Lab 4

思考题

Thinking 4.1

```
.macro SAVE_ALL
.set noreorder
.set noat
    move    k0, sp
.set reorder
    bltz    sp, 1f
    li      sp, KSTACKTOP
.set noreorder
1:
    subu    sp, sp, TF_SIZE
    sw      k0, TF_REG29(sp)
    mfc0    k0, CP0_STATUS
    sw      k0, TF_STATUS(sp)
    mfc0    k0, CP0_CAUSE
    sw      k0, TF_CAUSE(sp)
    mfc0    k0, CP0_EPC
    sw      k0, TF_EPC(sp)
    mfc0    k0, CP0_BADVADDR
    sw      k0, TF_BADVADDR(sp)
    mfhi    k0
    sw      k0, TF_HI(sp)
    mflo    k0
    sw      k0, TF_LO(sp)
    sw      $0, TF_REG0(sp)
    sw      $1, TF_REG1(sp)
    sw      $2, TF_REG2(sp)
    sw      $3, TF_REG3(sp)
    sw      $4, TF_REG4(sp)
    sw      $5, TF_REG5(sp)
    sw      $6, TF_REG6(sp)
    sw      $7, TF_REG7(sp)
    sw      $8, TF_REG8(sp)
    sw      $9, TF_REG9(sp)
    sw      $10, TF_REG10(sp)
    sw      $11, TF_REG11(sp)
    sw      $12, TF_REG12(sp)
    sw      $13, TF_REG13(sp)
    sw      $14, TF_REG14(sp)
```

```
    sw      $15, TF_REG15(sp)
    sw      $16, TF_REG16(sp)
    sw      $17, TF_REG17(sp)
    sw      $18, TF_REG18(sp)
    sw      $19, TF_REG19(sp)
    sw      $20, TF_REG20(sp)
    sw      $21, TF_REG21(sp)
    sw      $22, TF_REG22(sp)
    sw      $23, TF_REG23(sp)
    sw      $24, TF_REG24(sp)
    sw      $25, TF_REG25(sp)
    sw      $26, TF_REG26(sp)
    sw      $27, TF_REG27(sp)
    sw      $28, TF_REG28(sp)
    sw      $30, TF_REG30(sp)
    sw      $31, TF_REG31(sp)
.set at
.set reorder
.endm
/*
 * Note that we restore the IE flags from stack. This means
 * that a modified IE mask will be nullified.
 */
.macro RESTORE_SOME
.set noreorder
.set noat
    lw      v0, TF_STATUS(sp)
    mtc0    v0, CP0_STATUS
    lw      v1, TF_LO(sp)
    mtlo    v1
    lw      v0, TF_HI(sp)
    lw      v1, TF_EPC(sp)
    mthi    v0
    mtc0    v1, CP0_EPC
    lw      $31, TF_REG31(sp)
    lw      $30, TF_REG30(sp)
    lw      $28, TF_REG28(sp)
    lw      $25, TF_REG25(sp)
    lw      $24, TF_REG24(sp)
    lw      $23, TF_REG23(sp)
    lw      $22, TF_REG22(sp)
    lw      $21, TF_REG21(sp)
    lw      $20, TF_REG20(sp)
    lw      $19, TF_REG19(sp)
    lw      $18, TF_REG18(sp)
    lw      $17, TF_REG17(sp)
    lw      $16, TF_REG16(sp)
```

```
    lw      $15, TF_REG15(sp)
    lw      $14, TF_REG14(sp)
    lw      $13, TF_REG13(sp)
    lw      $12, TF_REG12(sp)
    lw      $11, TF_REG11(sp)
    lw      $10, TF_REG10(sp)
    lw      $9, TF_REG9(sp)
    lw      $8, TF_REG8(sp)
    lw      $7, TF_REG7(sp)
    lw      $6, TF_REG6(sp)
    lw      $5, TF_REG5(sp)
    lw      $4, TF_REG4(sp)
    lw      $3, TF_REG3(sp)
    lw      $2, TF_REG2(sp)
    lw      $1, TF_REG1(sp)
.set at
.set reorder
.endm
```

SAVE_ALL 宏定义于 include/stackframe.h 中，保存现场时使用了 `$k0` 寄存器， `$k0` 为内核用寄存器，它的修改不会影响现场。

可以。它们被正常保存在 Tf 中。

do_syscall 读取了这下参数并传入 sys_* 。

cp0_epc 增加了 4，并将系统调用返回值存入了 `$v0` 寄存器，这样用户态可以得到正确的返回值并能从系统调用后的下一条指令继续执行。

Thinking 4.2

该 `envid` 可能已被废弃，通过 `envs[ENVX(envid)]` 并不是该进程的 `env` 。可能会错误地读到本不存在进程的 `env` 。

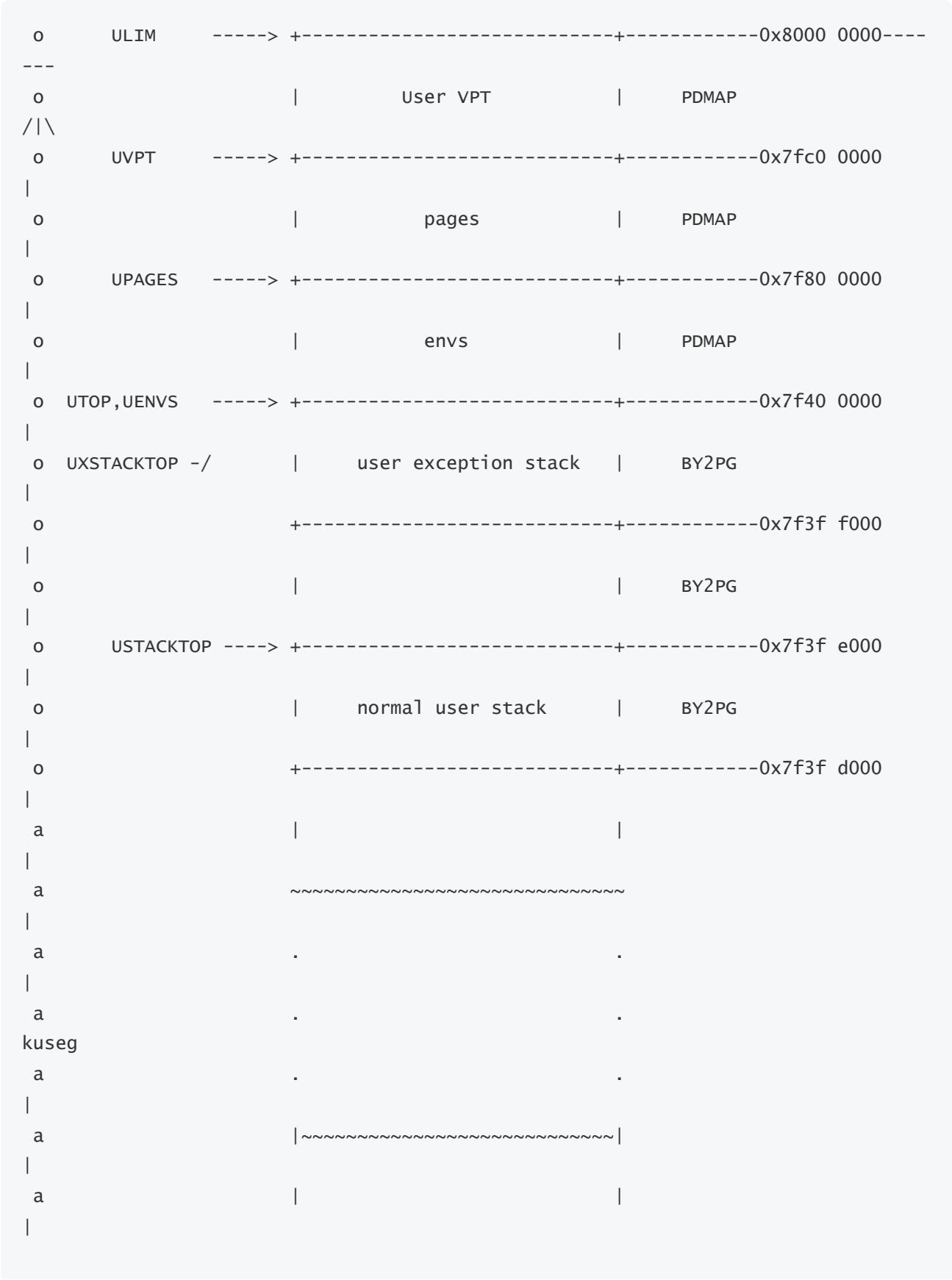
Thinking 4.3

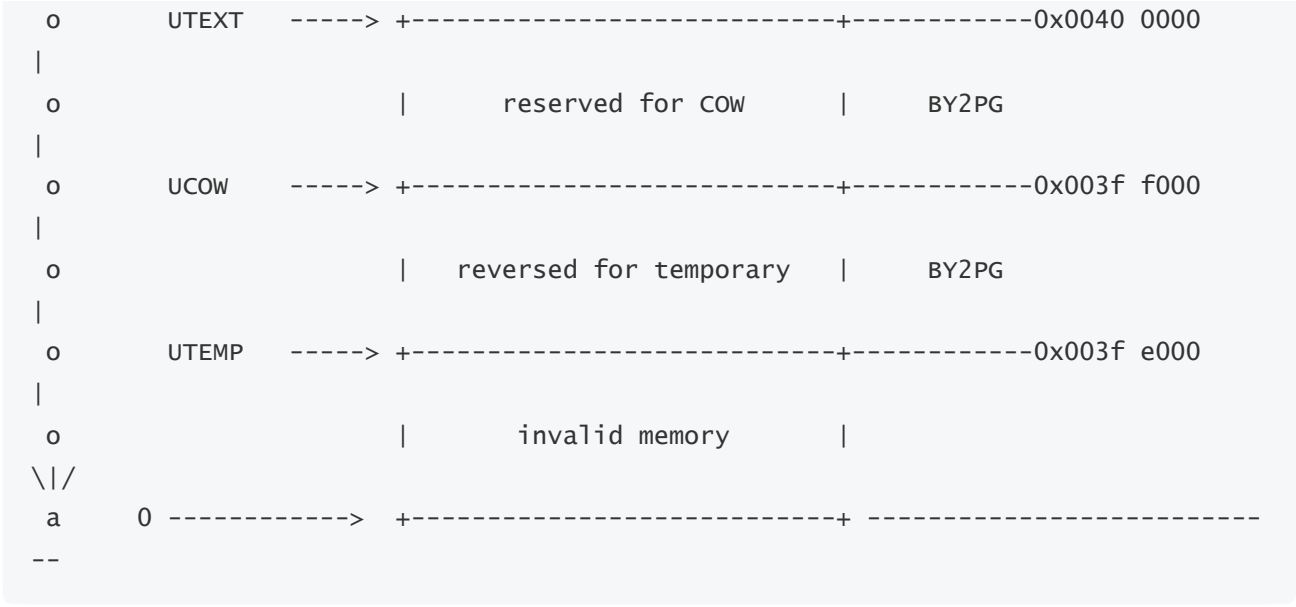
在 MOS 中，传入的 `envid` 为 0 时代表取当前进程，故为避免冲突不会产生为 0 的 `envid`

Thinking 4.4

C.

Thinking 4.5





kuseg 中， UTEMP 以下为 invalid memory；UTEMP 到 UTEXT 是为特殊用途保留的部分；UTOP以上的用户空间对于用户进程来说不可变；USTACKTOP+BY2PG 到 UTOP 为用户异常栈，不应被映射；USTACKTOP 到 USTACKTOP+BY2PG 未使用，不需要映射。

故需要映射的为 UTEXT 到 USTACKTOP 之间的使用了的页面。

Thinking 4.6

```
#define vpt ((volatile Pte *)UVPT)
#define vpd ((volatile Pde *)(UVPT + (PDX(UVPT) << PGSHIFT)))
```

vpt 和 vpd 分别是页表与页目录的虚拟地址，可直接作为指针访问以获取自身页表。

在 `env_alloc` 中利用 `env_setup_vm` 初始化用户地址空间时，我们将它自身的页表项映射到了 `UVPT` ，故可以如此访问。

将自身页表与页目录映射在自身的地址空间，体现自映射。

不能，映射时权限位仅有 `PTE_V` 。

Thinking 4.7

发生缺页异常时，会由 `do_tlb_mod` 将 epc 设置为 `env_user_tlb_mod_entry` ，使得返回时重写进入用户态的异常处理函数，将异常分配给用户处理函数进行处理，此即为异常重入。

缺页异常在用户态处理，故需要将 Trapframe 复制到用户空间以供读取。

Thinking 4.8

- 防止处理缺页异常时再次出现异常导致系统崩溃。
- 便于用户自定义异常处理方法。

Thinking 4.9

在 `syscall_exofork` 中发生的异常需要进行处理。

此时写 `env_user_tlb_mod_entry` 会发生缺页异常，触发写时复制机制，但未设置异常处理，导致系统崩溃。

难点分析

因为 lab3 schedule 的问题，ipc卡了许久才完成。

此外较为困难的是 fork 的流程，涉及多个系统调用与用户态的处理。

心得体会

第一次课上extra本想采取优雅一些的解法，奈何课程组测试点设置父进程并没有按照正常的方式，导致几种建树的方式均没能成功，最后只得暴力，秒过。第二次课上extra忘记考虑sem_id为负数，C语言基础不牢。