I2C internal peripheral

Contents

Stable: 25.09.2020 - 09:43 / Revision: 25.09.2020 - 09:38

Stable: 28.06.2020 - 09:43 / Revision: 28.06.2020 - 09:58

A quality version of this page, approved on *11 February 2019*, was based off this revision.

Template:ArticleMainWriter

Template:ArticleApprovedVersion

Contents

1 Article purpose

The purpose of this article is to:

- briefly introduce the I2C 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 necessary, how to configure the I2C peripheral.

2 Peripheral overview

The I2C bus interface serves as an interface between the microcontroller and the serial I2C bus.
It provides multi-master capability, and controls all I2C bus-specific sequencing, protocol, arbitration and timing.
The I2C controller allows to be a slave as well if need be.
It is also SMBus 2.0 compatible.
For more information about I2C please refer to this link: **I2C wikipedia**[1] or **i2c-bus.org**[2]
For more information about SMBus please refer to this link: **SMBus wikipedia**[3] or **i2c-bus.org**[4]

2.1 Features

Here are the main features:

- Multi-master
- Standard (100 KHz) and fast speed modes (400 KHz and Plus 1 MHz)
- I2C 10-bit address
- I2C slave capabilities (programmable I2C address)
- DMA capabilities
- SMBus 2.0 compatible
  - Standard bus protocol (quick command; byte, word, block read/write)
  - Host notification
  - Alert

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

2.2 Security support

- There are six I2C instances.
  - I2C instances 1, 2, 3 and 5 are **non-secure**.
  - I2C instances 4 and 6 can be **secure** (under ETZPC control).

3 Peripheral usage and associated software

3.1 Boot time

The I2C peripheral is usually not used at boot time. But it may be used by the SSBL and/or FSBL (see Boot chains overview), for example, to configure a PMIC (if any), or to access data stored in an external EEPROM.

3.2 Runtime

3.2.1 Overview

I2C4&6 instances can be allocated to:

- the Arm® Cortex®-A7 secure core to be controlled in OP-TEE by the OP-TEE I2C driver

All I2C instances can be allocated to:

- the Arm® Cortex®-A7 non-secure core to be controlled in Linux® by the I2C framework

All but I2C4&6 instances can be allocated to:

- the Arm® Cortex®-M4 to be controlled in STM32Cube MPU Package by STM32Cube I2C driver

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

3.2.2 Software frameworks

Domain	Peripheral	Software components			Comment
OP-TEE	Linux	STM32Cube			
Low speed interface	I2C	OP-TEE I2C driver	I2C Engine framework	STM32Cube I2C driver	

3.2.3 Peripheral configuration

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

For Linux® kernel configuration, please refer to I2C configuration.

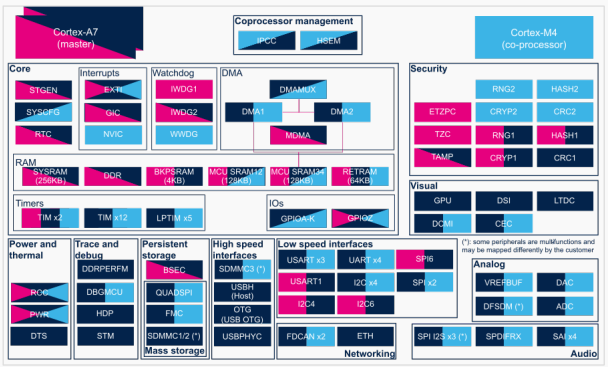
Please refer to I2C device tree configuration for detailed information on how to configure I2C peripherals.

3.2.4 Peripheral assignment

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

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

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



Internal peripherals

Domain	Periphera l	Runtime allocation				Comment
Instance	Cortex-A7 secure (OP-TEE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)			
Low speed interface	I2C	I2C1				Assignment (single choice)
		I2C2				Assignment (single choice)
		I2C3				Assignment (single choice)
		I2C4				Assignment (single choice). Used for PMIC control on ST boards .
		I2C5				Assignment (single choice)
		I2C6				Assignment (single choice)

4 References

- http://en.wikipedia.org/wiki/I2C
- https://www.i2c-bus.org/specification/
- https://en.wikipedia.org/wiki/System_Management_Bus
- https://www.i2c-bus.org/smbus/

Stable: 25.09.2020 - 08:36 / Revision: 25.09.2020 - 08:35

Template:ArticleMainWriter

Template:ArticleApprovedVersion

Contents

1 Article purpose

The purpose of this article is to:

- briefly introduce the I2C 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 necessary, how to configure the I2C peripheral.

2 Peripheral overview

The I2C bus interface serves as an interface between the microcontroller and the serial I2C bus.
It provides multi-master capability, and controls all I2C bus-specific sequencing, protocol, arbitration and timing.
The I2C controller allows to be a slave as well if need be.
It is also SMBus 2.0 compatible.
For more information about I2C please refer to this link: **I2C wikipedia**[1] or **i2c-bus.org**[2]
For more information about SMBus please refer to this link: **SMBus wikipedia**[3] or **i2c-bus.org**[4]

2.1 Features

Here are the main features:

- Multi-master
- Standard (100 KHz) and fast speed modes (400 KHz and Plus 1 MHz)
- I2C 10-bit address
- I2C slave capabilities (programmable I2C address)
- DMA capabilities
- SMBus 2.0 compatible
  - Standard bus protocol (quick command; byte, word, block read/write)
  - Host notification
  - Alert

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

2.2 Security support

- There are six I2C instances.
  - I2C instances 1, 2, 3 and 5 are **non-secure**.
  - I2C instances 4 and 6 can be **secure** (under ETZPC control).

3 Peripheral usage and associated software

3.1 Boot time

The I2C peripheral is usually not used at boot time. But it may be used by the SSBL and/or FSBL (see Boot chains overview), for example, to configure a PMIC (if any), or to access data stored in an external EEPROM.

3.2 Runtime

3.2.1 Overview

I2C4&6 instances can be allocated to:

- the Arm® Cortex®-A7 secure core to be controlled in OP-TEE by the OP-TEE I2C driver

All I2C instances can be allocated to:

- the Arm® Cortex®-A7 non-secure core to be controlled in Linux® by the I2C framework

All but I2C4&6 instances can be allocated to:

- the Arm® Cortex®-M4 to be controlled in STM32Cube MPU Package by STM32Cube I2C driver

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

3.2.2 Software frameworks

Domain	Peripheral	Software components			Comment
OP-TEE	Linux	STM32Cube			
Low speed interface	I2C	OP-TEE I2C driver	I2C Engine framework	STM32Cube I2C driver	

3.2.3 Peripheral configuration

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

For Linux® kernel configuration, please refer to I2C configuration.

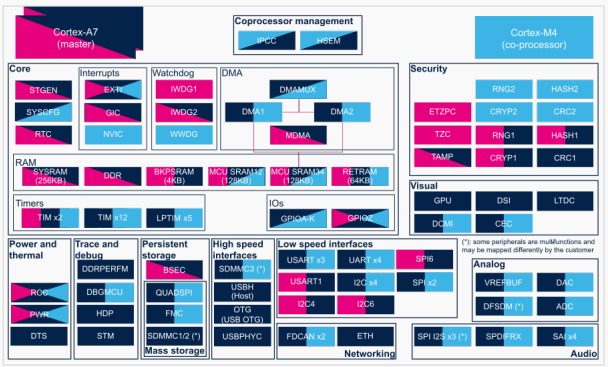
Please refer to I2C device tree configuration for detailed information on how to configure I2C peripherals.

3.2.4 Peripheral assignment

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

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

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



Internal peripherals

Domain	Periphera l	Runtime allocation				Comment
Instance	Cortex-A7 secure (OP-TEE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)			
Low speed interface	I2C	I2C1				Assignment (single choice)
		I2C2				Assignment (single choice)
		I2C3				Assignment (single choice)
		I2C4				Assignment (single choice). Used for PMIC control on ST boards .
		I2C5				Assignment (single choice)
		I2C6				Assignment (single choice)

4 References

- http://en.wikipedia.org/wiki/I2C
- https://www.i2c-bus.org/specification/
- https://en.wikipedia.org/wiki/System_Management_Bus
- https://www.i2c-bus.org/smbus/

Stable: 17.06.2020 - 15:26 / Revision: 16.01.2020 - 09:30

Template:ArticleMainWriter

Template:ArticleApprovedVersion

Contents

1 Article purpose

The purpose of this article is to:

- briefly introduce the I2C 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 necessary, how to configure the I2C peripheral.

2 Peripheral overview

The I2C bus interface serves as an interface between the microcontroller and the serial I2C bus.
It provides multi-master capability, and controls all I2C bus-specific sequencing, protocol, arbitration and timing.
The I2C controller allows to be a slave as well if need be.
It is also SMBus 2.0 compatible.

For more information about I2C please refer to this link: **I2C wikipedia**[1] or **i2c-bus.org**[2]

For more information about SMBus please refer to this link: **SMBus wikipedia**[3] or **i2c-bus.org**[4]

2.1 Features

Here are the main features:

- Multi-master
- Standard (100 KHz) and fast speed modes (400 KHz and Plus 1 MHz)
- I2C 10-bit address
- I2C slave capabilities (programmable I2C address)
- DMA capabilities
- SMBus 2.0 compatible
  - Standard bus protocol (quick command; byte, word, block read/write)
  - Host notification
  - Alert

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

2.2 Security support

- There are six I2C instances.
  - I2C instances 1, 2, 3 and 5 are **non-secure**.
  - I2C instances 4 and 6 can be **secure** (under ETZPC control).

3 Peripheral usage and associated software

3.1 Boot time

The I2C peripheral is usually not used at boot time. But it may be used by the SSBL and/or FSBL (see Boot chains overview), for example, to configure a PMIC (if any), or to access data stored in an external EEPROM.

3.2 Runtime

3.2.1 Overview

I2C4&6 instances can be allocated to:

- the Arm® Cortex®-A7 secure core to be controlled in OP-TEE by the OP-TEE I2C driver

All I2C instances can be allocated to:

- the Arm® Cortex®-A7 non-secure core to be controlled in Linux® by the I2C framework

All but I2C4&6 instances can be allocated to:

- the Arm® Cortex®-M4 to be controlled in STM32Cube MPU Package by STM32Cube I2C driver

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

3.2.2 Software frameworks

Domain	Peripheral	Software components			Comment
OP-TEE	Linux	STM32Cube			
Low speed interface	I2C	OP-TEE I2C driver	I2C Engine framework	STM32Cube I2C driver	

3.2.3 Peripheral configuration

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

For Linux® kernel configuration, please refer to I2C configuration.

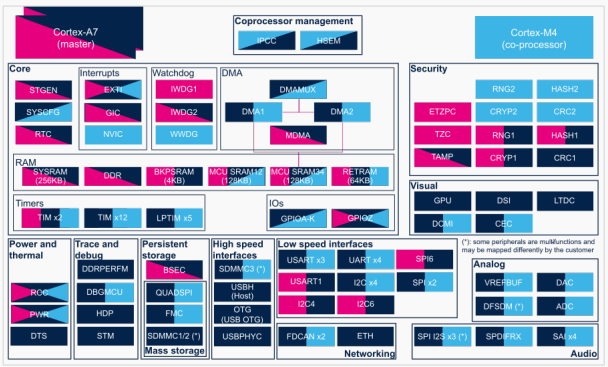
Please refer to I2C device tree configuration for detailed information on how to configure I2C peripherals.

3.2.4 Peripheral assignment

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

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

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



Internal peripherals

Domain	Periphera l	Runtime allocation				Comment
Instance	Cortex-A7 secure (OP-TEE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)			
Low speed interface	I2C	I2C1				Assignment (single choice)
		I2C2				Assignment (single choice)
		I2C3				Assignment (single choice)
		I2C4				Assignment (single choice). Used for PMIC control on ST boards .
		I2C5				Assignment (single choice)
		I2C6				Assignment (single choice)

4 References

- http://en.wikipedia.org/wiki/I2C
- https://www.i2c-bus.org/specification/
- https://en.wikipedia.org/wiki/System_Management_Bus
- https://www.i2c-bus.org/smbus/

Subcategories

This category has the following 2 subcategories, out of 2 total.

- STM32MP15 Discovery kits (6 P)
- STM32MP15 Evaluation boards (9 P)

Pages in category "Getting started with STM32MP1 boards"

The following 6 pages are in this category, out of 6 total.

- How to add a new NAND or SNAND flash in Yocto
- LEDs and buttons on STM32 MPU boards
- STM32MP1 Developer Package
- STM32MP1 Developer Package for Android
- STM32MP1 Distribution Package
- STM32MP1 Distribution Package for Android

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

Template:ArticleMainWriter

Template:ArticleApprovedVersion

Contents

1 Article purpose

The purpose of this article is to:

- briefly introduce the I2C 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 necessary, how to configure the I2C peripheral.

2 Peripheral overview

The I2C bus interface serves as an interface between the microcontroller and the serial I2C bus.
It provides multi-master capability, and controls all I2C bus-specific sequencing, protocol, arbitration and timing.
The I2C controller allows to be a slave as well if need be.
It is also SMBus 2.0 compatible.
For more information about I2C please refer to this link: **I2C wikipedia**[1] or **i2c-bus.org**[2]
For more information about SMBus please refer to this link: **SMBus wikipedia**[3] or **i2c-bus.org**[4]

2.1 Features

Here are the main features:

- Multi-master
- Standard (100 KHz) and fast speed modes (400 KHz and Plus 1 MHz)
- I2C 10-bit address
- I2C slave capabilities (programmable I2C address)
- DMA capabilities
- SMBus 2.0 compatible
  - Standard bus protocol (quick command; byte, word, block read/write)
  - Host notification
  - Alert

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

2.2 Security support

- There are six I2C instances.
  - I2C instances 1, 2, 3 and 5 are **non-secure**.
  - I2C instances 4 and 6 can be **secure** (under ETZPC control).

3 Peripheral usage and associated software

3.1 Boot time

The I2C peripheral is usually not used at boot time. But it may be used by the SSBL and/or FSBL (see Boot chains overview), for example, to configure a PMIC (if any), or to access data stored in an external EEPROM.

3.2 Runtime

3.2.1 Overview

I2C4&6 instances can be allocated to:

- the Arm® Cortex®-A7 secure core to be controlled in OP-TEE by the OP-TEE I2C driver

All I2C instances can be allocated to:

- the Arm® Cortex®-A7 non-secure core to be controlled in Linux® by the I2C framework

All but I2C4&6 instances can be allocated to:

- the Arm® Cortex®-M4 to be controlled in STM32Cube MPU Package by STM32Cube I2C driver

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

3.2.2 Software frameworks

Domain	Peripheral	Software components			Comment
OP-TEE	Linux	STM32Cube			
Low speed interface	I2C	OP-TEE I2C driver	I2C Engine framework	STM32Cube I2C driver	

3.2.3 Peripheral configuration

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

For Linux® kernel configuration, please refer to I2C configuration.

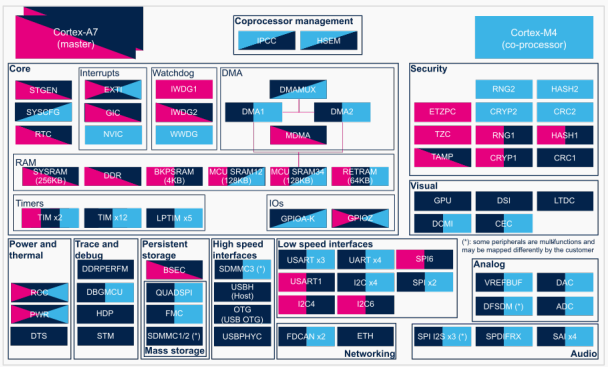
Please refer to I2C device tree configuration for detailed information on how to configure I2C peripherals.

3.2.4 Peripheral assignment

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

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

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



Internal peripherals

Domain	Periphera l	Runtime allocation				Comment
Instance	Cortex-A7 secure (OP-TEE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)			
Low speed interface	I2C	I2C1				Assignment (single choice)
		I2C2				Assignment (single choice)
		I2C3				Assignment (single choice)
		I2C4				Assignment (single choice). Used for PMIC control on ST boards .
		I2C5				Assignment (single choice)
		I2C6				Assignment (single choice)

4 References

- http://en.wikipedia.org/wiki/I2C
- https://www.i2c-bus.org/specification/
- https://en.wikipedia.org/wiki/System_Management_Bus
- https://www.i2c-bus.org/smbus/

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

Template:ArticleMainWriter

Template:ArticleApprovedVersion

Contents

1 Article purpose

The purpose of this article is to:

- briefly introduce the I2C 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 necessary, how to configure the I2C peripheral.

2 Peripheral overview

The I2C bus interface serves as an interface between the microcontroller and the serial I2C bus.
It provides multi-master capability, and controls all I2C bus-specific sequencing, protocol, arbitration and timing.
The I2C controller allows to be a slave as well if need be.
It is also SMBus 2.0 compatible.
For more information about I2C please refer to this link: **I2C wikipedia**[1] or **i2c-bus.org**[2]
For more information about SMBus please refer to this link: **SMBus wikipedia**[3] or **i2c-bus.org**[4]

2.1 Features

Here are the main features:

- Multi-master
- Standard (100 KHz) and fast speed modes (400 KHz and Plus 1 MHz)
- I2C 10-bit address
- I2C slave capabilities (programmable I2C address)
- DMA capabilities
- SMBus 2.0 compatible
  - Standard bus protocol (quick command; byte, word, block read/write)
  - Host notification
  - Alert

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

2.2 Security support

- There are six I2C instances.
  - I2C instances 1, 2, 3 and 5 are **non-secure**.
  - I2C instances 4 and 6 can be **secure** (under ETZPC control).

3 Peripheral usage and associated software

3.1 Boot time

The I2C peripheral is usually not used at boot time. But it may be used by the SSBL and/or FSBL (see Boot chains overview), for example, to configure a PMIC (if any), or to access data stored in an external EEPROM.

3.2 Runtime

3.2.1 Overview

I2C4&6 instances can be allocated to:

- the Arm® Cortex®-A7 secure core to be controlled in OP-TEE by the OP-TEE I2C driver

All I2C instances can be allocated to:

- the Arm® Cortex®-A7 non-secure core to be controlled in Linux® by the I2C framework

All but I2C4&6 instances can be allocated to:

- the Arm® Cortex®-M4 to be controlled in STM32Cube MPU Package by STM32Cube I2C driver

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

3.2.2 Software frameworks

Domain	Peripheral	Software components			Comment
OP-TEE	Linux	STM32Cube			
Low speed interface	I2C	OP-TEE I2C driver	I2C Engine framework	STM32Cube I2C driver	

3.2.3 Peripheral configuration

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

For Linux® kernel configuration, please refer to I2C configuration.

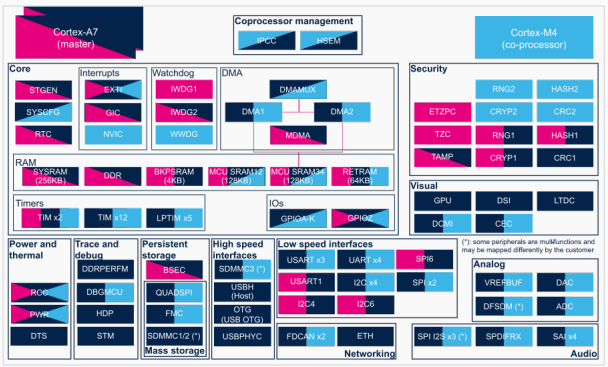
Please refer to I2C device tree configuration for detailed information on how to configure I2C peripherals.

3.2.4 Peripheral assignment

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

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

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



Internal peripherals

Domain	Periphera l	Runtime allocation				Comment
Instance	Cortex-A7 secure (OP-TEE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)			
Low speed interface	I2C	I2C1				Assignment (single choice)
		I2C2				Assignment (single choice)
		I2C3				Assignment (single choice)
		I2C4				Assignment (single choice). Used for PMIC control on ST boards .
		I2C5				Assignment (single choice)
		I2C6				Assignment (single choice)

4 References

- http://en.wikipedia.org/wiki/I2C
- https://www.i2c-bus.org/specification/
- https://en.wikipedia.org/wiki/System_Management_Bus
- https://www.i2c-bus.org/smbus/

Stable: 03.06.2021 - 12:48 / Revision: 03.06.2021 - 08:46

Template:ArticleMainWriter

Template:ArticleApprovedVersion

Contents

1 Article purpose

The purpose of this article is to:

- briefly introduce the I2C 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 necessary, how to configure the I2C peripheral.

2 Peripheral overview

The I2C bus interface serves as an interface between the microcontroller and the serial I2C bus.
It provides multi-master capability, and controls all I2C bus-specific sequencing, protocol, arbitration and timing.
The I2C controller allows to be a slave as well if need be.
It is also SMBus 2.0 compatible.
For more information about I2C please refer to this link: **I2C wikipedia**[1] or **i2c-bus.org**[2]
For more information about SMBus please refer to this link: **SMBus wikipedia**[3] or **i2c-bus.org**[4]

2.1 Features

Here are the main features:

- Multi-master
- Standard (100 KHz) and fast speed modes (400 KHz and Plus 1 MHz)
- I2C 10-bit address
- I2C slave capabilities (programmable I2C address)
- DMA capabilities
- SMBus 2.0 compatible
  - Standard bus protocol (quick command; byte, word, block read/write)
  - Host notification
  - Alert

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

2.2 Security support

- There are six I2C instances.
  - I2C instances 1, 2, 3 and 5 are **non-secure**.
  - I2C instances 4 and 6 can be **secure** (under ETZPC control).

3 Peripheral usage and associated software

3.1 Boot time

The I2C peripheral is usually not used at boot time. But it may be used by the SSBL and/or FSBL (see Boot chains overview), for example, to configure a PMIC (if any), or to access data stored in an external EEPROM.

3.2 Runtime

3.2.1 Overview

I2C4&6 instances can be allocated to:

- the Arm® Cortex®-A7 secure core to be controlled in OP-TEE by the OP-TEE I2C driver

All I2C instances can be allocated to:

- the Arm® Cortex®-A7 non-secure core to be controlled in Linux® by the I2C framework

All but I2C4&6 instances can be allocated to:

- the Arm® Cortex®-M4 to be controlled in STM32Cube MPU Package by STM32Cube I2C driver

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

3.2.2 Software frameworks

Domain	Peripheral	Software components			Comment
OP-TEE	Linux	STM32Cube			
Low speed interface	I2C	OP-TEE I2C driver	I2C Engine framework	STM32Cube I2C driver	

3.2.3 Peripheral configuration

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

For Linux® kernel configuration, please refer to I2C configuration.

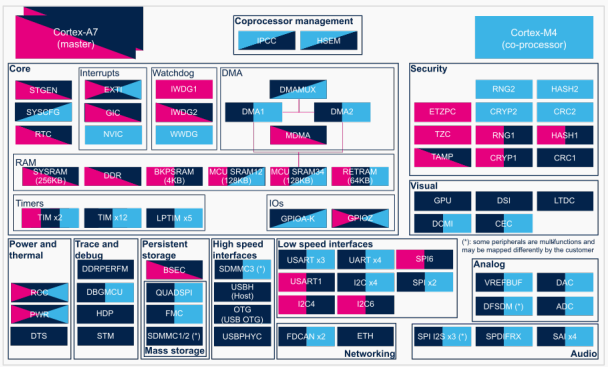
Please refer to I2C device tree configuration for detailed information on how to configure I2C peripherals.

3.2.4 Peripheral assignment

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

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

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



Internal peripherals

Domain	Periphera l	Runtime allocation				Comment
Instance	Cortex-A7 secure (OP-TEE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)			
Low speed interface	I2C	I2C1				Assignment (single choice)
		I2C2				Assignment (single choice)
		I2C3				Assignment (single choice)
		I2C4				Assignment (single choice). Used for PMIC control on ST boards .
		I2C5				Assignment (single choice)
		I2C6				Assignment (single choice)

4 References

- http://en.wikipedia.org/wiki/I2C
- https://www.i2c-bus.org/specification/
- https://en.wikipedia.org/wiki/System_Management_Bus
- https://www.i2c-bus.org/smbus/

Stable: 30.03.2021 - 10:03 / Revision: 30.03.2021 - 08:24

Template:ArticleMainWriter

Template:ArticleApprovedVersion

Contents

1 Article purpose

The purpose of this article is to:

- briefly introduce the I2C 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 necessary, how to configure the I2C peripheral.

2 Peripheral overview

The I2C bus interface serves as an interface between the microcontroller and the serial I2C bus.
It provides multi-master capability, and controls all I2C bus-specific sequencing, protocol, arbitration and timing.
The I2C controller allows to be a slave as well if need be.
It is also SMBus 2.0 compatible.
For more information about I2C please refer to this link: **I2C wikipedia**[1] or **i2c-bus.org**[2]
For more information about SMBus please refer to this link: **SMBus wikipedia**[3] or **i2c-bus.org**[4]

2.1 Features

Here are the main features:

- Multi-master
- Standard (100 KHz) and fast speed modes (400 KHz and Plus 1 MHz)
- I2C 10-bit address
- I2C slave capabilities (programmable I2C address)
- DMA capabilities
- SMBus 2.0 compatible
  - Standard bus protocol (quick command; byte, word, block read/write)
  - Host notification
  - Alert

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

2.2 Security support

- There are six I2C instances.
  - I2C instances 1, 2, 3 and 5 are **non-secure**.
  - I2C instances 4 and 6 can be **secure** (under ETZPC control).

3 Peripheral usage and associated software

3.1 Boot time

The I2C peripheral is usually not used at boot time. But it may be used by the SSBL and/or FSBL (see Boot chains overview), for example, to configure a PMIC (if any), or to access data stored in an external EEPROM.

3.2 Runtime

3.2.1 Overview

I2C4&6 instances can be allocated to:

- the Arm® Cortex®-A7 secure core to be controlled in OP-TEE by the OP-TEE I2C driver

All I2C instances can be allocated to:

- the Arm® Cortex®-A7 non-secure core to be controlled in Linux® by the I2C framework

All but I2C4&6 instances can be allocated to:

- the Arm® Cortex®-M4 to be controlled in STM32Cube MPU Package by STM32Cube I2C driver

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

3.2.2 Software frameworks

Domain	Peripheral	Software components			Comment
OP-TEE	Linux	STM32Cube			
Low speed interface	I2C	OP-TEE I2C driver	I2C Engine framework	STM32Cube I2C driver	

3.2.3 Peripheral configuration

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

For Linux® kernel configuration, please refer to I2C configuration.

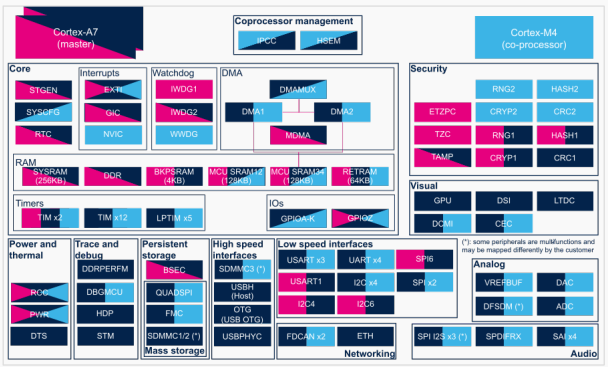
Please refer to I2C device tree configuration for detailed information on how to configure I2C peripherals.

3.2.4 Peripheral assignment

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

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

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



Internal peripherals

Domain	Periphera l	Runtime allocation				Comment
Instance	Cortex-A7 secure (OP-TEE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)			
Low speed interface	I2C	I2C1				Assignment (single choice)
		I2C2				Assignment (single choice)
		I2C3				Assignment (single choice)
		I2C4				Assignment (single choice). Used for PMIC control on ST boards .
		I2C5				Assignment (single choice)
		I2C6				Assignment (single choice)

4 References

- http://en.wikipedia.org/wiki/I2C
- https://www.i2c-bus.org/specification/
- https://en.wikipedia.org/wiki/System_Management_Bus
- https://www.i2c-bus.org/smbus/

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

Template:ArticleMainWriter

Template:ArticleApprovedVersion

Contents

1 Article purpose

The purpose of this article is to:

- briefly introduce the I2C 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 necessary, how to configure the I2C peripheral.

2 Peripheral overview

The I2C bus interface serves as an interface between the microcontroller and the serial I2C bus.
It provides multi-master capability, and controls all I2C bus-specific sequencing, protocol, arbitration and timing.
The I2C controller allows to be a slave as well if need be.
It is also SMBus 2.0 compatible.

For more information about I2C please refer to this link: **I2C wikipedia**[1] or **i2c-bus.org**[2]

For more information about SMBus please refer to this link: **SMBus wikipedia**[3] or **i2c-bus.org**[4]

2.1 Features

Here are the main features:

- Multi-master
- Standard (100 KHz) and fast speed modes (400 KHz and Plus 1 MHz)
- I2C 10-bit address
- I2C slave capabilities (programmable I2C address)
- DMA capabilities
- SMBus 2.0 compatible
  - Standard bus protocol (quick command; byte, word, block read/write)
  - Host notification
  - Alert

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

2.2 Security support

- There are six I2C instances.
  - I2C instances 1, 2, 3 and 5 are **non-secure**.
  - I2C instances 4 and 6 can be **secure** (under ETZPC control).

3 Peripheral usage and associated software

3.1 Boot time

The I2C peripheral is usually not used at boot time. But it may be used by the SSBL and/or FSBL (see Boot chains overview), for example, to configure a PMIC (if any), or to access data stored in an external EEPROM.

3.2 Runtime

3.2.1 Overview

I2C4&6 instances can be allocated to:

- the Arm® Cortex®-A7 secure core to be controlled in OP-TEE by the OP-TEE I2C driver

All I2C instances can be allocated to:

- the Arm® Cortex®-A7 non-secure core to be controlled in Linux® by the I2C framework

All but I2C4&6 instances can be allocated to:

- the Arm® Cortex®-M4 to be controlled in STM32Cube MPU Package by STM32Cube I2C driver

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

3.2.2 Software frameworks

Domain	Peripheral	Software components			Comment
OP-TEE	Linux	STM32Cube			
Low speed interface	I2C	OP-TEE I2C driver	I2C Engine framework	STM32Cube I2C driver	

3.2.3 Peripheral configuration

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

For Linux® kernel configuration, please refer to I2C configuration.

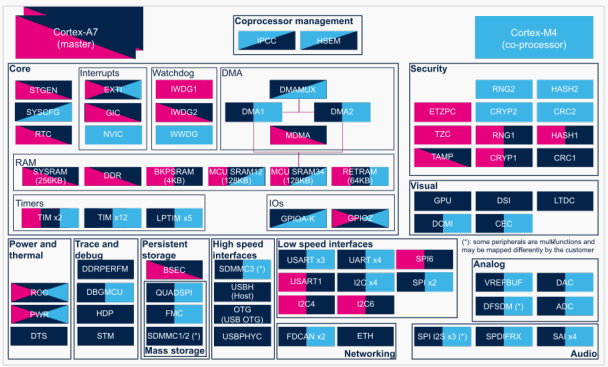
Please refer to I2C device tree configuration for detailed information on how to configure I2C peripherals.

3.2.4 Peripheral assignment

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

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

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



Internal peripherals

Domain	Periphera l	Runtime allocation				Comment
Instance	Cortex-A7 secure (OP-TEE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)			
Low speed interface	I2C	I2C1				Assignment (single choice)
		I2C2				Assignment (single choice)
		I2C3				Assignment (single choice)
		I2C4				Assignment (single choice). Used for PMIC control on ST boards .
		I2C5				Assignment (single choice)
		I2C6				Assignment (single choice)

4 References

- http://en.wikipedia.org/wiki/I2C
- https://www.i2c-bus.org/specification/
- https://en.wikipedia.org/wiki/System_Management_Bus
- https://www.i2c-bus.org/smbus/

Stable: 31.03.2021 - 11:58 / Revision: 23.03.2021 - 14:07

Template:ArticleMainWriter

Template:ArticleApprovedVersion

Contents

1 Article purpose

The purpose of this article is to:

- briefly introduce the I2C 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 necessary, how to configure the I2C peripheral.

2 Peripheral overview

The I2C bus interface serves as an interface between the microcontroller and the serial I2C bus.
It provides multi-master capability, and controls all I2C bus-specific sequencing, protocol, arbitration and timing.
The I2C controller allows to be a slave as well if need be.
It is also SMBus 2.0 compatible.
For more information about I2C please refer to this link: **I2C wikipedia**[1] or **i2c-bus.org**[2]
For more information about SMBus please refer to this link: **SMBus wikipedia**[3] or **i2c-bus.org**[4]

2.1 Features

Here are the main features:

- Multi-master
- Standard (100 KHz) and fast speed modes (400 KHz and Plus 1 MHz)
- I2C 10-bit address
- I2C slave capabilities (programmable I2C address)
- DMA capabilities
- SMBus 2.0 compatible
  - Standard bus protocol (quick command; byte, word, block read/write)
  - Host notification
  - Alert

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

2.2 Security support

- There are six I2C instances.
  - I2C instances 1, 2, 3 and 5 are **non-secure**.
  - I2C instances 4 and 6 can be **secure** (under ETZPC control).

3 Peripheral usage and associated software

3.1 Boot time

The I2C peripheral is usually not used at boot time. But it may be used by the SSBL and/or FSBL (see Boot chains overview), for example, to configure a PMIC (if any), or to access data stored in an external EEPROM.

3.2 Runtime

3.2.1 Overview

I2C4&6 instances can be allocated to:

- the Arm® Cortex®-A7 secure core to be controlled in OP-TEE by the OP-TEE I2C driver

All I2C instances can be allocated to:

- the Arm® Cortex®-A7 non-secure core to be controlled in Linux® by the I2C framework

All but I2C4&6 instances can be allocated to:

- the Arm® Cortex®-M4 to be controlled in STM32Cube MPU Package by STM32Cube I2C driver

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

3.2.2 Software frameworks

Domain	Peripheral	Software components			Comment
OP-TEE	Linux	STM32Cube			
Low speed interface	I2C	OP-TEE I2C driver	I2C Engine framework	STM32Cube I2C driver	

3.2.3 Peripheral configuration

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

For Linux® kernel configuration, please refer to I2C configuration.

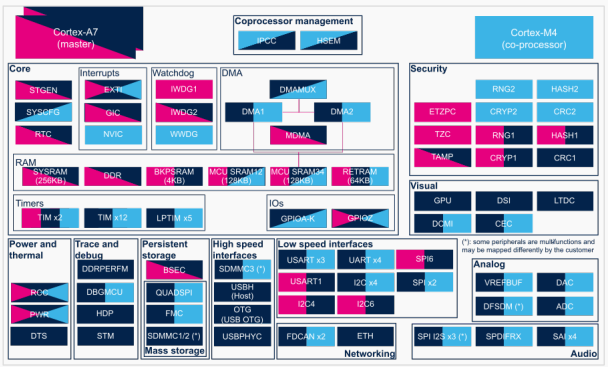
Please refer to I2C device tree configuration for detailed information on how to configure I2C peripherals.

3.2.4 Peripheral assignment

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

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

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



Internal peripherals

Domain	Periphera l	Runtime allocation				Comment
Instance	Cortex-A7 secure (OP-TEE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)			
Low speed interface	I2C	I2C1				Assignment (single choice)
		I2C2				Assignment (single choice)
		I2C3				Assignment (single choice)
		I2C4				Assignment (single choice). Used for PMIC control on ST boards .
		I2C5				Assignment (single choice)
		I2C6				Assignment (single choice)

4 References

- http://en.wikipedia.org/wiki/I2C
- https://www.i2c-bus.org/specification/
- https://en.wikipedia.org/wiki/System_Management_Bus
- https://www.i2c-bus.org/smbus/

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

Template:ArticleMainWriter

Template:ArticleApprovedVersion

Contents

1 Article purpose

The purpose of this article is to:

- briefly introduce the I2C 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 necessary, how to configure the I2C peripheral.

2 Peripheral overview

The I2C bus interface serves as an interface between the microcontroller and the serial I2C bus.
It provides multi-master capability, and controls all I2C bus-specific sequencing, protocol, arbitration and timing.
The I2C controller allows to be a slave as well if need be.
It is also SMBus 2.0 compatible.
For more information about I2C please refer to this link: **I2C wikipedia**[1] or **i2c-bus.org**[2]
For more information about SMBus please refer to this link: **SMBus wikipedia**[3] or **i2c-bus.org**[4]

2.1 Features

Here are the main features:

- Multi-master
- Standard (100 KHz) and fast speed modes (400 KHz and Plus 1 MHz)
- I2C 10-bit address
- I2C slave capabilities (programmable I2C address)
- DMA capabilities
- SMBus 2.0 compatible
  - Standard bus protocol (quick command; byte, word, block read/write)
  - Host notification
  - Alert

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

2.2 Security support

- There are six I2C instances.
  - I2C instances 1, 2, 3 and 5 are **non-secure**.
  - I2C instances 4 and 6 can be **secure** (under ETZPC control).

3 Peripheral usage and associated software

3.1 Boot time

The I2C peripheral is usually not used at boot time. But it may be used by the SSBL and/or FSBL (see Boot chains overview), for example, to configure a PMIC (if any), or to access data stored in an external EEPROM.

3.2 Runtime

3.2.1 Overview

I2C4&6 instances can be allocated to:

- the Arm® Cortex®-A7 secure core to be controlled in OP-TEE by the OP-TEE I2C driver

All I2C instances can be allocated to:

- the Arm® Cortex®-A7 non-secure core to be controlled in Linux® by the I2C framework

All but I2C4&6 instances can be allocated to:

- the Arm® Cortex®-M4 to be controlled in STM32Cube MPU Package by STM32Cube I2C driver

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

3.2.2 Software frameworks

Domain	Peripheral	Software components			Comment
OP-TEE	Linux	STM32Cube			
Low speed interface	I2C	OP-TEE I2C driver	I2C Engine framework	STM32Cube I2C driver	

3.2.3 Peripheral configuration

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

For Linux® kernel configuration, please refer to I2C configuration.

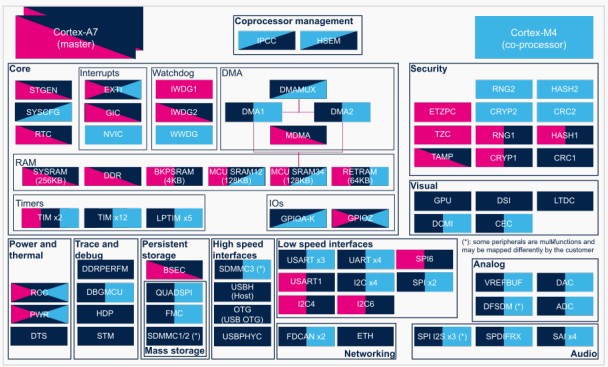
Please refer to I2C device tree configuration for detailed information on how to configure I2C peripherals.

3.2.4 Peripheral assignment

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

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

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



Internal peripherals

Domain	Periphera l	Runtime allocation				Comment
Instance	Cortex-A7 secure (OP-TEE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)			
Low speed interface	I2C	I2C1				Assignment (single choice)
		I2C2				Assignment (single choice)
		I2C3				Assignment (single choice)
		I2C4				Assignment (single choice). Used for PMIC control on ST boards .
		I2C5				Assignment (single choice)
		I2C6				Assignment (single choice)

4 References

- http://en.wikipedia.org/wiki/I2C
- https://www.i2c-bus.org/specification/
- https://en.wikipedia.org/wiki/System_Management_Bus
- https://www.i2c-bus.org/smbus/

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

Template:ArticleMainWriter

Template:ArticleApprovedVersion

Contents

1 Article purpose

The purpose of this article is to:

- briefly introduce the I2C 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 necessary, how to configure the I2C peripheral.

2 Peripheral overview

The I2C bus interface serves as an interface between the microcontroller and the serial I2C bus.
It provides multi-master capability, and controls all I2C bus-specific sequencing, protocol, arbitration and timing.
The I2C controller allows to be a slave as well if need be.
It is also SMBus 2.0 compatible.
For more information about I2C please refer to this link: **I2C wikipedia**[1] or **i2c-bus.org**[2]
For more information about SMBus please refer to this link: **SMBus wikipedia**[3] or **i2c-bus.org**[4]

2.1 Features

Here are the main features:

- Multi-master
- Standard (100 KHz) and fast speed modes (400 KHz and Plus 1 MHz)
- I2C 10-bit address
- I2C slave capabilities (programmable I2C address)
- DMA capabilities
- SMBus 2.0 compatible
  - Standard bus protocol (quick command; byte, word, block read/write)
  - Host notification
  - Alert

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

2.2 Security support

- There are six I2C instances.
  - I2C instances 1, 2, 3 and 5 are **non-secure**.
  - I2C instances 4 and 6 can be **secure** (under ETZPC control).

3 Peripheral usage and associated software

3.1 Boot time

The I2C peripheral is usually not used at boot time. But it may be used by the SSBL and/or FSBL (see Boot chains overview), for example, to configure a PMIC (if any), or to access data stored in an external EEPROM.

3.2 Runtime

3.2.1 Overview

I2C4&6 instances can be allocated to:

- the Arm® Cortex®-A7 secure core to be controlled in OP-TEE by the OP-TEE I2C driver

All I2C instances can be allocated to:

- the Arm® Cortex®-A7 non-secure core to be controlled in Linux® by the I2C framework

All but I2C4&6 instances can be allocated to:

- the Arm® Cortex®-M4 to be controlled in STM32Cube MPU Package by STM32Cube I2C driver

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

3.2.2 Software frameworks

Domain	Peripheral	Software components			Comment
OP-TEE	Linux	STM32Cube			
Low speed interface	I2C	OP-TEE I2C driver	I2C Engine framework	STM32Cube I2C driver	

3.2.3 Peripheral configuration

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

For Linux® kernel configuration, please refer to I2C configuration.

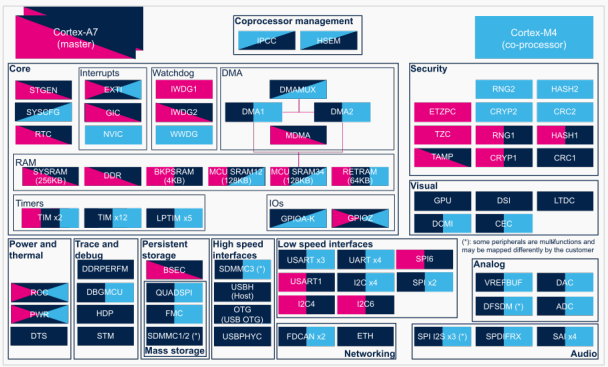
Please refer to I2C device tree configuration for detailed information on how to configure I2C peripherals.

3.2.4 Peripheral assignment

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

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

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



Internal peripherals

Domain	Periphera l	Runtime allocation				Comment
Instance	Cortex-A7 secure (OP-TEE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)			
Low speed interface	I2C	I2C1				Assignment (single choice)
		I2C2				Assignment (single choice)
		I2C3				Assignment (single choice)
		I2C4				Assignment (single choice). Used for PMIC control on ST boards .
		I2C5				Assignment (single choice)
		I2C6				Assignment (single choice)

4 References

- http://en.wikipedia.org/wiki/I2C
- https://www.i2c-bus.org/specification/
- https://en.wikipedia.org/wiki/System_Management_Bus
- https://www.i2c-bus.org/smbus/

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

Template:ArticleMainWriter

Template:ArticleApprovedVersion

Contents

1 Article purpose

The purpose of this article is to:

- briefly introduce the I2C 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 necessary, how to configure the I2C peripheral.

2 Peripheral overview

The I2C bus interface serves as an interface between the microcontroller and the serial I2C bus.
It provides multi-master capability, and controls all I2C bus-specific sequencing, protocol, arbitration and timing.
The I2C controller allows to be a slave as well if need be.
It is also SMBus 2.0 compatible.
For more information about I2C please refer to this link: **I2C wikipedia**[1] or **i2c-bus.org**[2]
For more information about SMBus please refer to this link: **SMBus wikipedia**[3] or **i2c-bus.org**[4]

2.1 Features

Here are the main features:

- Multi-master
- Standard (100 KHz) and fast speed modes (400 KHz and Plus 1 MHz)
- I2C 10-bit address
- I2C slave capabilities (programmable I2C address)
- DMA capabilities
- SMBus 2.0 compatible
  - Standard bus protocol (quick command; byte, word, block read/write)
  - Host notification
  - Alert

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

2.2 Security support

- There are six I2C instances.
  - I2C instances 1, 2, 3 and 5 are **non-secure**.
  - I2C instances 4 and 6 can be **secure** (under ETZPC control).

3 Peripheral usage and associated software

3.1 Boot time

The I2C peripheral is usually not used at boot time. But it may be used by the SSBL and/or FSBL (see Boot chains overview), for example, to configure a PMIC (if any), or to access data stored in an external EEPROM.

3.2 Runtime

3.2.1 Overview

I2C4&6 instances can be allocated to:

- the Arm® Cortex®-A7 secure core to be controlled in OP-TEE by the OP-TEE I2C driver

All I2C instances can be allocated to:

- the Arm® Cortex®-A7 non-secure core to be controlled in Linux® by the I2C framework

All but I2C4&6 instances can be allocated to:

- the Arm® Cortex®-M4 to be controlled in STM32Cube MPU Package by STM32Cube I2C driver

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

3.2.2 Software frameworks

Domain	Peripheral	Software components			Comment
OP-TEE	Linux	STM32Cube			
Low speed interface	I2C	OP-TEE I2C driver	I2C Engine framework	STM32Cube I2C driver	

3.2.3 Peripheral configuration

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

For Linux® kernel configuration, please refer to I2C configuration.

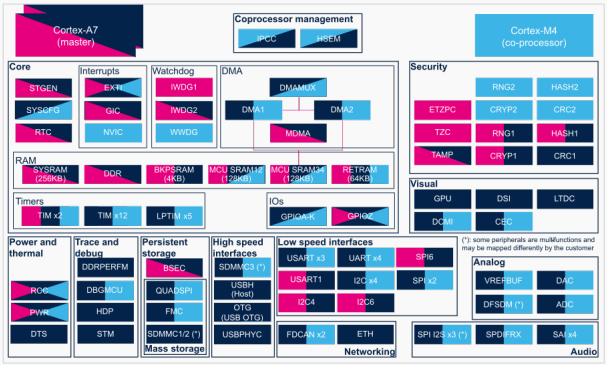
Please refer to I2C device tree configuration for detailed information on how to configure I2C peripherals.

3.2.4 Peripheral assignment

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

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

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



Internal peripherals

Domain	Peripheral	Runtime allocation				Comment
Instance	Cortex-A7 secure (OP-TEE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)			
Low speed interface	I2C	I2C1				Assignment (single choice)
		I2C2				Assignment (single choice)
		I2C3				Assignment (single choice)
		I2C4				Assignment (single choice). Used for PMIC control on ST boards .
		I2C5				Assignment (single choice)
		I2C6				Assignment (single choice)

4 References

- http://en.wikipedia.org/wiki/I2C
- https://www.i2c-bus.org/specification/
- https://en.wikipedia.org/wiki/System_Management_Bus
- https://www.i2c-bus.org/smbus/