Zephyr® Project

Developer Summit

Zephyr® Project
Developer Summit

Introduce Hardware-Level Device Isolation to Zephyr

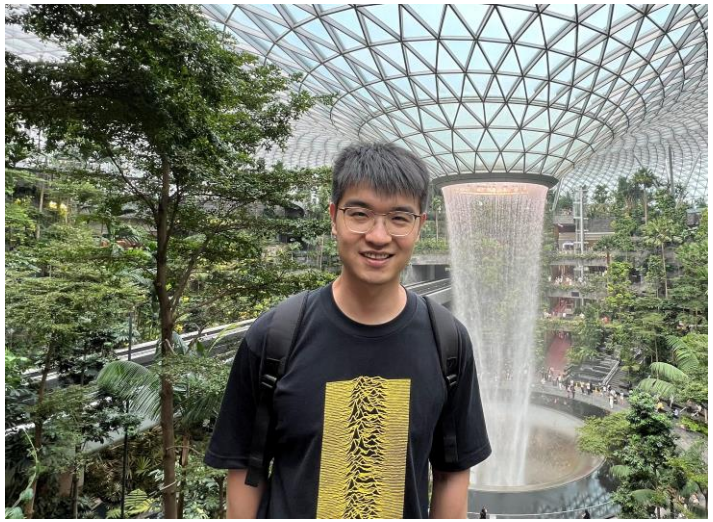
Jaxson Han & Huifeng Zhang – Arm

jaxson.han@arm.com
GitHub: https://github.com/povergoing

huifeng.zhang@arm.com
GitHub: https://github.com/SgrrZhf

Introduce Hardware-Level Device Isolation to Zephyr

Authors



Jaxson Han

Huifeng Zhang

Zephyr® Project
Developer Summit

Contents

- Background
- SMMU
- Zephyr device model
- Zephyr HW-level device isolation
- Conclusion

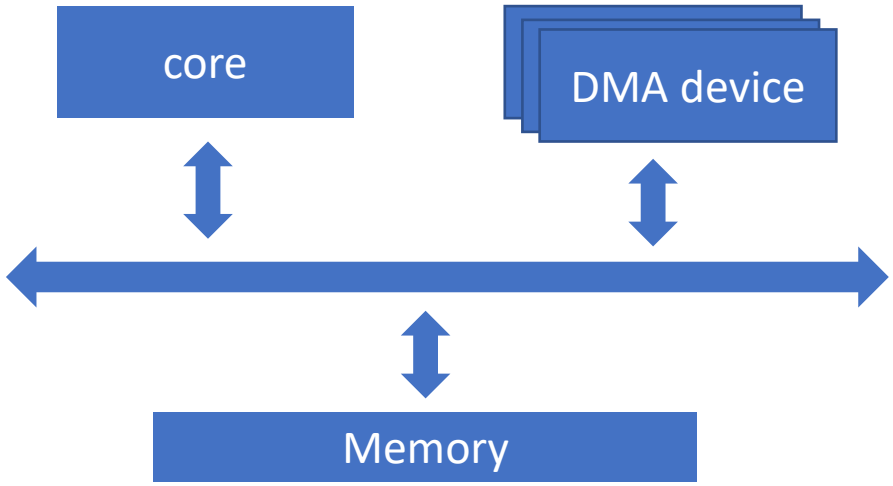
Background

More and more DMA device drivers on RTOS

- An observation:
  - The number of DMA devices on Low-power platforms is increasing.
    - IoT industry
    - …
  - More RTOS on high-performance platforms with variety of DMA devices
    - Automotive Industry (high-performance & safety)
- New challenges for Zephyr:

  DMA device bypass the system access control?

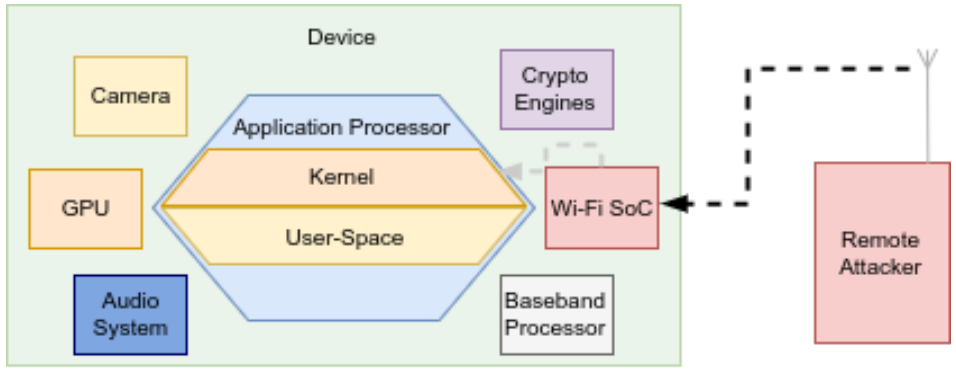
  How to restrict DMA devices?

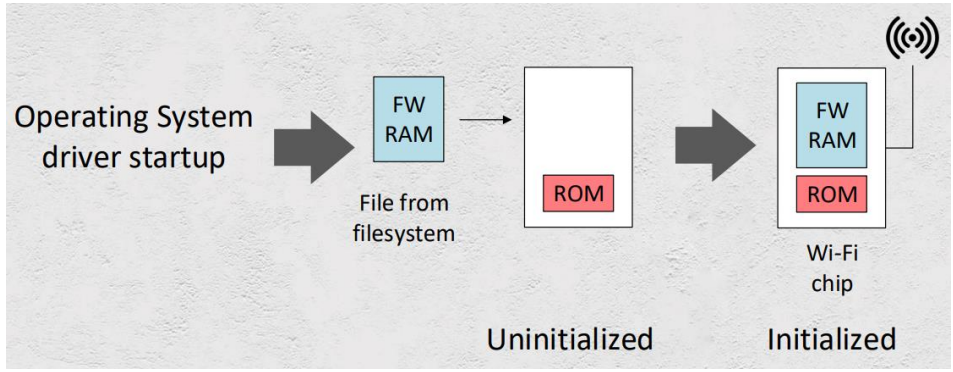


Background

DMA devices might be buggy or even malicious

- DMA devices can break the system
  - WiFi chip bug[1], [2], [3]
    - permission leaks
    - remote control
  - DMA attack[4],[5],[6]
    - steal data or cryptographic keys
    - install or run spyware and other exploits
    - modify the system to allow backdoors or other malware
- More DMA drivers added into Zephyr





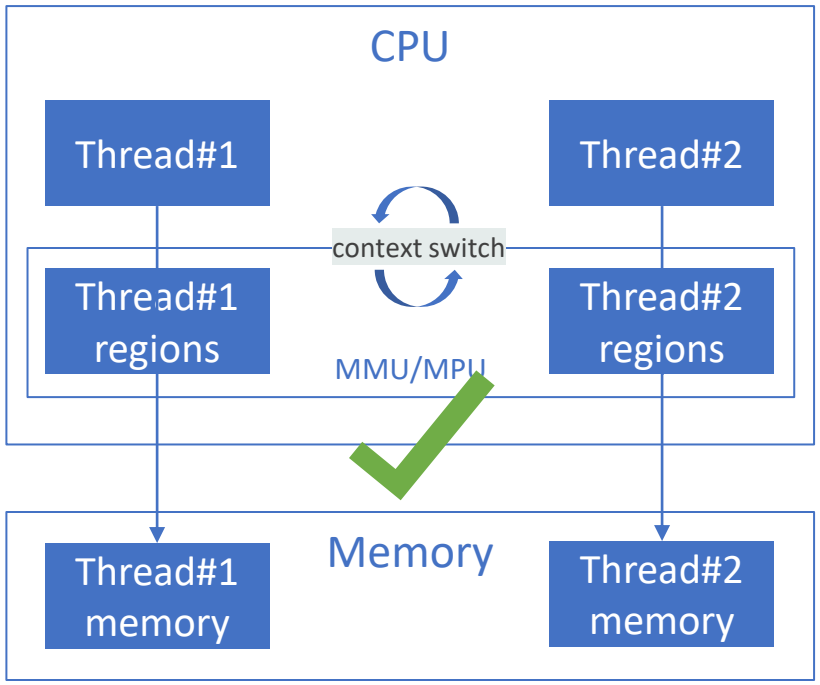
How to restrict DMA devices on Zephyr?

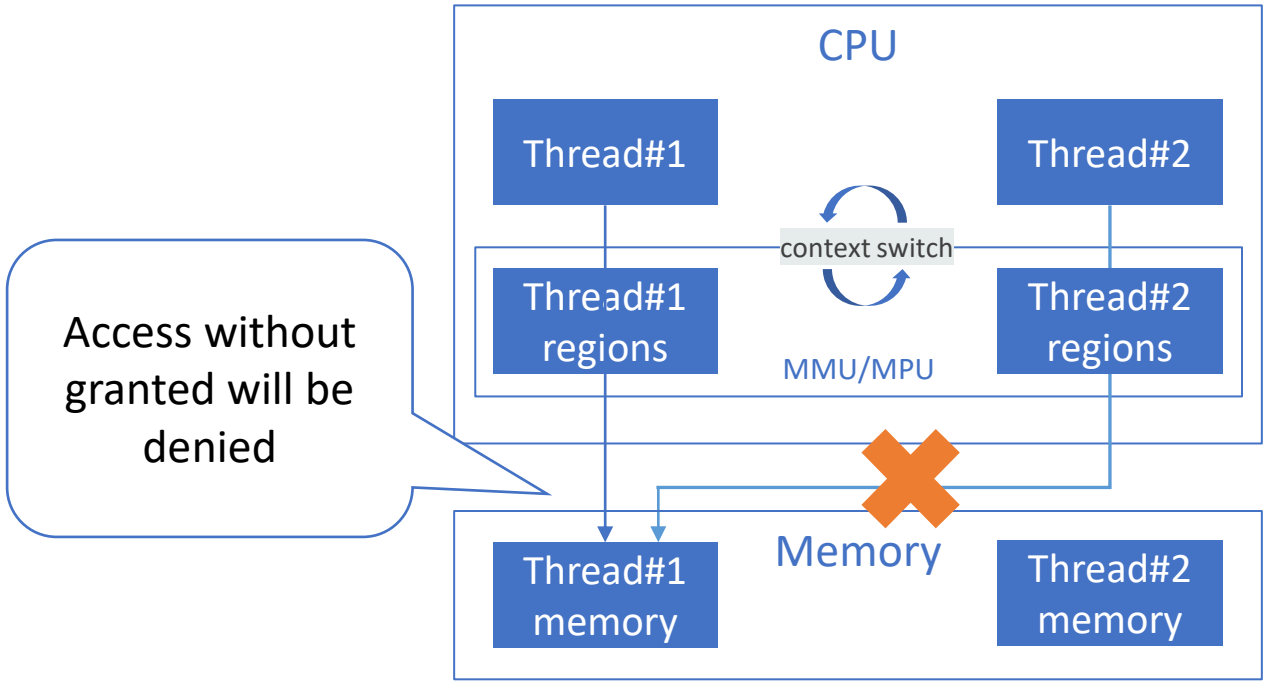
[1] https://googleprojectzero.blogspot.com/2017/04/over-air-exploiting-broadcoms-wi-fi_4.html
[2] https://googleprojectzero.blogspot.com/2017/04/over-air-exploiting-broadcoms-wi-fi_11.html
[3] https://www.bleepingcomputer.com/news/security/vulnerabilities-found-in-highly-popular-firmware-for-wifi-chips/
[4] https://web.archive.org/web/20160304055745/http://www.hermann-uwe.de/blog/physical-memory-attacks-via-firewire-dma-part-1-overview-and-mitigation
[5] https://www.manageengine.com/device-control/prevent-dma-attacks.html
[6] https://en.wikipedia.org/wiki/DMA_attack

Background

Why HW-level device isolation is needed

- Zephyr uses MMU/MPU to isolate the thread memory regions to protect the system.

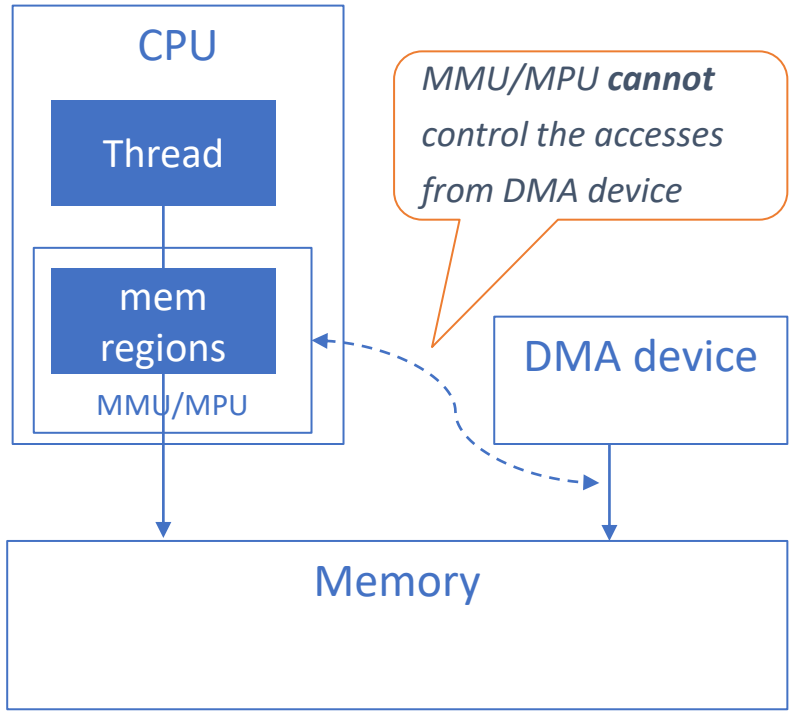


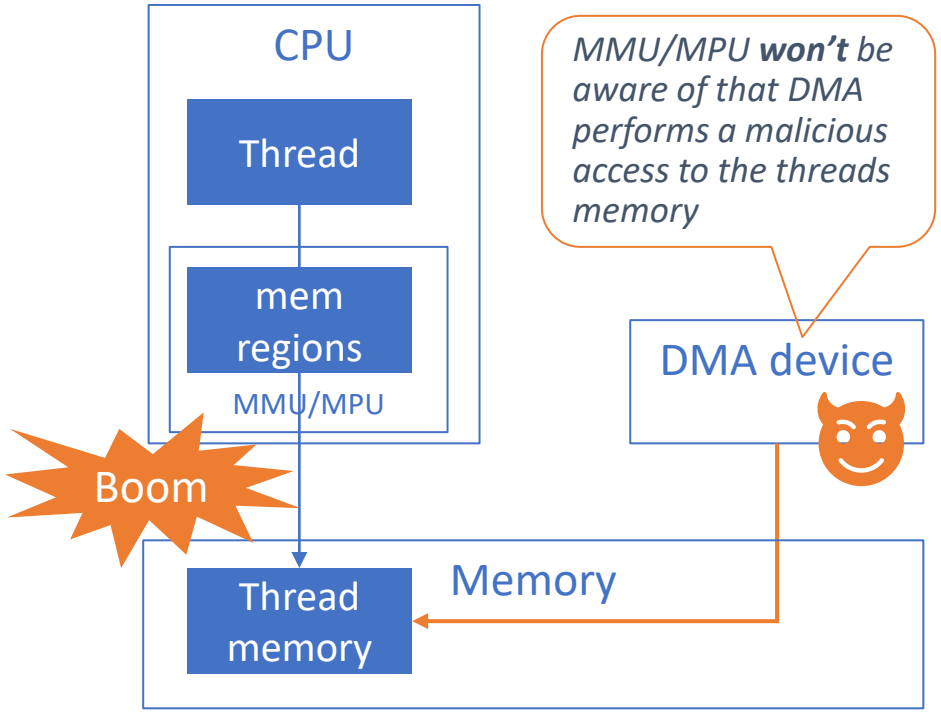


Background

Why HW-level device isolation is needed

- However, MMU/MPU can only restrict memory accesses from CPUs.
- Memory accesses from DMA are **NOT** protected by MMU/MPU
- May cause system crash or security issues





Background

How to mitigate the issue

- Hardware-level isolation can mitigate the issue
  - SMMU (System MMU, Arm)
  - IOMMU (IO MMU, Intel)
  - xPPU (Xilinx peripheral protection unit)
  - Other vendor FireWall
  - ..
- SMMU support allows systems to share A-profile page tables with peripherals
  - widely used in Rich OSes (e.g. Linux) and hypervisor
  - isolate the DMA access
  - eliminate the requirement for physically contiguous pages for DMA buffers
  - extend old DMA device access
  - accelerate virtualization
  - be flexible to switch passthrough or virtualization

Background

How to mitigate the issue

- But too powerful
  - Fully supporting SMMU/IOMMU increases overhead for Zephyr
  - ***Inappropriate*** to add SMMU/IOMMU for low-power platforms.
    - increased cost
    - power consuming
- What we do
  - Partially enable SMMUv3 (as an impl example) for DMA device isolation
    - lower the overhead by using linear mapping (lower TLB miss)
    - isolate every DMA devices to improve the security
  - Add a Subsystem interface to manage DMA device isolation
    - easier to use for driver, components and applications
    - to support more hardware-level isolation technologies

Background

What's more apart from the isolation

- Zephyr as a Virtual Machine Manager (VMM) or a hypervisor?
  - needs SMMU/IOMMU driver to support the virtualization
    - passthrough, VirtIO or …
  - accelerate the DMA access

- Zephyr to support more platforms?
  - devices require SMMU/IOMMU in some high-performance platform
  - To support some 32-bit DMA devices on 64-bit platforms

Zephyr® Project
Developer Summit

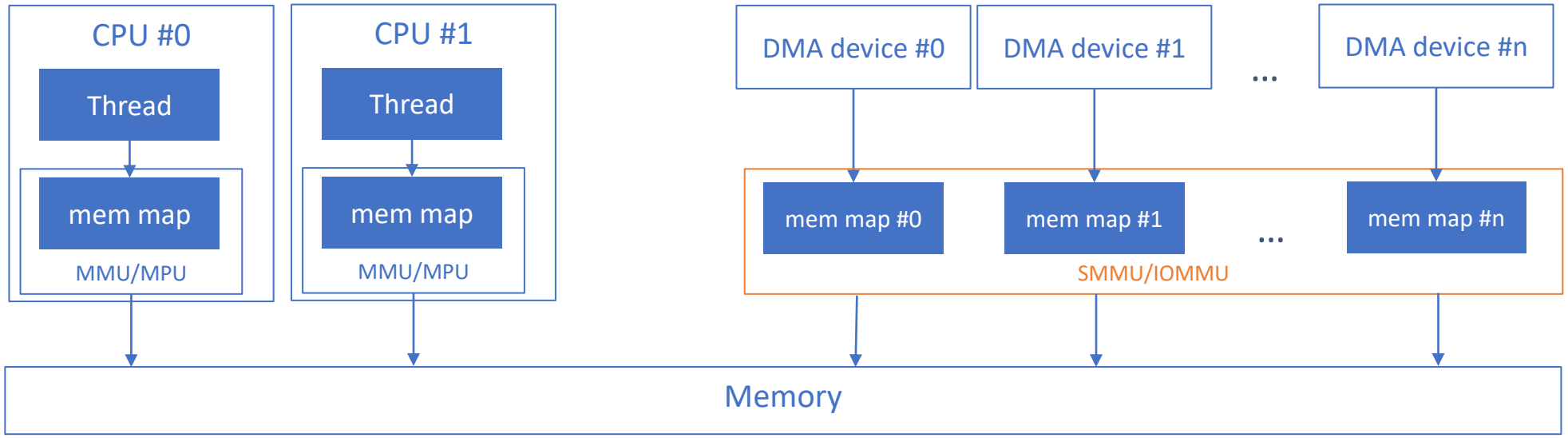
Contents

- Background
- **SMMU**
- Zephyr device model
- Zephyr HW-level device isolation
- Conclusion

SMMU

Introduction

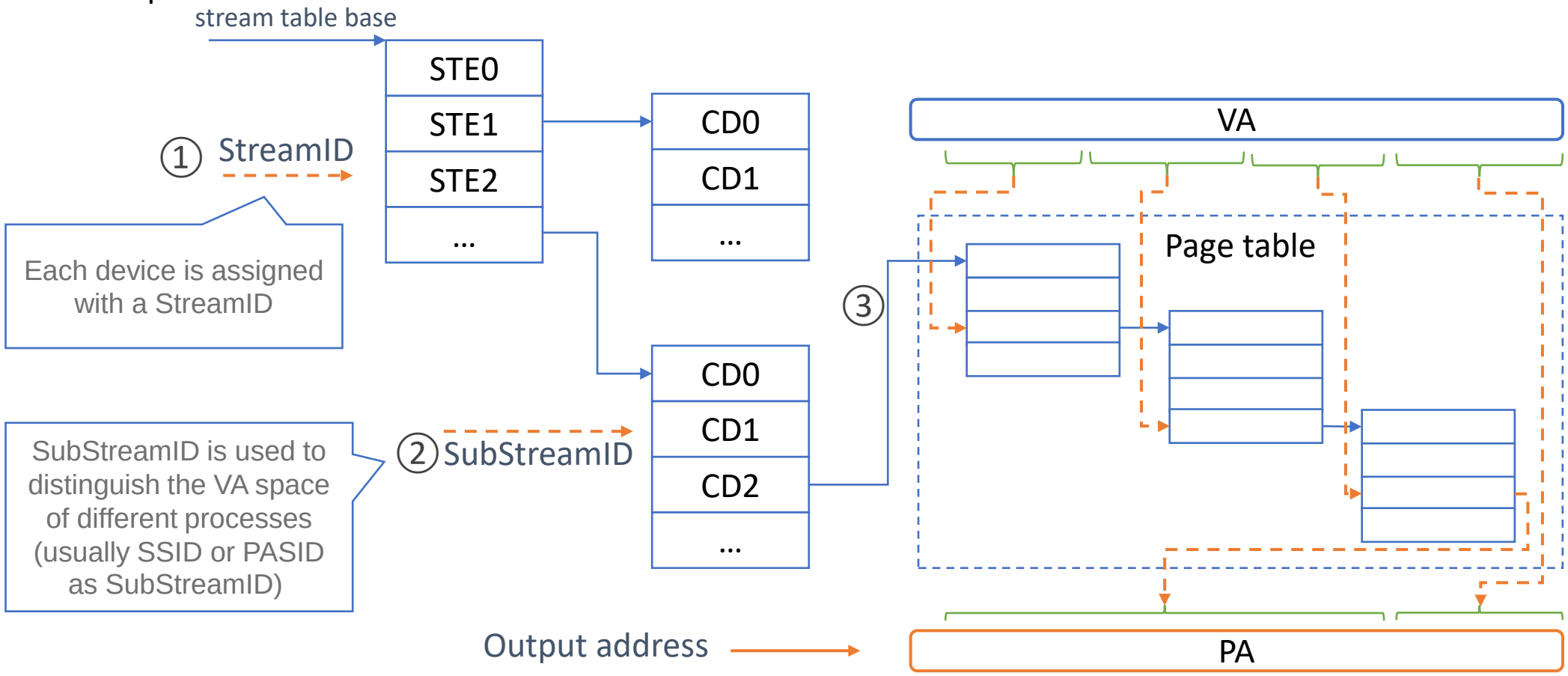
- Similar to MMU:
  - performs address translation and access control
- But differently:
  - for bus initiators external to the CPU (DMA devices).
- Multi page-tables to support multi DMA devices



SMMU

Address translation for DMA devices

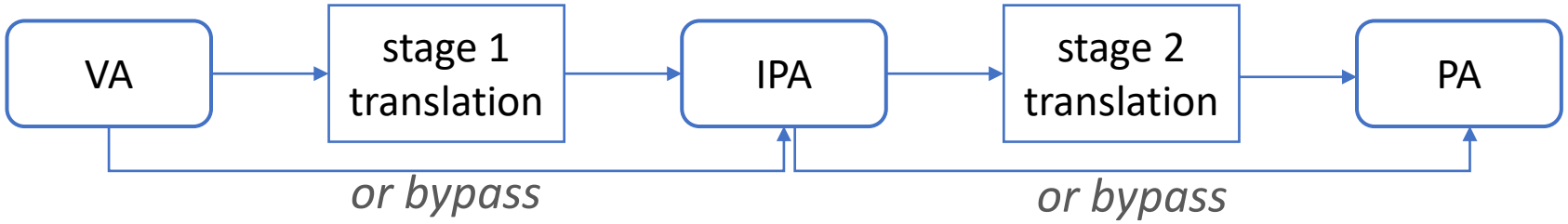
- SMMU has a ***stream table***, **context desc tables** and **page tables** (same with MMU)
- Access issued from a device contains ***StreamID***, ***SubStreamID*** and ***VA***
- Will output a ***PA*** after translation



SMMU

Configurable capabilities

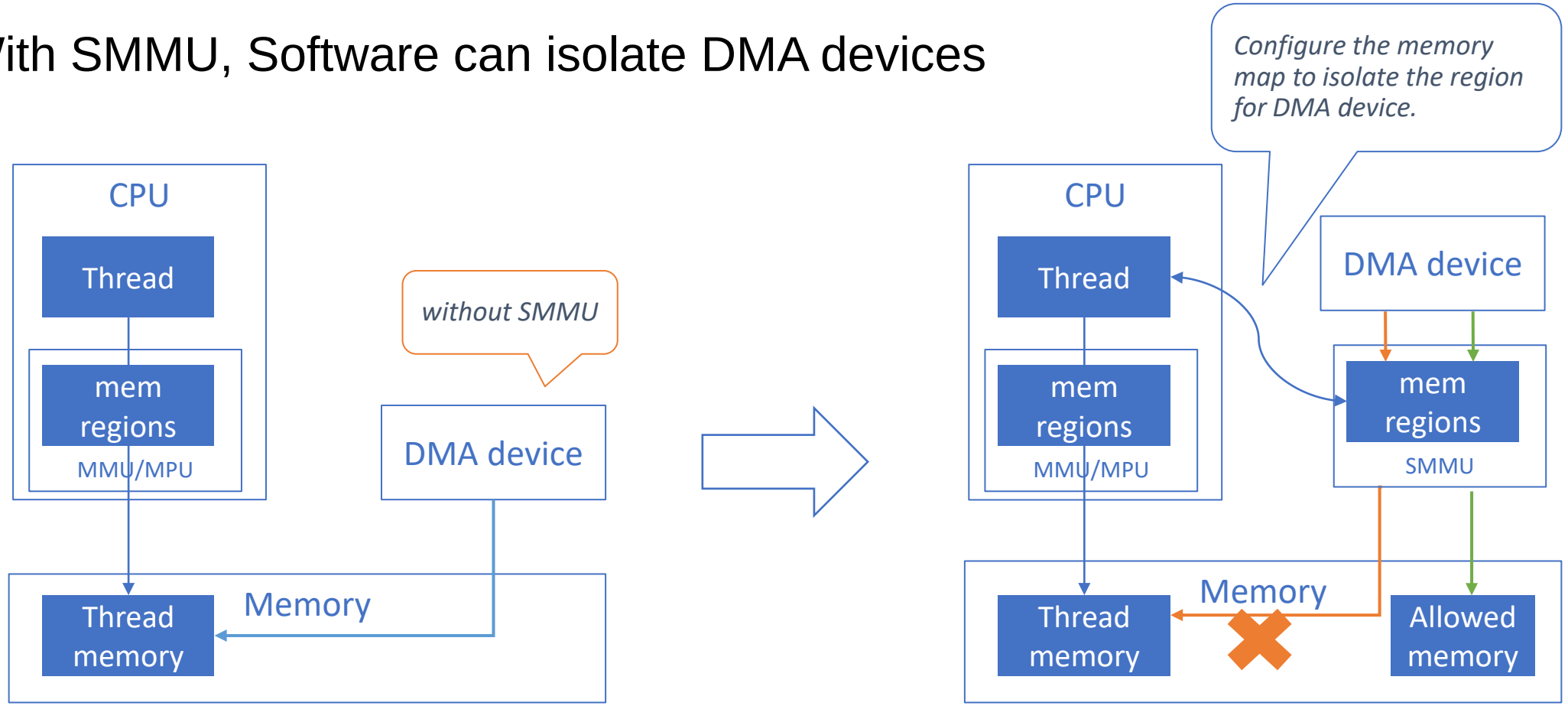
- Bypass the translation (thus, VA = PA)
- Bypass Substream ID
- Reduce translation table level by using block attr(similar with MMU)
- Supports the 2nd stage translation (VA -> IPA -> PA)
  - for hypervisor (virtualization)



SMMU

Isolate DMA devices

- With SMMU, Software can isolate DMA devices



Zephyr® Project
Developer Summit

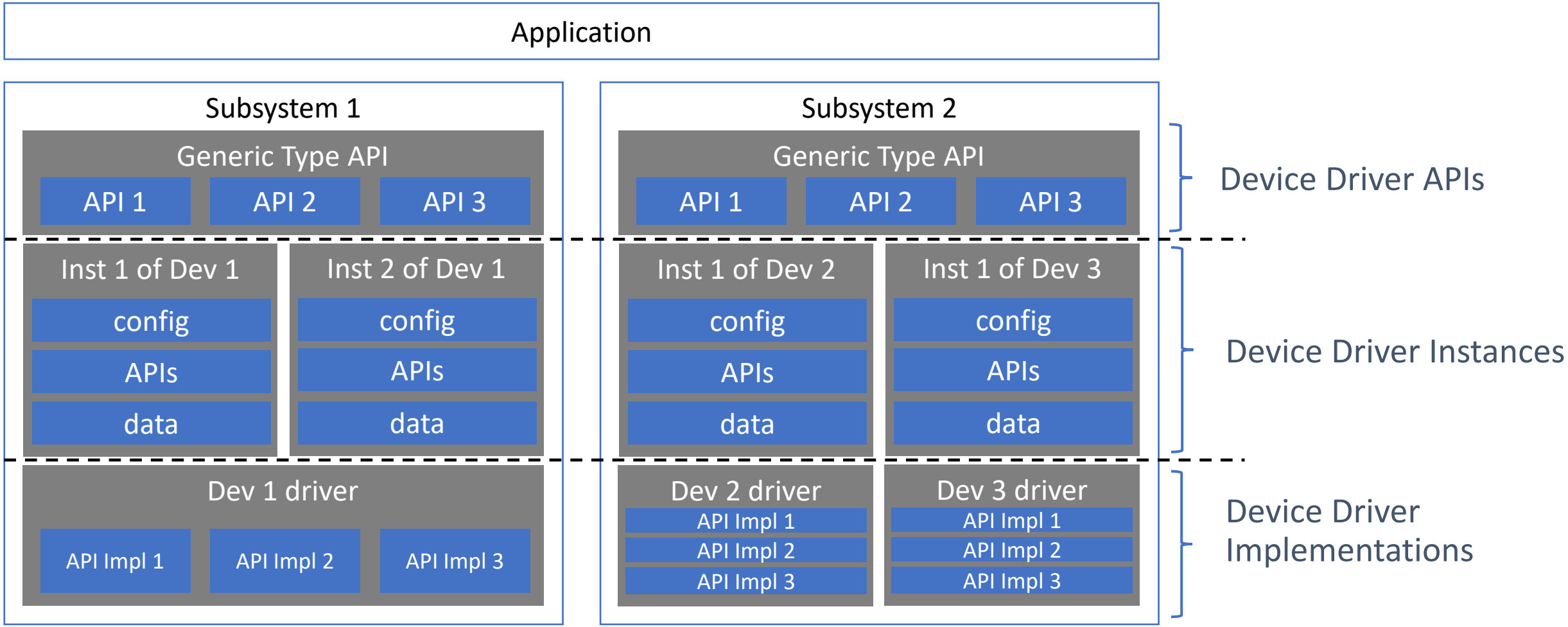
Contents

- Background
- SMMU
- **Zephyr Device Model**
- Zephyr HW-level device isolation
- Conclusion

Zephyr® Project
Developer Summit

Zephyr Device Model

Overview



refs: https://docs.zephyrproject.org/latest/_images/device_driver_model.svg

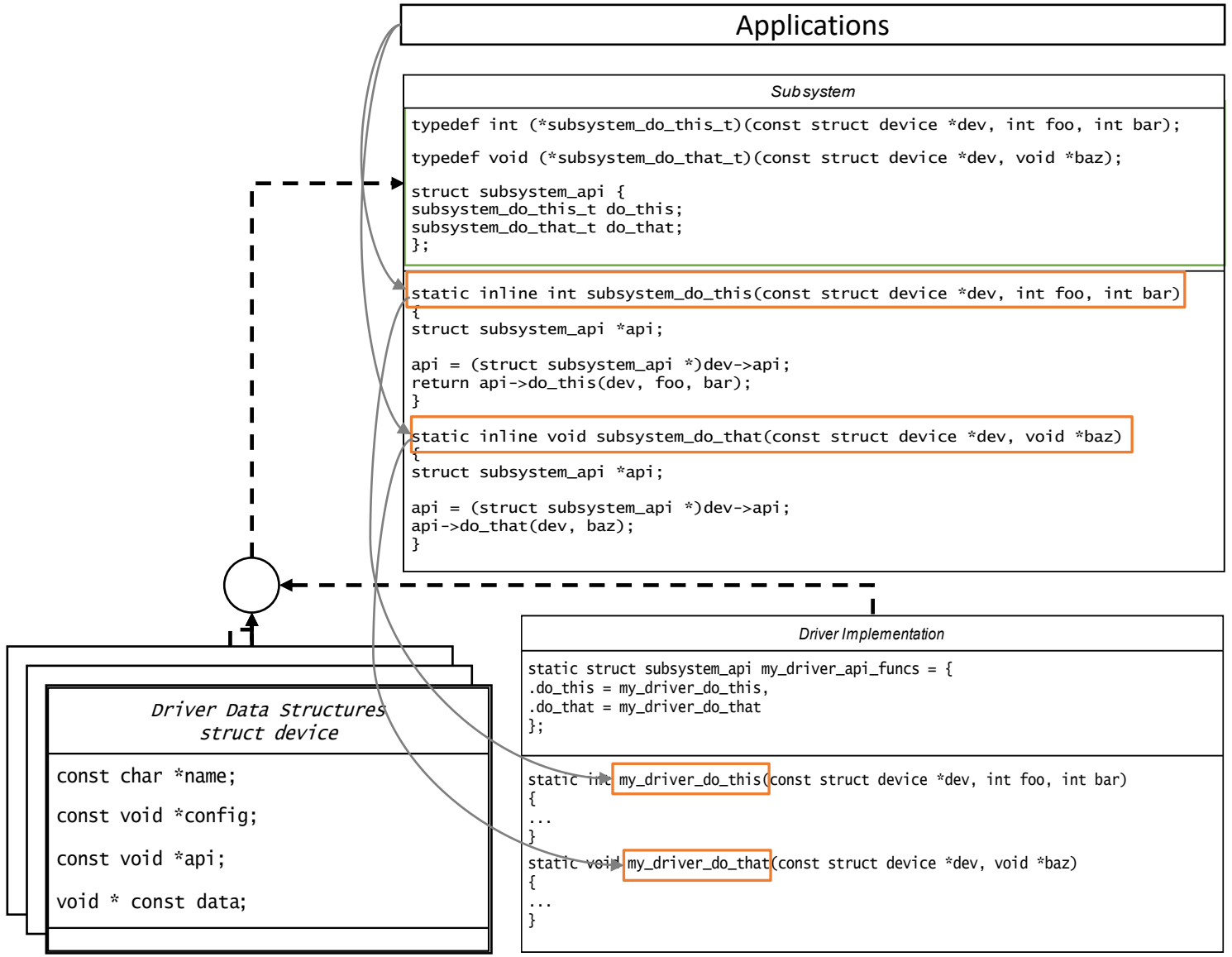
Zephyr® Project
Developer Summit

Zephyr Device Model

Overview

- Subsystem
  - Defines generic Device Driver APIs
  - simply program for applications
  - not specific to any particular driver implementation.
- Driver Data Structures
  - holds Devices data to support multiple devices driver instances
- Device Driver Implementations
  - implements a device-independent subsystem API
  - fills in the pointer to the Driver Data Structures.

call
implement

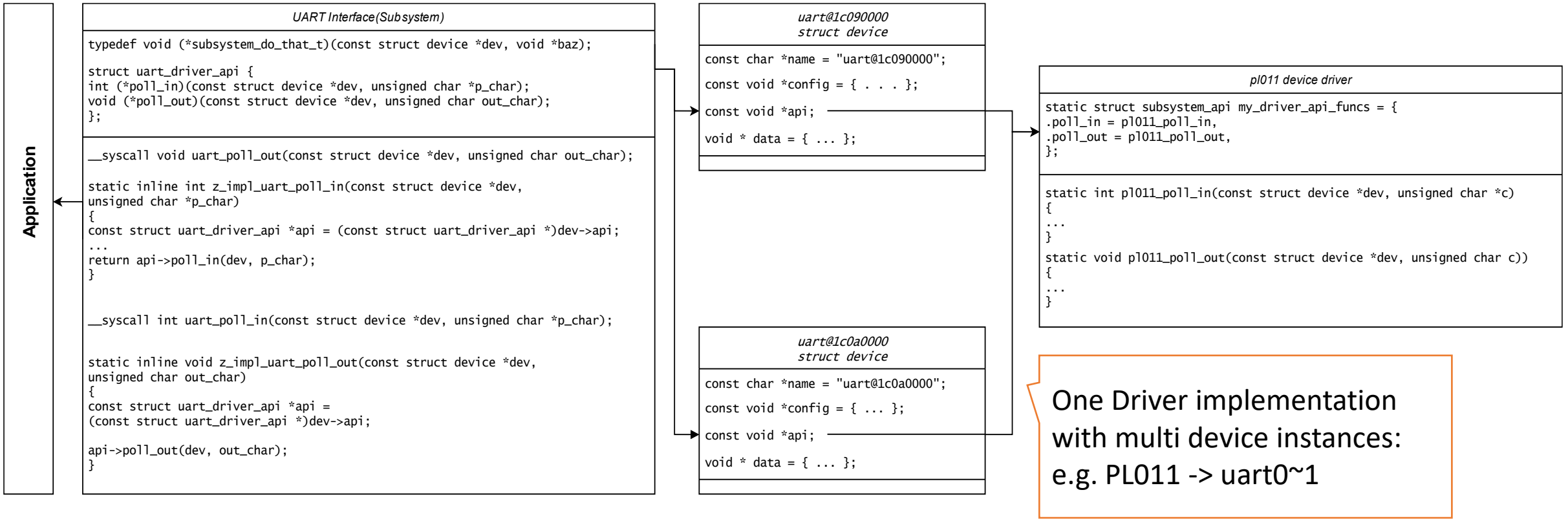


Zephyr® Project
Developer Summit

Zephyr Device Model

A real example

- Single Driver, Multiple Instances



One Driver implementation with multi device instances: e.g. PL011 -> uart0~1

Zephyr® Project
Developer Summit

Contents

- Background
- SMMU
- Zephyr device model
- **Zephyr HW-level device isolation**
- Conclusion

Zephyr® Project
Developer Summit

Zephyr HW-level device Isolation

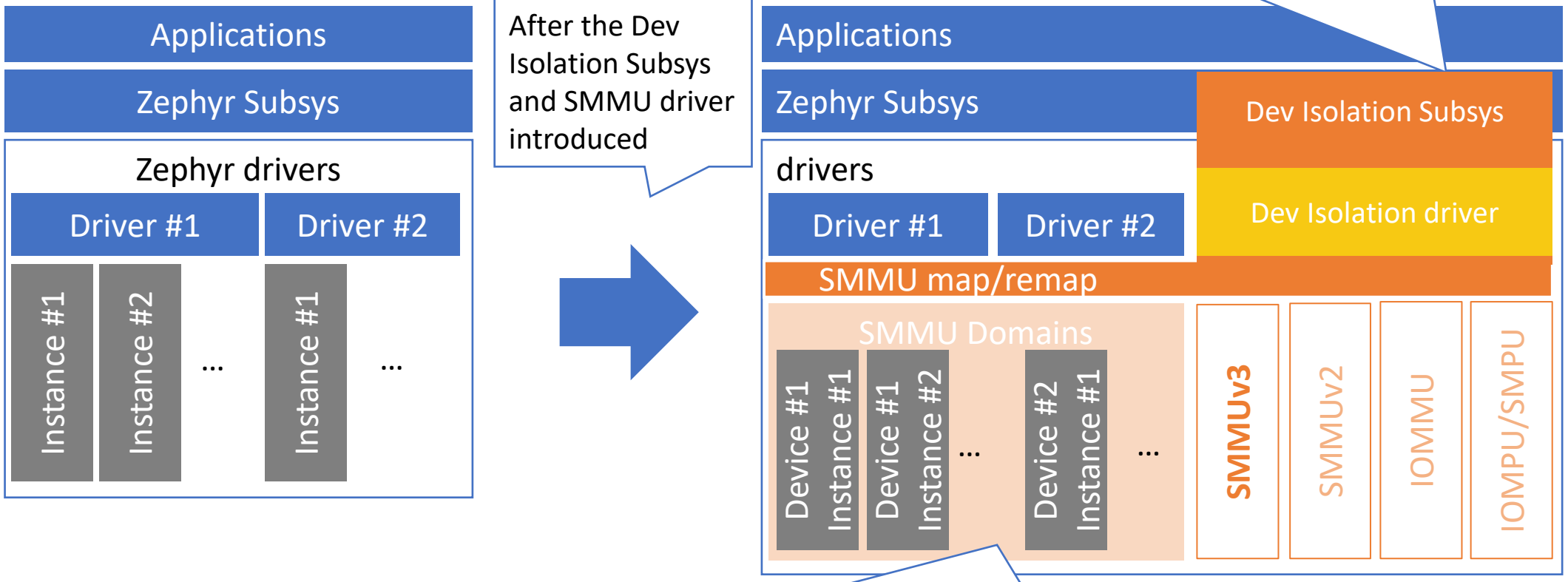
Sub Contents

- The Design: Overview
- The design: DTS
- The Subsystem interface
- The implementation
- Add latency for DMA operations?

Zephyr HW-level device Isolation

The Design: Overview

Zephyr Dev Isolation Subsys consists of 2 parts:
1) register the devices
2) restrict the memory for devices



Domain: Defined an address space. One Domain has one linear space with multi regions.

Zephyr HW-level device Isolation

The design: DTS: PCI device + SMMUv3

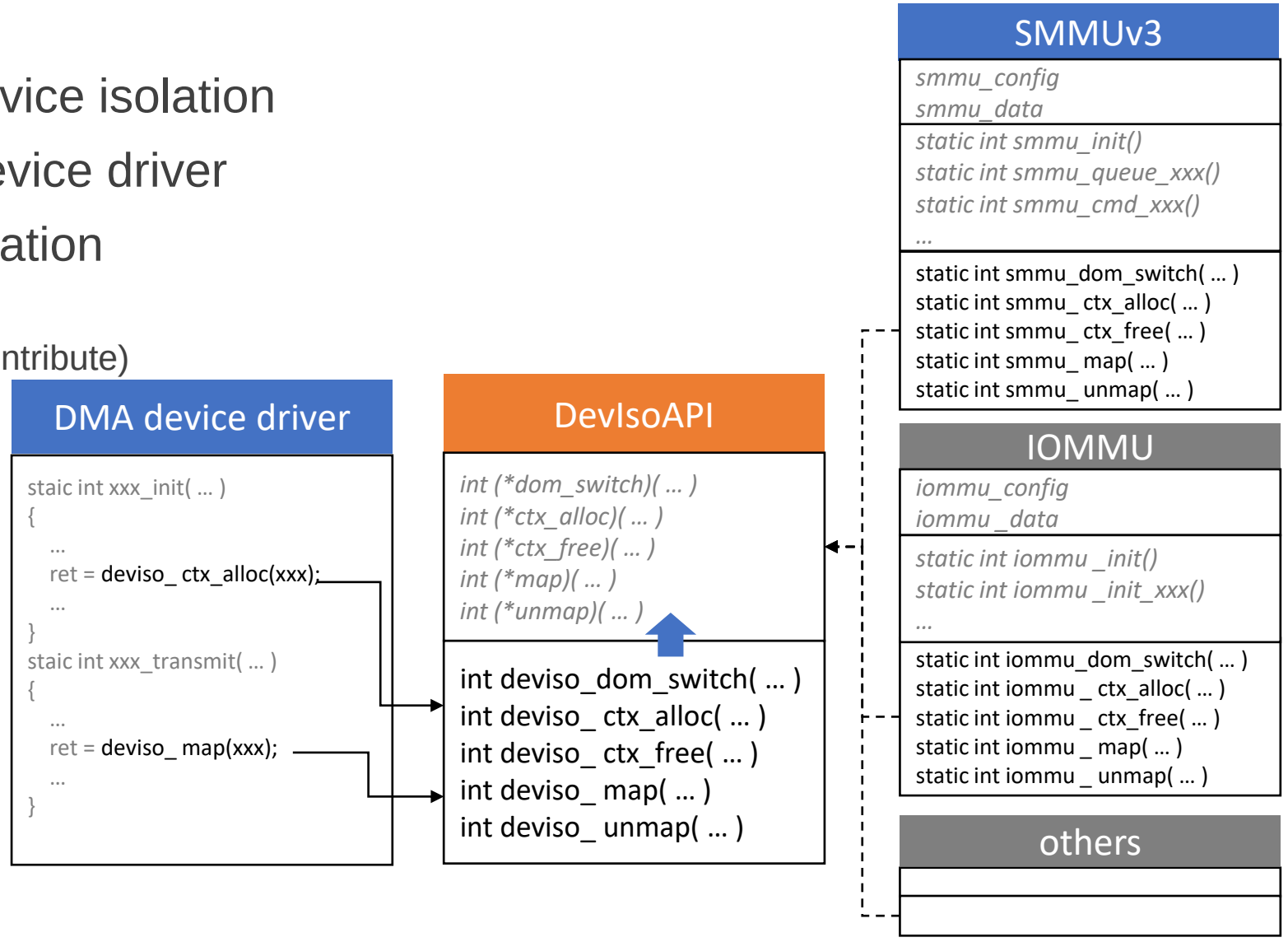
- To make them work, the DTS should provide :
  - *SMMU nodes:*
    - *compatible*
    - *base address*
    - *length/size*
  - *The DMA devices:*
    - *the PCI & dev information*
    - *SMMU phandle*
    - *RID base address(BDF on PCI)*
    - *StreamID*

```
pci: pci@40000000 {
     compatible = "pci-host-ecam-generic";
     reg = <0x40000000 0x10000000>;
     msi-parent = <&its>;

     #address-cells = <2>;
     #size-cells = <1>;
     ranges = <0x2000000 0x50000000 0x50000000 0x10000000>;

     smmu-maps = <&smmu 0 0 0x10000>;

     ahci: ahci0 {
          compatible = "ata-ahci";

          vendor-id = <0x0abc>;
          device-id = <0xaced>;

          status = "okay";
     };
};

smmu: smmu@2b400000 {
     compatible = "arm,smmu-v3";
     reg = <0x2b400000 0x100000>;
     #smmu-map-cells = <3>;
};
```

Zephyr HW-level device Isolation

The Subsystem interface

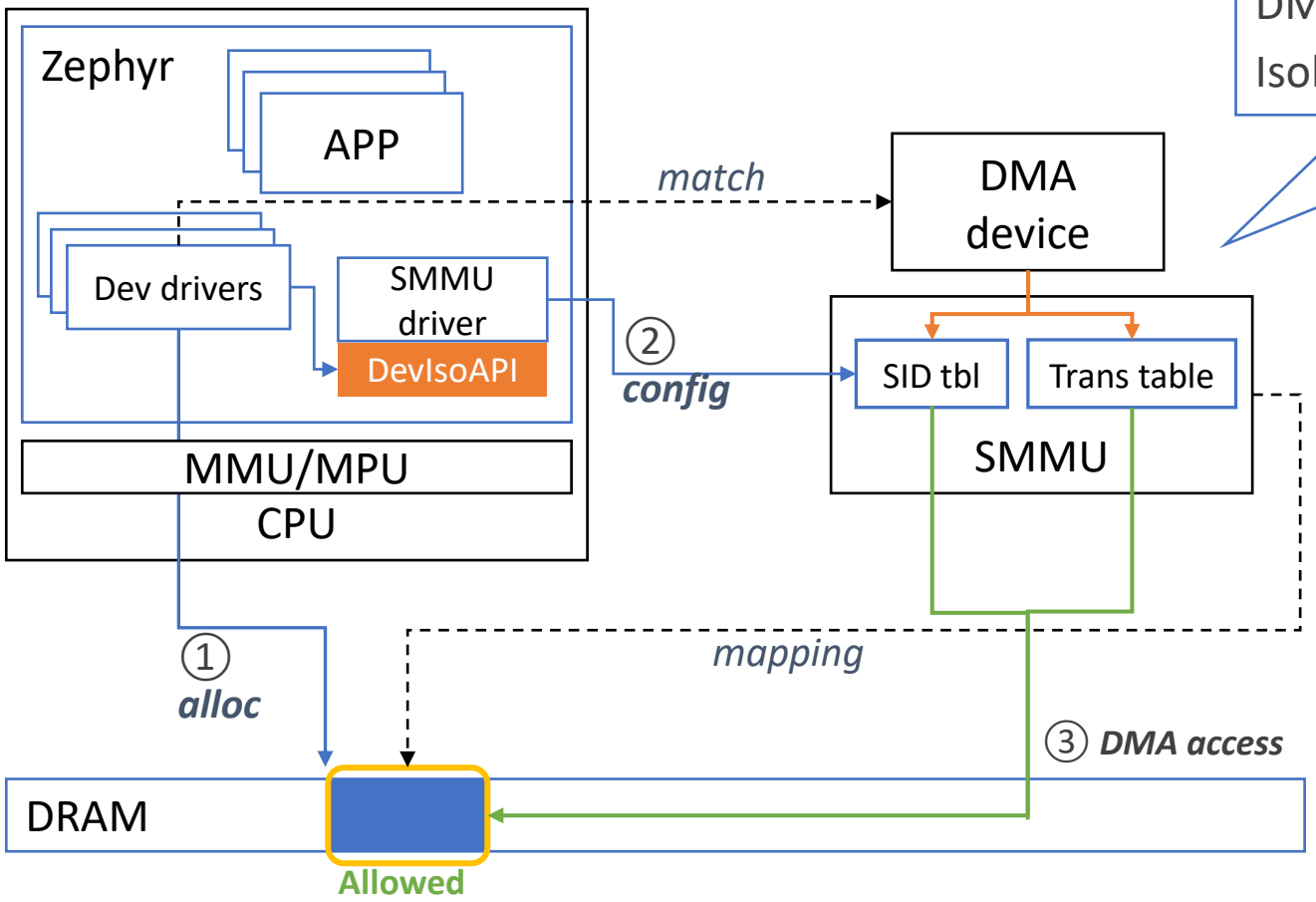
- Defines generic APIs for device isolation
- Simply program for DMA device driver
- Support multiple implementation
  - SMMUv3
  - IOMMU (welcome anyone to contribute)
  - …
- Easy to add new one



Zephyr HW-level device Isolation

The implementation (PoC)

board: fvp_base_revc_2xaemv8a
DMA dev: PCI + AHCI
Isolation driver: SMMUv3



① *Dev driver allocate a DMA buffer for device.*

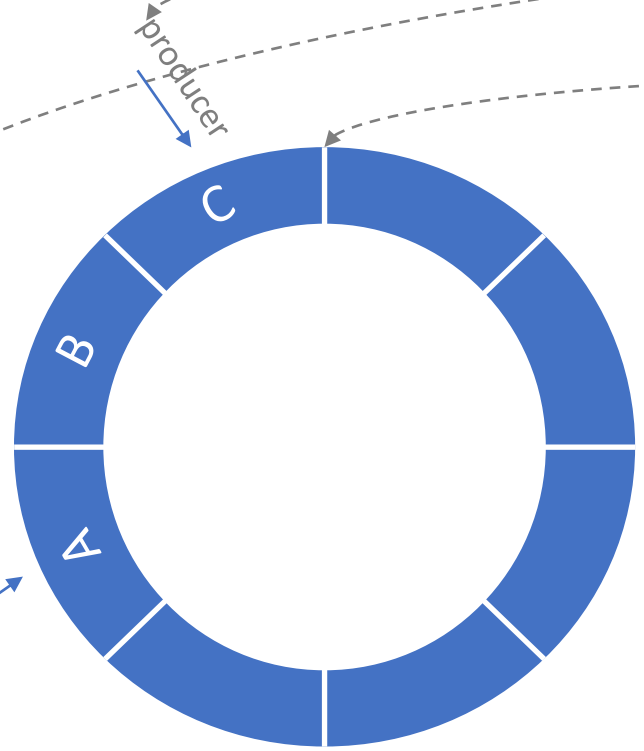
② *Dev driver call DevIsoAPI to map the buffer for the device.*

③ *Device copy data to DRAM.*

Zephyr HW-level device Isolation

The implementation: SMMUv3 control

SMMUv3 driver controls the SMMU by sending cmds to SMMU via a cmd queue which is a ring buffer.



struct smmu_queue_local_copy
uint32_t prod
uint32_t cons

struct smmu_queue
struct smmu_queue_local_copy lc;
void *base;
mem_addr_t base_dma;
mm_reg_t prod_reg;
mm_reg_t cons_reg;
uint64_t q_base;
int size_log2;

struct smmu_cmdq_entry
uint8_t opcode;
struct tlbi;
struct cfgi
struct prefetch
struct sync

smmu command queue functions
int smmu_q_has_space(struct smmu_queue *q)
uint32_t smmu_q_inc_prod(struct smmu_queue *q)
void make_cmd(uint64_t *cmd, struct smmu_cmdq_entry *entry)
void smmu_cmdq_enqueue_cmd(struct smmu_device_data *data, struct smmu_cmdq_entry *entry)
int smmu_sync(struct smmu_device_data *data)
void smmu_invalidate_sid(struct smmu_device_data *data, uint32_t sid)
void smmu_prefetch_sid(struct smmu_device_data *data, uint32_t sid)
void smmu_invalidate_all_sid(struct smmu_device_data *data)
void smmu_tlbi_all(struct smmu_device_data *data)

Zephyr HW-level device Isolation

Add latency for DMA operations?

- SMMU (lower down the tlb miss):
  - statically linear address mapping
  - use block page
  - skip sub stream id
  - appropriately use ATS (Address translation services, platform specific)
  - try not switch translation tables

Zephyr® Project
Developer Summit

Contents

- Background
- SMMU/IOMMU
- Zephyr device model
- Zephyr HW-level device isolation
- **Conclusion**

Conclusion

- DMA can break the system
  - buggy
  - malicious
- DMA device + Zephyr RTOS risk is increasing
  - more DMA on low-power platforms
  - more RTOS on high-performance platforms
- Add HW-level device isolation
  - enable SMMU but lower the overhead
  - add subsys for easy extension
- Future work:
  - send to upstream
  - measure the latency

Thank you!

Q&A