SDMMC internal peripheral

Contents

SDMMC internal peripheral

Stable: 14.05.2020 - 07:13 / Revision: 14.05.2020 - 07:12

Contents

1 Article purpose

The purpose of this article is to

- briefly introduce the SDMMC 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 SDMMC peripheral.

2 Peripheral overview

The **SDMMC** peripheral is used to interconnect STM32 MPU to SD memory cards, SDIO and MMC devices.

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

SDMMC1/2/3 instances are either **non-secure** or **secure** peripherals (under ETZPC control).

- **When an SDMMC instance is secure internal, the DMA cannot be used to perform data transfers.**
- **STMicroelectronics does not provide secure MMC driver (see below chapter)**

3 Peripheral usage and associated software

3.1 Boot time

SDMMC1/2 instances can be used to support memory boot on SD or MMC Flash devices.

The SDMMC3 is not used at boot time.

The SDMMC instances are ordered by address in the device tree arch/arm/boot/dts /stm32mp157c.dtsi file:

```
sdmmc3: sdmmc@48004000 {
...
sdmmc1: sdmmc@58005000 {
...
sdmmc2: sdmmc@58007000 {
```

By default, in OpenSTLinux distribution, **sdmmc3 is disabled** so the sdmmc1 (SD card on Evaluation boards and Discovery kits) and sdmmc2 (eMMC on Evaluation boards and Wifi on Discovery kits) are respectively aliased to mmc0 and mmc1.
If you enable sdmmc3, it will take the mmc0 alias and the aliases above will shift, so don't forget to update the Linux kernel boot command accordingly!
For instance, 'root=/dev/mmcblk0p6' will become 'root=/dev/mmcblk1p6' to mount the rootfs from the sdmmc1 (SD card) when sdmmc3 is enabled.

3.2 Runtime

3.2.1 Overview

SDMMC1/2/3 instances can be allocated to:

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

or

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

Chapter #Peripheral assignment describes which peripheral instance can be assigned to which context.

3.2.2 Software frameworks

Do	Peri	Software frameworks			Comment
main Cortex-A7 secure (OP-TEE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)			
Mass storage	SDMMC		Linux MMC framework	STM32Cube SDMMC driver	

3.2.3 Peripheral configuration

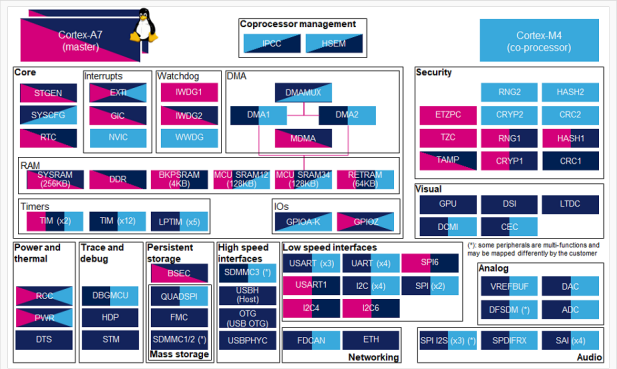
The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration can be done alone via the STM32CubeMX tool for all internal peripherals, and then manually completed (particularly for external peripherals), according to the information given in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

For Linux® kernel configuration, please refer to SDMMC device tree configuration.

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.

Do	Per	Runtime allocation			Comment
main	iph Cor era l				

Domain Instance	Peripheral	Runtime allocation				Comment
		Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)			
Mass storage	SDMMC	SDMMC1				
		SDMMC2				
		SDMMC3				Assignment (single choice)

4 How to go further

5 References

Microprocessor Unit

MultimediaCard

Direct Memory Access

SD memory card (https://www.sdcard.org)

former spelling for e•MMC ('e' in italic)

Open Portable Trusted Execution Environment

STM32MP15 resources

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

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

Return to STM32MP15 resources.

ETZPC internal peripheral

Stable: 20.05.2020 - 10:36 / Revision: 18.05.2020 - 12:05

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

Return to ETZPC internal peripheral.

Device tree

Stable: 04.02.2020 - 07:47 / Revision: 04.02.2020 - 07:34

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

Return to Device tree.

OpenSTLinux distribution

Stable: 10.10.2019 - 12:43 / Revision: 10.10.2019 - 12:42

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

Return to OpenSTLinux distribution.

MMC overview

Stable: 14.05.2020 - 09:26 / Revision: 14.05.2020 - 09:25

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

Return to MMC overview.

STM32CubeMP1 architecture

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

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

Return to STM32CubeMP1 architecture.

STM32CubeMX

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

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

Return to STM32CubeMX.

SDMMC device tree configuration

Stable: 14.05.2020 - 07:28 / Revision: 14.05.2020 - 07:27

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

Return to SDMMC device tree configuration.

STM32MPU Embedded Software architecture overview

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

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

Return to STM32MPU Embedded Software architecture overview.

How to assign an internal peripheral to a runtime context

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

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

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