



I2C internal peripheral



# I2C internal peripheral

Stable: 21.02.2020 - 08:46 / Revision: 03.02.2020 - 13:32

## Contents

1 Article purpose .....	2
2 Peripheral overview .....	2
<b>2.1 Features .....</b>	<b>2</b>
<b>2.2 Security support .....</b>	<b>3</b>
3 Peripheral usage and associated software .....	3
<b>3.1 Boot time .....</b>	<b>3</b>
<b>3.2 Runtime .....</b>	<b>3</b>
3.2.1 Overview .....	3
3.2.2 Software frameworks .....	3
3.2.3 Peripheral configuration .....	4
3.2.4 Peripheral assignment .....	4
4 References .....	6

## 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](#)<sup>[1]</sup> or [i2c-bus.org](#)<sup>[2]</sup>

For more information about SMBus please refer to this link: [SMBus wikipedia](#)<sup>[3]</sup> or [i2c-bus.org](#)<sup>[4]</sup>

### 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 (see [PMIC hardware components](#)), 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<sup>®</sup> Cortex<sup>®</sup>-A7 secure core to be controlled in OP-TEE by the [OP-TEE I2C driver](#)

All I2C instances can be allocated to:

- the Arm<sup>®</sup> Cortex<sup>®</sup>-A7 non-secure core to be controlled in U-Boot or Linux<sup>®</sup> by the [I2C framework](#)

All but I2C4&6 instances can be allocated to:

- the Arm<sup>®</sup> Cortex<sup>®</sup>-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

Do	Peri	Software frameworks	Comment
mai Cor tex	Cor tex		

Do	Peri	Software frameworks			Comment
main -A7 secure (O P- TE E)	-A7	Cortex-M4			
	no	(STM32Cube)			
Low speed I2C interf face	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<sup>®</sup> kernel configuration, please refer to [I2C configuration](#).

Please refer to [I2C device tree configuration](#) for detailed information on how to configure I2C peripherals.

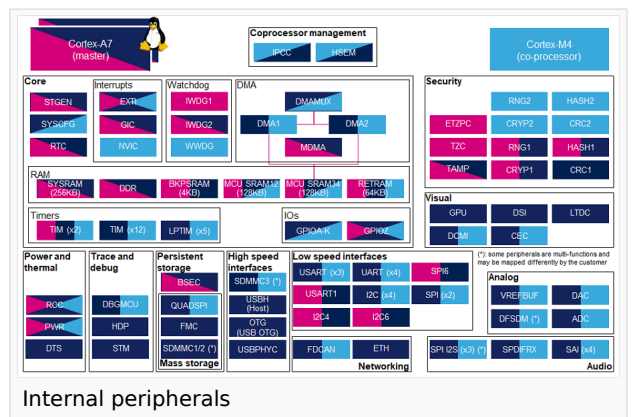
### 3.2.4 Peripheral assignment

**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 possibilities might be described in [STM32MP15 reference manuals](#).



Do	Peri	Runtime allocation			Comme
main in era	I2C				nt



I2C internal peripheral

main processor	I2C	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)			Assignment
Low speed		I2C1				Assignment (single choice)
		I2C2				Assignment (single choice)
		I2C3				Assignment (single choice)
	I2C	I2C4				Assignment (single choice). Used for P MIC



Do ma in f a c e	Per iph era l	Runtime allocation				Comme nt  I o n S T b o a r d s.
		I2C5				Assig nment (singl e choic e )
		I2C6				Assig nment (singl e choic e )

## 4 References

- <http://en.wikipedia.org/wiki/I2C>
- <https://www.i2c-bus.org/specification/>
- [https://en.wikipedia.org/wiki/System\\_Management\\_Bus](https://en.wikipedia.org/wiki/System_Management_Bus)
- <https://www.i2c-bus.org/smbus/>

Inter-Integrated Circuit (Bi-directional 2-wire bus standard for efficient inter-IC control.)

System Management Bus

Direct Memory Access

Second Stage Boot Loader

First Stage Boot Loader

Power Management Integrated Circuit

Electrically-erasable programmable read-only memory

Open Portable Trusted Execution Environment

Das U-Boot -- the Universal Boot Loader (see [U-Boot\\_overview](#))

Microprocessor Unit