OpenAMP

Concept, Technology and Practice

DING Meng, WANG Qi

Foundational Technologies, Siemens

SIEMENS

Outline

- OpenAMP concepts
- Proof-of-concept for industrial use cases
- Key technologies
- Summary

OpenAMP Concepts

SIEMENS

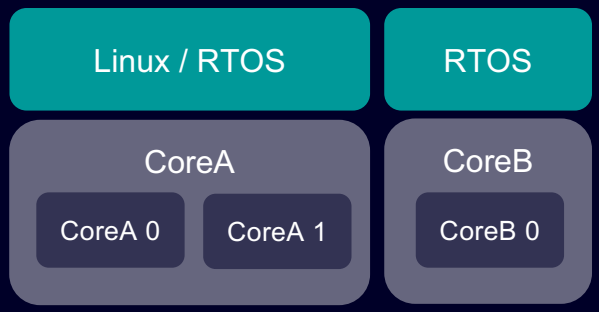
SMP vs AMP for Multicore Processing

Symmetric Multi-processing (SMP)

A single instance of an OS manages all CPUs simultaneously.

V.S.

Asymmetric Multi-processing (AMP)



Separate instances of OS run on different CPUs.

By leveraging specialized processors for specific tasks, AMP systems can achieve superior performance and reduce resource conflicts for those tasks compared to SMP systems.

AMP use cases:

Needs for multiple environments with different characteristics:

- Real-time (RTOS) and general purpose (i.e. Linux)
- Safe/Secure environment and regular environment
- GPL and non-GPL environments

Integration of code written for multiple environments:

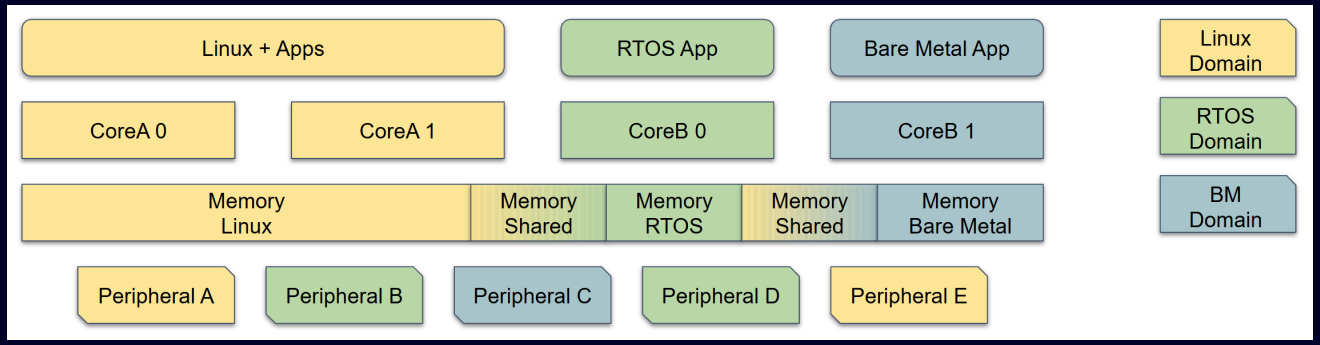
- Legacy OS and new OS

SIEMENS

OpenAMP for AMP Solutions

OpenAMP provides standards, runtime libraries and tooling built on top of existing open-source projects to simplify runtime collaboration, including:

- Resource assignment
- Runtime control (Remoteproc)
- IPC: Inter-processor communication (RPMsg)



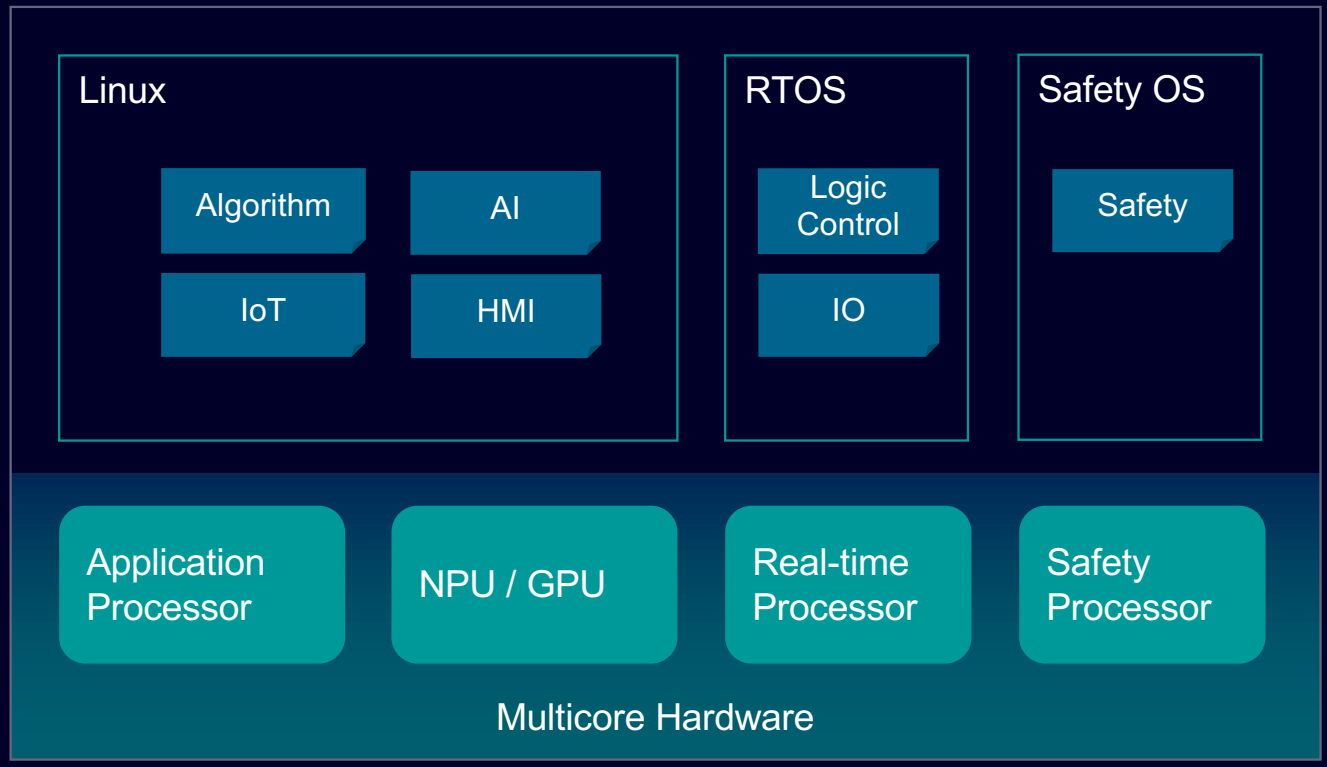
SIEMENS

Proof-of-Concept for Industrial Use Cases

SIEMENS

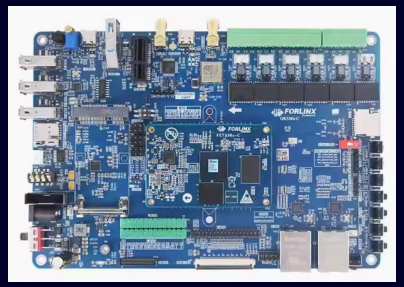
Industrial Use Cases on AMP Architecture

- Linux and RTOS on one device enhances flexibility for industrial applications and use cases.
- Example domains, e.g., industrial control, automotive, robotics, health.



SIEMENS

Example of Different Hardware for AMP

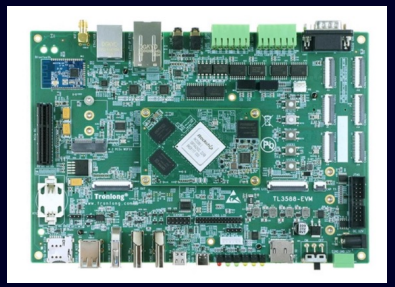


Allwinner T536
on Forlinx OK536N-C

Linux	RTOS
4 x ARM Cortex-A55 @ 1.6GHz	RISC-V @ 600MHz

Linux & RTOS on separated processers.

(Selected for PoC)



Rockchip RK3588
on Tronlong TL3588-EVM

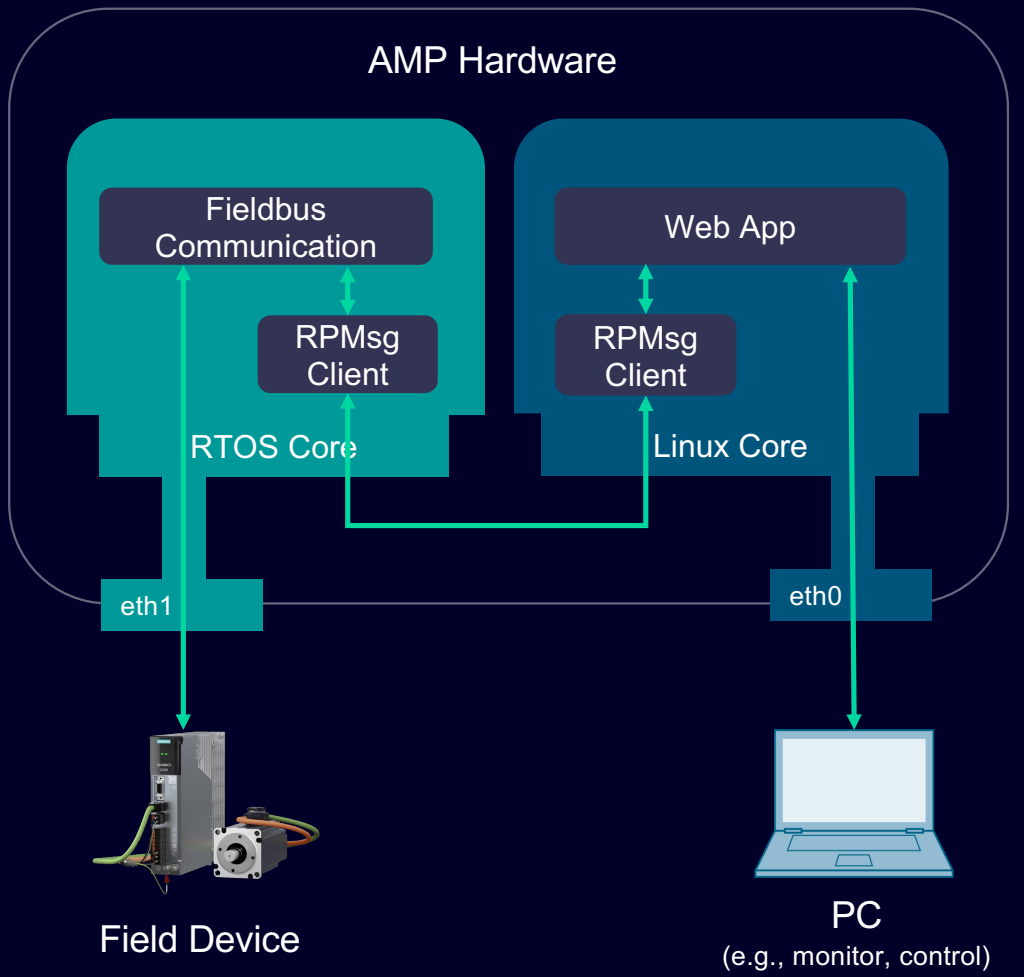
Linux		RTOS	Not Used
4 x ARM Cortex-A55 @ 1.8GHz	4 x ARM Cortex-A76 @ 2.4GHz		3 x ARM M0 @ 200MHz

Linux & RTOS on separated cores of same processor.

SIEMENS

Proof-of-Concept of End-to-end Application

- AMP system setup
  Linux and RTOS deployed on selected hardware and communicates with each other.
- Fieldbus communication on RTOS
  Application communicates with field device on RTOS side.
- Applications on Linux
  User can use Web App on the PC to monitor and control the servo.



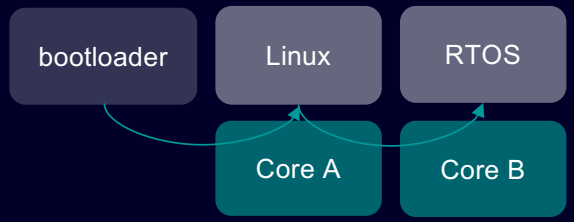
SIEMENS

Key Technologies

SIEMENS

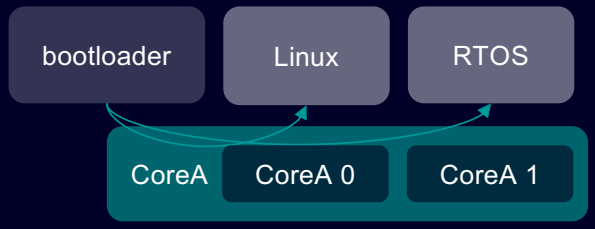
Booting Options for Linux and RTOS

Option 1
(Used on Allwinner T536)



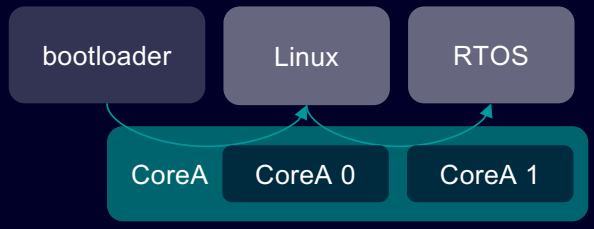
- Bootloader boot Linux when power-on.
- RTOS is booted from Linux via remoteproc.

Option 2
(Used on Rockchip RK3588)



- Bootloader boot RTOS and Linux one by one when power-on.
- Customization is needed in bootloader.

Option 3
(Supported by openEuler MICA)



- Bootloader boot Linux when power-on.
- RTOS is booted from Linux via IPI (Inter-Processor Interrupt).

Findings:

- Single-Point-of-Failure: if the system only rely on one master processor to manage all slave processors, when the master fails, it's possible that the entire system can be non-operational.
- Secure boot mechanisms for both Linux and RTOS need to be considered in industrial cases

SIEMENS

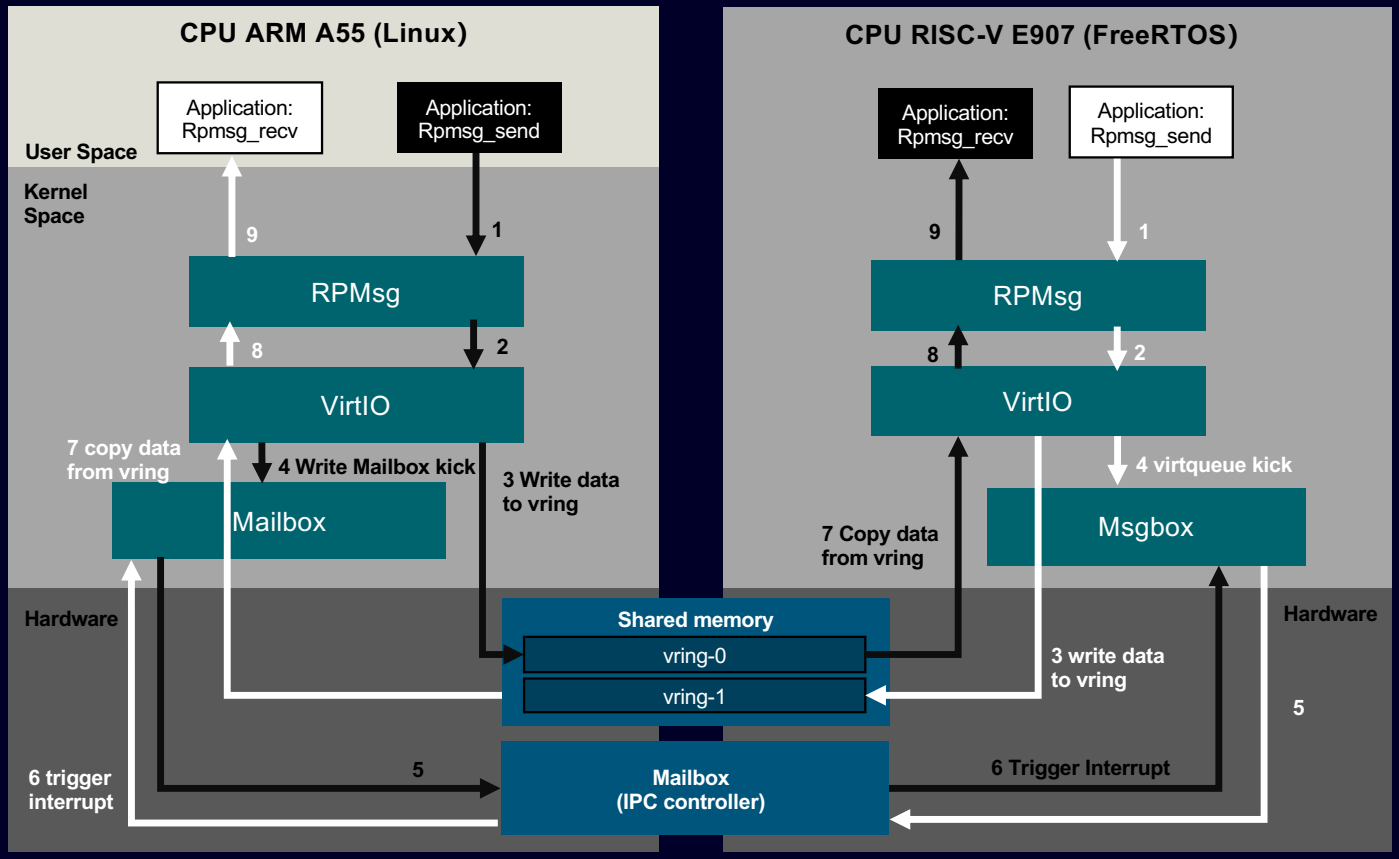
Communication between Linux and RTOS

Inter-Processor Communication (IPC):

- IPC between Linux and RTOS based on RPMsg.
- VirtIO is for hardware abstraction and I/O virtualization.
- Mailbox and shared-memory are required for control and data exchange on the hardware.
- Standardization of the IPC is promoted by the OpenAMP project.

Findings:

- RPMsg based on low-level memory exchange is not easy to be used by high-level applications.
- Limitation of RPMsg communication, e.g., message size.



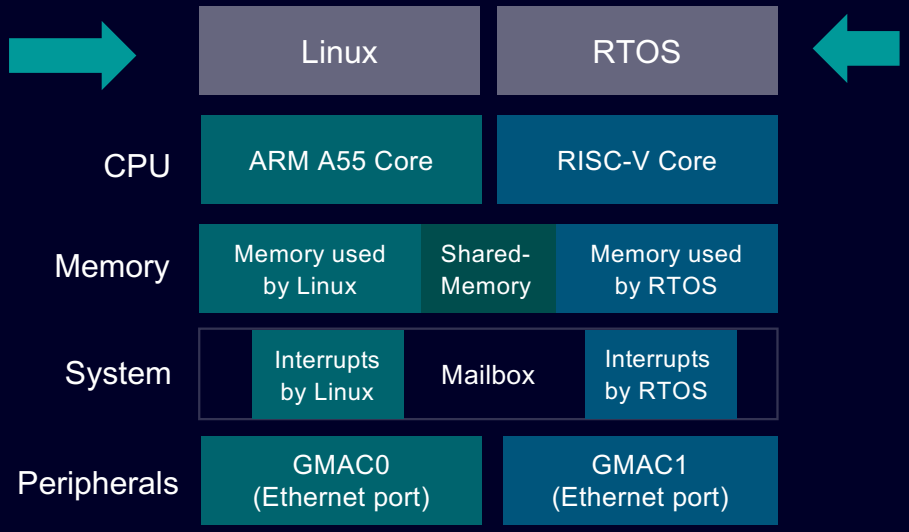
RPMsg Communication Data flow

SIEMENS

Resource Allocation used in PoC

Resource Configuration for Linux in Device Tree

```
reserved-memory {
    e907_dram_reserved: e907_dram@60000000 {
        reg = <0x0 0x60000000 0x0 0x4000000>;
        no-map;
    };

    rv_vdev0buffer: vdev0buffer@42400000 {
        compatible = "shared-dma-pool";
        reg = <0x0 0x42400000 0x0 0x40000>;
        no-map;
    };
    rv_vdev0vring0: vdev0vring0@42440000 {    …    };
    rv_vdev0vring1: vdev0vring1@42442000 {    …    };
};

msgbox@3003000 {
    #mbox-cells = <0x01>;
    reg = <0x00 0x3003000 0x00 0x1000 ...>;
    interrupts = <0x00 0x00 0x04 0x00 0xd7 0x04 0x00 0xb7 0x04>;…};

&gmac0 {
    phy-mode = "rgmii";
    pinctrl-0 = <&gmac0_pins_default>;
    ...    };

&soc {
    e907_rproc: e907_rproc@1a00000 {
        mboxes = <&msgbox 4>, <&msgbox 6>;
        memory-region = <&rv_vdev0buffer>, <&rv_vdev0vring0>...
        fw-region = <&e907_mem_fw>
        share-irq = "e907";
        ...    };
}

e907_mem_fw: e907_mem_fw@46000000 {/*boot0 & uboot0 load elf addr*/
    reg = <0x0 0x46000000 0x0 0x00200000>;
};
```



Resource Configuration for FreeRTOS in source code

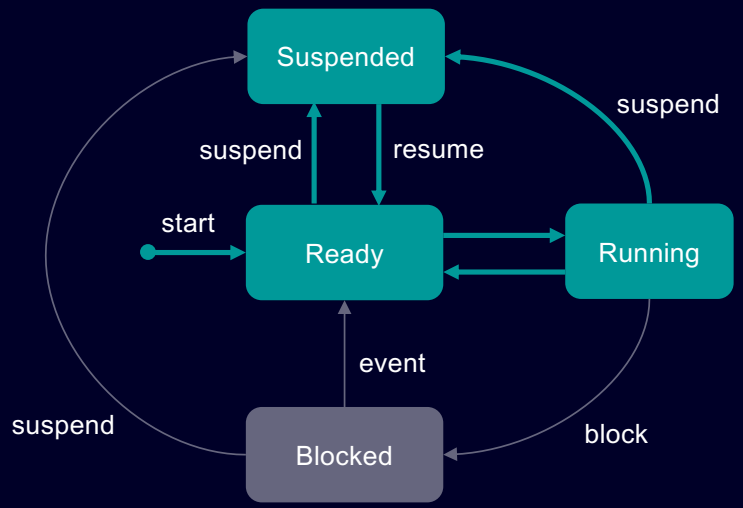
```
CONFIG_ARCH_START_ADDRESS=0x60000000
CONFIG_ARCH_MEM_LENGTH=0x4000000
...
config VDEV_BUF_RESERVED_MEM_ADDR
    default 0x42400000
config VDEV_BUF_RESERVED_MEM_SIZE
    default 0x40000
config VDEV_VRING0_RESERVED_MEM_ADDR
    default 0x42440000
config VDEV_VRING0_RESERVED_MEM_SIZE
    default 0x2000
config VDEV_VRING1_RESERVED_MEM_ADDR
    default 0x42442000
...
#define MSGBOX_A27L2   1/* correspond to msgbox driver MSGBOX_ARM*/
{
    .vring_ipi = {
        .info = {
            .remote_id = MSGBOX_A27L2,
            .read_ch = CONFIG_MBOX_CHANNEL,
        },    ...    },    };
...
{.port = 1, .type = SUNXI_ETH_TYPE_DWMAC,
        .io_base = SUNXI_ETH1_PBASE, .syscfg_base = SUNXI_ETH1_SYSCFG,
        .gpio = GPIOJ(16), .mux = 5, .driv_level = GPIO_DRIVING_LEVEL3,
        .phy_rst = GPIOB(8), .phy_rst_delay = { 0, 10000, 150000 },
        ...},}
...
static struct sunxi_resource_table __resource resource_table = {
    .rvdev_shm = {...},
    .vdev = {...},
    .vring0 = {...},
    .vring1 = {...},
    ...
}
```

- Resources need to be assigned to both Linux and RTOS before building the image.
- Hardware drivers (e.g., GMAC) need to be supported on both side.

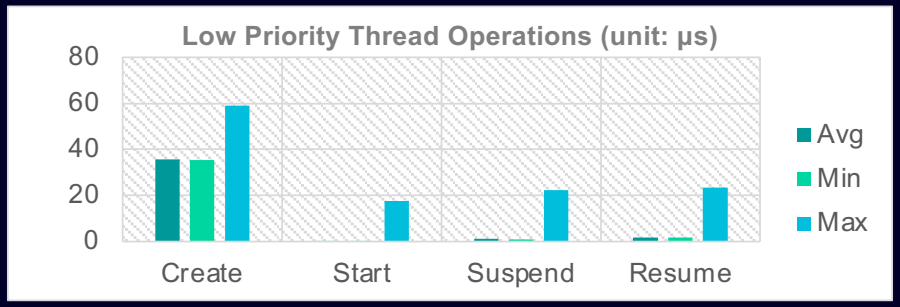
SIEMENS

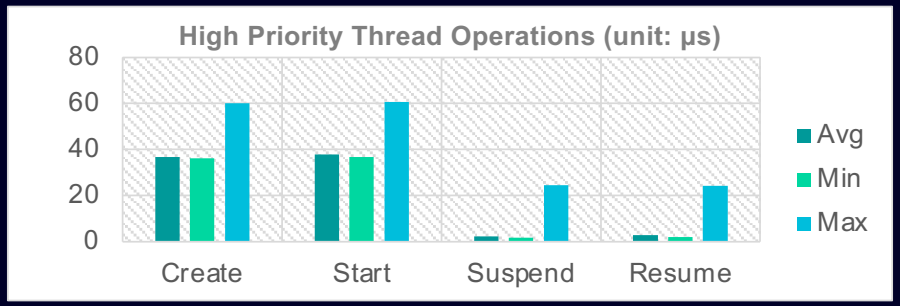
Performance Test (1/2)
Task Scheduling Performance in FreeRTOS

Task States & Scheduling in FreeRTOS



Latency of Thread Operations in FreeRTOS





Test setup:

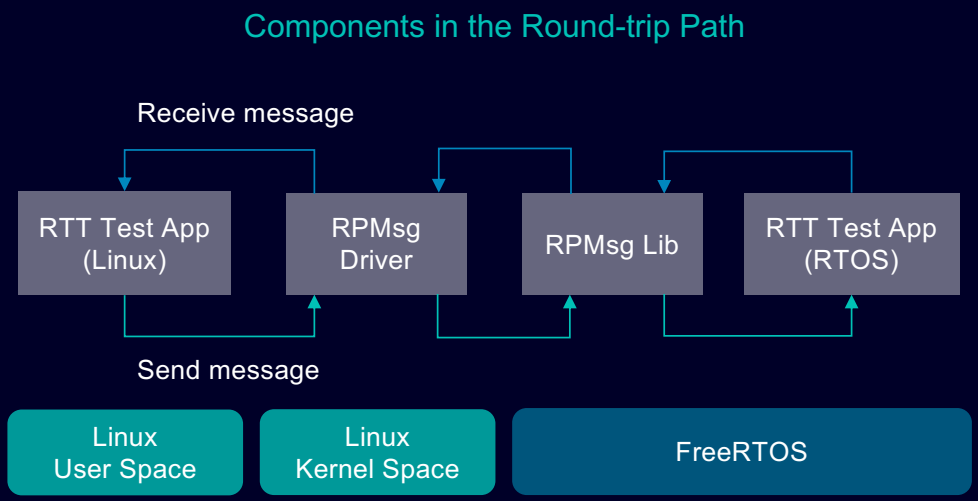
- Measure latencies of thread task operations (e.g., create, start, suspend, resume) of low priority and high priority threads, with 3 million iterations.
- Tests run in FreeRTOS on RISC-V core of the hardware (Allwinner T536) based on rtos-benchmark tool from zephyr.

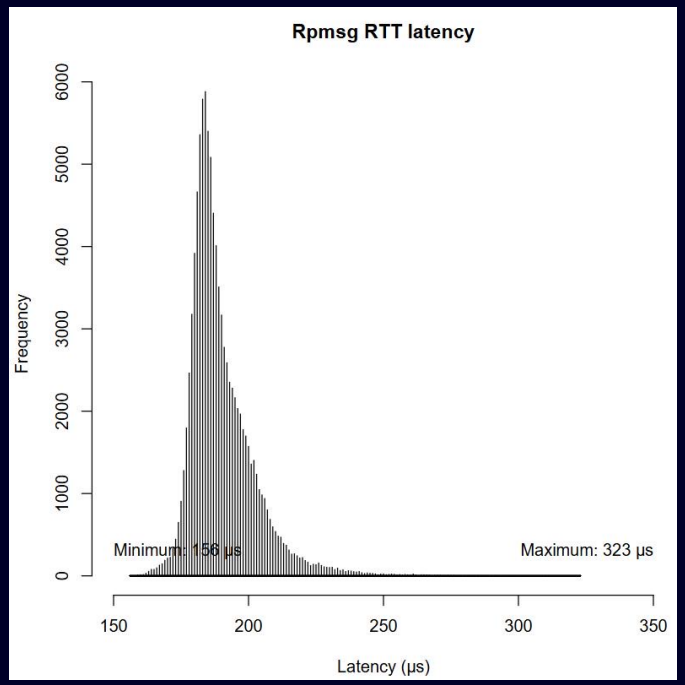
Test results:

- High priority thread has higher latency on start operation, because:
  - Start a low priority thread will stay in ready state until higher tasks release the resources, the time from ready to running is not included.
  - Start a higher priority thread will immediately preempt resources and cause context switch from ready state to running state.

SIEMENS

Performance Test (2/2)
RPMsg Communication Performance between Linux and FreeRTOS





Test setup:

- Measure round-trip time (RTT) of RPMsg communication between Linux and FreeRTOS.
- Message size 64 bytes, with 10 million iterations.
- Tested on hardware Allwinner T536.
- No real-time tuning on Linux side.

Test results:

- Most of the round-trip latencies are below 250 µs, with minimum latency 156 µs and maximum latency 323 µs.
- The latency can be improved with additional real-time tuning on Linux side.

SIEMENS

Summary

Summary

- AMP (Asymmetric Multi-processing) is suitable for multiple environments on one device, e.g., real-time / safety together with general-purpose (i.e., Linux).
- The OpenAMP project is a community effort that is promoting and implementing AMP solutions.
- A proof-of-concept for an industrial use case is implemented on selected hardware.
- Understanding of the technologies including runtime control, IPC (inter-processor communication), resource allocation, and performance are introduced.
- Different architectures (e.g., SMP, AMP, Hypervisor) have their own suitable scenarios that need to be considered carefully according to design goals.

Thank you

DING Meng
Foundational Technologies
Siemens Ltd., China
meng.ding@siemens.com

WANG Qi
Foundational Technologies
Siemens Ltd., China
wq.wang@siemens.com

SIEMENS