The Road to the Mainline ZynqMP VCU Driver

FOSDEM ’21

Michael Tretter – m.tretter@pengutronix.de

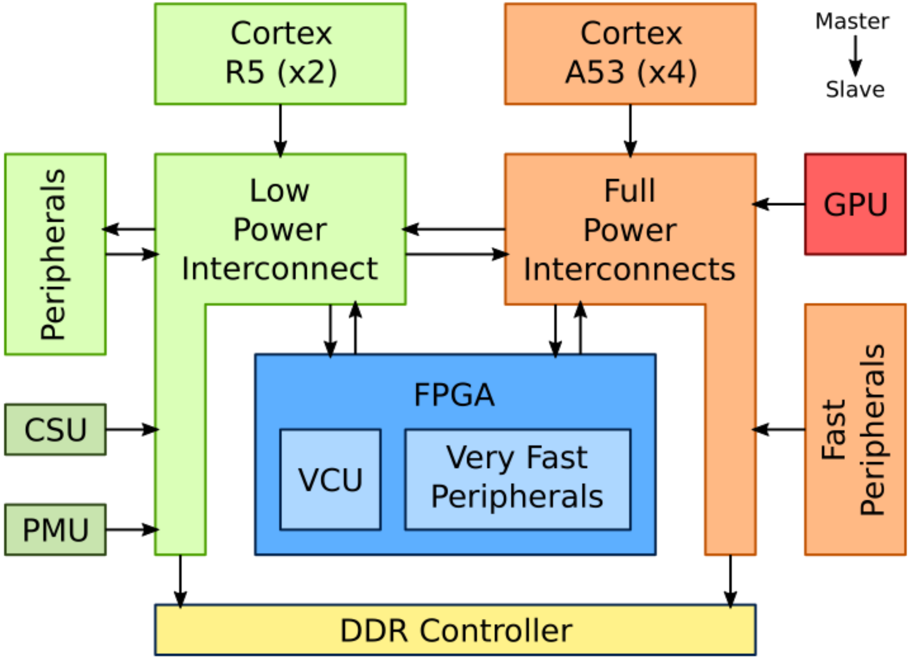
Agenda

- Xilinx Zynq® UltraScale+™ MPSoC
- H.264/H.265 Video Codec Unit
- Video Encoders in Mainline Linux
- VCU Mainline Driver: Allegro
- A Glimpse into the Future

Xilinx Zynq® UltraScale+™ MPSoC

ZynqMP Platform Overview

- Luca Ceresoli: ARM64 + FPGA and more: Linux on the Xilinx ZynqMP
- https://archive.fosdem.org/2018/schedule/event/arm64_and_fpga



ZynqMP Mainline Status

- Mainline Linux just works, e.g., on ZCU104 Evaluation Kit
- U-Boot, Barebox, FSBL
- Sometimes more reliable with Xilinx downstream
- Xilinx is actively mainlining their drivers

Make Sure that Your ZynqMP has a VCU

ZU # E V

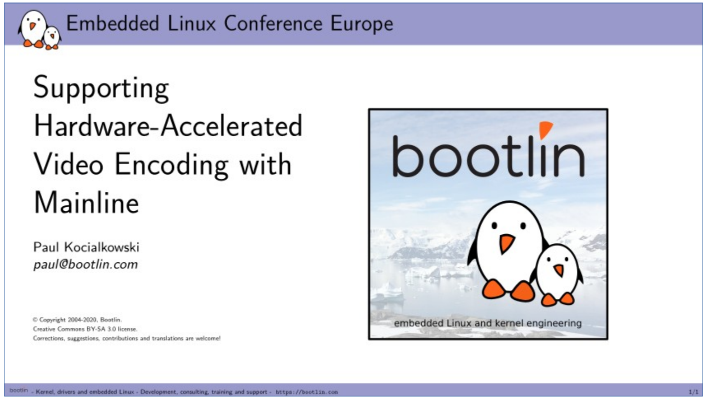
- ZU: Zynq Ultrascale+
- #: Value Index
- C/E: Processor System Identifier
- G/V: Engine Type

Focus on Video Encoding

- VCU supports video decoding, as well
- Linux mainline driver only supports encoding
- Decoding might be focus in a future talk

Basic Video Encoding Knowledge Expected

- Paul Kocialkowski: Supporting Hardware-Accelerated Video Encoding with Mainline
- https://www.youtube.com/watch?v=S5wCdZfGFew



Embedded Linux Conference Europe

Supporting Hardware-Accelerated Video Encoding with Mainline

Paul Kocialkowski
paul@bootlin.com

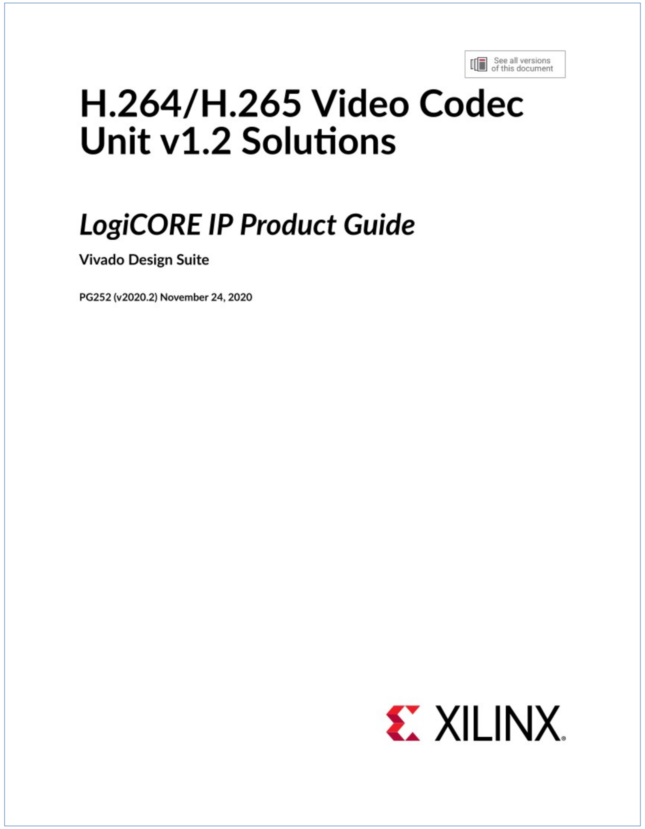
© Copyright 2004-2020, Bootlin.
Creative Commons BY-SA 3.0 license.
Corrections, suggestions, contributions and translations are welcome!

bootlin - Kernel, drivers and embedded Linux - Development, consulting, training and support - https://bootlin.com 1/1

H.264/H.265 Video Codec Unit

VCU: Documentation

- Hardware configuration
- Software usage
- Available on the Xilinx Website



See all versions of this document

H.264/H.265 Video Codec Unit v1.2 Solutions

LogiCORE IP Product Guide

Vivado Design Suite

PG252 (v2020.2) November 24, 2020

XILINX.

VCU: Features

The encoder engine is designed to process video streams using the **HEVC (ISO/IEC 23008-2 high-efficiency Video Coding)** and **AVC (ISO/IEC 14496-10 Advanced Video Coding)** standards. It provides complete support for these standards, including support for 8-bit and 10-bit color, Y- only (monochrome), 4:2:0 and 4:2:2 Chroma formats, up to **4K UHD at 60 Hz** performance.

(PG252, p. 14)

VCU: Integration into SoC

The VCU is an **integrated block in the programmable logic** (PL) of selected Zynq UltraScale+ MPSoCs with **no direct connections to the processing system** (PS), and contains encoder and decoder interfaces. The VCU also contains additional functions that facilitate the interface between the VCU and the PL.

(PG 252, p. 12)

VCU: Architecture

The encoder is **controlled by a microcontroller** (MCU) subsystem. VCU applications running on the APU use the Xilinx® **VCU Control Software** library API to interact with the encoder microcontroller. The **microcontroller firmware** [...] **is not user modifiable**.

(PG 252, p. 14)

VCU Control Software

The VCU Control Software is the lowest level software visible to VCU application developers. All VCU applications must use a Xilinx® provided VCU Control Software, directly or indirectly.

The VCU Control Software includes custom **kernel modules,** custom **user space library**, and the ctrlsw_encoder and ctrlsw_decoder **applications**. The **OpenMAX IL (OMX) layer** is integrated on top of the VCU Control Software.

(PG 252, p. 188)

User Space Library

- https://github.com/Xilinx/vcu-ctrl-sw
- License: MIT with the following exception

Use of the Software is limited solely to applications:

(a) running on a Xilinx device, or

(b) that interact with a Xilinx device through a bus or interconnect.

Under the Hood: Kernel/User Space Interface

- Device node: /dev/allegroIP
- Kernel driver handles opaque data structures
- User space fills opaque data structures
- Data structures contain physical addresses

Why not Use the Downstream Driver?

- Out-of-tree kernel module
- Non-standard user space interface
- Physical addresses exposed to user space
- Security issues of physical addresses
- No chance of mainlining
- Possibly unmaintained userspace

Possible Pitfalls

- Not using the Xilinx suggested way
- MCU firmware still required
- Unsupported and undocumented ABI

Video Encoders in Mainline Linux

Video Encoder APIs

- OpenMAX
- VDPAU
- VA-API
- Mesa + DRM
- V4L2

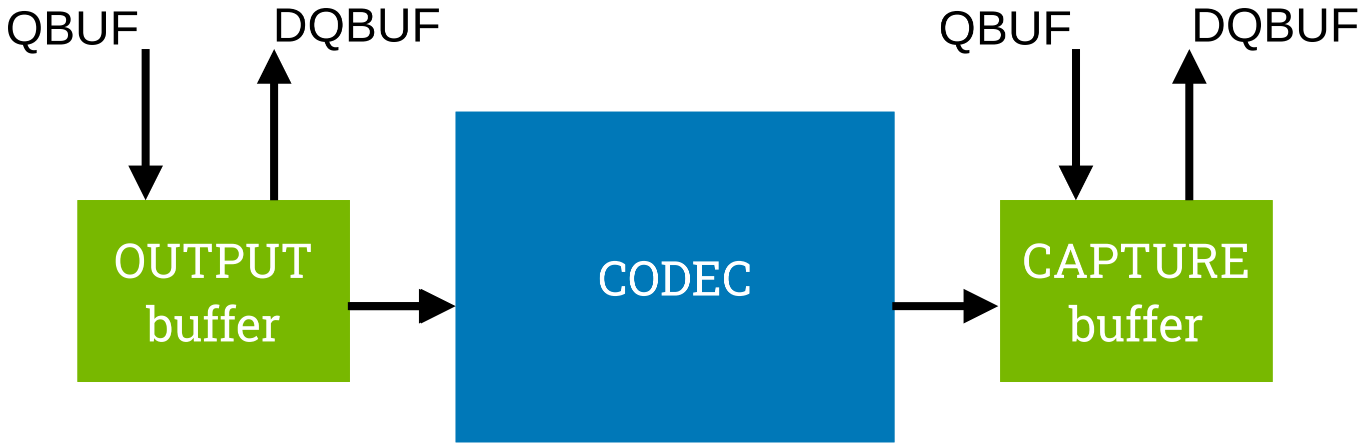
V4L2 mem2mem Drivers

- Examples: vicodec, coda, hantro, vc4, ...
- Specification in Linux kernel source
- API test suite: v4l2-compliance

Correction:
The V4L2 mem2mem driver for the VideoCore 4 found on the Raspberry Pi is called bcm2835-codec and currently only in the Raspberry Pi downstream repository.

Codec Devices: Terminology

- Hans Verkuil – V4L2: A Status Update (ELC-E 2019)
- https://www.youtube.com/watch?v=RUe7_Tx0vOI



Stateful vs. Stateless Encoder

- Reference frame handling
- Rate control handling
- Meta data generation

VCU Mainline Driver: Allegro

Hardware Interface

- Memory mapped registers (in FPGA space)
- Memory mapped SRAM (in FPGA space)
- MCU: MicroBlaze microprocessor
- MCU accesses system memory via AXI

Load Firmware

- Reset MCU
- Load al5e_b.fw to beginning of SRAM
- Load al5e.fw somewhere in system memory
- Write address of al5e.fw in a specific register

Mailbox Interface

- Two mailboxes located at end of SRAM
- One mailbox per direction
- Head and tail pointer
- Interrupt to MCU
- Interrupt from MCU

Initialize MCU

- Write to WAKEUP register
- Receive INIT message
- Send another INIT message
- Receive another INIT message

Configure Encoder

- Application opens device
- Sets encoding parameters using V4L2 API
- Queues buffers to OUTPUT and CAPTURE
- No interaction with the VCU

Create Channel

- Channel is a hardware context
- Message: CREATE_CHANNEL
- MCU requests reference and intermediate buffers

Peculiarity 1: Unstable Firmware ABI

- Size of message changes between firmware releases
- Order and size of fields in messages changes
- Abstraction layer to support different message formats

Encode Frames

- Send CAPTURE buffer: PUT_STREAM_BUFFER
- Send OUTPUT buffer: ENCODE_FRAME

Peculiarity 2: Shadow Queues

- VCU does not run at the expected frame rate
- Queue multiple buffers in the VCU
- Shadow queues in driver necessary

Handle Coded Data

- Receive response message from MCU
- Read partition table in CAPTURE buffer
- Find VCL NAL units in CAPTURE buffer

Peculiarity 3: Meta Data Generator

- VCU does not produce meta data
- Stateless encoder interface expects meta data
- Driver has to generate the meta data

Peculiarities

1) Unstable firmware ABI
2) Shadow queues
3) Meta data generator

Current State of the Allegro Driver

- Supports H.264 encoding
- Supports Const QP, VBR, CBR rate control modes
- About to move from staging to mainline in 5.12
- Patch series for HEVC encoding
- https://lore.kernel.org/linux-media/20201203110106.2939463-1-m.tretter@pengutronix.de/

Addendum:
HEVC support is expected to be included in 5.12

How to Use the Allegro Driver

- Configure bitstream according to Xilinx documentation
- Add device tree node (example in Documentation/devicetree/bindings/media/allegro.txt)
- Enable CONFIG_VIDEO_ALLEGRO_DVT

A Glimpse into the Future

Encoder Buffer

- Other names: L2Cache, prefetch-buffer, CacheLevel2
- Block RAM or UltraRam as cache for reference frames
- Reduces AXI memory bandwidth requirement

AXI Performance Monitoring

- Monitor memory traffic on AXI ports
- Measure effects of Encoder Buffer
- Not sure how to expose to user space

Slices

- Allow parallel processing and improve error resilience
- Encoder needs slices for certain resolutions
- Slices are placed in the middle of CAPTURE buffers

IOMMU

- ZynqMP contains an IOMMU
- DMA_BUF import from other device
- V4L2 buffers have IOMMU addresses

PL-RAM

- DDR memory dedicated to FPGA
- Capture in FPGA might use the PL-RAM
- MCU only able to address a 2 GB window
- Affects Firmware, OUTPUT and CAPTURE buffers
- DMA allocator in driver must be aware

Unsupported Features

- Dual Pass Encoding
- VCU Sync IP
- Video Decoder
- VCU DDR4 LogiCORE IP
- ...

Conclusion

- Try the allegro driver
- Report bugs
- Request features
- Send patches

Thank You!

linux-media@vger.kernel.org

m.tretter@pengutronix.de