WWDG internal peripheral

Stable: 12.02.2019 - 09:18 / Revision: 18.01.2019 - 17:04

Contents

1 Article purpose

The purpose of this article is to:

- briefly introduce the WWDG 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 WWDG peripheral.

2 Peripheral overview

The **WWDG** peripheral is a watchdog unit that can be used to protect the Cortex®-M4 based coprocessor firmware from endless loops or to monitor some real-time activities. This peripheral is clocked by the bus on which it is connected, thus it is frozen as soon as the system goes to Stop or Standby low power mode (except if the Stop emulation mode is enabled via DBGSTOP bit in DBGMCU_CR register). This block has an early interrupt feature that allows to get an interrupt (on GIC or NVIC) one cycle before reaching the final reset: this can allow to trigger a recovery mechanism on Cortex®-M4 or on Cortex®-A7.

On WWDG expiration, a MCU reset is generated, reseting Cortex®-M4 sub-system and the WWDG itself. This MCU reset also generates an interrupt on GIC thanks to EXTI. This allows Cortex®-A7 to detect Cortex®-M4 crashed and to recover it by stopping associated services, reloading Cortex®-M4 firmware and restarting Cortex®-M4.

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.

2.2 Security support

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

3 Peripheral usage and associated software

3.1 Boot time

The WWDG is not used at boot time.

3.2 Runtime

3.2.1 Overview

WWDG can be allocated to the Cortex®-M4 for using in STM32Cube with STM32Cube WWDG driver.
As there is only one WWDG counter cycle between the WWDG early interrupt and the WWDG reset generation, ST preconizes to allocate the WWDG early interrupt to Cortex®-M4 for a better reactivity and not to Cortex®-A7.

3.2.2 Software frameworks

Domain	Peripheral	Software frameworks			Comment
		Cortex-A7 secure (OP-TEE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)	
Core/Watchdog	WWDG			STM32Cube WWDG driver	

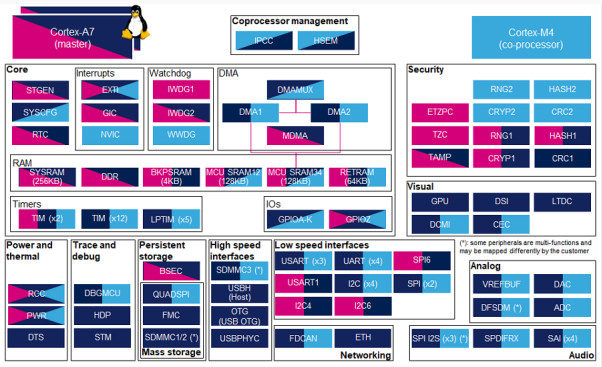
3.2.3 Peripheral configuration

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.



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	Peripheral	Runtime allocation				Comment
		Instance	Cortex-A7 secure (OP-TEE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)	
Core/Watchdog	WWDG	WWDG			☐	

4 References