Qualcomm Camera Subsystem driver

Introduction

This file documents the Qualcomm Camera Subsystem driver located under drivers/media/platform/qcom/camss.

The current version of the driver supports the Camera Subsystem found on Qualcomm MSM8916/APQ8016 and MSM8996/APQ8096 processors.

The driver implements V4L2, Media controller and V4L2 subdev interfaces. Camera sensor using V4L2 subdev interface in the kernel is supported.

The driver is implemented using as a reference the Qualcomm Camera Subsystem driver for Android as found in Code Aurora [1] [2].

Qualcomm Camera Subsystem hardware

The Camera Subsystem hardware found on 8x16 / 8x96 processors and supported by the driver consists of:

- 2 / 3 CSIPHY modules. They handle the Physical layer of the CSI2 receivers. A separate camera sensor can be connected to each of the CSIPHY module;
- 2 / 4 CSID (CSI Decoder) modules. They handle the Protocol and Application layer of the CSI2 receivers. A CSID can decode data stream from any of the CSIPHY. Each CSID also contains a TG (Test Generator) block which can generate artificial input data for test purposes;
- ISPIF (ISP Interface) module. Handles the routing of the data streams from the CSIDs to the inputs of the VFE;
- 1 / 2 VFE (Video Front End) module(s). Contain a pipeline of image processing hardware blocks. The VFE has different input interfaces. The PIX (Pixel) input interface feeds the input data to the image processing pipeline. The image processing pipeline contains also a scale and crop module at the end. Three RDI (Raw Dump Interface) input interfaces bypass the image processing pipeline. The VFE also contains the AXI bus interface which writes the output data to memory.

Supported functionality

The current version of the driver supports:

- Input from camera sensor via CSIPHY;
- Generation of test input data by the TG in CSID;
- RDI interface of VFE
    - Raw dump of the input data to memory.

      Supported formats:

        - YUYV/UYVY/YVYU/VYUY (packed YUV 4:2:2 - V4L2_PIX_FMT_YUYV / V4L2_PIX_FMT_UYVY / V4L2_PIX_FMT_YVYU / V4L2_PIX_FMT_VYUY);
        - MIPI RAW8 (8bit Bayer RAW - V4L2_PIX_FMT_SRGGB8 / V4L2_PIX_FMT_SGRBG8 / V4L2_PIX_FMT_SGBRG8 / V4L2_PIX_FMT_SBGGR8);
        - MIPI RAW10 (10bit packed Bayer RAW - V4L2_PIX_FMT_SBGGR10P / V4L2_PIX_FMT_SGBRG10P / V4L2_PIX_FMT_SGRBG10P / V4L2_PIX_FMT_SRGGB10P / V4L2_PIX_FMT_Y10P);
        - MIPI RAW12 (12bit packed Bayer RAW - V4L2_PIX_FMT_SRGGB12P / V4L2_PIX_FMT_SGBRG12P / V4L2_PIX_FMT_SGRBG12P / V4L2_PIX_FMT_SRGGB12P).
        - (8x96 only) MIPI RAW14 (14bit packed Bayer RAW - V4L2_PIX_FMT_SRGGB14P / V4L2_PIX_FMT_SGBRG14P / V4L2_PIX_FMT_SGRBG14P / V4L2_PIX_FMT_SRGGB14P).

    - (8x96 only) Format conversion of the input data.

      Supported input formats:

        - MIPI RAW10 (10bit packed Bayer RAW - V4L2_PIX_FMT_SBGGR10P / V4L2_PIX_FMT_Y10P).

      Supported output formats:

        - Plain16 RAW10 (10bit unpacked Bayer RAW - V4L2_PIX_FMT_SBGGR10 / V4L2_PIX_FMT_Y10).

- PIX interface of VFE
    - Format conversion of the input data.

      Supported input formats:

        - YUYV/UYVY/YVYU/VYUY (packed YUV 4:2:2 - V4L2_PIX_FMT_YUYV / V4L2_PIX_FMT_UYVY / V4L2_PIX_FMT_YVYU / V4L2_PIX_FMT_VYUY).

      Supported output formats:

        - NV12/NV21 (two plane YUV 4:2:0 - V4L2_PIX_FMT_NV12 / V4L2_PIX_FMT_NV21);
        - NV16/NV61 (two plane YUV 4:2:2 - V4L2_PIX_FMT_NV16 / V4L2_PIX_FMT_NV61).
        - (8x96 only) YUYV/UYVY/YVYU/VYUY (packed YUV 4:2:2 - V4L2_PIX_FMT_YUYV / V4L2_PIX_FMT_UYVY / V4L2_PIX_FMT_YVYU / V4L2_PIX_FMT_VYUY).
    - Scaling support. Configuration of the VFE Encoder Scale module for downscalling with ratio up to 16x.
    - Cropping support. Configuration of the VFE Encoder Crop module.
- Concurrent and independent usage of two (8x96: three) data inputs - could be camera sensors and/or TG.

Driver Architecture and Design

The driver implements the V4L2 subdev interface. With the goal to model the hardware links between the modules and to expose a clean, logical and usable interface, the driver is split into V4L2 sub-devices as follows (8x16 / 8x96):

- 2 / 3 CSIPHY sub-devices - each CSIPHY is represented by a single sub-device;
- 2 / 4 CSID sub-devices - each CSID is represented by a single sub-device;
- 2 / 4 ISPIF sub-devices - ISPIF is represented by a number of sub-devices equal to the number of CSID sub-devices;
- 4 / 8 VFE sub-devices - VFE is represented by a number of sub-devices equal to the number of the input interfaces (3 RDI and 1 PIX for each VFE).

The considerations to split the driver in this particular way are as follows:

- representing CSIPHY and CSID modules by a separate sub-device for each module allows to model the hardware links between these modules;
- representing VFE by a separate sub-devices for each input interface allows to use the input interfaces concurrently and independently as this is supported by the hardware;
- representing ISPIF by a number of sub-devices equal to the number of CSID sub-devices allows to create linear media controller pipelines when using two cameras simultaneously. This avoids branches in the pipelines which otherwise will require a) userspace and b) media framework (e.g. power on/off operations) to make assumptions about the data flow from a sink pad to a source pad on a single media entity.

Each VFE sub-device is linked to a separate video device node.

The media controller pipeline graph is as follows (with connected two / three OV5645 camera sensors):

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\admin-guide\media\[linux-master][Documentation][admin-guide][media]qcom_camss.rst, line 142)

Unknown directive type "kernel-figure".

```
.. kernel-figure:: qcom_camss_graph.dot
    :alt:   qcom_camss_graph.dot
    :align: center

    Media pipeline graph 8x16
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\admin-guide\media\[linux-master][Documentation][admin-guide][media]qcom_camss.rst, line 148)

Unknown directive type "kernel-figure".

```
.. kernel-figure:: qcom_camss_8x96_graph.dot
    :alt:   qcom_camss_8x96_graph.dot
    :align: center

    Media pipeline graph 8x96
```

Implementation

Runtime configuration of the hardware (updating settings while streaming) is not required to implement the currently supported functionality. The complete configuration on each hardware module is applied on STREAMON ioctl based on the current active media links, formats and controls set.

The output size of the scaler module in the VFE is configured with the actual compose selection rectangle on the sink pad of the 'msm_vfe0_pix' entity.

The crop output area of the crop module in the VFE is configured with the actual crop selection rectangle on the source pad of the

'msm_vfe0_pix' entity.

Documentation

APQ8016 Specification: https://developer.qualcomm.com/download/sd410/snapdragon-410-processor-device-specification.pdf Referenced 2016-11-24.

APQ8096 Specification: https://developer.qualcomm.com/download/sd820e/qualcomm-snapdragon-820e-processor-apq8096sge-device-specification.pdf Referenced 2018-06-22.

References

[1] https://source.codeaurora.org/quic/la/kernel/msm-3.10/

[2] https://source.codeaurora.org/quic/la/kernel/msm-3.18/