



StarFive
赛昉科技

VisionFive 2 AI Kit Quick Start Guide

Version: 1.0

Date: 2024/11/07

Doc ID: VisionFive 2-QSGCH-002

Legal Statements

Important legal notice before reading this documentation.

PROPRIETARY NOTICE

Copyright©Guangdong StarFive Technology Co., Ltd., 2024. All rights reserved.

Information in this document is provided "as is," with all faults. Contents may be periodically updated or revised due to the product development. Guangdong StarFive Technology Co., Ltd.(hereinafter "StarFive") reserves the right to make changes without further notice to any products herein.

StarFive expressly disclaims all warranties, representations, and conditions of any kind, whether express or implied, including, but not limited to, the implied warranties or conditions of merchantability, fitness for a particular purpose and non-infringement.

StarFive does not assume any liability rising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation indirect, incidental, special, exemplary, or consequential damages.

All material appearing in this document is protected by copyright and is the property of StarFive. You may use this document or any part of the document for internal or educational purposes only, provided you do not modify, edit or take out of context the information in this document in any manner. Information contained in this document may be used, at your sole risk, for any purposes. StarFive authorizes you to copy this document, provided that you retain all copyright and other proprietary notices contained in the original materials on any copies of the materials and that you comply strictly with these terms. This copyright permission does not constitute an endorsement of the products or services.

Contact Us

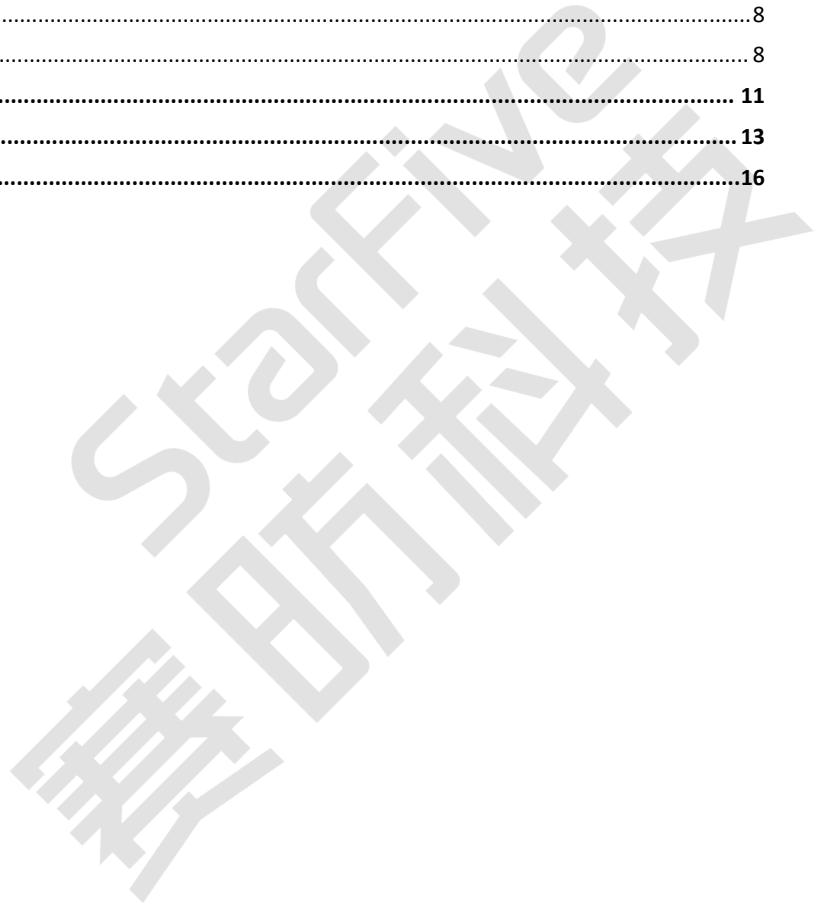
Address: Room S201, Zone A, No. 2, Haoyang Road, Yunlu Community, Daliang Subdistrict, Shunde District, Foshan, Guangdong, China, 528300

Website: <http://www.starfivetech.com> <http://www.starfivetech.com>

Email: sales@starfivetech.com(sales) , support@starfivetech.com(support)

Contents

List of Tables.....	4
List of Figures.....	5
Legal Statements.....	2
Preface.....	6
1. Introduction.....	7
2. Preparation.....	8
2.1. Hardware Preparation.....	8
2.2. Software Preparation.....	8
3. Compilation.....	11
4. Demo.....	13
5. Appendix.....	16



List of Tables

Table 0-1 Version History.....6



List of Figures

Figure 2-1 VisionFive 2 AI Kit.....8

Figure 3-1 Missing head file..... 11

Figure 3-2 Error..... 11

Figure 3-3 Hailo-8 Module Information..... 12

Figure 4-1 Enable Hailo Monitor..... 13

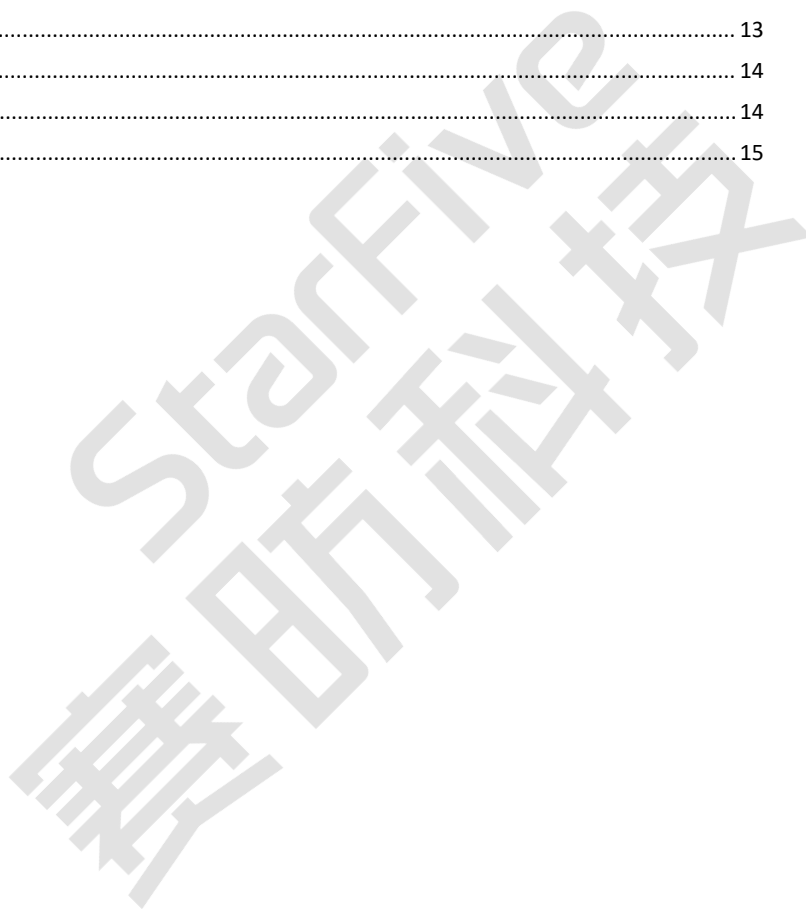
Figure 4-2 Specify environment variables..... 13

Figure 4-3 Monitor Data..... 13

Figure 4-4 Change Startup Scripts..... 14

Figure 4-6 Error..... 14

Figure 4-7 Error..... 15



Preface

About this guide and technical support information.

About this document

This document mainly provides the users with basic information and compilation methods about StarFive VisionFive 2 AI Kit, including hardware and software preparation, compilation, demo cases.






Version History

Table 0-1 Version History

Version	Released	Revision
1.0	2024/11/07	The First Official Release.

Notes and notices

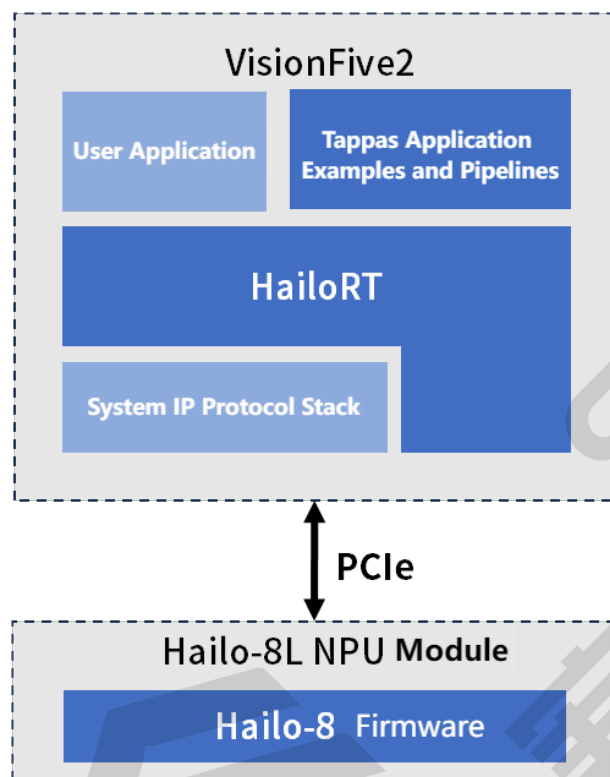
The following notes and notices might appear in this guide:

-  **Tip:**
Suggests how to apply the information in a topic or step.
-  **Note:**
Explains a special case or expands on an important point.
-  **Important:**
Points out critical information concerning a topic or step.
-  **CAUTION:**
Indicates that an action or step can cause loss of data, security problems, or performance issues.
-  **Warning:**
Indicates that an action or step can result in physical harm or cause damage to hardware.

1. Introduction

VisionFive 2 AI Kit is jointly launched by StarFive and Hailo. It aims to provide the most cost-effective high-performance RISC-V AI solutions for applications in edge computing, industrial intelligence, security, robotics, gateway routing, smart home and other fields.

VisionFive 2 AI Kit bundles the VisionFive 2 high performance RISC-V SBC with a Hailo-8L M.2 AI acceleration module: Using VisionFive 2 as the platform, HailoRT, HailoRT-driver, Tappas and other components were ported and run on it. By combining PCIe with Hailo-8L AI module, AI application acceleration was achieved, and YOLOv5, YOLOv8, MobileNet_SSD and other models were successfully run on it. The Kit provides an low latency and power efficient way to integrate complex AI vision applications and efficiently perform deep learning inference tasks such as object recognition, image segmentation, and pose analysis.



2. Preparation

This chapter mainly introduces the preparations required for VisionFive 2 AI Kit, including:

- [Hardware Preparation \(on page 8\)](#)
- [Software Preparation \(on page 8\)](#)

2.1. Hardware Preparation

Required Hardware

- VisionFive 2
- Micro SD card (32 GB or more)
- Hailo-8L M.2 AI acceleration module
- USB camera (logi HD 1080P)
- An HDMI monitor
- Heat sink or fan (for cooling Hailo-8L M.2 AI acceleration module)
- A keyboard
- A mouse

Connection Type

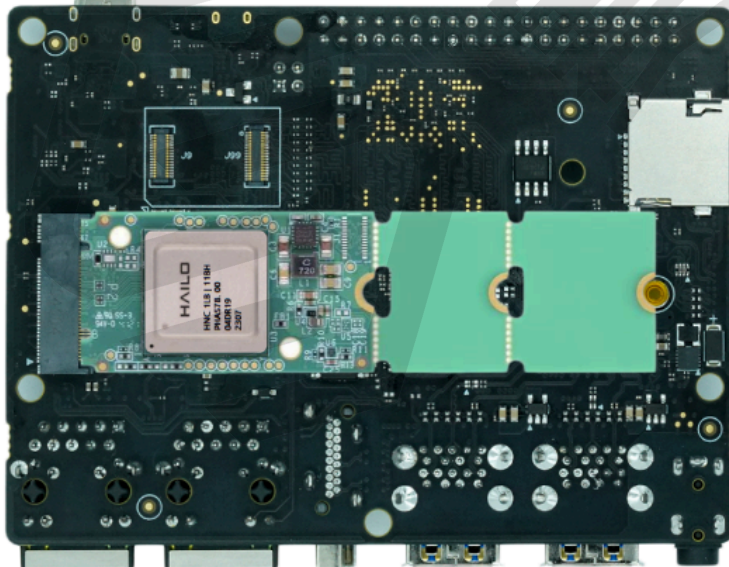
The Hailo-8L AI acceleration module is connected through the M.2 M-Key of VisionFive 2, as shown in the following figure:



Tip:

StarFive recommends you to use a heat sink or a fan for heat dissipation.

Figure 2-1 VisionFive 2 AI Kit



2.2. Software Preparation

This section introduces the following 3 required software preparations:

- [Required Image/Code Version \(on page 9\)](#)
- [Image Flash/Dependency Library Installation \(on page 9\)](#)
- [Obtain Source Code and Apply Patches \(on page 10\)](#)

Required Image/Code Version

- [Debian12](#) (202409)
- [debian-deb](#) (To adapt to the updated files such as Debian kernel packages required by Hailo)
- Tappas (3.30.0, daffd36ecab5110d47107255fd7ec4c779758f2e)
- HailoRT (v4.19.0, ac19e12b86170e1b0967e7d8aa607a0100cb0077)
- HailoRT-driver (v4.19.0, eb2a8649752abd424c6d2e5109e9ec92d6d2d5f6)
- [Hailort and Tappas patches](#) (Used for adapting VisionFive 2's HailoRT and Tappas patches)
- [NpuDetectorLib demo](#)

Image Flash/Dependency Library Installation

1. Please refer to [the link](#) to flash Debian into SD card.



Tip:

StarFive recommends you to perform following operations under user account to avoid permission exceptions caused by root account.

2. Update kernel and head files:

- a. Run the following command to download and extract the hailort-deb-1.tar.gz file:

```
$ tar -xzf hailort-deb-1.tar.gz
```

- b. Run the following command to update u-boot-menu:

```
$ cd hailort-deb
$ sudo dpkg -i u-boot-menu_4.2.2-SF113_all.deb
```

- c. Run the following command to update the kernel, header files, libc library, and VPU driver:

```
$ cd 6.6
$ sudo dpkg -i ./*
```

- d. Run the following command to reboot system:

```
$ sudo reboot
```

3. Run the following command to install dependencies:

```
$ sudo apt install make cmake automake build-essential autoconf bc bison meson flex wget curl git
git-lfs libgirepository1.0-dev gcc g++ rsync x11-utils -y
```

4. Install Python environment. Install python3-platformdirs. The new version of Snapshot does not support python3 platformdirs below version 3.0 and requires manual retrieval and installation:

```
$ wget
https://snapshot.debian.org/archive/debian/20221210T034654Z/pool/main/p/platformdirs/python3-platformd
irs_2.6.0-1_all.deb && sudo dpkg -i python3-platformdirs_2.6.0-1_all.deb
$ sudo apt install python3.11 python3.11-dev python3-setuptools python3-virtualenv python3-pip
python-gi-dev -y
```

5. Run the following command to install GStreamer related library:

```
$ sudo apt install libgstreamer-plugins-bad1.0-dev libgstreamer-plugins-base1.0-dev libgstreamer1.0-dev
-y
```

6. Run the following command to install OpenCV and GStreamer add-ons:

```
$ sudo apt install libopencv*-dev libssl-dev pciutils libcairo2-dev libzmq3-dev gstreamer1.0-tools -y
```

7. Execute the following command to install Rust:



Note:

This is required when compiling packages such as pydantic.

```
$ curl --proto '=https' --tlsv1.2 -sSf https://sh.rustup.rs | sh
$ source "$HOME/.cargo/env"
```

8. Run the following command to specify python3 as python3.11:

```
$ sudo update-alternatives --install /usr/bin/python3 python3 /usr/bin/python3.11 1
```

Obtain Source Code and Apply Patches

- Run the following command to obtain the Tappas source code and apply it:

```
$ git clone https://github.com/hailo-ai/tappas.git
$ cd tappas
$ git checkout daffd36ecab5110d47107255fd7ec4c779758f2e
$ cp path_to_hailo_patches/0001-Added-tappas-adaptation-to-VisionFive2.patch ./
$ git apply 0001-Added-tappas-adaptation-to-VisionFive2.patch
```

- Run the following command to obtain the HailoRT source code and apply it:



Note:

Please pull and store it in the Tappas path for future compilation.

```
$ mkdir -p tappas/hailort
$ cd tappas/hailort
$ git clone https://github.com/hailo-ai/hailort.git sources
$ cd source
$ git checkout ac19e12b86170e1b0967e7d8aa607a0100cb0077
$ cp path_to_hailo_patches/0001-Added-HailoRT-adaptation-to-VisionFive2.patch ./
$ git apply 0001-Added-HailoRT-adaptation-to-VisionFive2.patch
```

- Run the following command to obtain the HailoRT-drivers source code and apply it:

```
$ git clone https://github.com/hailo-ai/hailort-drivers.git
$ cd hailort-driver
$ git checkout eb2a8649752abd424c6d2e5109e9ec92d6d2d5f6
```

3. Compilation

HailoRT

1. Run the following command to compile HailoRT:

```
$ cd tappas/hailort/sources
$ cmake -S. -Bbuild -DCMAKE_BUILD_TYPE=Release -DHAILO_BUILD_GSTREAMER=1
$ sudo cmake --build build --config release --target gsthailo install -j4
```

As shown in the following figure, after the compilation and installation of HailoRT, the `hailort_dma-heap.h` head file is missing from the installation path and needs to be copied to the corresponding path manually:

Figure 3-1 Missing head file

```
user@starfive:/usr/local/include/hailo$ ls
buffer.hpp          hailort_common.hpp      platform.h
device.hpp          hailort_defaults.hpp   quantization.hpp
dma_mapped_buffer.hpp hef.hpp                 runtime_statistics.hpp
event.hpp           infer_model.hpp        stream.hpp
expected.hpp        inference_pipeline.hpp transform.hpp
hailort.h           network_group.hpp      vdevice.hpp
hailort.hpp         network_rate_calculator.hpp vstream.hpp
user@starfive:/usr/local/include/hailo$ ls /home/user/tappas/hailort/sources/hailort/libhailort/include/hailo/
buffer.hpp          hailort_defaults.hpp   quantization.hpp
device.hpp          hailort_dma-heap.h    runtime_statistics.hpp
dma_mapped_buffer.hpp hef.hpp                 stream.hpp
event.hpp           infer_model.hpp        transform.hpp
expected.hpp        inference_pipeline.hpp vdevice.hpp
hailort.h           network_group.hpp      vstream.hpp
hailort.hpp         network_rate_calculator.hpp platform.h
hailort_common.hpp platform.h
user@starfive:/usr/local/include/hailo$
```

2. Run the following command to copy the `hailort_dma-heap.h` head file to the corresponding path:

```
$ sudo cp hailort/libhailort/include/hailo/hailort_dma-heap.h /usr/local/include/hailo/
```

Otherwise, the following errors will be encountered during Tappas compilation and installation:

Figure 3-2 Error

```
-- Build files have been written to: /home/user/tappas/hailort/sources/hailort/libhailort/bindings/gstreamer/build
[ 6%] Building CXX object CMakeFiles/gsthailo.dir/gst-hailo/gsthailoplugin.cpp.o
In file included from /home/user/tappas/hailort/sources/hailort/libhailort/bindings/gstreamer/gst-hailo/os/linux/dma_b
uf_allocator_wrapper.hpp:24,
                 from /home/user/tappas/hailort/sources/hailort/libhailort/bindings/gstreamer/gst-hailo/gsthailonet.hp
p:31,
                 from /home/user/tappas/hailort/sources/hailort/libhailort/bindings/gstreamer/gst-hailo/gsthailoplugin
.cpp:23:
/home/user/tappas/hailort/sources/hailort/libhailort/bindings/gstreamer/gst-hailo/os/linux/../../gsthailo_dmabuf_alloc
ator.hpp:24:10: fatal error: hailo/hailort_dma-heap.h: No such file or directory
   24 | #include "hailo/hailort_dma-heap.h"
      |          ^
compilation terminated.
gmake[2]: *** [CMakeFiles/gsthailo.dir/build.make:76: CMakeFiles/gsthailo.dir/gst-hailo/gsthailoplugin.cpp.o] Error 1
gmake[1]: *** [CMakeFiles/Makefile2:83: CMakeFiles/gsthailo.dir/all] Error 2
gmake: *** [Makefile:136: all] Error 2
```

HailoRT-driver

1. Run the following command to compile HailoRT-driver:

```
$ cd hailort-driver
$ make all -j$(nproc)
$ sudo make install
$ sudo modprobe hailo_pci
```

2. Firmware download and automatic loading settings.

- a. Run the following command to enter the top-level path of the `pci-driver` source code:

```
$ cd hailort-driver
$ ./download_firmware.sh
$ sudo mkdir -p /lib/firmware/hailo/
$ sudo mv hailo8_fw.<VERSION>.bin /lib/firmware/hailo/hailo8_fw.bin
```

```
$ sudo cp ./linux/pci/51-hailo-udev.rules /etc/udev/rules.d/  
$ sudo udevadm control --reload-rules && sudo udevadm trigger
```



Note:

<VERSION> is the same as the current HailoRT-driver version, please check and confirm on your own. In this example, it is 4.19.0.

- b. After the above operations, you need to restart HailoRT-driver. The following command can be used to verify whether HailoRT and HailoRT-driver are compiled and installed properly:

```
$ hailortcli fw-control identify
```

You can view the connected Hailo-8 module information:

Figure 3-3 Hailo-8 Module Information

```
user@starfive:~$ hailortcli fw-control identify  
Executing on device: 0001:01:00.0  
Identifying board  
Control Protocol Version: 2  
Firmware Version: 4.19.0 (release,app,extended context switch buffer)  
Logger Version: 0  
Board Name: Hailo-8  
Device Architecture: HAILO8  
Serial Number: HLLW MBA224901983  
Part Number: HM218B1C2LAE  
Product Name: HAILO-8 AI ACC M.2 B+M KEY MODULE EXT TEMP
```

Tappas

Run the following command to enter the top-level directory of Tappas source code:

```
$ cd tappas/  
$ ./install.sh --skip-hailort --target-platform vf2  
$ source /home/user/.hailo/tappas/tappas_env
```

4. Demo

System Setting

After booting up, StarFive recommends to turn off CPU auto frequency regulation and run the following command under root:

```
$ su -  
$ echo performance > /sys/devices/system/cpu/cpu0/cpufreq/scaling_governor
```

Enable HailoRT Monitor (optional)

1. Run the following command to enable Hailo monitor, which can monitor the utilization rate of NPU devices, models used, and FPS output in real-time:

```
$ hailortcli monitor
```

Figure 4-1 Enable Hailo Monitor



```
user@starfive: ~  
-----  
Device ID          Utilization (%)   Architecture  
-----  
Model             Utilization (%)   FPS             PID  
-----  
Model             Stream            Direction       Avg    Max    Min    Capacity  
-----  
Monitor did not retrieve any files. This occurs when there is no application currently running.  
If this is not the case, verify that environment variable "HAILO_MONITOR" is set to 1.  
█
```

2. Open another shell (this shell must be the one running the demo program), and specify the environment variables:

```
$ export HAILO_MONITOR=1
```

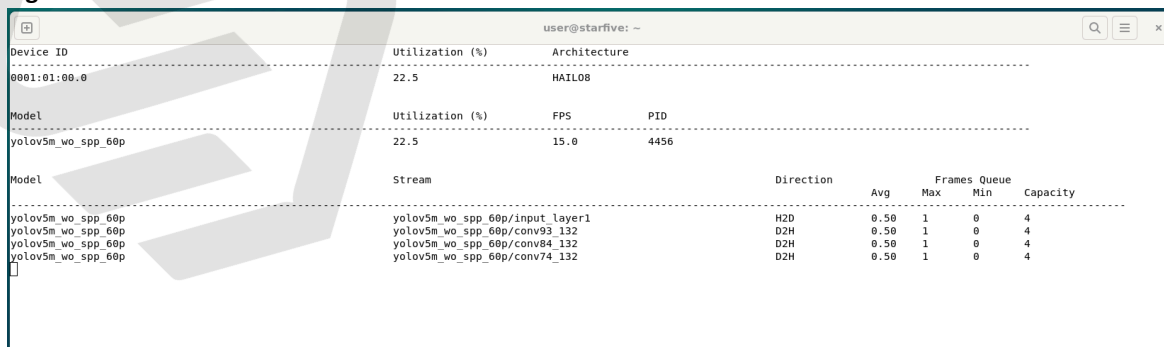
Figure 4-2 Specify environment variables



```
user@starfive:~/tappas/apps/h8/gstreamer/general/detection$ export HAILO_MONITOR=1  
user@starfive:~/tappas/apps/h8/gstreamer/general/detection$ ./detection.sh -i /dev/video4 -  
-network yolov5█
```

3. When executing the demonstration case, HailoRT detects that the *HAILO.MONITOR* variable of this shell process is 1, and will output real-time monitoring data of the NPU module:

Figure 4-3 Monitor Data



```
user@starfive: ~  
-----  
Device ID          Utilization (%)   Architecture  
-----  
0001:01:00:0      22.5              HAILO8  
-----  
Model             Utilization (%)   FPS             PID  
-----  
yolov5m_wo_spp_60p  22.5              15.0           4456  
-----  
Model             Stream            Direction       Avg    Max    Min    Capacity  
-----  
yolov5m_wo_spp_60p  yolov5m_wo_spp_60p/input_layer1  H2D    0.50  1    0    4  
yolov5m_wo_spp_60p  yolov5m_wo_spp_60p/conv93_132    D2H    0.50  1    0    4  
yolov5m_wo_spp_60p  yolov5m_wo_spp_60p/conv84_132    D2H    0.50  1    0    4  
yolov5m_wo_spp_60p  yolov5m_wo_spp_60p/conv74_132    D2H    0.50  1    0    4  
-----  
Monitor did not retrieve any files. This occurs when there is no application currently running.  
If this is not the case, verify that environment variable "HAILO_MONITOR" is set to 1.  
█
```

Tappas Demo

The following demo cases can be directly used under Tappas.

**Note:**

The demo under Tappas uses GStreamer videosink with xvimagesink and ximagesink, while VisionFive 2's Debian uses Wayland protocol, these demos need to modify their startup scripts by changing the ximagesink in video_stink_ element to waylandink:

Figure 4-4 Change Startup Scripts

```
video_sink_element=$( [ "$XV_SUPPORTED" = "true" ] && echo "xvimagesink" || echo "ximagesink")
video_sink_element=$( [ "$XV_SUPPORTED" = "true" ] && echo "xvimagesink" || echo "waylandink")
```

Otherwise, the following error will occur during runtime:

Figure 4-6 Error

```
Setting pipeline to PAUSED ...
Config file doesn't exist, using default parameters
Pipeline is PREROLLING ...
Redistribute latency...
X Error of failed request: BadValue (integer parameter out of range for operation)
Major opcode of failed request: 131 (XInputExtension)
Minor opcode of failed request: 46 ( )
Value in failed request: 0xd
Serial number of failed request: 51
Current serial number in output stream: 55
```

• Instance_segmentation:

```
$ cd /home/user/tappas/apps/h8/gstreamer/general/instance_segmentation/
$ ./instance_segmentation -i /dev/video4
```

**Note:**

The command `/dev/video4` specifies the used USB camera.

• Detection:

```
$ cd /home/user/tappas/apps/h8/gstreamer/general/instance_segmentation/
$ ./instance_segmentation -i /dev/video4 -- video4
```

**Note:**

Specify the use of YOLOv5 model through `-network`.

• Cascading_networks:

```
$ cd /home/user/tappas/apps/h8/gstreamer/general/cascading_networks/
```

1. face_detection_and_landmarks:

```
./face_detection_and_landmarks.sh -i /dev/video4
```

2. object_detection_and_pose_estimation:

```
./object_detection_and_pose_estimation.sh -i /dev/video4
```

**Note:**

When the following error occurs, check if the `TAPPAS-WORKSPACE` environment variable is missing.

**Figure 4-7 Error**

```
[HailoRT] [error] CHECK failed - Failed opening file, path: /apps/h8/gstreamer/general/cascading_networks/resources/lightface_slim.hef
[HailoRT] [error] CHECK_SUCCESS failed with status=HAILO_OPEN_FILE_FAILURE(13)
[HailoRT] [error] Failed parsing HEF file
[HailoRT] [error] Failed creating HEF
[HailoRT] [error] CHECK_SUCCESS failed with status=HAILO_OPEN_FILE_FAILURE(13)
[HailoRT] [error] CHECK_SUCCESS failed with status=HAILO_OPEN_FILE_FAILURE(13)
[HailoRT] [error] CHECK_SUCCESS failed with status=HAILO_OPEN_FILE_FAILURE(13)
[HailoRT] [error] CHECK_SUCCESS failed with status=HAILO_OPEN_FILE_FAILURE(13)
CHECK_EXPECTED failed with status=13
[HailoRT] [error] CHECK failed - Failed opening file, path: /apps/h8/gstreamer/general/cascading_networks/resources/tddfa_mobilenet_v1.hef
[HailoRT] [error] CHECK_SUCCESS failed with status=HAILO_OPEN_FILE_FAILURE(13)
[HailoRT] [error] Failed parsing HEF file
[HailoRT] [error] Failed creating HEF
[HailoRT] [error] CHECK_SUCCESS failed with status=HAILO_OPEN_FILE_FAILURE(13)
[HailoRT] [error] CHECK_SUCCESS failed with status=HAILO_OPEN_FILE_FAILURE(13)
[HailoRT] [error] CHECK_SUCCESS failed with status=HAILO_OPEN_FILE_FAILURE(13)
```

If this error occurs, you can set the environment variable: `export TAPPAS-WORKSPACE=path_to-tappas/tappas/`, where `path_to-tappas` is the path where the Tappas source code directory is located.

NpuDetectorLib Demo

1. Run the following command to download and compile NpuDetectorLib:

```
$ tar -xvf NpuDetectorLib.tar
$ cd NpuDetectorLib
$ cmake -H. -Bbuild -DSHOW_LABEL=ON -DBUILD_TESTER=ON
$ cmake --build build
```

2. Download the required resources:

```
$ ./get_sources.sh
```

3. Run the following command to demo target recognition:

- \$./build/tests/TestExecutable -i /dev/video4 -m models/yolov8s_nms.json -a yolov8_nms
- \$./build/tests/TestExecutable -i /dev/video4 -m models/yolov8s_pose.json -a yolov8_pose

5. Appendix

Click to [buy USB Camera](#).

