Lab 3

思考题

Thinking 3.1

在 `UVPT` 上映射自己的页表，并使其具有只读权限。

Thinking 3.2

`elf_load_seg` 在 `load_icode` 中被调用，`data` 参数实参 `e`，而 `load_icode` 中的 `e` 为 `env_create` 中刚申请并设置的 `struct Env *e`

```
struct Env *env_create(const void *binary, size_t size, int priority) {
    struct Env *e;
    /* Step 1: Use 'env_alloc' to alloc a new env, with 0 as 'parent_id'. */
    /* Exercise 3.7: Your code here. (1/3) */
    if (env_alloc(&e, 0) < 0) {
        return NULL;
    }

    /* Step 2: Assign the 'priority' to 'e' and mark its 'env_status' as runnable. */
    /* Exercise 3.7: Your code here. (2/3) */
    e->env_pri = priority;
    e->env_status = ENV_RUNNABLE;

    /* Step 3: Use 'load_icode' to load the image from 'binary', and insert 'e' into
     * 'env_sched_list' using 'TAILQ_INSERT_HEAD'. */      Red, 上周 • first
    /* Exercise 3.7: Your code here. (3/3) */
    load_icode(e, binary, size);
    TAILQ_INSERT_HEAD(&env_sched_list, e, env_sched_link);

    return e;
}
```

这个参数就是用于在加载二进制文件的镜像中说明，这个被加载的二进制镜像是属于进程e的。

不能没有这个参数。这个参数增加了函数的灵活性，便于根据具体情况传入不同函数进行实现。

Thinking 3.3

- va 是否对齐
- bin_size 与 BYP2G-offset 的大小关系不同
- va + bin_size 后在页内的位置不同
- sgsize 与 bin_size 大小关系不同

Thinking 3.4

虚拟地址

Thinking 3.5

- 0 号异常 `handle_int`：`kern/genex.S`
- 1 号异常 `handle_mod`：通过 `do_tlb_mod` 实现，位于 `kern/tlbex.c`
- 2，3 号异常 `handle_tlb`：通过 `do_tlb_refill` 实现，位于 `kern/tlb_asm.S`

Thinking 3.6

```
LEAF(enable_irq)
    li      t0, (STATUS_CU0 | STATUS_IM4 | STATUS_IEc) /* 取中断使能 */
    mtc0    t0, CP0_STATUS /* 将中断使能写入 CP0_STATUS */
    jr      ra /* 返回 */
END(enable_irq)

timer_irq:
    sw      zero, (KSEG1 | DEV_RTC_ADDRESS | DEV_RTC_INTERRUPT_ACK) /* 写地址 (KSEG1
| DEV_RTC_ADDRESS | DEV_RTC_INTERRUPT_ACK) 响应时钟中断 */
    li      a0, 0 /* 传参 0 */
    j       schedule /* 跳转到 schedule 中进行调度 */
```

Thinking 3.7

设置了一个进程就绪队列，并且给每一个进程添加了一个时间片。时钟中断发生时，系统跳转到 schedule 函数，进行进程的调度，如果进程时间片消耗完则将其置于就绪队列尾部，并从头部取一个新进程设置时间片并运行。

难点分析

课下内容根据提示补全代码较为轻松，但 schedule 由于指导书描述问题在写 lab3 时出现问题，未测试出，在写 lab4 时才发现。

心得体会

经过上机，感受到不仅需要掌握当前 lab 的内容，还要掌握过去所学内容。上机时 exam 对着 hint 抄很快就过了，但是 extra 由于对 Lab2 访存相关知识还是没用完全掌握，忽略了页内偏移的存在，认为 page 的地址就是元素存储地址，导致一直没通过。同时也有一些疑问：初始化分支提供的 `include/pmap.h` 中的 `va2pa` 函数实现是否与其命名不同？而在 extra 的题面中说明已经有函数实现了 va 向 pa 的转换，这部分描述是否不正确？