SYSCFG internal peripheral

Contents

1. SYSCFG internal peripheral 3
2. How to assign an internal peripheral to a runtime context 8
3. I2C overview 8
4. STM32CubeMP1 architecture 8
5. STM32CubeMX 8
6. STM32MP15 Linux kernel overview 8
7. STM32MP15 resources 8
8. STM32MPU Embedded Software architecture overview 8
9. TF-A overview 8
10. U-Boot overview 8

Stable: 04.02.2020 - 15:40 / Revision: 04.02.2020 - 15:31

Stable: 04.02.2020 - 15:40 / Revision: 04.02.2020 - 15:51

Contents

1 Article purpose 4
2 Peripheral overview 5
2.1 Features 5
2.2 Security support 5
3 Peripheral usage and associated software 6
3.1 Boot time 6
3.2 Runtime 6
3.2.1 Overview 6
3.2.2 Software frameworks 6
3.2.3 Peripheral configuration 6
3.2.4 Peripheral assignment 6
4 References 8

1 Article purpose

The purpose of this article is to:

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

2 Peripheral overview

The **SYSCFG** peripheral is used to configure various system aspects like IOs compensation, Ethernet clocking path, …

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 really implemented in ST software.

2.2 Security support

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

3 Peripheral usage and associated software

3.1 Boot time

The SYSCFG peripheral is configured by TF-A and U-Boot at boot time.

3.2 Runtime

3.2.1 Overview

Linux and STM32Cube can directly change the SYSCFG at runtime from various drivers.

For instance, I2C fast mode plus (FM+) can be enabled for each instance in the SYSCFG so:

- Linux I2C driver uses syscon [1] to enable this mode in the SYSCFG for the instances allocated to itself
- STM32Cube I2C HAL driver uses its SYSCFG HAL driver to do the same on the instances allocated to itself

3.2.2 Software frameworks

Domain	Peripheral	Software components			Comment
OP-TEE	Linux	STM32Cube			
Core	SYSCFG		Linux syscon framework[1]	STM32Cube SYSCFG driver	

3.2.3 Peripheral configuration

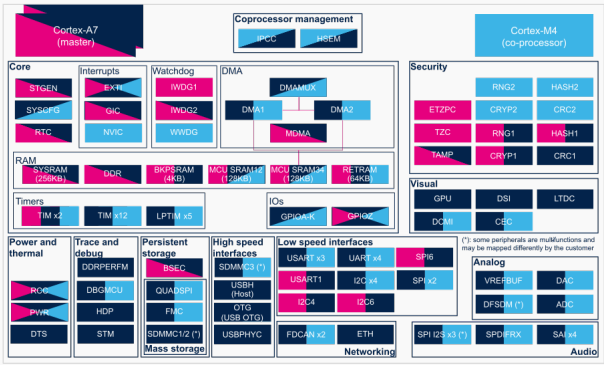
The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration can be done alone via the STM32CubeMX tool for all internal peripherals, and then manually completed (particularly for external peripherals), according to the information given in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

4 References

- 1.01.1 Documentation/devicetree/bindings/mfd/syscon.txt

System Configuration

Inter-Integrated Circuit (Bi-directional 2-wire bus standard for efficient inter-IC control.)

Open Portable Trusted Execution Environment

Linux® is a registered trademark of Linus Torvalds.

Cortex®

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: 30.03.2021 - 10:03 / Revision: 30.03.2021 - 08:24

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

Return to I2C overview.

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: 18.03.2021 - 15:06 / Revision: 18.03.2021 - 15:05

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

Return to STM32MP15 Linux kernel overview.

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.

Stable: 22.04.2021 - 11:23 / Revision: 09.04.2021 - 13:17

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

Return to TF-A overview.

Stable: 01.03.2021 - 10:58 / Revision: 01.03.2021 - 10:55

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

Return to U-Boot overview.