Metis Compute Board User Guide

2025-12-10
AX-000936-UG
Version: Issue 2

Document and proprietary information

Document and information property

This document and the information contained herein are the property of Axelera AI. It must not be reproduced or otherwise disclosed without prior consent from Axelera AI.

Trademarks

The Axelera "AX" logo is a trademark of Axelera AI BV, registered in the Netherlands and other countries. “AXELERA” and “METIS” are registered as word marks across a number of countries/regions.

All other product and company names and registered trademarks may be property of their respective owners.

Document revision history

Revision	Date	Description
1 – PRELIMINARY	2025-09-30	Initial Issue.
2 – PRELIMINARY	2025-12-10	Updated Section *3.3 - Flashing a BSP image* Added: • *3.4 Metis DKMS driver update* • *3.5 SBC upgrade from v1.2.x to v1.3.1* • *3.6* Entering Maskrom Mode

TABLE OF CONTENTS

1 Introduction

The Metis Compute Board is designed around the Axelera AI Metis AIPU AI inference accelerator and features the Arm-based Rockchip RK3588 edge computing System-on-Chip. This User Guide provides instructions on how to use the product including the Board Support Package (BSP) and covers flashing the board with the latest BSP release, setting up the system and installing Axelera AI Voyager SDK to run your application.

For more information, please refer to the Metis Compute Board Datasheet.

Figure 1: Axelera AI Metis Compute Board

2 Hardware setup

This section covers connecting peripherals, network, and power cables to the Compute Board.

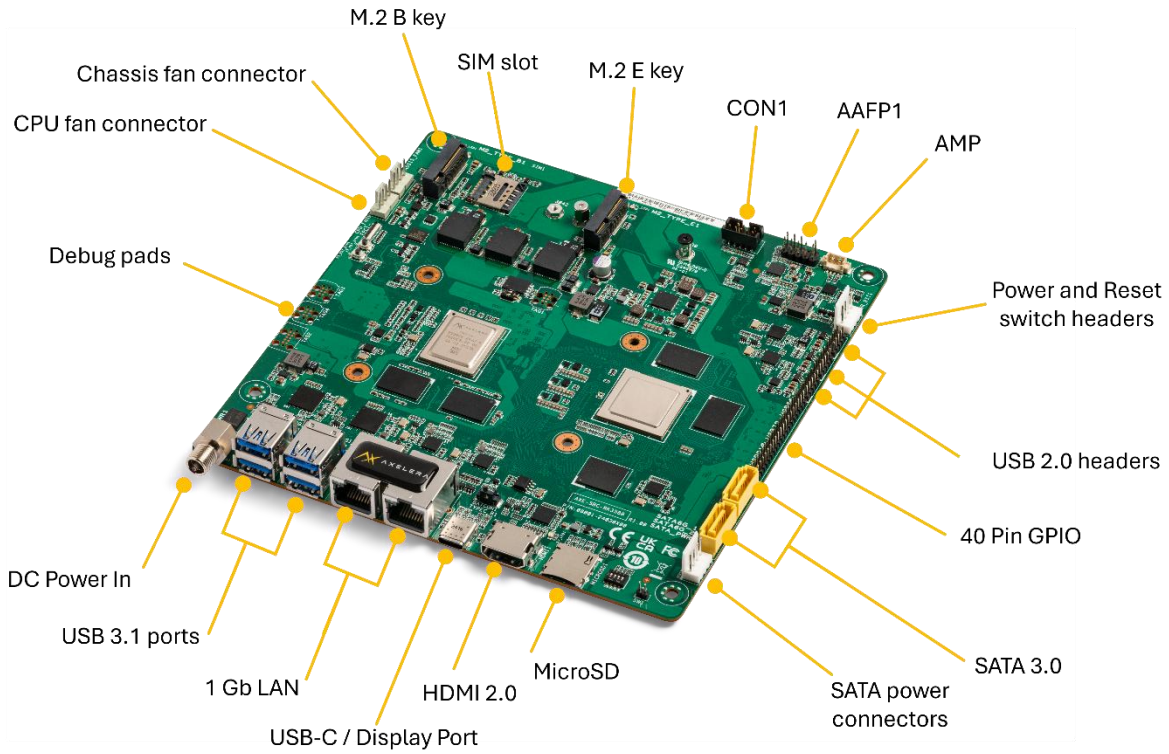

Figure 2: Compute Board - Ports, connectors, and headers

As shown in the figure above:

1. Connect the CPU and chassis fans to their respective fan headers on the board
2. (Optional) Connect the Power and Reset switches to the **Power and Reset switch headers**

Next, connect the peripherals and power:

1. Plug mouse and a keyboard to the **USB** ports
2. Connect a LAN cable to the **LAN** port and to your network equipment
3. Attach an HDMI cable to the **HDMI** port and your display device
4. Connect the DC power cable to the **DC Power In** on the board

   Use a 12V, 7A DC adapter with a barrel connector size of 5.5mm (outer) x 2.5mm (inner).

The compute board will automatically boot and perform a Power-On Self-Test (POST) once the power cable is connected, displaying the Axelera logo. A shell can be accessed by clicking the icon in the top-left corner.

3 Flashing with the latest BSP version

The Metis Compute Board comes pre-installed with the latest version of the BSP. If you need to reinstall the BSP, this chapter provides instructions on how to download and flash the latest version.

3.1 Pre-requisites

- A Linux host with:
  - An internet connection
  - A USB Type-C interface
  - An SD card reader
- A USB Type-C data cable

3.2 Preparation

Please execute the following steps.

NOTE: All commands in this section are executed on the Linux host and not on the Metis Compute Board.

3.2.1 Connect the Metis Compute Board to the host over USB

For board-to-host communication, you **must** use a USB-C data cable to connect the Metis Compute Board to the host.

3.2.2 Add the current user to the `plugdev` group

To interact with the USB port, you must be part of the **`plugdev`** group:

```
sudo usermod -aG plugdev $USER
```

INFO: You must log off and log back in for the change to become effective.

3.2.3 Add udev rules

Add udev rules to ensure you have the necessary permissions for board-to-host communication. This allows the host to correctly detect and interact with the device over USB.

```
sudo nano /etc/udev/rules.d/51-android.rules
```

Edit the file, adding the following line:

```
SUBSYSTEM=="usb", ATTR{idVendor}=="2207", ATTR{idProduct}=="0006", MODE="0666",
GROUP="plugdev"
```

Now apply the new rules by running:

```
sudo chmod a+r /etc/udev/rules.d/51-android.rules
sudo udevadm control --reload-rules
sudo udevadm trigger
```

3.2.4 *Install the Android Debug Bridge (`adb`)*

Run the following commands to install `**adb**` and start the server process.

```
sudo apt update
sudo apt install adb
adb kill-server
adb start-server
```

```
* daemon not running; starting now at tcp:5037
* daemon started successfully
```

INFO: It is important to run `sudo apt update` if this is a fresh install, otherwise the `adb` package will not be found.

3.2.5 *Download the upgrade_tool and move it to /usr/local/bin*

```
wget https://software.axelera.ai/artifactory/axelera-bsp/voyager/bsp/aisbc/upgrade_tool
chmod +x upgrade_tool
sudo
# <type password: AxeRoot2025>
mv upgrade_tool /usr/local/bin
```

3.2.6 *Verify that the Metis Compute Board is correctly enumerated as a USB device*

```
adb devices -l
```

Expected output:

```
List of devices attached
0123456789ABCDEF       device usb:2-5 transport_id:4
```

If the device is not listed, make sure that the host and the Metis Compute Board are connected correctly and the `**adb**` server is running.

TIP: Make a note of the device number, for example, **`usb:2-5`** as it will be required in a later step.

3.3 Flashing a BSP image

Follow these steps to flash the latest BSP image:.

3.3.1 Locate and Download the Latest BSP Image

1. **Browse to the BSP repository:**

https://software.axelera.ai/ui/native/axelera-bsp/voyager/bsp/aisbc/

You will see a directory structure similar to:

```
Index of axelera-bsp/voyager/bsp/aisbc/
Name        Last Modified           Size
----------------------------------------
../
1.1.0/      08-08-25 11:04:39
1.2.x/      29-09-25 15:54:05
1.3.x/      28-11-25 11:19:41   <-- Latest release
upgrade_tool 08-08-25 11:17:41  3.2 MB
```

TIP: Always select the highest version directory (for example, directory 1.3.x).

2. **Locate latest version:**

Inside the latest version folder, look for the update image file. It will have this pattern:

```
voyager-image-weston-antelao-3588-<version>.update.img
```

Example for version 1.3.1:

```
voyager-image-weston-antelao-3588-1.3.1.update.img
```

This is the file you need to flash. Other files (e.g., .mender, .deb) are for different purposes.

3. **Download the update image using wget:**

Example:

```
wget https://software.axelera.ai/artifactory/axelera-bsp/voyager/bsp/aisbc/1.3.x/voyager-image-weston-antelao-3588-1.3.1.update.img
```

NOTE: Ensure the firmware image and the previously downloaded update tool are in the same directory.

3.3.2 Connect to the Metis Compute Board

Use the USB endpoint output from the `**adb devices -l**` from earlier to connect to the Metis Compute Board using a shell:

```
adb -s usb:2-5 shell
```

Now that the ADB bridge is active, run the following command to reboot the board into flashing mode:

```
reboot loader
```

3.3.3 Flash the new BSP image to the Metis Compute Board

Run the upgrade tool downloaded previously.

```
upgrade_tool LD
```

Expected output:

```
DevNo=1 Vid=0x2207,Pid=0x350b,LocationID=25     Mode=Loader
SerialNo=3fa2db1451c2fe08
```

TIP: If you do not see the expected output, the board may not have fully entered loader mode or the USB connection may not have initialized correctly.

Try running **`upgrade_tool LD`** command again.

If the device still does not appear after multiple attempts, check the USB connection, ensure the board is in loader mode, and verify that **`adb`** is running correctly.
Once the device is detected you can proceed with the upgrade.

Start the update:

```
sudo upgrade_tool uf voyager-image-weston-antelao-3588-
1.3.1.update.img
```

Expected output:

```
Loading firmware...
Support Type:3588       FW Ver:1.0.00   FW Time:2025-09-12 16:29:46
Loader ver:1.0b Loader Time:2025-09-11 19:21:58
Start to upgrade firmware...
Test Device Start
Test Device Success
Check Chip Start
Check Chip Success
Get FlashInfo Start
Get FlashInfo Success
Prepare IDB Start
Prepare IDB Success
Download IDB Start
Download IDB Success
Download Firmware Start
Download Image... (100%)
Download Firmware Success
Upgrade firmware ok.
```

3.3.4 Reconnect to the host after the update

After the update has completed, use the following to reconnect to the board:

```
adb -s usb:2-5 shell
```

If the connection fails, the board may still be booting. You can monitor its status by running:

```
adb devices -l
```

Once the board appears in the device list, you can issue the shell command **`adb -s usb:2-5 shell`** to establish the connection.

3.4 Metis DKMS driver update

NOTE:

- Commands in this section are to be executed on the SBC host.

- The Metis driver needs to be installed on the SBC and not inside docker container.

Locate and Download the Latest Kernel Driver deb file

1. **Browse to the BSP repository:**

https://software.axelera.ai/ui/native/axelera-bsp/voyager/bsp/aisbc/

You will see a directory structure similar to:

```
Index of axelera-bsp/voyager/bsp/aisbc/
Name        Last Modified          Size
------------------------------------------
../
1.1.0/      08-08-25 11:04:39
1.2.x/      29-09-25 15:54:05
1.3.x/      28-11-25 11:19:41   <-- Latest release
upgrade_tool 08-08-25 11:17:41  3.2 MB
```

TIP: Always select the highest version directory (for example, directory 1.3.x).

2. Locate latest version:

Inside the latest version folder, look for the Kernel driver deb file. It will have this pattern:

```
kernel-module-metis-6.1.148-rockchip-standard_<version>_arm64.deb
```

Example for version 1.5.1-r0:

```
kernel-module-metis-6.1.148-rockchip-standard_1.5.1-r0_arm64.deb
```

3. Download the package:

```
wget https://software.axelera.ai/artifactory/axelera-
bsp/voyager/bsp/aisbc/1.3.x/kernel-module-metis-6.1.148-rockchip-
standard_1.5.1-r0_arm64.deb
```

4. Get current driver version

```
cat /sys/class/metis/version
```

5. *Become root - password: AxeRoot2025*

```
su
```

6. *Mount filesystem as read write*

```
mount -o remount,rw /
```

7. *Copy new Metis deb file and install it*

```
dpkg -i <package_name.deb>
```

8. *Make the filesystem read-only again*

```
mount -o remount,r /
```

9. *Power cycle the SBC*

10. *Check the new driver version*

```
cat /sys/class/metis/version
```

3.5 SBC upgrade from v1.2.x to v1.3.1

IMPORTANT:

This is a single-step, in-place update from v1.2.x to v1.3.1 only.

Do not attempt this procedure on any other version.

The update process is designed to preserve existing data and should not affect your current configuration. upgrade_tool should already be downloaded, if not see **3.2.5 Download the upgrade_tool and move it to /usr/local/bin**

NOTE: To perform this update, the board must be connected to the host machine via its USB-C port.

1. Download the v1.2.x->v1.3.1 upgrade image

```
wget https://software.axelera.ai/artifactory/axelera-bsp/voyager/bsp/aisbc/1.3.x/OTA-patch-voyager-image-weston-antelao-3588-1.3.1.update.img
```

2. From a shell on the board (e.g., via adb shell, SSH, or console), set the board to Maskrom mode. See ***3.6* Entering Maskrom Mode** for more information.

```
reboot bootloader
```

3. From the host machine, verify the board is correctly connected and in Maskrom mode:

```
upgrade_tool LD
```

Expected output:

```
List of rockusb connected(1)
DevNo=1 Vid=0x2207,Pid=0x350b,LocationID=8132 Mode=Loader
SerialNo=298c5d3d852bd326
```

4. Upgrade, once verified

```
sudo upgrade_tool UF OTA-patch-voyager-image-weston-antelao-3588-1.3.1.update.img
```

INFO: This process flashes the new version with the updated partition scheme and automatically reboots the board.

It does so without overwriting data in the `/data` partition or writable sections of the root filesystem, such as `/home/antelao`.

3.6 Entering Maskrom Mode

Maskrom mode is a low-level recovery state that allows direct communication with the board's bootloader over USB. This mode is typically used for flashing firmware or recovering a device when the normal boot process fails.

Pre-requisites

- A Linux host with:
  - ADB installed
  - A USB Type-C interface
- A USB Type-C data cable

1. Connected the SBC to the host machine using a USB-C cable.
2. Locate Metis Compute Board as a USB device

```
adb devices -l
```

Expected output:

```
List of devices attached
0123456789ABCDEF       device usb:2-5 transport_id:4
```

If the device is not listed, make sure that the host and the Metis Compute Board are connected correctly and the `**adb**` server is running.

> **TIP:** Make a note of the device number, for example, **`usb:2-5`** as it will be used by the adb command to connect to the remote system.

3. Start the connection:

```
adb -s usb:2-5 shell
```

4. Now that the ADB bridge is active, run the following command to reboot the board into Maskrom mode:

```
reboot bootloader
```

4 System Setup

4.1 Generic System Configuration

4.1.1 Prerequisites

Metis Compute Board needs to be connected to the internet via Ethernet.

4.1.2 Log in to the system

You can log in via SSH or ADB, but ADB is recommended, especially when configuring a static IP or renaming the connection because you'll need to bring the link down to make changes.

SSH login

Log in using as user '**`antelao`**'. The default password is '**`AxeAntelao2025`**'. The system also has a root user configured (username '**`root`**', password '**`AxeRoot2025`**').

ADB login

Use **`adb devices -l`**` to determine where the board is connected, for example, usb:2-5, and then connect with:

```
adb -s usb:2-5 shell
```

4.1.3 Change the default password

For security reasons, to is recommended to immediately change the default password on first login using the '**`passwd`**' command.

To change the password for the currently logged in user, run:

```
passwd
```

Then follow the prompts:

```
Changing password for antelao.
Current password:
New password:
Retype new password:
passwd: password updated successfully
```

4.1.4 *Set a static IP address [optional]*

Use the `**nmcli**` utility to set a static IP address for the Metis Compute Board.

> **TIP:** It is recommended to use the `adb` connection to perform this step, otherwise you will lose connectivity on a wired connection.

Check active connections:

```
nmcli dev status
```

```
DEVICE  TYPE      STATE        CONNECTION
eth1    ethernet  connected    Wired connection 2
eth0    ethernet  unavailable  --
dummy0  dummy     unmanaged    --
lo      loopback  unmanaged    --
```

In this example, `**Wired connection 2**` is active on `**eth1**` with the default configuration.

1. Disconnect the network interface **eth1** and edit the connection:

```
nmcli dev disconnect eth1
nmcli con edit "Wired connection 2"
```

An interactive prompt will open where you can edit all properties of the connection.

2. To view all of the current settings, enter:

```
nmcli> print
```

```
===============================================================
            Connection profile details (Wired connection 2)
===============================================================
connection.id:                          Wired connection 2
connection...
---------------------------------------------------------------
802-3-ethernet.port:                    --
802-3-ethernet...
---------------------------------------------------------------
ipv4.method:                            auto
ipv4...
---------------------------------------------------------------
ipv6.method:                            auto
ipv6...
```

```
------------------------------------------------------------
proxy.method:                           none
proxy...
------------------------------------------------------------
```

3. Rename the default name (e.g. **"Wired connection 2"**) to something simpler, such as **eth1**.

```
nmcli> set "Wired connection 2" eth1
```

4. Set a static IP address, for example to `**10.42.0.77**`:

```
nmcli> set ipv4.address 10.42.0.77/24
```

```
Do you also want to set 'ipv4.method' to 'manual'? [yes]: yes
```

```
nmcli> set ipv4.gateway 10.42.0.1
```

5. Verify the connection, save the configuration and reconnect the device:

```
nmcli> verify
```

```
Verify connection: OK
```

```
nmcli> save
```

```
Connection 'eth1' (282725cf-b2d1-3a4d-ae66-7240c06abd1f) successfully updated.
```

```
nmcli> quit
```

6. Reconnect to the interface

```
nmcli dev connect eth1
```

7. Verify if the process was successful:

```
ip addr show eth1
```

```
4: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq qlen 1000
    link/ether 42:20:9a:5d:67:98 brd ff:ff:ff:ff:ff:ff
    inet 10.42.0.77/24 brd 10.42.0.255 scope global noprefixroute eth1
       valid_lft forever preferred_lft forever
    inet6 fe80::bbf5:174d:c2c6:dc2e/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

4.1.5 *Setting up WireGuard VPN [optional]*

The configuration in this section depends entirely on how WireGuard is hosted, so no specific details are provided here. It is assumed that your configuration file is located at:

```
/etc/wireguard/wg0.conf
```

INFO: Replace this path in all following commands with the one for your setup.

1. Using the `**nmcli**` utility, the first step is to import the new connection from the configuration file:

```
nmcli connection import type wireguard file /etc/wireguard/wg0.conf
```

2. As in the previous example for setting a static IP address, you can edit the connection as needed using:

```
nmcli connection edit wg0
```

3. To activate the connection:

```
nmcli connection up wg0
```

4. Verify it is working, check the device status from `**nmcli**`:

```
nmcli device status
```

```
DEVICE  TYPE       STATE                  CONNECTION
[...]
wg0     wireguard  connected              wg0
[...]
```

5. Verify that the `**wg0**` interface has the correct IP address assigned, for example:

```
ip addr show wg0
```

```
23: wg0: <POINTOPOINT,NOARP,UP,LOWER_UP> mtu 1420 qdisc noqueue qlen 1000
    link/[65534]
    inet 10.8.0.2/24 brd 10.8.0.255 scope global noprefixroute wg0
       valid_lft forever preferred_lft forever
```

6. Ping, to verify the connection and the DNS resolution are working correctly:

```
ping -I wg0 axelera.ai
```

```
PING axelera.ai (199.60.103.26): 56 data bytes
64 bytes from 199.60.103.26: seq=0 ttl=51 time=83.498 ms
64 bytes from 199.60.103.26: seq=1 ttl=51 time=93.698 ms
64 bytes from 199.60.103.26: seq=2 ttl=51 time=48.087 ms
64 bytes from 199.60.103.26: seq=3 ttl=51 time=44.390 ms
^C
--- axelera.ai ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 44.390/67.418/93.698 ms
```

7. Close the VPN connection:

```
nmcli connection down wg0
```

4.1.6 Changing SSH server configuration [optional]

The SSH server running on the distribution is based on OpenSSH. Specifically, the **`sshd systemd`** socket is responsible for the port listening.

Edit the configuration to change the port from the default `**22**` to, for example, `**2222**`:

```
systemctl edit sshd.socket
```

```
### Editing /etc/systemd/system/sshd.socket.d/override.conf
### Anything between here and the comment below will become the new contents of
[Socket]

ListenStream=2222

### Lines below this comment will be discarded

### /lib/systemd/system/sshd.socket
# [Unit]
# Conflicts=sshd.service
# Wants=sshdgenkeys.service
#
# [Socket]
# ExecStartPre=/bin/mkdir -p /var/run/sshd
# ListenStream=22
```

```
# Accept=yes
#
# [Install]
# WantedBy=sockets.target
```

For the rest of the configuration, which is OpenSSH-specific, the default configuration file for it can be found at `/etc/ssh/sshd_config`.

To change these configurations, it is sufficient to edit the above file.

For example, to disable password logins for root, open `/etc/ssh/sshd_config` with a text editor and change from the default:

```
PermitRootLogin yes
```

to:

```
PermitRootLogin no
```

4.1.7 Automatically mounting external disk [optional]

To mount an external disk is automatically on boot, and therefore to run all the commands shown in this example, it is **necessary** to have root access to the machine.

The first step is to connect the disk itself, which in this specific example is connected through one of the available SATA ports.

The first thing to do is to identify the UUID and the type of the disk itself, by running:

```
lsblk -f
```

```
NAME        FSTYPE FSVER LABEL UUID                                 FSAVAIL
FSUSE% MOUNTPOINTS
sda
`-sda1      ext4         work  <your_uuid>
```

Take note of the `UUID` and the `FSTYPE` of the volume you would like to mount. In this example, `/dev/sda1` of type `ext4` will be mounted to `/media/external`.

Next, create the directory which will be used as a mounting point:

```
mkdir -p /media/external
```

Finally, open `/etc/fstab` with your preferred text editor, and add the following line at the end of the file:

```
UUID=<your_uuid> /media/external ext4 defaults,nofail 0 2
```

To check that the procedure was successful, it is possible to mount all drives listed in `/etc/fstab` without rebooting by running:

```
mount -a
```

And checking the disk is now mounted correctly by running **`lsblk`** again, which now shows `/media/external` as mountpoint:

```
lsblk -f
```

```
NAME        FSTYPE FSVER LABEL UUID                                 FSAVAIL
FSUSE% MOUNTPOINTS
sda
`-sda1      ext4         work  <your_uuid>                          416.6G
4% /media/external
```

After rebooting the machine, the disk should be automatically mounted if everything was done correctly.

5 Setting up the Voyager SDK

The Voyager SDK enables building high-performance inferencing applications on Axelera AI Metis devices. On the Metis Compute Board, the Voyager SDK runs in a Docker container. Although you can manually create a container using the Voyager SDK installer, we recommend using the pre-built container included in the SDK. Instructions for using the pre-built container are detailed below.

5.1.1 Prerequisites

- An active internet connection is required for the download phase – connect the Metis Compute Board to the internet via Ethernet.

5.1.2 Download the Voyager SDK Docker container

Run the following commands to download your desired version of the Voyager SDK container.

```
cd /home/antelao
./setup_axelera_environment.sh <SDK-version>
```

For example, to download version 1.4.0 run:

```
./setup_axelera_environment.sh 1.4.0
```

INFO: This step may take several minutes due to the download of approximately 6GB of data.

5.1.3 Starting the container

To use the container, run **`python3 start_axelera.py start --container-name <container-name> --version <SDK-version>`** located in the home directory, for example, to run container Voyager-SDK, version 1.4.0:

```
python3 start_axelera.py start --container-name "Voyager-SDK" --
version  "1.4.0"
```

After the container starts and completes its initial setup, a shell session will open inside the Docker container.

Monitor the console output for status messages and potential errors during execution. The expected output should resemble the following:

```
[INFO] Initializing Voyager SDK container 'Voyager-SDK' (Version 1.4.0)
[INFO] Console/terminal access detected - container ready for display hot-plug when needed for GUI debugging.


============================================================
[CRITICAL] CRITICAL SETUP REQUIRED AFTER CONTAINER STARTS
============================================================


REQUIRED COMMAND:
make clobber-libs && make operators


IMPORTANT NOTES:
   * This command must be executed once after container startup
   * Internet connection is required for package downloads
   * Command execution may take several minutes to complete
   * Do not interrupt the process once started
```

INFO: If a display is connected when the Docker container is created, the inference output will appear in a window.

If the container has just been created, run the following command from within the container shell as part of the initial setup:

```
make clobber-libs && make operators
```

This step is only required during the container's first-time setup and is not needed for subsequent container starts once setup has been successfully completed.

For detailed guidance on using the Voyager SDK, refer to the Voyager SDK documentation available at https://github.com/axelera-ai-hub/voyager-sdk.

5.1.4 Useful information

The directory `/home/antelao/shared` is shared between the container and the bare-metal OS of the container to allow simple file sharing and data persistence. Any files created in the container outside of this directory will **not** persist across shutdown/restart of the Docker container.

5.2 Support

For further information and support please visit:

- **Axelera AI Community**: https://community.axelera.ai/
- **Axelera AI Customer Portal**: https://support.axelera.ai

6 Legal Notice

This document is provided for information purposes only and shall not be regarded as a warranty of a certain functionality, condition, or quality of a product. Axelera AI BV ("Axelera") makes no representations or warranties, expressed or implied, as to the accuracy or completeness of the information contained in this document and assumes no responsibility for any errors contained herein. Axelera shall have no liability for the consequences or use of such information or for any infringement of patents or other rights of third parties that may result from its use. This document is not a commitment to develop, release, or deliver any Material (defined below), code, or functionality.

Axelera reserves the right to make corrections, modifications, enhancements, improvements, and any other changes to this document, at any time without notice.

Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete.

AXELERA products are sold subject to the Axelera standard terms and conditions of sale supplied at the time of order acknowledgement, unless otherwise agreed in an individual sales agreement signed by authorized representatives of Axelera and the Customer ("Terms of Sale"). Axelera hereby expressly objects to applying any customer general terms and conditions with regards to the purchase of the Axelera product referenced in this document. No contractual obligations are formed either directly or indirectly by this document.

Axelera products are not designed, authorized, or warranted to be suitable for use in medical, military, aircraft, space, or life support equipment, nor in applications where failure or malfunction of the Axelera product can reasonably be expected to result in personal injury, death, or property or environmental damage. Axelera accepts no liability for inclusion and/or use of Axelera products in such equipment or applications and therefore such inclusion and/or use is at customer's own risk.

Axelera makes no representation or warranty that products based on this document will be suitable for any specified use. Testing of all parameters of each product is not necessarily performed by Axelera. It is customer's sole responsibility to evaluate and determine the applicability of any information contained in this document, ensure the product is suitable and fit for the application planned by customer, and perform the necessary testing for the application in order to avoid a default of the application or the product. Weaknesses in customer's product designs may affect the quality and reliability of the Axelera product and may result in additional or different conditions and/or requirements beyond those contained in this document. Axelera accepts no liability related to any default, damage, costs, or problem which may be based on or attributable to: (i) the use of the Axelera product in any manner that is contrary to this document or (ii) customer product designs.

No license, either expressed or implied, is granted under any Axelera patent right, copyright, or other Axelera intellectual property right under this document. Information published by Axelera regarding third-party products or services does not constitute a license from Axelera to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the

patents or other intellectual property rights of the third party, or a license from Axelera under the patents or other intellectual property rights of Axelera.

Reproduction of information in this document is permissible only if approved in advance by Axelera in writing, reproduced without alteration and in full compliance with all applicable export laws and regulations, and accompanied by all associated conditions, limitations, and notices.

THIS DOCUMENT AND ALL AXELERA DESIGN SPECIFICATIONS, REFERENCE BOARDS, FILES, DRAWINGS, DIAGNOSTICS, LISTS, AND OTHER DOCUMENTS (TOGETHER AND SEPARATELY, "MATERIALS") ARE BEING PROVIDED "AS IS." AXELERA MAKES NO WARRANTIES, EXPRESSED, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO THE MATERIALS, AND EXPRESSLY DISCLAIMS ALL IMPLIED WARRANTIES OF NONINFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE. TO THE EXTENT NOT PROHIBITED BY LAW, IN NO EVENT WILL AXELERA BE LIABLE FOR ANY DAMAGES, INCLUDING WITHOUT LIMITATION ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, PUNITIVE, OR CONSEQUENTIAL DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF ANY USE OF THIS DOCUMENT, EVEN IF Axelera HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. Notwithstanding any damages that customer might incur for any reason whatsoever, Axelera's aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms of Sale for the product.

Trademarks

The Axelera "AX" logo is a trademark of Axelera AI BV, registered in the Netherlands and other countries. "AXELERA" and "METIS" are registered as word marks across a number of countries/regions. Other company and product names may be trademarks of the respective companies with which they are associated.