SVE Enablement in RMM for Realms

Arunachalam Ganapathy
01 June 2023
Agenda

**Brief Introduction**

- Scalable Vector Extension (SVE)
- Realm Management Extension (RME)
- Realm Management Monitor (RMM)

**SVE Support in RMM for Realms**

- Existing support in RMM for FP/NEON
- SVE Requirements and Design goals
- SVE Support – Lazy Enablement
- RMM SIMD abstraction layer
- RMM use of FPU at REL2
- SVE Hint bit handling

**Upstream Status**

- TF-RMM, tf-a-tests
- Related upcoming features: SME
Scalable Vector Extension (SVE)

- Added in Armv8-A (from v8.2)
- Key Features:
  - SIMD with larger and scalable vector length
  - Vector Length Agnostic (VLA)
  - Per lane predication
  - Gather load and Scatter Store
- Register state
  - Vector registers Z[0-31]. Scalable from 128-2048 bits. Shares the lower 128 bits with FP/NEON vector register.
  - Predicate registers P[0-15]
  - First Fault Register FFR
- Control Register
  - ZCR_ELx: Restricts Vector Length to current and lower ELs
Realm Management Extension (RME) and Realms

- RME added in Armv9-A (from v9.2)
  - RME allows to do Confidential Compute on Arm
  - Adds Root world, Realm world along with existing Non-secure and Secure world
  - Introduces Physical Address Space (PAS) for each world.
  - Dynamically assign granule from one PAS to another PAS, using Granule Protection Table (GPT)
- Realm Management Monitor (RMM)
  - Software component part of the system that implements Arm CCA
- Realms
  - RME in H/W and RMM in S/W provides support for NS VMM (hypervisor) to create protected execution environment called Realms.
Introduction

• Scalable Vector Extension (SVE)
• Realm Management Extension (RME)
• Realm Management Monitor (RMM)

SVE Support in RMM for Realms

• Existing support in RMM for FP/NEON
• SVE Requirements and Design goals
• SVE Support – Lazy Enablement
• RMM SIMD abstraction layer
• RMM use of FPU at REL2
• SVE Hint bit handling

Upstream Status

• TF-RMM, tf-a-tests
• Related upcoming features: SME
Support in TF-RMM v0.2.0 for FPU/NEON

- FP/NEON support in Realms
  - FPU support for Realms
  - Lazy enablement. REC runs with FPU traps enabled.
  - Upon FPU exception, RMM restores REC state and REC continues execution.
  - FPU state held in REC page

- Non Secure SVE state management
  - Special config for RMM to use FPU at REL2
    - Build flag RMM_FPU_USE_AT_REL2, OFF by default
    - When set, mbedtls uses AdvSIMD (which requires FPU registers) at REL2 for attestation/measurement functionality.
    - Needs crypto plugin in FVP
SVE support for Realms

Requirements:

• Support running multiple Realms with different vector length
• Handle FPU/SVE state, not leak/corrupt SVE Z/P registers b/w Realms and b/w Realm/NS world.
• Report H/W SVE status and max VL through RMM feature register 0
• Validate SVE flags, SVE VL on Realm creation. Propagate these flags to REC upon REC creation.
• Emulate sysregs read access to ID_AA64PFR0_EL1, ID_AA64ZFR0_EL1
• Support SVE hint bit (SMCCCv1.3)
# SVE support for Realms

<table>
<thead>
<tr>
<th>SVE in H/W</th>
<th>SVE enabled for Realms?</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>N/A</td>
<td>• Access to SVE results in undefined abort</td>
</tr>
</tbody>
</table>
| 1          | 0                       | • Emulate ID register read for Realms  
  • Clear SVE bits in ID_AA64PFR0_EL1  
  • Return 0 in ID_AA64ZFR0_EL1  
  • Inject undefined abort when Realm access SVE registers/instructions  
  • Manage NS SVE state  
  • Continue with lazy enablement of FPU |
| 1          | 1                       | • Manage NS SVE state  
  • Manage lazy enablement of SVE  
  • SVE VL for Realms configured by NS host |
Realm creation with SVE

- RMI Feature Register 0
  - SVE_EN: flag set to 1 enables SVE
  - SVE_VL: can range from 0 to 15. (128 – 2048 bits vector length)
  - NS Host queries this register and set the required values
- SVE Flags stored in realm data structure by RMM.
  - AUX granules hold the SVE state (this encompasses FPU state as well)
  - SVE state per REC is ~8700 bytes
SVE support - Lazy Enablement

- Handling REC SVE state
  - Like FP/NEON, SVE also uses Lazy enablement. REC runs with SVE traps enabled
  - Upon FPU or SVE exception, the whole SVE state of REC is restored after saving NS SVE state.
  - Realm’s SVE VL is programmed to ZCR_EL2 before the running the realm.

- Handling NS SVE state
  - While saving/restoring NS SVE state, ZCR_EL2 is programmed to the max SVE VL limited by EL3 through ZCR_EL3.
  - RMM saves and restores NS ZCR_EL2 register. RMM does not rely on EL3 to save and restore NS ZCR_EL2 but expects this to be passthrough.
RMM SIMD abstraction layer

• All FPU and SVE low level functions are abstracted under SIMD layer.
• Resides in lib/arch
• Currently SIMD type can be SIMD_FPU or SIMD_SVE, can be extended to add SIMD_SME
• Provides APIs for RMM core to initialize, configure, save and restore any type of SIMD state
• Manages
  o REC's SIMD state (could be FPU or SVE)
  o NS SIMD state (could be FPU or SVE)
  o RMM SIMD state (FPU only, when RMM_FPU_USE_AT_REL2=ON)
RMM Use of FPU at REL2

- Build flag `RMM_FPU_USE_AT_REL2=ON`. mbedtls uses FPU in RMM (at REL2)
- RMI and RSI Handlers calls attestation layer
- RMI Handlers
  - NS SIMD state is actively saved upon selected RMI handler that uses FPU
- RSI Handlers
  - REC SIMD state is actively saved for selected RSI handler, if REC has used SIMD (FPU or SVE) else NS SIMD state is saved.
- FPU_ALLOW()
  - Macro enables use of FPU. Disable FPU traps at EL2
  - Checks CPU live SIMD state is saved
SVE Hint bit handling in RMM (Planned)
SVE Hint bit handling in RMM (Planned)
Introduction

• Scalable Vector Extension (SVE)
• Realm Management Extension (RME)
• Realm Management Monitor (RMM)

SVE Support in RMM for Realms

• Requirements and Design Goals
• Existing support in RMM for FP/NEON
• SVE Support – Lazy Enablement
• SIMD abstraction layer
• RMM using FPU at REL2
• SVE Hint bit handling

Upstream Status

• TF-RMM, tf-a-tests
• Related upcoming features: SME
Upstream Status (v0.3.0)

• TF-RMM:
  • Patches merged in tf-rmm-0.3.0 release.
  • TF-RMM SVE support - patchset

• tf-a-tests:
  • Support added in TFTF framework for SVE tests in Realm payload

• Related upcoming features:
  • RMM support to handle NS SME state
  • RMM v1.0 specification does not cater for Realms to use SME
  • Patches (in review) for RMM SME support
Thank You
Danke
Gracias
Grazie
谢谢
ありがとう
Asante
Merci
감사합니다
धन्यवाद
شكراً
ধন্যবাদ
הם