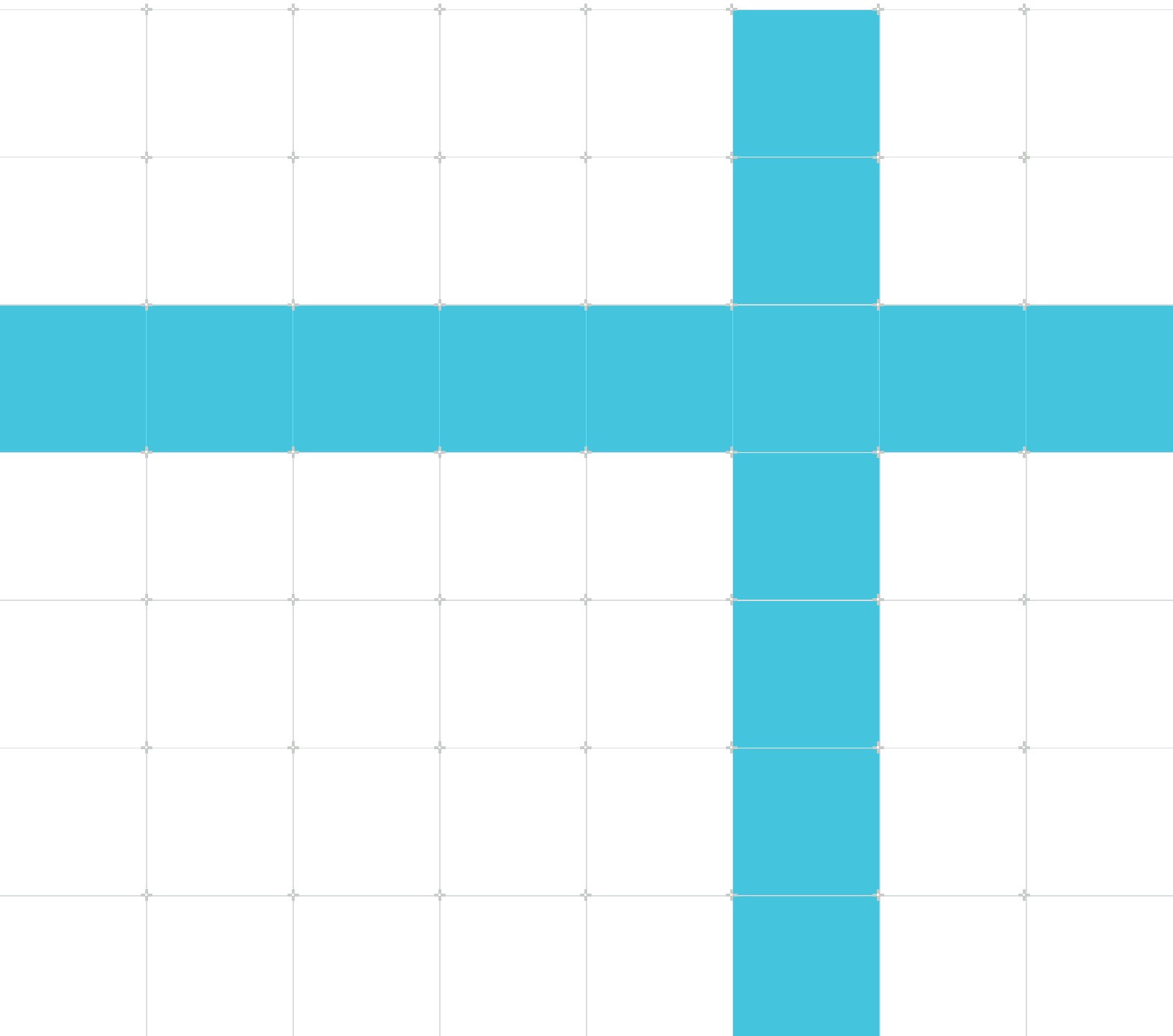
Cortex-M IDAU Frequently Asked Questions

Version 1.0

Issue 01
109508_0100_01_en



Cortex-M IDAU Frequently Asked Questions

Release information

Document history

Issue	Date	Confidentiality	Change
0100-01	4 December 2023	Non-Confidential	Initial release

Proprietary Notice

Confidentiality Status

This document is Non-Confidential. The right to use, copy and disclose this document may be subject to license restrictions in accordance with the terms of the agreement entered into by Arm and the party that Arm delivered this document to.

Unrestricted Access is an Arm internal classification.

Product Status

The information in this document is Final, that is for a developed product.

Feedback

Arm welcomes feedback on this product and its documentation. To provide feedback on the product, create a ticket on https://support.developer.arm.com

To provide feedback on the document, fill the following survey: https://developer.arm.com/documentation-feedback-survey.

Inclusive language commitment

Arm values inclusive communities. Arm recognizes that we and our industry have used language that can be offensive. Arm strives to lead the industry and create change.

We believe that this document contains no offensive language. To report offensive language in this document, email terms@arm.com.

Contents

1. Cortex-M IDAU Frequently Asked Questions....6
1.1 What is an IDAU?....6
1.2 Are there any documents that explain IDAU implementation and configuration?....6
1.3 What is the difference between an SAU and an IDAU?....6
1.4 What are the possible Security attribution unit configurations available in Cortex-M/Armv8-M based processors?....7
1.5 Which Security attribution unit has higher precedence – SAU or IDAU?....8
1.6 Can the IDAU override the security attribution to either Non-secure or Exempt attribute during the out of reset sequence?....9
1.7 Why do I need an IDAU when I already have an SAU?....9
1.8 Do we have any internal processor register that shows the IDAU security attribution? How do I verify whether the security attribution is from SAU or IDAU?....9
1.9 Does the IDAU affect the processor's internal PPB registers accesses?....9
1.10 Do Cortex-M and Armv8-M based processors implement IDAU within the processor level?....10
1.11 What is the role of the IDAU interface in Cortex-M processors?....10
1.12 Does Arm provide an example IDAU implementation?....11
1.13 I want to implement an IDAU that has configurable regions. Do IDAU and SAU region numbers have to be identical?....11
1.14 Is it a requirement that the Full IDAU returns different IDs for memory regions that differ in security attributes or are non-contiguous?....11
1.15 To simplify my IDAU design, can I mark an invalid memory region with the exempt attribute?....12

1. Cortex-M IDAU Frequently Asked Questions

This document provides answers for frequently asked questions related to the Implementation Defined Attribution Unit (IDAU) in Cortex-M/Armv8-M based systems.

1.1 What is an IDAU?

IDAU stands for Implementation Defined Attribution Unit. The IDAU defines memory regions as Secure, Non-secure, Non-secure Callable, or exempt from Security checking. The IDAU provides address lookups and generates Security attributes for the address being accessed. If the processor is designed to handle multiple concurrent transfers, it must have multiple IDAUs. These IDAUs must have consistent security attribute mappings.

1.2 Are there any documents that explain IDAU implementation and configuration?

Yes. The TrustZone Technology Microcontroller System Hardware Design Concepts User Guide explains how a system memory can be partitioned. It also provides some of the recommendations for an IDAU implementation from a system designer point of view.

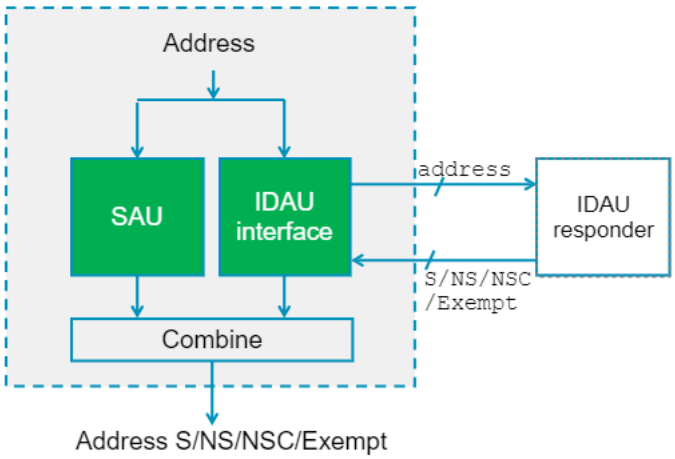
1.3 What is the difference between an SAU and an IDAU?

In Cortex-M/Armv8-M based processors that implement the Security Extensions, the Security attribution of a memory region is controlled by a combination of two attribution units:

- Security Attribution Unit (SAU)
- Implementation Defined Attribution Unit (IDAU)

The SAU is programmable in Secure state and has a programmers' model similar to the Memory Protection Unit (MPU). The IDAU is external to the processor and requires implementation by the chip designer.

Figure 1-1: IDAU responder



Both the SAU and IDAU are independently optional, and the final memory Security attribution is determined by the stricter Security attribution specified by the SAU or by the IDAU. For more details refer Section:2 in TrustZone Technology Microcontroller System Hardware Design Concepts User Guide.

1.4 What are the possible Security attribution unit configurations available in Cortex-M/Armv8-M based processors?

There are three possible configurations of the attribution units:

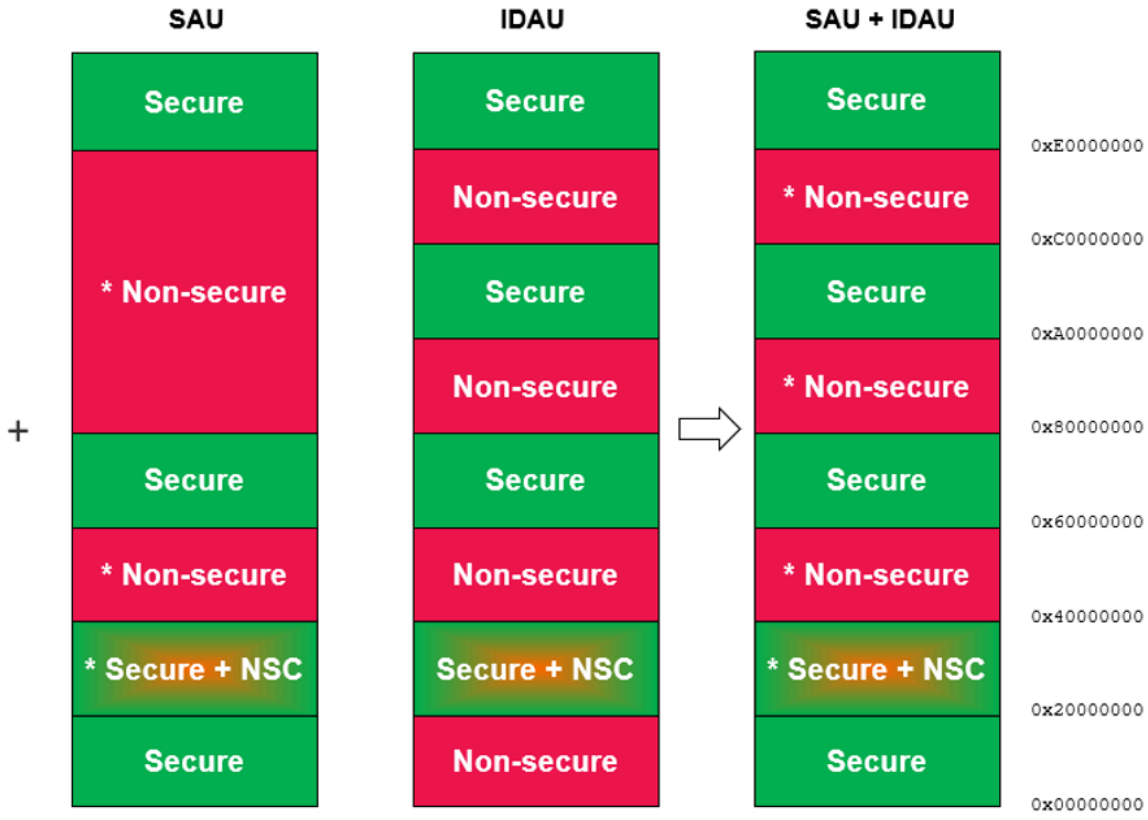
1. SAU only – Implemented within the processor. Memory partition can only be based on the number of SAU regions implemented in the system. Software programming of SAU regions is possible.
2. IDAU only – Implemented by chip designer at system level. Recommended for systems that have a fixed memory map configuration and do not need run-time configuration of security attributes.
3. SAU and IDAU – Gives more flexibility in memory partitioning because both the IDAU and SAU are available in a system.

1.5 Which Security attribution unit has higher precedence – SAU or IDAU?

Higher precedence is not based on the attribution unit, rather it is based on the security attribution type.

The IDAU does not overrule the SAU, nor does the SAU overrule the IDAU. Instead, the processor takes the more restrictive security attribute of the two. For example, if the IDAU marks a region as Secure but the SAU marks it as Non-secure, the processor treats it as Secure. Similarly, if the SAU marks a region as Secure and the IDAU marks it as Non-Secure, then the processor treats it as Secure. This is shown in the following figure.

Figure 1-2: Security attribution precedence



Note that any memory marked as exempt will be accessible from both Secure and Non-secure states. Arm strongly recommends that the exempt attribute is only used for read only areas of memory like the ROM tables.

1.6 Can the IDAU override the security attribution to either Non-secure or Exempt attribute during the out of reset sequence?

No.

Out of reset, the processor always enters into Secure state. Generally, this is handled by the SAU_CTRL.ALLNS bit. When the SAU_CTRL.ALL_NS bit is zero, the entire memory (except PPB) space is marked as Secure. After the reset sequence, Secure boot software can allow the security level for all memory regions to be specified by an IDAU by disabling all the SAU regions and setting SAU_CTRL.ALLNS to 1.

1.7 Why do I need an IDAU when I already have an SAU?

Consider an implementation that has only an SAU. The number of SAU regions is defined by the chip designer, for example 4 or 8. The number of SAU regions might not provide the software programmer with enough flexibility. Alternatively, chip designers can implement an IDAU to define a default security attribution for the system's memory map. The SAU can be optionally used to override the IDAU's security attributes for some parts of the memory where default settings are not suitable. It is also possible to implement a fixed (non-programmable) IDAU configuration for systems that require lower gate counts.

1.8 Do we have any internal processor register that shows the IDAU security attribution? How do I verify whether the security attribution is from SAU or IDAU?

From a software programmer's perspective, a Test Target (`TT`) instruction can be used to check whether the memory address configuration is from either SAU or IDAU. Also, as a part of Cortex-M Security Extensions (CMSE), `TT` intrinsics are available to get the security attribution information. For more information, refer to Armv8-M Security Extensions – Requirements on Development Tools.

1.9 Does the IDAU affect the processor's internal PPB registers accesses?

Internal PPB registers for peripherals like MPU, SysTick, DWT, FPB, and so on, are exempt from memory attribution by both the SAU and IDAU. For more information on the memory regions that are exempted by default, refer to the Armv8-M Architecture Reference Manual Rules R-LDTN and R-FGDW.

1.10 Do Cortex-M and Armv8-M based processors implement IDAU within the processor level?

No.

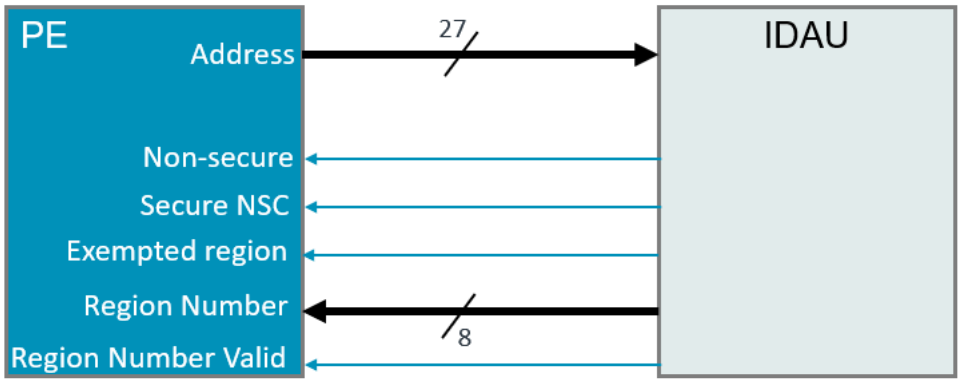
The IDAU is a simple hardware unit that needs to be designed by chip vendors at the SoC level. Cortex-M processors (based on Armv8-M architecture such as Cortex-M33 or Cortex-M55) contain common IDAU interfaces. When an address is presented on these IDAU interfaces, a combinatorial response on the security attribute (for example, Secure, Non-secure, NSC, or Exempt) is expected back from the IDAU logic corresponding to the memory address.

1.11 What is the role of the IDAU interface in Cortex-M processors?

In general, the IDAU interface is processor-specific. Further details on a specific Cortex-M processor's IDAU interface are available in the processor's Technical Reference Manual (TRM) and Implementation and Integration Manual (IIM).

However, there is a high similarity between the IDAU interfaces on different Cortex-M processors. Here is an example IDAU interface.

Figure 1-3: Example IDAU interface



- The address granularity of Secure/Non-secure partitions could be as low as 32-bytes.
- The IDAU region number value for each region must be unique. All Non-secure regions should have unique valid region numbers.
- The IDAUs allows designers to define exempted regions. To reduce the risk of code injection attacks, exempted regions must always be in Non-executable address range.

- If there are two IDAU interfaces in a processor (For example the Cortex-M33), then the mapping of the security attributes for the two IDAUs must be identical.

Each Cortex-M processor may provide a different number of IDAU interfaces. For example:

1. Cortex-M33 has two IDAU interfaces - IDAU-A and IDAU-B. IDAU-A is for instruction accesses and IDUA-B is for data accesses.
2. Cortex-M55 has three IDAU interfaces - IDAU-A is for instruction access and IDAU-B/IDAU-C interfaces are for data accesses.
3. Cortex-M85 has five IDAU interfaces - IDAU-A is for instruction access and the remaining IDAU interfaces are for data accesses.

In theory it is possible to design the IDAU to be programmable. However, the signals on the IDAU interface are likely to be on timing-critical paths which can make a complex IDAU impractical and could result in a higher gate count in the design. As a result, it is recommended that the IDAUs provide simple memory mapping with limited configurability.

1.12 Does Arm provide an example IDAU implementation?

No.

Arm does not provide IDAU as an end deliverable to customers. Rather it is expected that chip or SoC designers will implement IDAU logic as per their requirements.

1.13 I want to implement an IDAU that has configurable regions. Do IDAU and SAU region numbers have to be identical?

No, the region numbers need not be same.

1.14 Is it a requirement that the Full IDAU returns different IDs for memory regions that differ in security attributes or are non-contiguous?

Yes. To achieve better security in a Full IDAU configuration, it is required to return three different IDs (Secure, Non-secure, and NSC) to allow Secure software to perform reliable security verification using the `TT` instruction.

1.15 To simplify my IDAU design, can I mark an invalid memory region with the exempt attribute?

Marking an invalid memory region (that is, memory without region numbers) with an exempt attribute has serious downsides. Specifically:

- The `TT` instruction is used by some CMSE library functions to check ranges of addresses. If a region does not have a number, these library functions might not behave as expected. For example, if a pointer to a region without a number is passed from Non-secure software to Secure software.
- There is no way to change the security attributes on a region that is marked as exempt, and that region will always be accessible to both security states at all times.

Hence it is recommended that setting a region to be Non-secure by default is a safe option as it can be overridden by the SAU. Also, it is highly recommended to mark a memory region as exempt only where there are no security concerns within the system. Marking large areas of address space as exempt by default can lead to dangerous security issues.