Brief Updates on Interrupt Handling in TF-M

Kevin Peng
August 2021
Agenda

• Interrupt Handling Support in TF-M
• High-Level Implementation Details
• How to Enable an Interrupt in TF-M
• Q & A
Interrupt Handling Support

• Both interrupt handling types defined in FF-M v1.1 are supported
  • Second-Level Interrupt Handling (SLIH)
    – Stable maintenance
  • First-Level Interrupt Handling (FLIH)
    – Initial support
    – Subject to change
High-Level Implementation Details

SLIH

SPM

Set Signal

Scheduler

Secure Partition

psa_wait

Interrupt Handling
High-Level Implementation Details

Privileged FLIH

- Applies to Partitions in L1 & PRoT Partitions in L2 and L3
The first SVC is to retain the Exception priority of the interrupt for the FLIH Function

- When Exception Return to FLIH Function, it is the SVC who’s cleared while the IRQ keeps active
How to Enable an Interrupt in TF-M

• Assigning the interrupt to a Secure Partition.
• Granting the Secure Partition access permissions to the MMIO of the interrupt.
• Configurating the interrupt.
• Integrating the interrupt handling function generated by TF-M to the Vector Table.
How to Enable an Interrupt in TF-M

Assigning the interrupt to a Secure Partition

IRQ items in FF-M v1.0

```c
"irqs": [
    { "source": "5", "signal": "DUAL_TIMER_SIGNAL" },
    { "source": "TIMER_1_SOURCE", "signal": "TIMER_1_SIGNAL" }
],
```

Build System

```c
#define {{signal}} <value>
```

Build System

```c
#define {{name}}_SIGNAL <value>
```

Build System

```c
psa_flih_result_t {{name}}_flih(void)
```

MMIO Regions

```c
"mmio_regions": [
    {
        "name": "TFM_PERIPHERAL_TIMER0",
        "permission": "READ-WRITE"
    }
],
```

```c
#define TFM_PERIPHERAL_TIMER0 (&tfm_peripheral_timer0)
```

```c
struct platform_data_t {
    uint32_t periph_start;
    uint32_t periph_limit;
    int16_t periph_ppc_bank;
    int16_t periph_ppc_loc;
};
tfm_spm_hal_configure_default_isolation
```
How to Enable an Interrupt in TF-M

Granting the Secure Partition access permissions to the device of the interrupt.

• The MMIO Regions
• The Device Drivers
  • target_sources(some_partition_lib
    PRIVATE
    some_driver_code.c)
How to Enable an Interrupt in TF-M

Configurating the interrupt.

• Setting Priority
  • The priority value must be less than the value of ``PendSV`` (0x80) and greater than the value of ``SVC`` (0x0).
  • HAL API:
    – `enum tfm_plat_err_t tfm_spm_hal_set_secure_irq_priority(IRQn_Type irq_line);`

• Targeting Interrupts to Secure
  • HAL API:
    – `enum irq_target_state_t tfm_spm_hal_set_irq_target_state(IRQn_Type irq_line, enum irq_target_state_t target_state);`
How to Enable an Interrupt in TF-M

Integrating the interrupt handling function generated by TF-M to the Vector Table.

- TF-M generates interrupt handling functions for each interrupt assigned to Secure Partitions in building
  - `void irq_{source}_Handler(void)` or
  - `void {{irq.source}}_Handler(void)`
    - `-> spm_interrupt_handler`
- Platforms integrate the handling functions in their own manner
References

• Secure IRQ Integration guide - [Docs: Add Secure IRQ integration guide](https://git.trustedfirmware.org/TF-M/tf-m-tests.git/tree/test/test_services/tfm_flih_test_service) (Patch in review)

• FLIH Test Partition: [https://git.trustedfirmware.org/TF-M/tf-m-tests.git/tree/test/test_services/tfm_flih_test_service](https://git.trustedfirmware.org/TF-M/tf-m-tests.git/tree/test/test_services/tfm_flih_test_service)

• SLIH Test Partition: [https://git.trustedfirmware.org/TF-M/tf-m-tests.git/tree/test/test_services/tfm_slih_test_service](https://git.trustedfirmware.org/TF-M/tf-m-tests.git/tree/test/test_services/tfm_slih_test_service)