HDP internal peripheral

Contents

Stable: 07.09.2021 - 09:41 / Revision: 12.08.2021 - 15:35

Stable: 01:00.2021 - 00:41 | Revision: 12:00.2021 - 13:00

A quality version of this page, approved on *15 February 2019*, was based off this revision.

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

1 Article purpose

The purpose of this article is to

- briefly introduce the **HDP** peripheral (hardware debug port) 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 needed, how to configure the HDP peripheral.

2 Peripheral overview

The **HDP** peripheral is used to output some internal signals on up to 8 GPIO pins.

Follow the sequence below to connect a GPIO to an internal signal via the HDP:

- First of all, look for the internal signal you want to monitor in the HDP signal multiplexing table of the STM32MP15 reference manuals:
  - Search for the HDP signal on which you can get it among eight possible choices.
  - Note the corresponding **HDPx multiplexing value** to select.
- Then, look for the most suitable GPIO pin on which you can output HDPx in the data brief:
  - Note the GPIO **bank** and **pin**.
  - Note the corresponding GPIO **alternate function** (AF) to select.

The GPIO **bank**, **pin**, **alternate function** and **HDPx multiplexing value** are the information required to configure each HDP signal.

2.1 Features

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

2.2 Security support

The HDP is a **non-secure** peripheral.

3 Peripheral usage and associated software

3.1 Boot time

The HDP is not used at boot time.

3.2 Runtime

3.2.1 Overview

The HDP can be allocated to the Arm® Cortex®-A7 non-secure core to be used under Linux®HDP driver.

3.2.2 Software frameworks

Domain	Peripheral	Software components		Comment
OP-TEE	Linux	STM32Cube		
Trace & Debug	HDP		HDP Linux driver	

3.2.3 Peripheral configuration

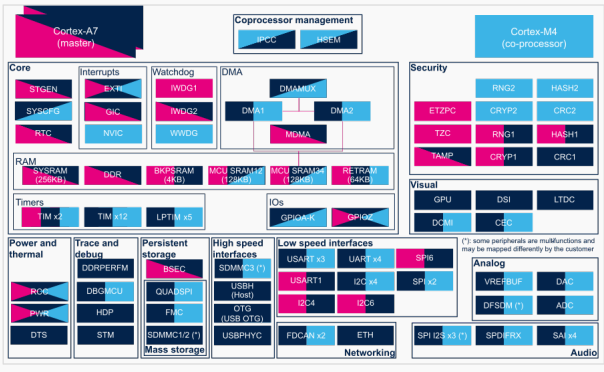
The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration by itself can be performed via the STM32CubeMX tool for all internal peripherals. It can then be manually completed (especially for external peripherals) according to the information 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	Periphera l	Runtime allocation		Comment
	Cortex-A7	Cortex-A7	Cortex-M4	

Domain	Periphera	Runtime allocation				Comment
Instance	secure (OP-TEE)	non-secure (Linux)	(STM32Cube)			
Trace & Debug	HDP	HDP				

Stable: 12.08.2021 - 15:30 / Revision: 12.08.2021 - 15:30

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

1 Article purpose

The purpose of this article is to

- briefly introduce the **HDP** peripheral (hardware debug port) 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 needed, how to configure the HDP peripheral.

2 Peripheral overview

The **HDP** peripheral is used to output some internal signals on up to 8 GPIO pins.

Follow the sequence below to connect a GPIO to an internal signal via the HDP:

- First of all, look for the internal signal you want to monitor in the HDP signal multiplexing table of the STM32MP15 reference manuals:
  - Search for the HDP signal on which you can get it among eight possible choices.
  - Note the corresponding **HDPx multiplexing value** to select.
- Then, look for the most suitable GPIO pin on which you can output HDPx in the data brief:
  - Note the GPIO **bank** and **pin**.
  - Note the corresponding GPIO **alternate function** (AF) to select.

The GPIO **bank**, **pin**, **alternate function** and **HDPx multiplexing value** are the information required to configure each HDP signal.

2.1 Features

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

2.2 Security support

The HDP is a **non-secure** peripheral.

3 Peripheral usage and associated software

3.1 Boot time

The HDP is not used at boot time.

3.2 Runtime

3.2.1 Overview

The HDP can be allocated to the Arm® Cortex®-A7 non-secure core to be used under Linux®HDP driver.

3.2.2 Software frameworks

Domain	Peripheral	Software components			Comment
OP-TEE	Linux	STM32Cube			
Trace & Debug	HDP		HDP Linux driver		

3.2.3 Peripheral configuration

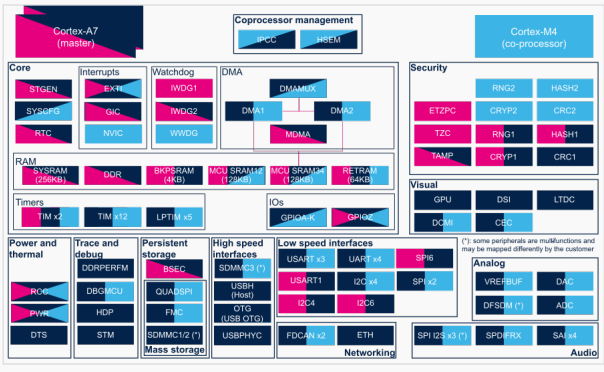
The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration by itself can be performed via the STM32CubeMX tool for all internal peripherals. It can then be manually completed (especially for external peripherals) according to the information 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	Periphera l	Runtime allocation		Comment
	Cortex-A7	Cortex-A7	Cortex-M4	

Domain	Periphera	Runtime allocation				Comment
Instance	secure (OP-TEE)	non-secure (Linux)	(STM32Cube)			
Trace & Debug	HDP	HDP				

Stable: 21.09.2021 - 14:25 / Revision: 21.09.2021 - 14:24

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

1 Article purpose

The purpose of this article is to

- briefly introduce the **HDP** peripheral (hardware debug port) 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 needed, how to configure the HDP peripheral.

2 Peripheral overview

The **HDP** peripheral is used to output some internal signals on up to 8 GPIO pins.

Follow the sequence below to connect a GPIO to an internal signal via the HDP:

- First of all, look for the internal signal you want to monitor in the HDP signal multiplexing table of the STM32MP15 reference manuals:
  - Search for the HDP signal on which you can get it among eight possible choices.
  - Note the corresponding **HDPx multiplexing value** to select.
- Then, look for the most suitable GPIO pin on which you can output HDPx in the data brief:
  - Note the GPIO **bank** and **pin**.
  - Note the corresponding GPIO **alternate function** (AF) to select.

The GPIO **bank**, **pin**, **alternate function** and **HDPx multiplexing value** are the information required to configure each HDP signal.

2.1 Features

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

2.2 Security support

The HDP is a **non-secure** peripheral.

3 Peripheral usage and associated software

3.1 Boot time

The HDP is not used at boot time.

3.2 Runtime

3.2.1 Overview

The HDP can be allocated to the Arm® Cortex®-A7 non-secure core to be used under Linux®HDP driver.

3.2.2 Software frameworks

Domain	Peripheral	Software components			Comment
OP-TEE	Linux	STM32Cube			
Trace & Debug	HDP		HDP Linux driver		

3.2.3 Peripheral configuration

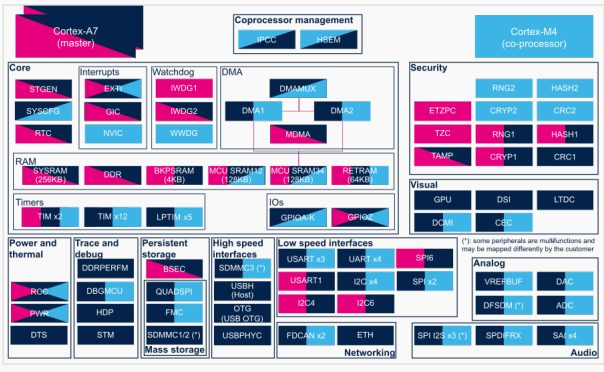
The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration by itself can be performed via the STM32CubeMX tool for all internal peripherals. It can then be manually completed (especially for external peripherals) according to the information 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	Periphera l	Runtime allocation		Comment
	Cortex-A7	Cortex-A7	Cortex-M4	

Domain	Periphera	Runtime allocation				Comment
Instance	secure (OP-TEE)	non-secure (Linux)	(STM32Cube)			
Trace & Debug	HDP	HDP				

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

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

1 Article purpose

The purpose of this article is to

- briefly introduce the **HDP** peripheral (hardware debug port) 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 needed, how to configure the HDP peripheral.

2 Peripheral overview

The **HDP** peripheral is used to output some internal signals on up to 8 GPIO pins.

Follow the sequence below to connect a GPIO to an internal signal via the HDP:

- First of all, look for the internal signal you want to monitor in the HDP signal multiplexing table of the STM32MP15 reference manuals:
  - Search for the HDP signal on which you can get it among eight possible choices.
  - Note the corresponding **HDPx multiplexing value** to select.
- Then, look for the most suitable GPIO pin on which you can output HDPx in the data brief:
  - Note the GPIO **bank** and **pin**.
  - Note the corresponding GPIO **alternate function** (AF) to select.

The GPIO **bank**, **pin**, **alternate function** and **HDPx multiplexing value** are the information required to configure each HDP signal.

2.1 Features

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

2.2 Security support

The HDP is a **non-secure** peripheral.

3 Peripheral usage and associated software

3.1 Boot time

The HDP is not used at boot time.

3.2 Runtime

3.2.1 Overview

The HDP can be allocated to the Arm® Cortex®-A7 non-secure core to be used under Linux®HDP driver.

3.2.2 Software frameworks

Domain	Peripheral	Software components		Comment
OP-TEE	Linux	STM32Cube		
Trace & Debug	HDP		HDP Linux driver	

3.2.3 Peripheral configuration

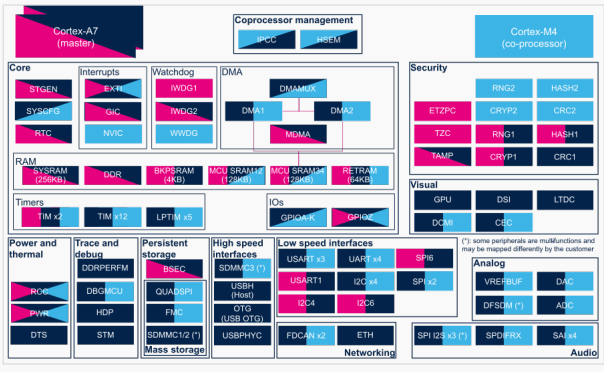
The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration by itself can be performed via the STM32CubeMX tool for all internal peripherals. It can then be manually completed (especially for external peripherals) according to the information 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	Periphera l	Runtime allocation		Comment
	Cortex-A7	Cortex-A7	Cortex-M4	

Domain	Periphera	Runtime allocation				Comment
Instance	secure (OP-TEE)	non-secure (Linux)	(STM32Cube)			
Trace & Debug	HDP	HDP				

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

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

1 Article purpose

The purpose of this article is to

- briefly introduce the **HDP** peripheral (hardware debug port) 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 needed, how to configure the HDP peripheral.

2 Peripheral overview

The **HDP** peripheral is used to output some internal signals on up to 8 GPIO pins.

Follow the sequence below to connect a GPIO to an internal signal via the HDP:

- First of all, look for the internal signal you want to monitor in the HDP signal multiplexing table of the STM32MP15 reference manuals:
  - Search for the HDP signal on which you can get it among eight possible choices.
  - Note the corresponding **HDPx multiplexing value** to select.
- Then, look for the most suitable GPIO pin on which you can output HDPx in the data brief:
  - Note the GPIO **bank** and **pin**.
  - Note the corresponding GPIO **alternate function** (AF) to select.

The GPIO **bank**, **pin**, **alternate function** and **HDPx multiplexing value** are the information required to configure each HDP signal.

2.1 Features

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

2.2 Security support

The HDP is a **non-secure** peripheral.

3 Peripheral usage and associated software

3.1 Boot time

The HDP is not used at boot time.

3.2 Runtime

3.2.1 Overview

The HDP can be allocated to the Arm® Cortex®-A7 non-secure core to be used under Linux®HDP driver.

3.2.2 Software frameworks

Domain	Peripheral	Software components			Comment
OP-TEE	Linux	STM32Cube			
Trace & Debug	HDP		HDP Linux driver		

3.2.3 Peripheral configuration

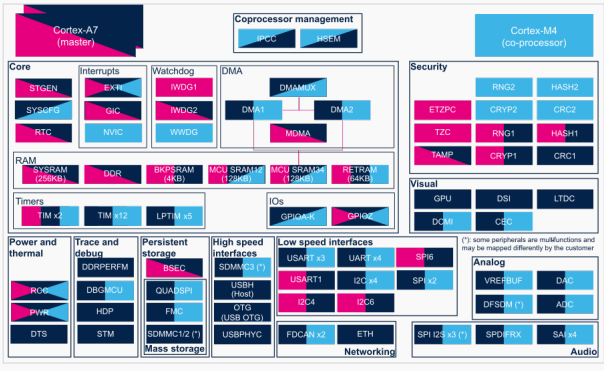
The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration by itself can be performed via the STM32CubeMX tool for all internal peripherals. It can then be manually completed (especially for external peripherals) according to the information 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	Periphera l	Runtime allocation		Comment
	Cortex-A7	Cortex-A7	Cortex-M4	

Domain	Periphera	Runtime allocation				Comment
Instance	secure (OP-TEE)	non-secure (Linux)	(STM32Cube)			
Trace & Debug	HDP	HDP				

Stable: 17.11.2021 - 16:41 / Revision: 17.11.2021 - 10:47

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

1 Article purpose

The purpose of this article is to

- briefly introduce the **HDP** peripheral (hardware debug port) 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 needed, how to configure the HDP peripheral.

2 Peripheral overview

The **HDP** peripheral is used to output some internal signals on up to 8 GPIO pins.

Follow the sequence below to connect a GPIO to an internal signal via the HDP:

- First of all, look for the internal signal you want to monitor in the HDP signal multiplexing table of the STM32MP15 reference manuals:
  - Search for the HDP signal on which you can get it among eight possible choices.
  - Note the corresponding **HDPx multiplexing value** to select.
- Then, look for the most suitable GPIO pin on which you can output HDPx in the data brief:
  - Note the GPIO **bank** and **pin**.
  - Note the corresponding GPIO **alternate function** (AF) to select.

The GPIO **bank**, **pin**, **alternate function** and **HDPx multiplexing value** are the information required to configure each HDP signal.

2.1 Features

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

2.2 Security support

The HDP is a **non-secure** peripheral.

3 Peripheral usage and associated software

3.1 Boot time

The HDP is not used at boot time.

3.2 Runtime

3.2.1 Overview

The HDP can be allocated to the Arm® Cortex®-A7 non-secure core to be used under Linux®HDP driver.

3.2.2 Software frameworks

Domain	Peripheral	Software components			Comment
OP-TEE	Linux	STM32Cube			
Trace & Debug	HDP		HDP Linux driver		

3.2.3 Peripheral configuration

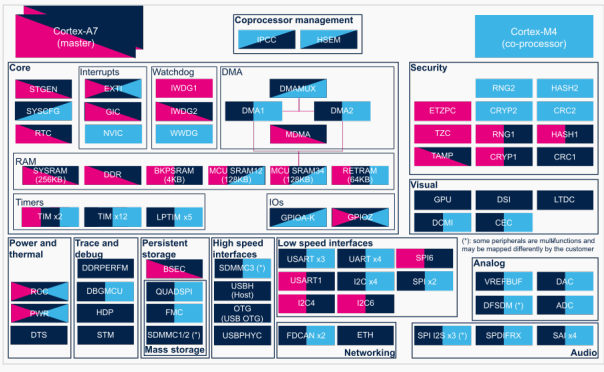
The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration by itself can be performed via the STM32CubeMX tool for all internal peripherals. It can then be manually completed (especially for external peripherals) according to the information 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	Periphera l	Runtime allocation		Comment
	Cortex-A7	Cortex-A7	Cortex-M4	

Domain	Periphera	Runtime allocation				Comment
Instance	secure (OP-TEE)	non-secure (Linux)	(STM32Cube)			
Trace & Debug	HDP	HDP				

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

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

1 Article purpose

The purpose of this article is to

- briefly introduce the **HDP** peripheral (hardware debug port) 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 needed, how to configure the HDP peripheral.

2 Peripheral overview

The **HDP** peripheral is used to output some internal signals on up to 8 GPIO pins.

Follow the sequence below to connect a GPIO to an internal signal via the HDP:

- First of all, look for the internal signal you want to monitor in the HDP signal multiplexing table of the STM32MP15 reference manuals:
  - Search for the HDP signal on which you can get it among eight possible choices.
  - Note the corresponding **HDPx multiplexing value** to select.
- Then, look for the most suitable GPIO pin on which you can output HDPx in the data brief:
  - Note the GPIO **bank** and **pin**.
  - Note the corresponding GPIO **alternate function** (AF) to select.

The GPIO **bank**, **pin**, **alternate function** and **HDPx multiplexing value** are the information required to configure each HDP signal.

2.1 Features

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

2.2 Security support

The HDP is a **non-secure** peripheral.

3 Peripheral usage and associated software

3.1 Boot time

The HDP is not used at boot time.

3.2 Runtime

3.2.1 Overview

The HDP can be allocated to the Arm® Cortex®-A7 non-secure core to be used under Linux®HDP driver.

3.2.2 Software frameworks

Domain	Peripheral	Software components		Comment
OP-TEE	Linux	STM32Cube		
Trace & Debug	HDP		HDP Linux driver	

3.2.3 Peripheral configuration

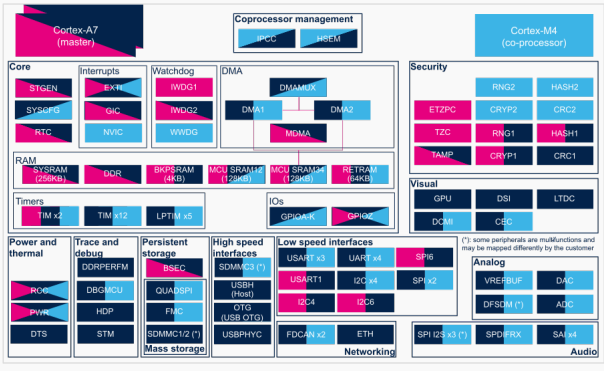
The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration by itself can be performed via the STM32CubeMX tool for all internal peripherals. It can then be manually completed (especially for external peripherals) according to the information 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	Periphera l	Runtime allocation		Comment
	Cortex-A7	Cortex-A7	Cortex-M4	

Domain	Periphera	Runtime allocation				Comment
Instance	secure (OP-TEE)	non-secure (Linux)	(STM32Cube)			
Trace & Debug	HDP	HDP				