VREFBUF internal peripheral

Contents

VREFBUF internal peripheral

Stable: 22.11.2019 - 07:49 / Revision: 20.11.2019 - 16:56

A quality version of this page, accepted on *22 November 2019*, was based off this revision.

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

1 Article purpose

The purpose of this article is to

- briefly introduce the VREFBUF peripheral 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 VREFBUF peripheral.

2 Peripheral overview

The **VREFBUF** peripheral is an internal voltage regulator.

2.1 Features

The VREFBUF is supplied via the VDDA pin. When enabled, it can provide a reference voltage in the range of: 1,5V, 1,8V, 2,048V or 2,5V.

The VREFBUF can be used to provide an analog voltage reference for:

- ADC internal peripheral[1]
- DAC internal peripheral[2]
- External components through the dedicated VREF+ pin.

The VREFBUF can be left unused. In this case, an external voltage regulator can provide reference voltage to VREF+ pin.

Refer to the 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 VREFBUF is a **non-secure** peripheral.

3 Peripheral usage and associated software

3.1 Boot time

The VREFBUF is usually not used at boot time. But it may be needed by the SSBL (see Boot chains overview), to supply the internal ADC[1] for example.

3.2 Runtime

3.2.1 Overview

The VREFBUF can be allocated to the Arm® Cortex®-A7 non-secure to be used under Linux® with regulator framework[3].

The VREFBUF is a system resource[4] which needs to be also controlled by the resource manager[4] in case its consumers (e.g. ADC[1], DAC[2] or an external device connected to VREF+ pin) are spread across:

- the Arm® Cortex®-A7 non-secure context
- the Arm® Cortex®-M4 context

For this reason, the direct control of VREFBUF from the Arm® Cortex®-M4 is not recommended in STM32Cube[5] by default.
It's recommended to implement it in STM32Cube **only if** all consumers and the VDDA supply pin are controlled in the Arm® Cortex®-M4 context.

The Peripheral assignment chapter describes which peripheral instance can be assigned to which context.

3.2.2 Software frameworks

Do	Peri	Software frameworks			Comment
mai Cor n tex -A7 sec ure (O P- TE E)	Cor tex -A7 no n- sec ure (Li nux)	Cortex-M4 (STM32Cube)			
An alo g	V R EF B UF		Linux regulator framework		

3.2.3 Peripheral configuration

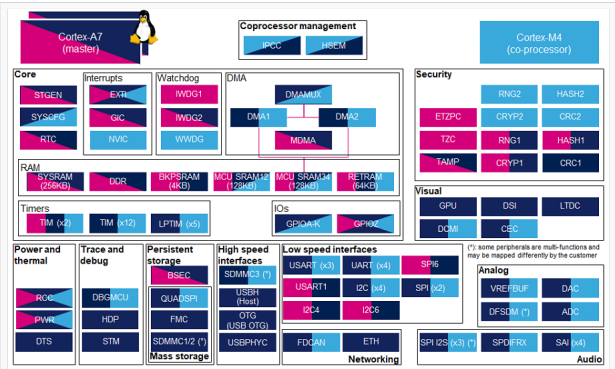
The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration by itself can be performed via the STM32CubeMX tool for all internal peripherals. It can then be manually completed (especially for external peripherals) according to the information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

- For the Linux kernel configuration, please refer to device internal regulator. An example can be found also in ADC DT configuration example
- In case the control of VREFBUF consumers are spread across the various cores, see also Resource manager for coprocessing

3.2.4 Peripheral assignment

Check boxes illustrate the possible peripheral allocations supported by STM32 MPU Embedded Software:

- means that the peripheral can be assigned () to the given runtime context.
- is used for system peripherals that cannot be unchecked because they are statically connected in the device.



Internal peripherals

Refer to How to assign an internal peripheral to a runtime context for more information on how to assign peripherals manually or via STM32CubeMX.
The present chapter describes STMicroelectronics recommendations or choice of implementation. Additional possiblities might be described in STM32MP15 reference manuals.

Domain	Peripheral	Runtime allocation				Comment
Instance	Cortex-A7 secure (OP-TEE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)			
Analog	VREFBUF	VREFBUF				Assignment (single choice)

4 References

- 1.01.11.2 ADC internal peripheral
- 2.02.1 DAC internal peripheral
- Regulator overview, Linux® regulator framework overview
- 4.04.1 Resource manager for coprocessing, focus on system resources
- STM32CubeMP1 architecture

voltage reference buffer (STM32 specific)

Analog-to-digital converter. The process of converting a sampled analog signal to a digital code that represents the amplitude of the original signal sample.

Digital-to-analog converter (Electronic circuit that converts a binary number into a continuously varying value.)

Second Stage Boot Loader

Open Portable Trusted Execution Environment

VREFBUF internal peripheral

Stable: 21.02.2020 - 08:59 / Revision: 14.02.2020 - 10:13

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

1 Article purpose

The purpose of this article is to

- briefly introduce the VREFBUF peripheral 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 VREFBUF peripheral.

2 Peripheral overview

The **VREFBUF** peripheral is an internal voltage regulator.

2.1 Features

The VREFBUF is supplied via the VDDA pin. When enabled, it can provide a reference voltage in the range of: 1,5V, 1,8V, 2,048V or 2,5V.

The VREFBUF can be used to provide an analog voltage reference for:

- ADC internal peripheral[1]
- DAC internal peripheral[2]
- External components through the dedicated VREF+ pin.

The VREFBUF can be left unused. In this case, an external voltage regulator can provide reference voltage to VREF+ pin.

Refer to the 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 VREFBUF is a **non-secure** peripheral.

3 Peripheral usage and associated software

3.1 Boot time

The VREFBUF is usually not used at boot time. But it may be needed by the SSBL (see Boot chains overview), to supply the internal ADC[1] for example.

3.2 Runtime

3.2.1 Overview

The VREFBUF can be allocated to the Arm® Cortex®-A7 non-secure to be used under Linux® with regulator framework[3].

The VREFBUF is a system resource[4] which needs to be also controlled by the resource manager[4] in case its consumers (e.g. ADC[1], DAC[2] or an external device connected to VREF+ pin) are spread across:

- the Arm® Cortex®-A7 non-secure context
- the Arm® Cortex®-M4 context

For this reason, the direct control of VREFBUF from the Arm® Cortex®-M4 is not recommended in STM32Cube[5] by default.
It's recommended to implement it in STM32Cube **only if** all consumers and the VDDA supply pin are controlled in the Arm® Cortex®-M4 context.

The Peripheral assignment chapter describes which peripheral instance can be assigned to which context.

3.2.2 Software frameworks

Domain	Peripheral	Software frameworks	Comment

Domain	Peri	Software frameworks			Comment
Cortex-A7 secure (OP-TEE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)			
Analog	VREFBUF		Linux regulator framework		

3.2.3 Peripheral configuration

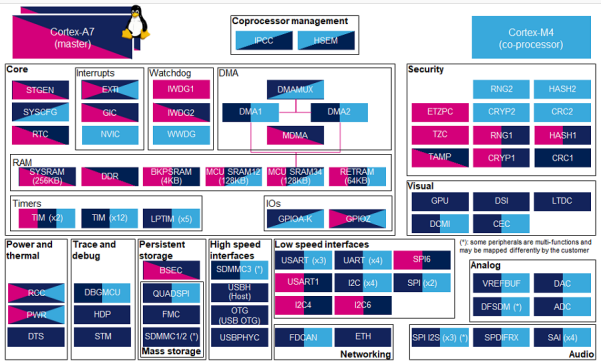
The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration by itself can be performed via the STM32CubeMX tool for all internal peripherals. It can then be manually completed (especially for external peripherals) according to the information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

- For the Linux kernel configuration, please refer to device internal regulator. An example can be found also in ADC DT configuration example
- In case the control of VREFBUF consumers are spread across the various cores, see also Resource manager for coprocessing

3.2.4 Peripheral assignment

Check boxes illustrate the possible peripheral allocations supported by STM32 MPU Embedded Software:

- means that the peripheral can be assigned () to the given runtime context.
- is used for system peripherals that cannot be unchecked because they are statically connected in the device.



Internal peripherals

Refer to How to assign an internal peripheral to a runtime context for more information on how to assign peripherals manually or via STM32CubeMX.
The present chapter describes STMicroelectronics recommendations or choice of implementation. Additional possiblities might be described in STM32MP15 reference manuals.

Domain	Peripheral	Runtime allocation			Comment

Domain Instance	Peripheral	Runtime allocation				Comment
	x-A7 secure (OP-TEE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)			
Analog	VREFBUF	VREFBUF				Assignment (single choice)

4 References

- 1.01.11.2 ADC internal peripheral
- 2.02.1 DAC internal peripheral
- Regulator overview, Linux® regulator framework overview
- 4.04.1 Resource manager for coprocessing, focus on system resources
- STM32CubeMP1 architecture

voltage reference buffer (STM32 specific)

Analog-to-digital converter. The process of converting a sampled analog signal to a digital code that represents the amplitude of the original signal sample.

Digital-to-analog converter (Electronic circuit that converts a binary number into a continuously varying value.)

Second Stage Boot Loader

Open Portable Trusted Execution Environment

VREFBUF internal peripheral

Stable: 22.01.2020 - 16:05 / Revision: 22.01.2020 - 10:03

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

1 Article purpose

The purpose of this article is to

- briefly introduce the VREFBUF peripheral 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 VREFBUF peripheral.

2 Peripheral overview

The **VREFBUF** peripheral is an internal voltage regulator.

2.1 Features

The VREFBUF is supplied via the VDDA pin. When enabled, it can provide a reference voltage in the range of: 1,5V, 1,8V, 2,048V or 2,5V.

The VREFBUF can be used to provide an analog voltage reference for:

- ADC internal peripheral[1]
- DAC internal peripheral[2]
- External components through the dedicated VREF+ pin.

The VREFBUF can be left unused. In this case, an external voltage regulator can provide reference voltage to VREF+ pin.

Refer to the 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 VREFBUF is a **non-secure** peripheral.

3 Peripheral usage and associated software

3.1 Boot time

The VREFBUF is usually not used at boot time. But it may be needed by the SSBL (see Boot chains overview), to supply the internal ADC[1] for example.

3.2 Runtime

3.2.1 Overview

The VREFBUF can be allocated to the Arm® Cortex®-A7 non-secure to be used under Linux® with regulator framework[3].

The VREFBUF is a system resource[4] which needs to be also controlled by the resource manager[4] in case its consumers (e.g. ADC[1], DAC[2] or an external device connected to VREF+ pin) are spread across:

- the Arm® Cortex®-A7 non-secure context
- the Arm® Cortex®-M4 context

For this reason, the direct control of VREFBUF from the Arm® Cortex®-M4 is not recommended in STM32Cube[5] by default.
It's recommended to implement it in STM32Cube **only if** all consumers and the VDDA supply pin are controlled in the Arm® Cortex®-M4 context.

The Peripheral assignment chapter describes which peripheral instance can be assigned to which context.

3.2.2 Software frameworks

Domain	Peripheral	Software frameworks	Comment
Cortex-A7	Cortex-A7 non-secure	Cortex-M4	

Do	Peri	Software frameworks			Comment
secure (OP-TEE)	non-secure (Linux)	(STM32Cube)			
Analog	VREFBUF		Linux regulator framework		

3.2.3 Peripheral configuration

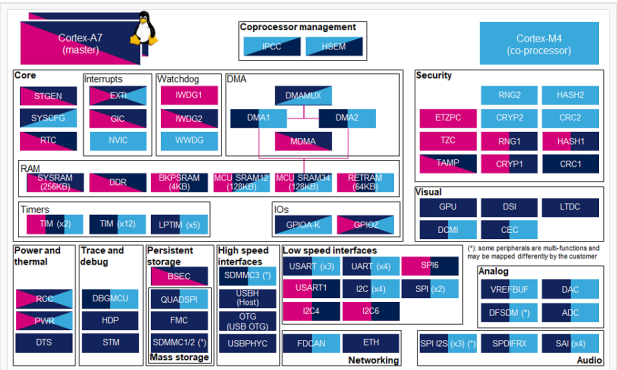
The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration by itself can be performed via the STM32CubeMX tool for all internal peripherals. It can then be manually completed (especially for external peripherals) according to the information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

- For the Linux kernel configuration, please refer to device internal regulator. An example can be found also in ADC DT configuration example
- In case the control of VREFBUF consumers are spread across the various cores, see also Resource manager for coprocessing

3.2.4 Peripheral assignment

Check boxes illustrate the possible peripheral allocations supported by STM32 MPU Embedded Software:

- means that the peripheral can be assigned () to the given runtime context.
- is used for system peripherals that cannot be unchecked because they are statically connected in the device.



Internal peripherals

Refer to How to assign an internal peripheral to a runtime context for more information on how to assign peripherals manually or via STM32CubeMX.
The present chapter describes STMicroelectronics recommendations or choice of implementation. Additional possiblities might be described in STM32MP15 reference manuals.

Do	Per	Runtime allocation			Comment
main	inh				
	Cortex-A7				

Domain instance	Peripheral secure (OP-TEE)	Runtime allocation				Comment
		Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)			
Analog	VREFBUF	VREFBUF				Assignment (single choice)

4 References

- 1.01.11.2 ADC internal peripheral
- 2.02.1 DAC internal peripheral
- Regulator overview, Linux® regulator framework overview
- 4.04.1 Resource manager for coprocessing, focus on system resources
- STM32CubeMP1 architecture

voltage reference buffer (STM32 specific)

Analog-to-digital converter. The process of converting a sampled analog signal to a digital code that represents the amplitude of the original signal sample.

Digital-to-analog converter (Electronic circuit that converts a binary number into a continuously varying value.)

Second Stage Boot Loader

Open Portable Trusted Execution Environment

VREFBUF internal peripheral

Stable: 31.01.2020 - 13:22 / Revision: 31.01.2020 - 13:14

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

1 Article purpose

The purpose of this article is to

- briefly introduce the VREFBUF peripheral 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 VREFBUF peripheral.

2 Peripheral overview

The **VREFBUF** peripheral is an internal voltage regulator.

2.1 Features

The VREFBUF is supplied via the VDDA pin. When enabled, it can provide a reference voltage in the range of: 1,5V, 1,8V, 2,048V or 2,5V.

The VREFBUF can be used to provide an analog voltage reference for:

- ADC internal peripheral[1]
- DAC internal peripheral[2]
- External components through the dedicated VREF+ pin.

The VREFBUF can be left unused. In this case, an external voltage regulator can provide reference voltage to VREF+ pin.

Refer to the 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 VREFBUF is a **non-secure** peripheral.

3 Peripheral usage and associated software

3.1 Boot time

The VREFBUF is usually not used at boot time. But it may be needed by the SSBL (see Boot chains overview), to supply the internal ADC[1] for example.

3.2 Runtime

3.2.1 Overview

The VREFBUF can be allocated to the Arm® Cortex®-A7 non-secure to be used under Linux® with regulator framework[3].

The VREFBUF is a system resource[4] which needs to be also controlled by the resource manager[4] in case its consumers (e.g. ADC[1], DAC[2] or an external device connected to VREF+ pin) are spread across:

- the Arm® Cortex®-A7 non-secure context
- the Arm® Cortex®-M4 context

For this reason, the direct control of VREFBUF from the Arm® Cortex®-M4 is not recommended in STM32Cube[5] by default.
It's recommended to implement it in STM32Cube **only if** all consumers and the VDDA supply pin are controlled in the Arm® Cortex®-M4 context.

The Peripheral assignment chapter describes which peripheral instance can be assigned to which context.

3.2.2 Software frameworks

Do	Peri	Software frameworks	Comment
mai n Cor tex -A7 sec ure (O P- TE E)	Cor tex -A7 no n- sec ure (Li nux)	Cortex-M4 (STM32Cube)	

Domain	Peripheral	Software frameworks			Comment
Analog	VREFBUF		Linux regulator framework		

3.2.3 Peripheral configuration

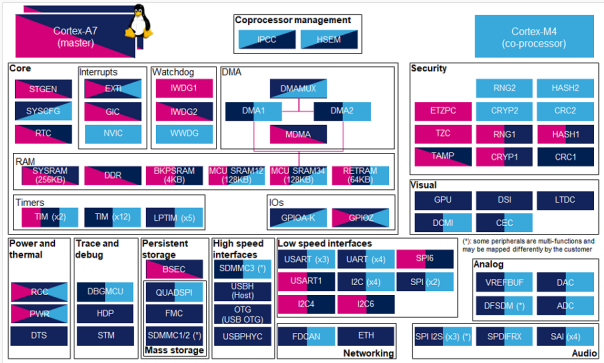
The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration by itself can be performed via the STM32CubeMX tool for all internal peripherals. It can then be manually completed (especially for external peripherals) according to the information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

- For the Linux kernel configuration, please refer to device internal regulator. An example can be found also in ADC DT configuration example
- In case the control of VREFBUF consumers are spread across the various cores, see also Resource manager for coprocessing

3.2.4 Peripheral assignment

Check boxes illustrate the possible peripheral allocations supported by STM32 MPU Embedded Software:

- means that the peripheral can be assigned () to the given runtime context.
- is used for system peripherals that cannot be unchecked because they are statically connected in the device.



Internal peripherals

Refer to How to assign an internal peripheral to a runtime context for more information on how to assign peripherals manually or via STM32CubeMX.
The present chapter describes STMicroelectronics recommendations or choice of implementation. Additional possiblities might be described in STM32MP15 reference manuals.

Domain	Peripheral	Runtime allocation		Comment
Instance	cortex-A7 secure (TrustZone)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)	

Domain	Peripheral	Runtime allocation				Comment
	OP-TEE)					
Analog	VREFBUF	VREFBUF				Assignment (single choice)

4 References

- 1.01.11.2 ADC internal peripheral
- 2.02.1 DAC internal peripheral
- Regulator overview, Linux® regulator framework overview
- 4.04.1 Resource manager for coprocessing, focus on system resources
- STM32CubeMP1 architecture

voltage reference buffer (STM32 specific)

Analog-to-digital converter. The process of converting a sampled analog signal to a digital code that represents the amplitude of the original signal sample.

Digital-to-analog converter (Electronic circuit that converts a binary number into a continuously varying value.)

Second Stage Boot Loader

Open Portable Trusted Execution Environment

VREFBUF internal peripheral

Stable: 31.01.2020 - 13:04 / Revision: 31.01.2020 - 13:02

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

1 Article purpose

The purpose of this article is to

- briefly introduce the VREFBUF peripheral 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 VREFBUF peripheral.

2 Peripheral overview

The **VREFBUF** peripheral is an internal voltage regulator.

2.1 Features

The VREFBUF is supplied via the VDDA pin. When enabled, it can provide a reference voltage in the range of: 1,5V, 1,8V, 2,048V or 2,5V.

The VREFBUF can be used to provide an analog voltage reference for:

- ADC internal peripheral[1]
- DAC internal peripheral[2]
- External components through the dedicated VREF+ pin.

The VREFBUF can be left unused. In this case, an external voltage regulator can provide reference voltage to VREF+ pin.

Refer to the 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 VREFBUF is a **non-secure** peripheral.

3 Peripheral usage and associated software

3.1 Boot time

The VREFBUF is usually not used at boot time. But it may be needed by the SSBL (see Boot chains overview), to supply the internal ADC[1] for example.

3.2 Runtime

3.2.1 Overview

The VREFBUF can be allocated to the Arm® Cortex®-A7 non-secure to be used under Linux® with regulator framework[3].

> The VREFBUF is a system resource[4] which needs to be also controlled by the resource manager[4] in case its consumers (e.g. ADC[1], DAC[2] or an external device connected to VREF+ pin) are spread across:
>
> - the Arm® Cortex®-A7 non-secure context
> - the Arm® Cortex®-M4 context
>
> For this reason, the direct control of VREFBUF from the Arm® Cortex®-M4 is not recommended in STM32Cube[5] by default.
> It's recommended to implement it in STM32Cube **only if** all consumers and the VDDA supply pin are controlled in the Arm® Cortex®-M4 context.

The Peripheral assignment chapter describes which peripheral instance can be assigned to which context.

3.2.2 Software frameworks

Do	Peri	Software frameworks			Comment
mai n Cor tex -A7 sec ure (O P- TE E)	Cor tex -A7 no n- sec ure (Li nux)	Cortex-M4 (STM32Cube)			
An alo	V R EF B		Linux regulator		

Domain	Peripheral	Software frameworks		Comment
		framework		

3.2.3 Peripheral configuration

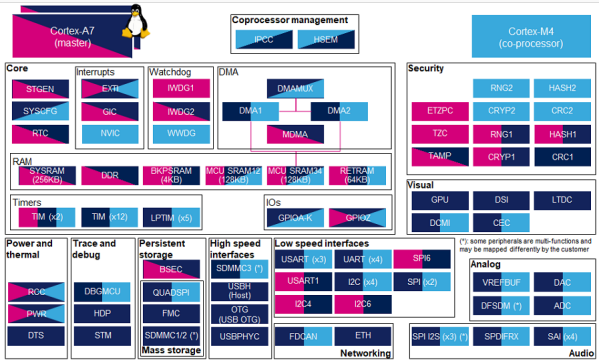
The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration by itself can be performed via the STM32CubeMX tool for all internal peripherals. It can then be manually completed (especially for external peripherals) according to the information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

- For the Linux kernel configuration, please refer to device internal regulator. An example can be found also in ADC DT configuration example
- In case the control of VREFBUF consumers are spread across the various cores, see also Resource manager for coprocessing

3.2.4 Peripheral assignment

Check boxes illustrate the possible peripheral allocations supported by STM32 MPU Embedded Software:

- means that the peripheral can be assigned () to the given runtime context.
- is used for system peripherals that cannot be unchecked because they are statically connected in the device.



Internal peripherals

Refer to How to assign an internal peripheral to a runtime context for more information on how to assign peripherals manually or via STM32CubeMX.
The present chapter describes STMicroelectronics recommendations or choice of implementation. Additional possiblities might be described in STM32MP15 reference manuals.

Domain	Peripheral	Runtime allocation				Comment
Instance	Cortex-A7 secure (OP-TEE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)			
	V					

Domain	Peripheral	Runtime allocation				Comment
Analog	VREFBUF	VREFBUF				Assignment (single choice)

4 References

- 1.01.11.2 ADC internal peripheral
- 2.02.1 DAC internal peripheral
- Regulator overview, Linux® regulator framework overview
- 4.04.1 Resource manager for coprocessing, focus on system resources
- STM32CubeMP1 architecture

voltage reference buffer (STM32 specific)

Analog-to-digital converter. The process of converting a sampled analog signal to a digital code that represents the amplitude of the original signal sample.

Digital-to-analog converter (Electronic circuit that converts a binary number into a continuously varying value.)

Second Stage Boot Loader

Open Portable Trusted Execution Environment

VREFBUF internal peripheral

Stable: 21.02.2020 - 08:58 / Revision: 07.02.2020 - 08:29

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

1 Article purpose

The purpose of this article is to

- briefly introduce the VREFBUF peripheral 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 VREFBUF peripheral.

2 Peripheral overview

The **VREFBUF** peripheral is an internal voltage regulator.

2.1 Features

The VREFBUF is supplied via the VDDA pin. When enabled, it can provide a reference voltage in the range of: 1,5V, 1,8V, 2,048V or 2,5V.

The VREFBUF can be used to provide an analog voltage reference for:

- ADC internal peripheral[1]
- DAC internal peripheral[2]
- External components through the dedicated VREF+ pin.

The VREFBUF can be left unused. In this case, an external voltage regulator can provide reference voltage to VREF+ pin.

Refer to the 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 VREFBUF is a **non-secure** peripheral.

3 Peripheral usage and associated software

3.1 Boot time

The VREFBUF is usually not used at boot time. But it may be needed by the SSBL (see Boot chains overview), to supply the internal ADC[1] for example.

3.2 Runtime

3.2.1 Overview

The VREFBUF can be allocated to the Arm® Cortex®-A7 non-secure to be used under Linux® with regulator framework[3].

The VREFBUF is a system resource[4] which needs to be also controlled by the resource manager[4] in case its consumers (e.g. ADC[1], DAC[2] or an external device connected to VREF+ pin) are spread across:

- the Arm® Cortex®-A7 non-secure context
- the Arm® Cortex®-M4 context

For this reason, the direct control of VREFBUF from the Arm® Cortex®-M4 is not recommended in STM32Cube[5] by default.
It's recommended to implement it in STM32Cube **only if** all consumers and the VDDA supply pin are controlled in the Arm® Cortex®-M4 context.

The Peripheral assignment chapter describes which peripheral instance can be assigned to which context.

3.2.2 Software frameworks

Do	Peri	Software frameworks			Comment
mai n Cor tex -A7 sec ure (O P- TE E)	Cor tex -A7 no n- sec ure (Li nux)	Cortex-M4 (STM32Cube)			
An	V R				

Domain	Peripheral	Software frameworks			Comment
ano g	EP B UF		Linux regulator framework		

3.2.3 Peripheral configuration

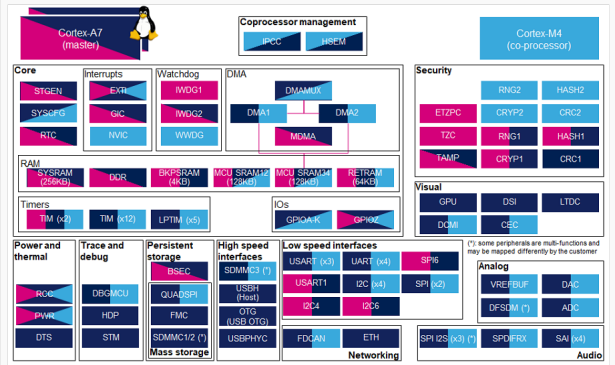
The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration by itself can be performed via the STM32CubeMX tool for all internal peripherals. It can then be manually completed (especially for external peripherals) according to the information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

- For the Linux kernel configuration, please refer to device internal regulator. An example can be found also in ADC DT configuration example
- In case the control of VREFBUF consumers are spread across the various cores, see also Resource manager for coprocessing

3.2.4 Peripheral assignment

Check boxes illustrate the possible peripheral allocations supported by STM32 MPU Embedded Software:

- means that the peripheral can be assigned () to the given runtime context.
- is used for system peripherals that cannot be unchecked because they are statically connected in the device.



Internal peripherals

Refer to How to assign an internal peripheral to a runtime context for more information on how to assign peripherals manually or via STM32CubeMX.
The present chapter describes STMicroelectronics recommendations or choice of implementation. Additional possiblities might be described in STM32MP15 reference manuals.

Domain	Peripheral	Runtime allocation		Comment
Instance	Cortex-A7 secure (	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)	

Domain	Peripheral	Runtime allocation				Comment
	OP-TEE)					
Analog	VREFBUF	VREFBUF				Assignment (single choice)

4 References

- 1.01.11.2 ADC internal peripheral
- 2.02.1 DAC internal peripheral
- Regulator overview, Linux® regulator framework overview
- 4.04.1 Resource manager for coprocessing, focus on system resources
- STM32CubeMP1 architecture

voltage reference buffer (STM32 specific)

Analog-to-digital converter. The process of converting a sampled analog signal to a digital code that represents the amplitude of the original signal sample.

Digital-to-analog converter (Electronic circuit that converts a binary number into a continuously varying value.)

Second Stage Boot Loader

Open Portable Trusted Execution Environment

VREFBUF internal peripheral

Stable: 27.09.2019 - 07:08 / Revision: 18.09.2019 - 07:38

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

1 Article purpose

The purpose of this article is to

- briefly introduce the VREFBUF peripheral 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 VREFBUF peripheral.

2 Peripheral overview

The **VREFBUF** peripheral is an internal voltage regulator.

2.1 Features

The VREFBUF is supplied via the VDDA pin. When enabled, it can provide a reference voltage in the range of: 1,5V, 1,8V, 2,048V or 2,5V.

The VREFBUF can be used to provide an analog voltage reference for:

- ADC internal peripheral[1]
- DAC internal peripheral[2]
- External components through the dedicated VREF+ pin.

The VREFBUF can be left unused. In this case, an external voltage regulator can provide reference voltage to VREF+ pin.

Refer to the 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 VREFBUF is a **non-secure** peripheral.

3 Peripheral usage and associated software

3.1 Boot time

The VREFBUF is usually not used at boot time. But it may be needed by the SSBL (see Boot chains overview), to supply the internal ADC[1] for example.

3.2 Runtime

3.2.1 Overview

The VREFBUF can be allocated to the Arm® Cortex®-A7 non-secure to be used under Linux® with regulator framework[3].

> The VREFBUF is a system resource[4] which needs to be also controlled by the resource manager[4] in case its consumers (e.g. ADC[1], DAC[2] or an external device connected to VREF+ pin) are spread across:
>
> - the Arm® Cortex®-A7 non-secure context
> - the Arm® Cortex®-M4 context
>
> For this reason, the direct control of VREFBUF from the Arm® Cortex®-M4 is not recommended in STM32Cube[5] by default.
> It's recommended to implement it in STM32Cube **only if** all consumers and the VDDA supply pin are controlled in the Arm® Cortex®-M4 context.

The Peripheral assignment chapter describes which peripheral instance can be assigned to which context.

3.2.2 Software frameworks

Do	Peri	Software frameworks			Comment
mai n Cor tex -A7 sec ure (O P- TE E)	Cor tex -A7 no n- sec ure (Li nux)	Cortex-M4 (STM32Cube)			
An alo	V R EF B		Linux regulator		

Domain	Peripheral	Software frameworks		Comment
		framework		

3.2.3 Peripheral configuration

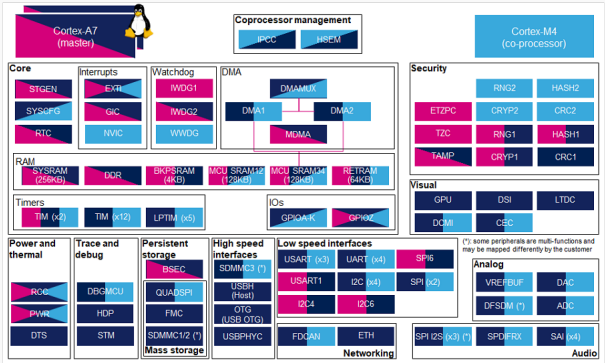
The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration by itself can be performed via the STM32CubeMX tool for all internal peripherals. It can then be manually completed (especially for external peripherals) according to the information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

- For the Linux kernel configuration, please refer to device internal regulator. An example can be found also in ADC DT configuration example
- In case the control of VREFBUF consumers are spread across the various cores, see also Resource manager for coprocessing

3.2.4 Peripheral assignment

Check boxes illustrate the possible peripheral allocations supported by STM32 MPU Embedded Software:

- means that the peripheral can be assigned () to the given runtime context.
- is used for system peripherals that cannot be unchecked because they are statically connected in the device.



Internal peripherals

Refer to How to assign an internal peripheral to a runtime context for more information on how to assign peripherals manually or via STM32CubeMX.
The present chapter describes STMicroelectronics recommendations or choice of implementation. Additional possiblities might be described in STM32MP15 reference manuals.

Domain	Periph	Runtime allocation				Comment
Instance	Cortex-A7 secure (OP-TEE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)			
	V					

Domain	Peripheral	Runtime allocation				Comment
Analog	VREFBUF	VREFBUF				Assignment (single choice)

4 References

- 1.01.11.2 ADC internal peripheral
- 2.02.1 DAC internal peripheral
- Regulator overview, Linux® regulator framework overview
- 4.04.1 Resource manager for coprocessing, focus on system resources
- STM32CubeMP1 architecture

voltage reference buffer (STM32 specific)

Analog-to-digital converter. The process of converting a sampled analog signal to a digital code that represents the amplitude of the original signal sample.

Digital-to-analog converter (Electronic circuit that converts a binary number into a continuously varying value.)

Second Stage Boot Loader

Open Portable Trusted Execution Environment

VREFBUF internal peripheral

Stable: 15.10.2019 - 11:55 / Revision: 15.10.2019 - 11:55

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

1 Article purpose

The purpose of this article is to

- briefly introduce the VREFBUF peripheral 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 VREFBUF peripheral.

2 Peripheral overview

The **VREFBUF** peripheral is an internal voltage regulator.

2.1 Features

The VREFBUF is supplied via the VDDA pin. When enabled, it can provide a reference voltage in the range of: 1,5V, 1,8V, 2,048V or 2,5V.

The VREFBUF can be used to provide an analog voltage reference for:

- ADC internal peripheral[1]
- DAC internal peripheral[2]
- External components through the dedicated VREF+ pin.

The VREFBUF can be left unused. In this case, an external voltage regulator can provide reference voltage to VREF+ pin.

Refer to the 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 VREFBUF is a **non-secure** peripheral.

3 Peripheral usage and associated software

3.1 Boot time

The VREFBUF is usually not used at boot time. But it may be needed by the SSBL (see Boot chains overview), to supply the internal ADC[1] for example.

3.2 Runtime

3.2.1 Overview

The VREFBUF can be allocated to the Arm® Cortex®-A7 non-secure to be used under Linux® with regulator framework[3].

The VREFBUF is a system resource[4] which needs to be also controlled by the resource manager[4] in case its consumers (e.g. ADC[1], DAC[2] or an external device connected to VREF+ pin) are spread across:

- the Arm® Cortex®-A7 non-secure context
- the Arm® Cortex®-M4 context

For this reason, the direct control of VREFBUF from the Arm® Cortex®-M4 is not recommended in STM32Cube[5] by default.
It's recommended to implement it in STM32Cube **only if** all consumers and the VDDA supply pin are controlled in the Arm® Cortex®-M4 context.

The Peripheral assignment chapter describes which peripheral instance can be assigned to which context.

3.2.2 Software frameworks

Do	Peri	Software frameworks			Comment
mai n Cor tex -A7 sec ure (O P- TE E)	Cor tex -A7 no n- sec ure (Li nux)	Cortex-M4 (STM32Cube)			
An	V R				

Domain	Peripheral	Software frameworks		Comment
ano g	EP B UF		Linux regulator framework	

3.2.3 Peripheral configuration

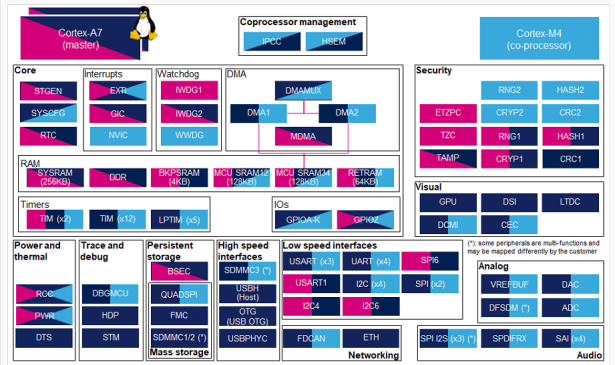
The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration by itself can be performed via the STM32CubeMX tool for all internal peripherals. It can then be manually completed (especially for external peripherals) according to the information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

- For the Linux kernel configuration, please refer to device internal regulator. An example can be found also in ADC DT configuration example
- In case the control of VREFBUF consumers are spread across the various cores, see also Resource manager for coprocessing

3.2.4 Peripheral assignment

Check boxes illustrate the possible peripheral allocations supported by STM32 MPU Embedded Software:

- means that the peripheral can be assigned () to the given runtime context.
- is used for system peripherals that cannot be unchecked because they are statically connected in the device.



Internal peripherals

Refer to How to assign an internal peripheral to a runtime context for more information on how to assign peripherals manually or via STM32CubeMX.
The present chapter describes STMicroelectronics recommendations or choice of implementation. Additional possiblities might be described in STM32MP15 reference manuals.

Domain	Peripheral	Runtime allocation		Comment
Instance	Cortex-A7 secure (	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)	

Domain	Peripheral	Runtime allocation				Comment
	OP-TEE)					
Analog	VREFBUF	VREFBUF				Assignment (single choice)

4 References

- 1.01.11.2 ADC internal peripheral
- 2.02.1 DAC internal peripheral
- Regulator overview, Linux® regulator framework overview
- 4.04.1 Resource manager for coprocessing, focus on system resources
- STM32CubeMP1 architecture

voltage reference buffer (STM32 specific)

Analog-to-digital converter. The process of converting a sampled analog signal to a digital code that represents the amplitude of the original signal sample.

Digital-to-analog converter (Electronic circuit that converts a binary number into a continuously varying value.)

Second Stage Boot Loader

Open Portable Trusted Execution Environment

VREFBUF internal peripheral

Stable: ***Not stable*** */ Revision: 30.01.2020 - 08:45*

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

1 Article purpose

The purpose of this article is to

- briefly introduce the VREFBUF peripheral 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 VREFBUF peripheral.

2 Peripheral overview

The **VREFBUF** peripheral is an internal voltage regulator.

2.1 Features

The VREFBUF is supplied via the VDDA pin. When enabled, it can provide a reference voltage in the range of: 1,5V, 1,8V, 2,048V or 2,5V.

The VREFBUF can be used to provide an analog voltage reference for:

- ADC internal peripheral[1]
- DAC internal peripheral[2]
- External components through the dedicated VREF+ pin.

The VREFBUF can be left unused. In this case, an external voltage regulator can provide reference voltage to VREF+ pin.

Refer to the 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 VREFBUF is a **non-secure** peripheral.

3 Peripheral usage and associated software

3.1 Boot time

The VREFBUF is usually not used at boot time. But it may be needed by the SSBL (see Boot chains overview), to supply the internal ADC[1] for example.

3.2 Runtime

3.2.1 Overview

The VREFBUF can be allocated to the Arm® Cortex®-A7 non-secure to be used under Linux® with regulator framework[3].

The VREFBUF is a system resource[4] which needs to be also controlled by the resource manager[4] in case its consumers (e.g. ADC[1], DAC[2] or an external device connected to VREF+ pin) are spread across:

- the Arm® Cortex®-A7 non-secure context
- the Arm® Cortex®-M4 context

For this reason, the direct control of VREFBUF from the Arm® Cortex®-M4 is not recommended in STM32Cube[5] by default.
It's recommended to implement it in STM32Cube **only if** all consumers and the VDDA supply pin are controlled in the Arm® Cortex®-M4 context.

The Peripheral assignment chapter describes which peripheral instance can be assigned to which context.

3.2.2 Software frameworks

Do	Peri	Software frameworks			Comment
mai n Cor tex -A7 sec ure (O P- TE E)	Cor tex -A7 no n- sec ure (Li nux)	Cortex-M4 (STM32Cube)			
An alo	V R EF B		Linux regulator		

Domain	Peripheral	Software frameworks		Comment
		framework		

3.2.3 Peripheral configuration

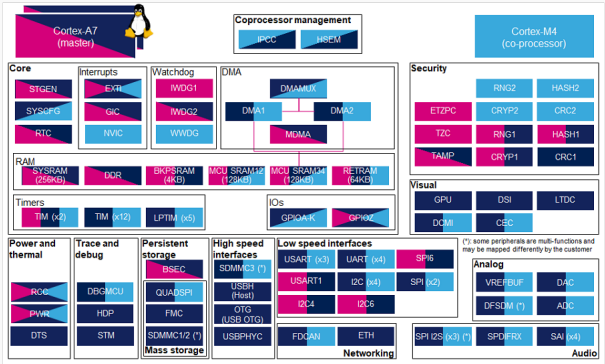
The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration by itself can be performed via the STM32CubeMX tool for all internal peripherals. It can then be manually completed (especially for external peripherals) according to the information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

- For the Linux kernel configuration, please refer to device internal regulator. An example can be found also in ADC DT configuration example
- In case the control of VREFBUF consumers are spread across the various cores, see also Resource manager for coprocessing

3.2.4 Peripheral assignment

Check boxes illustrate the possible peripheral allocations supported by STM32 MPU Embedded Software:

- means that the peripheral can be assigned () to the given runtime context.
- is used for system peripherals that cannot be unchecked because they are statically connected in the device.



Internal peripherals

Refer to How to assign an internal peripheral to a runtime context for more information on how to assign peripherals manually or via STM32CubeMX.
The present chapter describes STMicroelectronics recommendations or choice of implementation. Additional possiblities might be described in STM32MP15 reference manuals.

Domain	Peripheral	Runtime allocation				Comment
Instance	Cortex-A7 secure (OP-TEE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)			
	V					

Domain	Peripheral	Runtime allocation				Comment
Analog	VREFBUF	VREFBUF				Assignment (single choice)

4 References

- 1.01.11.2 ADC internal peripheral
- 2.02.1 DAC internal peripheral
- Regulator overview, Linux® regulator framework overview
- 4.04.1 Resource manager for coprocessing, focus on system resources
- STM32CubeMP1 architecture

voltage reference buffer (STM32 specific)

Analog-to-digital converter. The process of converting a sampled analog signal to a digital code that represents the amplitude of the original signal sample.

Digital-to-analog converter (Electronic circuit that converts a binary number into a continuously varying value.)

Second Stage Boot Loader

Open Portable Trusted Execution Environment

VREFBUF internal peripheral

Stable: 12.03.2020 - 13:05 / Revision: 12.02.2020 - 16:31

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

1 Article purpose

The purpose of this article is to

- briefly introduce the VREFBUF peripheral 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 VREFBUF peripheral.

2 Peripheral overview

The **VREFBUF** peripheral is an internal voltage regulator.

2.1 Features

The VREFBUF is supplied via the VDDA pin. When enabled, it can provide a reference voltage in the range of: 1,5V, 1,8V, 2,048V or 2,5V.

The VREFBUF can be used to provide an analog voltage reference for:

- ADC internal peripheral[1]
- DAC internal peripheral[2]
- External components through the dedicated VREF+ pin.

The VREFBUF can be left unused. In this case, an external voltage regulator can provide reference voltage to VREF+ pin.

Refer to the 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 VREFBUF is a **non-secure** peripheral.

3 Peripheral usage and associated software

3.1 Boot time

The VREFBUF is usually not used at boot time. But it may be needed by the SSBL (see Boot chains overview), to supply the internal ADC[1] for example.

3.2 Runtime

3.2.1 Overview

The VREFBUF can be allocated to the Arm® Cortex®-A7 non-secure to be used under Linux® with regulator framework[3].

The VREFBUF is a system resource[4] which needs to be also controlled by the resource manager[4] in case its consumers (e.g. ADC[1], DAC[2] or an external device connected to VREF+ pin) are spread across:

- the Arm® Cortex®-A7 non-secure context
- the Arm® Cortex®-M4 context

For this reason, the direct control of VREFBUF from the Arm® Cortex®-M4 is not recommended in STM32Cube[5] by default.
It's recommended to implement it in STM32Cube **only if** all consumers and the VDDA supply pin are controlled in the Arm® Cortex®-M4 context.

The Peripheral assignment chapter describes which peripheral instance can be assigned to which context.

3.2.2 Software frameworks

Do	Peri	Software frameworks			Comment
mai n Cor tex -A7 sec ure (O P- TE E)	Cor tex -A7 no n- sec ure (Li nux)	Cortex-M4 (STM32Cube)			
An	V R				

Domain	Peripheral	Software frameworks		Comment
ano g	EP B UF		Linux regulator framework	

3.2.3 Peripheral configuration

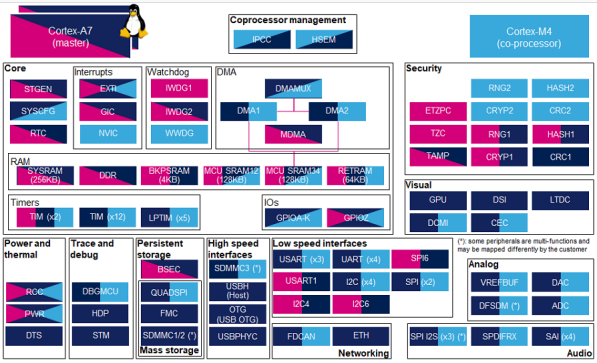
The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration by itself can be performed via the STM32CubeMX tool for all internal peripherals. It can then be manually completed (especially for external peripherals) according to the information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

- For the Linux kernel configuration, please refer to device internal regulator. An example can be found also in ADC DT configuration example
- In case the control of VREFBUF consumers are spread across the various cores, see also Resource manager for coprocessing

3.2.4 Peripheral assignment

Check boxes illustrate the possible peripheral allocations supported by STM32 MPU Embedded Software:

- means that the peripheral can be assigned () to the given runtime context.
- is used for system peripherals that cannot be unchecked because they are statically connected in the device.



Internal peripherals

Refer to How to assign an internal peripheral to a runtime context for more information on how to assign peripherals manually or via STM32CubeMX.
The present chapter describes STMicroelectronics recommendations or choice of implementation. Additional possiblities might be described in STM32MP15 reference manuals.

Domain	Peripheral	Runtime allocation		Comment
Instance	Cortex-A7 secure (	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)	

Domain	Peripheral	Runtime allocation				Comment
	OP-TEE)					
Analog	VREFBUF	VREFBUF				Assignment (single choice)

4 References

- 1.01.11.2 ADC internal peripheral
- 2.02.1 DAC internal peripheral
- Regulator overview, Linux® regulator framework overview
- 4.04.1 Resource manager for coprocessing, focus on system resources
- STM32CubeMP1 architecture

voltage reference buffer (STM32 specific)

Analog-to-digital converter. The process of converting a sampled analog signal to a digital code that represents the amplitude of the original signal sample.

Digital-to-analog converter (Electronic circuit that converts a binary number into a continuously varying value.)

Second Stage Boot Loader

Open Portable Trusted Execution Environment

VREFBUF internal peripheral

Stable: 12.02.2020 - 16:42 / Revision: 12.02.2020 - 16:41

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

1 Article purpose

The purpose of this article is to

- briefly introduce the VREFBUF peripheral 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 VREFBUF peripheral.

2 Peripheral overview

The **VREFBUF** peripheral is an internal voltage regulator.

2.1 Features

The VREFBUF is supplied via the VDDA pin. When enabled, it can provide a reference voltage in the range of: 1,5V, 1,8V, 2,048V or 2,5V.

The VREFBUF can be used to provide an analog voltage reference for:

- ADC internal peripheral[1]
- DAC internal peripheral[2]
- External components through the dedicated VREF+ pin.

The VREFBUF can be left unused. In this case, an external voltage regulator can provide reference voltage to VREF+ pin.

Refer to the 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 VREFBUF is a **non-secure** peripheral.

3 Peripheral usage and associated software

3.1 Boot time

The VREFBUF is usually not used at boot time. But it may be needed by the SSBL (see Boot chains overview), to supply the internal ADC[1] for example.

3.2 Runtime

3.2.1 Overview

The VREFBUF can be allocated to the Arm® Cortex®-A7 non-secure to be used under Linux® with regulator framework[3].

> The VREFBUF is a system resource[4] which needs to be also controlled by the resource manager[4] in case its consumers (e.g. ADC[1], DAC[2] or an external device connected to VREF+ pin) are spread across:
>
> - the Arm® Cortex®-A7 non-secure context
> - the Arm® Cortex®-M4 context
>
> For this reason, the direct control of VREFBUF from the Arm® Cortex®-M4 is not recommended in STM32Cube[5] by default.
> It's recommended to implement it in STM32Cube **only if** all consumers and the VDDA supply pin are controlled in the Arm® Cortex®-M4 context.

The Peripheral assignment chapter describes which peripheral instance can be assigned to which context.

3.2.2 Software frameworks

Do	Peri	Software frameworks			Comment
mai n Cor tex -A7 sec ure (O P- TE E)	Cor tex -A7 no n- sec ure (Li nux)	Cortex-M4 (STM32Cube)			
An alo	V R EF B		Linux regulator		

Domain	Peripheral	Software frameworks		Comment
	VREF		framework	

3.2.3 Peripheral configuration

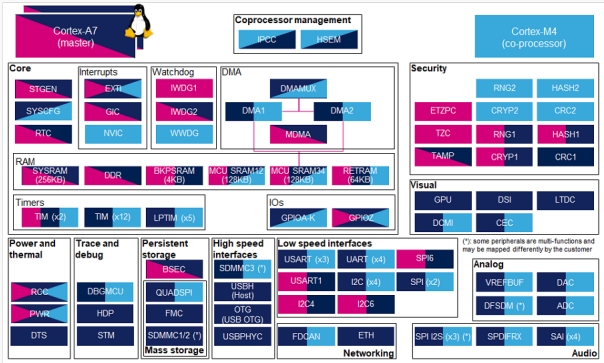
The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration by itself can be performed via the STM32CubeMX tool for all internal peripherals. It can then be manually completed (especially for external peripherals) according to the information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

- For the Linux kernel configuration, please refer to device internal regulator. An example can be found also in ADC DT configuration example
- In case the control of VREFBUF consumers are spread across the various cores, see also Resource manager for coprocessing

3.2.4 Peripheral assignment

Check boxes illustrate the possible peripheral allocations supported by STM32 MPU Embedded Software:

- means that the peripheral can be assigned () to the given runtime context.
- is used for system peripherals that cannot be unchecked because they are statically connected in the device.



Internal peripherals

Refer to How to assign an internal peripheral to a runtime context for more information on how to assign peripherals manually or via STM32CubeMX.
The present chapter describes STMicroelectronics recommendations or choice of implementation. Additional possiblities might be described in STM32MP15 reference manuals.

Domain	Peripheral	Runtime allocation				Comment
Instance	Cortex-A7 secure (OP-TEE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)			
	V					

Domain	Peripheral	Runtime allocation				Comment
Analog	VREFBUF	VREFBUF				Assignment (single choice)

4 References

- 1.01.11.2 ADC internal peripheral
- 2.02.1 DAC internal peripheral
- Regulator overview, Linux® regulator framework overview
- 4.04.1 Resource manager for coprocessing, focus on system resources
- STM32CubeMP1 architecture

voltage reference buffer (STM32 specific)

Analog-to-digital converter. The process of converting a sampled analog signal to a digital code that represents the amplitude of the original signal sample.

Digital-to-analog converter (Electronic circuit that converts a binary number into a continuously varying value.)

Second Stage Boot Loader

Open Portable Trusted Execution Environment

VREFBUF internal peripheral

Stable: 21.02.2020 - 08:39 / Revision: 04.02.2020 - 15:22

Template:ArticleMainWriter Template:ArticleApprovedVersion

Contents

1 Article purpose

The purpose of this article is to

- briefly introduce the VREFBUF peripheral 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 VREFBUF peripheral.

2 Peripheral overview

The **VREFBUF** peripheral is an internal voltage regulator.

2.1 Features

The VREFBUF is supplied via the VDDA pin. When enabled, it can provide a reference voltage in the range of: 1,5V, 1,8V, 2,048V or 2,5V.

The VREFBUF can be used to provide an analog voltage reference for:

- ADC internal peripheral[1]
- DAC internal peripheral[2]
- External components through the dedicated VREF+ pin.

The VREFBUF can be left unused. In this case, an external voltage regulator can provide reference voltage to VREF+ pin.

Refer to the 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 VREFBUF is a **non-secure** peripheral.

3 Peripheral usage and associated software

3.1 Boot time

The VREFBUF is usually not used at boot time. But it may be needed by the SSBL (see Boot chains overview), to supply the internal ADC[1] for example.

3.2 Runtime

3.2.1 Overview

The VREFBUF can be allocated to the Arm® Cortex®-A7 non-secure to be used under Linux® with regulator framework[3].

The VREFBUF is a system resource[4] which needs to be also controlled by the resource manager[4] in case its consumers (e.g. ADC[1], DAC[2] or an external device connected to VREF+ pin) are spread across:

- the Arm® Cortex®-A7 non-secure context
- the Arm® Cortex®-M4 context

For this reason, the direct control of VREFBUF from the Arm® Cortex®-M4 is not recommended in STM32Cube[5] by default.
It's recommended to implement it in STM32Cube **only if** all consumers and the VDDA supply pin are controlled in the Arm® Cortex®-M4 context.

The Peripheral assignment chapter describes which peripheral instance can be assigned to which context.

3.2.2 Software frameworks

Do	Peri	Software frameworks			Comment
mai n Cor tex -A7 sec ure (O P- TE E)	Cor tex -A7 no n- sec ure (Li nux)	Cortex-M4 (STM32Cube)			
An	V R				

Domain	Peripheral	Software frameworks			Comment
ano g	EP B UF		Linux regulator framework		

3.2.3 Peripheral configuration

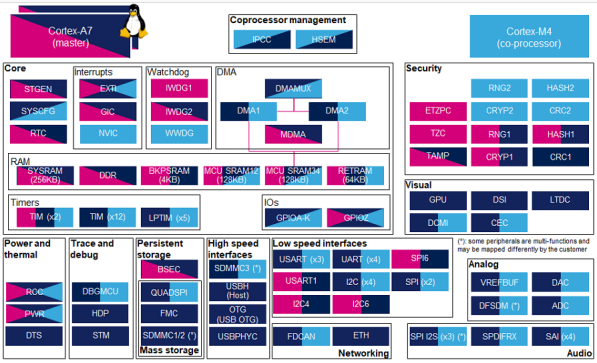
The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration by itself can be performed via the STM32CubeMX tool for all internal peripherals. It can then be manually completed (especially for external peripherals) according to the information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

- For the Linux kernel configuration, please refer to device internal regulator. An example can be found also in ADC DT configuration example
- In case the control of VREFBUF consumers are spread across the various cores, see also Resource manager for coprocessing

3.2.4 Peripheral assignment

Check boxes illustrate the possible peripheral allocations supported by STM32 MPU Embedded Software:

- means that the peripheral can be assigned () to the given runtime context.
- is used for system peripherals that cannot be unchecked because they are statically connected in the device.



Internal peripherals

Refer to How to assign an internal peripheral to a runtime context for more information on how to assign peripherals manually or via STM32CubeMX.
The present chapter describes STMicroelectronics recommendations or choice of implementation. Additional possiblities might be described in STM32MP15 reference manuals.

Domain	Peripheral	Runtime allocation		Comment
Instance	Cortex-A7 secure (	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)	

Domain	Peripheral	Runtime allocation				Comment
	(OP-TEE)					
Analog	VREFBUF	VREFBUF				Assignment (single choice)

4 References

- 1.01.11.2 ADC internal peripheral
- 2.02.1 DAC internal peripheral
- Regulator overview, Linux® regulator framework overview
- 4.04.1 Resource manager for coprocessing, focus on system resources
- STM32CubeMP1 architecture

voltage reference buffer (STM32 specific)

Analog-to-digital converter. The process of converting a sampled analog signal to a digital code that represents the amplitude of the original signal sample.

Digital-to-analog converter (Electronic circuit that converts a binary number into a continuously varying value.)

Second Stage Boot Loader

Open Portable Trusted Execution Environment