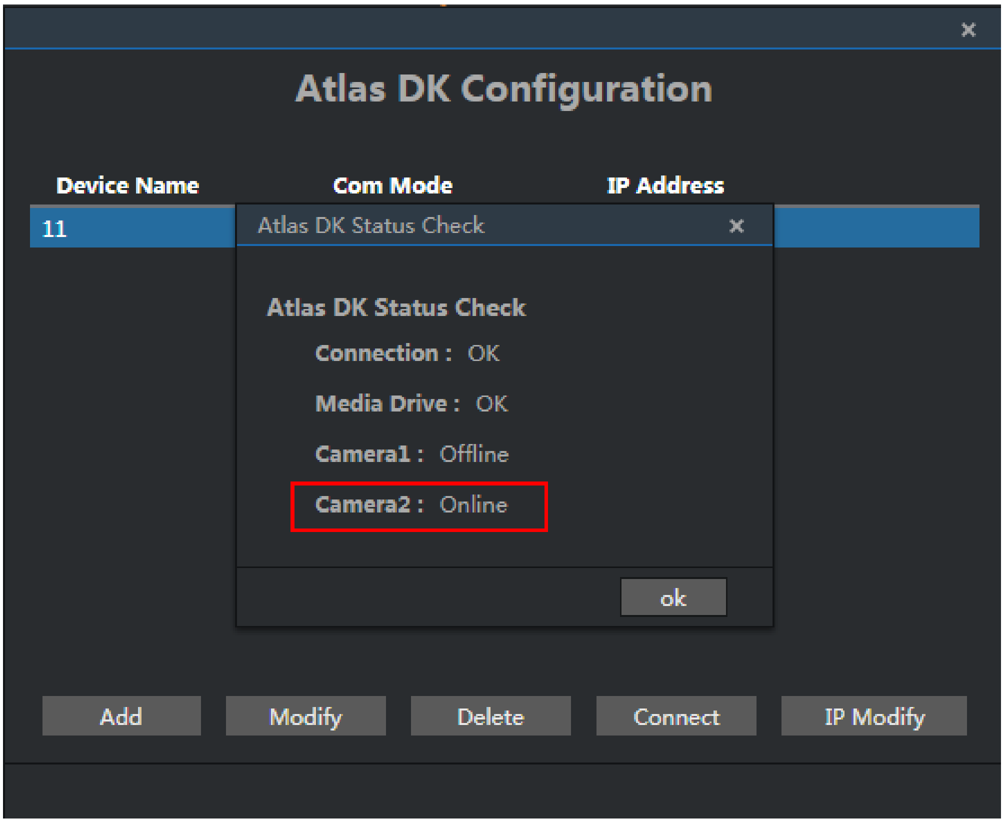
Device Name	Com Mode	IP Address
test	USB	192.168.1.2

NOTE

If no developer board connection exists on the configuration page, click **Add** to add a connection. For details, see **7.8 Managing the Atlas 200 DK Developer Board**

Step 2 Select the Atlas DK connection in use and click **Connect** to check the connection status of the Atlas DK, as shown in **Figure 7-6**.

Figure 7-6 Atlas 200 DK status dialog box



Atlas DK Configuration

Atlas DK Status Check

Connection : OK

Media Drive : OK

Camera1 : Offline

Camera2 : Online

In the preceding figure, **Camera2** is set to **Online**, indicating that the channel to which that the current camera belongs is **Channel-2**.

----End

7.8 Managing the Atlas 200 DK Developer Board

Mind Studio provides functions for adding, modifying, deleting, and connecting Atlas 200 DK developer boards, and modifying their IP addresses.

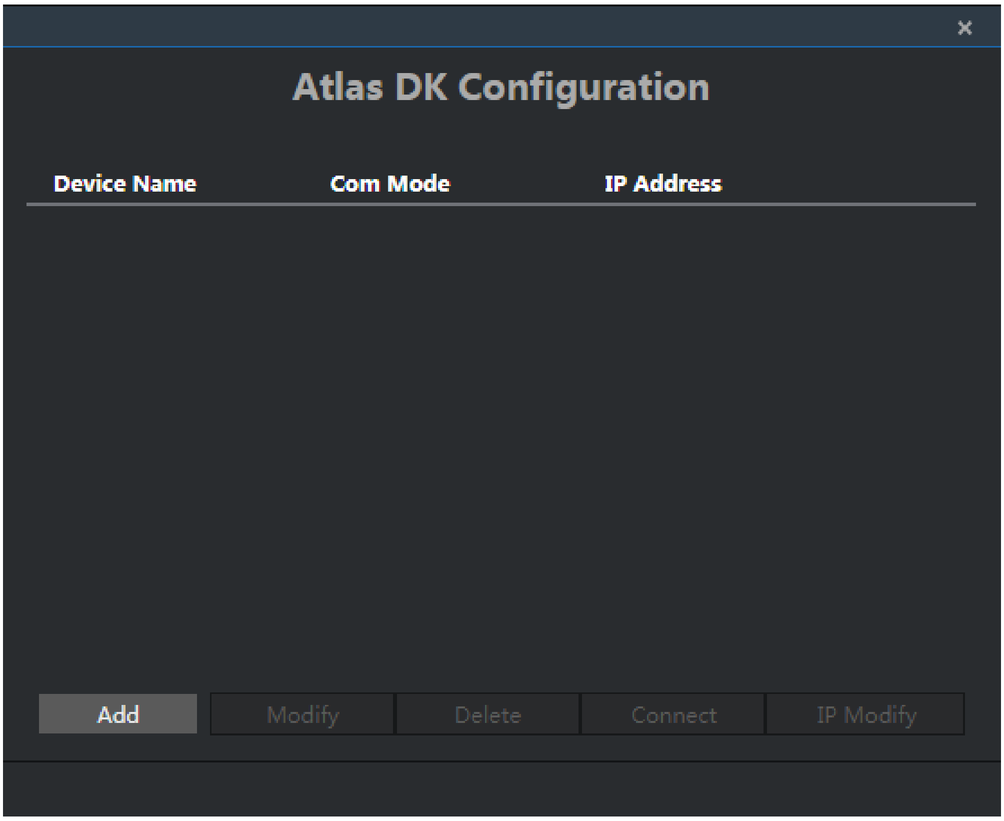
NOTE

The current configuration management function supports only single-user and single-task scenarios. Multi-user and multi-task scenarios are not supported

Accessing the Function

On the menu bar of Mind Studio, choose **Tools > Atlas DK Configuration**, as shown in Figure 7-7.

Figure 7-7 Atlas DK Configuration

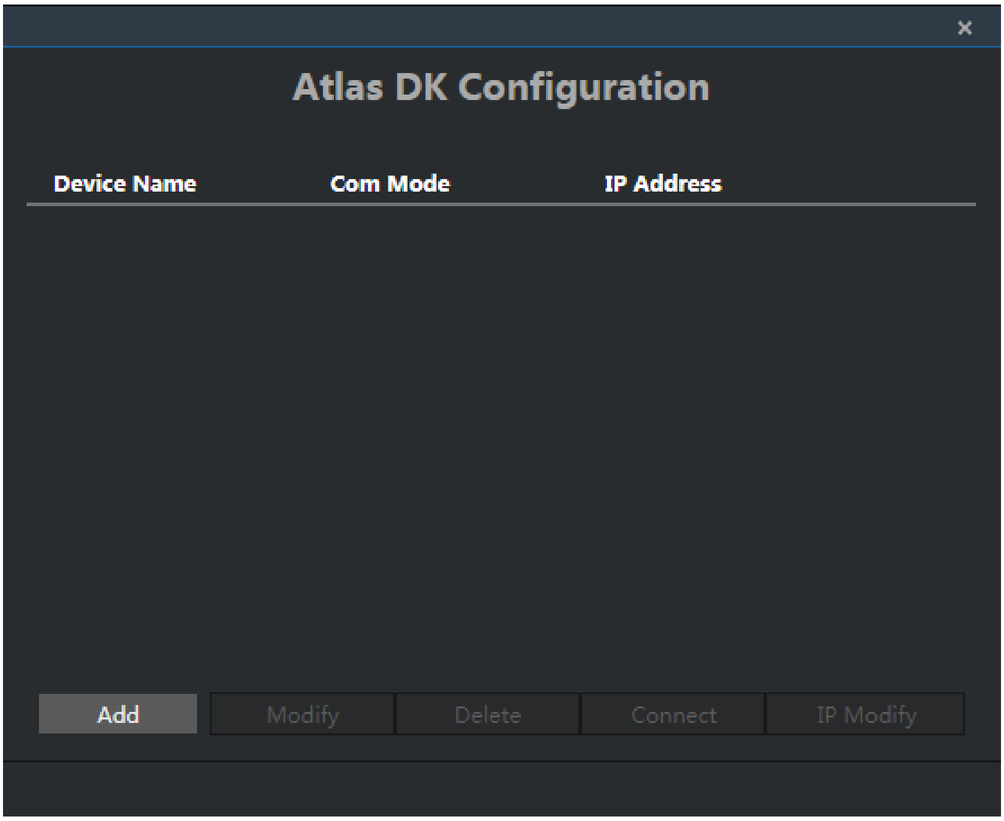


Adding a Device

Step 1 On the menu bar of Mind Studio, choose **Tools > Atlas DK Configuration**.

The **Atlas DK Configuration** dialog box is displayed, as shown in **Figure 7-8**.

Figure 7-8 Atlas DK Configuration dialog box



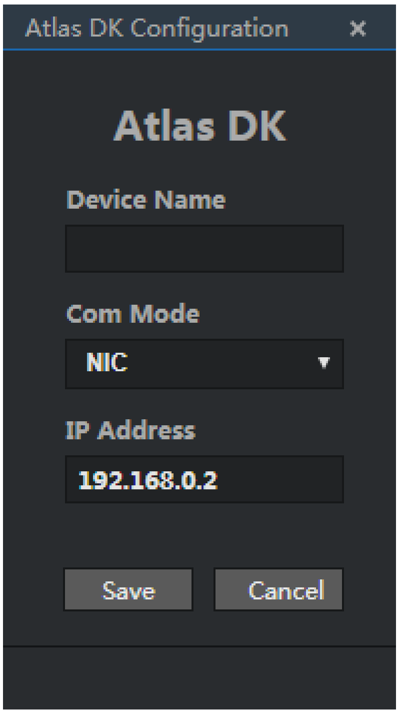
Step 2 Click **Add**.

The device information dialog box is displayed.

Step 3 Configure the basic information about the Atlas 200 DK to be added.

See **Figure 7-9**.

Figure 7-9 Atlas DK Configuration > Add



Atlas DK Configuration

Atlas DK

Device Name:

Com Mode: NIC

IP Address: 192.168.0.2

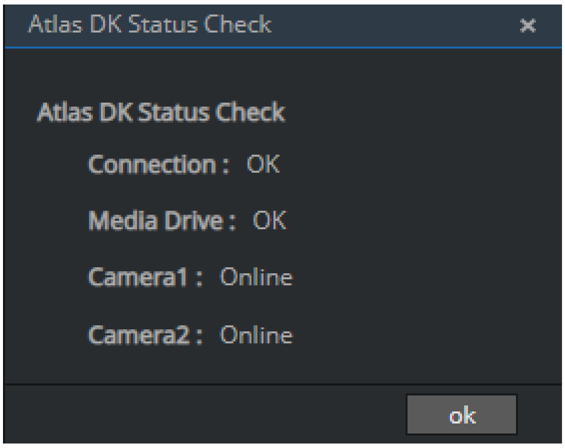
The configuration parameters are described as follows:

- **Device Name**: user-defined device name
- **Com Mode**: communication mode, which can be **NIC** or **USB**
  - **NIC** indicates using general network communication.
  - **USB** indicates using a USB virtual NIC.
- **IP Address**: IP address of the Atlas 200 DK device

Step 4 Click **Save**.

After the configuration is complete, the **Atlas 200 DK Status Check** dialog box is displayed, showing the status of each module, as shown in **Figure 7-10**.

Figure 7-10 Atlas DK Status Check dialog box



Atlas DK Status Check

Atlas DK Status Check

Connection : OK

Media Drive : OK

Camera1 : Online

Camera2 : Online

The parameters are described as follows:

- **Connection**: communication status between Mind Studio and the Atlas 200 DK developer board
- **Media Drive**: status of the audio and video processing module (Hi3559)
- **Camera1**: whether channel 1 is connected to a camera
- **Camera2**: whether channel 2 is connected to a camera

NOTE

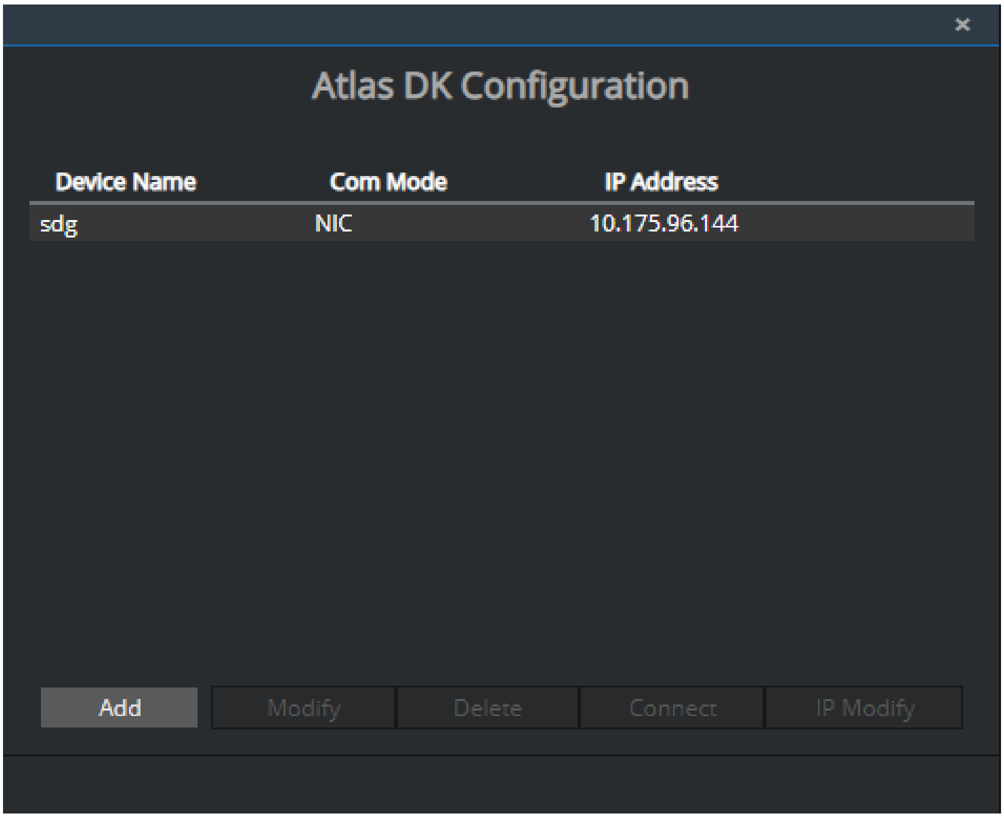
- During device addition, the network connection between the PC and the Atlas 200 DK device is checked. If the network connection is normal, the configuration information of the Atlas 200 DK device is added to the **Atlas DK Configuration** device list. Otherwise, a message is displayed indicating that the connection fails and the device configuration information is not added to the **Atlas DK Configuration** device list.
- If a USB virtual NIC is used for communication with the Atlas 200 DK device, the IP address of Mind Studio must be the default setting **192.168.1.2**, and the IP address of the USB virtual NIC on the PC must be on the same network segment as **192.168.1.2**. The default subnet mask is **255.255.255.0**. You can also use terminal software (such as Xshell) to access the system terminal of the Atlas 200 DK device over a NIC or USB virtual NIC.

----End

Modifying a Device

Select a device from the **Atlas DK Configuration** device list. Otherwise, the **Modify** option is unavailable, as shown in **Figure 7-11**.

Figure 7-11 Atlas DK Configuration dialog box

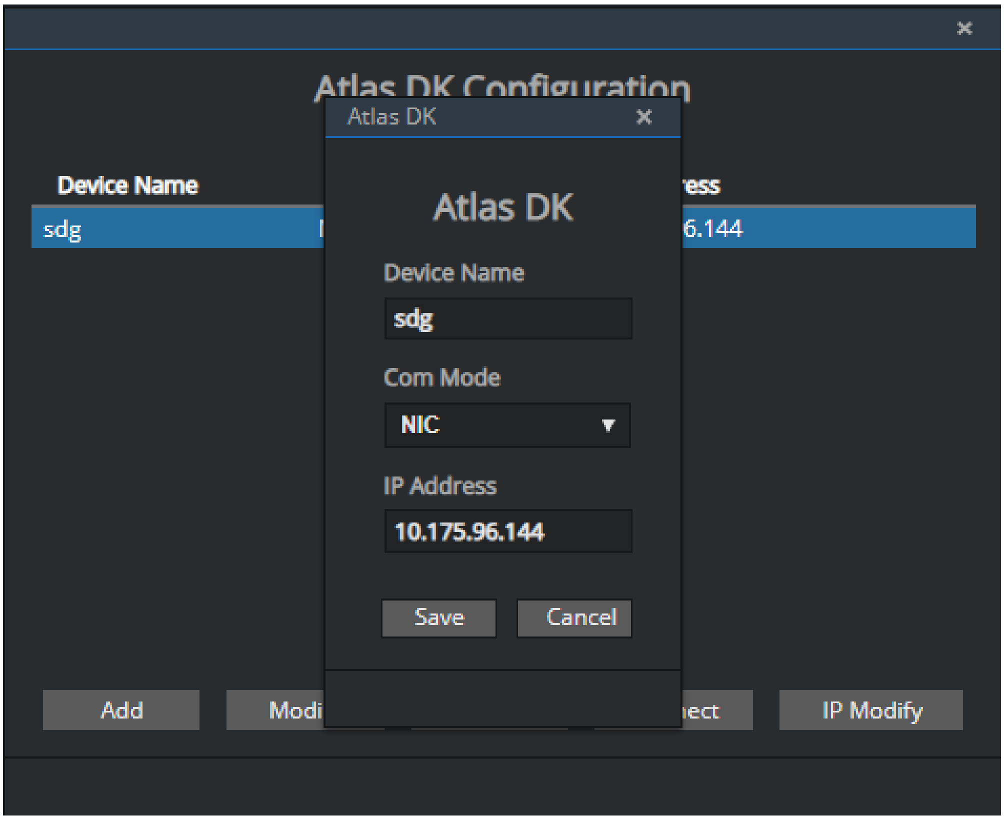


Atlas DK Configuration

Device Name	Com Mode	IP Address
sdg	NIC	10.175.96.144

Select the device to be modified, click **Modify** to modify the device configuration information, and click **Save**, as shown in Figure 7-12.

Figure 7-12 Modifying the Atlas DK configuration



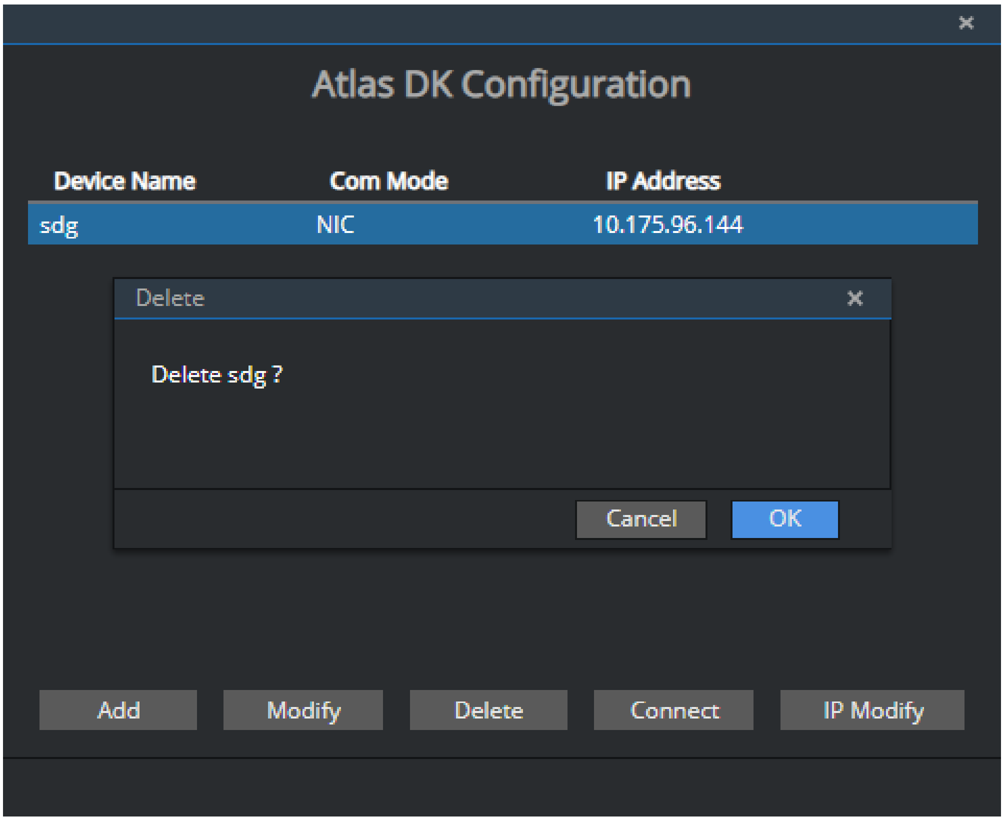
Device Name: sdg
Com Mode: NIC
IP Address: 10.175.96.144

Deleting a Device

Select a device from the **Atlas DK Configuration** device list. Otherwise, the **Delete** button is unavailable, as shown in Figure 7-11.

Select the device to be deleted and click **Delete**, as shown in Figure 7-13.

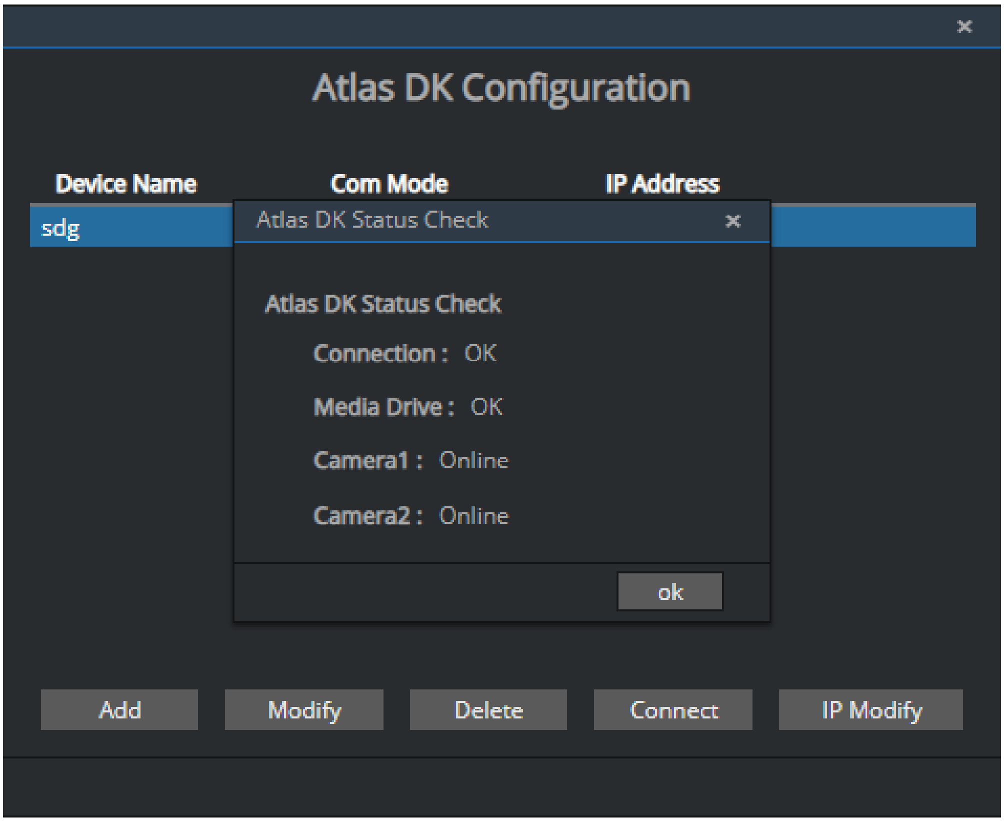
Figure 7-13 Atlas DK Configuration > Delete



Connecting a Device

Select the device to be operated from the **Atlas DK Configuration** device list, and click **Connect**, as shown in **Figure 7-14**.

Figure 7-14 Atlas DK Configuration > Connect



Atlas DK Configuration

Device Name	Com Mode	IP Address
sdg		

Atlas DK Status Check

Atlas DK Status Check

Connection : OK

Media Drive : OK

Camera1 : Online

Camera2 : Online

For details about the description of the parameters, see **Adding a Device**.

7.9 Verifying Software Integrity

Before installing the software package, you are advised to check whether the software package is incomplete or damaged due to network or storage device faults.

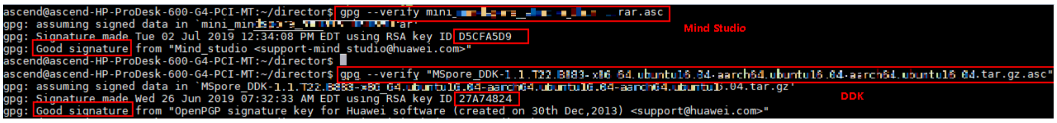
In the **director** directory where the installation packages are located, perform the following steps:

Step 1 Configure the OpenPGP public keys. For details, see **7.10 Configuring OpenPGP Public Keys**.

Step 2 Run the following commands as the Mind Studio installation user to check whether the Mind Studio and DDK software packages are valid and integrated, as shown in **Figure 7-15**.

- Mind Studio:
  ```
  gpg --verify "mini_mind_studio_*.rar.asc"
  ```
- DDK:
  ```
  gpg --verify "MSpore_DDK****tar.gz.asc"
  ```

Figure 7-15 Verifying the software package integrity



```
ascend@ascend-HP-ProDesk-600-G4-PCI-MT:~/director$ gpg --verify mini_[illegible] rar.asc
gpg: assuming signed data in `mini_min[illegible]ar'
gpg: Signature made Tue 02 Jul 2019 12:34:08 PM EDT using RSA key ID D5CFA5D9
gpg: Good signature from "Mind_studio <support-mind_studio@huawei.com>"
ascend@ascend-HP-ProDesk-600-G4-PCI-MT:~/director$
ascend@ascend-HP-ProDesk-600-G4-PCI-MT:~/director$ gpg --verify "MSpore_DDK-1.1.T22.B883-x86_64.ubuntu16.04-aarch64.ubuntu16.04-aarch64.ubuntu16.04.tar.gz.asc"
gpg: assuming signed data in `MSpore_DDK-1.1.T22.B883-x86_64.ubuntu16.04-aarch64.ubuntu16.04-aarch64.ubuntu16.04.tar.gz'
gpg: Signature made Wed 26 Jun 2019 07:32:33 AM EDT using RSA key ID 27A74824
gpg: Good signature from "OpenPGP signature key for Huawei software (created on 30th Dec,2013) <support@huawei.com>"
```

- In the command output, **D5CFA5D9** indicates the public key ID of Mind Studio and **27A74824** indicates the public key ID of the DDK.
- If the message **Good signature** is displayed without **WARNING** or **FAIL**, the signature is valid and the integrity verification is passed.
- If **WARNING** or **FAIL** is displayed, the verification fails. Rectify the fault by referring to the handling suggestions described in **8.3 What Do I Do If a WARNING or FAIL Result Is Returned in the Integrity Verification of a Software Package?**.

NOTE

- Replace **mini_mind_studio_*.rar.asc** and **MSpore_DDK****tar.gz.asc** with the actual verification files of the installation packages.
- The integrity verification can be performed only when the software package and the .asc file are stored in the same path.

----End

7.10 Configuring OpenPGP Public Keys

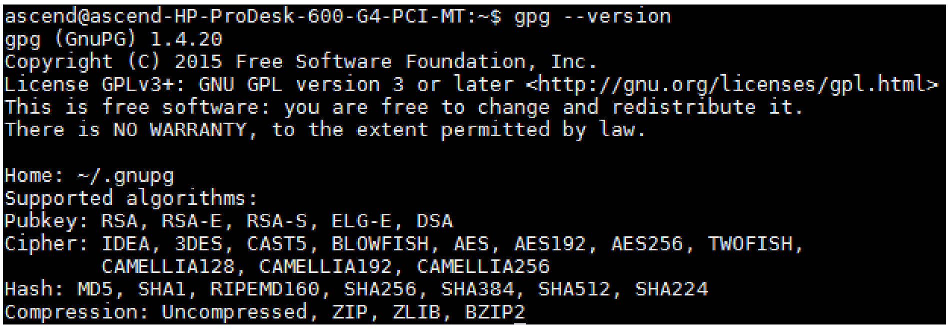
Prerequisites

- The public keys are configured by the installation user of Mind Studio.
- The GnuPG tool is installed on Linux.

  Verification method:

  – If the GnuPG tool has been installed, run the **gpg --version** command in the shell. Information in **Figure 7-16** is displayed:

  Figure 7-16 Command output



```
ascend@ascend-HP-ProDesk-600-G4-PCI-MT:~$ gpg --version
gpg (GnuPG) 1.4.20
Copyright (C) 2015 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Home: ~/.gnupg
Supported algorithms:
Pubkey: RSA, RSA-E, RSA-S, ELG-E, DSA
Cipher: IDEA, 3DES, CAST5, BLOWFISH, AES, AES192, AES256, TWOFISH,
        CAMELLIA128, CAMELLIA192, CAMELLIA256
Hash: MD5, SHA1, RIPEMD160, SHA256, SHA384, SHA512, SHA224
Compression: Uncompressed, ZIP, ZLIB, BZIP2
```

  – If the GnuPG tool is not installed, install the tool by following the instructions provided in its official website **https://www.gnupg.org/**.

Configuring Public Keys

Step 1 Obtain the public key files.

NOTE

To differentiate the public key used by Mind Studio and that used by the DDK, you are advised to rename the downloaded public key files or import the files to different directories. In this example, the public key files are renamed as follows: **KEYS_mind_studio.txt** for Mind Studio and **KEYS_DDK.txt** for the DDK.

- To obtain the Mind Studio public key, perform the following steps:

  Upload the **KEYS.zip** file in the **resource** folder extracted from the .zip package to the Linux OS where Mind Studio is installed,

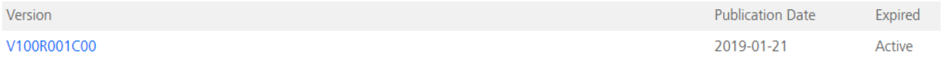
  for example, **home/*username*/openpgp/keys** directory. Run the following command to decompress the package, and then rename the public key file **KEYS_mind_studio.txt**:

  ```
  unzip KEYS.zip
  ```

- To obtain the DDK public key, perform the following steps:

1. Go to the **OpenPGP download page** and click the download link, as shown in **Figure 7-17**. The file download page is displayed.

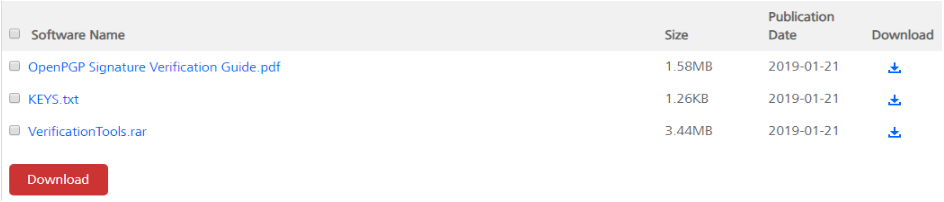
   Figure 7-17 OpenPGP download page

Version	Publication Date	Expired
V100R001C00	2019-01-21	Active

   The **KEYS** file is the public key file, as shown in **Figure 7-18**.

   Figure 7-18 Selecting the KEYS file

☐ Software Name	Size	Publication Date	Download
☐ OpenPGP Signature Verification Guide.pdf	1.58MB	2019-01-21	⤓
☐ KEYS.txt	1.26KB	2019-01-21	⤓
☐ VerificationTools.rar	3.44MB	2019-01-21	⤓

   NOTE

   To switch to the language, click 🌐 Worldwide ⌄ in the upper right corner.

2. Rename the downloaded **KEYS.txt** file **KEYS_DDK.txt** and upload it to the Linux OS where Mind Studio is installed,

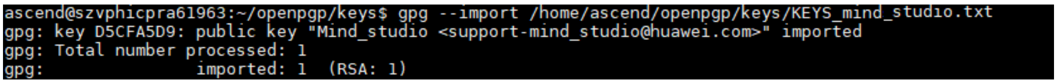
   for example, **/home/*username*/openpgp/keys**.

Step 2 Import the public key files.

Run the following command to go to the directory where a public key file is stored and import the public key: (The following uses the Mind Studio public key as an example. Import the DDK public key the same way.)

```
gpg --import "/home/username/openpgp/keys/KEYS_mind_studio.txt"
```

Figure 7-19 Importing a public key file



```
ascend@szvphicpra61963:~/openpgp/keys$ gpg --import /home/ascend/openpgp/keys/KEYS_mind_studio.txt
gpg: key D5CFA5D9: public key "Mind_studio <support-mind_studio@huawei.com>" imported
gpg: Total number processed: 1
gpg:               imported: 1  (RSA: 1)
```

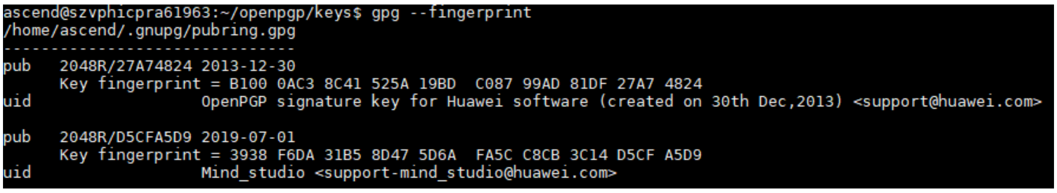
NOTE

/home/*username*/openpgp/keys indicates the absolute path of the public key file **KEYS**. ***username*** must be replaced with the name of the Mind Studio installation user.

Step 3 Run the following command to view the import result:

```
gpg --fingerprint
```

Figure 7-20 Checking the import result



```
ascend@szvphicpra61963:~/openpgp/keys$ gpg --fingerprint
/home/ascend/.gnupg/pubring.gpg
------------------------------
pub   2048R/27A74824 2013-12-30
      Key fingerprint = B100 0AC3 8C41 525A 19BD  C087 99AD 81DF 27A7 4824
uid                  OpenPGP signature key for Huawei software (created on 30th Dec,2013) <support@huawei.com>

pub   2048R/D5CFA5D9 2019-07-01
      Key fingerprint = 3938 F6DA 31B5 8D47 5D6A  FA5C C8CB 3C14 D5CF A5D9
uid                  Mind_studio <support-mind_studio@huawei.com>
```

Step 4 Verify the public keys.

- The validity of an OpenPGP public key must be verified based on the public key ID, fingerprint, UID, and the publisher of the public key. The published information of the OpenPGP public keys is as follows:
  - Mind Studio public key information:
    1. Public key ID: **D5CFA5D9**
    2. Public key fingerprint: **3938 F6DA 31B5 8D47 5D6A FA5C C8CB 3C14 D5CF A5D9**
    3. User ID (UID): **Mind_studio <support-mind_studio@huawei.com>**
  - DDK public key information:
    1. Public key ID: **27A74824**
    2. Public key fingerprint: **B100 0AC3 8C41 525A 19BD C087 99AD 81DF 27A7 4824**
    3. UID: **OpenPGP signature key for Huawei software (created on 30th Dec,2013) <support@huawei.com>**

  NOTE

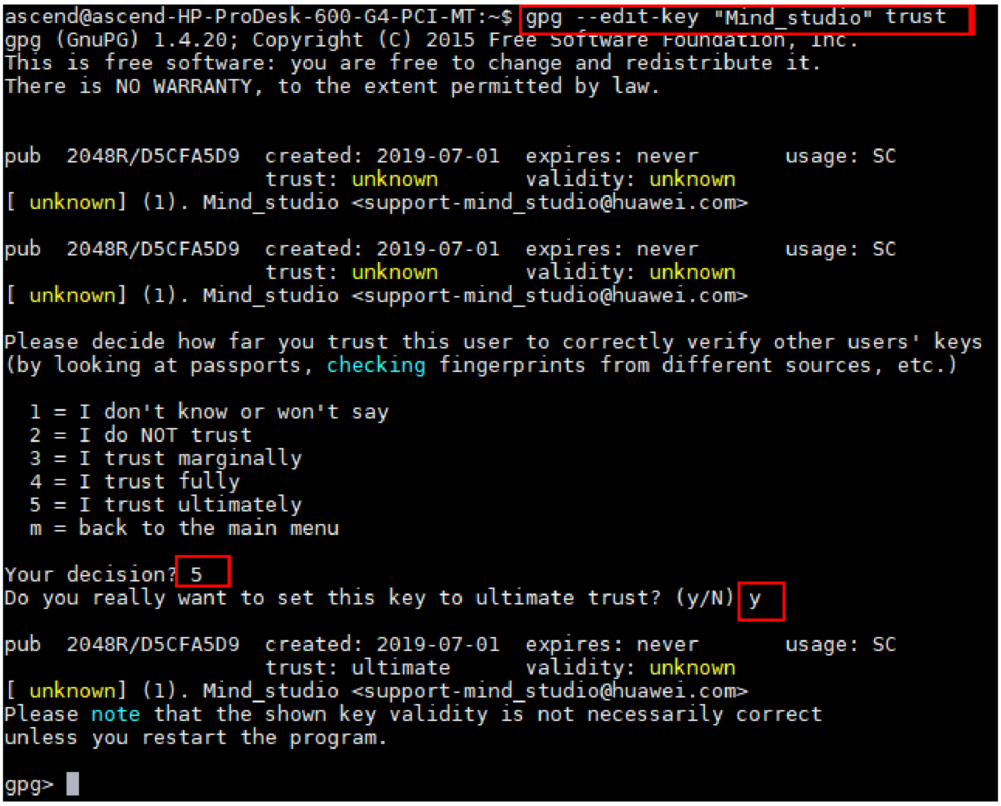
  The public key fingerprint in the public key information is only an example. For details, see the obtained public key file **KEYS.txt**.

  Verify the public key information and perform the following steps to set the trust level for the public keys (The following uses the Mind Studio public key as an example. Set the trust level of the DDK public key the same way.):

- Run the following command to set the trust levels of the public keys:
  - For Mind Studio:
    ```
    gpg --edit-key "Mind_studio" trust
    ```
  - For the DDK:
    ```
    gpg --edit-key "OpenPGP signature key for Huawei software" trust
    ```

  Information similar to the following is displayed. Enter **5** after **Your decision?**, which indicates **I trust ultimately**. Enter **y** after **Do you really want to set this key to ultimate trust? (y/N)**.

Figure 7-21 Setting the trust level for a public key



```
ascend@ascend-HP-ProDesk-600-G4-PCI-MT:~$ gpg --edit-key "Mind_studio" trust
gpg (GnuPG) 1.4.20; Copyright (C) 2015 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.


pub  2048R/D5CFA5D9  created: 2019-07-01  expires: never       usage: SC
                     trust: unknown       validity: unknown
[ unknown] (1). Mind_studio <support-mind_studio@huawei.com>

pub  2048R/D5CFA5D9  created: 2019-07-01  expires: never       usage: SC
                     trust: unknown       validity: unknown
[ unknown] (1). Mind_studio <support-mind_studio@huawei.com>

Please decide how far you trust this user to correctly verify other users' keys
(by looking at passports, checking fingerprints from different sources, etc.)

  1 = I don't know or won't say
  2 = I do NOT trust
  3 = I trust marginally
  4 = I trust fully
  5 = I trust ultimately
  m = back to the main menu

Your decision? 5
Do you really want to set this key to ultimate trust? (y/N) y

pub  2048R/D5CFA5D9  created: 2019-07-01  expires: never       usage: SC
                     trust: ultimate      validity: unknown
[ unknown] (1). Mind_studio <support-mind_studio@huawei.com>
Please note that the shown key validity is not necessarily correct
unless you restart the program.

gpg> 
```

Step 5 Run the **quit** command to exit.

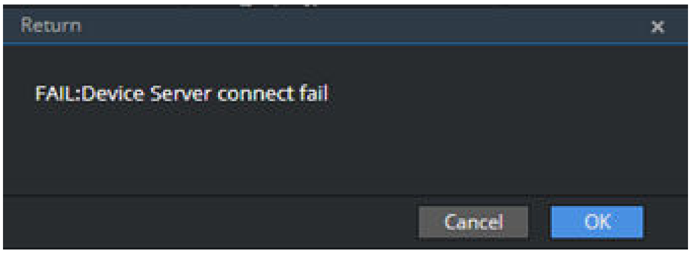
----End

8 FAQs

8.1 What Do I Do If the Network Connection Fails During Configuration Management?

If the following failure occurs, check the following items first:

- Whether the PC is properly connected to the Atlas 200 DK
- Whether the entered Atlas 200 DK IP address is correct



FAIL:Device Server connect fail

8.2 What Do I Do If a Redundant Mounted Disk Appears Due to Manual Removal of the SD Card During SD Card Creation?

If the SD card is manually removed during SD card creation a redundant temporarily mounted disk is generated. You can perform the following steps to remove it.

Step 1 Log in to the Ubuntu server where Mind Studio is located as the Mind Studio installation user and run the **su - root** command to switch to the **root** user.

Step 2 Run the **df -h** command to view the temporarily mounted **/dev/loop0** disk.

```
root@kickseed:~# df -h
Filesystem       Size    Used      Avail   Use% Mounted on
/dev/loop0       745M  745M     0        100% /home/ubuntu/studio/scripts/180919002200
/dev/sdc1        118G   60M       112G  1%    /home/ubuntu/studio/scripts/sd_mount_dir
```

Step 3 Run the **umount** command to unmount the disk. Replace */dev/loop0* and */dev/sdc1* in the commands based on the actual query result in **Step 2**.

```
root@kickseed:~# umount /dev/loop0
root@kickseed:~# umount /dev/sdc1
```

If "target is busy" is displayed, restart the Ubuntu PC and repeat **Step 1** to **Step 3**.

----End

8.3 What Do I Do If a WARNING or FAIL Result Is Returned in the Integrity Verification of a Software Package?

If a WARNING or FAIL result is returned in the integrity verification of a software package, the verification fails. Rectify the fault by referring to the handling suggestions described in **Table 8-1**.

Table 8-1 Verification result examples

Verification Result Description	Displayed Information	Verification Result	Solution
The signature verification is successful without exception.	gpg: Signature made Thu Jan 9 15:29:06 2014 CST using RSA key ID 27A74824 gpg: Good signature from "OpenPGP signature key for Huawei software (created on 30th Dec,2013) <support@huawei.com>"	PASS	N/A

Verification Result Description	Displayed Information	Verification Result	Solution
The signature verification fails.	gpg: Signature made Thu Jan 9 15:29:06 2014 CST using RSA key ID 27A74824 gpg: BAD signature from "OpenPGP signature key for Huawei software (created on 30th Dec,2013) <support@huawei.com>"	FAIL	Download the target file again.
The public key cannot be found.	gpg: Signature made Thu Jan 9 15:20:01 2014 CST using RSA key ID 27A74824 gpg: Can't check signature: public key not found	FAIL	Download the public key again. For details, see **Configuring OpenPGP Public Keys**.
The signature verification is successful, but the public key is not ultimately trusted.	gpg: Signature made Thu Jan 9 15:29:06 2014 CST using RSA key ID 27A74824 gpg: Good signature from "OpenPGP signature key for Huawei software (created on 30th Dec,2013) <support@huawei.com>" gpg: WARNING: This key is not certified with a trusted signature! gpg: There is no indication that the signature belongs to the owner. Primary key fingerprint: B100 0AC3 8C41 525A 19BD C087 99AD 81DF 27A7 4824	WARNING	After confirming that key ID (**27A74824**), set the trust level of Huawei public key to **5**. For details, see **Configuring OpenPGP Public Keys**.
The corresponding source file cannot be found.	gpg: no signed data gpg: can't hash datafile: No data	FAIL	Download the target file again.
The signature has expired.	gpg: Signature made 04/24/13 10:50:29 CST using RSA key ID 133B64E5 gpg: Expired signature from " OpenPGP signature test key <support@huawei.com>" gpg: Signature expired 04/25/13 10:50:29 CST	FAIL	Download an updated target file.

Verification Result Description	Displayed Information	Verification Result	Solution
The signature verification is successful, but the public key has been revoked.	gpg: Signature made 06/13/13 11:14:49 CST using RSA key ID 133B64E5 gpg: Good signature from " OpenPGP signature test key <support@huawei.com>" gpg: WARNING: This key has been revoked by its owner! gpg: This could mean that the signature is forged. gpg: reason for revocation: Key is no longer used gpg: revocation comment:	WARNING	Download the latest public key and an updated target file.
The corresponding signature file cannot be found in the source file.	None	WARNING	Download the signature file corresponding to the target file.

8.4 What Do I Do If the Trust Relationship with the Developer Board Fails to Be Established by Using UI Host?

Symptom

On the Ubuntu server where the UI Host is located, the following command is run to connect to the Atlas 200 DK developer board in SSH mode. A message is displayed, indicating that no trust relationship exists.

The following command is run on UI Host to re-establish the trust relationship:

ssh-keygen -f "$HOME/.ssh/known_hosts" -R ***192.168.1.2***

192.168.1.2 is the IP address of the Atlas 200 DK developer board.

The following error is reported:

```
ECDSA host key for 192.168.1.2 has changed and you have requested strict checking.
```

Solution

This error is caused by the invalid SSH information stored on the local host. Therefore, you need to clear the local SSH information and establish the connection again.

Step 1 On UI Host, clear the public key information about the connection to the 192.168.1.2 host of the current user.

ssh-keygen -R *192.168.1.2*

Step 2 Re-connect to the Atlas 200 DK developer board in SSH mode.

ssh HwHiAiUser@192.168.1.2

When the following information is displayed, enter **yes** to re-establish the SSH connection.

```
The authenticity of host '192.168.1.2' can't be established.
ECDSA key fingerprint is 53:b9:f9:30:67:ec:34:88:e8:bc:2a:a4:6f:3e:97:95.
Are you sure you want to continue connecting (yes/no)?
```

----End

8.5 What Do I Do If the Developer Board Cannot Connect to the UI Host?

Symptom

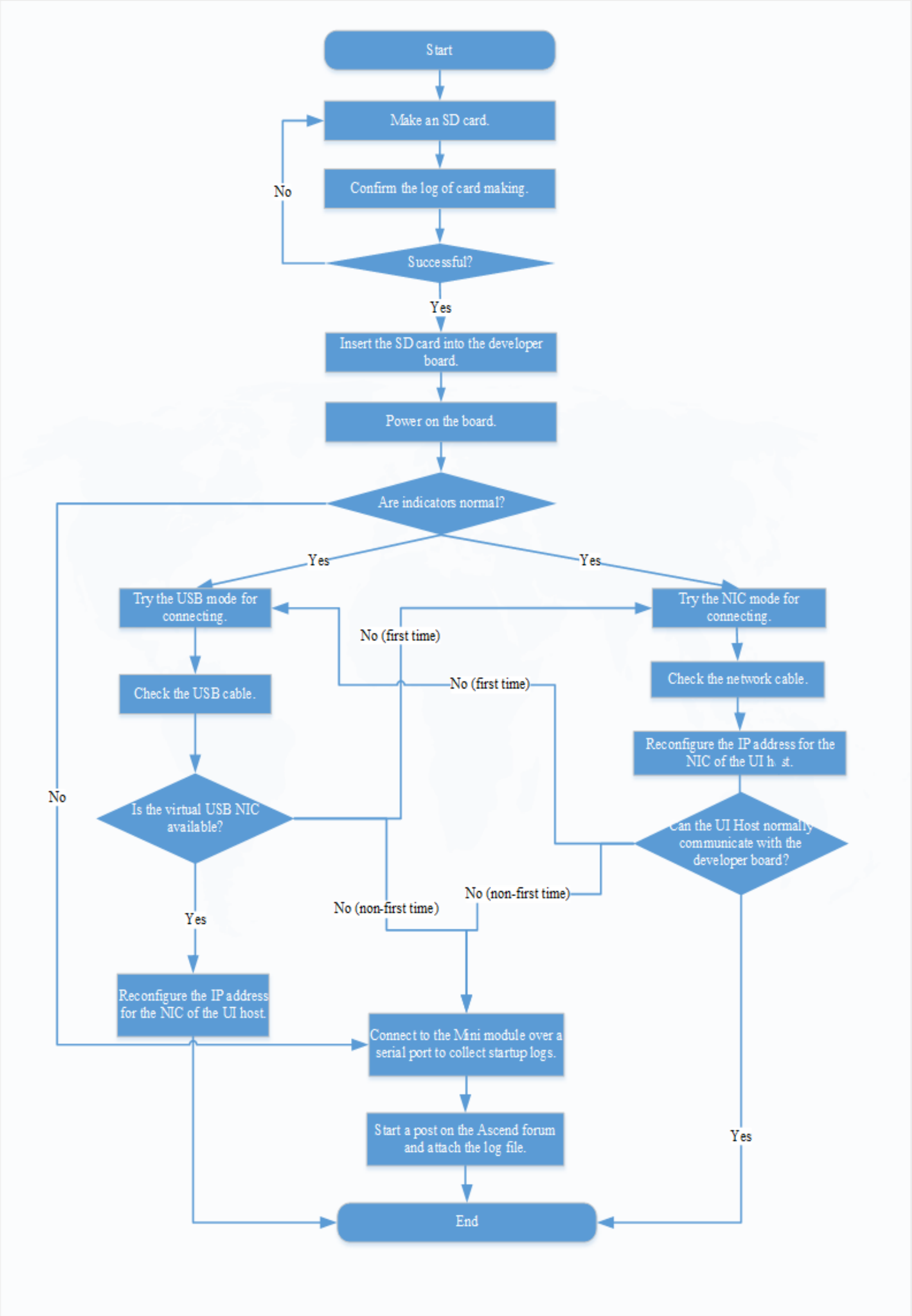
The symptoms are as follows:

- After a prepared SD card is inserted into the developer board and the developer board is powered on, the LED1 and LED2 indicators on the developer board are abnormal.
- After a prepared SD card is inserted into the developer board, the developer board is connected to the UI host in USB mode, and the developer board is powered on and started, no virtual NIC information is displayed on the UI host.
- After a prepared SD card is inserted into the developer board, the developer board is connected to the UI host in NIC mode, the developer board is powered on and started, and the NIC information of the UI host is configured, the UI host fails to communicate with the developer board.

Fault Locating

Perform troubleshooting by referring to **Figure 8-1**.

Figure 8-1 Troubleshooting process


Start
Make an SD card.
Confirm the log of card making.
No
Successful?
Yes
Insert the SD card into the developer board.
Power on the board.
Are indicators normal?
Yes
Yes
No
Try the USB mode for connecting.
Try the NIC mode for connecting.
No (first time)
No (first time)
Check the USB cable.
Check the network cable.
Reconfigure the IP address for the NIC of the UI host.
Is the virtual USB NIC available?
Can the UI Host normally communicate with the developer board?
Yes
No (non-first time)
No (non-first time)
Reconfigure the IP address for the NIC of the UI host.
Connect to the Mini module over a serial port to collect startup logs.
Start a post on the Ascend forum and attach the log file.
Yes
End

Solution

Step 1 Ensure that the SD card is made correctly and successfully.

In the **sd_card_making_log** file in the directory where the card creation scripts are located, check whether the card is made successfully. If the card fails to be created, try again.

Step 2 Insert the SD card into the Atlas 200 DK developer board and power on the board.

- If LED1 and LED2 on the Atlas 200 DK developer board are normal, that is, LDE1 and LDE2 are both on after the developer board is started, go to **Step 3**.
- If LED1 and LED2 on the Atlas 200 DK developer board are abnormal, that is, LED1 and LED2 are not on after the developer board is started for a long time (more than 15 minutes), go to **Step 4**.

Step 3 Connect the Atlas 200 DK developer board to the UI host.

- The UI host is connected to the developer board in USB mode, but the virtual USB NIC is not displayed on the UI host.

  Check the USB network cable and ensure that both ends of the USB network cable are properly connected.

  If the USB virtual NIC is still not displayed on the UI host, try using the NIC mode to connect the UI host to the developer board.
- The UI host is connected to the developer board in NIC mode. The UI host fails to communicate with the developer board after the IP address is configured for the NIC of the UI host.

  Check the network cable and ensure that the two connected ports of the network cable are normal, and the reconfigure the IP address of the NIC on the UI host.

  If the UI host still fails to communicate with the developer board, try using the USB mode to connect the UI host to the developer board.

If the UI host fails to connect to the developer board in either USB or NIC mode, go to **Step 4**.

Step 4 Connect the Atlas 200 AI accelerator module of the Atlas 200 DK developer board to the UI host over a serial cable by referring to **7.3 Connecting to the Atlas 200 DK Developer Board over a Serial Port**.

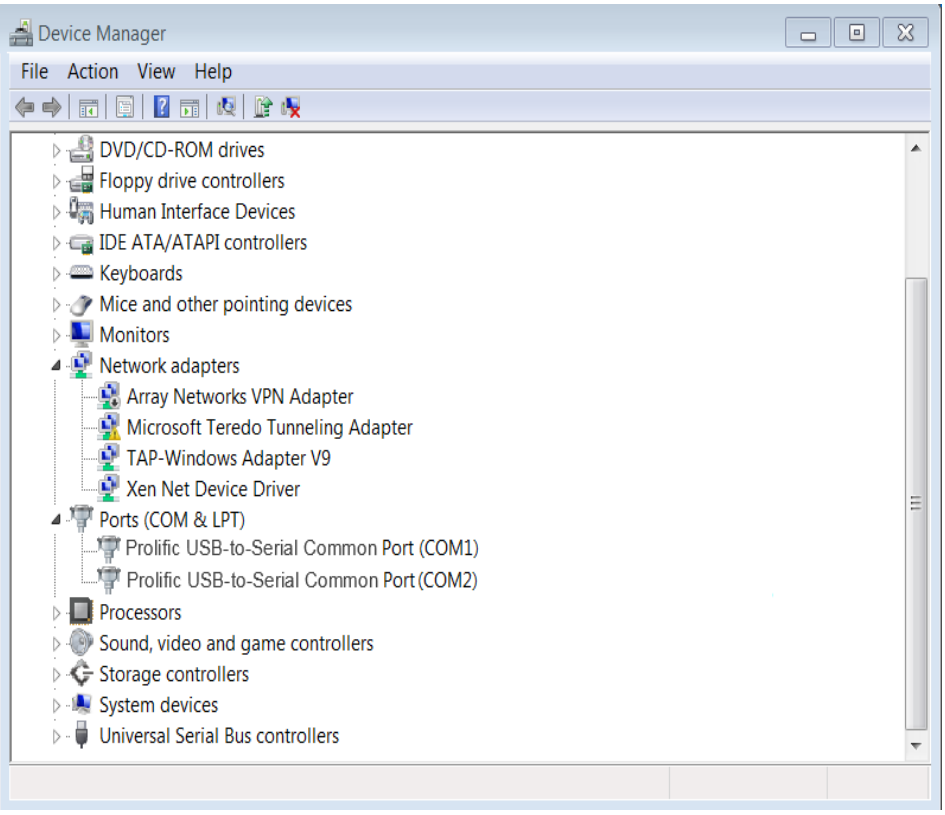
Step 5 Install the network debugging tool and USB-to-serial driver on the UI host.

- Recommended network debugging tool: IPOP
- USB-to-serial driver: PL2303 driver

Step 6 Start the network project debugging tool, for example, IPOP. The serial port window is displayed.

1. Click the **Terminal** tab page.
2. On the menu bar, click . The **Connect List** dialog box is displayed.
3. Configure the connection.
   - **ConnName**: indicates a user-defined connection name.
   - **Type**: indicates a port type. Choose **COMX**. You can view the available COM ports in the device manager of the computer. Remove and insert the serial cable on the UI host to determine the COM port used by the Atlas 200 DK, as shown in **Figure 8-2**.

Figure 8-2 Viewing COM ports


Device Manager
File
Action
View
Help
DVD/CD-ROM drives
Floppy drive controllers
Human Interface Devices
IDE ATA/ATAPI controllers
Keyboards
Mice and other pointing devices
Monitors
Network adapters
Array Networks VPN Adapter
Microsoft Teredo Tunneling Adapter
TAP-Windows Adapter V9
Xen Net Device Driver
Ports (COM & LPT)
Prolific USB-to-Serial Common Port (COM1)
Prolific USB-to-Serial Common Port (COM2)
Processors
Sound, video and game controllers
Storage controllers
System devices
Universal Serial Bus controllers

– Set the baud rate to **115200**.

4. Click **OK**.

Step 7 Power on the Atlas 200 DK developer board, and view the Atlas 200 DK startup information in the COM connection window of the IPOP tool.

There are many startup logs. Click 🔴 on the menu bar to save the startup logs to the installation directory of the IPOP tool. When this button changes to 🟢, a message is displayed at the bottom of the IPOP tool, indicating that the log file is saved. You can obtain the log file named after the current time from the installation directory of the IPOP tool according to the message.

Step 8 Start a help post on the **Ascend Forum** and upload the startup log file as an attachment. Huawei engineers will answer your question as soon as possible.

----End

A Change History

Release Date	Description
2020-05-30	This issue is the first official release.