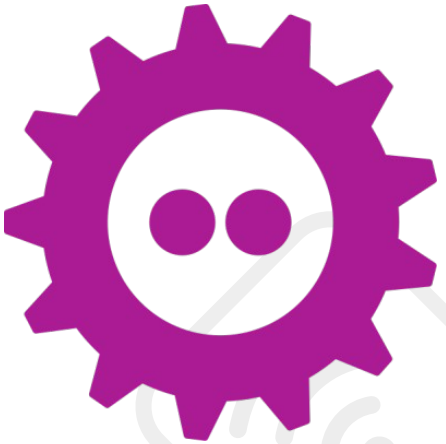
CMRX



Designing a real-time microkernel for microcontrollers

Eduard Drusa

www.linkedin.com/in/eduard-drusa

https://github.com/ventZl/cmrx.git

nopisonnope@gmail.com | FOSDEM 2025, Brussels

Who would win?

DEC PDP-7

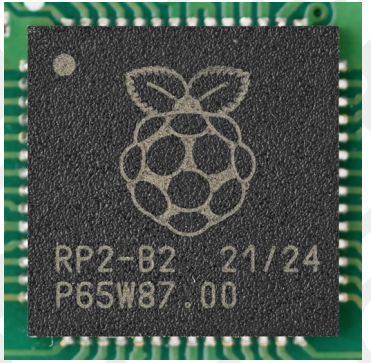
1964

$72,000

Raspberry Pi RP2040

2021

1.16 €



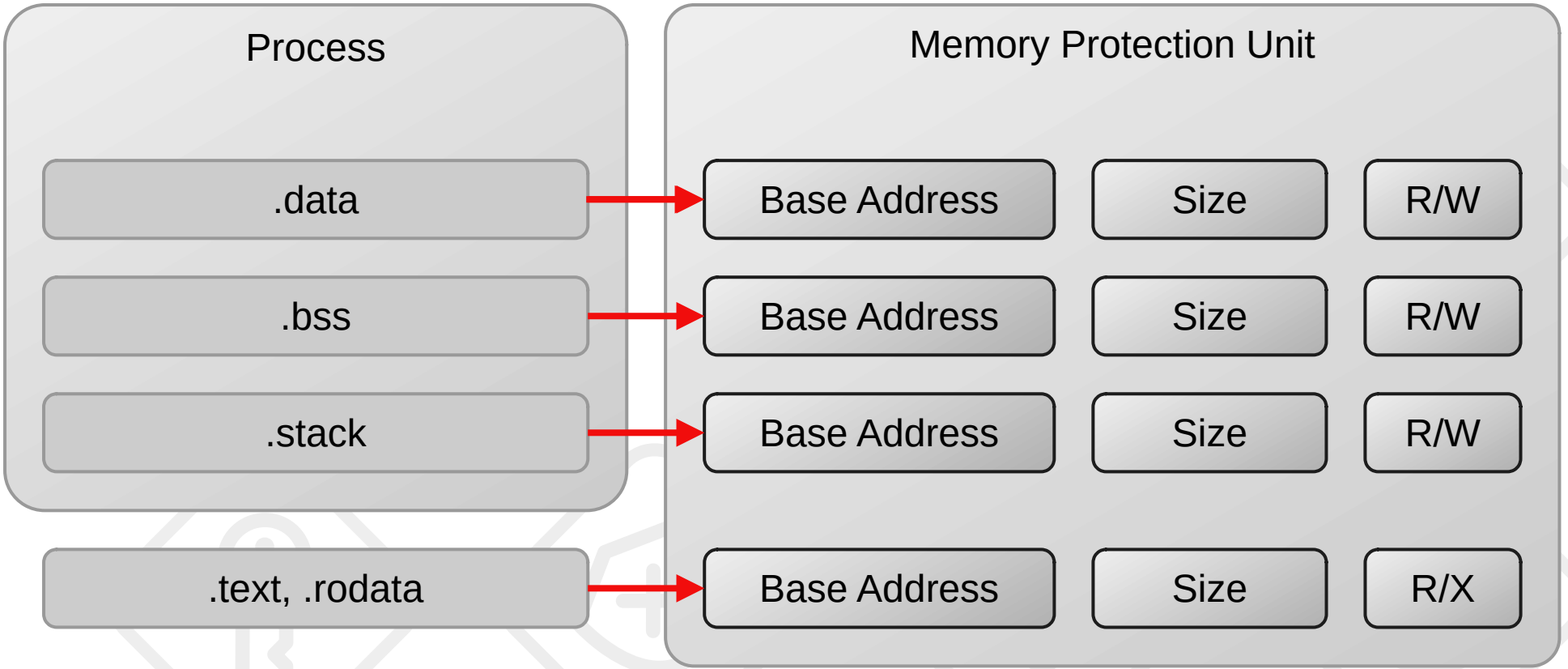
Who am I?

- Read „Principles of Operating Systems“ instead of „The Little Prince“
- In 1999, randomly spotted article: „QNX: Operating system on a single floppy“
- Nomading across industries, ending up in automotive, then switching to freelance.

The Problem

- Bulk of IoT systems runs unprotected kernels
- Microcontrollers lack memory management units
- Existing solutions are too complicated to be used
- Developers generally avoid the topic of memory isolation

Isolation Using Processes

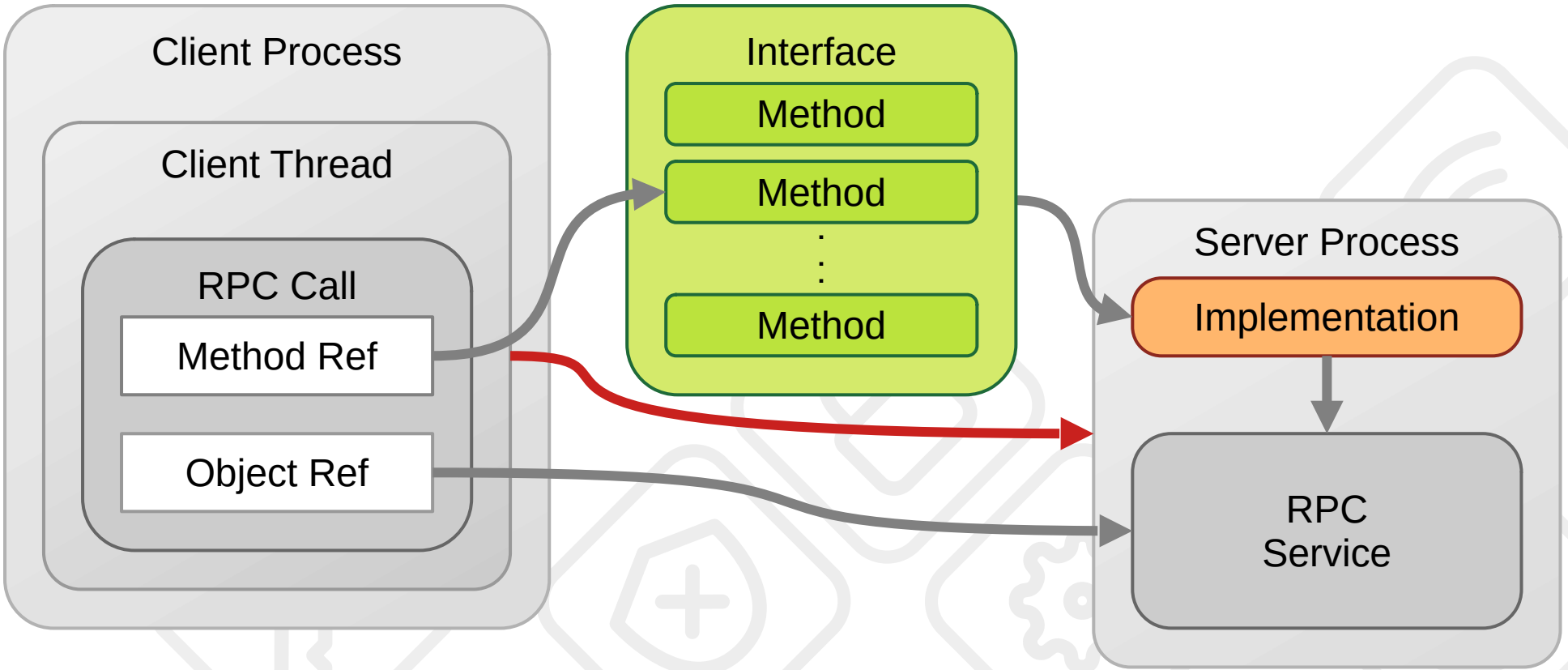


```c
#include <cmrx/application.h>
#include <pico/stdlib.h>
#include <cmrx/ipc/timer.h>

#define BUFSIZE 512

static unsigned blink_rate = 500000;
static char buffer[BUFSIZE];

static int blinky_main()
{
    gpio_init(PICO_DEFAULT_LED_PIN);
    gpio_set_dir(PICO_DEFAULT_LED_PIN, GPIO_OUT);
    while (1) {
        gpio_put(PICO_DEFAULT_LED_PIN, 1);
        usleep(blink_rate);
        gpio_put(PICO_DEFAULT_LED_PIN, 0);
        usleep(blink_rate);
    }
    return 0;
}

OS_APPLICATION_MMIO_RANGES(blinky, 0x40000000, 0x50000000, 0xd0000000, 0xe0000000);
OS_APPLICATION(blinky);
OS_THREAD_CREATE(blinky, blinky_main, NULL, 32);
```

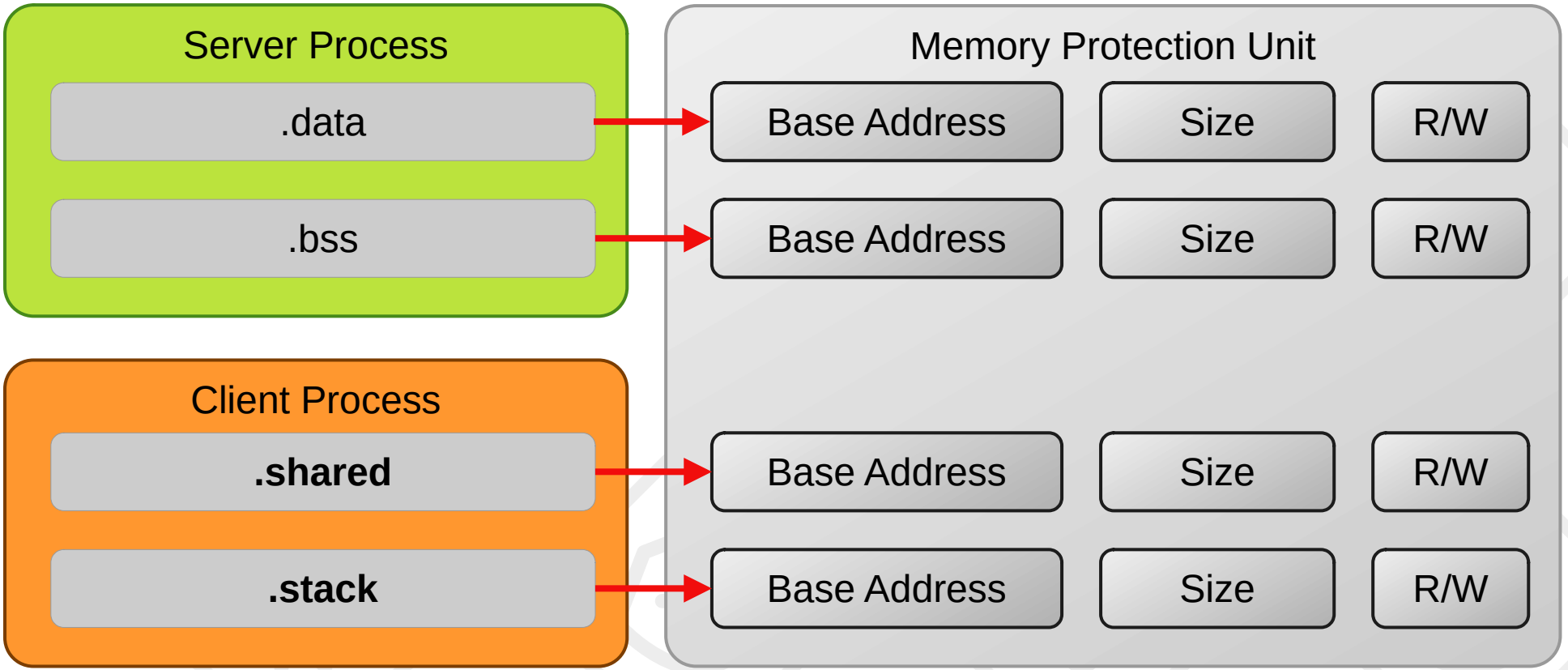
GAME OVER:
You designed the thing to be useless

nopisonnope@gmail.com | FOSDEM 2025, Brussels

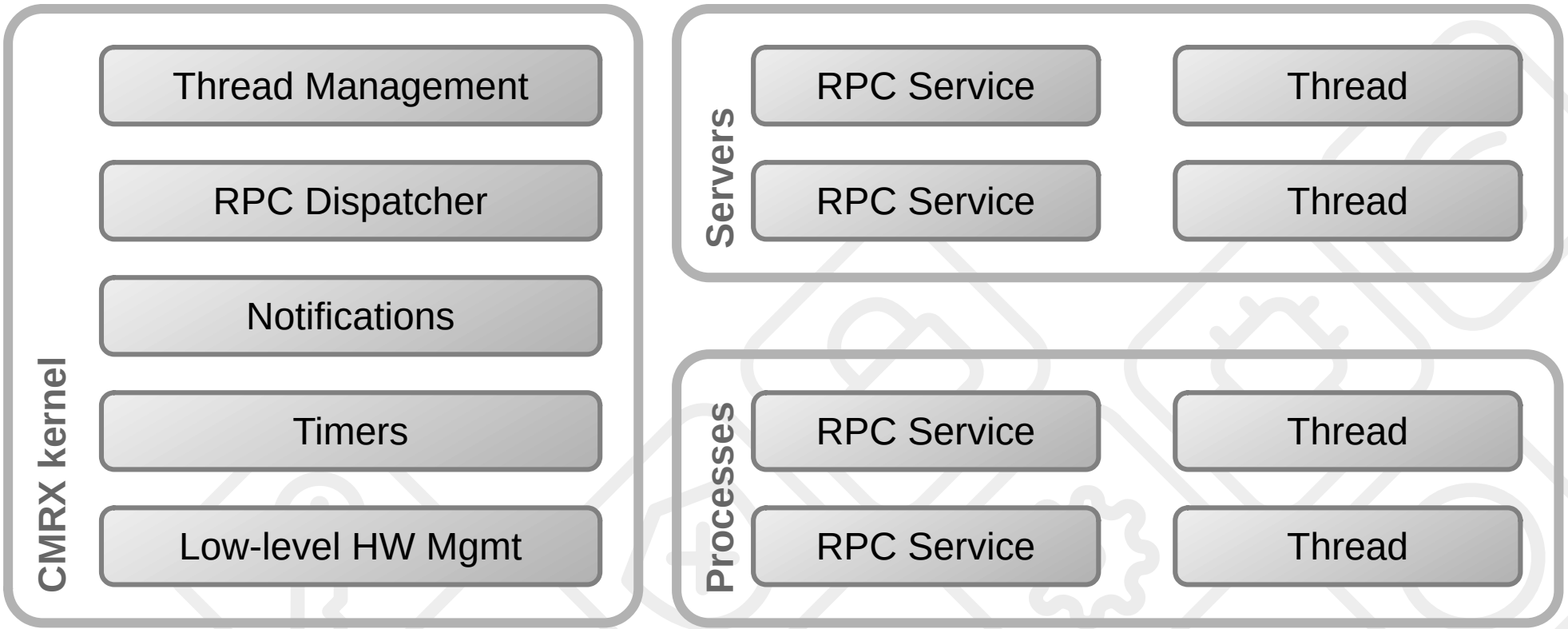
Remote Procedure Calling



RPC Enables Shared Memory

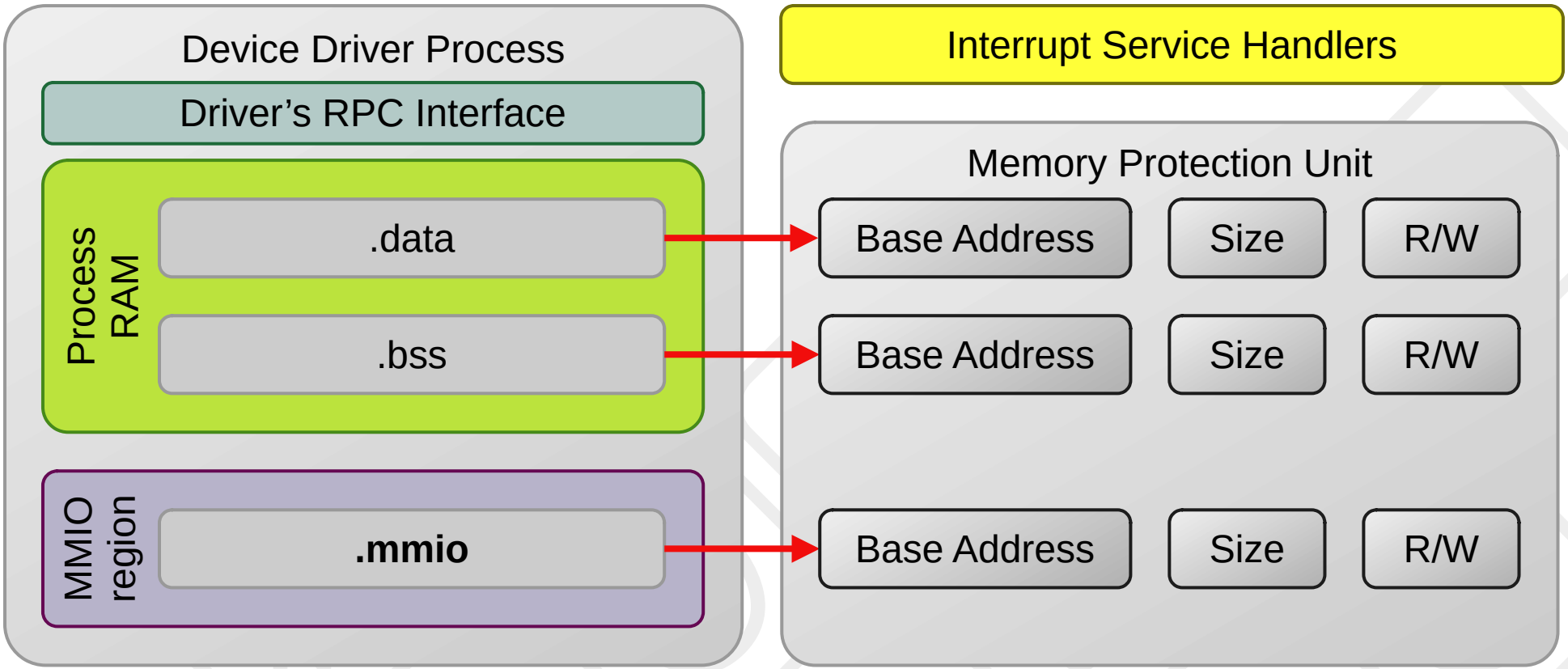


CMRX microkernel



nopisonnope@gmail.com | FOSDEM 2025, Brussels

Drivers in Userspace

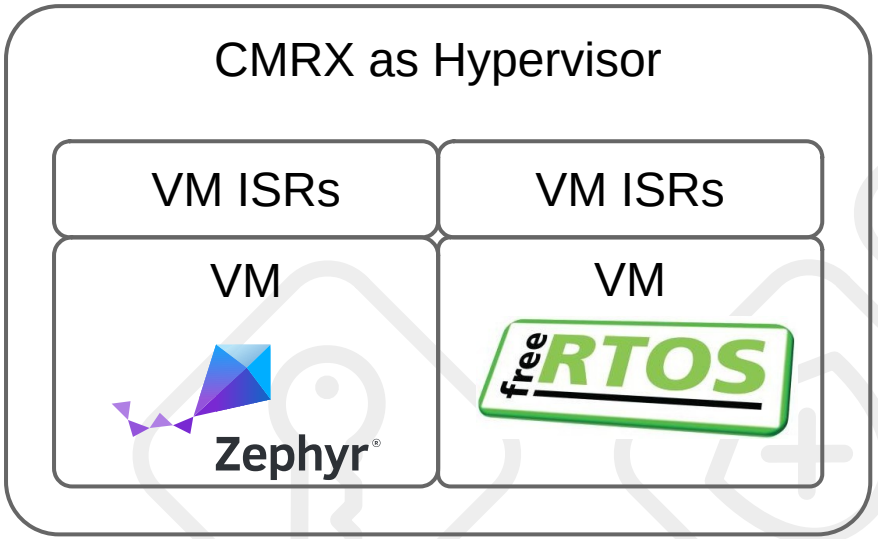


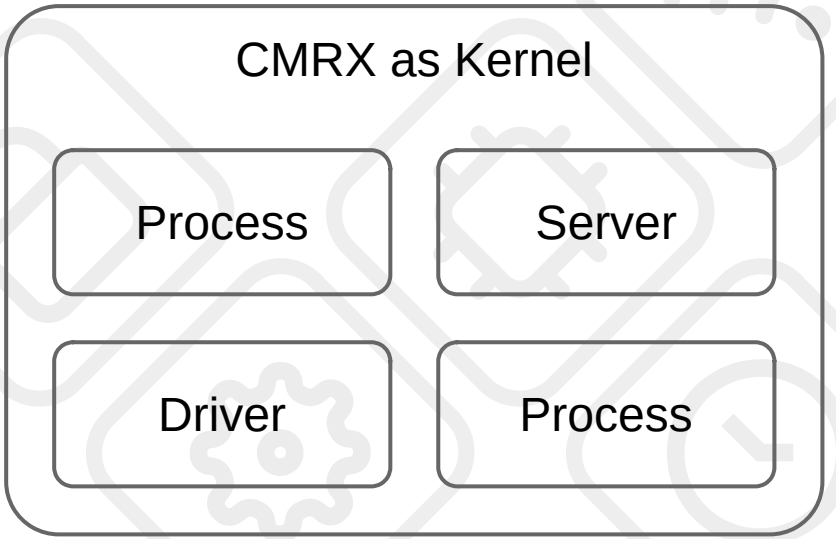
nopisonnope@gmail.com | FOSDEM 2025, Brussels

```c
#include <cmrx/application.h>
#include <pico/stdlib.h>
#include <cmrx/ipc/timer.h>

#define BUFSIZE 512

static unsigned blink_rate = 500000;
static char buffer[BUFSIZE];

static int blinky_main()
{
    gpio_init(PICO_DEFAULT_LED_PIN);
    gpio_set_dir(PICO_DEFAULT_LED_PIN, GPIO_OUT);
    while (1) {
        gpio_put(PICO_DEFAULT_LED_PIN, 1);
        usleep(blink_rate);
        gpio_put(PICO_DEFAULT_LED_PIN, 0);
        usleep(blink_rate);
    }
    return 0;
}

OS_APPLICATION_MMIO_RANGES(blinky, 0x40000000, 0x50000000, 0xd0000000, 0xe0000000);
OS_APPLICATION(blinky);
OS_THREAD_CREATE(blinky, blinky_main, NULL, 32);
```

Use cases for CMRX

- Paravirtualized Hypervisor
- Full Hypervisor
- Mixed-Criticality Systems
- Component-Based Systems





Future Work

- Implementation of capabilities and ACLs
- Clean-up of API (yes, there are useless syscalls)
- Registration of servers as syscalls
- Add support for TrustZone
- Port to RISC-V
- Finish the support for SMP

Q&A