1 练习题 1

内核从完成必要的初始化到用户态程序的过程中的函数调用关系主要体现在 kernel\arch\aarch64 目录中的 main.c 代码中。其中关键的步骤（省略了测试相关代码）如下：

```
1    uart_init();
2    mm_init();
3    arch_interrupt_init();
4    create_root_thread();
5    eret_to_thread(switch_context());
```

内核启动后会依次初始化 uart 模块和内存管理模块，我们在前两个实验中已经完善了这部分代码。

之后内核会通过 arch_interrupt_init 函数初始化异常向量表，然后通过 create_root_thread 创建根进程，也就是内核启动之后第一个运行的进程。在 create_root_thread 函数中，内核会创建一个主线程，eret_to_thread 则会从内核态切换到用户态。

2 代码简要说明

本次实验大多数代码都是在理解已经实现好的函数的基础上，根据实验要求在恰当的地方调用函数。下面对一些相对不那么直观的函数调用进行说明。

2.1

load_binary 函数需要先从 seg_sz 和 p_vaddr 计算需要 map 的大小，之后用 obj_alloc 函数和 pmo_init 新建合适的 pmo 对象，然后调用 memset 和 memcpy 函数进行初始化，最后设定 flags，并映射对应的 vmspace.

2.2

irq_entry.S 中异常向量表的填写和函数跳转：结合 irq_entry.h 中定义好的宏，我们直接 exception_entry 设置好这些异常向量即可。之后，对于 sync_el1t 和 sync_el1h，根据实验文档中的教学，我们直接分别 bl 到 unexpected_handler 和 handle_entry_c 即可。

2.3

handle_trans_fault 函数中，如果 $pa = 0$，即没有分配过物理页，则需要调用 get_pages，获取新的物理页并换算出物理地址，之后 if 与 else 分支的逻辑便是一致的，即根据 fault 地址换算出 offset，然后调用上一个实验中已经实现过的 map_range_in_pgtbl 函数进行页表映射即可。

2.4

irq_entry.S 中 exception_enter 与 exception_exit 的填写：

`enter` 的逻辑是，先根据 `ARCH_EXEC_CONT_SIZE` 的值递减 `sp` 寄存器，再将 `x0` 到 `x29` 寄存器的内容保存到栈上，再根据已经给出代码对 `sp_el0`、`elr_el1`、`spsr_el1` 内容的转移，保存 `x21` 到 `x23` 和 `x30` 寄存器的内容。一开始并未保存 `x30` 是因为我们使用 `stp` 汇编指令，一次成对保存两个寄存器，所以 `x30` 内容可以留到后面一起保存。

`exit` 的逻辑则和 `enter` 相反，先恢复 `x21` 到 `x23` 和 `x30` 寄存器的内容，再恢复 `x0` 到 `x29` 寄存器的内容，最后根据 `ARCH_EXEC_CONT_SIZE` 的值递增 `sp` 寄存器。

2.5

`sys_thread_exit` 函数中，由于我们当前实验步骤仅仅启用 ID 为 0 的 CPU 运行进程，在线程退出的时候要先使用 `smp_get_cpu_id` 函数获取当前的 cpuid，然后通过 `current_threads` 数组获取当前的线程指针，设置其 state，在调用 `obj_free` 将其释放，最后将 `current_threads` 数组中对应 cpuid 的指针置空即可完成线程释放。

3 运行结果

```
os@os-lab-vm:~/OSlearning/chcore-lab$ make grade
================
Grading lab 3...(may take 50 seconds)
GRADE: Cap create pretest: 10
GRADE: Bad instruction 1: 10
GRADE: Bad instruction 2: 10
GRADE: Fault (1/3): 2
GRADE: Fault (2/3): 3
GRADE: Fault (3/3): 15
GRADE: User Application (1/3): 2
GRADE: User Application (2/3): 3
GRADE: User Application (3/3): 15
GRADE: Put, Get and Exit (1/4): 2
GRADE: Put, Get and Exit (2/4): 3
GRADE: Put, Get and Exit (3/4): 15
GRADE: Put, Get and Exit (4/4): 10
================
Score: 100/100
```

图 1: make grade 结果