Activity Monitors Unit (AMU) extension in AArch64 Linux

Author: Ionela Voinescu <ionela.voinescu@arm.com>

Date: 2019-09-10

This document briefly describes the provision of Activity Monitors Unit support in AArch64 Linux.

Architecture overview

The activity monitors extension is an optional extension introduced by the ARMv8.4 CPU architecture.

The activity monitors unit, implemented in each CPU, provides performance counters intended for system management use. The AMU extension provides a system register interface to the counter registers and also supports an optional external memory-mapped interface.

Version 1 of the Activity Monitors architecture implements a counter group of four fixed and architecturally defined 64-bit event counters.

- CPU cycle counter: increments at the frequency of the CPU.
- Constant counter: increments at the fixed frequency of the system clock.
- Instructions retired: increments with every architecturally executed instruction.
- Memory stall cycles: counts instruction dispatch stall cycles caused by misses in the last level cache within the clock domain.

When in WFI or WFE these counters do not increment.

The Activity Monitors architecture provides space for up to 16 architected event counters. Future versions of the architecture may use this space to implement additional architected event counters.

Additionally, version 1 implements a counter group of up to 16 auxiliary 64-bit event counters.

On cold reset all counters reset to 0.

Basic support

The kernel can safely run a mix of CPUs with and without support for the activity monitors extension. Therefore, when CONFIG_ARM64_AMU_EXTN is selected we unconditionally enable the capability to allow any late CPU (secondary or hotplugged) to detect and use the feature.

When the feature is detected on a CPU, we flag the availability of the feature but this does not guarantee the correct functionality of the counters, only the presence of the extension.

Firmware (code running at higher exception levels, e.g. arm-tf) support is needed to:

- Enable access for lower exception levels (EL2 and EL1) to the AMU registers.
- Enable the counters. If not enabled these will read as 0.
- Save/restore the counters before/after the CPU is being put/brought up from the 'off' power state.

When using kernels that have this feature enabled but boot with broken firmware the user may experience panics or lockups when accessing the counter registers. Even if these symptoms are not observed, the values returned by the register reads might not correctly reflect reality. Most commonly, the counters will read as 0, indicating that they are not enabled.

If proper support is not provided in firmware it's best to disable CONFIG_ARM64_AMU_EXTN. To be noted that for security reasons, this does not bypass the setting of AMUSERENR_EL0 to trap accesses from EL0 (userspace) to EL1 (kernel). Therefore, firmware should still ensure accesses to AMU registers are not trapped in EL2/EL3.

The fixed counters of AMUv1 are accessible though the following system register definitions:

- SYS_AMEVCNTR0_CORE_EL0
- SYS_AMEVCNTR0_CONST_EL0
- SYS_AMEVCNTR0_INST_RET_EL0
- SYS_AMEVCNTR0_MEM_STALL_EL0

Auxiliary platform specific counters can be accessed using SYS_AMEVCNTR1_EL0(n), where n is a value between 0 and 15.

Details can be found in: arch/arm64/include/asm/sysreg.h.

Userspace access

Currently, access from userspace to the AMU registers is disabled due to:

- Security reasons: they might expose information about code executed in secure mode.
- Purpose: AMU counters are intended for system management use.

Also, the presence of the feature is not visible to userspace.

Virtualization

Currently, access from userspace (EL0) and kernelspace (EL1) on the KVM guest side is disabled due to:

- Security reasons: they might expose information about code executed by other guests or the host.

Any attempt to access the AMU registers will result in an UNDEFINED exception being injected into the guest.