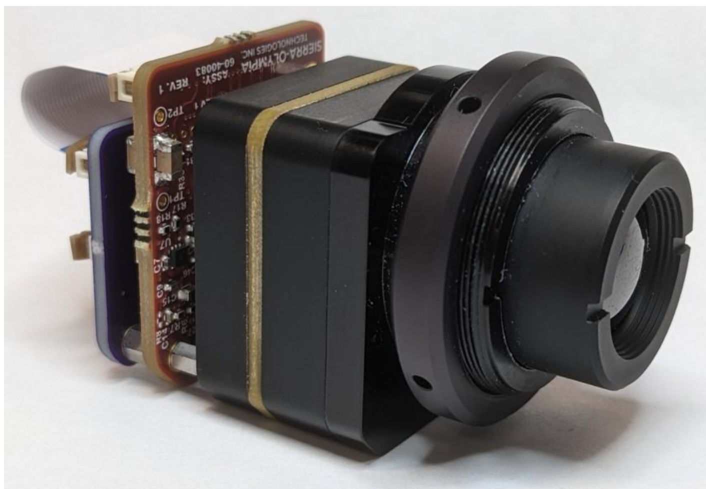
SIERRA-OLYMPIA TECHNOLOGIES INC.



USER GUIDE VIENTO MIPI

603 Portway Avenue, Suite 100 | Hood River, OR 97031 | 855.222.1801 | sierraolympia.com

Product Export Classification Control Number (ECCN): 60/30Hz: 6A003.b.4.b, 9Hz: 6A993. This document does not contain export-controlled technology.

Table of Contents

Table of Figures

1 SAFETY CONDITIONS

Read all instructions prior to use.

Observe ESD (electrostatic discharge) precautions when handling.

The camera requires reasonable thermal sinking when operating. Use stirred air and conduction to outside environment when installed in an enclosure.

The camera must be operated within the environmental limits.

Repairs and service are to be completed only by Sierra Olympic Technologies. Please refer any issues to your sales representative.

At the end of its service life, dispose of the imager in accordance with local disposal regulations for electronic devices. Alternatively, you can return the imager back to Sierra Olympia Technologies, Inc.

2 EXPORT NOTICE

Product Export Classification Control Number (ECCN): 60/30Hz: 6A003.b.4.b, 9Hz: 6A993.

This document does not contain export-controlled technology

3 REFERENCE DOCUMENTS

Document Number	Document Title
20-70062	Mechanical ICD, Viento 10 MIPI
20-70061	Mechanical ICD, Viento HD10 MIPI
20-70060	Electrical ICD, Viento Family MIPI
10-80023	Viento Family MIPI Firmware API

4 INTRODUCTION

MIPI CSI-2 is an industry standard electrical interface commonly used in embedded devices to transport video data for cameras. MIPI is NOT universally "plug and play", and drivers are required for each camera/operating system/hardware platform combination. These drivers must be developed by customers for their specific application.

To assist customers in developing drivers for their application, SOTI offers:

1. A working example composed of a NVIDIA JETSON ORIN NANO developer kit connected to a Viento 10 (or HD10) with MIPI CSI-2 backend.
2. Functional sample driver source code for Jetson Linux.

The sample driver is a minimum viable product with the following technical limitations:

1. Changing video resolution at runtime is not supported
2. Image adjustment features are not supported

The Viento camera incorporates a DRS Tenum 640/1280 and a MIPI CSI-2 board stack developed by Sierra Olympia Technologies, Inc. (SOTI). The Tenum is an LWIR microbolometer camera core designed and manufactured by DRS Leonardo and distributed by SOTI, who also designs and integrates different interface electronics for various applications.

This document details the use of a MIPI CSI-2 interface option, available as an OEM board-only package or a developer kit. While many different video processing platforms are used to ingest and process video over MIPI, the SOTI developer's kit specifically targets the NVIDIA Jetson Orin Nano developer kit with the reference driver supporting a single CSI-2 input.

The developer kit offers customers a quick, functional reference design for integrating a Viento camera into their own hardware and software environments. Although the kit may not exactly match a customer's final application, it provides a versatile base for development, with customization options in both hardware and software to fit specific needs. Customers are empowered to handle modifications independently or work with SOTI's support team for tailored adjustments as required.

5 INCLUDED ITEMS

Viento HD10/10 MIPI Cameras contain the following items:

- Camera with interface board(s)
- USB Delivery drive/file share
    - User Guide (this document)
    - Interface Control Documents
    - Driver source code

Viento 10/HD10 MIPI Developer's Kits contain the following items:

- Camera with interface board(s)
- NVIDIA Jetson Orin Nano Developer Kit (MPN 945-137766-0000-000)
- SD Card
- Power Cable
- FFC cables
- USB Delivery drive
    - User Guide (this document)
    - Interface Control Documents

6 REQUIRED EQUIPMENT

To follow the instructions in this user guide, the following equipment is required.

Camera-only (OEM) orders:

- Device with MIPI connection
- MIPI drivers
- Power cable
- FFC cable – either 28 pin, 22 pin or 15 pin depending on which interface is being used, please see the Electrical ICD, doc number 20-70060, for more details

Developer Kit Orders:

- Monitor
- Keyboard
- Mouse

7 DEVELOPER KIT QUICK START

The Developer Kit is delivered with pre-configured software to demonstrate a working system immediately after setup. This section describes the setup process along with two methods for capturing video from the MIPI camera.

7.1 Hardware Connections

The Viento 10 MIPI Developer Kit is delivered with the camera attached to an NVIDIA Jetson Orin Nano developer kit.

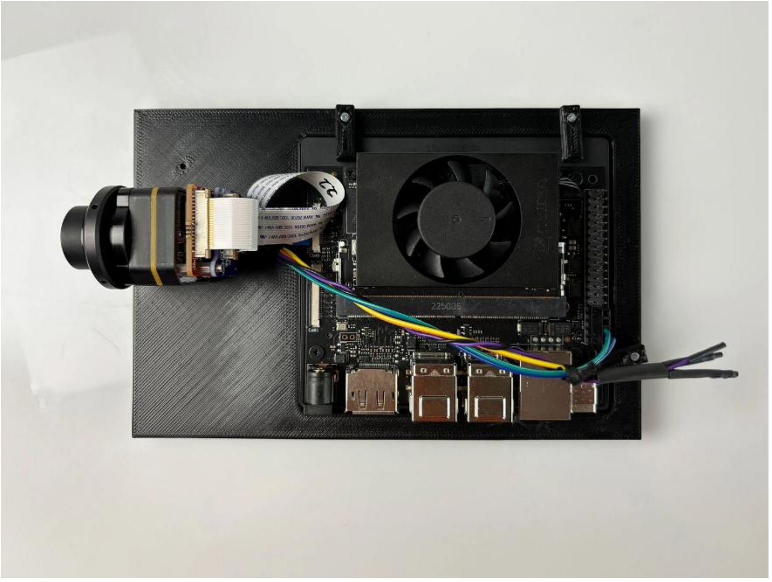


Figure 1: Viento 10 MIPI developer kit with camera connected to Jetson Orin Nano.

Connect monitor, keyboard, and mouse to the Orin Nano.

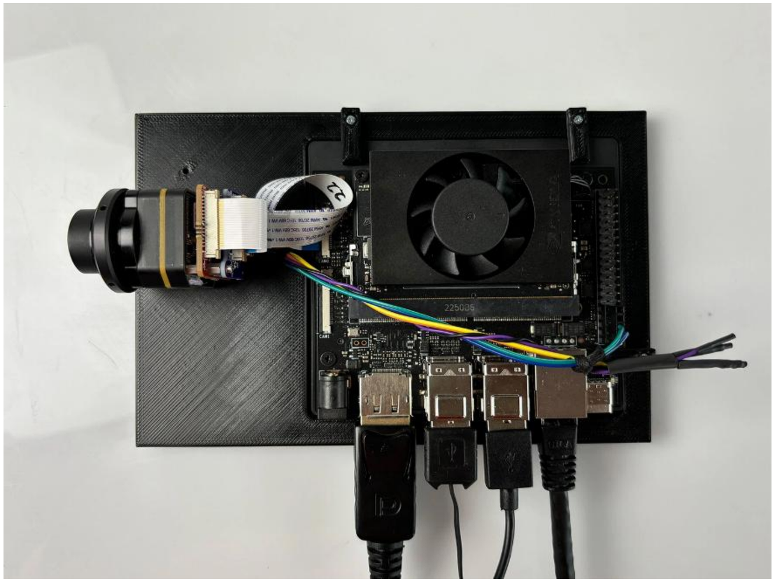


Figure 2: Connecting user peripherals to Jetson Orin Nano.

The microSD card included with the Orin Nano has been loaded with the NVIDIA base image and then updated with:

- OS updates
- MIPI drivers described in section 8
- Video and image application utilized in this document
- Convenient desktop shortcuts for demos described in this document

7.2 Software Usage

Connect the power supply to start up and automatically log into the Orin Nano.

> **Note, camera is powered from 5V power on the 40-pin header which is energized whenever the supply is connected. Disconnect the power cord when the dev kit is not in use to remove camera power.**

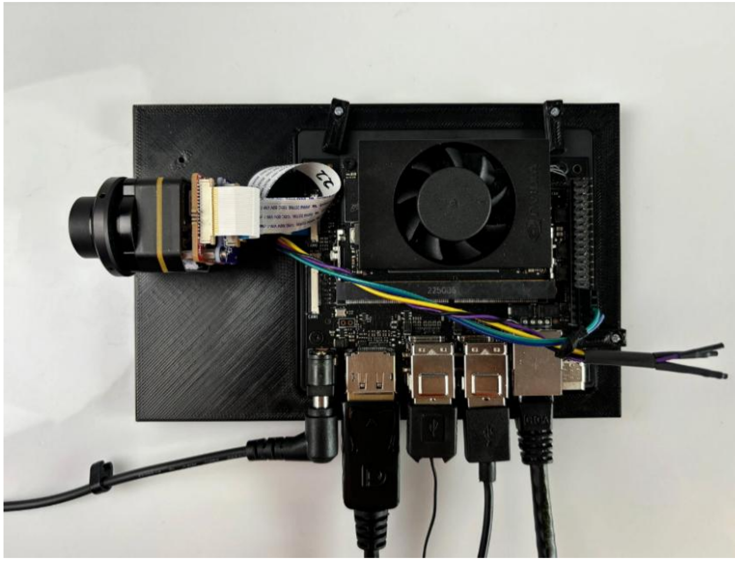


Figure 3: Power connected to the dev kit.

The default user credentials are:

- Username: soti
- Password: password

After startup the desktop will look similar to this:

Figure 4: User desktop on powerup.

7.2.1 Using This Guide

Throughout this document commands that should be entered into a terminal on the Orin Nano will be set in a monospace font and preceded by a dollar sign character that indicates the command prompt. When entering these commands manually, the dollar sign should be excluded. For example, to list the contents of the soti_demo_scripts folder when positioned in the home folder, enter the command:

```
$ ls soti_demo_scripts/
```

When entered properly, the terminal window should look like this:

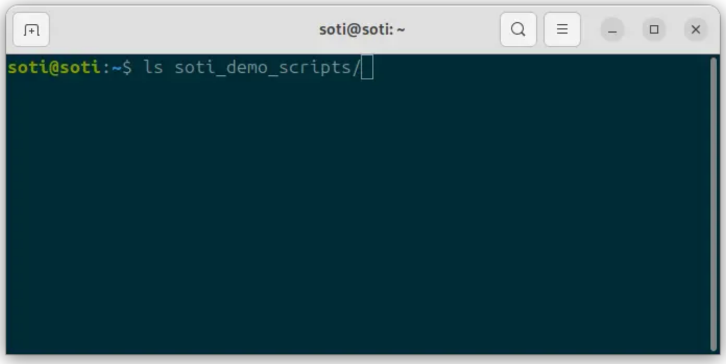


Figure 5: Entering a command in a terminal.

The command is then executed by pressing the "enter" key.

The `soti_demo_scripts` folder contains several sample scripts that demonstrate various methods of using the MIPI camera. Many of these are discussed later in this document. The scripts will use the camera connected to CAM0 by default, and will automatically adjust to the correct frame resolution. If multiple cameras are connected CAM1 can be targeted by passing command line arguments to the script. Review the explanatory text in each script file for complete usage instructions.

7.2.2 Display 16-bit Video with Python and OpenCV

Python can use OpenCV to display 16-bit video that is adjusted into the viewable range with histogram normalization. The version of OpenCV for Python included in the Ubuntu 20.04 distribution used on the Orin Nano does not handle 16-bit video correctly. A more recent build with proper 16-bit support has been installed on the dev kit using Pip. The /dev/video0 device can be seen by executing the OpenCV grabber script:

```
$ ~/soti_demo_scripts/y16Grabber.py
```

or by clicking the Y16 Video Grabber icon in the favorites bar on the left of the desktop.

To view the /dev/video1 device, run the following:

```
$ ~/soti_demo_scripts/y16Grabber.py --device_index 1
```

> **Note: When closing the video window, a single 16-bit frame will be captured to file *y16Grabber.py-Frame0.png* in the home folder (or pwd if launched from terminal).**

Figure 6: Python/OpenCV 16-bit Video.

The internal test pattern generator may also be enabled to test if the MIPI byte swap is set to match the devkit's expected endian-ness.

1. Set "i2cbus" variable to one of the following in a terminal:
   CAM0: `$ i2cbus=9`
   CAM1: `$ i2cbus=10`
2. Enable test pattern. 30Hz for 640 cameras, 32.3Hz for 1280 cameras:
   `$ i2ctransfer -f -y $i2cbus w2@0x33 0x0B 0xEE`
   For 640 camera, test pattern may set to 60Hz with the following command:
   `$ i2ctransfer -f -y $i2cbus w2@0x33 0x0B 0xAE`
   To restore the camera to normal operation, enter this command:
   `$ i2ctransfer -f -y $i2cbus w2@0x33 0x0B 0x2E`

   NOTE: See the Viento Family MIPI Firmware API, doc number 10-80023, for a full description of this and other available commands.

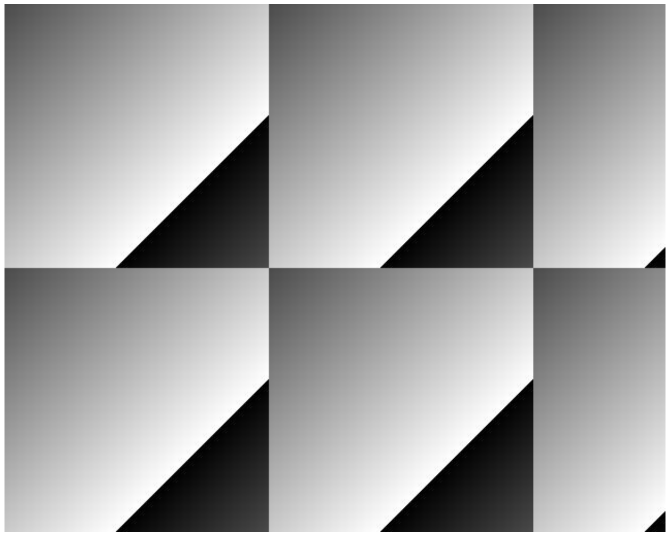


Figure 7: 16-bit Test Pattern generator with correct byte swap.

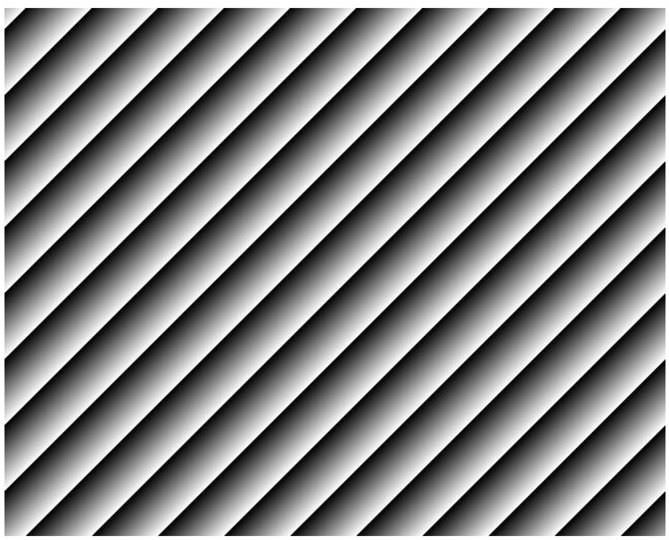


Figure 8: 16-bit Test Pattern generator with incorrect byte swap.

7.2.3 Capture 16-bit Image with v4l2

To capture and display a single 16-bit image from a video stream using v4l2 controls, start a terminal and run the script `capRaw.sh`:

```
$ soti_demo_scripts/capRaw.sh
```

The captured image is stored in the same directory used to run the capture command. To view the image, launch ImageJ and open `raw-test.raw` with these settings that match the video stream size, e.g.:

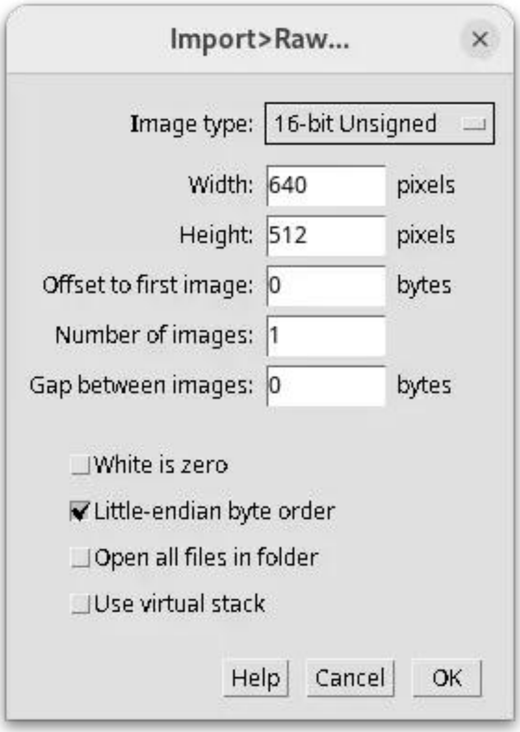


Figure 9: ImageJ settings.

ImageJ will automatically perform histogram scaling to show an image something like this:

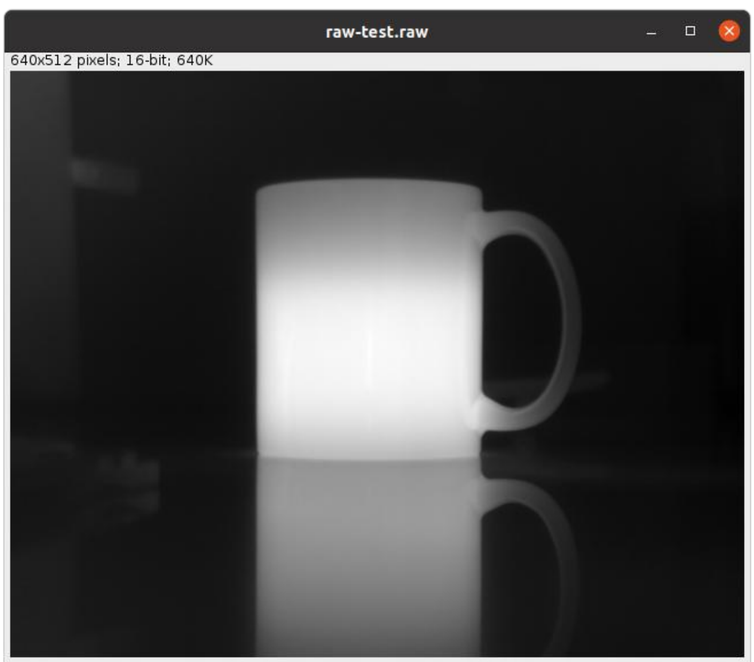


Figure 10: 16-bit image.

7.2.4 Display Video with FFplay

FFmpeg can be used to display video that is adjusted into the viewable range with histogram normalization:

```
$ ~/soti_demo_scripts/ffplay.sh
```

The video output can be closed by pressing `q` or clicking the window's close button. The resulting video will look something like this:

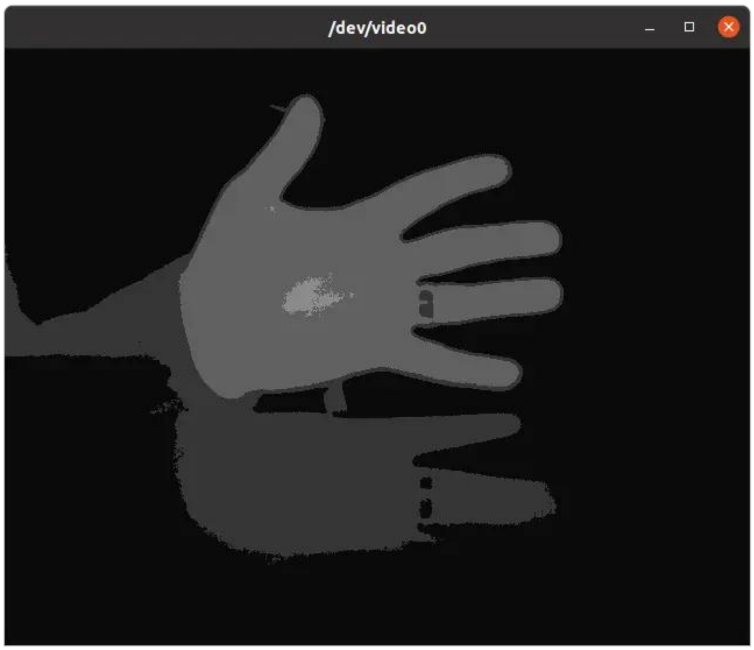


Figure 11: FFplay 16-bit Video.

NOTE: The FFmpeg build available from the Ubuntu repositories do not handle 16-bit unsigned video correctly. Only 8 bits of the video are processed, so the video is of low quality. Building FFmpeg manually with 16-bit support can correct this problem.

7.3 Limitations

7.3.1 V4L2 Driver

- The reference MIPI driver contains the minimum set of features to enable display and capture of video. The following standard V4L2 functions are unimplemented:
  1. group_hold
  2. sensor_mode
  3. gain
  4. exposure

5. frame_rate
6. sensor_configuration
7. sensor_mode_i2c_packet
8. sensor_control_i2c_packet
9. bypass_mode
10. override_enable
11. height_align
12. size_align
13. write_isp_format
14. sensor_signal_properties
15. sensor_image_properties
16. sensor_control_properties
17. sensor_dv_timings
18. low_latency_mode
19. preferred_stride
20. sensor_modes

- The Device Tree camera definition supports only one camera at a fixed resolution.
- Support exists only for NVIDIA JetPack 5.1.3

7.3.2 **Video Display**

Options for viewing 16-bit video with pre-built software are limited. Custom solutions can be developed using OpenCV.

8 DRIVER SETUP

This section contains instructions on building a V4L2 video driver for a DRS Tenum 640 or 1280 with the Sierra-Olympia MIPI adaptor board. These steps have already been completed on the Developer Kit provided by Sierra-Olympia and are intended for customers developing drivers for other embedded systems. It assumes that the reader has basic familiarity with Linux operating systems and software development for Linux. At the completion of these instructions, the reader will be able to capture 16-bit raw video produced by a JETSON ORIN NANO Developer Kit with a Tenum camera attached to the MIPI connector.

Two files distributed with this document are required for completion:

1. `setupBuildEnvironment`—Creates convenience environment variables for build environment.

2. `5.1.3_nano_drs-tenum_v0.1.1.tar.xz`—Patch file containing driver source and modifications.

8.1 Setup Hardware

8.1.1 Ubuntu 18.04

Development must be done on Ubuntu 18.04 for best compatibility with NVIDIA JetPack 5.1.3. This can be accomplished on real hardware or a virtual machine. The Ubuntu computer used for development will be referred to as the build host, or host, in this guide.

> **NOTE: At least 50 GiB of free storage space must be available on the host.**
>
> **NOTE: The SDK Manager flash operation can be unreliable when executed on a Virtual Machine. Consider using real hardware if the flashing process fails.**

8.1.2 JETSON ORIN NANO Developer Kit

Install the NVIDIA SDK Manager on the build host by following the "Get Started" instructions.

> **NOTE: A free NVIDIA developer account is required for download and use of the NVIDIA SDK Manager.**

Connect the JETSON ORIN NANO (hereafter referred to as "the Orin Nano") to the host using a USB 3.0 cable connected to the USB-C port on the Orin Nano. Insert a blank microSD card (or one that can be formatted) that's at least 64 GiB in size into the Orin Nano, and then power-on the Orin Nano.

> **NOTE: The Orin Nano's USB port is active only when in recovery mode. To force the Orin Nano into recovery mode, use a jumper to connect the "FC REC" pin to a "GND" pin. After powering on you should see a USB device listed in the output of lsusb on the host computer with a name similar to "NVIDIA Corp. APX".**

Follow NVIDIA's instructions for installing the NVIDIA software on both the target and the host using the SDK Manager on the host PC (the "Optional Flow with Host Linux PC" in NVIDIA's instructions). These instructions are summarized below for clarity.

WARNING: Install JetPack 5.1.3 for compatibility with the example driver in this document. Other versions of JetPack are not compatible with this example driver.

In the SDK Manager step 01, use the following selections:

- Product Category: Jetson (yes), Data Science (no)
- System Configuration:
  - Host Machine Ubuntu 18.04 - x86_64 (yes)
  - Target Hardware Jetson Orin Nano modules [8GB developer kit version] (yes)
- SDK Version: JetPack 5.1.3
- Additional SDKs: DeepStream (no)

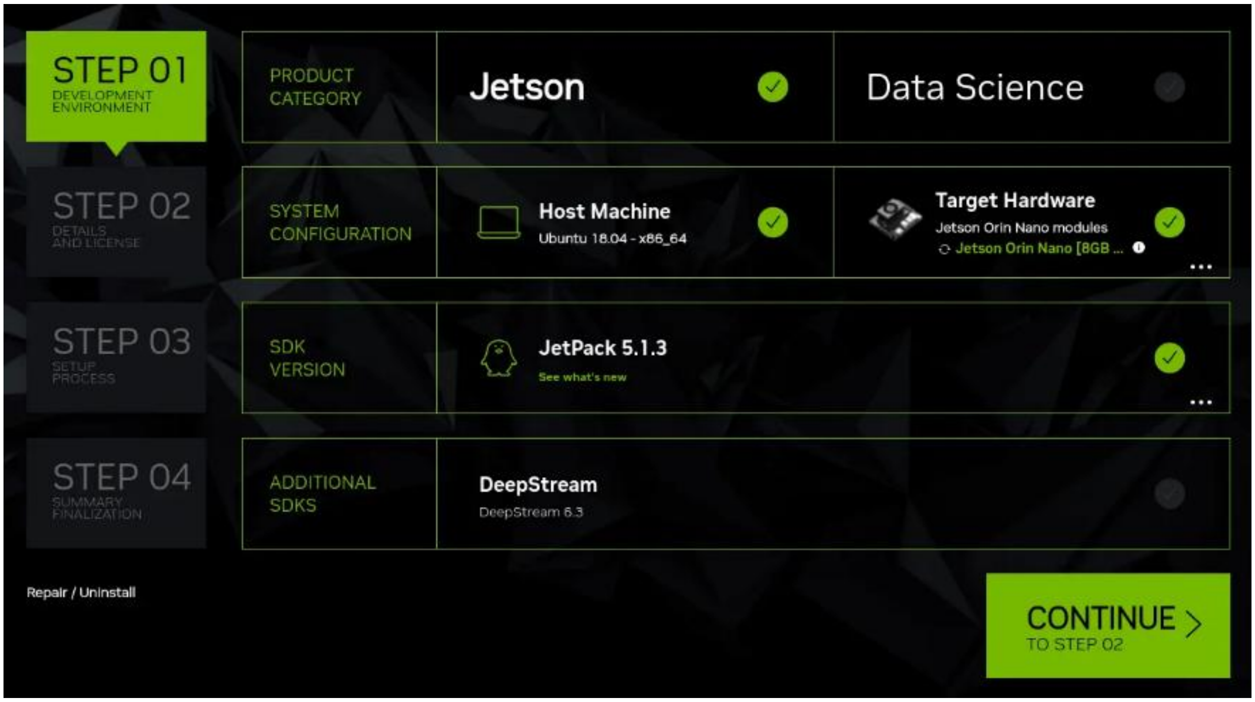


Figure 12: SDK Manager Step 01.

Then click "Continue to Step 02". Use the default selections as shown below, accept the terms and conditions, and then click "Continue to Step 03".

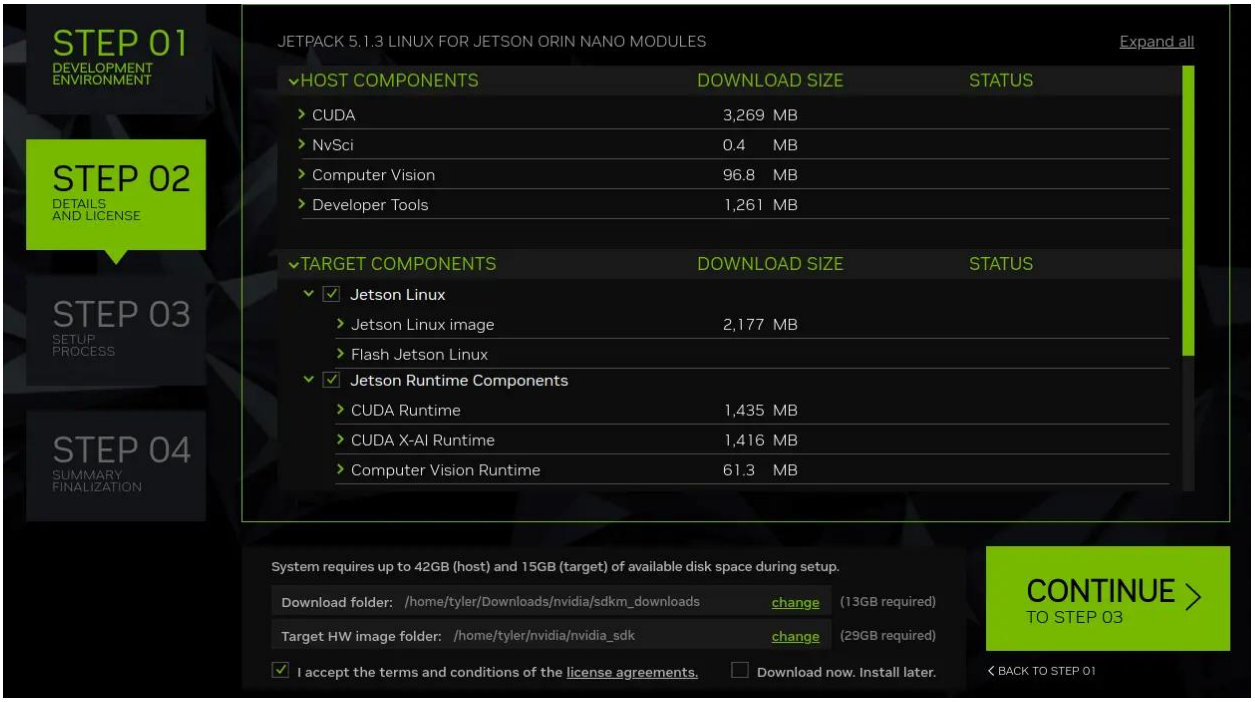


Figure 13: SDK Manager Step 02-a.

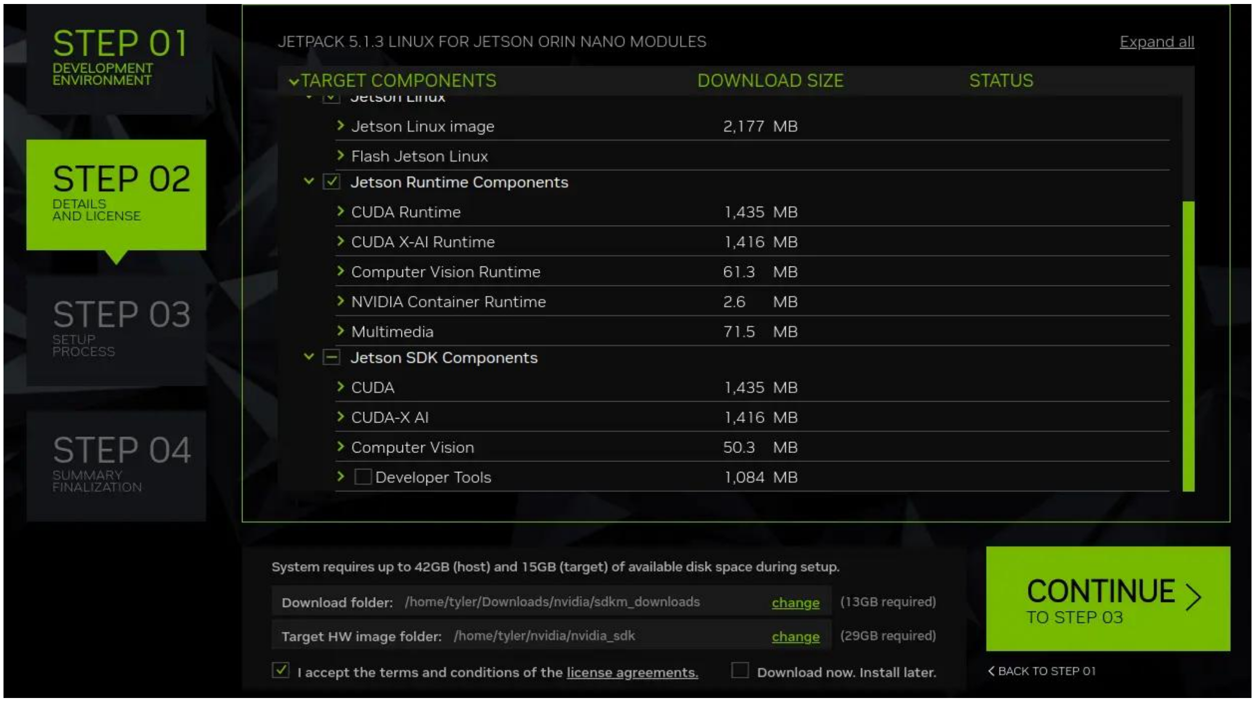


Figure 14: SDK Manager Step 02-b.

Use the selections below for step 03, and then click "Flash".

1. OEM Configuration: Pre-Config
   Enter the desired username and password for the target.
2. Storage Device: SD Card

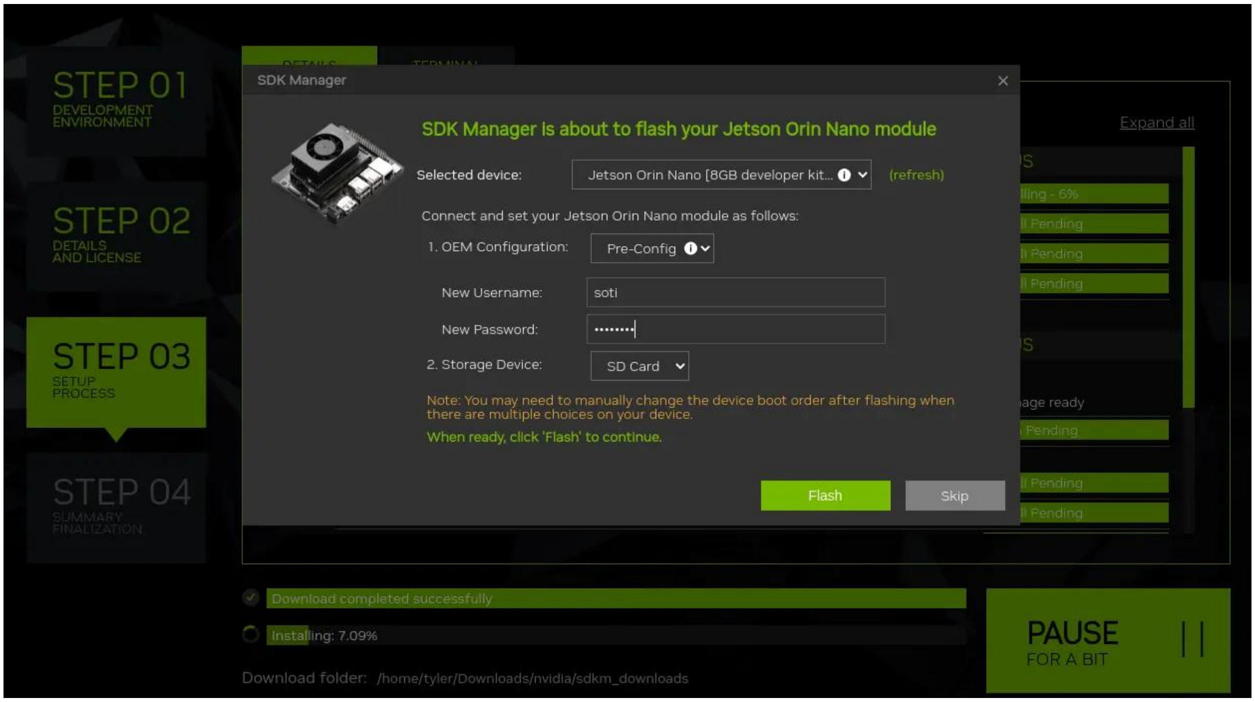


Figure 15: SDK Manager Step 03-a.

The flashing process will first install the operating system on the Orin Nano, and then show a prompt before installing SDK components to the Orin Nano. At this point, connect the Orin Nano to a mouse, keyboard, and monitor. Log in to the system using the username and password defined earlier and complete the Ubuntu welcome prompts.

When booting the Orin Nano after initial setup, the desktop environment on the target will look something like Figure 4.

Flashing the SDK components to the Orin Nano can be done via USB or Ethernet. Ethernet is faster and therefor recommended. Connect Ethernet to the Orin Nano, and set or note its IP address. Enter this information into the SDK manager prompt on the build host and click "Install".

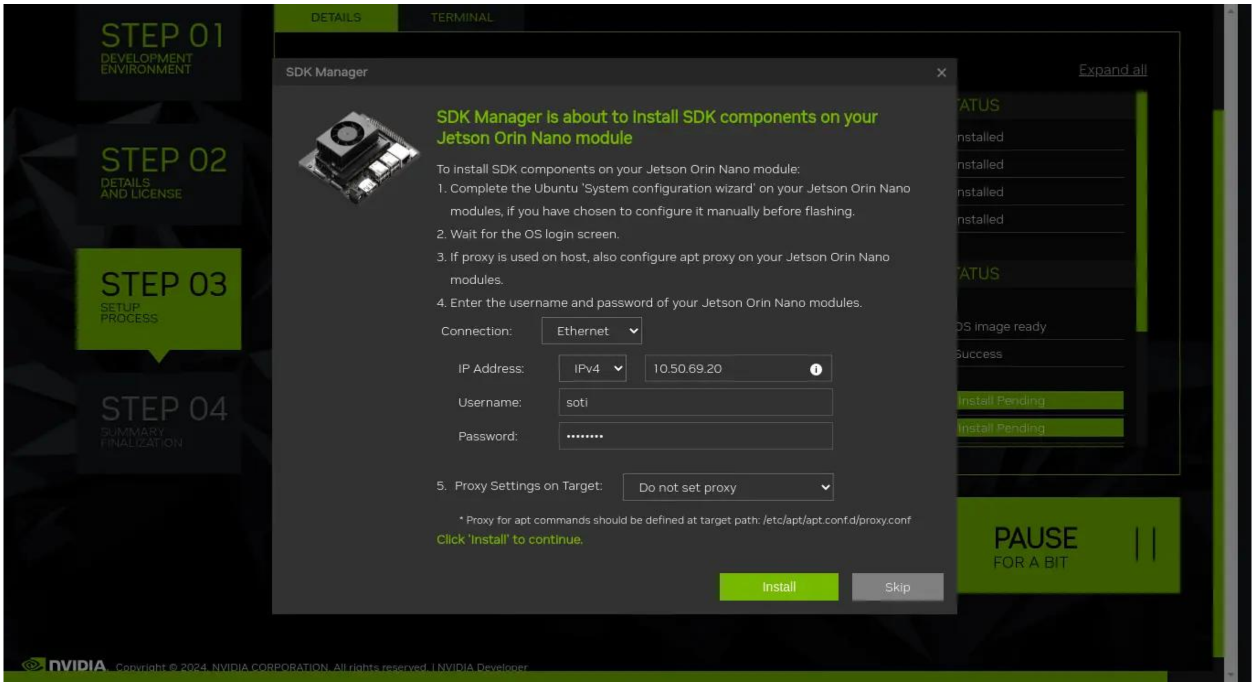


Figure 16: SDK Manager Step 03-b.

Flashing can take up to one hour, and on completion will report the installation status.

Figure 17: SDK Manager Step 04.

WARNING: Do not install any Ubuntu updates on the target. Doing so may lead to automatic upgrades to newer JetPack versions and a loss of compatibility with this example.

8.1.3 Manual Flashing

Attempting to flash the Orin Nano with a microSD card < 64 GiB in size will result in a SDK Manager failure.

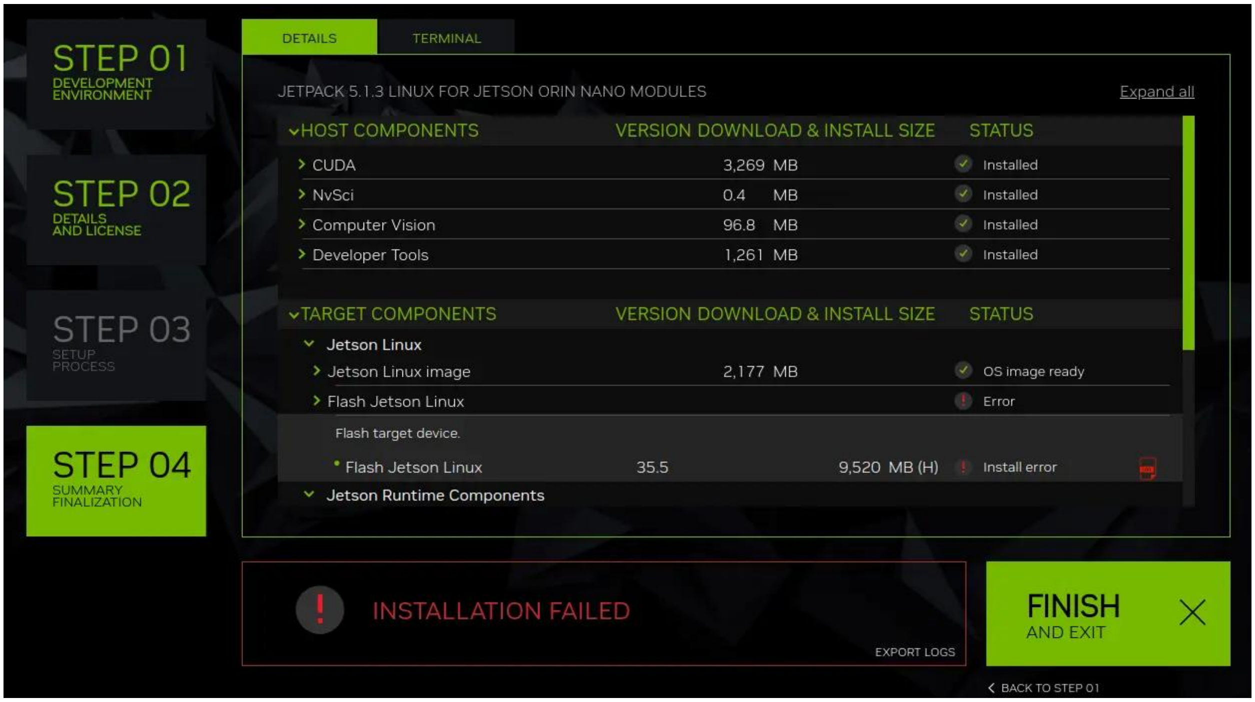


Figure 18: SDK Manager failed flash.

However, SD cards sized 32–64 GiB and can be flashed on the host computer using NVIDIA's instructions. The JetPack 5.1.3 microSD card image for Jetson Orin Nano Developer Kit can be downloaded here. The microSD card location on the Orin Nano is shown below.

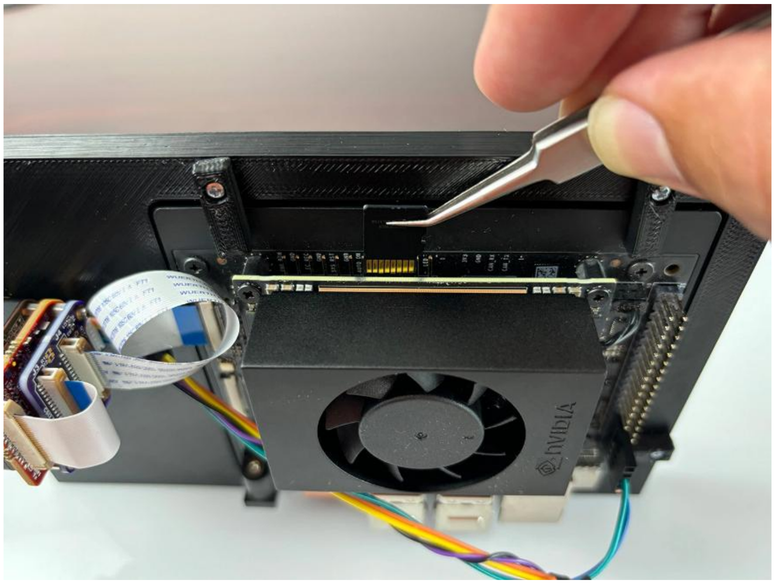


Figure 19: Inserting microSD card.

The username and password for the target will be set by the user on first boot when using the SD card flash method.

The QSPI storage on the Orin Nano must be flashed with firmware that matches the JetPack 5.1.3. microSD card image. Using the same hardware arrangement described above run the following commands from a terminal:

```
$ cd <JETPACK_INSTALL_DIR>/Linux_for_Tegra
$ sudo ./flash.sh -c bootloader/t186ref/cfg/flash_t234_qspi.xml \
    jetson-orin-nano-devkit mmcblk1p1
```

The flash process will take 5–10 minutes.

8.2 Build Driver

In this guide it is assumed that a directory named JetPack_5.1.3_Linux_JETSON_ORIN_NANO_TARGETS is created on the host by the SDK Manager during setup, and that location will be used in the following examples. Other directory names can be used if desired. For more information please refer to the RidgeRun wiki.

8.2.1 Get Kernel Sources

Download the "Driver Package (BSP) Sources" for NVIDIA Jetson Linux 35.5.0 sources (part of JetPack 5.1.3) from NVIDIA. The link will download a file with the name `public_sources.tbz2`.

Extract the kernel sources:

```
$ cd ~/Downloads
$ tar -xf public_sources.tbz2
```

Copy `kernel_src.tbz2` to <JETPACK_INSTALL_DIR>/Linux_for_Tegra/sources. In this guide <JETPACK_INSTALL_DIR> will be `JetPack_5.1.3_Linux_JETSON_ORIN_NANO_TARGETS`.

```
$ cd Linux_for_Tegra/source/public
$ mkdir -p <JETPACK_INSTALL_DIR>/Linux_for_Tegra/sources
$ cp kernel_src.tbz2 <JETPACK_INSTALL_DIR>/Linux_for_Tegra/sources
```

Extract kernel sources:

```
$ cd <JETPACK_INSTALL_DIR>/Linux_for_Tegra/sources
$ tar -xf kernel_src.tbz2
```

At this point, the kernel sources are placed in the expected sources folder within the JetPack path.

8.2.2 Install Toolchain

Download the toolchain tar file from NVIDIA.

To install the toolchain in the host machine, first create the toolchain folder:

```
$ sudo mkdir /opt/l4t-gcc-toolchain-64-bit
```

Enable permissions and enter folder:

```
$ sudo chown $USER:$USER /opt/l4t-gcc-toolchain-64-bit
$ cd /opt/l4t-gcc-toolchain-64-bit
```

Move and install toolchain:

```
$ mv ~/Downloads/aarch64--glibc--stable-final.tar.gz .
$ tar -xf aarch64--glibc--stable-final.tar.gz
```

The toolchain is now installed at `/opt/l4t-gcc-toolchain-64-bit`.

8.2.3 Prepare the Build Environment

A few additional build tools and libraries are required when building the driver. They are installed with the following command (on an Ubuntu host):

```
$ sudo apt install quilt flex bison libssl-dev
```

Some convenient environment variables, e.g. `$DEVDIR` used below, must be declared prior to using the toolchain. This is accomplished by sourcing `setupBuildEnvironment` (included with this document):

```
$ cd ~/Downloads
$ source setupBuildEnvironment
```

8.2.4 Apply the Patch

Move the `patches` folder included with this document into your `$DEVDIR/sources` directory, adjacent to the hardware and kernel directories.

Apply the patches from the `$DEVDIR/sources` directory as follows:

```
$ cd $DEVDIR/sources
$ quilt push 5.1.3_nano_drs-tenum_v0.1.3.patch
```

Additional patches are included that can be used to describe different Tenum camera hardware arrangements in the device tree. The options are:

Patch	**CAM0**	**CAM1**
5.1.3_nano_drs-tenum_v0.1.3.patch	640	None
cam1.diff	None	640
cam0-and-cam1.diff	640	640
tenum1280.diff	1280	640

These patches can be applied using the `quilt push` and `quilt pop` commands.

8.2.5 Build

The full Kernel compilation requires four steps:

1. Build Kernel Image
2. Build Device-Tree
3. Build Kernel Modules
4. Install Files

Create the default build configuration files (only needed for the first build).

```
$ cd $DEVDIR/sources/kernel/kernel-5.10
$ make O=$TEGRA_KERNEL_OUT tegra_defconfig
```

Build the kernel image:

```
$ make O=$TEGRA_KERNEL_OUT Image -j8
```

Build the Device-Tree binary:

```
$ make O=$TEGRA_KERNEL_OUT dtbs
```

Build the kernel modules:

```
$ make O=$TEGRA_KERNEL_OUT modules -j8
$ make O=$TEGRA_KERNEL_OUT modules_install \
    INSTALL_MOD_PATH=$KERNEL_MODULES_OUT
```

Delete the following files as they are not necessary:

```
$ rm $MODULES/build $MODULES/source
```

Copy the extra directory to enable graphics output:

```
$ cp -r $DEVDIR/rootfs/usr/lib/modules/5.10.192-tegra/extra $MODULES/
```

8.2.6 Copy Build Output to the Orin Nano

The scp application will be used to copy build output files to the target over Ethernet. The example commands below are constructed with the assumption that the SSH config file (`~/.ssh/config`) has an entry named "target" that contains the target's IP address, username, and (optionally) authentication details. If needed, see the Digital Ocean tutorial for help getting started with the SSH config file.

Copy the kernel image:

```
$ scp $IMAGE target:/tmp
```

Copy the Device-Tree binary:

```
$ scp $DTB target:/tmp
```

Copy the modules directory (only necessary the first time):

```
$ scp -r $MODULES target:/tmp
```

Copy the DRS Tenum modules only (for later changes in the modules):

```
$ scp $TENUM0640 target:/tmp
$ scp $TENUM1280 target:/tmp
```

8.2.7 Move Files to Final Destination on the Orin Nano

The following commands should be executed on the Orin Nano via SSH or physical console. Consider making backups before overwriting any existing files.

Move the kernel image:

```
$ sudo mv /tmp/Image /boot/Image
```

Move the Device-Tree binary (please check `/boot/extlinux/extlinux.conf` to verify the correct location):

```
$ sudo mv /tmp/tegra234-p3767-0003-p3768-0000-a0.dtb \
    /boot/dtb/kernel_tegra234-p3767-0003-p3768-0000-a0.dtb
```

Move the modules directory (only necessary the first time):

```
$ sudo mv /tmp/5.10.192 /lib/modules/
```

Move the DRS Tenum module only (for later changes in the module):

```
$ sudo mv /tmp/drstenum0640.ko \
    /lib/modules/5.10.192/kernel/drivers/media/i2c/drstenum0640.ko
$ sudo mv /tmp/drstenum1280.ko \
    /lib/modules/5.10.192/kernel/drivers/media/i2c/drstenum1280.ko
$ sudo depmod -a
```

Reboot:

```
$ sudo reboot
```

8.2.8 Enable the GUI

If the display fails to start after the above changes were made, configure the Orin Nano to use the new NVIDIA kernel modules with the following commands after logging into the Orin Nano via SSH:

```
$ sudo insmod /lib/modules/5.10.192/extra/opensrc-disp/nvidia.ko

$ sudo insmod /lib/modules/5.10.192/extra/opensrc-disp/nvidia-modeset.ko

$ sudo depmod -a

$ sudo reboot
```

8.2.9 Verify Driver Loaded

Reboot the target and check the kernel messages for mention of the `drstenumXXXX` driver. Two cameras are detected in this example:

```
$ sudo dmesg | grep drs

[   17.566002] drstenum0640 9-0033: probing v4l2 sensor at addr 0x33

[   17.577910] drstenum0640 9-0033: In drstenum0640_parse_dt

[   17.588955] drstenum0640 9-0033: In drstenum0640_power_get

[   17.589150] drstenum0640 9-0033: tegracam sensor
driver:drstenum0640_v2.0.6

[   17.600272] drstenum0640 9-0033: tegra camera driver registration
succeeded.

[   17.613059] tegra-camrtc-capture-vi tegra-capture-vi: subdev drstenum0640
9-0033 bound

[   17.627042] drstenum0640 9-0033: detected drstenum 640 sensor

[   17.639841] drstenum0640 10-0033: probing v4l2 sensor at addr 0x33

[   17.651416] drstenum0640 10-0033: In drstenum0640_parse_dt

[   17.668944] drstenum0640 10-0033: In drstenum0640_power_get

[   17.674817] drstenum0640 10-0033: tegracam sensor
driver:drstenum0640_v2.0.6

[   17.687923] drstenum0640 10-0033: tegra camera driver registration
succeeded.

[   17.687989] tegra-camrtc-capture-vi tegra-capture-vi: subdev drstenum0640
10-0033 bound
```

Also verify the driver module is loaded:

```
$ lsmod | grep drs

drstenum0640     16384 0
```

9 TENUM 1280 SUPPORT

The device tree can be configured to support Tenum 1280 cameras by applying the changes in tenum1280.diff:

```
$ cd $DEVDIR/sources
$ quilt push tenum1280.diff
```

The device tree binary must be rebuilt:

```
$ cd $DEVDIR/sources/kernel/kernel-5.10
$ make O=$TEGRA_KERNEL_OUT dtbs
```

Copy the updated binaries to the Orin Nano using the steps described earlier, and then restart the Orin Nano.

> **NOTE: Any preceding example commands holding references to the 640x512 resolution should be adjusted to 1280x1024 before use with a Tenum 1280.**

9.1 Tenum 1280 Configuration

SOTI's MIPI backend can support both 640 and 1280 Tenum cameras and will be configured to operate at the resolution matching the camera it is shipped with. The MIPI backend resolution may be changed after startup, if desired. The default resolution will be restored on the next power cycle. The example below demonstrates how to switch the backend from 640 to 1280 resolution using a system unit file.

First ensure that systemd tracks the `drstenum` kernel module's load state by adding the module's name to the bottom of the `/etc/modules-load.d/modules.conf` file.

Next create a script that sends the desired camera configuration commands in `/home/soti/soti_demo_scripts/1280init-16bit.sh` with these contents:

```
# Configure the MIPI FPGA and camera core for 16-bit unsigned 1280x1024 resolution.
# Done for cameras that are configured to output 640x512 by default.
# Check for input argument and set i2cbus accordingly.
if [ "$1" == "CAM1" ]; then
    i2cbus=10
else # Default to CAM0
    i2cbus=9
fislave_addr=0x33
# Send resolution configuration command to FPGA.
i2ctransfer -f -y $i2cbus w8@$slave_addr 0x05 0x96 0x33 0x00 0x05 0x00 0x04 0x2E
# Send Y16 bit depth commands to camera core through UART bridge (not
```

```
# necessary if camera NVPs already in Y16 mode).
i2ctransfer -f -y $i2cbus w9@$slave_addr 0x04 0x01 0xB0 0x04 0x00 0x07 0x00 0x07 0x3D
i2ctransfer -f -y $i2cbus w9@$slave_addr 0x04 0x01 0xB0 0x04 0x00 0x4C 0x00 0x00 0xFF
i2ctransfer -f -y $i2cbus w9@$slave_addr 0x04 0x01 0xB0 0x04 0x00 0x05 0x00 0x01 0x45
# Report status.
echo 1280 configuration complete!
```

Then write a systemd unit file that runs the above script after the drstenum kernel module is loaded in `/etc/systemd/system/1280init.service` with these contents:

```
[Unit]
Description=Initialize 1280x1024 resolution on Viento MIPI camera.
Requires=systemd-modules-load
[Service]
Type=simple
ExecStart=/bin/bash /home/soti/soti_demo_scripts/1280init-16bit.sh
[Install]
WantedBy=multi-user.target
```

Finally, enable and start the service:

```
$ sudo systemctl enable 1280init.service
$ sudo systemctl start 1280init.service
```

On the next boot, systemd should run the 1280init-16bit.sh script so that the camera is configured to 1280 resolution.

10 REVISION HISTORY

Revision	Date	Description	ECO
Rev A	2025-02-05	Initial release.	1810
Rev B	2025-04-01	Updated Section 4 and Figure 10. Clarified section 9.	1837
Rev C	2025-04-24	Update build instructions for dual cameras.	1860