arm

TF-A Tech Forum: RAS handling

Soby Mathew, Manish Pandey

23/03/2023

Agenda

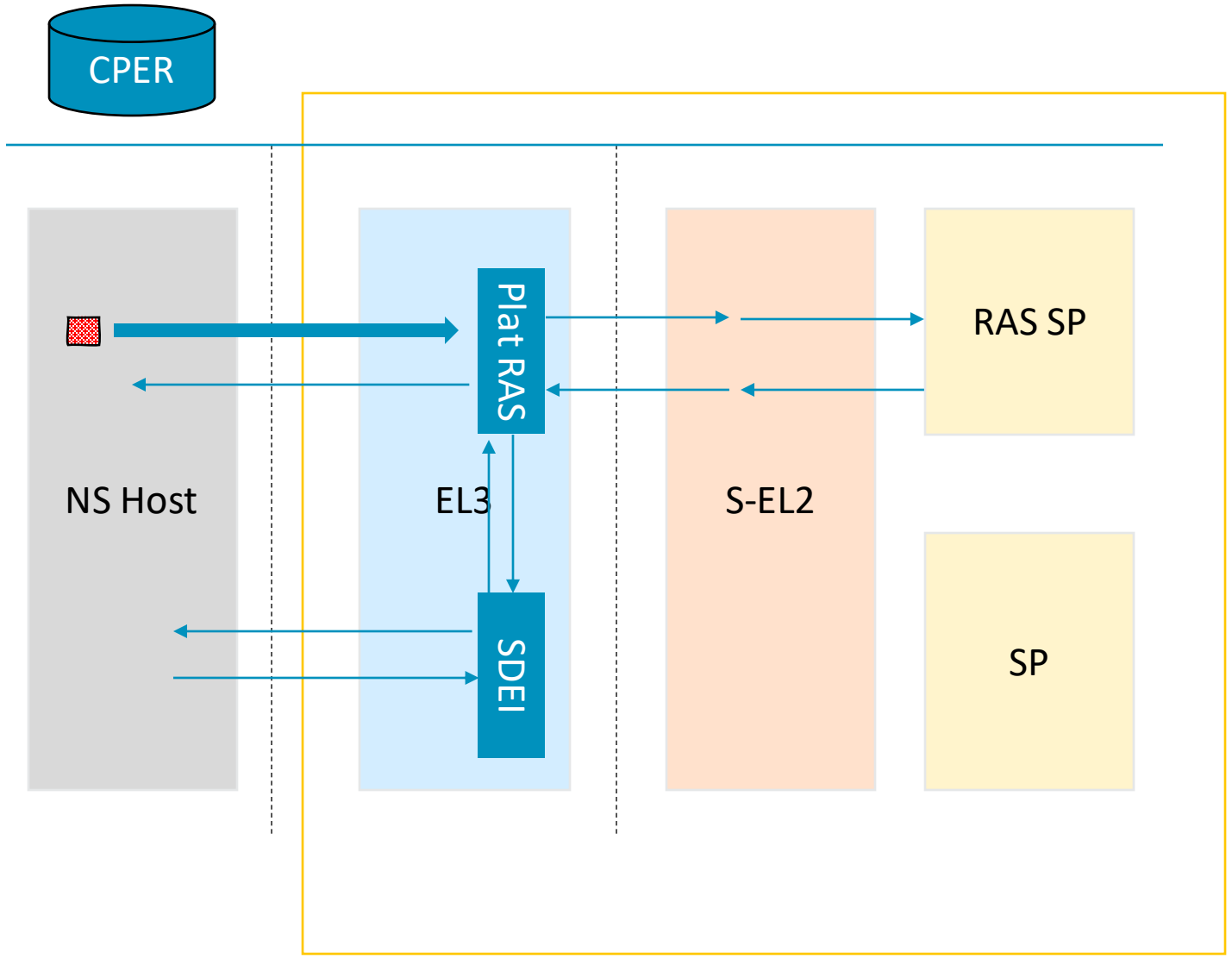
- Generic Concepts
  - RAS error handling philosophy
    - FFH
    - KFH
  - RAS flow in four world systems
- TF-A specific
  - EL3 exception handling
  - TF-A build macros
  - TF-A Changes
  - TFTF Tests
  - Ongoing work

RAS Error handling philosophy

- RAS Errors can be signalled via either External Abort (Sync EA and SError) or IRQ style interrupts
  - Former is referred as in-band error signal
- Two methods for handling RAS errors from *Non-Secure* World PoV.
  - Firmware first (FFH)
    - EA's and Error interrupts corresponding to NS nodes are handled first in Secure firmware and the original RAS exception is not visible to kernel.
      - Errors signaled back to NS world via suitable mechanism (like SDEI).
    - Kernel is prohibited from accessing the RAS error records directly.
      - Firmware creates CPER records for Kernel to navigate and process
  - Kernel first (KFH)
    - EA's attributed to kernel are handled first in kernel* and Error interrupts corresponding to NS nodes are configured as Non-Secure interrupts.
    - Kernel navigates the std error records directly.

    * There is a corner case when EA can pend at EL3 when entering EL3 from NS world. This needs special handling.

Typical FFH Flow for EA originating in NS World



- Contrast with flow for KFH
- Discuss flow for Error Interrupts
- Distinguish firmware domain and kernel domain.
  - Discuss error handling in firmware domain

arm

KFH Problem: Signal synchronized error back to NS

- An Async EA pertaining to NS world can be synchronized on entry to EL3 using IESB/esb().
  - This error would need to be signaled back to NS.
- The ideal solution is to inject SError back to the lower EL from EL3
  - This is how EL2 signals synchronized SA back to EL1.
  - EL2 has architectural support to inject Serror back to EL1
- EL3 does not have this capability.
  - Full SError emulation by EL3 is likely to be error prone.
- Almost full soln:
  - Set sctlr_el3.IESB. On taking an SMC exception to EL3, check for pending EA in isr_el1.A and if pending , eret to elr_el3 – 4. Pre-conditions : scr_el3.EA = 0, PSTATE.A = 1 on taking exception to El3.
    - The error is not consumed by EL3 but stays pending. The error is taken by the lower EL if the EL is the target of the Serror and the Error is unmasked.
    - Works well for the SMC calls into EL3.
  - FIQ vs SError priority is IMPDEF but Arch recommendation to pend SError if IESB is set.

Comparing KFH with FFH

Benefits of KFH
No Cycle stealing from NS for error handling
Less possibility for Recoverable RAS Errors from kernel domain panicking the system due to firmware issues.
Less context switch involved in handling the error (faster response and error propagation is limited).

Disadvantages of KFH
Upstream OS only support error records in Std format. Hence all nodes in system need to implement Std Error records.
No upstream solution to notify firmware domain errors back to kernel for book-keeping/maintenance records. [IMP DEF solutions possible].
More hardware compliance required for isolating kernel and firmware domain error handling.

KFH-FFH Hybrid Spectrum

- EA's Routed to Kernel
  - Errors synced by EL3 are signaled as SError
- Error interrupts are routed to Kernel
- Error nodes (both sysreg and MMIO) accessible by Kernel are std

- EA's Routed to Kernel
  - Errors synced by EL3 are signaled as SError
- Some error interrupts are routed to Kernel
  - Others routed to firmware. CPER created and signaled via IRQ to NS [1] for those routed to firmware.
- Error nodes (both sysreg and MMIO) accessible by Kernel are std

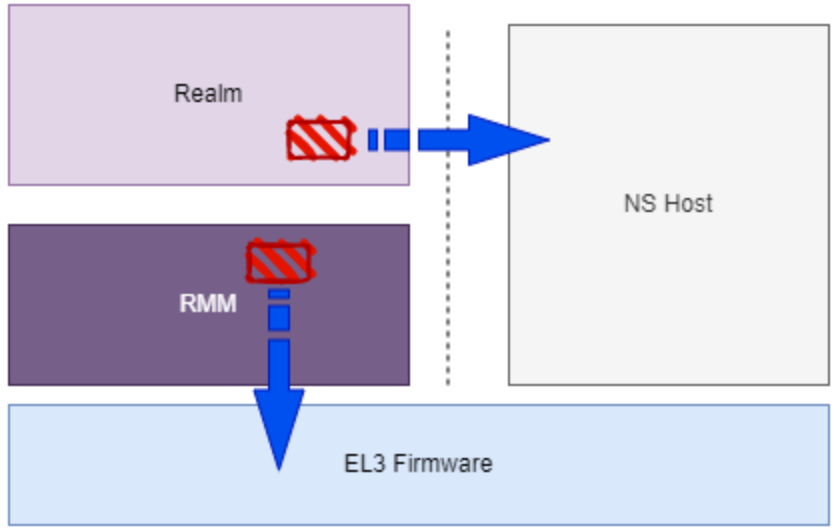
- EA's Routed to Kernel
  - Errors synced by EL3 are signaled as SError.
- All error interrupts are routed to Firmware
  - CPER created and signaled via IRQ to NS [1].
- Error nodes (both sysreg and MMIO) accessible by Kernel are std

- EA's Routed to firmware
  - Firmware creates CPER record.
  - Signaled back via SDEI
- All error interrupts are routed to Firmware

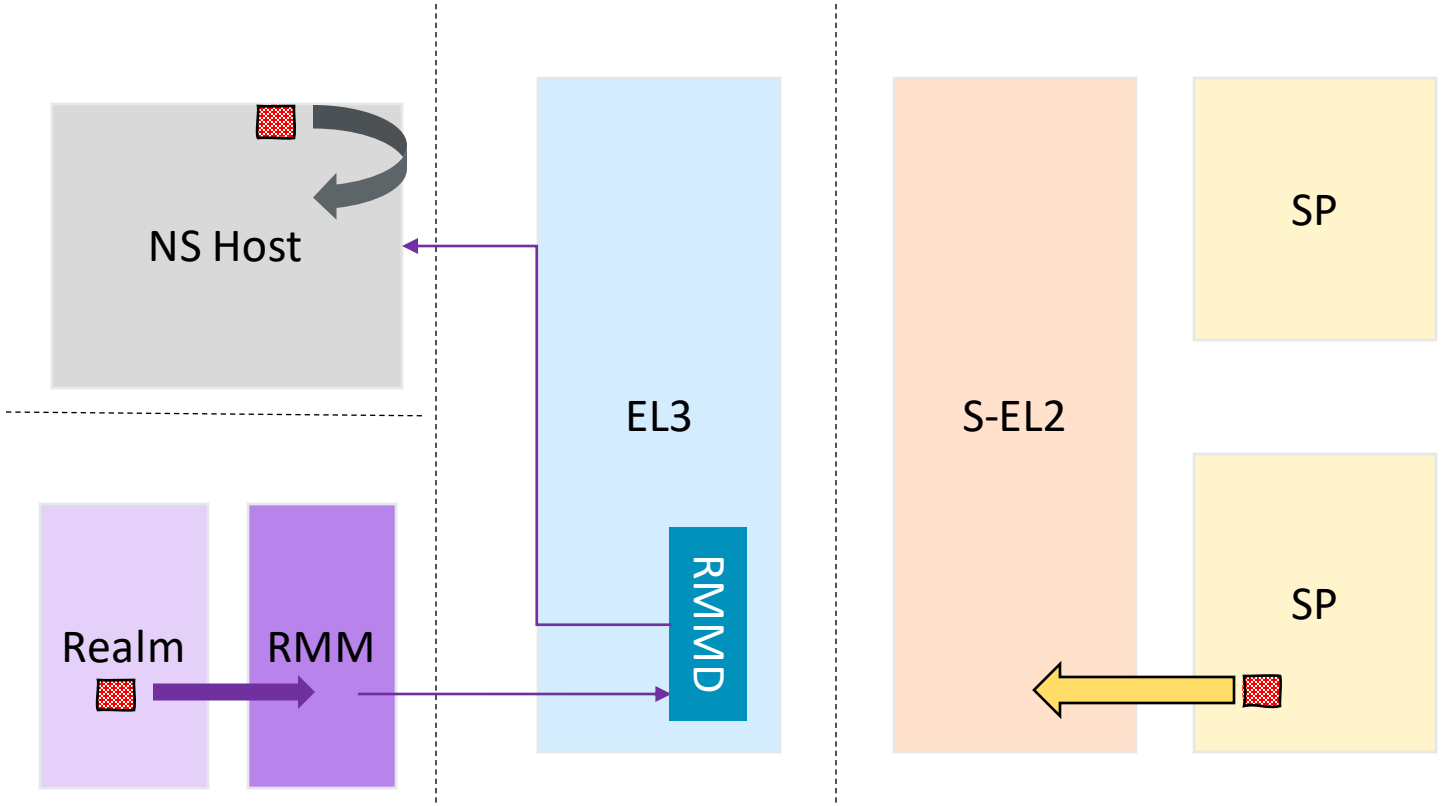
[1] UEFI spec allows signaling via IRQ or SDEI for FFH.

RAS Error handling for Realm World

- RAS Errors from Realms are considered kernel domain.
  - We can have KFH or FFH model for these errors.
  - KFH handling is naturally supported by RMM specification.
    - EAs and Interrupts for Realms are handled by NS.
  - In case of FFH, the secure world would need to be involved to handle the Error. (See backup for one possible flow).
- RMM is considered firmware domain.
  - Errors from RMM are to be contained with firmware domain.



KFH Flow RAS errors in 4 world System



arm

EL3 exception handling

- Vector entries for exceptions arising from current EL
  - Does not need any special attention
- Vector entries for exceptions arising from lower EL
  - Sync/IRQ/FIQ :
    - Check for EA due to synchronization barriers
      - FFH : Invoke platform handler
      - KFH : Reflect EA to lower EL (cater for EA masked in lower EL)
    - Handle original exception
  - SError :
    - FFH : Invoke platform handler
    - KFH : Never expected

TF-A build macros

TF-A build flag	FFH	KFH (default)
Support for RAS extension (Including IESB): ENABLE_FEAT_RAS	1	0 or 1
Firmware first handling : RAS_FFH_SUPPORT	1	0
EL3_EXCEPTION_HANDLING : TF-A exception handling framework	1	
HANDLE_EA_EL3_FIRST_NS : EA routed to EL3 (SCR_EL3.EA)	1	0
RAS_ALLOW_ERR_REC_ACCESS_NS : allow NS access (SCR_EL3.TERR)	0	1
SDEI_SUPPORT	1	
FAULT_INJECTION_SUPPORT : Lower EL's to inject fault for Testing		

*Existing RAS_EXTENSION is equivalent to "ENABLE_FEAT_RAS + RAS_FFH_SUPPORT"

TF-A changes

- Patches under review mp/feat_ras
  - Replace existing RAS_EXTENSION with FEAT_RAS
  - Provide support for Hybrid mode for RAS handling
  - Reflect SError to lower EL's (Fixing KFH problem)
  - Unify lower EL SError vector entry
  - Documentation
- Patches merged
  - Allow SErrors during EL3 execution
  - Remove saving of SCR_EL3 during EL3 entry
  - Some minor fixes and refactoring
    - Remove "handle_async_ea"
    - Remove "enter_lower_el_async_ea"
    - Free up scratch register(x30) vector entry path

TFTF tests

- Without RAS extension
  - verifies HANDLE_EA_EL3_FIRST_NS feature
  - test_inject_serror() : Cause SError in lower EL, ensure it traps to EL3
  - test_inject_syncEA() : Cause sync EA in lower EL, ensure it traps to EL3
- KFH
  - test_ras_kfh() : Register SError handler, inject SError, ensure the handler being called.
  - test_ras_kfh_esb() : Tests KFH problem mentioned earlier
- FFH
  - test_single_fault() : Register SDEI event, Inject Unrecoverable error, error traps in EL3, Get notified back to NS through SDE
  - test_uncontainable() : Inject uncontainable error, FVP does not have handler for this panic.
- * FVP platform for testing, need help from community to test it on real HW.

Ongoing work

- Cover FEAT_RAS_EXTENSION in feat detection mechanism
  - No need for platform to enable it explicitly
- Make KFH default configuration
- FEAT_IESB support
- Make some of the build flag internal
  - Get enabled by default if FFH support is there
- Introduce test to exercise full flow of FFH
  - Involving Secure Partitions in the flow
  - SDEI notifying NS world (already there)
- Introduce test for KFH
  - IRQ/FIQ error reflection back to NS
- Ensure that all possible exception handling paths are covered in testing

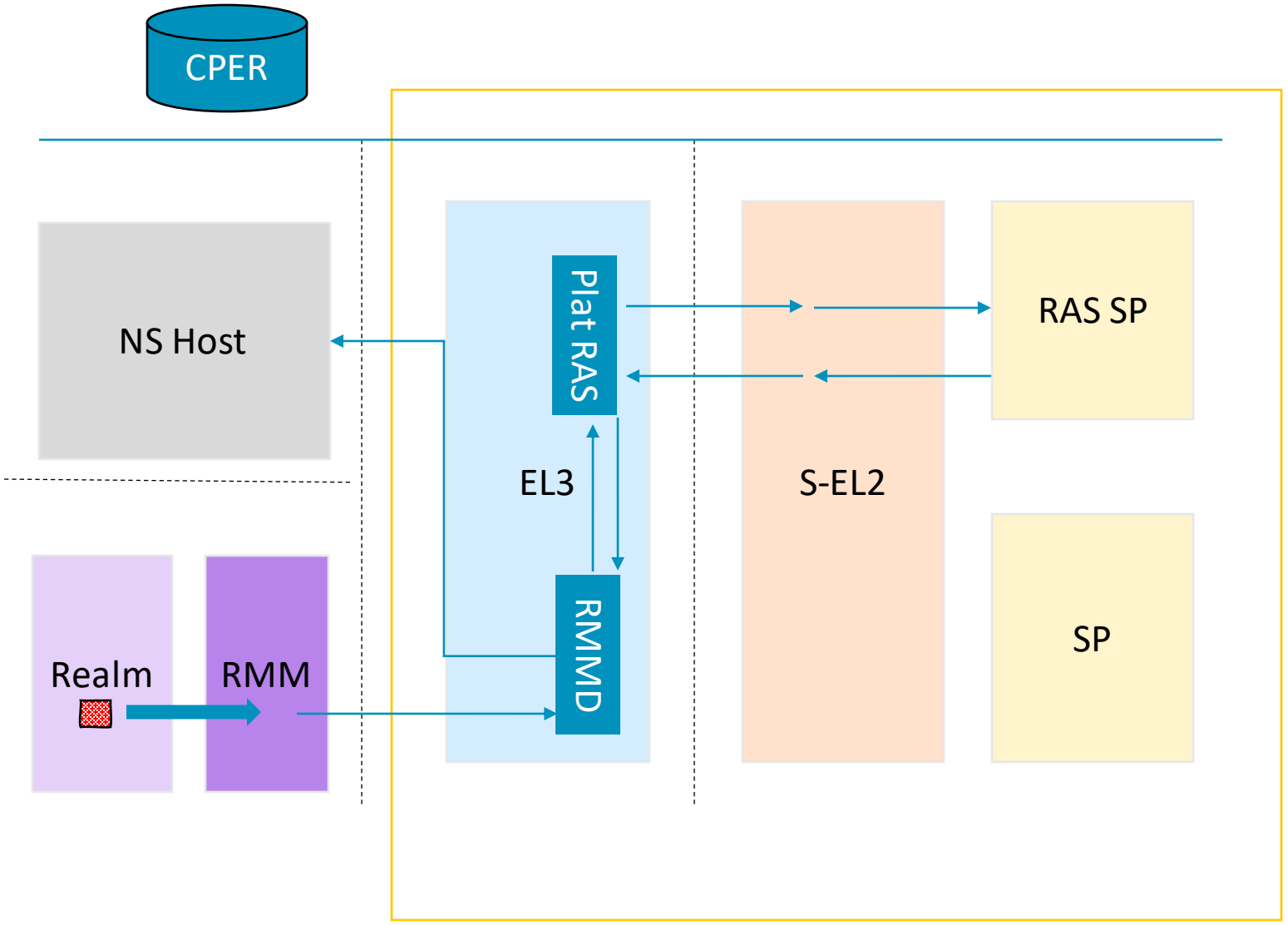
arm

Thank You
Danke
Gracias
Grazie
谢谢
ありがとう
Asante
Merci
감사합니다
धन्यवाद
Kiitos
شكرًا
ধন্যবাদ
תודה

Base assumptions for the RAS handling

1. Errors can be attributed and contained (if containable) to a domain.
2. The Uncorrectable Errors from any shared error nodes should only cause in-band RAS exceptions.
   - SBSA mandates this for memory/cache controllers.
   - Since Error interrupts are assigned either NS/S as part of boot and not changed thereafter, this is more critical for KFH.
3. RAS error nodes and controls must only be visible to the domain responsible for first handling.
   - In FFH, the RAS error nodes must not be visible to NS world – both Sysreg and MMIO.
   - In KFH, this would apply to error nodes exclusive to firmware domain.
     - For shared nodes in KFH, NS world should not be able to mask/disable error generation. eg: ERR<n>CTRL.UE in error record for memory/cache controllers should be RO/WI for NS.

Possible FFH Flow for EA originating in Realm



- RMMD needs to inspect SMCs between Realm and NS
- SDEI is not used for notification for errors from Realm
- Increases the amount of Secure firmware to be deployed for Arm CCA.
  - Realm world error handling will not work unless a platform specific RAS SP + glue code is present.

arm

TF-A exception handling

- Sync/IRQ/FIQ vector from lower EL
  - check_and_unmask_ea
    - FFH : esb -> check DISR_EL1 -> handle lower el ea if EA is present- > handover to platform
    - KFH : dsb (or IESB) -> check ISR_EL1 -> reflect ea to lower_el (cater for EA is masked in lower EL)
      - eret for IRQ/FIQ
      - elr_el3 – 4 for sync exception
- SError vector for lower EL
  - FFH : esb (to sync all errors)-> handle lower el ea -> handover to platform
    - If HANDLE_EA_EL3_FIRST_NS (without FEAT_RAS) is set -> handover to platform
  - KFH: Not expected