IWDG internal peripheral

Contents

Stable: 25.09.2020 - 09:42 / Revision: 25.09.2020 - 09:37

Stable: 25.09.2020 - 09:42 | Revision: 25.09.2020 - 09:37

A quality version of this page, approved on *25 September 2020*, was based off this revision.

Contents

1 Peripheral overview

The **IWDG** peripheral is a watchdog unit that can be used to protect application frameworks running on Cortex-A7 from endless loops. This peripheral supports an **independent** clocking source in order to be able to continue running even when the rest of the system is in low power mode (STOP, STANDBY). Another important feature of this block is the **early interrupt** feature that allows to trigger an interrupt at a given power supply threshold before reaching the final reset: this gives the opportunity to run a recovery mechanism that will try to revive the system with minimum impact.

1.1 Features

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

1.2 Security support

IWDG1 is **secure-aware** (under ETZPC control).
IWDG2 is **non-secure**.

2 Peripheral usage and associated software

2.1 Boot time

Pay attention to the fact that IWDG can be configured to be **automatically active** at startup (without any software intervention) via BSEC. When this is the case, the watchdog is anyway frozen during ROM code execution but it will start to decrement its counter as soon as the ROM code is left so it is important to reload the watchdog from the boot chain in this case. This behavior is implemented for **IWDG2 only** in STMicroelectronics distribution via the trusted boot chain only.
Notice also that BSEC features some freeze bits that allow to **freeze IWDG** during platform STOP and STANDBY low power periods, avoiding to have to wake up (via RTC) for the only purpose of reloading the watchdog.

2.2 Runtime

2.2.1 Overview

IWDG1 can be allocated to the Cortex-A7 secure to be used in the secure context by the customer application: this instance is not supported in STMicrolectronics distribution.
IWDG2 can be allocated to the Cortex-A7 non-secure to be used with Linux watchdog framework. In this configuration, the secure monitor (from OP-TEE -if present- or TF-A) is able to receive IWDG early interrupts that can be used in a tentative to reset the Cortex-A7 without interfering with Cortex-M4 execution.

2.2.2 Software frameworks

Domain	Peripheral	Software components		Comment
OP-TEE	Linux	STM32Cube		
Core /Watchdog	IWDG	TF-A	Linux watchdog framework	

2.2.3 Peripheral configuration

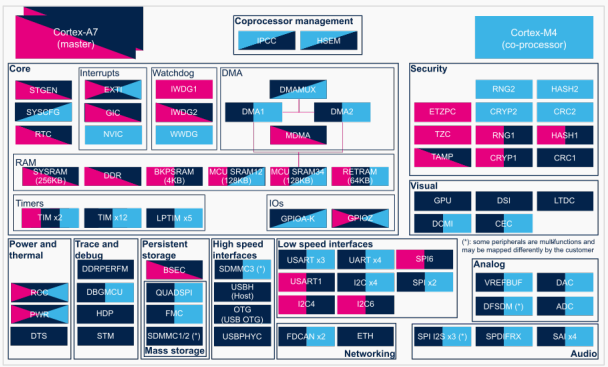
The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration can be done alone via the STM32CubeMX tool for all internal peripherals, and then manually completed (particularly for external peripherals), according to the information given in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

2.2.4 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 /Watchdog	IWDG	IWDG1				
		IWDG2				Shared (none or both): • Cortex-A7 non secure for reload • Cortex-A7 secure for early interrupt handling

Independent Watchdog

Cortex®

Read Only Memory

Linux® is a registered trademark of Linus Torvalds.

Open Portable Trusted Execution Environment

Stable: 24.09.2019 - 14:06 / Revision: 24.09.2019 - 07:57

Contents

1 Peripheral overview

The **IWDG** peripheral is a watchdog unit that can be used to protect application frameworks running on Cortex-A7 from endless loops. This peripheral supports an **independent** clocking source in order to be able to continue running even when the rest of the system is in low power mode (STOP, STANDBY). Another important feature of this block is the **early interrupt** feature that allows to trigger an interrupt at a given power supply threshold before reaching the final reset: this gives the opportunity to run a recovery mechanism that will try to revive the system with minimum impact.

1.1 Features

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

1.2 Security support

IWDG1 is **secure-aware** (under ETZPC control).
IWDG2 is **non-secure**.

2 Peripheral usage and associated software

2.1 Boot time

Pay attention to the fact that IWDG can be configured to be **automatically active** at startup (without any software intervention) via BSEC. When this is the case, the watchdog is anyway frozen during ROM code execution but it will start to decrement its counter as soon as the ROM code is left so it is important to reload the watchdog from the boot chain in this case. This behavior is implemented for **IWDG2 only** in STMicroelectronics distribution via the trusted boot chain only.
Notice also that BSEC features some freeze bits that allow to **freeze IWDG** during platform STOP and STANDBY low power periods, avoiding to have to wake up (via RTC) for the only purpose of reloading the watchdog.

2.2 Runtime

2.2.1 Overview

IWDG1 can be allocated to the Cortex-A7 secure to be used in the secure context by the customer application: this instance is not supported in STMicrolectronics distribution.
IWDG2 can be allocated to the Cortex-A7 non-secure to be used with Linux watchdog framework. In this configuration, the secure monitor (from OP-TEE -if present- or TF-A) is able to receive IWDG early interrupts that can be used in a tentative to reset the Cortex-A7 without interfering with Cortex-M4 execution.

2.2.2 Software frameworks

Domain	Peripheral	Software components		Comment
OP-TEE	Linux	STM32Cube		
Core /Watchdog	IWDG	TF-A	Linux watchdog framework	

2.2.3 Peripheral configuration

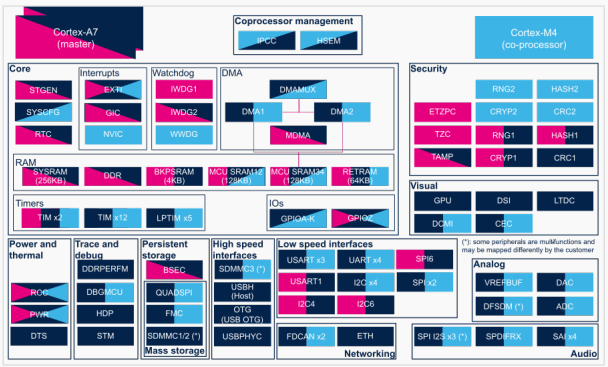
The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration can be done alone via the STM32CubeMX tool for all internal peripherals, and then manually completed (particularly for external peripherals), according to the information given in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

2.2.4 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 /Watchdog	IWDG	IWDG1				
		IWDG2				Shared (none or both): • Cortex-A7 non secure for reload • Cortex-A7 secure for early interrupt handling

Independent Watchdog

Cortex®

Read Only Memory

Linux® is a registered trademark of Linus Torvalds.

Open Portable Trusted Execution Environment

Stable: 12.03.2021 - 11:29 / Revision: 12.03.2021 - 11:15

Contents

1 Peripheral overview

The **IWDG** peripheral is a watchdog unit that can be used to protect application frameworks running on Cortex-A7 from endless loops. This peripheral supports an **independent** clocking source in order to be able to continue running even when the rest of the system is in low power mode (STOP, STANDBY). Another important feature of this block is the **early interrupt** feature that allows to trigger an interrupt at a given power supply threshold before reaching the final reset: this gives the opportunity to run a recovery mechanism that will try to revive the system with minimum impact.

1.1 Features

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

1.2 Security support

IWDG1 is **secure-aware** (under ETZPC control).
IWDG2 is **non-secure**.

2 Peripheral usage and associated software

2.1 Boot time

Pay attention to the fact that IWDG can be configured to be **automatically active** at startup (without any software intervention) via BSEC. When this is the case, the watchdog is anyway frozen during ROM code execution but it will start to decrement its counter as soon as the ROM code is left so it is important to reload the watchdog from the boot chain in this case. This behavior is implemented for **IWDG2 only** in STMicroelectronics distribution via the trusted boot chain only.
Notice also that BSEC features some freeze bits that allow to **freeze IWDG** during platform STOP and STANDBY low power periods, avoiding to have to wake up (via RTC) for the only purpose of reloading the watchdog.

2.2 Runtime

2.2.1 Overview

IWDG1 can be allocated to the Cortex-A7 secure to be used in the secure context by the customer application: this instance is not supported in STMicrolectronics distribution.
IWDG2 can be allocated to the Cortex-A7 non-secure to be used with Linux watchdog framework. In this configuration, the secure monitor (from OP-TEE -if present- or TF-A) is able to receive IWDG early interrupts that can be used in a tentative to reset the Cortex-A7 without interfering with Cortex-M4 execution.

2.2.2 Software frameworks

Domain	Peripheral	Software components		Comment
OP-TEE	Linux	STM32Cube		
Core /Watchdog	IWDG	TF-A	Linux watchdog framework	

2.2.3 Peripheral configuration

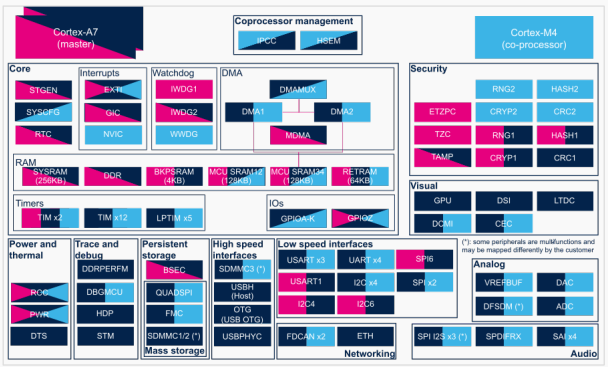
The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration can be done alone via the STM32CubeMX tool for all internal peripherals, and then manually completed (particularly for external peripherals), according to the information given in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

2.2.4 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 /Watchdog	IWDG	IWDG1				
		IWDG2				Shared (none or both): • Cortex-A7 non secure for reload • Cortex-A7 secure for early interrupt handling

Independent Watchdog

Cortex®

Read Only Memory

Linux® is a registered trademark of Linus Torvalds.

Open Portable Trusted Execution Environment

Stable: 31.07.2020 - 14:57 / Revision: 31.07.2020 - 14:56

Contents

1 Peripheral overview

The **IWDG** peripheral is a watchdog unit that can be used to protect application frameworks running on Cortex-A7 from endless loops. This peripheral supports an **independent** clocking source in order to be able to continue running even when the rest of the system is in low power mode (STOP, STANDBY). Another important feature of this block is the **early interrupt** feature that allows to trigger an interrupt at a given power supply threshold before reaching the final reset: this gives the opportunity to run a recovery mechanism that will try to revive the system with minimum impact.

1.1 Features

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

1.2 Security support

IWDG1 is **secure-aware** (under ETZPC control).
IWDG2 is **non-secure**.

2 Peripheral usage and associated software

2.1 Boot time

Pay attention to the fact that IWDG can be configured to be **automatically active** at startup (without any software intervention) via BSEC. When this is the case, the watchdog is anyway frozen during ROM code execution but it will start to decrement its counter as soon as the ROM code is left so it is important to reload the watchdog from the boot chain in this case. This behavior is implemented for **IWDG2 only** in STMicroelectronics distribution via the trusted boot chain only.
Notice also that BSEC features some freeze bits that allow to **freeze IWDG** during platform STOP and STANDBY low power periods, avoiding to have to wake up (via RTC) for the only purpose of reloading the watchdog.

2.2 Runtime

2.2.1 Overview

IWDG1 can be allocated to the Cortex-A7 secure to be used in the secure context by the customer application: this instance is not supported in STMicrolectronics distribution.
IWDG2 can be allocated to the Cortex-A7 non-secure to be used with Linux watchdog framework. In this configuration, the secure monitor (from OP-TEE -if present- or TF-A) is able to receive IWDG early interrupts that can be used in a tentative to reset the Cortex-A7 without interfering with Cortex-M4 execution.

2.2.2 Software frameworks

Domain	Peripheral	Software components		Comment
OP-TEE	Linux	STM32Cube		
Core /Watchdog	IWDG	TF-A	Linux watchdog framework	

2.2.3 Peripheral configuration

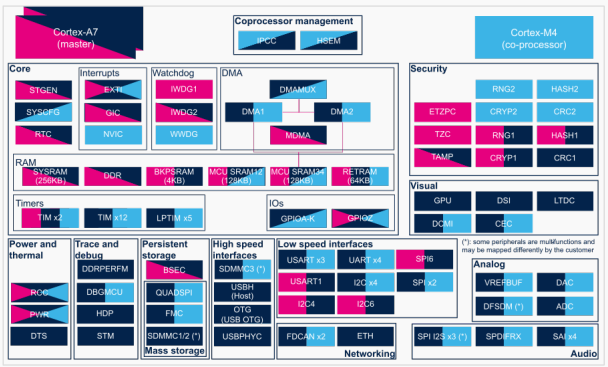
The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration can be done alone via the STM32CubeMX tool for all internal peripherals, and then manually completed (particularly for external peripherals), according to the information given in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

2.2.4 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 /Watchdog	IWDG	IWDG1				
		IWDG2				Shared (none or both): • Cortex-A7 non secure for reload • Cortex-A7 secure for early interrupt handling

Independent Watchdog

Cortex®

Read Only Memory

Linux® is a registered trademark of Linus Torvalds.

Open Portable Trusted Execution Environment

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

Contents

1 Peripheral overview

The **IWDG** peripheral is a watchdog unit that can be used to protect application frameworks running on Cortex-A7 from endless loops. This peripheral supports an **independent** clocking source in order to be able to continue running even when the rest of the system is in low power mode (STOP, STANDBY). Another important feature of this block is the **early interrupt** feature that allows to trigger an interrupt at a given power supply threshold before reaching the final reset: this gives the opportunity to run a recovery mechanism that will try to revive the system with minimum impact.

1.1 Features

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

1.2 Security support

IWDG1 is **secure-aware** (under ETZPC control).
IWDG2 is **non-secure**.

2 Peripheral usage and associated software

2.1 Boot time

Pay attention to the fact that IWDG can be configured to be **automatically active** at startup (without any software intervention) via BSEC. When this is the case, the watchdog is anyway frozen during ROM code execution but it will start to decrement its counter as soon as the ROM code is left so it is important to reload the watchdog from the boot chain in this case. This behavior is implemented for **IWDG2 only** in STMicroelectronics distribution via the trusted boot chain only.
Notice also that BSEC features some freeze bits that allow to **freeze IWDG** during platform STOP and STANDBY low power periods, avoiding to have to wake up (via RTC) for the only purpose of reloading the watchdog.

2.2 Runtime

2.2.1 Overview

IWDG1 can be allocated to the Cortex-A7 secure to be used in the secure context by the customer application: this instance is not supported in STMicrolectronics distribution.
IWDG2 can be allocated to the Cortex-A7 non-secure to be used with Linux watchdog framework. In this configuration, the secure monitor (from OP-TEE -if present- or TF-A) is able to receive IWDG early interrupts that can be used in a tentative to reset the Cortex-A7 without interfering with Cortex-M4 execution.

2.2.2 Software frameworks

Domain	Peripheral	Software components			Comment
OP-TEE	Linux	STM32Cube			
Core /Watchdog	IWDG	TF-A	Linux watchdog framework		

2.2.3 Peripheral configuration

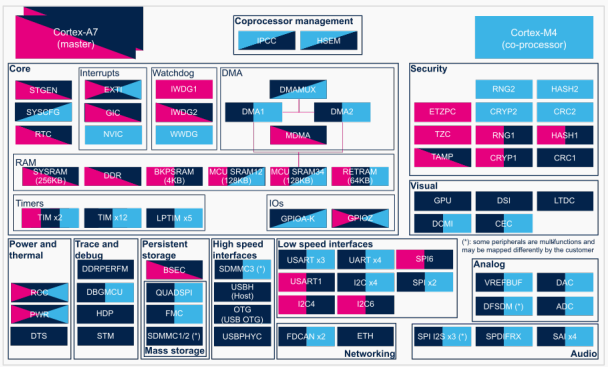
The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration can be done alone via the STM32CubeMX tool for all internal peripherals, and then manually completed (particularly for external peripherals), according to the information given in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

2.2.4 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 /Watchdog	IWDG	IWDG1				
		IWDG2				Shared (none or both): • Cortex-A7 non secure for reload • Cortex-A7 secure for early interrupt handling

Independent Watchdog

Cortex®

Read Only Memory

Linux® is a registered trademark of Linus Torvalds.

Open Portable Trusted Execution Environment

Stable: 13.05.2020 - 08:56 / Revision: 13.05.2020 - 08:54

Contents

1 Peripheral overview

The **IWDG** peripheral is a watchdog unit that can be used to protect application frameworks running on Cortex-A7 from endless loops. This peripheral supports an **independent** clocking source in order to be able to continue running even when the rest of the system is in low power mode (STOP, STANDBY). Another important feature of this block is the **early interrupt** feature that allows to trigger an interrupt at a given power supply threshold before reaching the final reset: this gives the opportunity to run a recovery mechanism that will try to revive the system with minimum impact.

1.1 Features

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

1.2 Security support

IWDG1 is **secure-aware** (under ETZPC control).
IWDG2 is **non-secure**.

2 Peripheral usage and associated software

2.1 Boot time

Pay attention to the fact that IWDG can be configured to be **automatically active** at startup (without any software intervention) via BSEC. When this is the case, the watchdog is anyway frozen during ROM code execution but it will start to decrement its counter as soon as the ROM code is left so it is important to reload the watchdog from the boot chain in this case. This behavior is implemented for **IWDG2 only** in STMicroelectronics distribution via the trusted boot chain only.
Notice also that BSEC features some freeze bits that allow to **freeze IWDG** during platform STOP and STANDBY low power periods, avoiding to have to wake up (via RTC) for the only purpose of reloading the watchdog.

2.2 Runtime

2.2.1 Overview

IWDG1 can be allocated to the Cortex-A7 secure to be used in the secure context by the customer application: this instance is not supported in STMicrolectronics distribution.
IWDG2 can be allocated to the Cortex-A7 non-secure to be used with Linux watchdog framework. In this configuration, the secure monitor (from OP-TEE -if present- or TF-A) is able to receive IWDG early interrupts that can be used in a tentative to reset the Cortex-A7 without interfering with Cortex-M4 execution.

2.2.2 Software frameworks

Domain	Peripheral	Software components			Comment
OP-TEE	Linux	STM32Cube			
Core /Watchdog	IWDG	TF-A	Linux watchdog framework		

2.2.3 Peripheral configuration

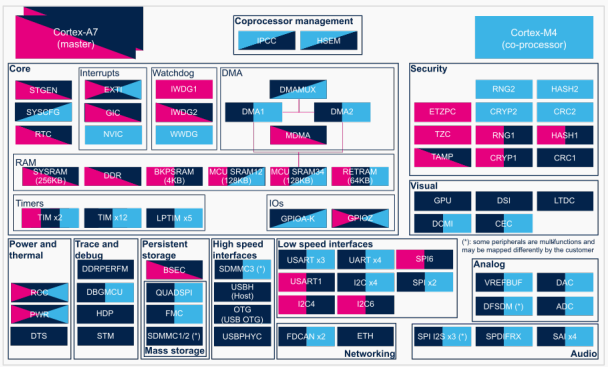
The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration can be done alone via the STM32CubeMX tool for all internal peripherals, and then manually completed (particularly for external peripherals), according to the information given in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

2.2.4 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 /Watchdog	IWDG	IWDG1				
		IWDG2				Shared (none or both): • Cortex-A7 non secure for reload • Cortex-A7 secure for early interrupt handling

Independent Watchdog

Cortex®

Read Only Memory

Linux® is a registered trademark of Linus Torvalds.

Open Portable Trusted Execution Environment

Stable: 01.12.2020 - 10:35 / Revision: 01.12.2020 - 09:20

Contents

1 Peripheral overview

The **IWDG** peripheral is a watchdog unit that can be used to protect application frameworks running on Cortex-A7 from endless loops. This peripheral supports an **independent** clocking source in order to be able to continue running even when the rest of the system is in low power mode (STOP, STANDBY). Another important feature of this block is the **early interrupt** feature that allows to trigger an interrupt at a given power supply threshold before reaching the final reset: this gives the opportunity to run a recovery mechanism that will try to revive the system with minimum impact.

1.1 Features

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

1.2 Security support

IWDG1 is **secure-aware** (under ETZPC control).
IWDG2 is **non-secure**.

2 Peripheral usage and associated software

2.1 Boot time

Pay attention to the fact that IWDG can be configured to be **automatically active** at startup (without any software intervention) via BSEC. When this is the case, the watchdog is anyway frozen during ROM code execution but it will start to decrement its counter as soon as the ROM code is left so it is important to reload the watchdog from the boot chain in this case. This behavior is implemented for **IWDG2 only** in STMicroelectronics distribution via the trusted boot chain only.
Notice also that BSEC features some freeze bits that allow to **freeze IWDG** during platform STOP and STANDBY low power periods, avoiding to have to wake up (via RTC) for the only purpose of reloading the watchdog.

2.2 Runtime

2.2.1 Overview

IWDG1 can be allocated to the Cortex-A7 secure to be used in the secure context by the customer application: this instance is not supported in STMicrolectronics distribution.
IWDG2 can be allocated to the Cortex-A7 non-secure to be used with Linux watchdog framework. In this configuration, the secure monitor (from OP-TEE -if present- or TF-A) is able to receive IWDG early interrupts that can be used in a tentative to reset the Cortex-A7 without interfering with Cortex-M4 execution.

2.2.2 Software frameworks

Domain	Peripheral	Software components			Comment
OP-TEE	Linux	STM32Cube			
Core /Watchdog	IWDG	TF-A	Linux watchdog framework		

2.2.3 Peripheral configuration

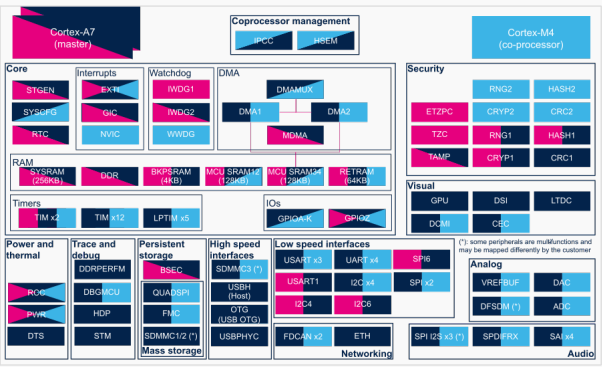
The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration can be done alone via the STM32CubeMX tool for all internal peripherals, and then manually completed (particularly for external peripherals), according to the information given in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

2.2.4 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 /Watchdog	IWDG	IWDG1				
		IWDG2				Shared (none or both): • Cortex-A7 non secure for reload • Cortex-A7 secure for early interrupt handling

Independent Watchdog

Cortex®

Read Only Memory

Linux® is a registered trademark of Linus Torvalds.

Open Portable Trusted Execution Environment

Stable: 19.02.2020 - 10:01 / Revision: 04.02.2020 - 15:23

Contents

1 Peripheral overview

The **IWDG** peripheral is a watchdog unit that can be used to protect application frameworks running on Cortex-A7 from endless loops. This peripheral supports an **independent** clocking source in order to be able to continue running even when the rest of the system is in low power mode (STOP, STANDBY). Another important feature of this block is the **early interrupt** feature that allows to trigger an interrupt at a given power supply threshold before reaching the final reset: this gives the opportunity to run a recovery mechanism that will try to revive the system with minimum impact.

1.1 Features

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

1.2 Security support

IWDG1 is **secure-aware** (under ETZPC control).
IWDG2 is **non-secure**.

2 Peripheral usage and associated software

2.1 Boot time

Pay attention to the fact that IWDG can be configured to be **automatically active** at startup (without any software intervention) via BSEC. When this is the case, the watchdog is anyway frozen during ROM code execution but it will start to decrement its counter as soon as the ROM code is left so it is important to reload the watchdog from the boot chain in this case. This behavior is implemented for **IWDG2 only** in STMicroelectronics distribution via the trusted boot chain only.
Notice also that BSEC features some freeze bits that allow to **freeze IWDG** during platform STOP and STANDBY low power periods, avoiding to have to wake up (via RTC) for the only purpose of reloading the watchdog.

2.2 Runtime

2.2.1 Overview

IWDG1 can be allocated to the Cortex-A7 secure to be used in the secure context by the customer application: this instance is not supported in STMicrolectronics distribution.
IWDG2 can be allocated to the Cortex-A7 non-secure to be used with Linux watchdog framework. In this configuration, the secure monitor (from OP-TEE -if present- or TF-A) is able to receive IWDG early interrupts that can be used in a tentative to reset the Cortex-A7 without interfering with Cortex-M4 execution.

2.2.2 Software frameworks

Domain	Peripheral	Software components		Comment
OP-TEE	Linux	STM32Cube		
Core /Watchdog	IWDG	TF-A	Linux watchdog framework	

2.2.3 Peripheral configuration

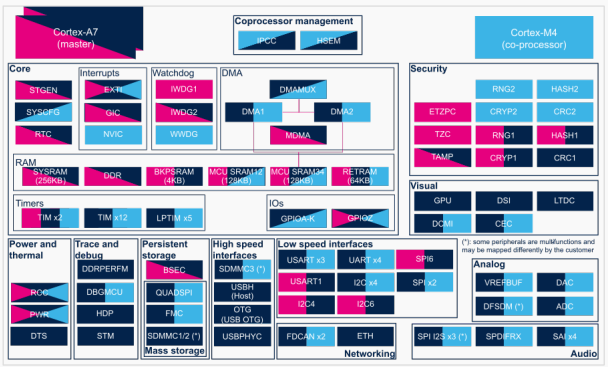
The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration can be done alone via the STM32CubeMX tool for all internal peripherals, and then manually completed (particularly for external peripherals), according to the information given in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

2.2.4 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 /Watchdog	IWDG	IWDG1				
		IWDG2				Shared (none or both): • Cortex-A7 non secure for reload • Cortex-A7 secure for early interrupt handling

Independent Watchdog

Cortex®

Read Only Memory

Linux® is a registered trademark of Linus Torvalds.

Open Portable Trusted Execution Environment

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

Contents

1 Peripheral overview

The **IWDG** peripheral is a watchdog unit that can be used to protect application frameworks running on Cortex-A7 from endless loops. This peripheral supports an **independent** clocking source in order to be able to continue running even when the rest of the system is in low power mode (STOP, STANDBY). Another important feature of this block is the **early interrupt** feature that allows to trigger an interrupt at a given power supply threshold before reaching the final reset: this gives the opportunity to run a recovery mechanism that will try to revive the system with minimum impact.

1.1 Features

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

1.2 Security support

IWDG1 is **secure-aware** (under ETZPC control).
IWDG2 is **non-secure**.

2 Peripheral usage and associated software

2.1 Boot time

Pay attention to the fact that IWDG can be configured to be **automatically active** at startup (without any software intervention) via BSEC. When this is the case, the watchdog is anyway frozen during ROM code execution but it will start to decrement its counter as soon as the ROM code is left so it is important to reload the watchdog from the boot chain in this case. This behavior is implemented for **IWDG2 only** in STMicroelectronics distribution via the trusted boot chain only.
Notice also that BSEC features some freeze bits that allow to **freeze IWDG** during platform STOP and STANDBY low power periods, avoiding to have to wake up (via RTC) for the only purpose of reloading the watchdog.

2.2 Runtime

2.2.1 Overview

IWDG1 can be allocated to the Cortex-A7 secure to be used in the secure context by the customer application: this instance is not supported in STMicrolectronics distribution.
IWDG2 can be allocated to the Cortex-A7 non-secure to be used with Linux watchdog framework. In this configuration, the secure monitor (from OP-TEE -if present- or TF-A) is able to receive IWDG early interrupts that can be used in a tentative to reset the Cortex-A7 without interfering with Cortex-M4 execution.

2.2.2 Software frameworks

Domain	Peripheral	Software components		Comment
OP-TEE	Linux	STM32Cube		
Core /Watchdog	IWDG	TF-A	Linux watchdog framework	

2.2.3 Peripheral configuration

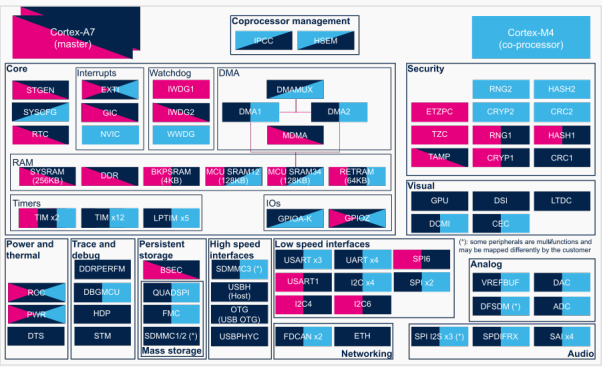
The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration can be done alone via the STM32CubeMX tool for all internal peripherals, and then manually completed (particularly for external peripherals), according to the information given in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

2.2.4 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 /Watchdog	IWDG	IWDG1				
		IWDG2				Shared (none or both): • Cortex-A7 non secure for reload • Cortex-A7 secure for early interrupt handling

Independent Watchdog

Cortex®

Read Only Memory

Linux® is a registered trademark of Linus Torvalds.

Open Portable Trusted Execution Environment

Stable: 19.10.2021 - 14:25 / Revision: 19.10.2021 - 14:25

Contents

1 Peripheral overview

The **IWDG** peripheral is a watchdog unit that can be used to protect application frameworks running on Cortex-A7 from endless loops. This peripheral supports an **independent** clocking source in order to be able to continue running even when the rest of the system is in low power mode (STOP, STANDBY). Another important feature of this block is the **early interrupt** feature that allows to trigger an interrupt at a given power supply threshold before reaching the final reset: this gives the opportunity to run a recovery mechanism that will try to revive the system with minimum impact.

1.1 Features

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

1.2 Security support

IWDG1 is **secure-aware** (under ETZPC control).
IWDG2 is **non-secure**.

2 Peripheral usage and associated software

2.1 Boot time

Pay attention to the fact that IWDG can be configured to be **automatically active** at startup (without any software intervention) via BSEC. When this is the case, the watchdog is anyway frozen during ROM code execution but it will start to decrement its counter as soon as the ROM code is left so it is important to reload the watchdog from the boot chain in this case. This behavior is implemented for **IWDG2 only** in STMicroelectronics distribution via the trusted boot chain only.
Notice also that BSEC features some freeze bits that allow to **freeze IWDG** during platform STOP and STANDBY low power periods, avoiding to have to wake up (via RTC) for the only purpose of reloading the watchdog.

2.2 Runtime

2.2.1 Overview

IWDG1 can be allocated to the Cortex-A7 secure to be used in the secure context by the customer application: this instance is not supported in STMicrolectronics distribution.
IWDG2 can be allocated to the Cortex-A7 non-secure to be used with Linux watchdog framework. In this configuration, the secure monitor (from OP-TEE -if present- or TF-A) is able to receive IWDG early interrupts that can be used in a tentative to reset the Cortex-A7 without interfering with Cortex-M4 execution.

2.2.2 Software frameworks

Domain	Peripheral	Software components			Comment
OP-TEE	Linux	STM32Cube			
Core /Watchdog	IWDG	TF-A	Linux watchdog framework		

2.2.3 Peripheral configuration

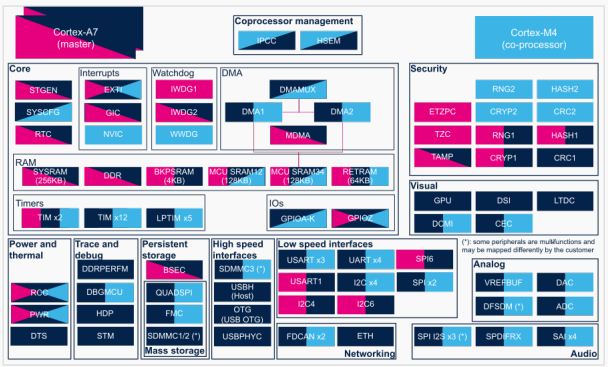
The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration can be done alone via the STM32CubeMX tool for all internal peripherals, and then manually completed (particularly for external peripherals), according to the information given in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

2.2.4 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 /Watchdog	IWDG	IWDG1				
		IWDG2				Shared (none or both): • Cortex-A7 non secure for reload • Cortex-A7 secure for early interrupt handling

Independent Watchdog

Cortex®

Read Only Memory

Linux® is a registered trademark of Linus Torvalds.

Open Portable Trusted Execution Environment

Stable: 20.07.2021 - 09:02 / Revision: 11.03.2021 - 08:07

Contents

1 Peripheral overview

The **IWDG** peripheral is a watchdog unit that can be used to protect application frameworks running on Cortex-A7 from endless loops. This peripheral supports an **independent** clocking source in order to be able to continue running even when the rest of the system is in low power mode (STOP, STANDBY). Another important feature of this block is the **early interrupt** feature that allows to trigger an interrupt at a given power supply threshold before reaching the final reset: this gives the opportunity to run a recovery mechanism that will try to revive the system with minimum impact.

1.1 Features

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

1.2 Security support

IWDG1 is **secure-aware** (under ETZPC control).
IWDG2 is **non-secure**.

2 Peripheral usage and associated software

2.1 Boot time

Pay attention to the fact that IWDG can be configured to be **automatically active** at startup (without any software intervention) via BSEC. When this is the case, the watchdog is anyway frozen during ROM code execution but it will start to decrement its counter as soon as the ROM code is left so it is important to reload the watchdog from the boot chain in this case. This behavior is implemented for **IWDG2 only** in STMicroelectronics distribution via the trusted boot chain only.
Notice also that BSEC features some freeze bits that allow to **freeze IWDG** during platform STOP and STANDBY low power periods, avoiding to have to wake up (via RTC) for the only purpose of reloading the watchdog.

2.2 Runtime

2.2.1 Overview

IWDG1 can be allocated to the Cortex-A7 secure to be used in the secure context by the customer application: this instance is not supported in STMicrolectronics distribution.
IWDG2 can be allocated to the Cortex-A7 non-secure to be used with Linux watchdog framework. In this configuration, the secure monitor (from OP-TEE -if present- or TF-A) is able to receive IWDG early interrupts that can be used in a tentative to reset the Cortex-A7 without interfering with Cortex-M4 execution.

2.2.2 Software frameworks

Domain	Peripheral	Software components		Comment
OP-TEE	Linux	STM32Cube		
Core /Watchdog	IWDG	TF-A	Linux watchdog framework	

2.2.3 Peripheral configuration

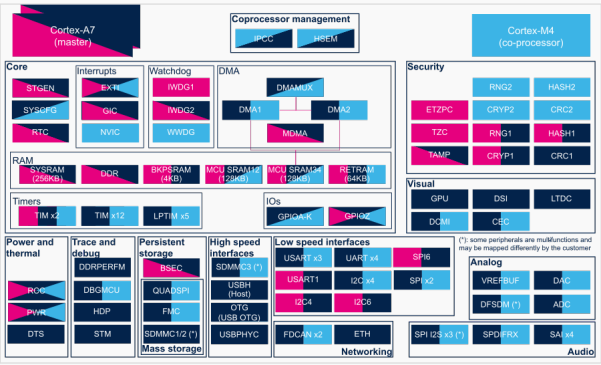
The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration can be done alone via the STM32CubeMX tool for all internal peripherals, and then manually completed (particularly for external peripherals), according to the information given in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

2.2.4 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 /Watchdog	IWDG	IWDG1				
		IWDG2				Shared (none or both): • Cortex-A7 non secure for reload • Cortex-A7 secure for early interrupt handling

Independent Watchdog

Cortex®

Read Only Memory

Linux® is a registered trademark of Linus Torvalds.

Open Portable Trusted Execution Environment

Stable: 26.03.2021 - 11:32 / Revision: 12.03.2021 - 11:07

Contents

1 Peripheral overview

The **IWDG** peripheral is a watchdog unit that can be used to protect application frameworks running on Cortex-A7 from endless loops. This peripheral supports an **independent** clocking source in order to be able to continue running even when the rest of the system is in low power mode (STOP, STANDBY). Another important feature of this block is the **early interrupt** feature that allows to trigger an interrupt at a given power supply threshold before reaching the final reset: this gives the opportunity to run a recovery mechanism that will try to revive the system with minimum impact.

1.1 Features

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

1.2 Security support

IWDG1 is **secure-aware** (under ETZPC control).
IWDG2 is **non-secure**.

2 Peripheral usage and associated software

2.1 Boot time

Pay attention to the fact that IWDG can be configured to be **automatically active** at startup (without any software intervention) via BSEC. When this is the case, the watchdog is anyway frozen during ROM code execution but it will start to decrement its counter as soon as the ROM code is left so it is important to reload the watchdog from the boot chain in this case. This behavior is implemented for **IWDG2 only** in STMicroelectronics distribution via the trusted boot chain only.
Notice also that BSEC features some freeze bits that allow to **freeze IWDG** during platform STOP and STANDBY low power periods, avoiding to have to wake up (via RTC) for the only purpose of reloading the watchdog.

2.2 Runtime

2.2.1 Overview

IWDG1 can be allocated to the Cortex-A7 secure to be used in the secure context by the customer application: this instance is not supported in STMicrolectronics distribution.
IWDG2 can be allocated to the Cortex-A7 non-secure to be used with Linux watchdog framework. In this configuration, the secure monitor (from OP-TEE -if present- or TF-A) is able to receive IWDG early interrupts that can be used in a tentative to reset the Cortex-A7 without interfering with Cortex-M4 execution.

2.2.2 Software frameworks

Domain	Peripheral	Software components		Comment
OP-TEE	Linux	STM32Cube		
Core /Watchdog	IWDG	TF-A	Linux watchdog framework	

2.2.3 Peripheral configuration

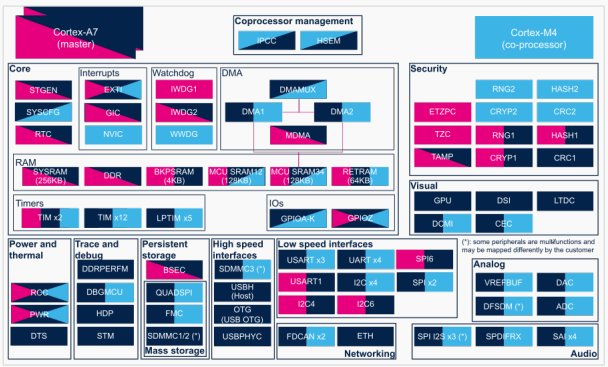
The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration can be done alone via the STM32CubeMX tool for all internal peripherals, and then manually completed (particularly for external peripherals), according to the information given in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

2.2.4 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 /Watchdog	IWDG	IWDG1				
		IWDG2				Shared (none or both): • Cortex-A7 non secure for reload • Cortex-A7 secure for early interrupt handling

Independent Watchdog

Cortex®

Read Only Memory

Linux® is a registered trademark of Linus Torvalds.

Open Portable Trusted Execution Environment

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

Contents

1 Peripheral overview

The **IWDG** peripheral is a watchdog unit that can be used to protect application frameworks running on Cortex-A7 from endless loops. This peripheral supports an **independent** clocking source in order to be able to continue running even when the rest of the system is in low power mode (STOP, STANDBY). Another important feature of this block is the **early interrupt** feature that allows to trigger an interrupt at a given power supply threshold before reaching the final reset: this gives the opportunity to run a recovery mechanism that will try to revive the system with minimum impact.

1.1 Features

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

1.2 Security support

IWDG1 is **secure-aware** (under ETZPC control).
IWDG2 is **non-secure**.

2 Peripheral usage and associated software

2.1 Boot time

Pay attention to the fact that IWDG can be configured to be **automatically active** at startup (without any software intervention) via BSEC. When this is the case, the watchdog is anyway frozen during ROM code execution but it will start to decrement its counter as soon as the ROM code is left so it is important to reload the watchdog from the boot chain in this case. This behavior is implemented for **IWDG2 only** in STMicroelectronics distribution via the trusted boot chain only.
Notice also that BSEC features some freeze bits that allow to **freeze IWDG** during platform STOP and STANDBY low power periods, avoiding to have to wake up (via RTC) for the only purpose of reloading the watchdog.

2.2 Runtime

2.2.1 Overview

IWDG1 can be allocated to the Cortex-A7 secure to be used in the secure context by the customer application: this instance is not supported in STMicrolectronics distribution.
IWDG2 can be allocated to the Cortex-A7 non-secure to be used with Linux watchdog framework. In this configuration, the secure monitor (from OP-TEE -if present- or TF-A) is able to receive IWDG early interrupts that can be used in a tentative to reset the Cortex-A7 without interfering with Cortex-M4 execution.

2.2.2 Software frameworks

Domain	Peripheral	Software components			Comment
OP-TEE	Linux	STM32Cube			
Core /Watchdog	IWDG	TF-A	Linux watchdog framework		

2.2.3 Peripheral configuration

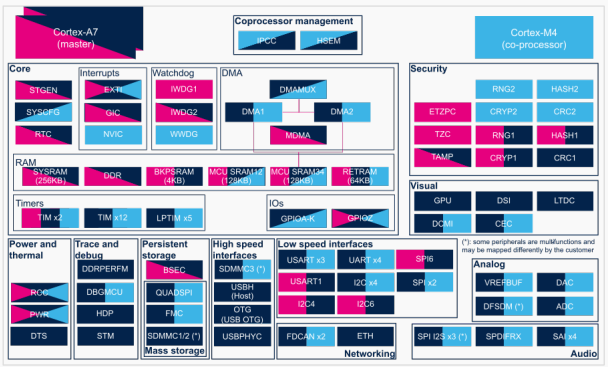
The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration can be done alone via the STM32CubeMX tool for all internal peripherals, and then manually completed (particularly for external peripherals), according to the information given in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

2.2.4 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 /Watchdog	IWDG	IWDG1				
		IWDG2				Shared (none or both): • Cortex-A7 non secure for reload • Cortex-A7 secure for early interrupt handling

Independent Watchdog

Cortex®

Read Only Memory

Linux® is a registered trademark of Linus Torvalds.

Open Portable Trusted Execution Environment

Stable: 31.01.2020 - 13:22 / Revision: 31.01.2020 - 13:13

Contents

1 Peripheral overview

The **IWDG** peripheral is a watchdog unit that can be used to protect application frameworks running on Cortex-A7 from endless loops. This peripheral supports an **independent** clocking source in order to be able to continue running even when the rest of the system is in low power mode (STOP, STANDBY). Another important feature of this block is the **early interrupt** feature that allows to trigger an interrupt at a given power supply threshold before reaching the final reset: this gives the opportunity to run a recovery mechanism that will try to revive the system with minimum impact.

1.1 Features

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

1.2 Security support

IWDG1 is **secure-aware** (under ETZPC control).
IWDG2 is **non-secure**.

2 Peripheral usage and associated software

2.1 Boot time

Pay attention to the fact that IWDG can be configured to be **automatically active** at startup (without any software intervention) via BSEC. When this is the case, the watchdog is anyway frozen during ROM code execution but it will start to decrement its counter as soon as the ROM code is left so it is important to reload the watchdog from the boot chain in this case. This behavior is implemented for **IWDG2 only** in STMicroelectronics distribution via the trusted boot chain only.
Notice also that BSEC features some freeze bits that allow to **freeze IWDG** during platform STOP and STANDBY low power periods, avoiding to have to wake up (via RTC) for the only purpose of reloading the watchdog.

2.2 Runtime

2.2.1 Overview

IWDG1 can be allocated to the Cortex-A7 secure to be used in the secure context by the customer application: this instance is not supported in STMicrolectronics distribution.
IWDG2 can be allocated to the Cortex-A7 non-secure to be used with Linux watchdog framework. In this configuration, the secure monitor (from OP-TEE -if present- or TF-A) is able to receive IWDG early interrupts that can be used in a tentative to reset the Cortex-A7 without interfering with Cortex-M4 execution.

2.2.2 Software frameworks

Domain	Peripheral	Software components		Comment
OP-TEE	Linux	STM32Cube		
Core /Watchdog	IWDG	TF-A	Linux watchdog framework	

2.2.3 Peripheral configuration

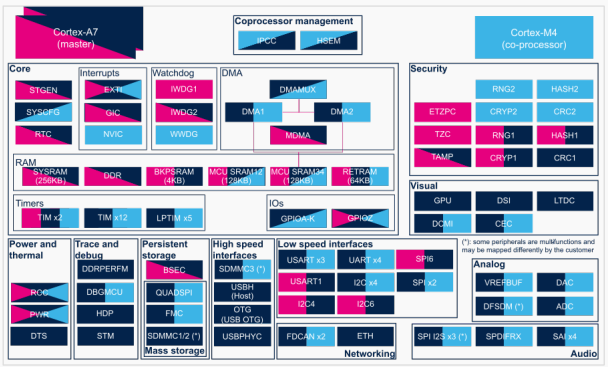
The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration can be done alone via the STM32CubeMX tool for all internal peripherals, and then manually completed (particularly for external peripherals), according to the information given in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

2.2.4 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 /Watchdog	IWDG	IWDG1				
		IWDG2				Shared (none or both): • Cortex-A7 non secure for reload • Cortex-A7 secure for early interrupt handling

Independent Watchdog

Cortex®

Read Only Memory

Linux® is a registered trademark of Linus Torvalds.

Open Portable Trusted Execution Environment