OTG internal peripheral

Contents

1. OTG internal peripheral 3
2. USBPHYC internal peripheral 6
3. STM32MP15 resources 7
4. STM32MP15 ROM code overview 7
5. Boot chains overview 7
6. STM32CubeProgrammer 7
7. USB overview 7
8. STM32CubeMX 7
9. OTG device tree configuration 8
10. STM32MPU Embedded Software architecture overview 8
11. How to assign an internal peripheral to a runtime context 8

OTG internal peripheral

Stable: 04.02.2020 - 16:08 / Revision: 04.02.2020 - 16:02

Contents

1 Article purpose 3
2 Peripheral overview 3
2.1 Features 3
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 4
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 OTG peripheral and its main features
- indicate the level of security supported by this hardware block
- explain how it can be allocated to the three runtime contexts and linked to the corresponding software components
- explain, when needed, how to configure the OTG peripheral.

2 Peripheral overview

The **OTG** peripheral is used to interconnect other systems with STM32 MPU devices, using USB standard.

2.1 Features

The **OTG** peripheral is a USB Dual-Role Device (DRD) controller that supports both device and host functions.
In Host mode, it supports high-speed (480 Mbit/s), full-speed (12 Mbit/s) and low-speed (1.5 Mbit/s).
In Peripheral mode, high-speed and full-speed are supported, not low-speed.
The **OTG** peripheral embeds a full-speed PHY and supports a UTMI interface connected to internal HS PHY.

The **OTG** peripheral is fully compliant with

- *On-The-Go and Embedded Host Supplement to the USB Revision 2.0 Specification*[1], Revision 2.0, May 8, 2009
- *Universal Serial Bus Revision 2.0 Specification*[2], Revision 2.0, April 27, 2000

- *USB 2.0 Link Power Management Addendum Engineering Change Notice to the USB 2.0 specification*[3], July 16, 2007
- *USB 2.0 Transceiver Macrocell Interface (UTMI) Specification*[4], Version 1.05, March 29, 2001
- *UTMI+ Specification*[5], Revision 1.0, February 25, 2004

Refer to STM32MP15 reference manuals for the complete hardware feature list, and to the software components (introduced below) to know which features are supported.

2.2 Security support

The **OTG** peripheral is a **non-secure** peripheral.

3 Peripheral usage and associated software

3.1 Boot time

The OTG peripheral is used by ROM code, FSBL and SSBL in device mode (DFU) to support serial boot for flash programming with STM32CubeProgrammer.
The SSBL can use it in host mode (mass storage), for instance to boot on a kernel stored on a USB key, or after a kernel panic to perform the crash dump saving to the USB key.

3.2 Runtime

3.2.1 Overview

The OTG peripheral can be allocated to the Arm® Cortex®-A7 non-secure core to be used under Linux® with USB framework.

3.2.2 Software frameworks

Do	Peri	Software frameworks			Comment
mai n Cortex-A7 secure (OP-TEE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)			
Hi	O				

Domain	Peripheral	Software frameworks		Comment
gh speed interface	TG (USBOTG)		Linux USB framework	

3.2.3 Peripheral configuration

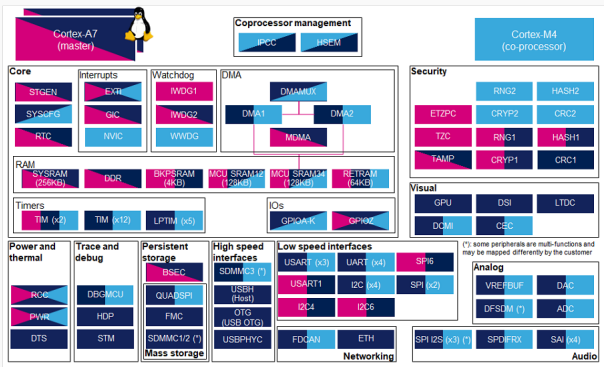
The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration by itself can be performed via the STM32CubeMX tool for all internal peripherals. It can then be manually completed (especially for external peripherals) according to the information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

For Linux kernel configuration, please refer to OTG device tree configuration.

3.2.4 Peripheral assignment

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

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



Internal peripherals

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

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

Domain	Peripheral	Runtime allocation				Comment
	(OPTEE)					
High speed interface	OTG (USB OTG)	OTG (USB OTG)				

4 References

- On-The-Go and Embedded Host Supplement to the USB Revision 2.0 Specification
- Universal Serial Bus Revision 2.0 Specification
- ECN USB 2.0 Link Power Management Addendum
- USB 2.0 Transceiver Macrocell Interface (UTMI) Specification
- UTMI+ Specification

USB On-The-Go (Capability/type of USB port, acting primarily as USB device, to also act as USB host. Also known as USB OTG.)

Microprocessor Unit

Dual-Role Device (USB standard defines host and device roles. OTG controllers support both roles and can be called Dual-Role Devices controllers.)

USB 2.0 Transceiver Macrocell Interface

Device Firmware Upgrade

Open Portable Trusted Execution Environment

USBPHYC internal peripheral

Stable: 16.01.2020 - 15:01 / Revision: 16.01.2020 - 14:57

Invalid target: no reviewed revision corresponds to the given ID.

Return to USBPHYC internal peripheral.

STM32MP15 resources

Stable: 24.06.2020 - 17:52 / Revision: 24.06.2020 - 12:32

Invalid target: no reviewed revision corresponds to the given ID.

Return to STM32MP15 resources.

STM32MP15 ROM code overview

Stable: 25.06.2020 - 09:47 / Revision: 25.06.2020 - 09:45

Invalid target: no reviewed revision corresponds to the given ID.

Return to STM32MP15 ROM code overview.

Boot chains overview

Stable: 25.06.2020 - 07:35 / Revision: 19.06.2020 - 07:24

Invalid target: no reviewed revision corresponds to the given ID.

Return to Boot chains overview.

STM32CubeProgrammer

Stable: 21.02.2020 - 10:10 / Revision: 20.02.2020 - 10:16

Invalid target: no reviewed revision corresponds to the given ID.

Return to STM32CubeProgrammer.

USB overview

Stable: 22.04.2020 - 10:41 / Revision: 22.04.2020 - 10:40

Invalid target: no reviewed revision corresponds to the given ID.

Return to USB overview.

STM32CubeMX

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

Invalid target: no reviewed revision corresponds to the given ID.

Return to STM32CubeMX.

OTG device tree configuration

Stable: 24.06.2020 - 12:06 / Revision: 22.06.2020 - 14:16

Invalid target: no reviewed revision corresponds to the given ID.

Return to OTG device tree configuration.

STM32MPU Embedded Software architecture overview

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

Invalid target: no reviewed revision corresponds to the given ID.

Return to STM32MPU Embedded Software architecture overview.

How to assign an internal peripheral to a runtime context

Stable: 22.06.2020 - 09:50 / Revision: 22.06.2020 - 09:49

Invalid target: no reviewed revision corresponds to the given ID.

Return to How to assign an internal peripheral to a runtime context.