# DMAMUX internal peripheral

# Contents



Stable: 04.02.2020 - 15:42 / Revision: 04.02.2020 - 15:34

Stable: 04.02.2020 - 15:42 / Revision: 04.02.2020 - 15:04

## Contents

# 1 Article purpose

The purpose of this article is to:

- briefly introduce the DMAMUX peripheral and its main features
- indicate the level of security supported by this hardware block
- explain how each instance can be allocated to the three runtime contexts and linked to the corresponding software components
- explain, when necessary, how to configure the DMAMUX peripheral.

# 2 Peripheral overview

The **DMAMUX** peripheral is used to perform requestor line (or device controller) selection for each channel from DMA instances: there is a single DMAMUX instance to cover both DMA1 and DMA2.

## 2.1 Features

Refer to the STM32MP15 reference manuals for the complete list of features, and to the software components, introduced below, to see which features are implemented.

## 2.2 Security support

The DMAMUX is a **non secure** peripheral.

# 3 Peripheral usage and associated software

## 3.1 Boot time

The DMAMUX is not used at boot time.

## 3.2 Runtime

### 3.2.1 Overview

The DMAMUX manages DMA1 and DMA2 requestor line selection via different registers so it is possible to concurrently access to DMAMUX from Cortex®-A7 non-secure and Cortex®-M4 contexts, as far as each core is only configuring the requestor lines for the DMA instances (DMA1 and/or DMA2) assigned to itself.

Finally, DMAMUX can be allocated to:

- the Arm®Cortex®-A7 non-secure core to be controlled in Linux® by the dmaengine framework

or

- the Arm®Cortex®-M4 to be controlled in STM32Cube MPU Package by the DMA HAL driver

### 3.2.2 Software frameworks

| Domain | Peripheral | Software components | | | Comment |
|---|---|---|---|---|---|
| OP-TEE | Linux | STM32Cube | | | |
| Core/DMA | DMAMUX | | Linux dmaengine framework | STM32Cube DMAMUX driver | |

### 3.2.3 Peripheral configuration

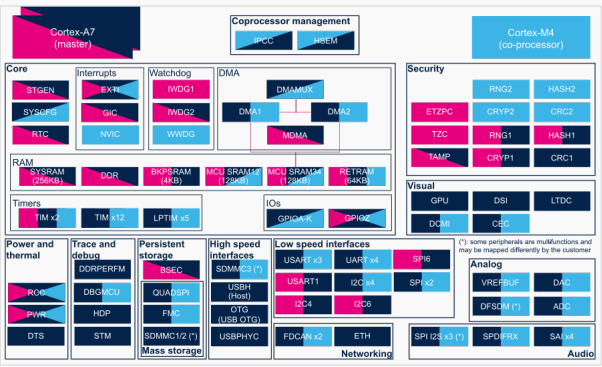
The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration can be done alone via the STM32CubeMX tool for all internal peripherals, and then manually completed (particularly for external peripherals), according to the information given in the corresponding software framework article.

### 3.2.4 Peripheral assignment

**Check boxes** illustrate the possible peripheral allocations supported by STM32 MPU Embedded Software:

- means that the peripheral can be assigned () to the given runtime context.
- is used for system peripherals that cannot be unchecked because they are statically connected in the device.

Refer to How to assign an internal peripheral to a runtime context for more information on how to assign peripherals manually or via STM32CubeMX.
The present chapter describes STMicroelectronics recommendations or choice of implementation. Additional possiblities might be described in STM32MP15 reference manuals.



Internal peripherals

| Domain | Peripheral | Runtime allocation | | | | Comment |
|---|---|---|---|---|---|---|
| Instance | Cortex-A7 secure (OP-TEE) | Cortex-A7 non-secure (Linux) | Cortex-M4 (STM32Cube) | | | |
| Core/DMA | DMAMUX | DMAMUX | | | | Shareable (multiple choices supported) |

Cortex®

*Arm® is a registered trademark of Arm Limited (or its subsidiaries) in the US and/or elsewhere.*

Linux® is a registered trademark of Linus Torvalds.

Microprocessor Unit

Open Portable Trusted Execution Environment

Direct Memory Access

Stable: 13.10.2020 - 08:29 / Revision: 13.10.2020 - 08:29

**Invalid target:** no reviewed revision corresponds to the given ID.

Return to DMA internal peripheral.

Stable: 22.03.2021 - 08:50 / Revision: 22.03.2021 - 08:43

**Invalid target:** no reviewed revision corresponds to the given ID.

Return to Dmaengine overview.

Stable: 08.03.2021 - 16:13 / Revision: 16.02.2021 - 17:11

**Invalid target:** no reviewed revision corresponds to the given ID.

Return to How to assign an internal peripheral to a runtime context.

Stable: 31.03.2021 - 11:58 / Revision: 23.03.2021 - 14:07

**Invalid target:** no reviewed revision corresponds to the given ID.

Return to STM32CubeMP1 architecture.

Stable: 23.09.2020 - 13:22 / Revision: 12.06.2020 - 13:25

**Invalid target:** no reviewed revision corresponds to the given ID.

Return to STM32CubeMX.
Stable: 17.11.2020 - 17:06 / Revision: 10.11.2020 - 07:49

**Invalid target:** no reviewed revision corresponds to the given ID.

Return to STM32MP15 resources.
Stable: 26.03.2021 - 11:32 / Revision: 12.03.2021 - 11:07

**Invalid target:** no reviewed revision corresponds to the given ID.

Return to STM32MPU Embedded Software architecture overview.