OAK-FFC-6P

Overview

The **OAK-FFC-6P** baseboard has **6 FFC interfaces** which allows for **six 2-lane MIPI** camera modules. It's a RVC3-based device and uses the OAK-SoM MAX.

To see **which cameras are compatible with this OAK FFC baseboard**, see the guide here: OAK FFC camera modules. Some of camera modules have a M12 mount, so you can use different lenses to get **custom FoV** (with wide or narrow FOV M12 lenses).

This board is also compatible with RPi camera interface. For that you will need a FFC from Arducam, which converts 26-pin Luxonis camera pinout to 22 pin RPi camera pinout.

In addition, IMU (over SPI) sensor is also added to this OAK FFC board.

RVC3 inside

This OAK device is built on top of the RVC3. Main features:

- **3.0 TOPS** for **AI** with INT8 quantization support
- Quad-core ARM **A53** @ 1.5GHz, running Yocto Linux, acting as a host computer
- **Imaging**: ISP, max 6 cameras, 500 MP/s HDR, 3A
- **Run any AI model**, even custom architectured/built ones - models need to be converted.
- **Cloud platform** - Robothub - connectivity out-of-the-box
- On-device SLAM / VIO support

- **Encoding**: H.264, H.265, MJPEG - 4K/75FPS, **Decoding**: 4K/60FPS
- **Computer vision:** warp/dewarp, resize, crop via ImageManip node, edge detection, feature tracking. You can also run custom CV functions
- **Stereo depth** perception with filtering, post-processing, RGB-depth alignment, and high configurability
- **Object tracking**: 2D and 3D tracking with ObjectTracker node

Developing with the OAK FFC

After connecting cameras to the baseboard, you can use the utilities/cam_test.py script to quickly test whether cameras are working as expected. By default, it will try to run 2x mono cameras on 2-lane mipi ports B (left) and C (right) and 2x color cameras on port A (rgb) and D (4-lane mipi ports).

If you have different cameras connected, you can specify which camera types to use with the `--cameras` argument. For example, if you have 3x mono cameras connected to ports A, B, and C, you can run the following command:

```
python3 cam_test.py --cameras rgb,m right,m left,m
```

Similarly, to add such configuration into your script you can use the following code:

```
cam_a = pipeline.create(dai.node.MonoCamera)
cam_a.setBoardSocket(dai.CameraBoardSocket.CAM_A) # Same as CameraBoardSocket.RGB
cam_a.setResolution(dai.MonoCameraProperties.SensorResolution.THE_400_P)

cam_b = pipeline.create(dai.node.MonoCamera)
cam_b.setBoardSocket(dai.CameraBoardSocket.CAM_B) # Same as CameraBoardSocket.LEFT

cam_c = pipeline.create(dai.node.MonoCamera)
cam_c.setBoardSocket(dai.CameraBoardSocket.CAM_C) # Same as CameraBoardSocket.RIGHT

cam_d = pipeline.create(dai.node.MonoCamera)
cam_d.setBoardSocket(dai.CameraBoardSocket.CAM_D)
```

Dimensions and Weight

- Width: 50mm
- Height: 33mm, without heatsink: 13mm
- Length: 70mm
- Weight: 69g, without heatsink: 33g

General information

- 6x 2-line MIPI camera interfaces (all of the CCMs supported in DepthAI can run on 2-lane MIPI, most of them also in full resolution)
- OAK-SoM MAX connected to the baseboard
- 5V power input via barrel jack
- USB 3.1 Gen 1 Type-C
- IMU support (BMI270 by default, footprint for BNO086)
- uSD card slot
- Onboard EEPROM
- Pads for DepthAI SoM 1.8V SPI
- Pads for DepthAI SoM 1.8V Aux Signals (I2C, UART, GPIO)
- PSRBS connector (5V, STROBE, RST, BOOT_SEL, FSIN_ALL, FSIN_STEREO)
- 1025 coin battery holder
- Design files produced with Altium Designer 22

Minimal and maximal perceiving distances of the camera

Minimal depth perceiving distance of the camera depends on mono camera FOV, resolution, baseline and stereo depth mode, more info is available on the Stereo Depth documentation.

Since this device has modular mono cameras, you can choose a custom stereo baseline (depending on how it is set up). When using OAK-FFC-OV9282, the formulas for min/max depth perceiving distances are:

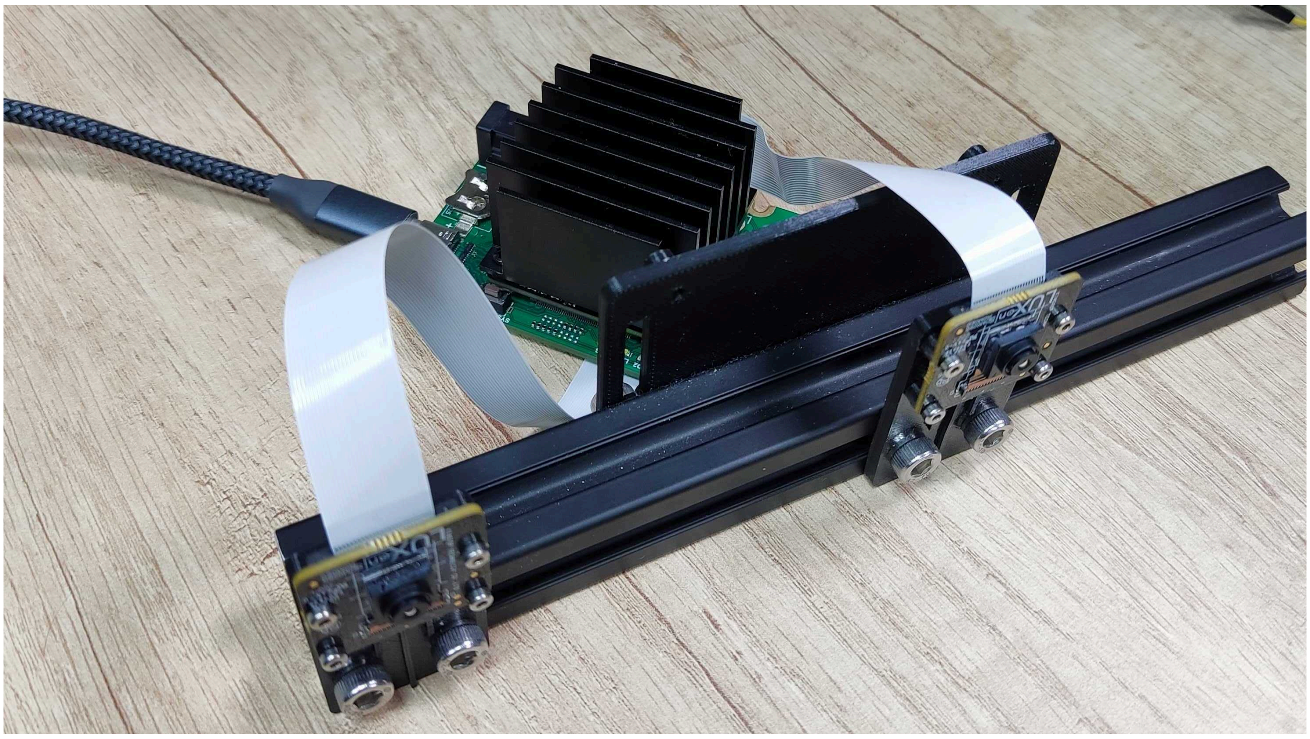
- Min distance (800P) = `882.5 * baseline / 95`
- Min distance (400P) = `441.25 * baseline / 95`
- Min distance with extended disparity (800P) = `882.5 * baseline / 190`

- Min distance with extended disparity (400P) = `max(441.25 * baseline / 190, 19.6)`
- Max perceivable distance (using subpixel) = `baseline/2 * tan((90 - 71.9/1280) * PI/180)`

For more information about the maximum distance see the Stereo Depth documentation.

Getting started

In this tutorial, we will be using OAK-FFC-6P with 2x OAK-FFC-OV9282 camera modules (9.75cm apart) on ports CAM_B and CAM_C. We are using OAK-FFC-Kit to mount the cameras. First we will install the correct depthai library version, then we will calibrate the stereo cameras, and finally we will check the results.



RVC3 library

As RVC3 is still in early access, you will need to install the correct version of depthai library on the host computer. You can do that by checking out on `rvc3_develop` on the **depthai-python** repository and running `python example/install_requirements.py`.

Now, let's check the cameras streams first. As we have OV9282 on ports B and C, we will run the following command:

```
python cam_test.py -cams left,m right,m

Enabled cameras:
    left : mono
    right : mono
DepthAI version: 2.19.1.0.dev+c5fd7c993a3d446f6b9fdb54653cb49c7523534f
DepthAI path: path/to/depthai.cp39-win_amd64.pyd
    Connected cameras: [{socket: LEFT/CAM_B, sensorName: OV9282, width: -1, height:
-1, orientation: AUTO, supportedTypes: [], hasAutofocus: 0, name: CAM-SA}, {socket:
RIGHT/CAM_C, sensorName: OV9282, width: -1, height: -1, orientation: AUTO,
supportedTypes: [], hasAutofocus: 0, name: CAM-SB}]
    -socket LEFT  : OV9282   -1 x   -1 focus:fixed -
    -socket RIGHT : OV9282   -1 x   -1 focus:fixed -
USB speed: UNKNOWN
IR drivers: []
Cam:      left          right    [host | capture timestamp]
FPS:  24.61| 21.96  24.31| 21.44
```

And we get the streams from both cameras.

RVC3 SSH into ARM

RVC3 also has quadcore ARM, so it's also possible to SSH into the device and install the same library (so `rvc3_develop`) there. To SSH into the device, please see gdocs here. Note that because the device doesn't have internet access, you'd need to download and copy the library from the host computer to the device.

Calibration

Now, let's calibrate the cameras. First, we need to be on the `rvc3_develop` branch on the **depthai** repository. We need to have all the requirements installed, and also make sure we still have the correct depthai library (as we did in the previous step). Following the Calibration docs here, I ran the calibrate.py file in the depthai repository:

```
python calibrate.py -s 5 -brd board.json -db -cd 0
```

And the board.json file is the following:

```json
{
    "board_config":
    {
        "name": "OAK-FFC-6P",
        "revision": "R0M0E0",
        "cameras":{
            "CAM_B": {
                "name": "left",
                "hfov":  77,
                "type": "mono",
                "extrinsics": {
                    "to_cam": "CAM_C",
                    "specTranslation": {
                        "x": -9.75,
                        "y": 0,
                        "z": 0
                    },
                    "rotation":{
                        "r": 0,
                        "p": 0,
                        "y": 0
                    }
                }
            },
            "CAM_C": {
                "name": "right",
                "hfov":  77,
                "type": "mono"
                }
        },
        "stereo_config":{
            "left_cam": "CAM_B",
            "right_cam": "CAM_C"
        }
    }
}
```

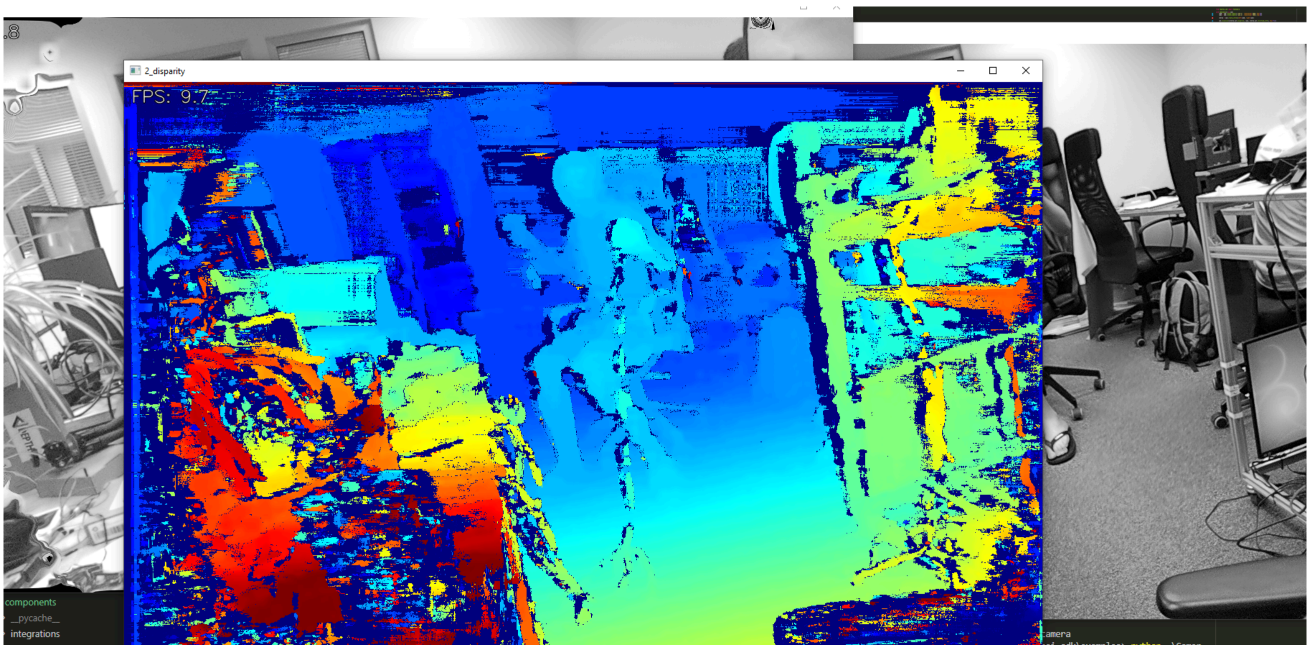
After 39 images at different angles and orientations, I got an average epipolar error of 0.14 pixels. Not great, not terrible, but will do for the tutorial. Now let's try our the stereo depth.

Testing the stereo results

We will be using SDK to test the results. Here's a simple script we can use to test the results:

```python
from depthai_sdk import OakCamera

with OakCamera() as oak:
    camb = oak.create_camera('camb,m', resolution='800p',fps=10)
    camc = oak.create_camera('camc,m', resolution='800p',fps=10)

    stereo = oak.create_stereo(left=camb, right=camc)

    oak.visualize([stereo.out.disparity, camc, stereo.out.rectified_left], fps=True)

    oak.start(blocking=True)
```

Note that depthai_sdk library should be installed from the `rvc3_develop` branch on depthai repository. After running the script, we get this result:



Not bad, but at the edges you can see some circular/warp artifacts, which is is caused due to a) not great calibration, b) camera distortion model we are using.

Hardware setup

The OAK-FFC-6P accepts 5V (+/-10%) from a 5.5m x 2.5mm barrel jack, and interfaces to a host via USB 3.1 Gen1 Type-C. With cameras and the OAK-SoM MAX, total power consumption usually stays below the 1.5 A specification, Type-C power of 1.5 A or greater is recommended.

Interfacing with the OAK-SoM MAX is also possible with OAK-FFC-6P connector pads QSPI, and AUX. These pads are designed for the Amphenol/FCI 20021121-00010T1LF or equivalent. Please refer to the schematics for pinout information.

The reset button soft resets the Luxonis DepthAI SoM only, SYS-RST button resets the power system of the DepthAI and PWR button is a GPIO button that can be used to turn off/on the SoM. BOOT button is meant to be used only for recovery purposes changing GPIO boot mode to eMMC recovery when the button is held on system power up.

The PG LED indicates 5V power is present and all outputs of the DC-DC converters are inside the specified nominal voltages. The LED1 indicates “BOOT” status of the DepthAI SoM. The LED2 is an “CONNECT” status LED which indicates that the OAK-SoM-Max is connected to the network/device.

FFC cables

For FFC cables we use Molex series 15166. Along with the OAK FFC board, we ship **26 pin count, same-sided, 152mm** cables (part number 151660281). If you would like to use **shorter/longer FFC cables**, you can get them here.

3D Models

- Board (PCBA) STEP files here

Files

- Altium Design Files
- Assembly Drawing
- Assembly Outputs
- Fabrication Drawing
- Fabrication Outputs
- Schematic

Got questions?

Head over to **Discussion Forum** for technical support or any other questions you might have.