

VREFBUF internal peripheral



Contents

1. VREFBUF internal peripheral	3
2. STM32MP15 resources	6
3. Boot chains overview	10
4. Regulator overview	14
5. STM32CubeMX	18
6. ADC device tree configuration	22
7. Resource manager for coprocessing	26
8. STM32MPU Embedded Software architecture overview	30
9. How to assign an internal peripheral to a runtime context	34
10. ADC internal peripheral	38
11. DAC internal peripheral	42
12. STM32CubeMP1 architecture	46



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	3
2 Peripheral overview	3
2.1 Features	4
2.2 Security support	4
3 Peripheral usage and associated software	4
3.1 Boot time	4
3.2 Runtime	4
3.2.1 Overview	4
3.2.2 Software frameworks	5
3.2.3 Peripheral configuration	5
3.2.4 Peripheral assignment	5
4 References	6

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
main Cortex-A7 secure (OPTEE)	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](#)

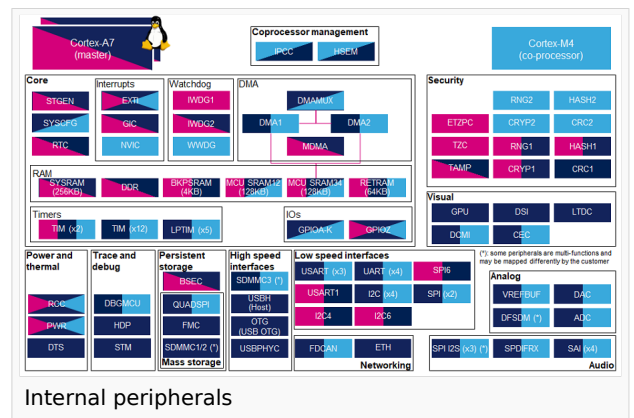
3.2.4 Peripheral assignment

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.

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](#).





Do	Per	Runtime allocation				Comme
ma	iph					nt
in	era					
	or					
	te					
	x-					
	A					
	7					
In	se	Cortex-A7	Cortex-M4			
st	cu	non-secure	(STM32Cube)			
a	re	(Linux)				
nc	(
e	O					
	P-					
	T					
	E					
	E)					
A	V					Assig
n	R					nment
al	E					(singl
o	F	VREFBUF				e
g	B					choice
	U)
	F					

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	7
2 Peripheral overview	7
2.1 Features	7
2.2 Security support	8
3 Peripheral usage and associated software	8
3.1 Boot time	8
3.2 Runtime	8
3.2.1 Overview	8
3.2.2 Software frameworks	8
3.2.3 Peripheral configuration	9
3.2.4 Peripheral assignment	9
4 References	10

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	al		

Do	Peri	Software frameworks			Comment
main	Cortex-A7	Cortex-M4 (STM32Cube)			
	Cortex-A7				
Secure	no-secure				
OS	(Linux)				
Application	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](#)

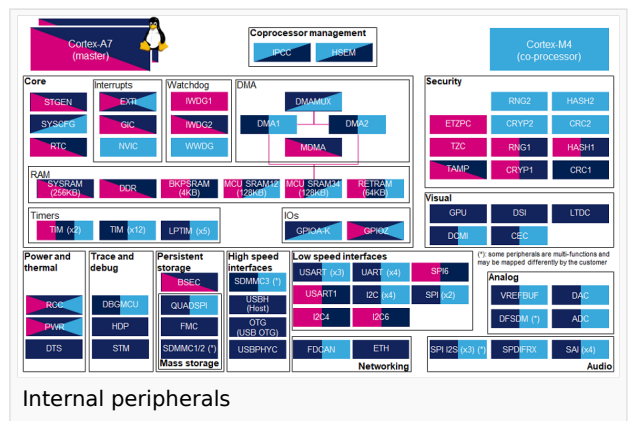
3.2.4 Peripheral assignment

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.

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](#).



Do	Peri	Runtime allocation			Comment
main	Cortex-A7				



Do	Per	Runtime allocation				Comme
ma in in st a nc e	iph era A 7 se cu re (O P T E E)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)			
A n a l o g	V R E F B U F	VREFBUF				Assig nment (singl e choic e)

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	11
2 Peripheral overview	11
2.1 Features	11
2.2 Security support	12
3 Peripheral usage and associated software	12
3.1 Boot time	12
3.2 Runtime	12
3.2.1 Overview	12
3.2.2 Software frameworks	12
3.2.3 Peripheral configuration	13
3.2.4 Peripheral assignment	13
4 References	14

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
no	Cortex-A7	Cortex-M4	

Do	Peri	Software frameworks		Comment
main secure (O P- TE E)	n- sec ure (Li nux)	(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](#)

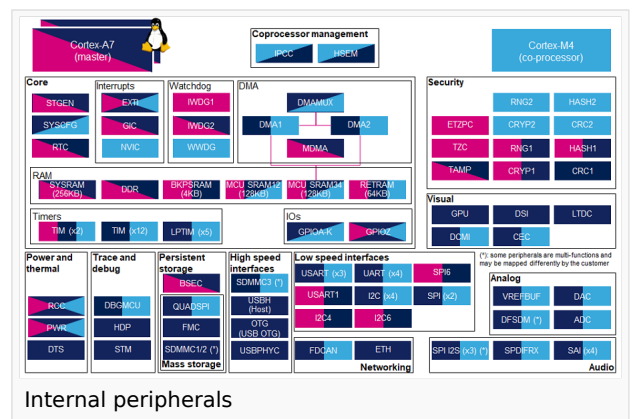
3.2.4 Peripheral assignment

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.

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](#).



Do	Peri	Runtime allocation		Comment
main secure (O P- TE E)	n- sec ure (Li nux)	(STM32Cube)		
Analog	VREFBUF		Linux regulator framework	



Do ma in st a nc e	Per in te r f a c e	Runtime allocation				Comme nt
		Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)			
A n a l o g	V R E F B U F	VREFBUF				Assig nment (singl e 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	15
2 Peripheral overview	15
2.1 Features	15
2.2 Security support	16
3 Peripheral usage and associated software	16
3.1 Boot time	16
3.2 Runtime	16
3.2.1 Overview	16
3.2.2 Software frameworks	16
3.2.3 Peripheral configuration	17
3.2.4 Peripheral assignment	17
4 References	18

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
main Cortex -A7 non-secure (OPE) (E)	Cortex -A7 non-secure (Linux)	Cortex-M4 (STM32Cube)	

Do main thread allocation	Peripheral	Software frameworks		Comment
	EFBUF		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](#)

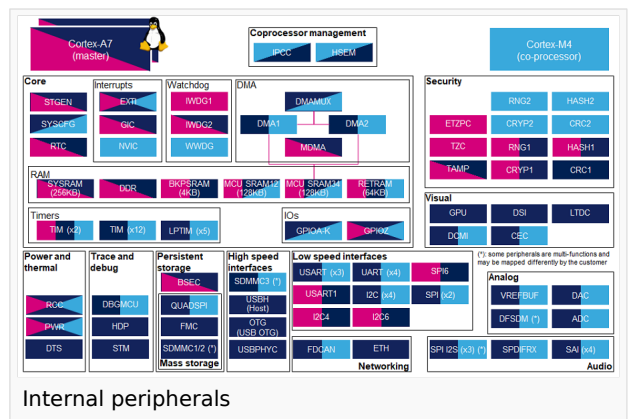
3.2.4 Peripheral assignment

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.

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](#).



Do main thread allocation	Peripheral	Runtime allocation		Comment
	Cortex-A7 secure (Linux)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)	



Do ma in	Per iph era l (T E E)	Runtime allocation				Comme nt
A n a l o g	V R E F B U F	VREFBUF				Assig nment (singl e 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	19
2 Peripheral overview	19
2.1 Features	19



2.2 Security support	19
3 Peripheral usage and associated software	20
3.1 Boot time	20
3.2 Runtime	20
3.2.1 Overview	20
3.2.2 Software frameworks	20
3.2.3 Peripheral configuration	21
3.2.4 Peripheral assignment	21
4 References	22

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 Cortex -A7 non-secure (OPE) TE E)	Cor tex -A7 no n- sec ure (Li nux)	Cortex-M4 (STM32Cube)			
An alo	V R E F B U F		Linux regulator		

Do	Peri	Software frameworks			Comment
main	internal		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](#)

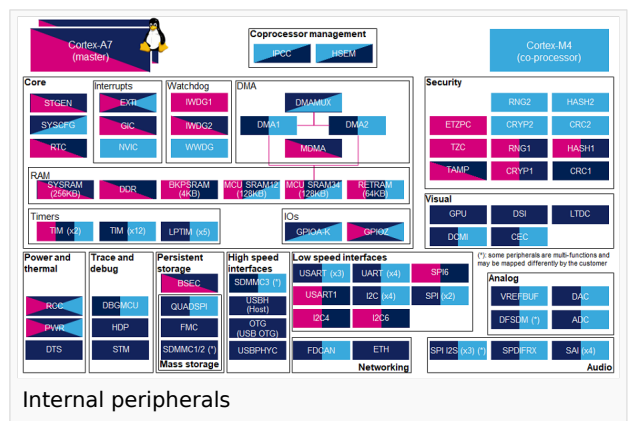
3.2.4 Peripheral assignment

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.

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](#).



Do	Peri	Runtime allocation			Comme
main	internal				
in	Core				
st	7				
a	se				
nc	cu				
e	re				
	(
	O				
	P				
	T				
	E				
	E)				
	V				



Do	Per	Runtime allocation				Comme
ma	iph	VREFBUF				nt
in	era					Assig
al	I					nment
o	B					(singl
g	U					e
	F					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	23
2 Peripheral overview	23
2.1 Features	23
2.2 Security support	23
3 Peripheral usage and associated software	24
3.1 Boot time	24
3.2 Runtime	24
3.2.1 Overview	24
3.2.2 Software frameworks	24
3.2.3 Peripheral configuration	25



3.2.4 Peripheral assignment	25
4 References	26

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 tex -A7 sec ure (O P- TE E)	Cor tex -A7 no n- sec ure (Li nux)	Cortex-M4 (STM32Cube)	
An	V R		

Do	Peri	Software frameworks		Comment
main	Peripheral		Linux regulator framework	
	UF			

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](#)

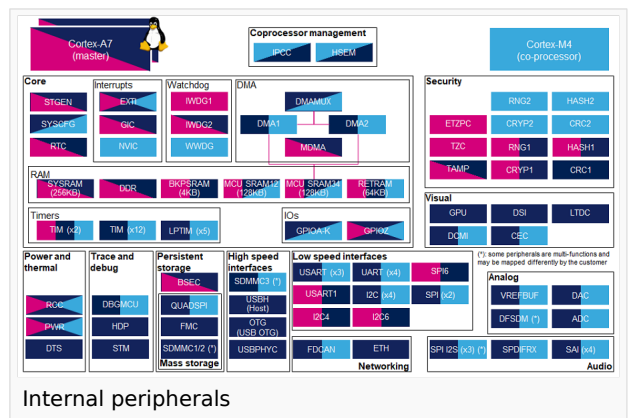
3.2.4 Peripheral assignment

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.

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](#).



Do	Peri	Runtime allocation		Comment
main	Peripheral			
in	Core			
st	Context			
a	Secure			
nc	(Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)	
e)			



Do ma in	Per iph era l (T E E)	Runtime allocation				Comme nt
A n a l o g	V R E F B U F	VREFBUF				Assig nment (singl e 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	27
2 Peripheral overview	27
2.1 Features	27



2.2 Security support	27
3 Peripheral usage and associated software	28
3.1 Boot time	28
3.2 Runtime	28
3.2.1 Overview	28
3.2.2 Software frameworks	28
3.2.3 Peripheral configuration	29
3.2.4 Peripheral assignment	29
4 References	30

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 Cortex -A7 non-secure (OTE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)			
Analo	VREFB		Linux regulator		

Do	Peri	Software frameworks			Comment
main	internal		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](#)

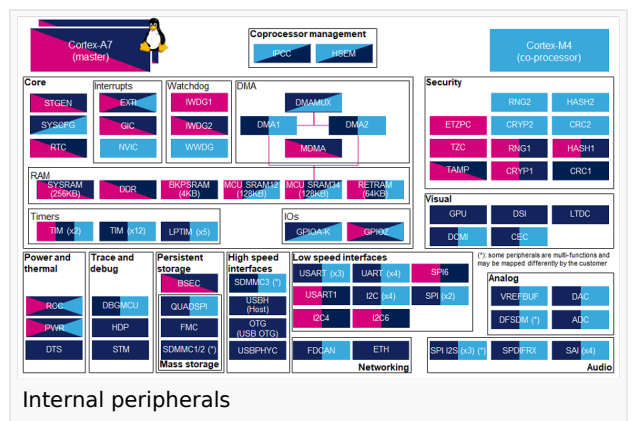
3.2.4 Peripheral assignment

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.

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](#).



Do	Peri	Runtime allocation			Comme
main	internal				
in	internal				
st	internal				
se	internal				
cu	internal				
re	internal				
(internal				
O	internal				
P	internal				
T	internal				
E	internal				
)	internal				
V	internal				



Do	Per	Runtime allocation				Comme
ma	iph	VREFBUF				nt
in	era					Assig
al	I					nment
o	B					(singl
g	U					e
	F					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	31
2 Peripheral overview	31
2.1 Features	31
2.2 Security support	31
3 Peripheral usage and associated software	32
3.1 Boot time	32
3.2 Runtime	32
3.2.1 Overview	32
3.2.2 Software frameworks	32
3.2.3 Peripheral configuration	33



3.2.4 Peripheral assignment	33
4 References	34

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 tex -A7 sec ure (O P- TE E)	Cor tex -A7 no n- sec ure (Li nux)	Cortex-M4 (STM32Cube)	
An	V R		

Do	Peri	Software frameworks		Comment
main	Peripheral		Linux regulator framework	
	UF			

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](#)

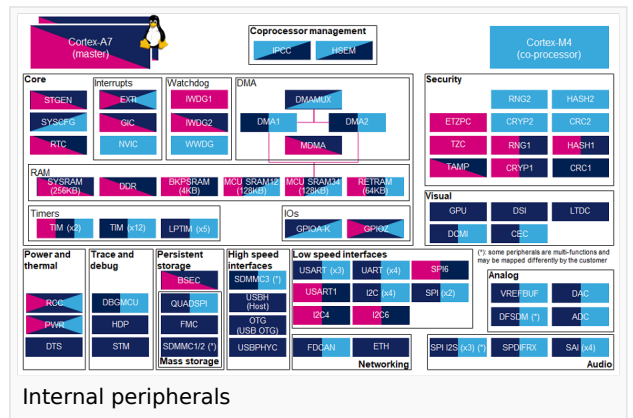
3.2.4 Peripheral assignment

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.

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](#).



Do	Peri	Runtime allocation		Comment
main	Peripheral			
in	Core			
st	Context			
a	Secure			
nc	(
e)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)	



Do ma in	Per iph era l (T E E)	Runtime allocation				Comme nt
A n a l o g	V R E F B U F	VREFBUF				Assig nment (singl e 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	35
2 Peripheral overview	35
2.1 Features	35



2.2 Security support	35
3 Peripheral usage and associated software	36
3.1 Boot time	36
3.2 Runtime	36
3.2.1 Overview	36
3.2.2 Software frameworks	36
3.2.3 Peripheral configuration	37
3.2.4 Peripheral assignment	37
4 References	38

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
main Cortex -A7 non-secure (OTE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)			
Analogue	VREFBUF		Linux regulator		

Do	Peri	Software frameworks			Comment
main	internal		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](#)

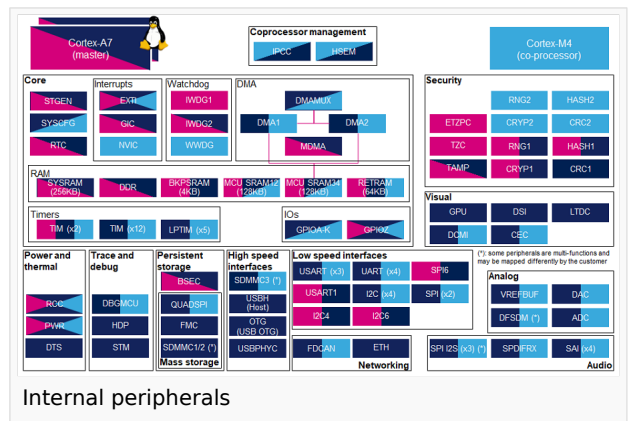
3.2.4 Peripheral assignment

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.

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](#).



Do	Peri	Runtime allocation			Comme
main	internal				
in	internal				
inst	internal				
anc	internal				
nc	internal				
e	internal				
	V				



Do	Per	Runtime allocation				Comme
ma	iph	VREFBUF				nt
in	era					Assig
al	I					nment
o	B					(singl
g	U					e
	F					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	39
2 Peripheral overview	39
2.1 Features	39
2.2 Security support	39
3 Peripheral usage and associated software	40
3.1 Boot time	40
3.2 Runtime	40
3.2.1 Overview	40
3.2.2 Software frameworks	40
3.2.3 Peripheral configuration	41



3.2.4 Peripheral assignment	41
4 References	42

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 tex -A7 sec ure (O P- TE E)	Cor tex -A7 no n- sec ure (Li nux)	Cortex-M4 (STM32Cube)	
An	V R		

Do	Peri	Software frameworks		Comment
main	Peripheral		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](#)

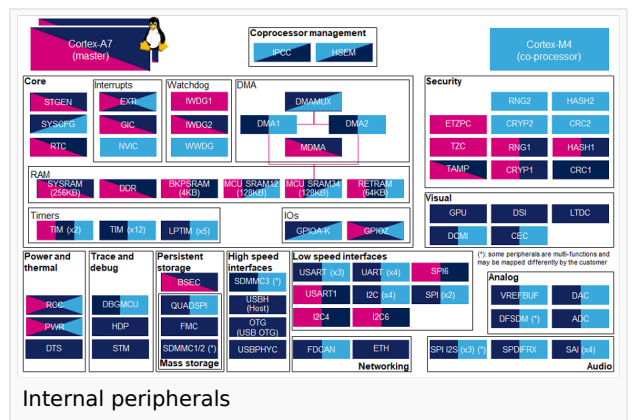
3.2.4 Peripheral assignment

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.

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](#).



Do	Peri	Runtime allocation		Comment
main	Peripheral	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)	



Do ma in	Per iph era l (T E E)	Runtime allocation				Comme nt
A n a l o g	V R E F B U F	VREFBUF				Assig nment (singl e choic e)

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	43
2 Peripheral overview	43
2.1 Features	43



2.2 Security support	43
3 Peripheral usage and associated software	44
3.1 Boot time	44
3.2 Runtime	44
3.2.1 Overview	44
3.2.2 Software frameworks	44
3.2.3 Peripheral configuration	45
3.2.4 Peripheral assignment	45
4 References	46

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 Cortex -A7 non-secure (OTE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)			
Analo	VREFB		Linux regulator		

Do	Peri	Software frameworks			Comment
main	internal		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](#)

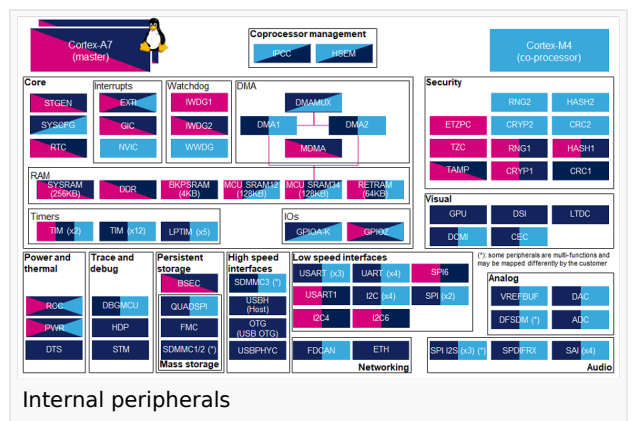
3.2.4 Peripheral assignment

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.

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](#).



Do	Peri	Runtime allocation			Comme
main	internal				
in	internal				
st	internal				
se	internal				
cu	internal				
re	internal				
(internal				
O	internal				
P	internal				
T	internal				
E	internal				
E)	internal				
V	internal				



Do	Per	Runtime allocation				Comme
ma	iph	VREFBUF				nt
in	era					Assig
al	I					nment
o	B					(singl
g	U					e
	F					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	47
2 Peripheral overview	47
2.1 Features	47
2.2 Security support	47
3 Peripheral usage and associated software	48
3.1 Boot time	48
3.2 Runtime	48
3.2.1 Overview	48
3.2.2 Software frameworks	48
3.2.3 Peripheral configuration	49



3.2.4 Peripheral assignment	49
4 References	50

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 tex -A7 sec ure (O P- TE E)	Cor tex -A7 no n- sec ure (Li nux)	Cortex-M4 (STM32Cube)	
An	V R		

Do	Peri	Software frameworks		Comment
main	Peripheral		Linux regulator framework	
	UF			

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](#)

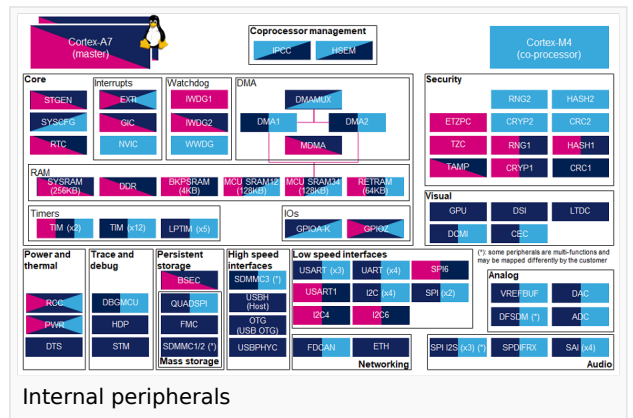
3.2.4 Peripheral assignment

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.

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](#).



Do	Peri	Runtime allocation		Comment
main	Peripheral			
in	Core			
st	Core			
a	Core			
nc	Core			
e	Core			
	Core	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)	



Do ma in	Per iph era l (T E E)	Runtime allocation				Comme nt
A n a l o g	V R E F B U F	VREFBUF				Assig nment (singl e choic e)

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