IOMMU Evaluation in Automotive use-cases

Khiem Nguyen / Engineer
Renesas

@KhiemNguyenT
WHO AM I ?

- **Name**: Khiem Trong. Nguyen (KHIEM Nguyen)
- **Company**: Renesas Design Vietnam

- **Career**: 10 years experiences in embedded software development
  - Development and verification for Mobile and In-vehicle software platform
  - Development for test automation solutions of In-vehicle software platform

- **Email**: khiem.nguyen.xt@renesas.com
Renesas Design Vietnam Co., Ltd. (RVC) was founded in October 2004, as one of the main design centers in Renesas group.

Business line: Design of semiconductor for both hardware and software.
IOMMU EVALUATION IN AUTOMOTIVE USE-CASES
OPEN SOURCE SUMMIT NORTH AMERICA 2018

08/30/2018, VANCOUVER
KHIEM NGUYEN
SENIOR STAFF ENGINEER
RENESAS DESIGN VIETNAM
RENESAS ELECTRONICS CORPORATION
AGENDA

- Motivation
- IOMMU Overview
- IOMMU support for Automotive use-cases
  - Buffer Sharing
  - Resource Isolation
- Conclusion
MOTIVATION
MOTIVATION (1/3)

- Nowadays, multiple applications or services running in parallel in a vehicle.
- The services can be implemented in multiple silicon or share the power of a single silicon.
MOTIVATION (2/3)

Image from GENIVI Graphics Sharing and Distributed HMI Compositing presentation in Dec 2017
MOTIVATION (3/3)

Need IOMMU support?  

Optional  

Required  

Image taken from The AGL Software Defined Connected Car Architecture white paper
IOMMU OVERVIEW
IOMMU OVERVIEW – IOMMU HARDWARE

- Have similar capability like MMU for CPU cores.
- Have been supported in recent ARM architecture, i.e. ARMv7 and ARMv8.x
- In ARMv7, it’s an optional extension called LPAE (Large Physical Address Extension), be available in ARM Cortex-A15 and newer ARMv7 cores.
IOMMU OVERVIEW – IOMMU HARDWARE

- Support 03 translation granule type.
  - 4KB, 16KB and 64KB
- Support 1-stage address translation and 2-stage address translation.
  - 1 stage (Physical Address <-> Virtual Address)
  - 2 stage (Physical Address <-> Immediate Physical Address <-> Virtual Address)
IOMMU OVERVIEW – IOMMU FRAMEWORK

- Be a part of Linux kernel for a long time (dated back to 2007).
- Support 4KB (default configuration) page size, 16KB and 64KB.
- Support managing address translation for Linux device drivers.
  - Initialize the translation contexts (MMU pagetables)
  - Initialize the memory mapping, physical to virtual, and register DMA callback handlers for each device attached to the contexts.
- Re-mapping the buffers when sharing buffers are triggered across the contexts.
IOMMU OVERVIEW – IOMMU FRAMEWORK

Without IOMMU support

With IOMMU support
IOMMU SUPPORT FOR AUTOMOTIVE USE-CASES
IOMMU SUPPORT FOR AUTOMOTIVE USE-CASES
HOW IOMMU SUPPORT AUTOMOTIVE USE-CASES?

- Guarantee the correct and efficient dataflow among the devices.
  - **Buffer sharing**
    - Among devices in same contexts
    - Among devices in different contexts
    - Between non-IOMMU-supported devices and IOMMU-supported devices
  - **Resource isolation**
    - Create different memory mapping to grant access or block the access to specific memory areas.
BUFFER SHARING
VIDEO CAPTURE: PROCESSING FLOW AND ISSUES

1. Demand for BIG buffer allocation
2. Sharing data between IOMMU contexts

User applications’ processing

Kernel drivers’ processing

User

Video Capture

Kernel

V4L2 / Video Capture

HW

IOMMU context 0

IOMMU context 1

IOMMU context 0

Memory

Control flow

Data flow

GStreamer

waylandsink

Map buffer

Map buffer

Video Capture

IOMMU context 0

IOMMU context 1

Image Processing

Image Blending

Display composition

Display unit

Wayland

Weston

Demand for BIG buffer allocation

Sharing data between IOMMU contexts

User applications’ processing

Kernel drivers’ processing

IOMMU context 0

IOMMU context 1

Map buffer

Map buffer

Control flow

Data flow

© 2018 Renesas Electronics Corporation. All rights reserved.
BUFFER SHARING
BIG BUFFER ALLOCATION - USE CONTIGUOUS DMA BUFFER – 1/3

Why ?

- The multimedia use-cases require a lot of CMA buffers.
  - With the default kernel allocation (4KB per allocation), it takes time to map/re-map big CMA buffers (e.g. 2MB, 4MB, etc).
- Should use CMA contiguous allocation for big buffer allocation.
  - Improve the performance
  - Buffer sharing among two or more devices with different memory requirements.

Use DMA_ATTR_FORCE_CONTIGUOUS

- The IOMMU support for ARM64 architecture has been available since Linux kernel v4.12.

* See https://www.kernel.org/doc/Documentation/DMA-attributes.txt for more detail
BUFFER SHARING
BIG BUFFER ALLOCATION - USE CONTIGUOUS DMA BUFFER – 2/3

1. CMA buffer allocation (in one function call)
2. Map the buffer to corresponding IOMMU pagetable (in one function call)
3. Buffer is shared via DMA Buffer sharing Framework

User applications’ processing

User

Kernel

Video Capture

V4L2 / Video Capture

IOMMU context 0

V4L2 / vsp2driver

VSP Manager

IOMMU context 1

IOMMU context 0

Wayland

Display

Weston

DRM/KMS

Display Unit

Video Capture

V4L2 / vsp2driver

VSP Manager

IOMMU context 0

V4L2 / Video Capture

User applications’ processing

Data flow

control flow
Below shows the performance between non-IOMMU and IOMMU environment

<table>
<thead>
<tr>
<th>No.</th>
<th>Video capture</th>
<th>non-IOMMU</th>
<th>IOMMU enabled + use contiguous CMA Buffers</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Video capture (1080p)</td>
<td>60 fps</td>
<td>60 fps</td>
</tr>
<tr>
<td>2</td>
<td>Video capture with scaling (input/output: 1080p/720p)</td>
<td>60 fps</td>
<td>60 fps</td>
</tr>
</tbody>
</table>
CONCLUSION
CONCLUSION

- IOMMU can support buffer sharing and resource isolation, which are important criteria of implementing complex Automotive use-cases.

- To guarantee the system performance when IOMMU is enabled,
  - Use contiguous CMA buffer allocation for sharing big buffers.
  - Handle the address translation and dataflow across devices with different view of system memory.

- Future work
  - Check the dataflow and consider the solution to ensure the system performance in other automotive use-cases.
Thanks for your attention.

Q&A