|  |  |
| --- | --- |
| Lab 3: xv6: System Calls | |
| Student ID: | Student Name: |

1. **RISC-V privileged mode (1pt)**
2. RISC-V systems support three-level privilege mode. Write their name.
3. Explain the privilege instruction ecall, sret, and mret.
4. Before the first process init runs, the xv6 booting relates to

* kernel/entry.S
* kernel/start.c
* kernel/main.c

Answer in which privileged mode each program runs.

1. To implement the time-sharing, the OS sets the timer at the boot time. During the three phases (entry.S, start.c, main.c) in the problem 1.C, when does the OS set the timer?
2. **xv6 system call internal (1pt)**
3. xv6 separates trap into three types according to their source. Write their name.

**Please delete the text below when submitting**:

The xv6-risc book uses the term trap to refer the term exception, which we have discussed in our class. In fact, the terms: trap, exception are used differently in different textbooks. CSAPP separates the terms: interrupt, trap, fault, and uses the term exception to group them. While OSTEP and xv6-riscv book uses the term exception to denote the case fault in CSAPP, and the term trap to denote the case exception in CSAPP. The lack of uniformity of terminology means that the fine-grained distinction is not so important. The important thing is to understand that programs can have exception flow, which occur for various reasons (system calls, something illegal, interrupt), and how to handle it by the OS and hardware.

1. The OS assigns an integer to each system call to distinguish them. Which the xv6 code contains information about this? What is the system call number given to fork, wait system calls, respectively?
2. RISC-V CPU has a set of control registers that the kernel writes to tell the CPU how to handle traps, and that the kernel can read to find out about a trap that has occurred. Explain the role of following registers:

* stvec
* sepc
* secause
* sstatus

1. The xv6 boot is completed by running the first process init. To this end, the OS calls the userinit function in kernel/proc.c. The first process executes a small program written in RISC-V assembly, which makes the first system call in xv6. user/initcode.S loads the number for the exec system call, SYS\_EXEC, into register a7 and then calls ecall. Thus, the trap starts. Which xv6 codes should run sequentially, before running kernel/exec.c.

1. Debugging with GDB (0.2pt)
2. The Unix ps command (0.8pt)