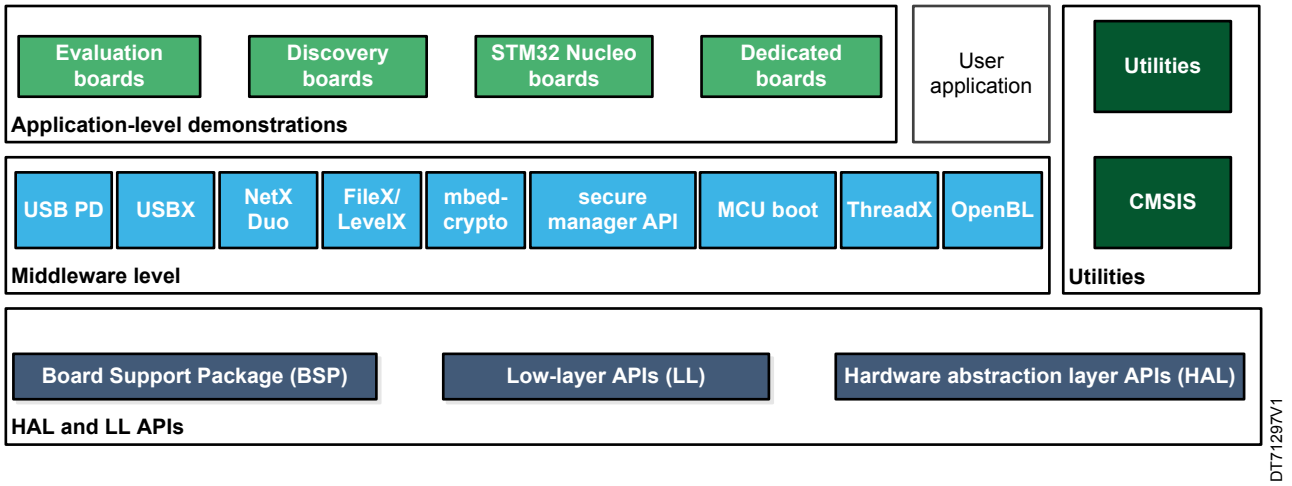
STM32Cube MCU Package examples for STM32H5 series

Introduction

The STM32CubeH5 MCU Package is delivered with a rich set of examples running on STMicroelectronics boards. The examples are organized by boards. They are provided with preconfigured projects for the main supported toolchains (refer to Figure 1. STM32CubeH5 firmware components).

Figure 1. STM32CubeH5 firmware components



1 Reference documents

The following items make up a reference set for the examples presented in this application note:

- The latest release of the STM32CubeH5 MCU Package for the 32-bit microcontrollers in the STM32H5 Series based on the Arm® Cortex®-M processor with Arm®TrustZone®
- *Getting started with STM32CubeH5 for STM32H5 Series* (UM3065)
- *Description of STM32H5 HAL and low-layer drivers* (UM3132)
- Wiki: Getting started with STM32H5 security
- Wiki: How to start with OEMiRoT on STM32H573 and 563–Arm® TrustZone® enabled
- Wiki: How to start with OEMiRoT on STM32H503
- Wiki: STiRoT for STM32H5

Note: Arm and TrustZone are registered trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere.

2 STM32CubeH5 examples

The examples are classified depending on the STM32Cube level that they apply to. They are named as follows:

- **Examples**
  These examples use only the HAL and BSP drivers (Middleware not used). Their objective is to demonstrate the product or peripheral features and usage. They are organized per peripheral (one folder per peripheral, such as TIM). Their complexity level ranges from the basic usage of a given peripheral, such as PWM generation using a timer, to the integration of several peripherals, such as how to use DAC for a signal generation with synchronization from TIM6 and DMA. The usage of the board resources is reduced to the strict minimum.
- **Examples_LL**
  These examples use only the LL drivers (HAL drivers and middleware components not used). They offer an optimum implementation of typical use cases of the peripheral features and configuration sequences. The examples are organized per peripheral (one folder for each peripheral, such as TIM) and are principally deployed on Nucleo boards.
- **Examples_MIX**
  These examples use only HAL, BSP, and LL drivers (Middleware components are not used). They aim at demonstrating how to use both HAL and LL APIs in the same application to combine the advantages of both APIs:
  - HAL offers high-level function-oriented APIs with high portability level by hiding product/IPs complexity for end-users.
  - LL provides low-level APIs at the register level with better optimization.

  The examples are organized per peripheral (one folder for each peripheral, such as TIM) and are exclusively deployed on Nucleo boards.
- **Applications**
  The applications demonstrate product performance and how to use the available middleware stacks. They are organized either by middleware (one folder per middleware, such as Azure® RTOS ThreadX) or product feature that requires high-level firmware bricks (such as ROT_AppliConfig). The integration of applications that use several middleware stacks is also supported.
- **Demonstrations**
  The demonstrations aim at integrating and running the maximum number of peripherals and middleware stacks to showcase the product features and performance.
- **Template project**
  The template project is provided to enable the user to quickly build a firmware application using HAL and BSP drivers on a given board.
- **Template_LL project**
  The template LL projects are provided to enable the user to quickly build a firmware application using LL drivers on a given board.

The examples are located under `STM32Cube_FW_H5_V1.0.0\Projects\`.

The examples in the default product configuration with the Arm® TrustZone® disabled have the same structure:

- `*\Inc` folder, containing all header files
- `*\Src` folder, containing the sources code
- `*\EWARM`, `*\MDK-ARM`, and `*\STM32CubeIDE` folders, containing the preconfigured project for each toolchain
- `*\README.md and *\readme.html` file, describing the example behavior and the environment required to run the example

The examples with the Arm® TrustZone® enabled are suffixed with "_TrustZone" and have the same structure:

- `*\Secure\Inc` folder, containing all secure project header files
- `*\Secure\Src` and `*\Secure_nsclib\` folders, containing all secure project sources code
- `*\NonSecure\Inc` folder, containing all non-secure project header files
- `*\NonSecure\Src` folder, containing all non-secure project sources code
- `*\EWARM`, `*\MDK-ARM`, and `*\STM32CubeIDE` folders, containing the preconfigured project for each toolchain
- `*\README.md and *\readme.html` file, describing the example behavior and the environment required to run the example

To run the example, proceed as follows:

1. Open the example using your preferred toolchain.
2. Rebuild all files and load the image into target memory.
3. Run the example by following the `*\README.md and *\readme.html` instructions.

Note: Refer to "Development toolchains and compilers" and "Supported devices and evaluation boards" sections of the firmware package release notes to know more about the software/hardware environment used for the MCU Package development and validation. The correct operation of the provided examples is not guaranteed in other environments, for example, when using different compilers or board versions.

The examples can be tailored to run on any compatible hardware: simply update the BSP drivers for your board, provided it has the same hardware functions (LED, LCD, pushbuttons, and others). The BSP is based on a modular architecture that can be easily ported to any hardware by implementing low-level routines.

contains the list of examples provided with the STM32CubeH5 MCU Package.

In this table, the label MX means that the projects are created using STM32CubeMX, the STM32Cube initialization code generator. Those projects can be opened with this tool to modify the projects themselves. The other projects are manually created to demonstrate the product features. In this table, the label TrustZone® means that the projects are created for devices with Arm® TrustZone® enabled. Read the project `*\README.md and *\readme.html` file for user option bytes configuration.

2.1 STM32CubeH5 firmware examples

Table 1. STM32CubeH5 firmware examples

STM32CubeMX-generated examples are marked MX.

Level	Module Name	Project Name	Description	STM32H5_CUSTOM_HW	STM32H573I-DK	NUCLEO-H563ZI	NUCLEO-H533RE	NUCLEO-H503RB
Templates_Board	-	-	This project provides a reference template for the NUCLEO-H533RE board based on the STM32Cube HAL API and the BSP drivers that can be used to build any firmware application.	-	-	-	MX	-
	Total number of templates_board: 1			**0**	**0**	**0**	**1**	**0**
Templates	-	Starter Project	This projects provides a reference template through the HAL API that can be used to build any firmware application.	-	-	-	-	X
		TrustZoneDisabled	This project provides a reference template based on the STM32Cube HAL API that can be used to build any firmware application when security is not enabled (TZEN=C3).	-	X	X	X	-
		TrustZoneEnabled	This project provides a reference template based on the STM32Cube HAL API that can be used to build any firmware application when TrustZone security is activated (TZEN=B4).	-	X	X	X	-
	ROT	OEMiROT_Appli	This project provides a OEMiROT boot path application example. Boot is performed through OEMiROT bootpath after authenticity and integrity checks of the project firmware and project data images.	-	-	-	-	X
		OEMiROT_Appli_TrustZone	This project provides a OEMiROT boot path reference template. Boot is performed through OEMiROT boot path after authenticity and integrity checks of the project firmware and project data images.	-	X	X	X	-
		SMAK_Appli	This project provides a Secure Manager boot path reference template. Boot is performed through Secure Manager boot path after authenticity and integrity checks of the project firmware images.	-	X	-	-	-
		STiROT_Appli	This project provides a STiROT boot path reference template. Boot is performed through STiROT boot path after authenticity and the integrity checks of the project firmware and data images.	-	X	-	X	-
		STiROT_Appli_TrustZone	This project provides a STiROT boot path reference template. Boot is performed through STiROT boot path after authenticity and integrity checks of the project firmware and project data images.	-	X	-	X	-
	Total number of templates: 16			**0**	**6**	**3**	**5**	**2**
Templates_LL	-	Starter project	This projects provides a reference template through the LL API that can be used to build any firmware application.	-	-	-	-	X
		TrustZoneDisabled	This projects provides a reference template through the LL API that can be used to build any firmware application.	-	X	X	X	-
	Total number of templates_ll: 4			**0**	**1**	**1**	**1**	**1**
Examples	-	BSP	How to use the different BSP drivers of the board.	-	X	-	-	-

Level	Module Name	Project Name	Description	STM32H5_CUSTOM_HW	STM32H573I-DK	NUCLEO-H563ZI	NUCLEO-H533RE	NUCLEO-H503RB
Examples	ADC	ADC_AnalogWatchdog	How to use an ADC peripheral with an ADC analog watchdog to monitor a channel and detect when the corresponding conversion data is outside the window thresholds.	-	-	MX	-	-
		ADC_DifferentialMode	This example describes how to configure and use the ADC1 to convert an external analog input in Differential Mode, difference between external voltage on VinN and VinP.	-	-	-	MX	-
		ADC_SingleConversion_TriggerSW_DMA	How to use an ADC peripheral to perform a single ADC conversion on a channel at each software start. Converted data is transferred by DMA into a table in RAM memory.	-	-	MX	-	-
		ADC_SingleConversion_TriggerSW_IT	How to use ADC to convert a single channel at each SW start, conversion performed using programming model: interrupt.	-	-	-	-	MX
	CEC	CEC_DataExchange_Device_1	This example shows how to configure and use the CEC peripheral to receive and transmit messages.	-	MX	-	-	-
		CEC_DataExchange_Device_2	This example shows how to configure and use the CEC peripheral to receive and transmit messages.	-	MX	-	-	-
	COMP	COMP_OutputBlanking	How to use the comparator-peripheral output blanking feature. The purpose of the output blanking feature in motor control is to prevent tripping of the current regulation upon short current spikes at the beginning of the PWM period.	-	-	-	-	MX
	CORDIC	CORDIC_Sin_DMA	How to use the CORDIC peripheral to calculate array of sines in DMA mode.	-	-	MX	-	-
	CORTEX	CORTEXM_MPU	Presentation of the MPU features. This example configures MPU attributes of different MPU regions then configures a memory area as privileged read-only, and attempts to perform read and write operations in different modes.	-	-	MX	-	-
		CORTEXM_ModePrivilege	How to modify the Thread mode privilege access and stack. Thread mode is entered on reset or when returning from an exception.	-	-	MX	-	-
		CORTEXM_SysTick	How to use the default SysTick configuration with a 1 ms timebase to toggle LEDs.	-	-	MX	MX	MX
	CRC	CRC_Bytes_Stream_7bit_CRC	How to configure the CRC using the HAL API. The CRC (cyclic redundancy check) calculation unit computes 7-bit CRC codes derived from buffers of 8-bit data (bytes). The user-defined generating polynomial is manually set to 0x65, that is, X^6 + X^5 + X^2 + 1, as used in the Train Communication Network, IEC 60870-5[17].	-	-	-	MX	MX
		CRC_Example	How to configure the CRC using the HAL API. The CRC (cyclic redundancy check) calculation unit computes the CRC code of a given buffer of 32-bit data words, using a fixed generator polynomial (0x4C11DB7).	-	-	-	MX	MX
		CRC_UserDefinedPolynomial	How to configure the CRC using the HAL API. The CRC (cyclic redundancy check) calculation unit computes the 8-bit CRC code for a given buffer of 32-bit data words, based on a user-defined generating polynomial.	-	-	MX	-	-
	CRYP	CRYP_AES_CCM	How to use the CRYPTO peripheral to encrypt and decrypt data using AES with cipher block chaining-message authentication code(CCM).	-	MX	-	-	-

Level	Module Name	Project Name	Description	STM32H5_CUSTOM_HW	STM32H573I-DK	NUCLEO-H563ZI	NUCLEO-H533RE	NUCLEO-H503RB
Examples	CRYP	CRYP_AES_GCM	How to use the CRYPTO peripheral to encrypt and decrypt data using AES with Galois/Counter mode (GCM).	-	MX	-	-	-
		CRYP_AES_GCM_Padding	How to use the CRYPTO peripheral to encrypt and decrypt data using AES with Galois/Counter mode (GCM) with Padding.	-	-	-	MX	-
		CRYP_SAES_ECB_CBC	How to use the secure AES co-processor (SAES) peripheral to encrypt and decrypt data using AES ECB and CBC algorithms.	-	MX	-	-	-
		CRYP_SAES_GCM	How to use the CRYPTO peripheral to encrypt and decrypt data using AES with Galois/Counter mode (GCM).	-	-	-	MX	-
		CRYP_SAES_SharedKey	How to use the secure AES co-processor (SAES) peripheral to share application keys with AES peripheral.	-	MX	-	MX	-
		CRYP_SAES_WrapKey	How to use the secure AES co-processor (SAES) peripheral to wrap application keys using hardware secret key DHUK then use it to encrypt in polling mode.	-	MX	-	-	-
	DAC	DAC_SignalsGeneration	How to use the DAC peripheral to generate several signals using the DMA controller and the DAC internal wave generator.	-	-	MX	-	-
	DCACHE	DCACHE_Maintenance	How to do data-cache maintenance on a shared memory buffer accessed by 2 masters (CPU and DMA).	-	MX	-	-	-
	DLYB	DLYB_OSPI_NOR_FastTuning	How to use delay block (DLYB) with a fast tuning.	-	MX	-	-	-
	DMA	DMA_DataHandling	How to use the DMA controller to do data handling between transferred data from the source and transfer to the destination through the HAL API.	-	-	MX	-	-
		DMA_FLASHToRAM	How to use a DMA to transfer a word data buffer from flash memory to embedded SRAM through the HAL API.	-	-	MX	MX	MX
		DMA_LinkedList	How to use the DMA to perform a list of transfers. The transfer list is organized as linked-list, each time the current transfer ends the DMA automatically reload the next transfer parameters, and starts it (without CPU intervention).	-	-	MX	-	-
		DMA_RepeatedBlock	How to configure and use the DMA HAL API to perform repeated block transactions.	-	-	-	-	MX
		DMA_Trigger	How to configure and use the DMA HAL API to perform DMA triggered transactions.	-	-	-	-	MX
	DTS	DTS_GetTemperature	How to configure and use the DTS to get the temperature of the die.	-	-	MX	-	-
	FDCAN	FDCAN_Adaptive_Bitrate_Receiver	This example describes how to configure the FDCAN peripheral to adapt to different CAN bit rates using restricted mode.	-	-	MX	-	-

Level	Module Name	Project Name	Description	STM32H5_CUSTOM_HW	STM32H573I-DK	NUCLEO-H563ZI	NUCLEO-H533RE	NUCLEO-H503RB
Examples	FDCAN	FDCAN_Adaptive_Bitrate_Transmitter	This example describes how to configure the FDCAN peripheral to adapt to different CAN bit rates using restricted mode.	-	-	MX	-	-
		FDCAN_Classic_Frame_Networking	This example describes how to configure the FDCAN peripheral to send and receive classic CAN frames between two FDCAN units.	-	-	MX	-	-
		FDCAN_Com_IT	This example describes how to configure the FDCAN peripheral to achieve interrupt process communication between two FDCAN units.	-	-	MX	-	-
		FDCAN_Com_Polling	This example describes how to configure the FDCAN peripheral to achieve polling process communication between two FDCAN units.	-	-	MX	-	-
		FDCAN_Loopback	This example describes how to configure the FDCAN peripheral to operate in loopback mode.	-	-	MX	-	-
		FDCAN_Power_Down	This example describes the functionality of the power down mode in the FDCAN peripheral.	-	-	MX	-	-
	FLASH	FLASH_EDATA_EraseProgram	How to configure and use the FLASH HAL API to erase and program the FLASH high-cycle data area.	-	-	MX	MX	-
		FLASH_EraseProgram	How to configure and use the FLASH HAL API to erase and program the internal at the beginning of the main program the HAL_Init() function is called to reset all the peripherals, initialize the flash interface and the systick.	-	-	MX	MX	MX
		FLASH_EraseProgram_TrustZone	How to configure and use the FLASH HAL API to erase and program the internal flash memory when TrustZone security is activated **(Option bit TZEN=B4)**.	-	-	MX	-	-
		FLASH_OBK_EraseProgram	How to configure and use the FLASH HAL API to erase and program the FLASH OBKeys: secure key storage area.	-	-	MX	-	-
		FLASH_SwapBanks	Guide through the configuration steps to program internal flash memory bank 1 and bank 2, and to swap between both of them by mean of the FLASH HAL API.	-	-	X	-	-
	FMAC	FMAC_IIR_PollingToDMA	How to use the FMAC peripheral to perform an IIR filter from polling mode to DMA mode.	-	-	MX	-	-
	FMC	FMC_NOR	This example describes how to configure the FMC controller to access the NOR memory.	MX	-	-	-	-
		FMC_SRAM	This example describes how to configure the FMC controller to access the SRAM memory.	MX	-	-	-	-
	GPIO	GPIO_EXTI	How to configure external interrupt lines.	-	-	MX	MX	-
		GPIO_IOToggle	How to configure and use GPIOs through the HAL API.	-	MX	MX	MX	MX

Level	Module Name	Project Name	Description	STM32H5_CUSTOM_HW	STM32H573I-DK	NUCLEO-H563ZI	NUCLEO-H533RE	NUCLEO-H503RB
Examples	GPIO	GPIO_IOToggle_TrustZone	How to use HAL GPIO to toggle secure and unsecure IOs when TrustZone security is activated (Option bit TZEN=B4).	-	-	MX	-	-
	GTZC	GTZC_MPCWM_IllegalAccess_TrustZone	- How to use GTZC MPCWM-TZIC to build any example when TrustZone security is activated **(Option bit TZEN=0xB4)**.	-	MX	-	-	-
		GTZC_TZSC_MPCBB_TrustZone	How to use HAL GTZC MPCBB to build any example with illegal access detection when TrustZone security is activated **(Option bit TZEN=B4)**.	-	-	MX	MX	-
	HAL	HAL_TimeBase_RTC_ALARM	How to customize HAL using RTC alarm as main source of time base, instead of Systick.	-	-	MX	-	-
		HAL_TimeBase_RTC_WKUP	How to customize HAL using RTC wakeup as main source of time base, instead of Systick.	-	-	MX	-	-
		HAL_TimeBase_TIM	How to customize HAL using a general-purpose timer as main source of time base instead of Systick.	-	-	MX	-	-
	HASH	HASH_HMAC_SHA2_384	This example provides a short description of how to use the HASH peripheral to hash data using long key and SHA_384 Algorithm.	-	-	MX	-	-
		HASH_HMAC_SHA2_512	This example provides a short description of how to use the HASH peripheral to hash data using long key and SHA_512 Algorithm.	-	-	MX	MX	-
		HASH_SHA256	This example provides a short description of how to use the HASH peripheral to hash data using SHA_256 Algorithm.	-	-	MX	-	-
		HASH_SHA384	This example provides a short description of how to use the HASH peripheral to hash data using SHA_384 Algorithm.	-	MX	MX	MX	-
		HASH_SHA512	This example provides a short description of how to use the HASH peripheral to hash data using SHA512 Algorithms.	-	-	MX	-	-
	I2C	I2C_Sensor_Private_Command_IT	How to handle I2C data buffer transmission/reception between STM32H5xx Nucleo and X-NUCLEO-IKS4A1, using an interrupt.	-	-	-	MX	-
		I2C_TwoBoards_AdvComIT	How to handle several I2C data buffer transmission/reception between a master and a slave device, using an interrupt.	-	-	-	-	MX
		I2C_TwoBoards_ComDMA	How to handle I2C data buffer transmission/reception between two boards via DMA.	-	-	MX	-	-
		I2C_TwoBoards_ComIT	How to handle I2C data buffer transmission/reception between two boards, using an interrupt.	-	-	-	-	MX
		I2C_TwoBoards_ComPolling	How to handle I2C data buffer transmission/reception between two boards, in polling mode.	-	-	MX	-	-

Level	Module Name	Project Name	Description	STM32H5_CUSTOM_HW	STM32H573I-DK	NUCLEO-H563ZI	NUCLEO-H533RE	NUCLEO-H503RB
Examples	I2C	I2C_TwoBoards_MultiMasterIT_Master	How to handle I2C data buffer communication between two boards, using an interrupt and two controllers and one target.	-	-	-	MX	-
		I2C_TwoBoards_MultiMasterIT_Slave	How to handle I2C data buffer communication between two boards, using an interrupt and two controllers and one target.	-	-	-	MX	-
		I2C_TwoBoards_RestartAdvComIT	How to perform multiple I2C data buffer transmission/reception between two boards, in interrupt mode and with restart condition.	-	-	MX	-	-
		I2C_TwoBoards_RestartComIT	How to handle single I2C data buffer transmission/reception between two boards, in interrupt mode and with restart condition.	-	-	-	-	MX
		I2C_WakeUpFromStop	How to handle I2C data buffer transmission/reception between two boards, using an interrupt when the device is in Stop mode.	-	-	MX	-	-
	I3C	I3C_Controller_Direct_Command_DMA	How to handle a Direct Command procedure between an I3C controller and an I3C target, using DMA.	-	-	MX	-	-
		I3C_Controller_ENTDAA_IT	How to handle an ENTDAA procedure between an I3C controller and one or more I3C targets.	-	-	-	-	MX
		I3C_Controller_HotJoin_IT	How to handle a HOTJOIN procedure between an I3C controller and I3C targets.	-	-	MX	-	-
		I3C_Controller_I2C_ComDMA	How to handle I2C communication as I3C controller data buffer transmission/reception between two boards, using DMA.	-	-	-	-	MX
		I3C_Controller_IBI_Wakeup_IT	How to handle an in-band-interrupt event between an I3C controller in low-power mode and I3C targets.	-	-	MX	-	-
		I3C_Controller_InBandInterrupt_IT	How to handle an in-band-interrupt event between an I3C controller and I3C targets.	-	-	-	-	MX
		I3C_Controller_Private_Command_IT	How to handle I3C as controller data buffer transmission/reception between two boards, using interrupt.	-	-	-	-	MX
		I3C_Controller_Sensor_IBI	How to handle the I3C controller to obtain accelerometer, gyroscope, and temperature data from the LSM6DSV16X sensor every time an IBI event occurs.	-	-	-	MX	-
		I3C_Controller_Switch_To_Target	How to handle a controller role request Direct Command procedure between an I3C controller and an I3C target, using interrupt.	-	-	MX	-	-
		I3C_Controller_WakeUpFromStop	How to handle I3C as controller data buffer transmission/reception between two boards, where the target is in Stop mode, using interrupt.	-	-	-	-	MX
		I3C_Sensor_Direct_Command_DMA	How to handle a Direct Command procedure between STM32H5xx Nucleo and X-NUCLEO-IKS4A1, using DMA.	-	-	-	MX	MX

Level	Module Name	Project Name	Description	STM32H5_CUSTOM_HW	STM32H573I-DK	NUCLEO-H563ZI	NUCLEO-H533RE	NUCLEO-H503RB
Examples	I3C	I3C_Sensor_Private_Command_IT	How to handle I3C as controller data buffer transmission/reception between STM32H5xx Nucleo and X-NUCLEO-IKS4A1, using interrupt.	-	-	-	MX	X
		I3C_Target_Direct_Command_DMA	How to handle a Direct Command procedure between an I3C controller and an I3C target, using a controller in DMA.	-	-	MX	-	-
		I3C_Target_ENTDAA_IT	How to handle an ENTDAA procedure between an I3C controller and one or more I3C targets.	-	-	-	-	MX
		I3C_Target_HotJoin_IT	How to handle a HOTJOIN procedure to an I3C controller.	-	-	MX	-	-
		I3C_Target_I2C_ComDMA	How to handle I2C data buffer transmission/reception between two boards, using DMA.	-	-	-	-	MX
		I3C_Target_IBI_Wakeup_IT	How to handle a in-band-interrupt procedure to an I3C controller in Stop Mode.	-	-	MX	-	-
		I3C_Target_InBandInterrupt_IT	How to handle an in-band-interrupt event to an I3C controller.	-	-	-	-	MX
		I3C_Target_Private_Command_IT	How to handle I3C as target data buffer transmission/reception between two boards, using interrupt.	-	-	-	-	MX
		I3C_Target_Switch_To_Controller	How to handle a controller role request procedure to an I3C controller.	-	-	MX	-	-
		I3C_Target_WakeUpFromStop	How to handle I3C as target data buffer transmission/reception between two boards, where target is in Stop mode, using interrupt.	-	-	-	-	MX
	ICACHE	ICACHE_External_Memory_Remap	How to execute code from an external clash remapped region configured through the ICACHE HAL driver.	-	MX	-	-	-
	IWDG	IWDG_Reset	How to handle the IWDG reload counter and simulate a software fault that generates an MCU IWDG reset after a preset laps of time.	-	-	MX	-	-
		IWDG_WindowMode	How to periodically update the IWDG reload counter and simulate a software fault that generates an MCU IWDG reset after a preset laps of time.	-	-	-	-	MX
	LPTIM	LPTIM_PWM_LSE	How to configure and use, through the HAL LPTIM API, the LPTIM peripheral using LSE as counter clock, to generate a PWM signal, in a low-power mode.	-	-	MX	-	-
		LPTIM_Timeout	How to implement, through the HAL LPTIM API, a timeout with the LPTIMER peripheral, to wake up the system from a low-power mode.	-	-	-	MX	-
	OCTOSPI	OSPI_NOR_AutoPolling_DTR	How to use an OSPI NOR memory in automatic polling mode.	-	MX	-	-	-

Level	Module Name	Project Name	Description	STM32H5_CUSTOM_HW	STM32H573I-DK	NUCLEO-H563ZI	NUCLEO-H533RE	NUCLEO-H503RB
Examples	OCTOSPI	OSPI_NOR_MemoryMapped	How to use a OSPI NOR memory in memory-mapped mode.	-	MX	-	-	-
		OSPI_NOR_ReadWrite_DMA_DTR	How to use a OSPI NOR memory in DMA mode.	-	MX	-	-	-
	OPAMP	OPAMP_Follower	How to configure the OPAMP peripheral in follower mode interconnected with DAC and COMP.	-	-	-	-	MX
	OTFDEC	OTFDEC_Data_Decrypt	This example describes how to decrypt data (encrypted with CRYPT peripheral) located on the OCTOSPI external flash using the OTFDEC peripheral. The LEDs of the STM32H573I-DK board are used to monitor the status as following: LED_GREEN is ON when the checked data is correct.	-	MX	-	-	-
	PKA	PKA_ECCDoubleBaseLadder	How to use the PKA to run ECC Double base ladder operation. This project is targeted to run on STM32H573IIKxQ devices on STM32H573I-DK board from STMicroelectronics.	-	MX	-	-	-
		PKA_ECCProjective2Affine	How to use the PKA to run ECC projective to affine operation. This project is targeted to run on STM32H573IIKxQ devices on STM32H573I-DK board from STMicroelectronics.	-	MX	-	-	-
		PKA_ECDSA_Sign	How to compute a signed message regarding the elliptic curve digital signature algorithm (ECDSA).	-	MX	-	MX	-
		PKA_ECDSA_Verify	How to determine if a given signature is valid regarding the elliptic curve digital signature algorithm (ECDSA).	-	-	MX	-	-
		PKA_ModExpProtected_IT	How to use the PKA to run protected modular exponentiation operation. At the beginning of the main program the HAL_Init() function is called to reset all the peripherals, initialize the flash interface and the systick.	-	-	-	MX	-
		PKA_ModularExponentiation_IT	How to use the PKA peripheral to execute modular exponentiation. This allows ciphering/deciphering a text in interrupt mode.	-	MX	-	-	-
	PWR	PWR_IO_Retention	How to use the standby IO retention.	-	-	MX	-	-
		PWR_LPMODE_RTC	How to enter the system to different available low power modes and wake up from these modes by using an interrupt from RTC wakeup timer.	-	-	MX	MX	-
		PWR_SLEEP	How to enter Sleep mode and wake up from this mode by using an interrupt.	-	-	MX	-	-
		PWR_STANDBY	How to enter Standby mode and wake up from this mode by using an external reset or the WKUP pin.	-	-	MX	-	MX
		PWR_STANDBY_RTC	How to enter Standby mode and wake-up from this mode by using an external reset or the RTC wakeup timer.	-	-	MX	-	MX
	RAMCFG	RAMCFG_ECC_Error_Generation	How to configure and use the RAMCFG HAL API to manage ECC errors via RAMCFG peripheral.	-	-	MX	MX	-

Level	Module Name	Project Name	Description	STM32H5_CUSTOM_HW	STM32H573I-DK	NUCLEO-H563ZI	NUCLEO-H533RE	NUCLEO-H503RB
Examples	RAMCFG	RAMCFG_WriteProtection	How to configure and use the RAMCFG HAL API to configure RAMCFG SRAM write protection page.	-	-	MX	MX	MX
	RCC	RCC_CRS_Synchronization_IT	Configuration of the clock recovery system (CRS) in interrupt mode, using the RCC/CRS HAL APIs.	-	-	MX	MX	-
		RCC_CRS_Synchronization_Polling	Configuration of the clock recovery system (CRS) in polling mode, using the RCC HAL API.	-	-	MX	-	-
		RCC_ClockConfig	Configuration of the system clock (SYSCLK) and modification of the clock settings in Run mode, using the RCC HAL API.	-	MX	MX	MX	MX
		RCC_LSEConfig	Enabling/disabling of the low-speed external (LSE) RC oscillator (about 32 KHz) at runtime, using the RCC HAL API.	-	-	-	-	MX
		RCC_LSIConfig	How to enable/disable the low-speed internal (LSI) RC oscillator (about 32 KHz) at runtime, using the RCC HAL API.	-	-	MX	-	-
	RNG	RNG_MultiRNG	Configuration of the RNG using the HAL API. This example uses the RNG to generate 32-bit long random numbers.	-	-	MX	-	-
		RNG_MultiRNG_IT	Configuration of the RNG using the HAL API. This example uses RNG interrupts to generate 32-bit long random numbers.	-	-	-	MX	MX
	RTC	RTC_ActiveTamper	Configuration of the active tamper detection with backup registers erase.	-	-	MX	-	-
		RTC_Alarm	Configuration and generation of an RTC alarm using the RTC HAL API.	-	-	MX	MX	-
		RTC_Calendar	Configuration of the calendar using the RTC HAL API.	-	MX	-	-	-
		RTC_LSI	Use of the LSI clock source autocalibration to get a precise RTC clock.	-	-	MX	-	-
		RTC_LowPower_STANDBY_WUT	How to periodically enter and wake up from STANDBY mode thanks to the RTC wake-up timer (WUT).	-	-	MX	-	-
		RTC_Tamper	Configuration of the tamper detection with backup registers erase.	-	-	-	MX	MX
		RTC_TimeStamp	Configuration of the RTC HAL API to demonstrate the timestamp feature.	-	-	-	-	MX
		RTC_TrustZone	How to configure the TrustZone-aware RTC peripheral when TrustZone security is activated (Option bit TZEN=B4): some features of the RTC can be secure while the others are non-secure.	-	-	MX	-	-

Level	Module Name	Project Name	Description	STM32H5_CUSTOM_HW	STM32H573I-DK	NUCLEO-H563ZI	NUCLEO-H533RE	NUCLEO-H503RB
Examples	SAI	SAI_AudioPlay	This example shows how to use the SAI HAL API to play an audio file using the DMA circular mode and how to handle the buffer update.	-	MX	-	-	-
	SD	SD_ReadWrite_DMALinkedList	This example performs some write and read transfers to SD card with SDMMC IP internal DMA mode based on Linked list feature.	-	MX	-	-	-
	SMBUS	SMBUS_TwoBoards_ComIT_Master	How to handle SMBUS data buffer transmission/reception between two boards, in interrupt mode.	-	-	-	-	MX
		SMBUS_TwoBoards_ComIT_Slave	How to handle SMBUS data buffer transmission/reception between two boards, in interrupt mode.	-	-	-	-	MX
	SPI	SPI_FullDuplex_ComDMA_Master	Data buffer transmission/reception between two boards via SPI using DMA.	-	-	-	-	MX
		SPI_FullDuplex_ComDMA_Slave	Data buffer transmission/reception between two boards via SPI using DMA.	-	-	-	-	MX
		SPI_FullDuplex_ComIT_Master	Data buffer transmission/reception between two boards via SPI using interrupt mode.	-	-	MX	-	-
		SPI_FullDuplex_ComIT_Slave	Data buffer transmission/reception between two boards via SPI using interrupt mode.	-	-	MX	-	-
		SPI_FullDuplex_ComPolling_Master	Data buffer transmission/reception between two boards via SPI using polling mode.	-	-	-	-	MX
		SPI_FullDuplex_ComPolling_Slave	Data buffer transmission/reception between two boards via SPI using polling mode.	-	-	-	-	MX
		SPI_FullDuplex_CrcComIT_Master	Data buffer transmission/reception with CRC between two boards via SPI using IT.	-	-	-	MX	-
		SPI_FullDuplex_CrcComIT_Slave	Data buffer transmission/reception with CRC between two boards via SPI using IT.	-	-	-	MX	-
	TIM	TIM_InputCapture	How to use the TIM peripheral to measure an external signal frequency.	-	-	MX	-	-
		TIM_OCActive	Configuration of the TIM peripheral in output compare active mode (when the counter matches the capture/compare register, the corresponding output pin is set to its active state).	-	-	MX	-	-
		TIM_OCInactive	Configuration of the TIM peripheral in output compare inactive mode with the corresponding Interrupt requests for each channel.	-	-	-	-	MX
		TIM_OCToggle	Configuration of the TIM peripheral to generate four different signals at four different frequencies.	-	-	-	-	MX

Level	Module Name	Project Name	Description	STM32H5_CUSTOM_HW	STM32H573I-DK	NUCLEO-H563ZI	NUCLEO-H533RE	NUCLEO-H503RB
Examples	TIM	TIM_PWMInput	How to use the TIM peripheral to measure the frequency and duty cycle of an external signal.	-	-	MX	-	-
		TIM_PWMOutput	This example shows how to configure the TIM peripheral in PWM (pulse width modulation) mode.	-	-	MX	-	-
	UART	UART_AutoBaudrate_Detection	How to use the HAL UART API for detecting automatically the baud rate.	-	-	-	MX	-
		UART_HyperTerminal_IT	UART transmission (transmit/receive) in Interrupt mode between a board and an HyperTerminal PC application.	-	-	-	MX	-
		UART_Printf	Re-routing of the C library printf function to the UART.	-	-	-	-	MX
		UART_ReceptionToIdle_CircularDMA	How to use the HAL UART API for reception to IDLE event in circular DMA mode.	-	-	-	-	MX
		UART_TwoBoards_ComDMA	UART transmission (transmit/receive) in DMA mode between two boards.	-	-	-	-	MX
		UART_TwoBoards_ComDMAlinkedlist	UART transmission (transmit/receive) in DMA mode using linkedlist between two boards.	-	-	MX	-	-
		UART_TwoBoards_ComIT	UART transmission (transmit/receive) in interrupt mode between two boards.	-	-	-	-	MX
		UART_TwoBoards_ComPolling	UART transmission (transmit/receive) in polling mode between two boards.	-	-	MX	-	-
	USART	USART_SlaveMode	This example describes an USART-SPI communication (transmit/receive) between two boards where the USART is configured as a target	-	-	-	-	MX
		USART_SlaveMode_DMA	This example describes an USART-SPI communication (transmit/receive) with DMA between two boards where the USART is configured as a target.	-	-	-	-	MX
	WWDG	WWDG_Example	Configuration of the HAL API to periodically update the WWDG counter and simulate a software fault that generates an MCU WWDG reset when a predefined time period has elapsed.	-	-	-	-	MX
	Total number of examples: 191			**2**	**26**	**77**	**36**	**50**
Examples_LL	ADC	ADC_AnalogWatchdog_Init	How to use an ADC peripheral with an ADC analog watchdog to monitor a channel and detect when the corresponding conversion data is outside the window thresholds.	-	-	MX	-	-
		ADC_Oversampling_Init	How to use an ADC peripheral with oversampling.	-	-	-	-	MX

Level	Module Name	Project Name	Description	STM32H5_CUSTOM_HW	STM32H573I-DK	NUCLEO-H563ZI	NUCLEO-H533RE	NUCLEO-H503RB
Examples_LL	ADC	ADC_SingleConversion_TriggerSW_IT_Init	How to use ADC to convert a single channel at each SW start, conversion performed using programming model: interrupt.	-	-	MX	-	-
		ADC_SingleConversion_TriggerSW_Init	How to use ADC to convert a single channel at each SW start, conversion performed using programming model: polling.	-	-	-	-	MX
	COMP	COMP_CompareGpioVsVrefInt_IT_Init	How to use a comparator peripheral to compare a voltage level applied on a GPIO pin to the the internal voltage reference (VrefInt), in interrupt mode.	-	-	-	-	MX
	CORTEX	CORTEX_MPU	Presentation of the MPU features. This example configures MPU attributes of different MPU regions then configures a memory area as privileged read-only, and attempts to perform read and write operations in different modes.	-	-	MX	-	MX
	CRC	CRC_CalculateAndCheck	How to configure the CRC calculation unit to compute a CRC code for a given data buffer, based on a fixed generator polynomial (default value 0x4C11DB7). The peripheral initialization is done using LL unitary service functions for optimization purposes (performance and size).	-	-	MX	-	-
		CRC_UserDefinedPolynomial	How to configure and use the CRC calculation unit to compute an 8-bit CRC code for a given data buffer, based on a user-defined generating polynomial. The peripheral initialization is done using LL unitary service functions for optimization purposes (performance and size).	-	-	-	-	MX
	CRS	CRS_Synchronization_IT	How to configure the clock recovery service in IT mode through the STM32H5xx CRS LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	-	-	MX	-	-
		CRS_Synchronization_Polling	How to configure the clock recovery service in polling mode through the STM32H5xx CRS LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	-	-	-	-	MX
	DMA	DMA_CopyFromFlashToMemory_Init	How to use a DMA channel to transfer a word data buffer from flash memory to embedded SRAM. The peripheral initialization uses LL initialization functions to demonstrate LL init usage.	-	-	MX	-	MX
	EXTI	EXTI_ToggleLedOnIT_Init	This example describes how to configure the EXTI and use GPIOs to toggle the user LEDs available on the board when a user button is pressed. This example is based on the STM32H5xx LL API. Peripheral initialization is done using LL initialization function to demonstrate LL init usage.	-	-	-	MX	MX
	GPIO	GPIO_InfiniteLedToggling_Init	How to configure and use GPIOs to toggle the on-board user LEDs every 250 ms. This example is based on the STM32H5xx LL API. The peripheral is initialized with LL initialization function to demonstrate LL init usage.	-	-	MX	-	-
	I2C	I2C_OneBoard_AdvCommunication_DMAAndIT_Init	How to exchange data between an I2C controller device in DMA mode and an I2C target device in interrupt mode. The peripheral is initialized with LL unitary service functions to optimize for performance and size.	-	-	MX	-	-
		I2C_OneBoard_Communication_DMAAndIT_Init	How to transmit data bytes from an I2C controller device using DMA mode to an I2C target device using interrupt mode. The peripheral is initialized with LL unitary service functions to optimize for performance and size.	-	-	MX	-	-
		I2C_OneBoard_Communication_IT_Init	How to handle the reception of one data byte from an I2C target device by an I2C controller device. Both devices operate in interrupt mode. The peripheral is initialized with LL initialization function to demonstrate LL init usage.	-	-	MX	-	-
		I2C_OneBoard_Communication_PollingAndIT_Init	How to transmit data bytes from an I2C controller device using polling mode to an I2C target device using interrupt mode. The peripheral is initialized with LL unitary service functions to optimize for performance and size.	-	-	MX	-	-

Level	Module Name	Project Name	Description	STM32H5_CUSTOM_HW	STM32H573I-DK	NUCLEO-H563ZI	NUCLEO-H533RE	NUCLEO-H503RB
Examples_LL	I2C	I2C_TwoBoards_MasterRx_SlaveTx_IT_Init	How to handle the reception of one data byte from an I2C target device by an I2C controller device. Both devices operate in interrupt mode. The peripheral is initialized with LL unitary service functions to optimize for performance and size.	-	-	-	-	MX
		I2C_TwoBoards_MasterTx_SlaveRx_DMA_Init	How to transmit data bytes from an I2C controller device using DMA mode to an I2C target device using DMA mode. The peripheral is initialized with LL unitary service functions to optimize for performance and size.	-	-	-	-	MX
		I2C_TwoBoards_MasterTx_SlaveRx_Init	How to transmit data bytes from an I2C controller device using polling mode to an I2C target device using interrupt mode. The peripheral is initialized with LL unitary service functions to optimize for performance and size.	-	-	-	-	MX
	I3C	I3C_Controller_Direct_Command_IT	How to handle a Direct Command procedure between an I3C Controller and an I3C target, using IT.	-	-	-	-	MX
		I3C_Controller_Direct_Command_Polling	How to handle a Direct Command procedure between an I3C Controller and an I3C target, using polling.	-	-	-	-	MX
		I3C_Controller_HotJoin_IT	How to handle a HOTJOIN procedure between an I3C controller and I3C targets.	-	-	-	-	MX
		I3C_Controller_InBandInterrupt_IT	How to handle an in-band-interrupt event between an I3C controller and I3C targets.	-	-	MX	-	-
		I3C_Controller_Private_Command_IT	How to handle I3C as controller data buffer transmission/reception between two boards, using interrupt.	-	-	MX	-	-
		I3C_Controller_WakeUpFromStop	How to handle I3C as controller data buffer transmission/reception between a target in Stop mode, using interrupt.	-	-	MX	-	-
		I3C_Target_Direct_Command_IT	How to handle a Direct Command procedure between an I3C controller and an I3C target, using controller in interrupt.	-	-	-	-	MX
		I3C_Target_Direct_Command_Polling	How to handle a Direct Command procedure between an I3C controller and an I3C target, using controller in polling.	-	-	-	-	MX
		I3C_Target_HotJoin_IT	How to handle a HOTJOIN procedure to an I3C controller.	-	-	-	-	MX
		I3C_Target_InBandInterrupt_IT	How to handle an in-band-interrupt event to an I3C controller.	-	-	MX	-	-
		I3C_Target_Private_Command_IT	How to handle I3C as target data buffer transmission/reception between two boards, using interrupt.	-	-	MX	-	-
		I3C_Target_WakeUpFromStop	How to handle I3C as target data buffer transmission/reception between two boards, using interrupt.	-	-	MX	-	-

Level	Module Name	Project Name	Description	STM32H5_CUSTOM_HW	STM32H573I-DK	NUCLEO-H563ZI	NUCLEO-H533RE	NUCLEO-H503RB
Examples_LL	IWDG	IWDG_RefreshUntilUserEvent_Init	How to configure the IWDG peripheral to ensure periodical counter update and generate an MCU IWDG reset when a USER push-button is pressed. The peripheral is initialized with LL unitary service functions to optimize for performance and size.	-	-	MX	-	-
	LPTIM	LPTIM_PulseCounter_Init	How to use the LPTIM peripheral in counter mode to generate a PWM output signal and update its duty cycle. This example is based on the STM32H5xx LPTIM LL API. The peripheral is initialized with LL initialization function to demonstrate LL init usage.	-	-	-	-	MX
	LPUART	LPUART_WakeUpFromStop_Init	Configuration of GPIO and LPUART peripherals to allow characters received on LPUART_RX pin to wake up the MCU from low-power mode. This example is based on the LPUART LL API. The peripheral initialization uses LL initialization function to demonstrate LL init usage.	-	-	-	-	MX
	PWR	PWR_EnterStandbyMode	How to enter the Standby mode and wake up from this mode by using an external reset or a wakeup pin.	-	-	MX	-	-
		PWR_EnterStopMode	How to enter Stop mode.	-	-	-	-	MX
	RCC	RCC_OutputSystemClockOnMCO	Configuration of MCO pin (PA8) to output the system clock.	-	-	-	-	MX
		RCC_UseHSEasSystemClock	Use of the RCC LL API to start the HSE and use it as system clock.	-	-	MX	-	-
		RCC_UseHSI_PLLasSystemClock	Modification of the PLL parameters in run time.	-	-	-	-	MX
	RTC	RTC_Alarm_Init	Configuration of the RTC LL API to configure and generate an alarm using the RTC peripheral. The peripheral initialization uses the LL initialization function.	-	-	-	-	MX
		RTC_Calendar_Init	Configuration of the LL API to set the RTC calendar. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	-	-	MX	-	-
		RTC_ExitStandbyWithWakeUpTimer_Init	How to periodically enter and wake up from STANDBY mode thanks to the RTC wakeup timer (WUT).	-	-	MX	-	-
		RTC_Tamper_Init	Configuration of the tamper using the RTC LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	-	-	-	-	MX
		RTC_TimeStamp_Init	Configuration of the timestamp using the RTC LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	-	-	MX	-	-
	SPI	SPI_OneBoard_HalfDuplex_DMA_Init	Configuration of GPIO and SPI peripherals to transmit bytes from an SPI leader device to an SPI follower device in DMA mode. This example is based on the STM32H5xx SPI LL API. The peripheral initialization uses the LL initialization function to demonstrate LL init usage.	-	-	-	MX	-

Level	Module Name	Project Name	Description	STM32H5_CUSTOM_HW	STM32H573I-DK	NUCLEO-H563ZI	NUCLEO-H533RE	NUCLEO-H503RB
Examples_LL	SPI	SPI_OneBoard_HalfDuplex_IT_Init	Configuration of GPIO and SPI peripherals to transmit bytes from an SPI leader device to an SPI follower device in interrupt mode. This example is based on the STM32H5xx SPI LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	-	-	MX	-	-
		SPI_TwoBoards_FullDuplex_DMA_Master_Init	Data buffer transmission and reception via SPI using DMA mode. This example is based on the STM32H5xx SPI LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	-	-	-	MX	-
		SPI_TwoBoards_FullDuplex_DMA_Slave_Init	Data buffer transmission and reception via SPI using DMA mode. This example is based on the STM32H5xx SPI LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	-	-	-	MX	-
		SPI_TwoBoards_FullDuplex_IT_Master_Init	Data buffer transmission and receptionvia SPI using Interrupt mode. This example is based on the STM32H5xx SPI LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	-	-	-	-	MX
		SPI_TwoBoards_FullDuplex_IT_Slave_Init	Data buffer transmission and receptionvia SPI using Interrupt mode. This example is based on the STM32H5xx SPI LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	-	-	-	-	MX
	TIM	TIM_BreakAndDeadtime_Init	Configuration of the TIM peripheral to generate three center-aligned PWM and complementary PWM signals, insert a defined deadtime value, use the break feature, and lock the break and dead-time configuration.	-	-	-	-	MX
		TIM_InputCapture_Init	Use of the TIM peripheral to measure a periodic signal frequency provided either by an external signal generator or by another timer instance. This example is based on the STM32H5xx TIM LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	-	-	-	-	MX
		TIM_OnePulse_Init	Configuration of a timer to generate a positive pulse in output compare mode with a length of tPULSE and after a delay of tDELAY. This example is based on the STM32H5xx TIM LL API. The peripheral initialization uses LL initialization function to demonstrate LL Init.	-	-	MX	-	-
		TIM_OutputCompare_Init	Configuration of the TIM peripheral to generate an output waveform in different output compare modes. This example is based on the STM32H5xx TIM LL API.	-	-	-	-	MX
		TIM_PWMOutput_Init	Use of a timer peripheral to generate a PWM output signal and update the PWM duty cycle. This example is based on the STM32H5xx TIM LL API. The peripheral initialization uses LL initialization function to demonstrate LL Init.	-	-	MX	-	-
		TIM_TimeBase_Init	Configuration of the TIM peripheral to generate a timebase. This example is based on the STM32H5xx TIM LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	-	-	-	-	MX
	USART	USART_Communication_Rx_IT_Continuous_Init	This example shows how to configure GPIO and USART peripheral for continuously receiving characters from HyperTerminal (PC) in asynchronous mode using interrupt mode. Peripheral initialization is done using LL unitary services functions for optimization purpose (performance and size).	-	-	MX	-	-
		USART_Communication_Rx_IT_Continuous_VCP_Init	This example shows how to configure GPIO and USART peripheral for continuously receiving characters from HyperTerminal (PC) in asynchronous mode using interrupt mode. Peripheral initialization is done using LL unitary services functions for optimization purpose (performance and size).	-	-	-	-	MX

Level	Module Name	Project Name	Description	STM32H5_CUSTOM_HW	STM32H573I-DK	NUCLEO-H563ZI	NUCLEO-H533RE	NUCLEO-H503RB
Examples_LL	USART	USART_Communication_Rx_IT_Init	This example shows how to configure GPIO and USART peripheral for receiving characters from HyperTerminal (PC) in asynchronous mode using interrupt mode. Peripheral initialization is done using LL initialization function to demonstrate LL init usage.	-	-	MX	-	-
		USART_Communication_TxRx_DMA	This example shows how to configure GPIO, USART2, and GPDMA1 peripherals to send characters asynchronously to/from an HyperTerminal (PC) in DMA mode. This example is based on STM32H5xx USART and DMA LL API. Peripheral initializations are done using LL unitary services functions for optimization purpose (performance and size).	-	-	-	MX	-
		USART_Communication_Tx_IT_Init	This example shows how to configure GPIO and USART peripheral to send characters asynchronously to HyperTerminal (PC) in Interrupt mode. This example is based on STM32H5xx USART LL API. Peripheral initialization is done using LL unitary services functions for optimization purpose (performance and size).	-	-	MX	-	-
		USART_Communication_Tx_IT_VCP_Init	This example shows how to configure GPIO and USART peripheral to send characters asynchronously to HyperTerminal (PC) in Interrupt mode. This example is based on STM32H5xx USART LL API. Peripheral initialization is done using LL unitary services functions for optimization purpose (performance and size).	-	-	MX	-	-
		USART_Communication_Tx_Init	This example shows how to configure GPIO and USART peripherals to send characters asynchronously to an HyperTerminal (PC) in Polling mode. If the transfer could not be completed within the allocated time, a timeout allows to exit from the sequence with a Timeout error code. This example is based on STM32H5xx USART LL API. Peripheral initialization is done using LL unitary services functions for optimization purpose (performance and size).	-	-	-	-	MX
		USART_Communication_Tx_VCP_Init	This example shows how to configure GPIO and USART peripherals to send characters asynchronously to an HyperTerminal (PC) in Polling mode. If the transfer could not be completed within the allocated time, a timeout allows to exit from the sequence with a Timeout error code. This example is based on STM32H5xx USART LL API. Peripheral initialization is done using LL unitary services functions for optimization purpose (performance and size).	-	-	-	-	MX
	UTILS	UTILS_ConfigureSystemClock	Use of UTILS LL API to configure the system clock using PLL with HSI as source clock.	-	-	MX	-	MX
		UTILS_ReadDeviceInfo	This example reads the UID, Device ID and Revision ID and saves them into a global information buffer.	-	-	MX	MX	MX
	WWDG	WWDG_RefreshUntilUserEvent_Init	Configuration of the WWDG to periodically update the counter and generate an MCU WWDG reset when a user button is pressed. The peripheral initialization uses the LL unitary service functions for optimization purposes (performance and size).	-	-	-	-	MX
	Total number of examples_ll: 74			**0**	**0**	**32**	**6**	**36**
Examples_MIX	ADC	ADC_SingleConversion_TriggerSW_IT	How to use ADC to convert a single channel at each SW start, conversion performed using programming model: interrupt.	-	-	MX	-	MX
	DMA	DMA_FLASHToRAM	How to use a DMA to transfer a word data buffer from flash memory to embedded SRAM through the STM32H5xx DMA HAL and LL API. The LL API is used for performance improvement.	-	-	MX	-	MX
	I3C	I3C_Controller_Private_Command_IT	How to handle I3C as controller data buffer transmission/reception between two boards, using interrupt.	-	-	MX	-	-

Level	Module Name	Project Name	Description	STM32H5_CUSTOM_HW	STM32H573I-DK	NUCLEO-H563ZI	NUCLEO-H533RE	NUCLEO-H503RB
Examples_MIX	I3C	I3C_Target_Private_Command_IT	How to handle I3C as target data buffer transmission/reception between two boards, using interrupt.	-	-	MX	-	-
	PWR	PWR_STOP	How to enter the STOP mode and wake up from this mode by using external reset or wakeup interrupt (all the RCC function calls use RCC LL API for minimizing footprint and maximizing performance).	-	-	MX	-	MX
	SPI	SPI_FullDuplex_ComPolling_Master	Data buffer transmission/reception between two boards via SPI using polling mode.	-	-	MX	-	MX
		SPI_FullDuplex_ComPolling_Slave	Data buffer transmission/reception between two boards via SPI using polling mode.	-	-	MX	-	MX
	TIM	TIM_PWMInput	Use of the TIM peripheral to measure an external signal frequency and duty cycle.	-	-	MX	-	MX
	UART	UART_HyperTerminal_IT	Use of a UART to transmit data (transmit/receive) between a board and an HyperTerminal PC application in interrupt mode. This example describes how to use the USART peripheral through the STM32H5xx UART HAL and LL API, the LL API being used for performance improvement.	-	-	MX	-	-
		UART_HyperTerminal_TxPolling_RxIT	Use of a UART to transmit data (transmit/receive) between a board and an HyperTerminal PC application both in polling and interrupt modes. This example describes how to use the USART peripheral through the STM32H5xx UART HAL and LL API, the LL API being used for performance improvement.	-	-	-	-	MX
	Total number of examples_mix: 16			**0**	**0**	**9**	**0**	**7**
Applications	-	OpenBootloader	This application exploits OpenBootloader middleware to demonstrate how to develop an IAP application and how use it.	-	X	-	-	-
	FPU	FPU_Fractal	This application demonstrates the benefits brought by the STM32H5 floating-point unit (FPU). The Cortex-M33 FPU is an implementation of the single precision variant of the ARMv8-M Floating-point extension, FPv5 architecture.	-	-	-	X	-
	FileX	FX_IAP	This application provides an example of Azure RTOS FileX stack usage on STM32H573I-DK board, it implements an in-application programming (IAP) demonstrating FileX's SD file access capabilities.	-	X	-	-	-
		Fx_Dual_Instance	This application provide user a working example of two storage media managed by two independent instances of FileX/LevelX running on STM32H573I-DK board.	-	MX	-	-	-
		Fx_File_Edit_Standalone	This application provides an example of FileX stack usage on STM32H573I-DK board, running in standalone mode (without ThreadX). It demonstrates how to create a Fat File system on the SD card memory using FileX API.	-	MX	-	MX	-
		Fx_MultiAccess	This application provides an example of Azure RTOS FileX stack usage on STM32H573I-DK board, it demonstrates the FileX's concurrent file access capabilities. The application is designed to execute file operations on the SD card device, the code provides all required software code for handling SD card I/O operations.	-	MX	-	-	-

Level	Module Name	Project Name	Description	STM32H5_CUSTOM_HW	STM32H573I-DK	NUCLEO-H563ZI	NUCLEO-H533RE	NUCLEO-H503RB
Applications	FileX	Fx_NoR_Write_Read_File	This application provides an example of Azure RTOS FileX and LevelX stacks usage on STM32H573I-DK board, it demonstrates how to create a FAT file system on the NOR flash using FileX alongside LevelX. The application is designed to execute file operations on the MX25LM51245G NOR flash device, the code provides all required software code for properly managing it.	-	MX	-	-	-
		Fx_uSD_File_Edit	This application provides an example of Azure RTOS FileX stack usage on STM32H573I-DK board, it shows how to develop a basic SD card file operations application.	-	MX	-	-	-
	NetXDuo	Nx_Iperf	This application provides an example of Azure RTOS NetXDuo stack usage .	-	-	MX	-	-
		Nx_Iperf_wifi	This application is a network traffic tool for measuring TCP and UDP performance with metrics around both throughput and latency.	-	MX	-	-	-
		Nx_MQTT_Client	This application provides an example of Azure RTOS NetX/NetXDuo stack usage.	-	MX	-	-	-
		Nx_Network_Basics_wifi	This application demonstrates WiFi connectivity on MXCHIP EMW3080 module for the STM32H573I-DK board from STMicroelectronics.	-	MX	-	-	-
		Nx_SNTP_Client	This application provides an example of Azure RTOS NetX/NetXDuo stack usage.	-	-	MX	-	-
		Nx_TCP_Echo_Client	This application provides an example of Azure RTOS NetX/NetXDuo stack usage .	-	MX	-	-	-
		Nx_TCP_Echo_Server	This application provides an example of Azure RTOS NetX/NetXDuo stack usage .	-	MX	-	-	-
		Nx_UDP_Echo_Client	This application provides an example of Azure RTOS NetX/NetXDuo stack usage.	-	-	MX	-	-
		Nx_UDP_Echo_Server	This application provides an example of Azure RTOS NetX/NetXDuo stack usage.	-	-	MX	-	-
		Nx_WebServer	This application provides an example of Azure RTOS NetX Duo stack usage on STM32H573G-DK board, it shows how to develop Web HTTP server based application.	-	MX	-	-	-
	ROT	OEMiROT_Appli	This project provides a OEMiROT boot path application example. Boot is performed through OEMiROT boot path after authenticity and the integrity checks of the project firmware and project data images.	-	-	-	-	X
		OEMiROT_Appli_TrustZone	This project provides a OEMiROT boot path application example. Boot is performed through OEMiROT boot path after authenticity and the integrity checks of the project firmware and project data images.	-	X	X	-	-
		OEMiROT_Boot	This project provides an OEMiROT example. OEMiROT boot path performs authenticity and the integrity checks of the project firmware and data images.	-	X	X	X	X

Level	Module Name	Project Name	Description	STM32H5_CUSTOM_HW	STM32H573I-DK	NUCLEO-H563ZI	NUCLEO-H533RE	NUCLEO-H503RB
Applications	ROT	SMAK_Appli	This project provides a secure manager boot path application example. Boot is performed through secure manager boot path after authenticity and integrity checks of the project firmware images.	-	X	-	-	-
		STiROT_Appli	This project provides a STiROT boot path application example. Boot is performed through STiROT boot path after authenticity and the integrity checks of the project firmware and project data images.	-	X	-	-	-
		STiROT_Appli_TrustZone	This project provides a STiROT boot path application example. Boot is performed through STiROT boot path after authenticity and the integrity checks of the project firmware and project data images.	-	X	-	-	-
		STiROT_OEMuROT_Appli	For STiROT_OEMuROT boot path, no dedicated project are provided. The projects OEMiROT_Boot and OEMiROT_Appli_TrustZone are used to demonstrate this boot path.	-	X	-	-	-
	ThreadX	Tx_CMSIS_Wrapper	This application provides an example of CMSIS RTOS adaptation layer for Azure RTOS ThreadX, it shows how to develop an application using the CMSIS RTOS 2 APIs.	-	-	-	-	X
		Tx_FreeRTOS_Wrapper	This application provides an example of Azure RTOS ThreadX stack usage, it shows how to develop an application using the FreeRTOS adaptation layer for ThreadX.	-	-	X	-	-
		Tx_LowPower	This application provides an example of Azure RTOS ThreadX stack usage, it shows how to develop an application using ThreadX lowpower feature.	-	-	MX	MX	-
		Tx_MPU	This application provides an example of Azure RTOS ThreadX stack usage, it shows how to develop an application using the ThreadX Module feature.	-	X	-	-	-
		Tx_SecureLEDToggle_TrustZone	This application provides an example of Azure RTOS ThreadX stack usage, it shows how to develop an application using the ThreadX when the TrustZone feature is enabled (TZEN=B4).	-	-	MX	-	-
		Tx_Thread_Creation	This application provides an example of Azure RTOS ThreadX stack usage, it shows how to develop an application using the ThreadX thread management APIs.	-	MX	-	MX	-
		Tx_Thread_MsgQueue	This application provides an example of Azure RTOS ThreadX stack usage, it shows how to develop an application using the ThreadX message queue APIs.	-	-	MX	-	-
		Tx_Thread_Sync	This application provides an example of Azure RTOS ThreadX stack usage, it shows how to develop an application using the ThreadX synchronization APIs.	-	-	MX	-	-
	USBPD	USBPD_SNK	This application is a USBPD type C Consumer using Azure RTOS.	-	-	MX	-	-
		USBPD_SNK_UX_Device_HID	This application provides an example of Azure RTOS USBX stack usage on STM32H563xx board, it shows how to develop USB Device Human Interface "HID" mouse based application.	-	-	MX	-	-
		USBPD_SRC	This application is a USBPD type C Provider using Azure RTOS.	-	MX	-	-	-

Level	Module Name	Project Name	Description	STM32H5_CUSTOM_HW	STM32H573I-DK	NUCLEO-H563ZI	NUCLEO-H533RE	NUCLEO-H503RB
Applications	USBPD	USBPD_SRC_UX_Host_MSC	This application is a USBPD type C Provider and USB Host using Azure RTOS USBX stack. It shows how to develop a USBPD type C Provider in the case of an USB host application based on Mass Storage "MSC" which is able to enumerate and communicates with a removable usb flash disk.	-	MX	-	-	-
	USBX	Ux_Device_CDC_ACM	This application provides an example of Azure RTOS USBX stack usage on NUCLEO-H533RE board, it shows how to develop USB Device communication Class "CDC_ACM" based application.	-	-	-	MX	MX
		Ux_Device_CDC_ECM	This application provides an example of Azure RTOS CDC_ECM stack usage on STM32H573I-DK board, it shows how to run web HTTP server based application stack over USB interface. The application is designed to load files and web pages stored in SD card using a web HTTP server through USB interface using CDC_ECM class, the code provides all required features to build a compliant web HTTP server. The main entry function tx_application_define() is called by ThreadX during kernel start, at this stage, the USBX initialize the network layer through USBx class (CDC_ECM) also the FileX and the NetXDuo system are initialized, the NX_IP instance and the web HTTP server are created and configured, then the application creates two main threads - app_ux_device_thread_entry (Prio : 10; PreemptionPrio : 10) used to initialize USB DRD HAL PCD driver and start the device.	-	MX	-	-	-
		Ux_Device_DFU	This application provides an example of Azure RTOS USBX stack usage on STM32H573I-DK board, it shows how to develop USB Device firmware upgrade "DFU" based application.	-	MX	-	-	-
		Ux_Device_HID	This application provides an example of Azure RTOS USBX stack usage on STM32H563xx board, it shows how to develop USB Device Human Interface "HID" mouse based application.	-	-	MX	-	-
		Ux_Device_HID_CDC_ACM	This application provides an example of Azure RTOS USBX stack usage on NUCLEO-H563ZI board, it shows how to develop a composite USB Device communication class "HID" and "CDC_ACM" based application.	-	-	MX	-	-
		Ux_Device_HID_Standalone	This application provides an example of Azure RTOS USBX stack usage on STM32H563xx board, it shows how to develop USB Device Human Interface "HID" mouse based bare metal application.	-	-	MX	-	-
		Ux_Device_MSC	This application provides an example of Azure RTOS USBX stack usage on STM32H573I-DK board, it shows how to develop USB Device mass storage class based application. The application is designed to emulate an USB MSC device, the code provides all required device descriptors framework and the associated class descriptor report to build a compliant USB MSC device.	-	MX	-	-	-
		Ux_Host_CDC_ACM	This application provides an example of Azure RTOS USBX stack usage .	-	MX	-	-	-
		Ux_Host_DualClass	This application provides an example of Azure RTOS USBX stack usage.	-	MX	-	-	-
		Ux_Host_HID	This application provides an example of Azure RTOS USBX stack usage.	-	MX	-	-	-
		Ux_Host_HID_CDC_ACM	This application provides an example of Azure RTOS USBX stack usage.	-	MX	-	-	-
		Ux_Host_HID_Standalone	This application provides an example of Azure RTOS USBX stack usage.	-	MX	-	-	-

Level	Module Name	Project Name	Description	STM32H5_CUSTOM_HW	STM32H573I-DK	NUCLEO-H563ZI	NUCLEO-H533RE	NUCLEO-H503RB
Applications	USBX	Ux_Host_MSC	This application provides an example of Azure RTOS USBX stack usage. It shows how to develop USB Host Mass Storage "MSC" able to enumerate and communicates with a removable usb flash disk.	-	MX	-	-	-
	Total number of applications: 58			**0**	**32**	**16**	**6**	**4**
Demonstrations	-	Demo	The STM32Cube demonstration platform comes on top of the STM32Cube as a firmware package that offers a full set of software components based on a modular architecture. All modules can be reused separately in standalone applications. All these modules are managed by the STM32Cube demonstration kernel that allows to dynamically add new modules and access common resources (storage, memory management, real-time operating system). The STM32Cube demonstration platform is built around a basic GUI interface. It is based on the STM32Cube HAL BSP and several middleware components.	-	X	-	-	-
	Total number of demonstrations: 1			**0**	**1**	**0**	**0**	**0**
Total number of projects: 361				**2**	**66**	**138**	**55**	**100**

Revision history

Table 2. **Document revision history**

Date	Version	Changes
23-Feb-2023	1	Initial release
26-Jun-2023	2	Updated Table 1
06-Mar-2024	3	Updated: • Section 1: Reference documents • Table 1. STM32CubeH5 firmware examples

Contents

List of tables