DMAMUX internal peripheral

Contents

DMAMUX internal peripheral

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

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

<table>
<tr><th>Do</th><th>Peri</th><th colspan="3">Software frameworks</th><th>Comment</th></tr>
<tr><td>mai
n
Cor
tex
-A7
sec
ure
(O
P-
TE
E)</td><td>Cor
tex
-A7
no
n-
sec
ure
(Li
nux
)</td><td>Cortex-M4

(STM32Cube)</td><td colspan="3"></td></tr>
<tr><td>Co
re/
D
M
A</td><td>D
M
A
M
U
X</td><td></td><td>Linux dmaengine framework</td><td>STM32Cube DMAMUX driver</td><td></td></tr>
</table>

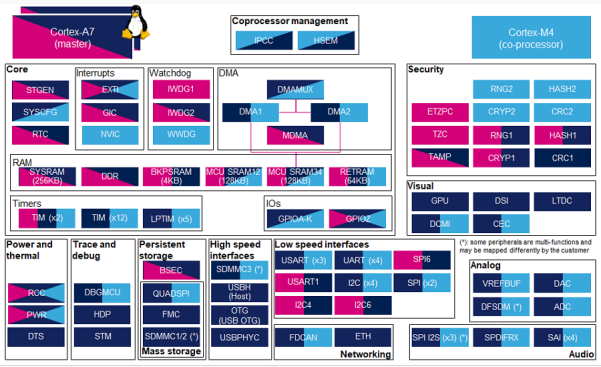
3.2.3 Peripheral configuration

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.



Internal peripherals

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.

Domain	Periph	Runtime allocation				Comment
Instance	Cortex-A7 secure (OP-TEE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)			
Core/DM	DMAMUX	DMAMUX				Shareable (multiple choices suppo

Domain	Peripheral	Runtime allocation				Comment
						rted)

Microprocessor Unit

Open Portable Trusted Execution Environment

Direct Memory Access

Permission error

Stable: 11.02.2019 - 11:21 / Revision: 18.01.2019 - 16:05

You do not have permission to read this page, for the following reason:

The action "Read pages" for the draft version of this page is only available for the groups ST_editors, ST_readers, Selected_editors, sysop, reviewer

Permission error

Stable: 24.06.2020 - 17:52 / Revision: 24.06.2020 - 12:32

You do not have permission to read this page, for the following reason:

The action "Read pages" for the draft version of this page is only available for the groups ST_editors, ST_readers, Selected_editors, sysop, reviewer

Permission error

Stable: 11.06.2020 - 12:39 / Revision: 11.06.2020 - 09:18

You do not have permission to read this page, for the following reason:

The action "Read pages" for the draft version of this page is only available for the groups ST_editors, ST_readers, Selected_editors, sysop, reviewer

Permission error

Stable: 21.02.2020 - 08:39 / Revision: 04.02.2020 - 15:22

You do not have permission to read this page, for the following reason:

The action "Read pages" for the draft version of this page is only available for the groups ST_editors, ST_readers, Selected_editors, sysop, reviewer

Permission error

Stable: 31.01.2020 - 13:04 / Revision: 31.01.2020 - 13:02

You do not have permission to read this page, for the following reason:

The action "Read pages" for the draft version of this page is only available for the groups ST_editors, ST_readers, Selected_editors, sysop, reviewer

Permission error

Stable: 15.10.2019 - 11:55 / Revision: 15.10.2019 - 11:55

You do not have permission to read this page, for the following reason:

The action "Read pages" for the draft version of this page is only available for the groups ST_editors, ST_readers, Selected_editors, sysop, reviewer

Permission error

Stable: 22.06.2020 - 09:50 / Revision: 22.06.2020 - 09:49

You do not have permission to read this page, for the following reason:

The action "Read pages" for the draft version of this page is only available for the groups ST_editors, ST_readers, Selected_editors, sysop, reviewer