1. Creating Petalinux Image from BSP :

Create a Petalinux project from kv260-psee-vitis-ai-2022-2.bsp - BSP provided in the repository.

```
petalinux-create -t project -s kv260-psee-vitis-ai-2022-2.bsp -n
kv260-psee-vitis-ai-2022-2
cd kv260-psee-vitis-ai-2022-2
petalinux-config -siletconfig
petalinux-build
```

After build completes, create wic image running following command:

```
petalinux-package --wic --images-dir images/linux/ --bootfiles
"ramdisk.cpio.gz.u-boot,boot.scr,Image,system.dtb,system-zynqmp-sck-kv-g-re
vB.dtb" --disk-name "mmcblk1"
```

This will create petalinux-sdimage.wic in kv260-psee-vitis-ai-2022-2/images/linux directory.

Burn the petalinux-sdimage.wic in SD card using tools like balena etcher.

Next boot the SD card in KV260 board and follow following steps from KV260 board petalinux terminal.

2. Copying Hardware overlay files

From repository copy
Kria-Prophesee-Event-VitisAI/Kria-Petalinux/Firmware/kv260-psee-dpu firmware folder and copy to /lib/firmware/Xilinx folder.

After copying file view the available hardware overlays using xmutil listapps and load the **kv260-psee-dpu** overlay running xmutil tool. Here are the commands for unloading and loading the hardware overlay.

```
sudo xmutil unloadapp
sudo xmutil loadapp kv260-psee-dpu
```

For building and running event camera ML application in Petalinux one need OpenEB SDK and Vitis AI library. Vitis AI library is already installed in Petalinux image through petalinux build process, while OpenEB SDK need to be manually installed. So next OpenEB SDK is build and installed in petalinux.

3. OpenEB build in Petalinux 2022.2

OpenEB development library is required for developing application based on Metavision SDK. So it is necessary to build OpenEB library in Petalinux for building event camera ML application. Follow the following steps to build and install OpenEB in petalinux:

3.1 Enabling packages in Petalinux rootfs config:

In petalinux build environment run `petalinux-config -c rootfs` and enable **wget unzip curl git protobuf opencv-dev.**
Next add following line in project-spec/conf/petalinuxbsp.conf:

```
IMAGE_INSTALL:append += "simd \
                  cmake \
                  hdf5 \
```

3.2 Install dependencies and python packages

```
$ sudo dnf -y install python3-pip python3-distutils
$ sudo dnf install libglew-dev
$ python3 -m pip install pip --upgrade
$ python3 -m pip install "opencv-python==4.5.5.64" "sk-video==1.1.10"
"fire==0.4.0" "numpy==1.23.4" "h5py==3.7.0" pandas scipy
$ python3 -m pip install "numba==0.56.3" "profilehooks==1.12.0"
"pytorch_lightning==1.8.6" "tqdm==4.63.0" "kornia==0.6.8"
```

Building and Installing pybind

```
$ wget https://github.com/pybind/pybind11/archive/v2.6.0.zip
$ unzip v2.6.0.zip
$ cd pybind11-2.6.0/
$ mkdir build && cd build
$ cmake .. -DPYBIND11_TEST=OFF
$ cmake --build .
$ sudo cmake --build . --target install
```

Building and installing glfw

```
$ wget https://github.com/glfw/glfw/releases/download/3.3.8/glfw-3.3.8.zip
$ unzip glfw-3.3.8.zip
$ cd glfw-3.3.8/
```

```
$ cmake -G "Unix Makefiles"
$ make
$ sudo make install
```

Building and Installing hdf5

```
$ wget
https://support.hdfgroup.org/ftp/HDF5/releases/hdf5-1.13/hdf5-1.13.2/src/hd
f5-1.13.2.tar.gz
$ gunzip  hdf5-1.13.2.tar.gz
$ tar -xvf  hdf5-1.13.2.tar
$ cd hdf5-1.13.2
$ ./configure --enable-cxx
$ make -j 4
$ make install
$ sudo cp bin/* /usr/bin/
$ sudo cp include/* /usr/include/
$ sudo cp lib/* /usr/lib
```

Alos copy FindHDF5.cmake at /usr/share/cmake-3.21/Modules

Building and Installing OpenEB:

```
$ git clone https://github.com/sanamshakya/openeb.git
$ cd openeb/
$ mkdir build && cd build
$ cmake .. -DBUILD_TESTING=OFF
$  cmake --build . --config Release -- -j 4
$  sudo cmake --build . --target install
```

1. Building event cam histo viewer application

Change to the event cam histo viewer application by using cd command in terminal.
Before building the application, load the environment variable by running following commands:

```
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/lib
export HDF5_PLUGIN_PATH=$HDF5_PLUGIN_PATH:/usr/local/lib/hdf5/plugin
```

Next build the application by running following commands:

```
mkdir build
cd build
cmake --build . --config Release
```

Now run the application by running the following command:
For file input:

```
./cd_histo_viewer -i <input raw event file>
```

For camera input:

```
# source Kria-Prophesee-Event-VitisAI/Kria-Ubuntu/scripts/load.sh
# echo on > /sys/class/video4linux/v4l-subdev<X>/device/power/control
# MV_LOG_LEVEL=TRACE V4L2_HEAP=reserved
V4L2_SENSOR_PATH=/dev/v4l-subdev<X> ./cd_histo_viewer
```