操作系统 · Lab2

计01　容逸朗　2020010869

功能简述

- 本次实验修改了 Lab1 中的 `sys_task_info` 和 `sys_get_time`，使得返回值被不同物理页面分割时也能正常返回；
- 实验主要实现了系统调用 `mmap` 和 `munmap`。
  - 首先需要判断 `port` 字段是否有设置 R/W/X 权限，若无则跳过；
  - 然后判断 `addr` 是否对齐页面；
  - 然后计算实际分配的虚拟地址（向上以页面大小取整）
  - 内存映射（拆除）前需要判断是否曾经映射过，若有（无）则报错；
  - 内存映射（mmap）还需要在权限位中加上 U，否则用户程序访问不了对应空间；
  - 然后分别调用 mm 的 `insert_framed_area` 函数和 `unmap_one` 函数增加（拆除）映射即可。

简答题

1. 请列举 SV39 页表页表项的组成，描述其中的标志位有何作用？

   SV39 的 [63:54] 位是保留字段，[53:10] 这 44 位是物理页号，最低的 8 位 [7:0] 则是标志位，标志位的含义如下：

   - V 位表示页表项是否合法；
   - R/W/X 分别代表页表项对应的页面是否能被读/写/执行；
   - U 表示对应页面能否在 U 态下访问；
   - A 表示自从页表项上的这一位被清零之后，页表项的对应虚拟页面是否被访问过；
   - D 表示自从页表项上的这一位被清零之后，页表项的对应虚拟页面是否被修改过。

2. 缺页

   缺页指的是进程访问页面时页面不在页表中或在页表中无效的现象，此时 MMU 将会返回一个中断， 告知 os 进程内存访问出了问题。os 选择填补页表并重新执行异常指令或者杀死进程。

   - 请问哪些异常可能是缺页导致的？

     考虑 `mcause` 的异常编号 (Exception Code) 字段，和缺页相关的异常如下所示：

     编号 12: 指令缺页异常；

     编号 13: load 缺页异常；

     编号 15: store/ AMO 缺页异常；

   - 发生缺页时，描述相关重要寄存器的值，上次实验描述过的可以简略。
     - `scause`: 记录中断 / 异常信息，其中 Interrupt 位记录是中断还是异常，Exception Code 记

录中断 / 异常的类型；
  - `stvec`：记录处理 trap 的程序地址，可根据 MODE 字段分为直接和向量两种模式；
  - `stval`：保存了异常的附加信息，在缺页发生时保存导致缺页异常的虚拟地址；
  - `sepc`：记录了当前指令的下一条指令地址，异常处理完成后由该条指令开始执行；
  - `sstatus`：记录当前系统状态，如特权级、中断状态等；
  - `sscratch`：保存内核栈的位置。

缺页有两个常见的原因，其一是 Lazy 策略，也就是直到内存页面被访问才实际进行页表操作。 比如，一个程序被执行时，进程的代码段理论上需要从磁盘加载到内存。但是 os 并不会马上这样做， 而是会保存 .text 段在磁盘的位置信息，在这些代码第一次被执行时才完成从磁盘的加载操作。

- 这样做有哪些好处?

  由于使用页面时才加载页面，这使得 Lazy 策略效率不会低于直接加载页面的方式。在页面被加载后没有被使用就被替换的情况下，Lazy 策略可以避免无用的加载，节省时间。

其实，我们的 mmap 也可以采取 Lazy 策略，比如：一个用户进程先后申请了 10G 的内存空间， 然后用了其中 1M 就直接退出了。按照现在的做法，我们显然亏大了，进行了很多没有意义的页表操作。

- 处理 10G 连续的内存页面，对应的 SV39 页表大致占用多少内存 (估算数量级即可)?

  SV39 页表相当于使用 8B（64 位地址）存放一页（4KB），故 10G 连续的内存页面需要的内存空间为：$10\text{GB} \div 4\text{KB} \times 8\text{B} = 20\text{MB}$；

- 请简单思考如何才能实现 Lazy 策略，缺页时又如何处理? 描述合理即可，不需要考虑实现。

  首先，在分配内存时仅作记录，不进行实际的内存分配。当出现缺页时，直接在异常处理模块中分配内存即可。

缺页的另一个常见原因是 swap 策略，也就是内存页面可能被换到磁盘上了，导致对应页面失效。

- 此时页面失效如何表现在页表项(PTE)上?

  页表项的 V 位会被置为 0。

3. 双页表与单页表

为了防范侧信道攻击，我们的 os 使用了双页表。但是传统的设计一直是单页表的，也就是说， 用户线程和对应的内核线程共用同一张页表，只不过内核对应的地址只允许在内核态访问。(备注：这里的单/双的说法仅为自创的通俗说法，并无这个名词概念，详情见 KPTI)

- 在单页表情况下，如何更换页表?

  更改 `satp` 。

- 单页表情况下，如何控制用户态无法访问内核页面? （tips:看看上一题最后一问）

  将页表项的 U 字段置为 0 即可。

- 单页表有何优势? （回答合理即可）

  在切换系统态时无需更换页表，因此不需跳板页，实现简单。

- 双页表实现下，何时需要更换页表？假设你写一个单页表操作系统，你会选择何时更换页表（回答合理即可）？

  双页表系统在切换系统态或用户程序切换时便需要更换页表；

  对于单页表操作系统，我会选择在用户切换线程时才需要更换页表。

Honor Code

1. 在完成本次实验的过程（含此前学习的过程）中，我曾分别与 **以下各位** 就（与本次实验相关的）以下方面做过交流，还在代码中对应的位置以注释形式记录了具体的交流对象及内容：

   无

2. 此外，我也参考了 **以下资料** ，还在代码中对应的位置以注释形式记录了具体的参考来源及内容：

   rCore-Tutorial-Guide 2023 春季学期 的第四部分。

3. 我独立完成了本次实验除以上方面之外的所有工作，包括代码与文档。 我清楚地知道，从以上方面获得的信息在一定程度上降低了实验难度，可能会影响起评分。
4. 我从未使用过他人的代码，不管是原封不动地复制，还是经过了某些等价转换。我未曾也不会向他人（含此后各届同学）复制或公开我的实验代码，我有义务妥善保管好它们。 我提交至本实验的评测系统的代码，均无意于破坏或妨碍任何计算机系统的正常运转。 我清楚地知道，以上情况均为本课程纪律所禁止，若违反，对应的实验成绩将按 “-100” 分计。