System Calls

COSC-361
Stephen Marz

TICKLE COLLEGE OF ENGINEERING
MIN H. KAO DEPARTMENT OF ELECTRICAL ENGINEERING & COMPUTER SCIENCE

1

What is a System Call?

- A way for unprivileged space (user space) to access kernel services.
- Examples:
  - exit
  - open
  - close
  - read
  - write

THE UNIVERSITY OF TENNESSEE KNOXVILLE

2

How System Calls Work

- Typically an assembly instruction causes the CPU to immediately jump to a certain kernel function.
- RISC-V: **ecall** (environment call)
- ARM: **svc** (supervisor call)
- x86_64: **syscall** (system call)

3

Trapping

- The ecall instruction causes the CPU to
  - Finish currently executing instruction
  - Halt execution
  - Switch to higher privilege level
  - Jump to trap vector
    - This is the memory address where the handler's code can be found.
  - It is responsibility of the OS to resume normal operations (in RISC-V, this is done by mret instruction).

4

System Calls in Linux

```
#define __NR_read 0
#define __NR_write 1
#define __NR_open 2
#define __NR_close 3
#define __NR_stat 4
#define __NR_fstat 5
#define __NR_lstat 6
#define __NR_poll 7
#define __NR_lseek 8
#define __NR_mmap 9
#define __NR_mprotect 10
#define __NR_munmap 11
#define __NR_brk 12
#define __NR_rt_sigaction 13
#define __NR_rt_sigprocmask 14
#define __NR_rt_sigreturn 15
#define __NR_ioctl 16
#define __NR_pread64 17
#define __NR_pwrite64 18
#define __NR_readv 19
#define __NR_writev 20
#define __NR_access 21
#define __NR_pipe 22
```


5

CPU Vectoring System

3.1.12 Machine Trap-Vector Base-Address Register (mtvec)

The mtvec register is an XLEN-bit read/write register that holds trap vector configuration, consisting of a vector base address (BASE) and a vector mode (MODE).

Figure 3.8: Machine trap-vector base-address register (mtvec).

The mtvec register must always be implemented, but can contain a hardwired read-only value. If mtvec is writable, the set of values the register may hold can vary by implementation. The value in the BASE field must always be aligned on a 4-byte boundary, and the MODE setting may impose additional alignment constraints on the value in the BASE field.

We allow for considerable flexibility in implementation of the trap vector base address. On the one hand, we do not wish to burden low-end implementations with a large number of state bits, but on the other hand, we wish to allow flexibility for larger systems.

Value	Name	Description
0	Direct	All exceptions set pc to BASE.
1	Vectored	Asynchronous interrupts set pc to BASE+4×cause.
≥2	—	*Reserved*

6

Trap Cause

Interrupt	Exception Code	Description
1	0	User software interrupt
1	1	Supervisor software interrupt
1	2	*Reserved*
1	3	Machine software interrupt
1	4	User timer interrupt
1	5	Supervisor timer interrupt
1	6	*Reserved*
1	7	Machine timer interrupt
1	8	User external interrupt
1	9	Supervisor external interrupt
1	10	*Reserved*
1	11	Machine external interrupt
1	≥12	*Reserved*
0	0	Instruction address misaligned
0	1	Instruction access fault
0	2	Illegal instruction
0	3	Breakpoint
0	4	Load address misaligned
0	5	Load access fault
0	6	Store/AMO address misaligned
0	7	Store/AMO access fault
0	8	Environment call from U-mode
0	9	Environment call from S-mode
0	10	*Reserved*
0	11	Environment call from M-mode
0	12	Instruction page fault
0	13	Load page fault
0	14	*Reserved*
0	15	Store/AMO page fault
0	≥16	*Reserved*

Table 3.6: Machine cause register (`mcause`) values after trap.

7

Linux System Call ABI

RISC-V

- Syscall number goes into register a0
- Parameters
  - a1
  - a2
  - a3
  - a4
  - a5
  - a6
- Only 6 parameters are possible

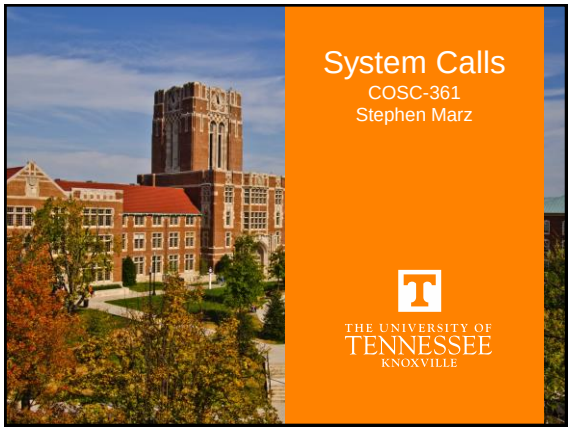
8

Linux System Call ABI

X86-64

- Syscall number goes into register rcx
- Parameters
  - rdi
  - rsi
  - rdx
  - r10
  - r8
  - r9
- Only 6 parameters are possible

9


System Calls
COSC-361
Stephen Marz
THE UNIVERSITY OF
TENNESSEE
KNOXVILLE

10