Cornelia Wulf
Chair of Adaptive Dynamic Systems
TU Dresden

TrustedIoT
Trusted Computing Architectures for IoT Devices

16.04.2024

The Chair of Adaptive Dynamic Systems

What we do?

ADS

Domain-Specific Computer Architectures
- Processors
- Accelerators
- Network-on-Chip
- Memory

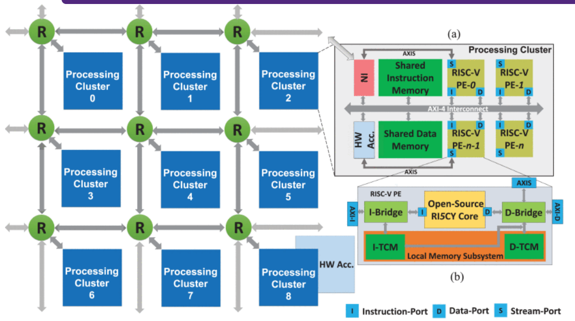
Hardware/Software Co-Design
- Modeling (e.g. UML)
- HW/SW Codesign of applications or middleware (ROS, RTOS)
- High Level Synthesis

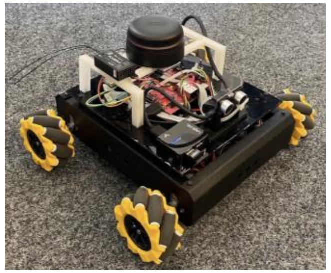
Runtime Reconfigurable Systems
- Virtualization
- Self-adaptation
- Dynamic and Partial Reconfiguration

Programming Methods and Tools
- Design Space Exploration
- Application Partitioning and Mapping
- LLVM
- Processing-in-Memory

DAKORE · marie · 6G-life · CeTI · TrustedIoT · EUFRATE · ScaDS.AI DRESDEN LEIPZIG

Evaluation with Application Scenarios from Adaptive Dynamic Systems, e.g.
Image and Signal Processing Algorithms from Robotics, AAL, I4.0, Automotive





The Chair of Adaptive Dynamic Systems
The Team

Chair:

Ms Prof. Dr.-Ing. Diana Goehringer

Postdoctoral Researcher:

Dr. Sergio Pertuz

PhD Candidate:

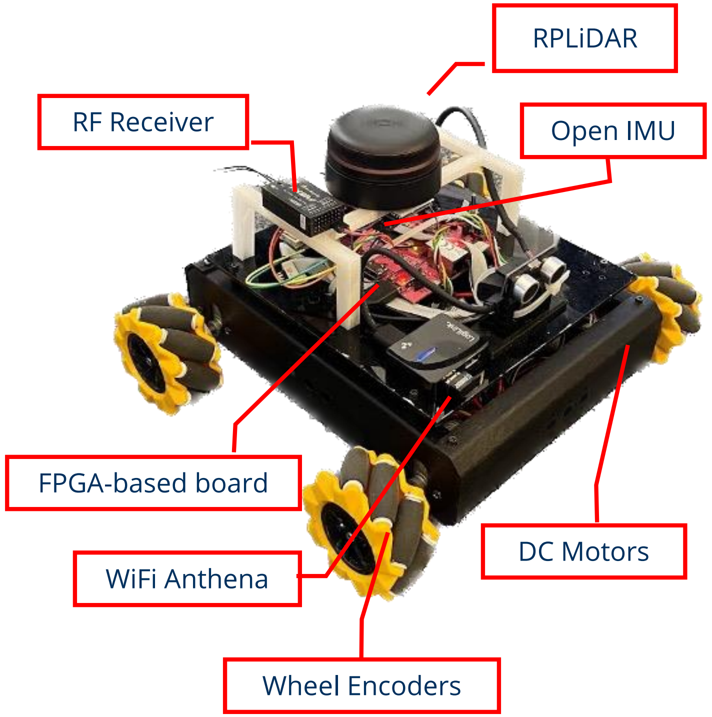
Ms Dipl.-Inf. Cornelia Wulf

Low-power FPGA-based mobile robot with enhanced IoT security:
Motivation

- Software side:
  Hypervisors isolate trusted from untrusted guest operating systems.

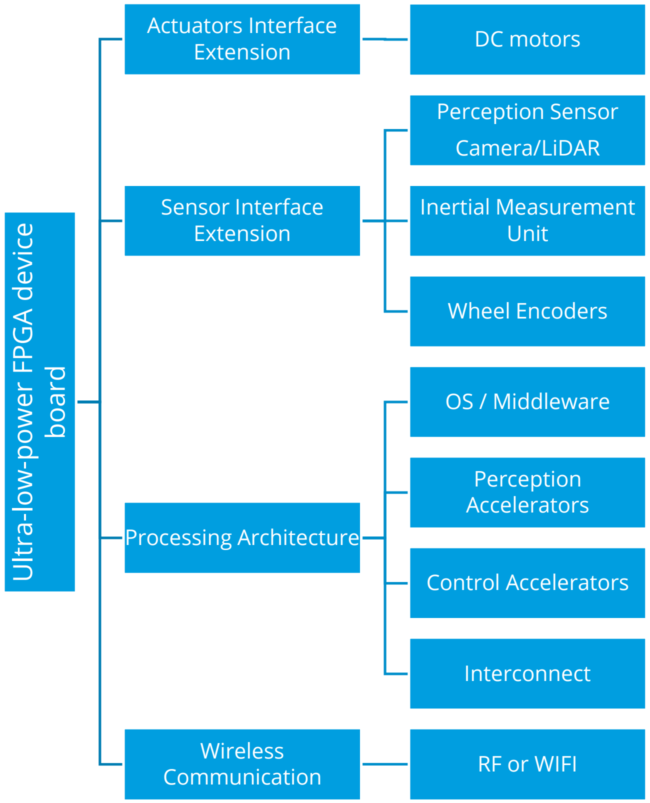
- Hardware side:
  Fine-grained isolation mechanism for shared usage of hardware accelerators is missing.

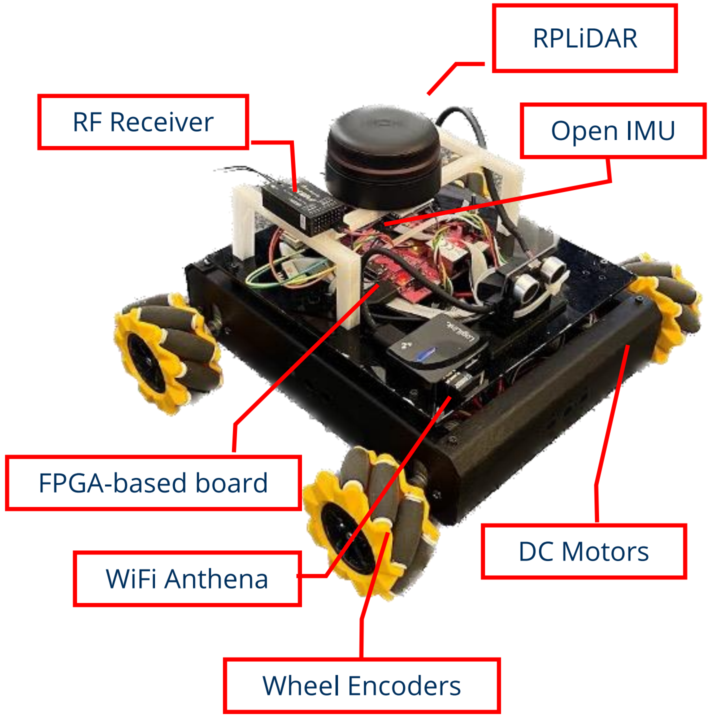
Focus on AXI memory-mapped interfaces



Low-power FPGA-based mobile robot with enhanced IoT security:

Proposed Architecture

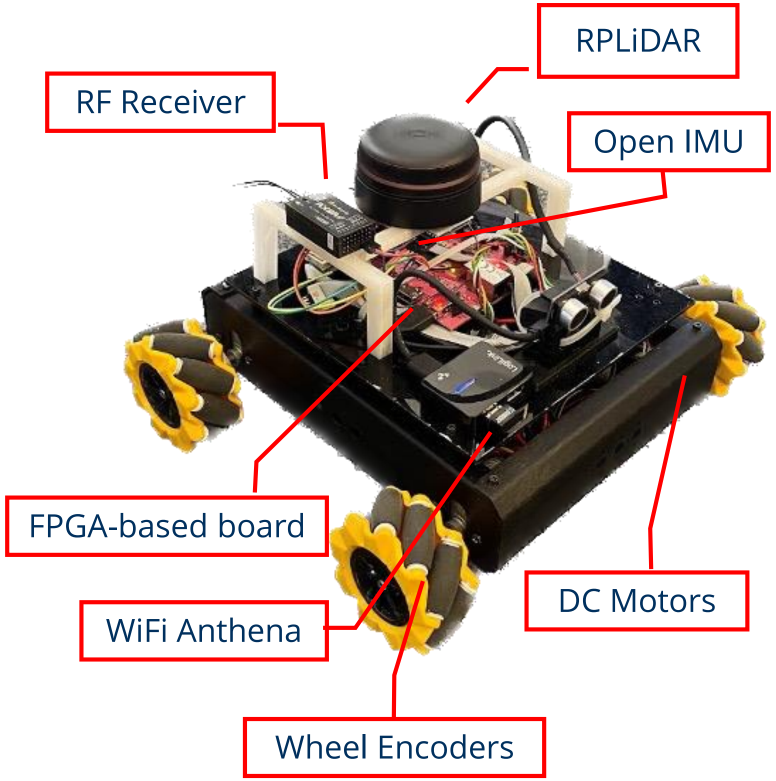




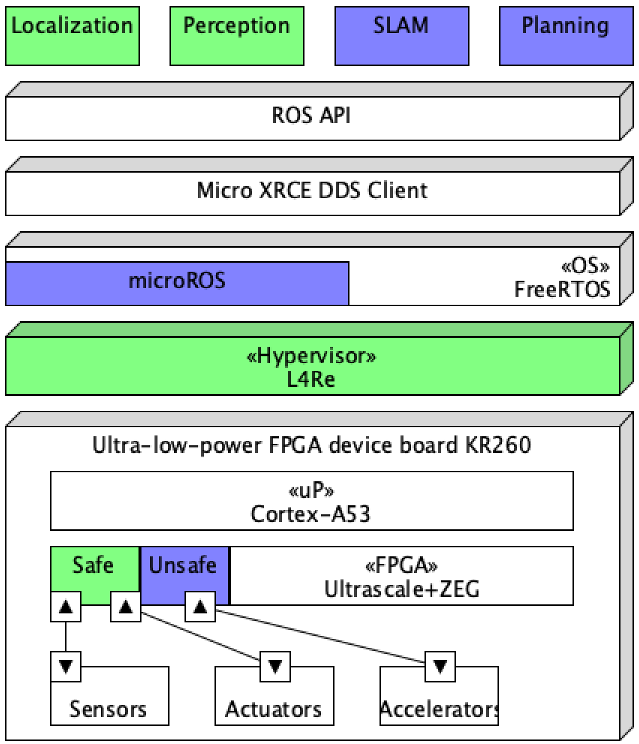
Low-power FPGA-based mobile robot with enhanced IoT security: Platform

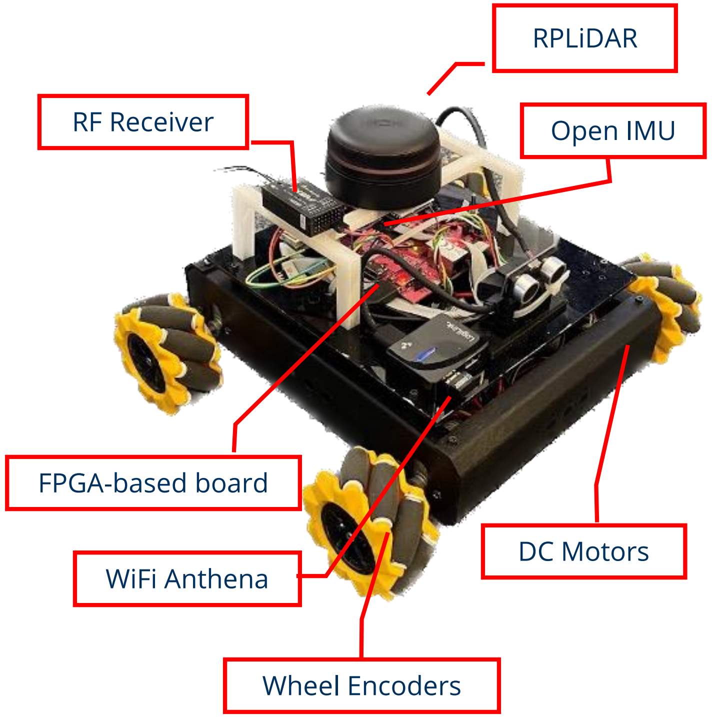
KR260 Robotics Starter Kit:

Zynq UltraScale+™ MPSoC EV (XCK26)



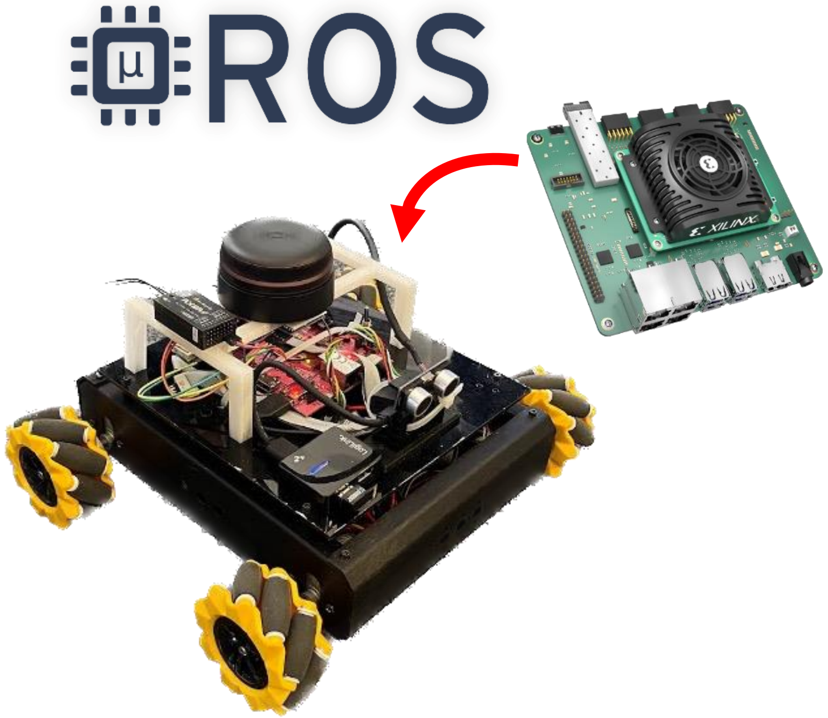
Low-power FPGA-based mobile robot with enhanced IoT security:
Architecture Stack





Low-power FPGA-based mobile robot with enhanced IoT security: Software and Middleware

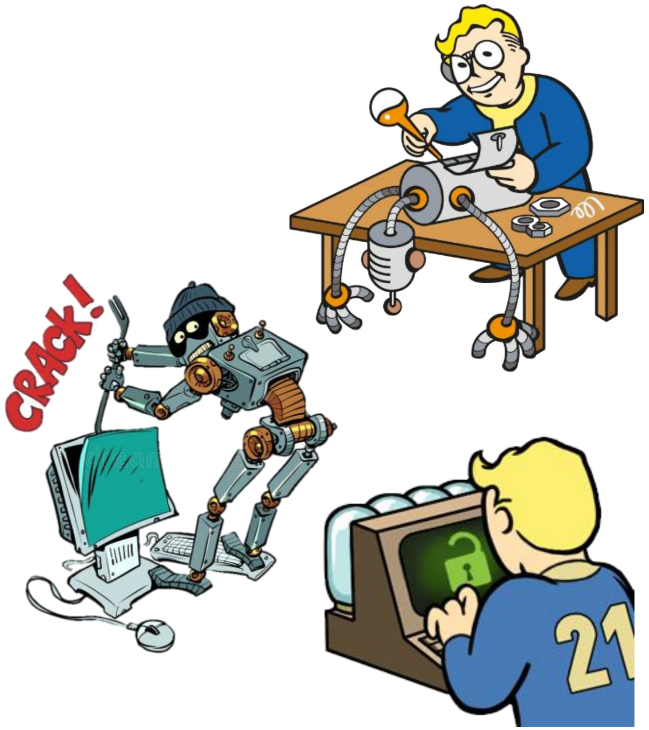
TUD implemented **ROS2/microROS (novelty)** on a Zynq/ZynqMP device and add a **hardware-based secure layer** to the networking and middleware to have improved security in robotics IoT.



Threat analysis overview: Scope

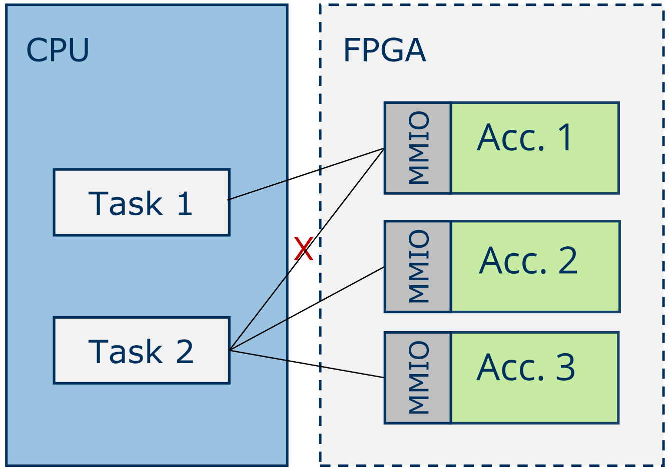
The security scope here is limited to robot middleware and medium-level software vulnerabilities. In particular, **three types of attackers** are considered:

- Human attackers interact physically with the robot (Robot User),
- Another robot or system is capable of physical interaction with the robot. (Third-Party Robotic System), and
- A human teleoperation the robot or sending commands to it through a client application (e.g., smartphone app) (Teleoperator / Remote User)



Protection of hardware accelerators

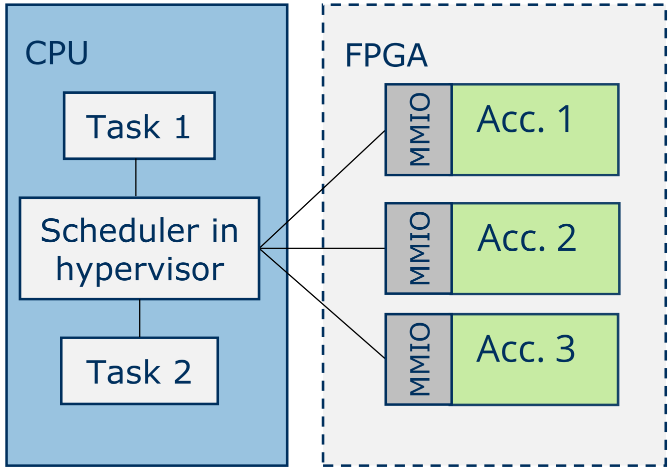
1. Fixed assignment:



Disadvantage:

- No flexibility
- No scalability

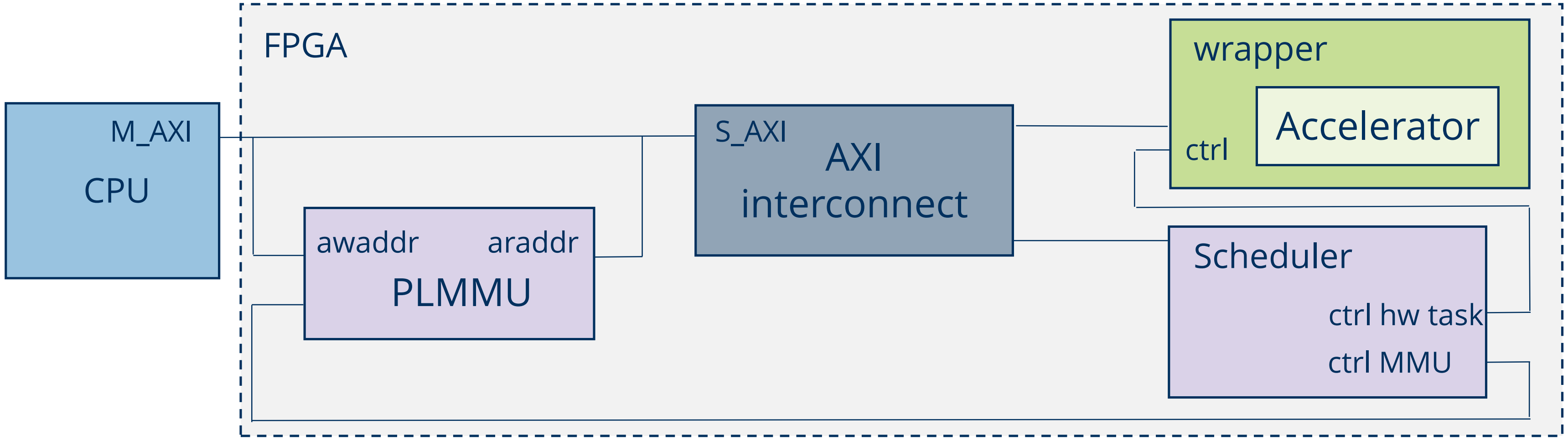
2. Access via software scheduler:



Disadvantage:

- Latency

Memory-mapped access of hardware accelerators



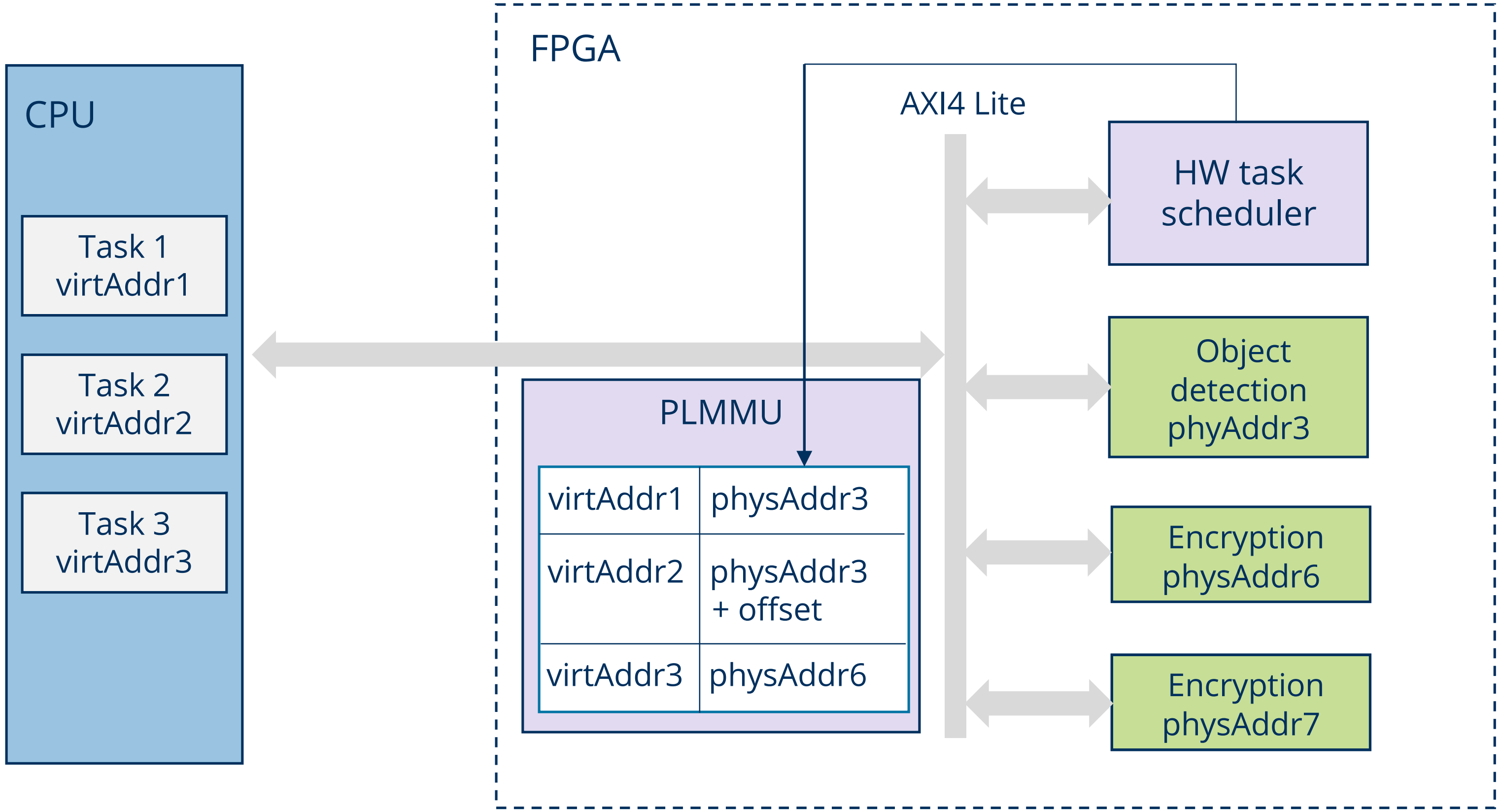
Custom MMU

Scheduler

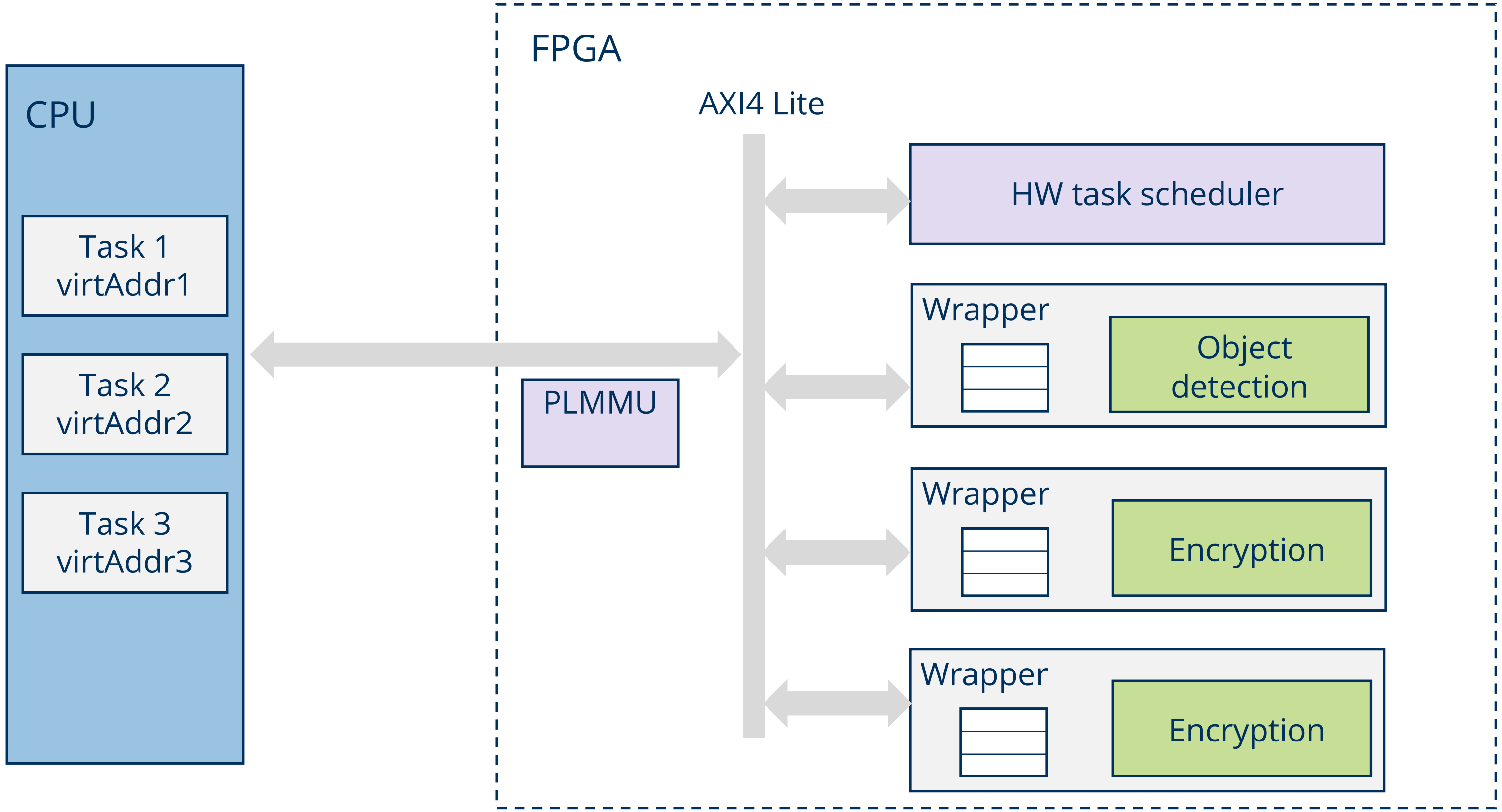
- Vitis HLS
- Input: Task ID, accelerator type, priority
- Chooses accelerator and updates the translation table

Priority queue for each accelerator

Memory-mapped access of hardware accelerators



Memory-mapped access of hardware accelerators

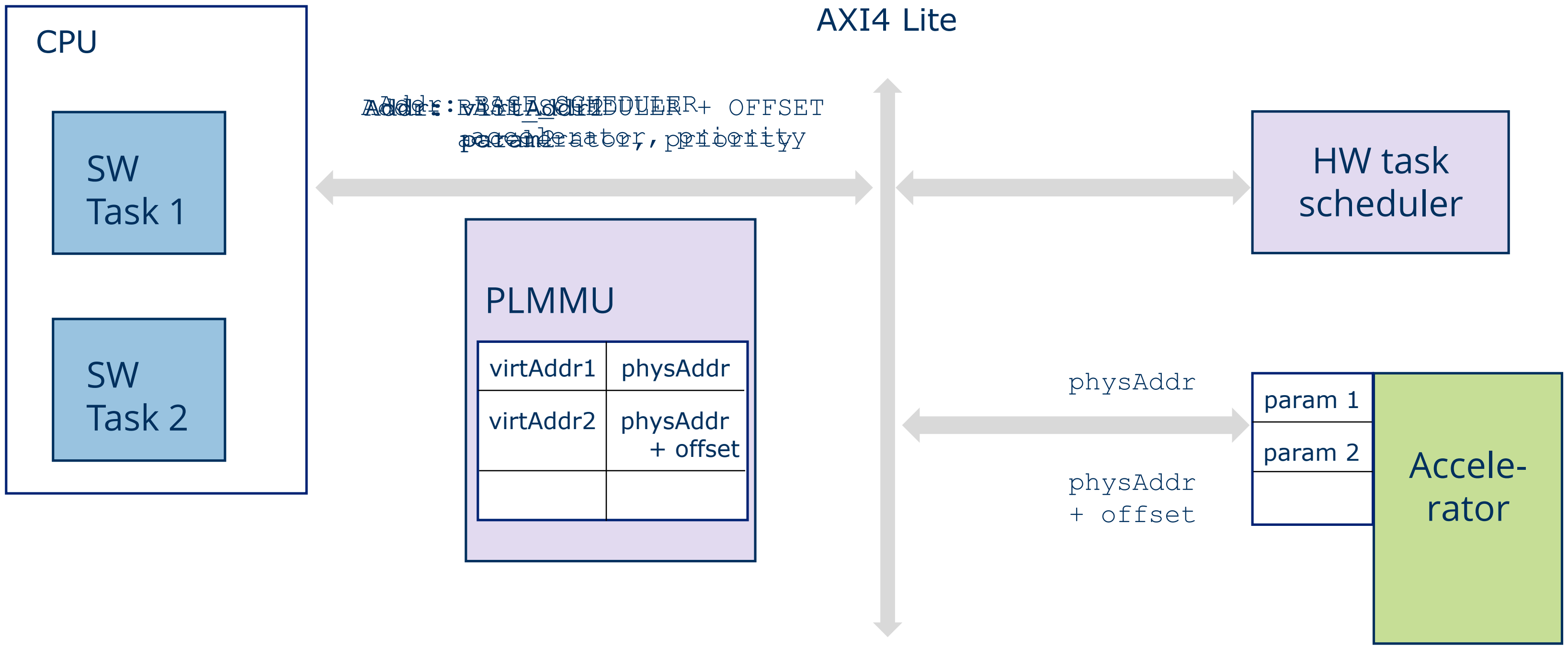


Memory-mapped access of hardware accelerators

l4rec.io

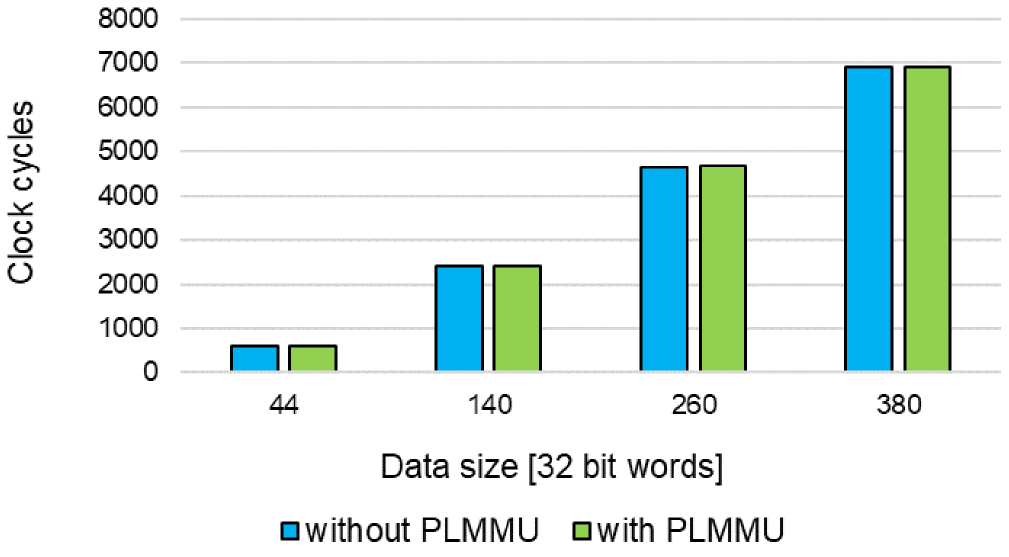
```
Io.hw.add_devices(function()
        hw_scheduler = Io.Hw.Device(function()
                Resource.regs = Io.Res.mmio(0xA0000000, 0xFFFFFFFF)
                Resource.irq1 = Io.Res.irq(121);
                                …
        end);

        task1 = Io.Hw.Device(function()
                Resource.regs = Io.Res.mmio(BASE_SCHEDULER, BASE_SCHEDULER + OFFSET)
                Resource.regs = Io.Res.mmio(BASE_VIRT1, HIGH_VIRT1)
        end);

        task2 = Io.Hw.Device(function()
                Resource.regs = Io.Res.mmio(BASE_SCHEDULER + OFFSET,
                                            BASE_SCHEDULER + 2 * OFFSET)
                Resource.regs = Io.Res.mmio(BASE_VIRT2, HIGH_VIRT2)
        end);
        …
End)
```

Hardware task scheduler



Evaluation

- Hardware accelerators: Encryption
- PLMMU adds negligible overhead to the communication between software task and hardware accelerator
- For a First Come First Serve strategy, the hardware task scheduler requires on average 169 clock cycles at 100 MHz.



Conclusion

Memory-mapped access controlled by a PLMMU and a hardware task scheduler

Advantages
- Prevention of unauthorized access
- Shared usage of hardware accelerators
- Preservation of priorities
- Latency reduction compared to a software approach

Disadvantage
- Overhead

Future work

- Static and / or dynamic priorities
- Virtualization of interrupts

Questions ?

Remarks

Discussion