清华大学本科生考试试题专页纸	
考试课程：操作系统 A（卷）	时间：2022年04月11日上午 09:50~12:15

系别：_____	班级：_____	学号：_____	姓名：_____

答卷注意事项	
	1. 答题前，请先在试题纸和答卷本上写明A卷或B卷、系别、班级、学号和姓名。
	2. 要求在答卷本上答题，要写明题号，不必抄题。
	3. 答题时，要书写清楚和整洁，让判卷人清楚回答内容。
	4.请回答所有题目，本试卷有1个判断题，8个简答题，共6页。
	5. 考试完毕，必须将试题纸和答卷本一起交回。

一、(12分)判断题

请写出“√”和“×”来判断下列题目的对错。

1. [ ] 操作系统是管理硬件资源、控制程序运行、改善人机界面和为应用软件提供支持的一种系统软件。
2. [ ] Linux操作系统是宏（单体）内核操作系统。
3. [ ] 编译器生成的执行程序中的地址既可以是操作系统中的虚拟地址也可以是物理地址。
4. [ ] 操作系统需要自己来建立和设置好它正常运行所需要的栈空间。
5. [ ] 本课程实验中，QEMU模拟的计算机加电启动后，CPU将直接跳转到rCore或uCore的入口地址执行。
6. [ ] 内存虚拟地址的页内偏移位数与物理地址的页内偏移位数是不相等的。
7. [ ] CLOCK算法属于LRU类算法，不会有Belady异常现象。
8. [ ] 在RISC-V CPU的Sv39模式下，可以指定逻辑页面和物理页面(帧)的大小为1GB。
9. [ ] 用户态应用程序的栈空间是由操作系统来分配的。
10. [ ] 本课程实验的RISC-V CPU具有M-mode、S-mode和U-mode，但实验编码仅涉及S-mode和U-mode。
11. [ ] 在RISC-V CPU的用户态(U-Mode)执行的代码无法执行特权级指令，从而避免了操作系统的代码和数据被用户态代码直接访问到。
12. [ ] 在RISC-V CPU的用户态(U-Mode)执行的代码可以通过设置相关CSR寄存器来关闭（disable）中断。

二、(88分)简答题

1. (15分)

1. 请指出下面的操作系统抽象可对应到哪个具体对象？

抽象：	进程　文件　地址空间　异常控制流　软件的执行环境
具体对象：	处理器　内存+外存　磁盘　支撑软件的软硬件栈　外设中断

2. 下一段可在Linux下执行的 C 程序，其中包含一个除以零的操作。请描述该程序在Linux上执行的输出结果，并简要说明原因。注：运行Linux的硬件平台可选择为RISC-V 64 或者 X86-64，请指明你选择的一种硬件平台。

```c
#include <stdio.h>

int main() {
   printf("1 / 0 = %d", 1 / 0);
   return 0;
}
```

3. 下一段可在Linux下执行的C 程序，睡眠5秒后打印出一个字符串，并把字符串内容存入一个文件中。请简要列出这个程序向Linux发出了哪些系统调用。

```c
#include <stdio.h>
#include <unistd.h>

int main() {
  sleep(5);

  const char* hello_string = "Hello Linux!\n";

  printf(hello_string);

  FILE *output_file = fopen("output.txt", "w");
  fwrite(hello_string, sizeof(hello_string),1,output_file);
  fclose(output_file);

  return 0;
}
```

2. (14分)

在实验环境搭建过程中，我们需要安装QEMU软件来模拟一台计算机，并会执行如下的命令行脚本，来让QEMU模拟加电启动虚拟计算机，运行操作系统和应用：

```
qemu-system-riscv64 -machine virt -nographic -bios ../bootloader/rustsbi-qemu.bin \
  -device loader,file=target/riscv64gc-unknown-none-elf/release/os.bin,addr=0x80200000
```

或

```
qemu-system-riscv64 -nographic -machine virt -bios ./bootloader/rustsbi-qemu.bin \
  -kernel build/kernel
```

并可得到类似如下内容：

```
.______       __    __      _______.___________.  _______..______   __
|   _  \     |  |  |  |     /       |           | /       ||   _  \ |  |
|  |_)  |    |  |  |  |    |   (----`---|  |----`|   (----`|  |_)  ||  |
|      /     |  |  |  |     \   \       |  |      \   \    |   _  < |  |
|  |\  \----.|  `--'  |.----)   |      |  |  .----)   |   |  |_)  ||  |
| _| `._____| \______/ |_______/       |__|  |_______/    |______/ |__|

[rustsbi] Implementation: RustSBI-QEMU Version 0.0.2
[rustsbi-dtb] Hart count: cluster0 with 1 cores
[rustsbi] misa: RV64ACDFIMSU
[rustsbi] mideleg: ssoft, stimer, sext (0x222)
[rustsbi] medeleg: ima, ia, bkpt, la, sa, uecall, ipage, lpage, spage (0xb1ab)
[rustsbi] pmp0: 0x10000000 ..= 0x10001fff (rwx)
[rustsbi] pmp1: 0x80000000 ..= 0x8fffffff (rwx)
[rustsbi] pmp2: 0x0 ..= 0xffffffffffffff (---)
qemu-system-riscv64: clint: invalid write: 00000004
[rustsbi] enter supervisor 0x80200000
Hello, world!
.text [0x80200000, 0x80202000)
```

请回答如下问题：（注：每小问回答的字数在20字以内）

1. 请问QEMU模拟的CPU是？模拟的CPU是32位还是64位？
2. QEMU在模拟加电启动计算机前，把哪些内容加载到它模拟的物理内存中了？
3. QEMU在执行操作系统的代码之前，做了哪些事情？
4. QEMU从加电启动到执行操作系统第一条指令的时间段中，是否出现了特权级切换的情况？
5. 操作系统要执行的第一条指令的物理地址是？
6. 操作系统在正常执行高级语言编写的函数之前，要完成哪些基本的初始化工作？

3. (8分)

阿图在学习了OS相关知识后，认为在更改页表基址或修改页表项之后一定要及时刷新 TLB，所以在完成第四章“地址空间”的实验（lab 2）中，在 mmap 和 munmap 函数中加入了大量的刷新 TLB 的指令。请回答如下实验相关问题：（注：每小问回答的字数在40字以内）

1. 刷新 TLB 的指令是否是特权指令？
2. 阿图插入的语句有必要吗？为什么？
3. 更换页表之后需要刷新数据cache 和指令cache吗？请说明原因。
4. 在内核页表映射和用户页表映射中都有 trampoline 这一页。请问这一页中放置了什么内容？这一页的作用是什么？

4. (10分)

下面是第三章“多道程序与分时多任务”进行任务上下文切换的uCore/rCore代码：

uCore（）

```
.globl swtch
swtch:
sd ra, 0(a0)
sd sp, 8(a0)
sd s0, 16(a0)
# ... save s1-s10
sd s11, 104(a0)

ld ra, 0(a1)
ld sp, 8(a1)
ld s0, 16(a1)
# ... restore s1-s10
ld s11, 104(a1)
ret
```

程序片段一：uCore中的swtch汇编函数

rCore:

```
.altmacro
.macro SAVE_SN n
sd s\n, (\n+2)*8(a0)
.endm
.macro LOAD_SN n
ld s\n, (\n+2)*8(a1)
.endm
.section .text
.globl __switch
__switch:
sd sp, 8(a0)
sd ra, 0(a0)
.set n, 0
.rept 12
SAVE_SN %n
.set n, n + 1
.endr
ld ra, 0(a1)
.set n, 0
.rept 12
LOAD_SN %n
.set n, n + 1
.endr
ld sp, 8(a1)
ret
```

程序片段二：rCore中的__switch汇编函数

（注：每小问回答的字数在40字以内）

1. 任务上下文切换与Trap上下文切换的区别是啥？
2. 任务上下文切换与函数上下文切换的区别是啥？
3. 选择下面中一个小问回答：
    - 请描述 uCore的swtch 函数的输入参数
    - 请描述 rCore的__switch函数的输入参数
4. RISC-V通用寄存器有32个，为何swtch函数或__switch函数仅处理了其中的一部分？
5. 选择下面中一个小问回答：
    - uCore的swtch 函数要完成执行流和内核栈的切换，请问这两者分别发生在程序片段一的哪一行？
    - rCore的__switch函数要完成执行流和内核栈的切换，请问这两者分别发生在程序片段二的哪一行？

注：执行流切换指： PC寄存器指向新运行任务（也称进程）的代码逻辑；内核栈切换指：从此开始的所有栈操作将在新栈上发生。

5. (10分)

在第四章“地址空间”的实验（lab 2）中，操作系统有了对特权级的支持和页机制的支持。stvec CSR寄存器保存了在产生中断、异常或系统调用时CPU要跳转并执行的地址。如果应用程序运行在用户态时产生了异常，系统调用或中断，导致RISC-V 64 CPU从用户态陷入到了内核态。请问：

(注：每小问回答的字数在40字以内)

1. 如果有特权级机制，用户态的应用程序执行特权指令后会产生异常，CPU硬件做哪会些处理，并跳转到哪里执行？
2. 如果有特权级但没有页机制，用户态的应用程序是否可以直接读写内核态中操作系统的代码或数据？请解释原因。
3. 在RVSC-V中，用户态应用程序通过什么指令实现系统调用并陷入到内核态？
4. 在RVSC-V中，内核通过什么指令实现返回到用户态进程？
5. 如果用户态应用程序通过系统调用让CPU跳转到这个地址并开始准备执行第一条指令时，产生了一个时钟中断。请问CPU接下来要做什么事情？

6. (6分)

一个计算机使用32位的虚拟地址，采用4KB大小的页面。应用程序的代码和数据都位于从虚拟地址0开始的连续4096个页面，栈位于最高虚拟地址的一个页面，不使用堆。为了能够让应用程序不产生缺页异常并正常执行，如果使用一级分页，一个虚地址中，页表索引位占20位，页内偏移位占12位，页表中至少需要多少个有效表项？如果使用两级分页，一个虚地址中，页目录索引位占10位，页表索引位占10位，页内偏移位占12位，页表至少需要多少个有效页表项？**请给出计算过程的描述。**

7. (12分)

在13个时间单位的时间段内，一个程序的虚拟页面访问顺序如下：

1	2	3	4	5	6	7	8	9	10	11	12	13
e	d	a	c	c	d	b	c	e	c	e	a	d

操作系统给这个程序分配了5个物理页帧，初始物理页帧为空。请回答如下问题：

1. 如采用工作集页面置换算法，工作集窗口 Δ 的大小 τ = 4 ，请问在哪些时刻会产生缺页异常，缺页异常次数总共是多少？

2. 如采用缺页率置换算法，容忍的缺页窗口 T = 2 ，请问在哪些时刻会产生缺页异常，缺页异常次数总共是多少？

8. (13分)

假设在一个程序执行的工作负载中，虚拟页码访问流中有一些长的页码访问序列的重复，序列之后有时会是一个随机的页码。例如，在序列 0，1，2...，511，431，0，1，2...，511，332，0，1，2...511... 中，就包含了0，1，...，511的重复序列，以及跟随在它们之后的随机页码431和332。请回答下列问题：(注：每小问回答的字数在40字以内)

1. 请描述页面置换算法 LRU，FIFO，Clock 的基本思路。
2. 物理页帧数具有哪种特征（即在哪个范围内），页面置换算法 LRU，FIFO，Clock 在处理换页时效果都一样差？请说明原因。
3. 如果给这个程序只分配 500 个物理页帧，请描述一个效果优于 LRU、FIFO 或 Clock 算法的针对这个程序的特定页面置换方法。