

**explain the previous program**

This ARM/Linux assembly prints “Hello World!” to standard output using the write system call, then terminates using the exit system call, with syscall numbers placed in r7 and arguments in r0–r2 as per the 32-bit ARM EABI convention for Linux syscalls.[[1]](#fn1)[[2]](#fn2)

**Data section**

* The string is defined as an ASCII literal labeled message, and length is computed as the assembler difference . - message, giving the byte count including the newline character .[[3]](#fn3)[[1]](#fn1)

**Syscall register convention**

* On 32-bit ARM Linux, r7 holds the syscall number and r0–r6 hold up to six arguments; write is number 4 and exit is number 1 in the classic ARM Linux syscall table.[[2]](#fn2)[[1]](#fn1)

**write call setup**

* r0 is set to 1 for the file descriptor stdout, r1 is loaded with the address of message, and r2 is set to the number of bytes to write; r7 is set to 4 to select sys\_write, and swi 0 triggers the supervisor call to enter the kernel and perform the write.[[4]](#fn4)[[3]](#fn3)[[1]](#fn1)

**exit call**

* To terminate, r7 is changed to 1 for sys\_exit and swi 0 is executed again; optionally r0 would carry the exit status code, with zero commonly used when indicating success, though the snippet omits setting r0 here.[[5]](#fn5)[[1]](#fn1)[[4]](#fn4)

**Notes on specifics**

* The immediate value to swi is ignored on ARM Linux, so swi 0 is conventional; on newer AArch64 the mechanism uses svc with different syscall numbers and registers, but this program targets 32-bit ARM.[[6]](#fn6)[[4]](#fn4)

⁂

1. <https://stackoverflow.com/questions/12946958/what-is-the-interface-for-arm-system-calls-and-where-is-it-defined-in-the-linux>

1. <https://linux-kernel-labs.github.io/refs/heads/master/lectures/syscalls.html>

1. <https://talk.dallasmakerspace.org/t/assembly-tutorial-syscalls-via-arm/24969>

1. <https://www.youtube.com/watch?v=kKtWsuuJEDs>

1. <https://www.youtube.com/watch?v=UW7a1n3VEFg>

1. <https://www.linkedin.com/pulse/system-calls-x86-64-svc-arm-mechanisms-practical-use-cases-david-zhu-y5ghc>

1. <https://en.eeworld.com.cn/news/mcu/eic599774.html>

1. <https://www.tutorialspoint.com/assembly_programming/assembly_system_calls.htm>

1. <https://man7.org/linux/man-pages/man2/syscall.2.html>

1. <https://www.reddit.com/r/C_Programming/comments/17rokbv/how_would_you_explain_to_a_5_year_old_what_a/>