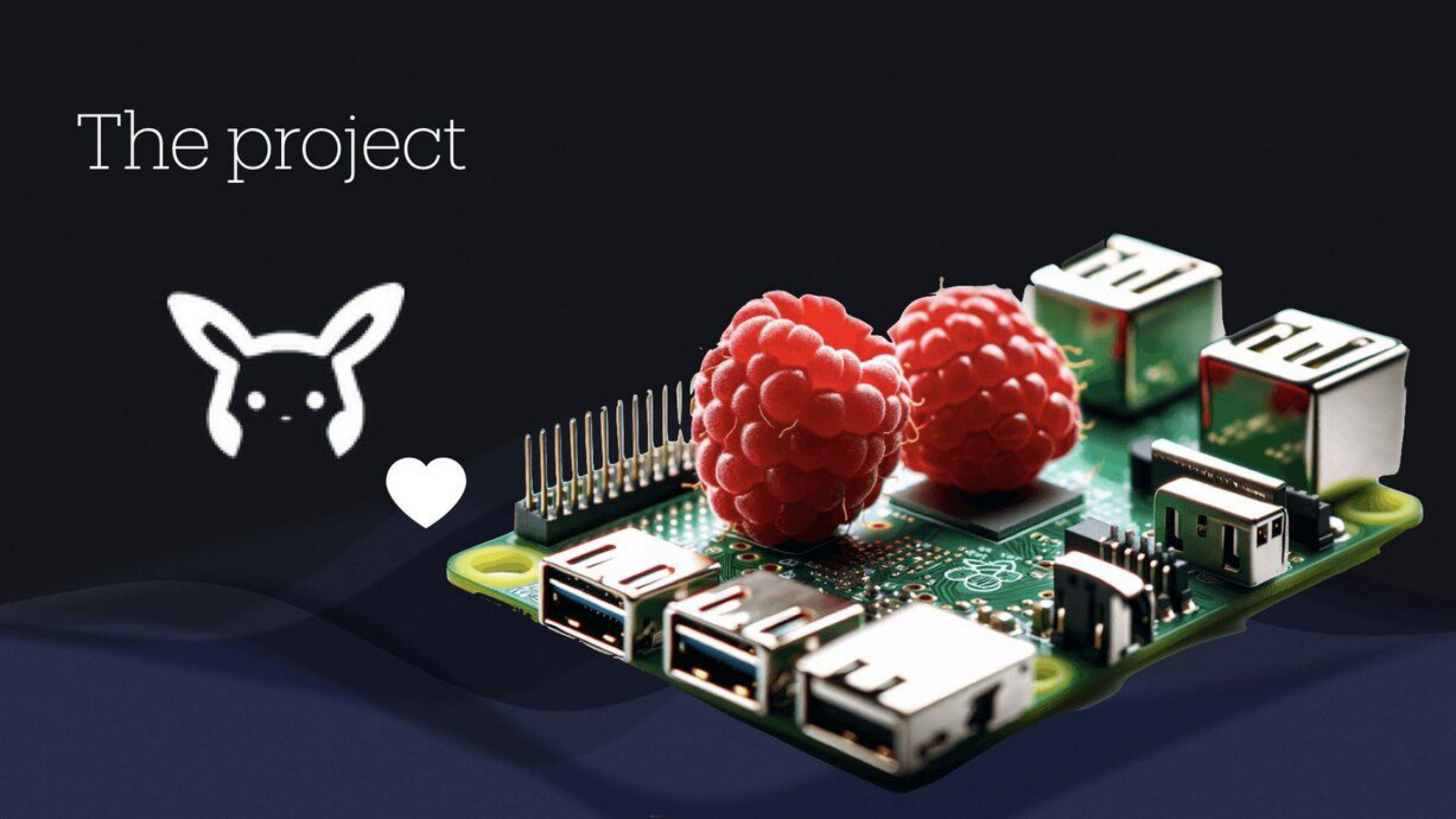
Pi-kachULM_OS

Gabriel DESFRENE

Hubert GRUNIAUX

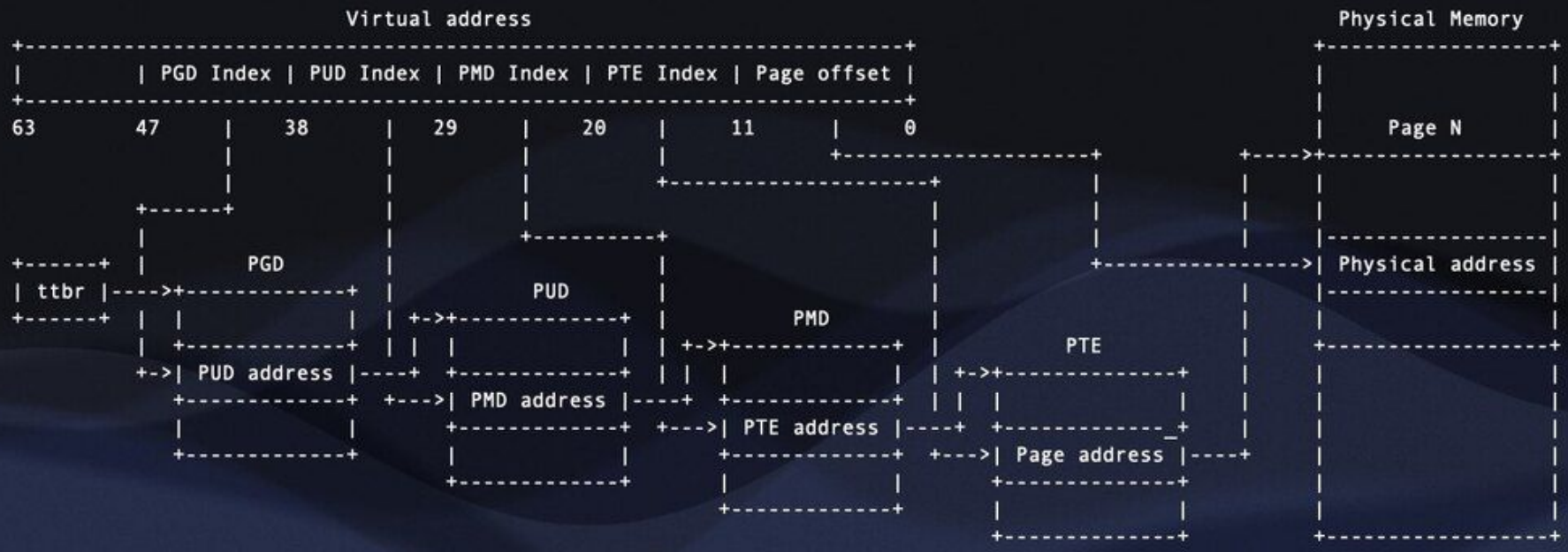
Émile Sauvat

The project



Memory

- Memory mapping



Memory

- Memory mapping

0x00000000_00000000
User Space

User ELF Program

User Heap

User Stack

0xffff0000_00000000
1:1 Mapping

0xffff1000_00000000
Video Memory

0xffff2000_00000000
Device Memory

0xffff3000_00000000
Custom Pages

0xffff4000_00000000
DMA Buffers

0xffff5000_00000000
Kernel Heap

0xffffa000_00000000
Ram FS

0xfffff000_00000000
Kernel Stack

Kernel Space

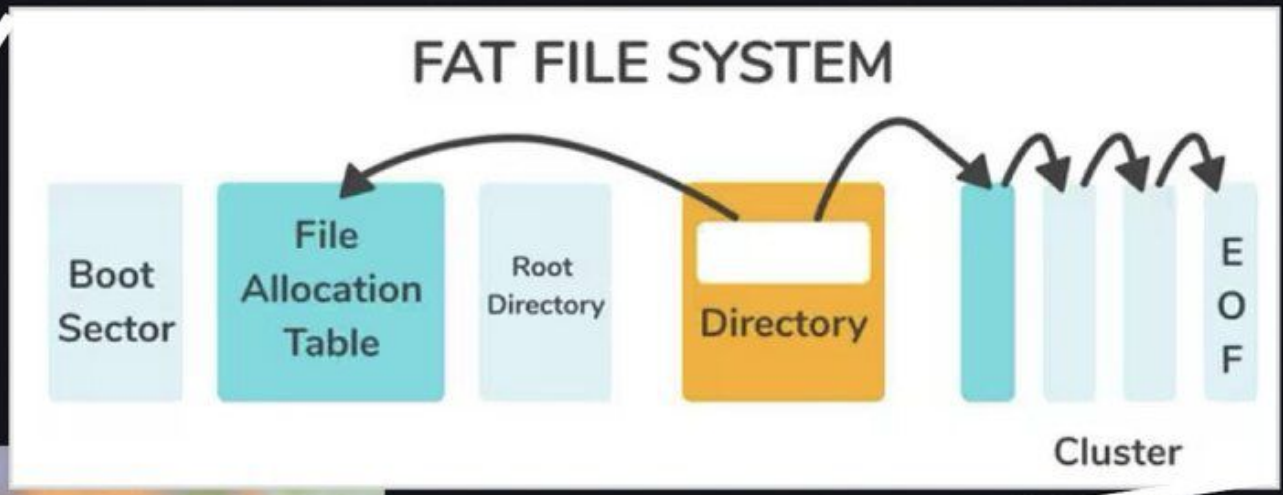
Memory

- Dynamical memory allocator
  - Linked list of memory block
  - kmalloc to create and allocate a memory block
  - kfree to deallocate a memory block
- Page Allocator
  - Binary heap tree to map the physical memory
  - Allocation in little time
  - Unallocate Pages

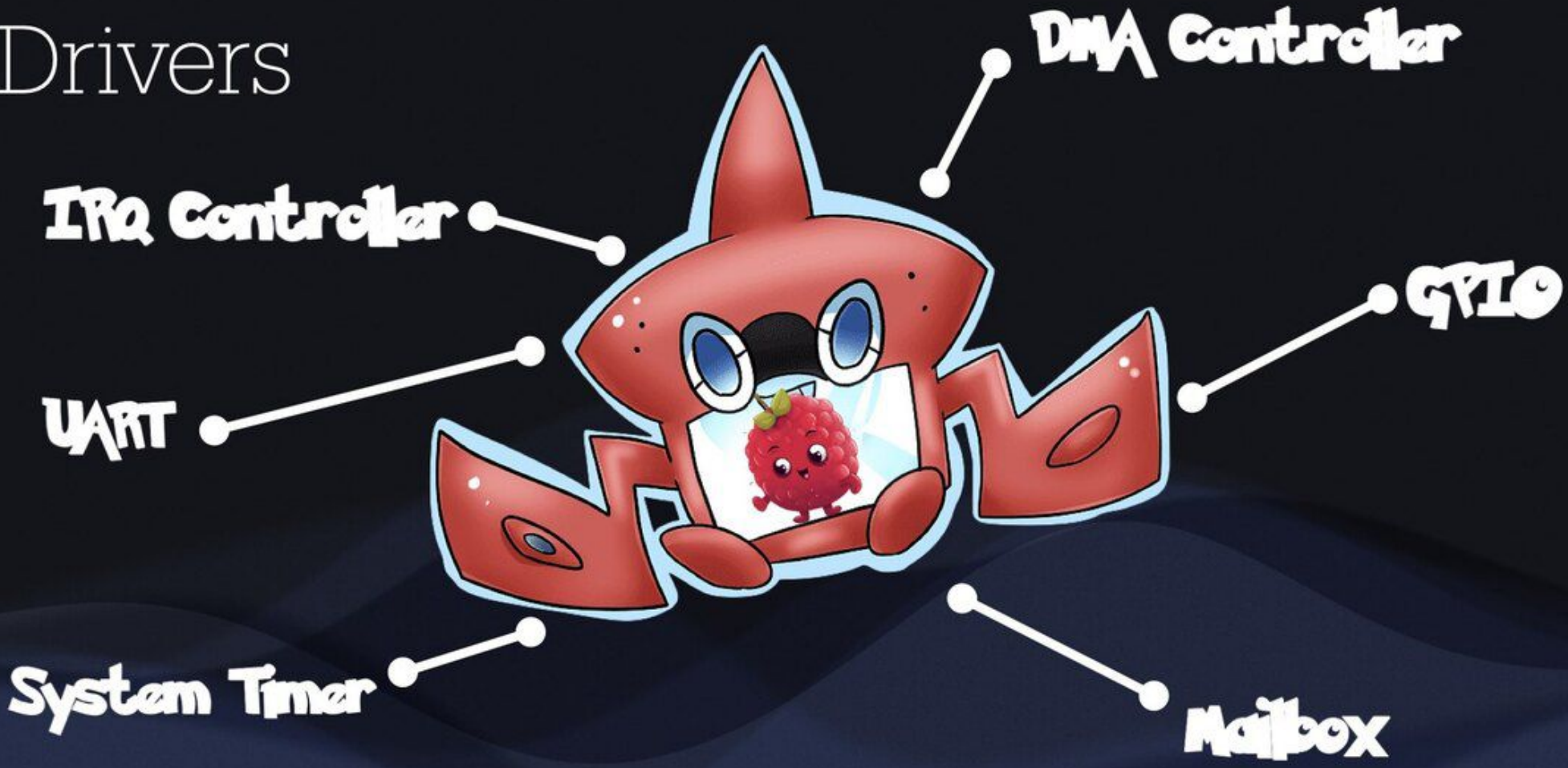
Binary heap tree

Memory

- A file system



Drivers

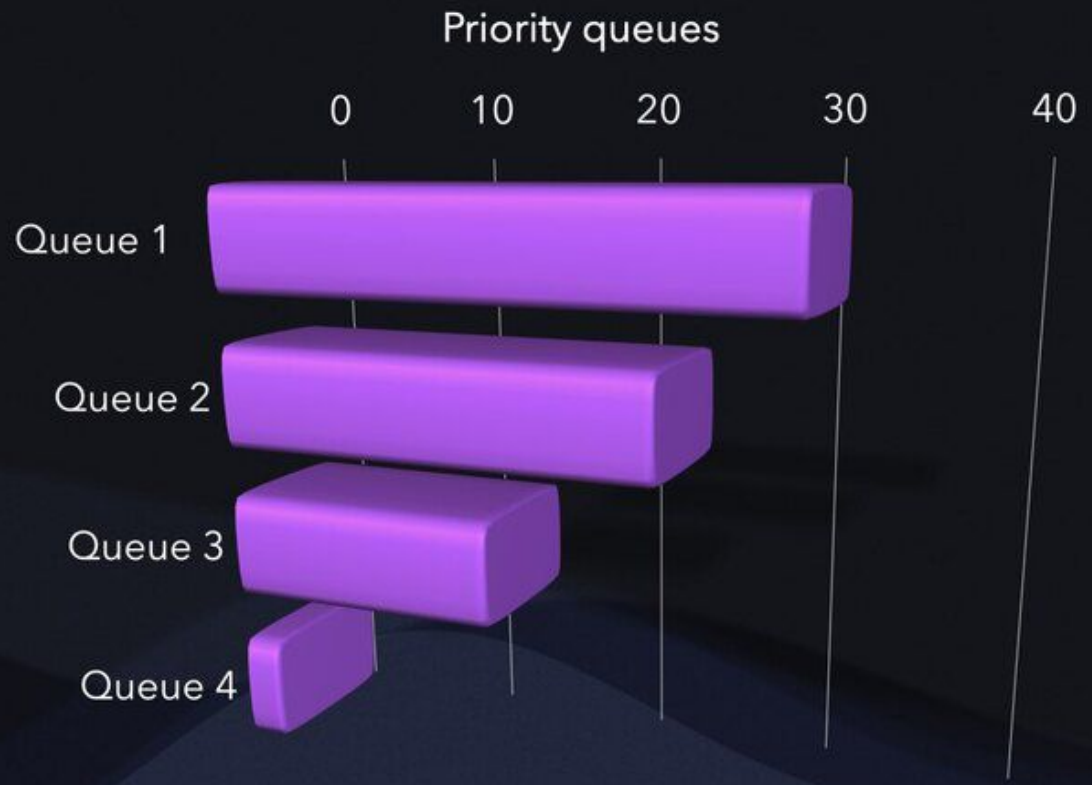

DMA Controller
IRQ Controller
GPIO
UART
System Timer
Mailbox

Task (our process & threads)

- User space processes and threads
  - Loaded from ELF file
  - Own virtual memory
  - Own files and windows
  - Preemptive scheduler !
- Kernel threads
  - Get a kernel function pointer
  - Kernel virtual memory
  - Preemptive scheduler !

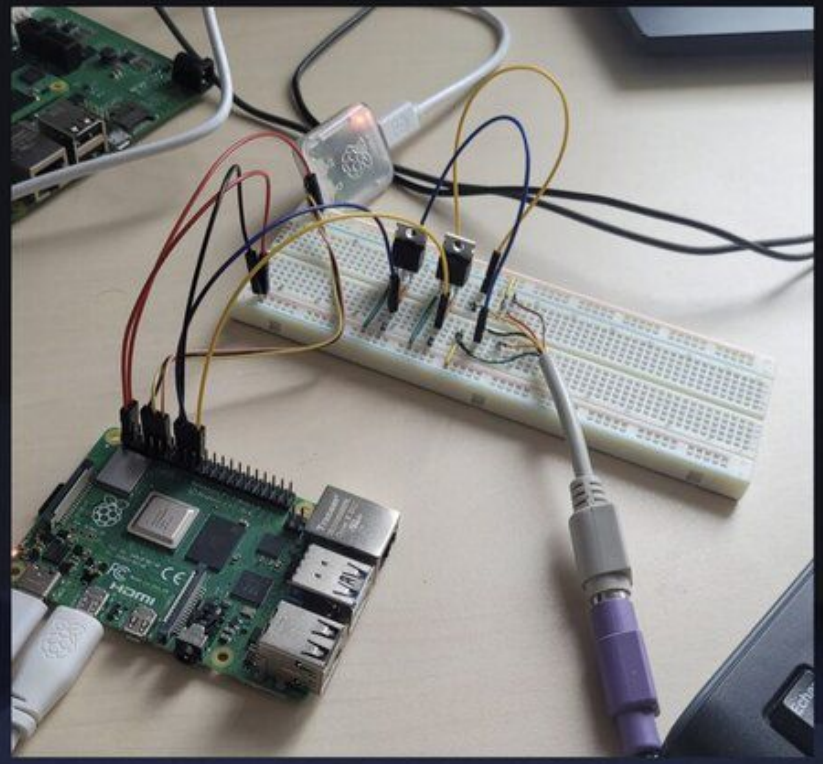
User space

- Process management



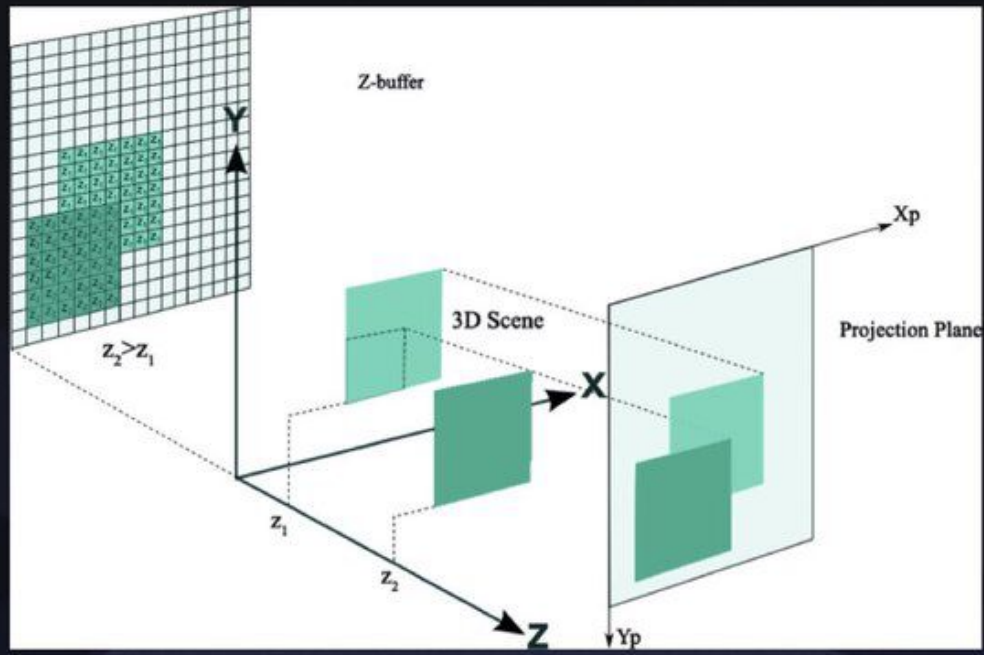
Devices

- PS2 keyboard
  - Looking like AZERTY but in fact QWERTY
  - A lot of transistors (fake) engineering
  - Sometimes a few parity issues



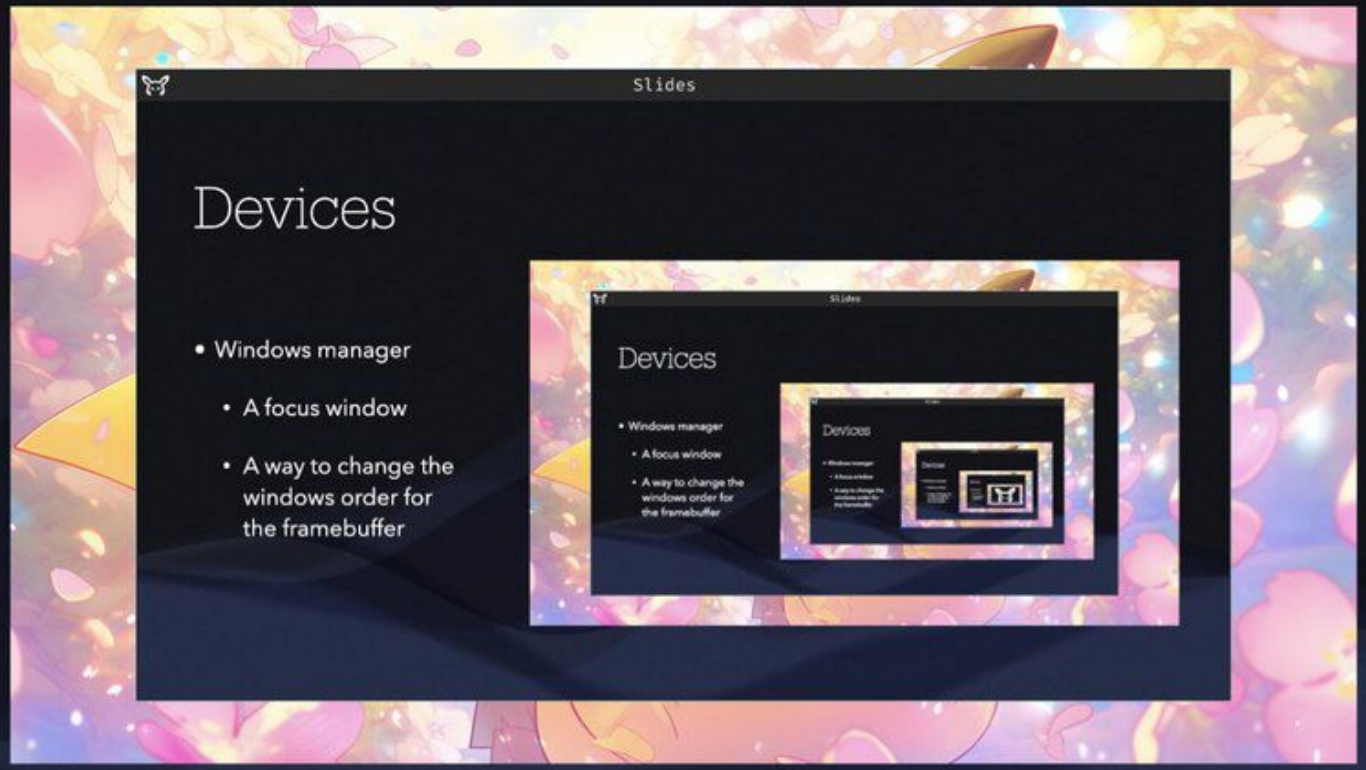
Devices

- *Framebuffer*
  - Change every pixel only once
  - Ordered list of window



Devices

- Windows manager
  - A focus window
  - A way to change the windows order for the framebuffer



```c
#include <sys/window.h>

int main() {
    sys_window_t* window = sys_window_create("My Window!", ...);
    sys_gfx_clear(window, 0xFF0000);

    bool should_close = false;
    while (!should_close) {
        sys_message_t msg;
        sys_wait_message(&msg);
        switch (msg.id) {
        case SYS_MSG_CLOSE:
            should_close = true;
            break;
        case SYS_MSG_RESIZE:
            sys_gfx_clear(window, 0xFF0000);
            break;
        }
    }

    sys_window_close(window);
    return 0;
}
```