



HDP internal peripheral

HDP internal peripheral



Contents

1. HDP internal peripheral	3
2. GPIO internal peripheral	7
3. HDP Linux driver	11
4. How to assign an internal peripheral to a runtime context	15
5. STM32CubeMX	19
6. STM32MP15 resources	23
7. STM32MPU Embedded Software architecture overview	27



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

Template:ArticleMainWriter Template:ArticleApprovedVersion

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



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.

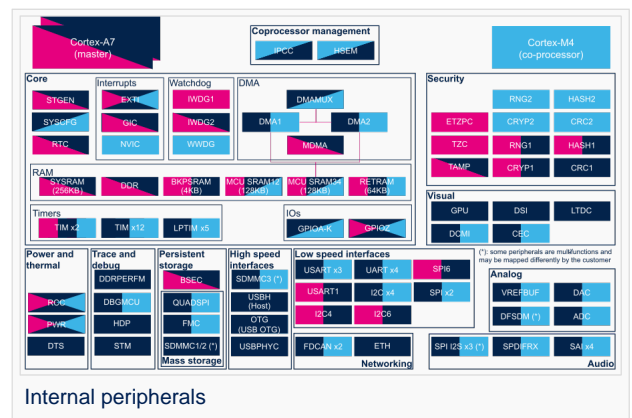
3.2.4 Peripheral assignment

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 possibilities might be described in STM32MP15 reference manuals



Domain	Periphera	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	8
2 Peripheral overview	9
2.1 Features	9
2.2 Security support	9
3 Peripheral usage and associated software	10
3.1 Boot time	10
3.2 Runtime	10
3.2.1 Overview	10
3.2.2 Software frameworks	10
3.2.3 Peripheral configuration	10
3.2.4 Peripheral assignment	10



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.

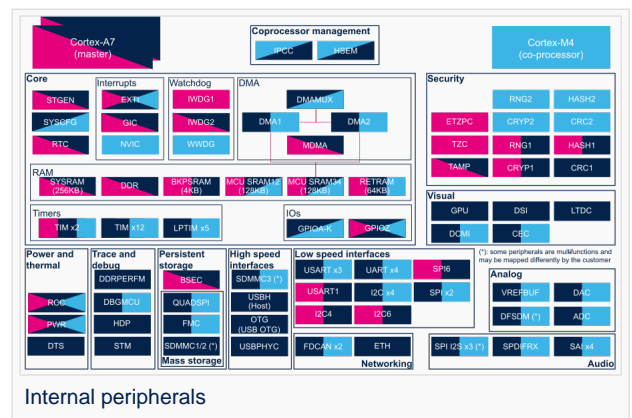
3.2.4 Peripheral assignment

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 possibilities might be described in *STM32MP15 reference manuals*



Internal peripherals

Domain	Periphera	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	12
2 Peripheral overview	13
2.1 Features	13
2.2 Security support	13
3 Peripheral usage and associated software	14
3.1 Boot time	14
3.2 Runtime	14
3.2.1 Overview	14
3.2.2 Software frameworks	14
3.2.3 Peripheral configuration	14
3.2.4 Peripheral assignment	14



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.

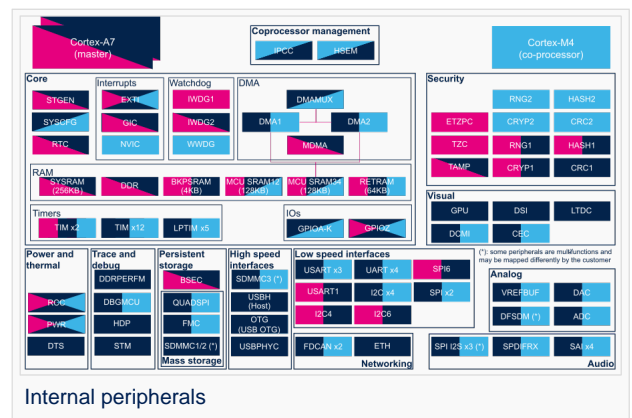
3.2.4 Peripheral assignment

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 possibilities might be described in STM32MP15 reference manuals



Domain	Periphera	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	16
2 Peripheral overview	17
2.1 Features	17
2.2 Security support	17
3 Peripheral usage and associated software	18
3.1 Boot time	18
3.2 Runtime	18
3.2.1 Overview	18
3.2.2 Software frameworks	18
3.2.3 Peripheral configuration	18
3.2.4 Peripheral assignment	18



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.

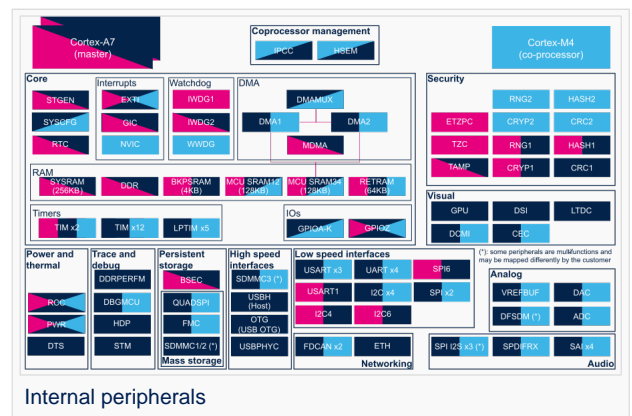
3.2.4 Peripheral assignment

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 possibilities might be described in *STM32MP15* reference manuals



Domain	Periphera	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	20
2 Peripheral overview	21
2.1 Features	21
2.2 Security support	21
3 Peripheral usage and associated software	22
3.1 Boot time	22
3.2 Runtime	22
3.2.1 Overview	22
3.2.2 Software frameworks	22
3.2.3 Peripheral configuration	22
3.2.4 Peripheral assignment	22



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.

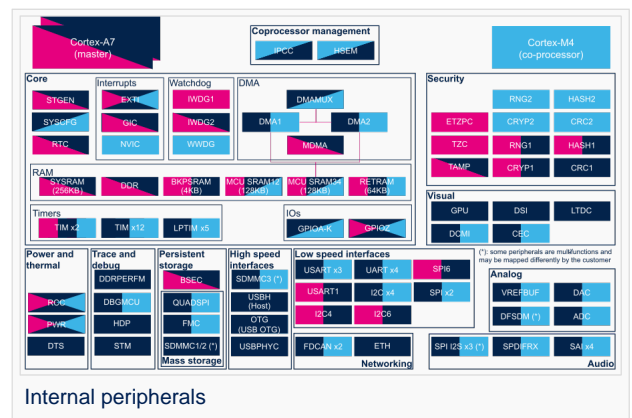
3.2.4 Peripheral assignment

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 possibilities might be described in STM32MP15 reference manuals



Domain	Periphera	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	24
2 Peripheral overview	25
2.1 Features	25
2.2 Security support	25
3 Peripheral usage and associated software	26
3.1 Boot time	26
3.2 Runtime	26
3.2.1 Overview	26
3.2.2 Software frameworks	26
3.2.3 Peripheral configuration	26
3.2.4 Peripheral assignment	26



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.

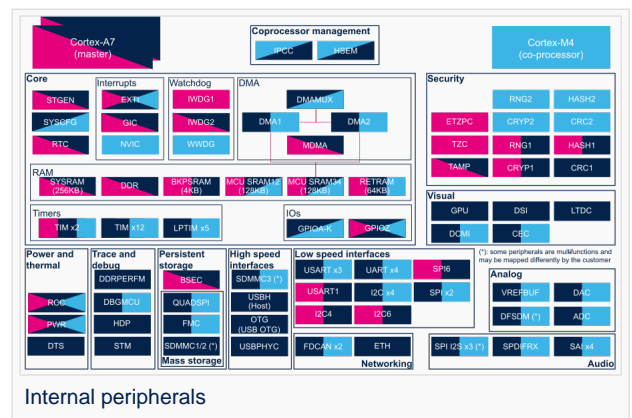
3.2.4 Peripheral assignment

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 possibilities might be described in *STM32MP15* reference manuals



Domain	Periphera	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	28
2 Peripheral overview	29
2.1 Features	29
2.2 Security support	29
3 Peripheral usage and associated software	30
3.1 Boot time	30
3.2 Runtime	30
3.2.1 Overview	30
3.2.2 Software frameworks	30
3.2.3 Peripheral configuration	30
3.2.4 Peripheral assignment	30



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.

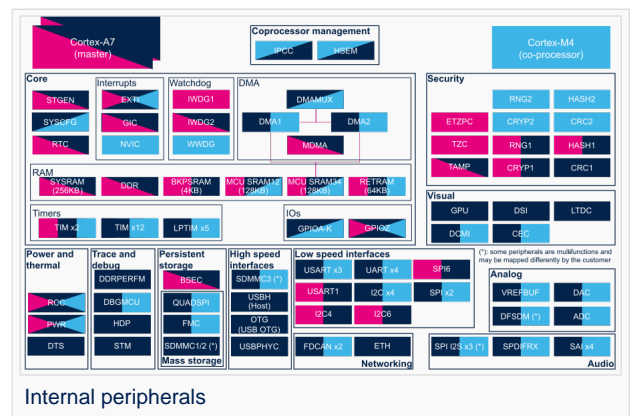
3.2.4 Peripheral assignment

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 possibilities might be described in STM32MP15 reference manuals



Domain	Periphera	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			