PROPHESEE **★ KV260 Starter Kit Manual** Search docs **□** kv260-psee **⊞** Overview **☐ Quick Start Setting up the Board and Application Deployment H** Tutorials **H** Architecture Repositories **Related Tutorials** Support

» kv260-psee » Setting up the Board and Application Deployment METAVISION INTELLIGENCE **Setting up the Board and Application Deployment** Introduction This document shows how to set up the board and how to run Prophesee applications. • Warning Prophesee Linux image does not start any windowing system. Hence, using the universal asynchronous receiver-transmitter (UART) interface is mandatory to launch the application. **Setting up the SD Card Image** Download Prophesee Kria™ Starter Kit Linux image and flash it into a microSD card (minimum 16 GB). • Note If you don't have access to this Knowledge Center page, contact support@prophesee.ai asking for a KC account and a specific access to the Kria Starter Kit page. Please follow AMD guide "Setting up the SD Card Image" to flash it with Prophesee's image. Once done, continue with this tutorial. **Hardware Setup** Let's see how to connect all the hardware interfaces required by Prophesee applications. Tip Before proceeding, we advise you to check AMD Kria KV260 Getting Started. Pay special attention to those sections: • Connecting Everything Booting your Starter Kit[™] Raspberry Pi J4 Camera Micro-USB J2 UART/JTAG Connector J11 **PMOD** MicroSD J3 JTAG DS1-DS6 Power Status LEDs 0 J8 (::::::::) HILL IAS Connector SOM Module SW2 Reset Button with Fansink • SW1 Firmware •• Update Button J7 IAS Connector 0 DS36 J13 PS Status LED Fan Power DS35 Heartbeat LED DS34 J12 PS Done LED DC Jack J10 U44 and U46 J5 RJ45 4x USB3.0 HDMI Ethernet DisplayPort X24750-022221 Prophesee sensor module To use a Prophesee event-based sensor with the Kria KV260 Starter Kit, you need to connect the sensor assembly kit flex to connector J9 on the Kria board (Raspberry Pi Camera Connector). Make sure the Kria board is turned off before connecting the module. **1** Tip Use the flex stiffener as a reference to ensure port connections. • It should be facing the SD card slot on the Kria board side. • It should be facing up on the Prophesee RPi sensor assembly kit. **UART** Some applications require using command line through a universal asynchronous receivertransmitter (UART), so let's configure it. Before powering on the Kria Starter Kit, connect a micro-USB cable from the J4 connector to your laptop. Run sudo dmesg | grep ttyUSB on your computer to check which serial port should be used: user@laptop:~\$ sudo dmesg | grep ttyUSB [275376.826662] usb 3-1: FTDI USB Serial Device converter now attached to ttyUSB0 [275376.828685] usb 3-1: FTDI USB Serial Device converter now attached to ttyUSB1 [275376.830614] usb 3-1: FTDI USB Serial Device converter now attached to ttyUSB2 [275376.832608] usb 3-1: FTDI USB Serial Device converter now attached to ttyUSB3 Four serial devices should appear as newly attached. The second port corresponds to the UART, which in the example above is ttyUSB1. You can then start minicom or any terminal emulator that can connect to a serial port and connect to serial device ttyUSB1: user@laptop:~\$ sudo minicom -D /dev/ttyUSB1 The communication parameters should be as follows: • 115200 baud • 8 data bits No parity • 1 stop bit No hardware or software flow control • Caution Please make sure to disable hardware or software flow control within the serial communication program. You can launch minicom with the -s option to have access to the Configuration menu and select "Serial Port Setup": Serial Device : /dev/ttyUSB1 B - Lockfile Location : /var/lock Callin Program - Callout Program Bps/Par/Bits : 115200 8N1 F - Hardware Flow Control : No G - Software Flow Control : No RS485 Enable RS485 Rts On Send : No J - RS485 Rts After Send : No - RS485 Rx During Tx : No . - RS485 Terminate Bus : No M - RS485 Delay Rts Before: 0 N - RS485 Delay Rts After: 0 Change which setting? Now, to power on the starter kit, connect the power supply to the AC plug. The power LEDs will illuminate and the board will boot up. Once booting is complete, a login prompt will appear in the terminal emulator. Use root for both the username and password. **Ethernet** It can be useful to be able to connect to the Kria board over Ethernet (e.g. to launch Metavision Viewer via SSH as explained below, to launch the full Active Marker demo). To do so, plug an RJ45 cable into the J10 connector and configure the connection from your minicom prompt: Kria board connected over a network The board should be assigned an IP through DHCP. Use the UART connection to get the IP: root@xilinx-kv260-starterkit-20222:~# ip addr dev eth0 Kria board connected directly to a PC Use the UART connection to assign a local IP address: root@xilinx-kv260-starterkit-20222:~# ip addr add 192.168.42.1/24 dev eth0 Configure the wired connection on your host PC, disabling the IPv6 and setting the IPv4 in manual mode as below: Kria Cancel Apply Security Details Automatic (DHCP) Link-Local Only IPv4 Method Manual Disable Shared to other computers Addresses 255.255.255.0 192.168.42.12 Ē DNS Automatic **Display** If you don't plan to connect to the board via SSH, you should connect a monitor to handle the display. It can be connected via either HDMI (J5) or DisplayPort (J6). **Loading Application Firmware** Starting here, the commands launched on the Kria board can be launched either via minicom or via SSH (e.g. ssh -X root@192.168.42.1). The Prophesee interface is already installed on the SD card and a script is provided to load it and configure the video pipeline: root@xilinx-kv260-starterkit-20222:~# /usr/bin/load-prophesee-kv260-<sensor>.sh • Note You can ignore any warnings or errors not related to Prophesee. Just make sure the output ends with: "prophesee-kv-260-imx636: loaded to slot 0". This confirms the process completed successfully. For more details on what the script does, you can check the Setting up the Video Pipeline tutorial. **Powering up the Sensor** Power up the sensor so that its registers can be accessed: root@xilinx-kv260-starterkit-20222:~# echo on > /sys/class/video4linux/v4l-subdev3/device/power/

BUY PRODUCTS PROPHESEE CONTACT

• Use the UART connection via minicom: Make sure you have a monitor plugged to the Kria board • From your minicom prompt, start an X server and run Metavision Viewer using the monitor as display:

the video4linux (V412) interface, there are 2 options:

using the keyboard of the host PC.

• Use the Ethernet connection via SSH:

explained above

raw events in hex format. For exemple:

• Note

or:

• Warning

Previous

© Copyright 2025, Prophesee.

root@xilinx-kv260-starterkit-20222:~# Xorg &

Running Metavision Viewer

• From your laptop shell prompt, connect to the board via SSH and launch Metavision Viewer: user@laptop:~\$ ssh -X root@192.168.42.1 root@xilinx-kv260-starterkit-20222:~# V4L2_HEAP=reserved V4L2_SENSOR_PATH=/dev/v4l-sub(

allows to access features like dumping a RAW file by pressing "space" key.

Note that in this configuration, you can send keyboard events to the application which

When connected to the Kria board, tools like yavta or v412 can also be used as well to capture

To run Metavision Viewer on the Kria board and see how it can communicate with sensor through

root@xilinx-kv260-starterkit-20222:~# DISPLAY=:0.0 V4L2_HEAP=reserved V4L2_SENSOR_PATH:

Note that in this configuration, you can't send directly keyboard events to the application

• Make sure you connected your Kria board to your laptop via Ethernet and configured it as

root@xilinx-kv260-starterkit-20222:~# yavta --capture=100 --nbufs 32 --file /dev/video0 root@xilinx-kv260-starterkit-20222:~# v4l2-ctl --stream-mmap --stream-count=100 --stream-to=f

Fore more details on the parameters, use the command yavta -h or v4l2-ctl -h

cases, it's important to monitor the camera's event rate to avoid potential bottlenecks. To set some camera features like ROI, biases, camera mode and so on, you can use a camera config file in JSON format. In that case, launch metavision_viewer with the option -j <camera_config_file>.json : root@xilinx-kv260-starterkit-20222:~# DISPLAY=:0.0 V4L2_HEAP=reserved V4L2_SENSOR_PATH=/dev/v4l-

A template of a camera config file can be downloaded here and then be edited manually. You can

also find some examples of usage in the Active Markers application.

Beware that the recording speed may be constrained by the SD card's write speed. The rate at

which data is written to the SD card could be slower than the sensor's data output. In such

Next Steps For the next steps, you might be interested in setting up and running the Active Markers application or editing kria applications.

Next **②**

PROPHESEE **★** KV260 Starter Kit Manual » kv260-psee » Active Markers Application METAVISION INTELLIGENCE BUY PRODUCTS PROPHESEE CONTACT Search docs □ kv260-psee **Active Markers Application ①** Overview Introduction **Unick Start ☐ Tutorials** This document shows how to start the Active Markers demo provided with the starter kit. The Setting up the video pipeline content is an adaptation of the existing documentation available in the Metavision SDK online Building the Hardware Design Using documentation. Vivado **Active Markers Application Required Material** Edit kria applications **H** Architecture The starter kit includes an Active Marker LED board, powered via USB and pre-programmed with **H** Repositories the necessary firmware for this application. **Related Tutorials** Support • Note If the LED board does not turn on when plugged into the USB port, try using one of the two black buttons on the backside to turn it on. **General Principle** The Active Markers use-case exploits the high speed capabilities of Prophesee Metavision sensors. The events generated by the sensor's high speed pixels enables to detect modulated light at high frequency, distinguish the IDs of various LEDs and compute the 6 degrees of freedom pose of the object to which the LEDs are attached.

Demodulation 3D Rendering Camera & tracking Events stream IDs 2D tracks 6 DOF pose Camera The camera is placed in front of the modulated LEDs. The focus should be done prior to that operation (with Metavision Viewer, with target objects described in this procedure.). Since the LEDs turn on very quickly, the sensor biases are configured to only generate events in response to blinking lights. To do so, the bias hpf (high pass filter) is set to a high value. We provide a configuration file that allows the camera to be set to the appropriate operating point. **Demodulation & Tracking** The LEDs on the marker board are programmed to emit unique time patterns, with each LED transmitting a time pattern that encodes its specific ID. More details on the modulation itself can be found in this section of the SDK documentation. By default, the LEDs on the supplied active marker board emit the IDs 20, 21, 22, 23, 24, 25, 26, 27 at the positions shown in the following figure: The demodulation software processes the IDs by referring to the active marker configuration file, which is also provided. The decoding of the events is integrated with LED tracking, allowing for a continuous estimation of their 2D coordinates. **Pose Estimation** With the 2D position of the markers, the knowledge of their physical position on the board and the camera calibration, we compute the absolute pose of the marker with respect to the camera. This is done using the Embedded application only. This 6 degrees of freedom pose is the solution to the Perspective-n-Point problem defined with the LEDs positions. • Warning This stage requires a calibrated camera, with estimated intrinsics and distortion model. The calibration process is detailed in the intrinsics calibration section of the SDK documentation. This is currently an unsupported feature with the Kria Starter Kit and will only be made available in a future release. We provide a canonical calibration file that gives a rough calibration model estimate for the 5mm lens provided with the starter kit. The purpose is to enable our customer to have an end-to-end experience all the way up to the visualization of 3D pose, but this does affect the precision of the pose estimation. **3D Rendering (Full Demo)** From the 6 DOF pose we render the marker in a virtual scene for illustration purposes. This last stage is meant to provide an example of a usage of the pose estimate. In this instance, it's for rendering purposes, but the pose could also be used as an input device in a human-machine interface. • Note The 3D rendering on the Kria board is a basic, non-real time implementation. It can be optimized to run directly on the Kria board, but for the time being the rendering is done on a laptop connected to it. **Running the Embedded Application Only** First, let's run the embedded-only application (metavision_embedded_active_marker_3d_tracking) that runs the Active Markers 2D tracking. Before launching the application, the FPGA and camera must be initialized. Connect to the Kria board (via minicom or SSH) and launch the following commands: cd /opt/metavision/embedded_active_marker_3d_tracking/embedded_active_marker chmod +x init_camera_<sensor>.sh ./init_camera_<sensor>.sh Now we recommend to do the camera focus by launching the following command: • If you use UART connection via minicom: killall -9 Xorg && sleep 1 Xorg & DISPLAY=:0.0 V4L2_HEAP=reserved V4L2_SENSOR_PATH=/dev/v4l-subdev3 metavision_viewer • If you use ethernet connection via SSH: V4L2_HEAP=reserved V4L2_SENSOR_PATH=/dev/v4l-subdev3 metavision_viewer This should show the Metavision Viewer window on the display connected to the board. Adjust the optic until the image is as sharp as possible. Now, launch the metavision_embedded_active_marker_3d_tracking application using the launch_<sensor>.sh cd /opt/metavision/embedded_active_marker_3d_tracking/embedded_active_marker chmod +x launch_<sensor>.sh ./launch_<sensor>.sh If you move the LED board in front of the camera, you should see that the LEDs are displayed and detected. **Running the Full Demo** The full demo allows to have the 3D rendering of the tracking on the host PC: Metavision-ActiveMarker3dTracking First, download the source code archive available at the following page. • Note If you don't have access to this Knowledge Center page, contact support@prophesee.ai asking for a KC account and a specific access to the Kria Starter Kit page. The archive contains the following files: metavision_active_marker_3d_tracking-openeb_v5.0.0 — CMakeLists.txt - metavision_active_marker_3d_tracking metavision_embedded_active_marker_3d_tracking

```
└─ target
Some configurations files are delivered in the folder utils:
  Camera config files:
                                       Contains the description of the camera configuration.
  camera_config_<sensor>.json
                                       For this application a specific set of biases is used.
  Active marker config file:
                                       Contains the description of the marker used.
  active_marker_config.json
                                       The IDs of LEDs as well as their physical location on the board.
                                       Contains the projection model of the camera used.
  Camera calibration files:
                                       Comes from a calibration that estimates intrinsics matrix
  camera_calibration_<sensor>.json
                                       and distortion coefficients.
  3D Scene files:
                                       Contains the 3D content of an industrial scene in order to visualize
  "scene" folder
                                       the camera's poses in a virtual environment.
To run the full demo on Ubuntu 22, follow those steps:
1. Install dependencies:
     sudo apt update
     sudo apt install -y software-properties-common gpg ca-certificates curl unzip cmake build-ess
     sudo apt install -y libboost-program-options-dev libeigen3-dev libceres-dev
     sudo apt install -y libogre-1.12-dev libimgui-dev libfreetype-dev
     # Install Sophus
     curl -L "https://github.com/strasdat/Sophus/archive/1.22.10.zip" --output /tmp/Sophus.zip
     unzip /tmp/Sophus.zip -d /tmp/
     cmake -S /tmp/Sophus-1.22.10 -B /tmp/Sophus-1.22.10
     sudo cmake --build /tmp/Sophus-1.22.10 --target install --parallel `nproc`
```

3. Extract the contents of the ZIP archive mentioned above and open a terminal in the unzipped

sudo cmake --build /tmp/active_marker_build --target install --parallel `nproc`

You should see the 3D rendering as mentioned at the beginning of the section

cmake -S metavision_active_marker_3d_tracking/ -B /tmp/active_marker_build -DCMAKE_BUILD_TYPE

This script connects to the Kria board via SSH using the root user. A root password must be

As explained before, the embedded application is built and integrated into the image directly

set for this to work. If not, first connect to the Kria board and set a password using sudo passwd

4. Compile the sample metavision_active_marker_3d_tracking as shown below:

utils/scripts/host/launch_<sensor>.sh <kria.IP.address>

- metavision_extract

- plugins.cfg.in

- resources.cfg.in

- scene

- scripts — host

— decor.mesh

— config_files

— active_marker_config.json

— camera_config_genx320.json camera_config_imx636.json

— active_marker_target.mesh

— industrial_scene.material

industrial_scene.scene

— camera_calibration_genx320.json — camera_calibration_imx636.json

⊢ ...

README.md

— utils

2. Install OpenEB 5.0

5. Launch the script:

• Note

root .

through Petalinux.

• Note

artifacts.

Two options are possible:

Custom Scene

active_marker_config.json file.

match the new board model.

environment: e.g., a living room:

<PETALINUX_PROJECT>:

folder.

Follow the OpenEB install guide

Let's see how to rebuild it from source in order to customize the metavision_active_marker_3d_tracking application. 1. Install Petalinux 2022.2 in <PETALINUX_TOOLS>

2. Clone the repository https://github.com/prophesee-ai/petalinux-projects in

3. Download the metavision_active_marker_3d_tracking source code at this page.

for a KC account and a specific access to the Kria Starter Kit page.

user/recipes-vision/metavision-active-marker-3d-tracking/files/

tracking/metavision_active_marker_3d_tracking-openeb_v5.0.0.bb.

git clone git@github.com:prophesee-ai/petalinux-projects.git -b kv260-2022.2

Build the Embedded Application from Sources

• Note It is important to keep the original file name metavision_active_marker_3d_tracking- openeb_v5.0.0.tar.gz as it is referenced in the recipe.

Otherwise, you should rename as well the reference of the file name <SRC_URI> inside the

Similar step should be performed if you generated a new .xsa file to configure your FPGA, as

explained in the page Tutorials/Build the Vivado Design. In that case, the recipe file to check is

<PETALINUX_PROJECT>/project-spec/meta-user/recipes-firmware/prophesee-kv260-<sensor>/prophesee-kv260-

<sensor>_1.0.0.bb . Modify the file to get the xsa from the "files" folder instead of the github

recipe file <PETALINUX_PROJECT>/project-spec/meta-user/recipes-vision/metavision-active-marker-3d-

4. If you want to customize the application, extract the archive and apply your changes in the

source code. Then create a new archive and copy it in petalinux-projects/project-spec/meta-

If you don't have access to this Knowledge Center page, contact support@prophesee.ai asking

Alternatively, you can also check the section edit kria applications. 5. Run Petalinux tools to build the system: cd <PETALINUX_PR0JECT> source <PETALINUX_TOOLS>/settings.sh petalinux-build 6. Generate a microSD card image: petalinux-package --wic --bootfiles "ramdisk.cpio.gz.u-boot,boot.scr,Image,system.dtb,system-Then you can use your new generated image to flash your SD card. **Custom Marker** Different active marker devices can be used as long as the modulation follows the same principle as explained in the Metavision documentation page for active markers.

The IDs of the LEDs can be arbitrarily changed, as well as their position.

• Reprogram the Active Marker board: contact Prophesee customer support

(support@prophesee.ai[™]) to obtain a different firmware. Then, update the IDs in the

• Design a custom Active Markers PCB: follow the suggested modulation to design your own

active markers PCB. In this case, update both the IDs and the positions in the configuration to

We offer an additional 3D scene configuration that allows users to run the app in a different virtual

The scene files can be found in the Active Marker source code archive, inside the folder utils as mentioned above. **Going Further** To go further with another application, you can also try the Event ML application developed by LogicTronix in coordination with Prophesee and AMD. The application demonstrates Yolov7 and

Yolov4-tiny ML models for Vitis Al. The source code of the demo is available here: https://github.com/LogicTronixInc/Kria-Prophesee-Event-VitisAI Next **②**

Previous © Copyright 2025, Prophesee.