|  |  |
| --- | --- |
| Lab 3: xv6: System Calls | |
| Student ID: 21102376 | Student Name: 이연우 |

1. **RISC-V privileged mode (1pt)**
2. RISC-V systems support three-level privilege mode. Write their name.

* **Machine mode**
* **Supervisor mode**
* **User mode**

1. Explain the privilege instruction ecall, sret, and mret.

* **Ecall : This instruction is used to make a system call from user mode to supervisor mode. When a program running in user mode executes an ecall instruction, the privilege level changes to supervisor mode, and the control is transferred to the supervisor’s exception handler. The system call number and arguments are passed via registers.**
* **Sret : This instruction is used to return from a supervisor-level exception. When the sret instruction is executed, the privilege level is changed to user mode, and the program resumes execution at the instruction following the ecall instruction.**
* **Mret : This instruction is similar to sret but is used to return from machine-level exceptions.**

1. Before the first process init runs, the xv6 booting relates to

* kernel/entry.S **: This program runs in machine mode, which is the highest privilege level. It sets up the initial stack and jumps to the start kernel function in supervisor mode.**
* kernel/start.c **: This program runs in supervisor mode and it sets up the page tables, enables virtual memory, and jumps to the main function in user mode.**
* kernel/main.c **: This program runs in user mode, and it initializes the system’s processes, sets up the file system, and starts the first process.**

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?

* **The timer is typically set by the OS during the kernel initialization process, which includes the execution of kernel/entry.S and kernel/start.c. Specifically, the timer is usually set in the start kernel function, which is part of kernel/start.c, before jumping to the main function is user mode. This ensures that the timer is set up before any user-level processes are started and that the time-sharing mechanism is active from the beginning of the system’s operation.**

1. **xv6 system call internal (1pt)**
2. xv6 separates trap into three types according to their source. Write their name.

* **Processor exceptions: traps caused by exceptional conditions detected by the processor, such as page faults and illegal instructions.**
* **System calls: traps initiated by user-level programs to request a service from the operating system.**
* **Interrupts: asynchronous traps caused by external devices, such as the timer or network card.**

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?

* **The system call numbers are defined in the file syscall.h. The system call number for fork is SYS\_fork, which has a value of 1, and the system call number for wait is SYS\_wait, which has a value of 2.**

1. 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: **This is the trap vector base address register, which holds the address of the trap handler code. When a trap occurs, the processor jumps to this address to start executing the trap handler.**
* sepc: **This is the trap program counter register, which holds the address of the instruction that caused the trap. This register is used to restart the instruction after the trap handler has completed.**
* secause: **This is the trap cause register, which holds the cause of the trap, such as an exception code, system call number, or interrupt number.**
* sstatus: **This is the supervisor status register, which holds various processor status flags, such as interrupt enable and user/supervisor mode.**

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?

* **The following xv6 codes should run sequentially:** 
  + **kernel/console.c: initializes the console device**
  + **kernel/kalloc.c: initializes the kernel memory allocator**
  + **kernel/proc.c: initializes the process table and creates the first kernel process (init) in the userinit function**
  + **kernel/vm.c: initializes the virtual memory system, including the page table for the kernel’s address space**

1. **The Unix ps command (1pt)**

**Text, letter

Description automatically generated**