In ARM architecture, system calls (syscall) and ioctl operations are executed in **Supervisor mode (SVC mode)**.

1. **Syscall (svc instruction in ARM)**:
   * When a user-space application makes a system call, it triggers a software interrupt using the SVC (Supervisor Call) instruction.
   * This switches the processor from **User mode** to **Supervisor mode (SVC mode)**. The kernel handles the system call in **SVC mode** and then switches back to User mode after execution.
2. **ioctl (Input/Output Control)**:
   * ioctl is a system call used for device-specific operations. Since ioctl is a system call, it follows the same mechanism as syscall and executes in **SVC mode** within the kernel.
   * The user-space process makes an ioctl call, which transitions to the kernel via SVC, where it is handled in Supervisor mode.

**Summary:**

* **User-space code runs in User mode**.
* **System calls (syscall, ioctl, etc.) execute in Supervisor (SVC) mode**.
* **The kernel operates in privileged modes** (such as SVC, IRQ, FIQ, etc.).

**1. ARM Processor Modes**

ARM architecture (AArch32) has several execution modes:

|  |  |  |
| --- | --- | --- |
| **Mode Name** | **Description** | **Privilege Level** |
| **User (USR)** | Runs user applications, restricted access | Unprivileged |
| **Supervisor (SVC)** | Kernel mode for handling syscalls | Privileged |
| **IRQ (Interrupt Request)** | Handles normal interrupts | Privileged |
| **FIQ (Fast Interrupt Request)** | Handles high-priority interrupts | Privileged |
| **Abort (ABT)** | Handles memory faults | Privileged |
| **Undefined (UND)** | Handles undefined instructions | Privileged |
| **System (SYS)** | Privileged mode (used for OS tasks) | Privileged |

* **User mode (USR)**: Runs normal applications, cannot access hardware or privileged instructions.
* **Supervisor mode (SVC)**: Kernel mode, has full access to system resources.
* **System mode (SYS)**: Similar to SVC but does not automatically switch back to User mode.

**2. How System Calls (syscall) Work in ARM**

A system call is a request from a user-space program to access privileged operations (e.g., file I/O, process management). In ARM:

The syscall number and arguments are usually passed through registers (e.g., r0, r1, r2).

Example:

mov r7, #\_\_NR\_write @ Syscall number for 'write'

mov r0, #1 @ File descriptor (stdout)

ldr r1, =msg @ Message to write

mov r2, #len @ Message length

svc #0 @ Trigger system call

The processor switches to **SVC mode**.

The kernel checks the syscall number (r7 in ARM EABI), fetches arguments, and executes the requested service. After completion, the result is placed in r0.The kernel executes a return sequence (via movs pc, lr or eret in ARMv8).

ioctl (Input/Output Control) is a system call used for device-specific operations. Since ioctl is a **system call**, it follows the same **User mode → SVC mode → User mode** transition.

In ARMv8 (AArch64), execution is managed by **Exception Levels (EL)** instead of modes:

|  |  |
| --- | --- |
| **Exception Level** | **Description** |
| **EL0** | User-space applications (unprivileged) |
| **EL1** | Kernel/OS (privileged, like SVC mode in ARMv7) |
| **EL2** | Hypervisor (for virtualization) |
| **EL3** | Secure Monitor (for TrustZone) |

* Instead of SVC, ARMv8 uses the SVC or HVC instruction.
* System calls transition from **EL0 → EL1** instead of **User mode → SVC mode**.

**5. Summary**

* **User-space programs execute in User mode (EL0 in ARMv8).**
* **System calls (syscall, ioctl, etc.) are handled in SVC mode (EL1 in ARMv8).**
* **The SVC instruction (or HVC for hypercalls) triggers the kernel.**
* **Kernel processes the request and returns control to User mode.**