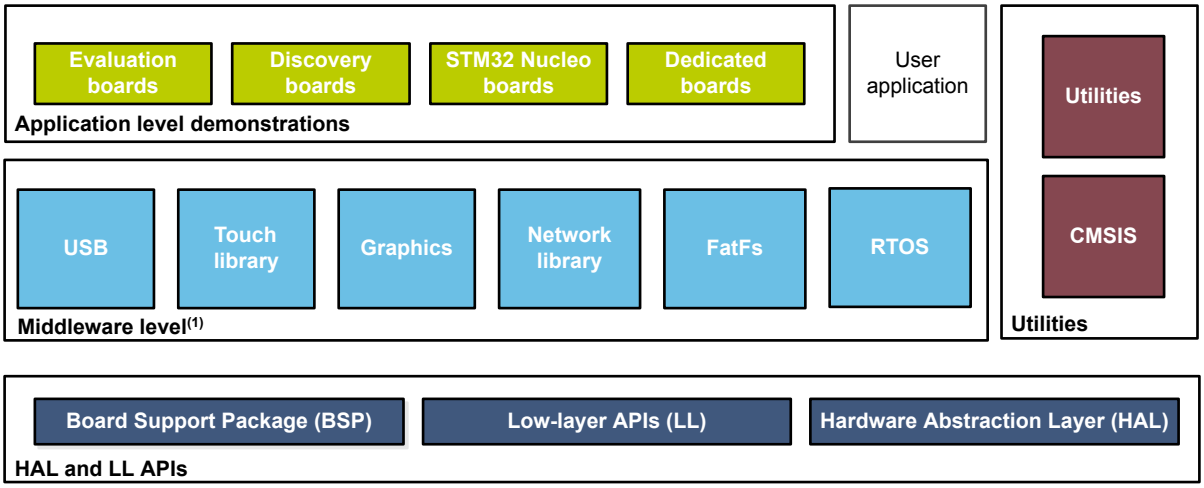
STM32Cube MCU Package examples for STM32H7 Series

Introduction

The STM32CubeH7 MCU Package is delivered with a rich set of examples running on STMicroelectronics boards. The examples are organized by board, and are provided with preconfigured projects for the main supported toolchains (see figure below).

Figure 1. STM32CubeH7 firmware components



(1) The set of middleware components depends on the product Series.

1 Reference documents

The reference documents are available on http://www.st.com/stm32cubefw:

- Latest release of STM32CubeH7 firmware package
- *Getting started with STM32CubeH7 for STM32H7 Series* (UM2204)
- *STM32CubeH7 demonstration platform* (UM2222)
- *Description of STM32H7 HAL drivers* (UM2217)
- *STM32Cube BSP driver development guidelines* (UM2298)
- *STM32Cube USB Device library* (UM1734)
- *STM32Cube USB host library* (UM1720)
- *Developing applications on STM32Cube with FatFs* (UM1721)
- *Developing applications on STM32Cube with RTOS* (UM1722)
- *Developing applications on STM32Cube with LwIP TCP/IP stack* (UM1713)
- *STM32Cube Ethernet IAP example* (UM1709)

The microcontrollers of the STM32H7 Series are based on Arm® Cortex® cores.

Note: Arm is a registered trademark of Arm Limited (or its subsidiaries) in the US and/or elsewhere.

2 STM32CubeH7 examples

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

- **Examples**: these examples use only the HAL and BSP drivers (middleware components not used). Their objective is to demonstrate the product/peripherals features and usage. They are organized per peripheral (one folder per peripheral, e.g. TIM). Their complexity level ranges from the basic usage of a given peripheral (e.g. PWM generation using timer) to the integration of several peripherals (e.g. how to use DAC for 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 and middleware components not used). They offer an optimum implementation of typical use cases of the peripheral features and configuration procedures. The examples are organized per peripheral (a folder for each peripheral, such as TIM).
- **Examples_MIX**: these examples use both HAL and LL drivers. They offer an optimum implementation of typical use cases of the peripheral features and configuration procedures. The examples are organized per peripheral (a folder for each peripheral, such as DMA2D).
- **Applications**: the applications demonstrate the product performance and how to use the available middleware stacks. They are organized either by middleware (a folder per middleware, e.g. USB Host) or by product feature that require high-level firmware bricks (e.g. Audio). 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 projects**: the template projects are provided to allow the user to quickly build a firmware application on a given board:
  - Templates for boards based on single-core STM32H7 microcontrollers (STM32H743I-EVAL, NUCLEOH743ZI, STM32H7B3I-EVAL, STM32H7B3I-DK, STM32H735G-DK, NUCLEO-H7A3ZI-Q and NUCLEO-H723ZG): STM32CubeH7 contains one HAL and one LL template projects.
  - Templates for boards based on dual-core STM32H7 microcontrollers (NUCLEO-H745ZI-Q, STM32H45I-DISCO, STM32H747I-DISCO and STM32H747I-EVAL):
    - One LL template project
    - Four HAL template projects:
      - *BootCM4_CM7:*
        - The Arm® Cortex®-M7 and Cortex®-M4 cores are both running from different Flash memory banks.
        - The system configuration is performed by the Arm® Cortex®-M7.
        - The Arm® Cortex®-M4 cores enters Stop mode after boot, and is then woken up by Arm® Cortex®-M7 using a hardware semaphore.
      - *BootCM7_CM4Gated:*
        - The Arm® Cortex®-M4 boot is gated using Flash memory option bytes.
        - The Arm® Cortex®-M7 and Cortex®-M4 cores are both running from different Flash memory banks.
        - The Arm® Cortex®-M7 core boots, performs the system configuration, and then enables Arm® Cortex®-M4 boot through the RCC.
      - *BootCM4_CM7Gated:*
        - The Arm® Cortex®-M7 boot is gated using Flash memory option bytes.
        - The Arm® Cortex®-M7 and Cortex®-M4 cores are both running from different Flash memory banks.
        - The Cortex®-M4 core boots , performs the system configuration, and then enables the Cortex®-M7 boot through the RCC.
      - *BootCM7_CM4Gated_RAM:*
        - The Arm® Cortex®-M4 boot is gated using Flash memory option bytes.
        - The Arm® Cortex®-M7 core and Arm® Cortex®-M4 core run from Flash memory bank 1 and from the D2 SRAM, respectively.
        - The Arm® Cortex®-M7 core performs the following actions at boot time:
          - system configuration
          - loading of the Arm® Cortex®-M4 code into the D2 SRAM
          - change of the Arm® Cortex®-M4 boot address and then enabling of Cortex®-M4 boot (through the RCC)
  - Template for the STM32H750B-DK board based on Value line STM32H7 microcontrollers:
    - *ExtMem_Boot:* reference boot code with execution from internal Flash memory. It configures external memories, and then jumps to the user application located in an external memory. Two use cases are possible, XiP and BootROM:
      - XiP: this use case is intended for eXecution in Place from external Flash memory (QUADSPI). In this case, the user application code shall be linked with the target execution memory address in external Quad-SPI Flash memory.
      - BootROM: this use case demonstrates how to boot from internal Flash memory, configure the external SDRAM, copy user application binary from the SDMMC Flash memory or from Quad-SPI Flash memory to the external SDRAM, and then jump to the user application. In this case, the user application code shall be linked with the target execution memory address in external SDRAM.
    - *Template_Project:* typical template with execution from external memory. Different configurations are available depending on the external memory boot capabilities:
      - XiP from QUADSPI, data in internal SRAM

- XiP from QUADSPI, data in external SDRAM
- BootROM: execution from external SDRAM, data in internal SRAM

Section ■ contains the list of examples provided with STM32CubeH7 MCU Package:

- ***Examples for boards based on single-core STM32H7 microcontrollers (STM32H743I-EVAL, NUCLEO-H743ZI, STM32H750B-DK, STM32H7B3I-EVAL, STM32H7B3I-DK, STM32H735G-DK, NUCLEO-H7A3ZI-Q and NUCLEO-H723ZG)***

  The STM32CubeH7 MCU Package contains one target project configuration per workspace (Arm® Cortex®-M7 core). All single-core examples have the same structure:

  – *Inc* folder that contains all header files.
  – *Src* folder for the sources code.
  – *EWARM*, *MDK-ARM* and *SW4STM32* (STM32H743I-EVAL, NUCLEO-H743ZI, STM32H750B-DK), *STM32CubeIDE* (STM32H7B3I-EVAL, STM32H7B3I-DK, STM32H735G-DK, NUCLEO-H7A3ZI-Q, NUCLEO-H723ZG) folders that contain the preconfigured project for each toolchain.
  – A *readme.txt* file describing the example behavior and the environment required to run the example.

- ***Examples for boards based on dual-core STM32H7 microcontrollers (NUCLEO-H745ZI-Q, STM32H745I-DISCO, STM32H747I-DISCO and STM32H747I-EVAL)*:**

  The STM32CubeH7 MCU Package contains two target project configurations per workspace (one per core), named STM32H7xyI_XXX_CM7 and STM32H7xyI_XXX_CM4. The projects can be configured individually by setting the following options: target microcontroller, linker options, read-only (RO) and read/write (RW) zones, and preprocessor symbols (CORE_CM4, CORE_CM7). This allows compiling user code linked and programmed separately for each core and generating two binaries: CM7 and CM4.

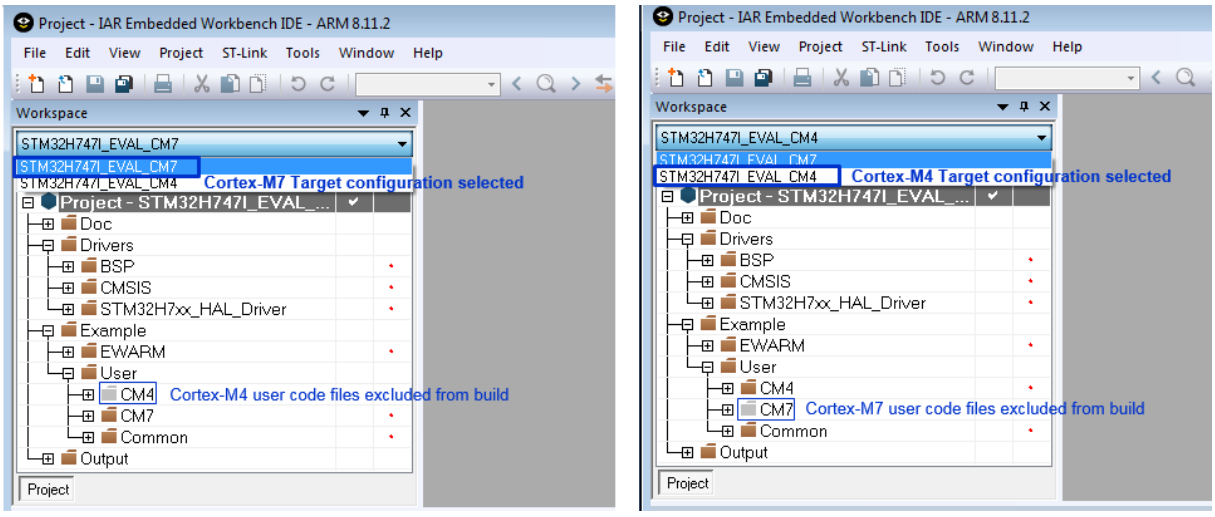
  The examples are structured as follows:

  – Common drivers files, used both for Arm® Cortex®-M7 and Arm® Cortex®-M4 cores, and including:
    ◦ CMSIS core files
    ◦ CMSIS device files
    ◦ HAL driver files
    ◦ BSP files
  – *EWARM*, *MDK-ARM* and *SW4STM32* folders containing the preconfigured projects
  – One *Src* and *Inc* folder per core.
  – A *Common* folder hosting system and shared source files both for Arm® Cortex®-M7 and Arm® Cortex®-M4 cores.
  – A *readme.txt* file describing the example behavior and the environment required to run the example.

Figure 2 and Figure 3 illustrate the organization of the examples and projects within the STM32CubeH7 MCU Package:

Figure 2. **STM32CubeH7 project tree**

Figure 3. **STM32CubeH7 project workspaces**



To run the example, proceed as follows:

1. *For single-core project examples (such as STM32H743I-EVAL board)*
   a. Browse to *\\Projects\\STM32H743I-EVAL\\Examples*.
   b. Open *\\GPIO*, then the *\\GPIO_EXTI* folder.
   c. Open the project using your preferred toolchain.
   d. Rebuild all files and load your image into the target memory.
   e. Run the example: each time you press the Tamper push-button, LED1 toggles (for more details, refer to the example *readme.txt* file).

2. *For dual-core project examples (such as STM32H747I-EVAL board):*
   a. Browse to *\\Projects\\STM32H747I-EVAL\\Examples*.
   b. Open *\\GPIO*, then the *\\GPIO_EXTI* folder.
   c. Open the project using your preferred toolchain
   d. For each target STM32H747I_EVAL_CM4 and STM32H747I_EVAL_CM7 (based on Arm® Cortex®-M7 and Cortex®-M4, respectively):
      i. Rebuild all files and load your image into the target memory.
      ii. After loading the two images, reset the board in order to boot CPU1 (Cortex®-M7) and CPU2 (Cortex®-M4) at once.
      iii. Each time you press the Tamper push-button:
         1. LED1 toggles once when an EXTI interrupt for Cortex®-M7 is detected.
         2. LED3 toggles once when an EXTI interrupt for Cortex®-M4 is detected.

      For more details, refer to the example *readme.txt* file.
3. For *Value line project example running on STM32H750B-DK board*:
   a. Browse to *Projects\\STM32H750B-DK\\Templates\ExtMem_Boot*.
   b. Open the ExtMem_Boot project with your preferred toolchain.
   c. Rebuild all files and load your image into the target internal Flash memory.
   d. Browse to *\\Projects\\STM32H750B-DK\\Examples*.
   e. Open *\\GPIO*, then the *\\GPIO_IOToggle* folder.
   f. Open the project using your preferred toolchain (keep the XIP_QSPI_InternalSRAM default configuration).
   g. Rebuild all files and load your image into the external Quad-SPI Flash memory.
   h. Run the example: LED1 toggles continuously (for more details, refer to the example *readme.txt* file).

Note: The principle of the STM32H750xx Value line application is to execute the user application from an external memory (Quad-SPI Flash memory by default or SDRAM). The Templates\ExtMem_Boot projects boot from the STM32H750xx internal Flash memory, configure external memories and then jump to the user application hosted in an external memory of the STM32H750B-DK board.

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 display, push-buttons, etc.). The BSP is based on a modular architecture that can be easily ported to any hardware by implementing the low-level routines.

Table 1. STM32CubeH7 firmware examples

Level	Module name	Project name	Description	STM32H7B3I-EVAL	STM32H7B3I-DK	STM32H750B-DK	STM32H747I-EVAL	STM32H747I-DISCO	STM32H745I-DISCO	STM32H743I-EVAL	STM32H735G-DK	NUCLEO-H7A3ZI-Q	NUCLEO-H745ZI-Q	NUCLEO-H743ZI	NUCLEO-H723ZG
Templates	-	BootCM4_CM7	This project provides a reference template that can be used to build any firmware application where both cores are used. It is mainly dedicated for devices where CPU1 (Arm®Cortex®-M7) and CPU2 (Arm®Cortex®-M4) are booting at once (with respect to the configured boot Flash memory options). System initialization, System clock, voltage scaling and L1-Cache configurations are done by CPU1 (Arm®Cortex®-M7).	-	-	-	X	X	X	-	-	-	X	-	-
		BootCM4_CM7Gated	This project provides a reference template that can be used to build any firmware application where both cores are used. It is mainly dedicated for devices where CPU2 (Arm®Cortex®-M4) is booting and CPU1 (Arm®Cortex®-M7) clock is gated.	-	-	-	X	X	X	-	-	-	X	-	-
		BootCM7_CM4Gated	This project provides a reference template that can be used to build any firmware application where both cores are used. It is mainly dedicated for devices where CPU1 (Arm®Cortex®-M7) is booting and CPU2 (Arm®Cortex®-M4) clock is gated.	-	-	-	X	X	X	-	-	-	X	-	-
		BootCM7_CM4Gated_RAM	This project provides a reference template that can be used to build any firmware application where both cores are used. It is mainly dedicated for devices where CPU1 (Arm®Cortex®-M7 in D1 Domain) is booting and CPU2 (Arm®Cortex®-M4 in D2 Domain) is gated (with respect to the configured boot Flash memory options).	-	-	-	X	X	X	-	-	-	X	-	-
		ExtMem_Boot	This directory contains a set of sources files and pre-configured projects that describe how to build an application for execution from external memory using the ExtMem_Boot firmware.	-	-	X	-	-	-	-	-	-	-	-	-
		Starter project	This project provides a reference template that can be used to build any firmware application.	X	X	-	-	-	-	X	X	X	-	X	X
		Template_Project	This project provides a reference template that can be used to build any firmware application with execution from external memory. This project is configured for STM32H750xx devices using STM32CubeH7 HAL and running on STM32H750B-DISCO board from STMicroelectronics.	-	-	X	-	-	-	-	-	-	-	-	-
	Total number of templates: 25			**1**	**1**	**2**	**4**	**4**	**4**	**1**	**1**	**1**	**4**	**1**	**1**
Templates_LL	-	Starter project	This project provides a reference template through the LL API that can be used to build any firmware application.	X	X	-	X	X	X	X	X	X	X	X	X
	Total number of templates_ll: 11			**1**	**1**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
Examples	-	BSP	This project uses STM32CubeH7 HAL and BSP. It provides a description of how to use the different BSP drivers for each STM32H7 board.	X	X	X	X	X	X	X	X	X	X	X	X

Level	Module name	Project name	Description	STM32H7B3I-EVAL	STM32H7B3I-DK	STM32H750B-DK	STM32H747I-EVAL	STM32H747I-DISCO	STM32H745I-DISCO	STM32H743I-EVAL	STM32H735G-DK	NUCLEO-H7A3ZI-Q	NUCLEO-H745ZI-Q	NUCLEO-H743ZI	NUCLEO-H723ZG
(Continued) **Examples**	ADC	ADC_AnalogWatchdog	This example provides a short description of how to use the ADC peripheral to perform conversions with analog watchdog and out-of-window interrupts enabled.	-	-	-	-	-	-	X	X	X	-	X	-
		ADC_DAC_Interconnect	This example describes how to configure and connect DAC output to ADC input and use the analog watchdog to monitor signal behavior.	X	-	-	-	-	-	X	-	-	-	X	-
		ADC_DMA_Transfer	This example describes how to configure and use the ADC to convert an external analog input and get the result using a DMA transfer through the HAL API.	X	-	-	-	-	-	X	X	X	-	X	-
		ADC_DifferentialMode	This example describes how to configure and use ADC2 to convert an external analog input in Differential mode (difference between external voltage on VINN and VINP).	-	-	-	-	-	-	X	-	X	-	X	-
		ADC_DualModeInterleaved	This example describes how to use two ADC peripherals to perform conversions in Dual interleaved mode.	-	X	-	X	X	X	X	-	X	X	X	-
		ADC_InternalChannelConversion	This example describes how to configure and use the ADC to retrieve the system battery level.	-	-	-	-	-	-	X	-	-	-	X	-
		ADC_OverSampler	This example describes how to configure and use the ADC to convert an external analog input combined with oversampling feature to increase resolution through the HAL API.	-	-	-	-	-	-	X	-	-	-	X	-
		ADC_Oversampling	This example describes how to use an ADC peripheral with ADC oversampling.	-	-	-	-	-	-	-	-	X	-	-	-
		ADC_RegularConversion_Polling	This example describes how to use the ADC in Polling mode to convert data through the HAL API.	-	-	-	-	-	-	X	-	-	-	X	-
		ADC_Regular_injected_groups	This example provides a short description of how to use the ADC peripheral to perform conversions using the two ADC groups: regular group for ADC conversions on main stream and injected group for ADC conversions limited to specific events (conversions injected within main conversion stream).	-	-	-	-	-	X	X	-	-	-	X	-
	CEC	CEC_DataExchange	This example shows how to configure and use the CEC peripheral to receive and transmit messages.	-	-	-	-	-	-	X	-	-	-	-	-
	COMP	COMP_AnalogWatchdog	This example shows how to use a pair of comparator peripherals to compare a voltage level applied to a GPIO pin to two thresholds: the internal voltage reference (V_{REFINT}) and a fraction of the internal voltage reference ($V_{REFINT}/4$), in Interrupt mode.	X	-	-	-	-	-	X	-	-	-	X	X
		COMP_Interrupt	This example shows how to configure the comparator peripheral to compare the external voltage applied to a specific pin to the internal voltage reference.	X	-	-	X	-	-	X	X	X	-	X	X
		COMP_OutputBlanking	This example shows how to use the comparator output blanking feature.	-	-	-	-	-	-	X	-	X	-	X	-
	CORDIC	CORDIC_Sin_DMA	This example shows how to use the CORDIC peripheral to calculate array of sines in DMA mode.	-	-	-	-	-	-	-	X	-	-	-	X

Level	Module name	Project name	Description	STM32H7B3I-EVAL	STM32H7B3I-DK	STM32H750B-DK	STM32H747I-EVAL	STM32H747I-DISCO	STM32H745I-DISCO	STM32H743I-EVAL	STM32H735G-DK	NUCLEO-H7A3ZI-Q	NUCLEO-H745ZI-Q	NUCLEO-H743ZI	NUCLEO-H723ZG
(Continued) **Examples**	CRC	CRC_Bytes_Stream_7bit_CRC	This example shows 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^7 + X^6 + X^5 + X^2 + 1, as used in the Train Communication Network, IEC 60870-5[17].	X	-	-	-	-	-	X	-	X	-	X	-
		CRC_Data_Reversing_16bit_CRC	This example shows how to configure the CRC using the HAL API. The CRC (cyclic redundancy check) calculation unit computes a 16-bit CRC code derived from a buffer of 32-bit data (words). Input and output data reversal features are enabled. The user-defined generating polynomial is manually set to 0x1021, that is X^16 + X^12 + X^5 + 1 which is the CRC-CCITT generating polynomial.	-	-	-	-	-	-	-	X	-	-	-	-
		CRC_Example	This example shows 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).	X	-	-	-	-	-	X	X	X	-	X	X
		CRC_UserDefinedPolynomial	This example shows 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).	-	-	-	X	-	X	X	X	X	-	X	X
	CRYP	CRYP_AESCCM_IT	This example shows how to use the CRYP peripheral to encrypt/ decrypt data (plaintext/ciphertext) in Interrupt mode using AES with Combined Cipher Machine (CCM), and then generate the authentication TAG .	X	-	-	-	-	-	X	-	-	-	-	-
		CRYP_AESGCM	This example shows how to use the CRYP peripheral to encrypt/ decrypt data (plaintext/ciphertext) using AES Galois/counter mode (GCM) and generate the authentication TAG .	-	-	-	X	-	-	X	-	-	-	-	-
		CRYP_AESModes	This example shows how to use the CRYP peripheral to encrypt/ decrypt data (plaintext/ciphertext) using AES ECB, CBC and CTR algorithms.	-	-	-	-	-	-	X	-	-	-	-	-
		CRYP_AESModes_DMA	This example shows how to use the CRYP peripheral to encrypt/ decrypt data (plaintext/ciphertext) using AES ECB algorithm in DMA mode with swapping.	-	-	-	-	-	-	X	-	-	-	-	-
		CRYP_AES_GCM	This example shows how to use the CRYP peripheral to encrypt and decrypt data using AES with Galois/Counter mode (GCM).	-	-	-	-	-	-	-	X	-	-	-	-
		CRYP_TDESModes	This example shows how to use the CRYP peripheral to encrypt/ decrypt data (plaintext/ciphertext) using TDES ECB and CBC algorithms.	-	-	-	-	-	-	X	-	-	-	-	-
	Cortex	CORTEXM_Cache	This example provides a description of how to do Data-cache maintenance on a shared memory buffer accessed by two masters (CPU and DMA).	X	-	-	-	-	-	X	-	-	-	-	-
	DAC	DAC_DualConversion	This example provides a short description of how to use the DAC peripheral in Dual conversion mode.	-	-	-	-	-	-	X	-	-	-	X	-

Level	Module name	Project name	Description	STM32H7B3I-EVAL	STM32H7B3I-DK	STM32H750B-DK	STM32H747I-EVAL	STM32H747I-DISCO	STM32H745I-DISCO	STM32H743I-EVAL	STM32H735G-DK	NUCLEO-H7A3ZI-Q	NUCLEO-H745ZI-Q	NUCLEO-H743ZI	NUCLEO-H723ZG
(Continued) **Examples**	(Continued) DAC	DAC_SignalsGeneration	This example shows how to use the DAC peripheral to generate several signals using the DMA controller.	X	X	-	X	-	X	X	X	X	-	X	X
		DAC_SimpleConversion	This example provides a short description of how to use the DAC peripheral to perform a simple conversion.	X	-	-	-	-	-	X	-	X	-	X	-
	DCMI	DCMI_CaptureMode	This example shows how to use the DCMI to interface with a camera module and continuously capture images into a Camera Frame Buffer located in external SDRAM.	-	-	-	-	X	-	-	-	-	-	-	-
		DCMI_SnapshotMode	This example shows how to use the DCMI to interface with a camera module, capture a single image in Camera Frame Buffer (320x240 with RGB565 format), and once the full frame camera is captured display it on the LCD in ARGB8888 format.	-	-	-	-	X	-	-	-	-	-	-	-
	DFSDM	DFSDM_AudioRecord	This example shows how to use the DFSDM HAL API to perform stereo audio recording.	X	-	-	X	-	-	X	X	-	-	-	-
	DMA	DMAMUX_RequestGen	This example shows how to use the DMA with the DMAMUX request generator to generate DMA transfer requests upon LPTIM2 output signal, knowing that LPTIM2 is configured in PWM with a 2s period.	X	X	X	X	X	X	X	X	X	-	X	X
		DMAMUX_SYNC	This example shows how to use the DMA with the DMAMUX to synchronize a transfer with LPTIM1 output signal.	X	-	-	-	-	-	X	-	-	-	-	-
		DMA_FIFOMode	This example provides a description of how to use a DMA to transfer a word data buffer from Flash memory to embedded SRAM with FIFO mode enabled and through the HAL API.	-	-	-	-	-	-	X	-	-	-	-	-
		DMA_FLASHToRAM	This example shows how to use a DMA to transfer a word data buffer from Flash memory to embedded SRAM through the HAL API.	X	-	-	-	-	-	X	-	-	-	-	-
	DMA2D	DMA2D_BlendingWithAlphaInversion	This example provides a description of how to configure the DMA2D peripheral in Memory-to-memory mode, Blending transfer and Alpha inversion mode.	X	X	-	X	X	-	X	X	-	-	-	-
		DMA2D_MemToMemWithBlending	This example provides a description of how to configure the DMA2D peripheral in Memory-to-memory mode and Blending transfer mode.	-	-	X	X	X	X	X	-	-	-	-	-
		DMA2D_MemToMemWithBlendingAndCLUT	This example shows how to configure the DMA2D peripheral in Memory-to-memory blending transfer mode and with indexed 256-color images (L8). It also shows how to use the DMA2D foreground/background CLUT in L8 color mode.	X	-	-	X	X	-	X	X	-	-	-	-
		DMA2D_MemToMemWithPFCandRed BlueSwap	This example shows how to configure the DMA2D peripheral in Memory-to-memory transfer mode with pixel format conversion and red and blue swap, and then display the result on the LCD.	X	-	-	X	X	-	X	X	-	-	-	-
		DMA2D_MemoryToMemory	This example provides a description of how to configure the DMA2D peripheral in Memory-to-memory transfer mode.	-	-	-	X	X	-	X	-	-	-	-	-
		DMA2D_RegToMemWithLCD	This example shows how to configure the DMA2D peripheral in Register-to-memory transfer mode and display the result on the LCD.	-	X	-	X	X	-	X	-	-	-	-	-

Level	Module name	Project name	Description	STM32H7B3I-EVAL	STM32H7B3I-DK	STM32H750B-DK	STM32H747I-EVAL	STM32H747I-DISCO	STM32H745I-DISCO	STM32H743I-EVAL	STM32H735G-DK	NUCLEO-H7A3ZI-Q	NUCLEO-H745ZI-Q	NUCLEO-H743ZI	NUCLEO-H723ZG
(Continued) **Examples**	DTS	DTS_GetTemperature	This example shows how to configure and use the DTS to get the temperature of the die.	X	X	-	-	-	-	-	X	X	-	-	X
	FDCAN	FDCAN_Classic_Frame_Networking	This example shows how to configure the FDCAN peripheral to send and receive Classic CAN frames in Normal mode.	-	-	-	-	-	X	X	-	-	-	-	-
		FDCAN_Clock_calibration	This example shows how to achieve clock calibration on an FDCAN unit.	-	-	-	-	-	-	X	-	-	-	-	-
		FDCAN_Com_IT	This example shows how to achieve Interrupt Process Communication between two FDCAN units.	-	-	-	-	-	-	X	-	-	-	-	-
		FDCAN_Com_polling	This example shows how to achieve Polling Process Communication between two FDCAN units.	-	-	-	-	-	-	X	-	-	-	-	-
		FDCAN_Image_transmission	This example shows the gain in time obtained by the activation of the Bit Rate Switching (BRS) feature.	-	-	-	-	-	X	X	-	-	-	-	-
		FDCAN_Loopback	This example shows how to configure the FDCAN to operate in Loopback mode.	X	-	-	-	-	-	X	X	-	-	-	-
	FLASH	FLASH_CoreConfiguration	This example guides the user through the configuration steps to copy a dedicated program by CPU1 (Arm®Cortex®-CM7) to Flash memory bank 2, to be executed by CPU2 (Arm®Cortex®-CM4).	-	-	-	X	-	X	-	-	-	-	-	-
		FLASH_EraseProgram	This example shows how to configure and use the FLASH HAL API to erase and program the internal Flash memory.	X	X	-	-	-	-	X	X	X	-	X	X
		FLASH_SwapBank	This example guides the user through the configuration steps to program internal Flash memory bank 1 and bank 2, and swap between both banks by means of the FLASH HAL API.	-	-	-	-	-	-	X	-	-	-	X	-
		FLASH_WriteProtection	This example shows how to configure and use the FLASH HAL API to enable and disable the write protection of the internal Flash memory.	X	X	-	-	-	-	X	-	X	-	-	-
	FMAC	FMAC_FIR_DMAToIT	This example shows how to use the FMAC peripheral to perform a FIR filter from DMA mode to Interrupt mode.	-	-	-	-	-	-	-	X	-	-	-	X
		FMAC_IIR_PollingToDMA	This example shows how to use the FMAC peripheral to perform an IIR filter from Polling mode to DMA mode.	-	-	-	-	-	-	-	X	-	-	-	X
	FMC	FMC_NOR	This example guides the user through the different configuration steps for configuring the FMC controller to access the PC28F128M29EWLA NOR Flash memory mounted on the STM32H743I-EVAL evaluation board, by means of the HAL API.	-	-	-	-	-	-	X	-	-	-	-	-
		FMC_SDRAM	This example describes how to configure the FMC controller to access the SDRAM.	X	X	X	-	-	-	X	-	-	-	-	-
		FMC_SDRAM_DataMemory	This example describes how to configure the FMC controller to access the SDRAM including heap and stack.	-	-	-	X	-	X	X	-	-	-	-	-
		FMC_SDRAM_LowPower	This example describes how to configure the FMC controller to access the SDRAM in low-power mode (SDRAM Self-refresh mode).	-	-	-	-	-	-	X	-	-	-	-	-

Level	Module name	Project name	Description	STM32H7B3I-EVAL	STM32H7B3I-DK	STM32H750B-DK	STM32H747I-EVAL	STM32H747I-DISCO	STM32H745I-DISCO	STM32H743I-EVAL	STM32H735G-DK	NUCLEO-H7A3ZI-Q	NUCLEO-H745ZI-Q	NUCLEO-H743ZI	NUCLEO-H723ZG
(Continued) **Examples**	(Continued) FMC	FMC_SRAM	This example describes how to configure the FMC controller to access the SRAM.	X	-	-	-	-	-	X	-	-	-	-	-
	GFXMMU	GFXMMU_DisplayCircularShape	This example describe how to enable and use the GFXMMU functionality to display an image with circular shape.	X	-	-	-	-	-	-	-	-	-	-	-
	GPIO	GPIO_EXTI	This example provides a description of how to configure external interrupt lines.	X	X	-	X	X	X	X	X	-	X	X	X
		GPIO_IOToggle	This example describe how to configure and use GPIOs through the HAL API.	X	X	X	-	-	-	-	-	X	-	-	-
	HAL	HAL_TimeBase_RTC_ALARM	This example describes how to customize the HAL timebase using an RTC alarm instead of the SysTick as main timebase source.	X	-	-	-	-	-	X	X	X	-	X	X
		HAL_TimeBase_RTC_WKUP	This example describes how to customize the HAL using an RTC wakeup as main timebase source, instead of the SysTick.	X	-	-	-	-	-	X	X	X	-	X	X
		HAL_TimeBase_TIM	This example describes how to customize the HAL using a general-purpose timer as main timebase source, instead of the SysTick.	X	-	-	-	-	-	X	X	X	-	X	X
	HASH	HASH_HMAC_SHA1MD5	This example describes how to use the HASH peripheral to hash data with HMAC SHA-1 and HMAC MD5 algorithms.	-	-	-	-	-	-	X	-	-	-	-	-
		HASH_SHA1MD5	This example shows how to use the HASH peripheral to hash data with SHA-1 and MD5 algorithms.	X	-	-	-	-	-	X	X	-	-	-	-
		HASH_SHA1MD5_DMA	This example describes how to use the HASH peripheral to hash data using SHA-1 and MD5 algorithms when data are fed to the HASH unit with DMA.	-	-	-	-	-	-	X	-	-	-	-	-
		HASH_SHA224SHA256_DMA	This example describes how to use the HASH peripheral to hash data with SHA224 and SHA256 algorithms.	-	-	-	X	-	-	X	-	-	-	-	-
	HRTIM	HRTIM_Arbitrary_Waveform	This example shows how to configure the HRTIM1 peripheral to generate an arbitrary signal.	-	-	-	-	-	-	X	-	-	-	X	-
		HRTIM_DAC_ADC_Interconnect	This example shows how to use the interconnection feature between HRTIM, DAC and ADC.	-	-	-	-	-	-	X	-	-	-	X	-
		HRTIM_ExternalEvents	This example shows how to use the external event to set and reset the HRTIM.	-	-	-	-	-	-	X	-	-	-	X	-
		HRTIM_FaultEvent	This example shows how to configure the HRTIM peripheral in PWM mode and configure and use the Fault event.	-	-	-	-	-	-	X	-	-	-	X	-
		HRTIM_MultiplePWM	This example shows how to configure HRTIM1 to generate up to five PWM signals with different duty cycle for each HRTIM output.	-	-	-	X	X	X	X	-	-	X	X	-
		HRTIM_PWM_DifferentFrequencies	This example shows how to configure HRTIM1 to generate up to six PWM signals with different timebase configuration for each slave timer.	-	-	-	-	-	-	X	-	-	-	X	-
	HSEM	HSEM_CoreNotification	This example describes how to use an embedded hardware semaphore to exchange notifications between cores.	-	-	-	X	X	X	-	-	-	-	-	-

Level	Module name	Project name	Description	STM32H7B3I-EVAL	STM32H7B3I-DK	STM32H750B-DK	STM32H747I-EVAL	STM32H747I-DISCO	STM32H745I-DISCO	STM32H743I-EVAL	STM32H735G-DK	NUCLEO-H7A3ZI-Q	NUCLEO-H745ZI-Q	NUCLEO-H743ZI	NUCLEO-H723ZG
(Continued) **Examples**	(Continued) HSEM	HSEM_CoreSync	This example describes how to use an embedded hardware semaphore to synchronize cores.	-	-	-	X	X	X	-	-	-	-	-	-
		HSEM_ProcessSync	This example describes how to use a hardware semaphore to synchronize processes. In this example, the hardware semaphore (9) is used to synchronize two processes.	X	-	-	-	-	-	X	-	-	-	X	-
		HSEM_ReadLock	This example describes how to enable, take, then release a semaphore using two different processes.	-	-	-	-	-	-	X	-	-	-	X	-
		HSEM_ResourceSharing	This example describes how to use an embedded hardware semaphore to share resources between cores.	-	-	-	X	X	X	-	-	-	-	-	-
	I2C	I2C_EEPROM_fast_mode_plus	This example describes how to handle I2C data buffer transmission/reception with DMA. In the example, the device communicates with an I2C EEPROM.	X	-	-	-	-	-	X	-	-	-	-	-
		I2C_TwoBoards_ComDMA	This example describes how to handle I2C data buffer transmission/reception between two boards via DMA.	-	-	-	-	-	-	-	X	X	-	X	X
		I2C_TwoBoards_ComIT	This example describes how to handle I2C data buffer transmission/reception between two boards, using an interrupt.	-	-	-	-	-	-	-	X	X	-	X	X
		I2C_TwoBoards_ComPolling	This example describes how to handle I2C data buffer transmission/reception between two boards in Polling mode.	-	-	-	-	-	-	-	X	X	-	X	X
		I2C_WakeUpFromStop	This example describes how to perform I2C data buffer transmission/reception between two instances using an interrupt, when one core is in Stop mode.	-	-	-	X	-	-	X	-	X	-	X	X
	IWDG	IWDG_WindowMode	This example describes how to periodically update the IWDG reload counter, and simulate a software fault that generates a MCU IWDG reset when a programmed time period has elapsed.	X	-	-	X	X	X	X	X	-	X	X	X
	JPEG	JPEG_DecodingFromFLASH_DMA	This example demonstrates how to decode a JPEG image stored in the internal Flash memory using the JPEG hardware decoder in DMA mode, and display the final ARGB8888 image on the LCD mounted on the board.	X	-	X	X	X	X	X	-	-	-	-	-
		JPEG_DecodingUsingFs_DMA	This example demonstrates how to read a JPEG file from an SD card memory using FatFs, decode it using the JPEG hardware decoder in DMA mode, and display the final ARGB8888 image on the KoD DSI-LCD mounted on the board or on an HDMI monitor connected through the DSI-HDMI bridge board (MB1232.A).	-	-	-	X	X	-	X	-	-	-	-	-
		JPEG_DecodingUsingFs_Interrupt	This example demonstrates how to read a JPEG file from the SD card memory using FatFs, decode it using the JPEG hardware decoder in Interrupt mode, and display the final ARGB8888 image on the KoD DSI-LCD mounted on the board or on an HDMI monitor connected through the DSI-HDMI bridge board (MB1232.A).	-	-	-	X	X	-	X	-	-	-	-	-

Level	Module name	Project name	Description	STM32H7B3I-EVAL	STM32H7B3I-DK	STM32H750B-DK	STM32H747I-EVAL	STM32H747I-DISCO	STM32H745I-DISCO	STM32H743I-EVAL	STM32H735G-DK	NUCLEO-H7A3ZI-Q	NUCLEO-H745ZI-Q	NUCLEO-H743ZI	NUCLEO-H723ZG
(Continued) **Examples**	(Continued) JPEG	JPEG_DecodingUsingFs_Polling	This example demonstrates how to read a JPEG file from the SD card memory using FatFs, decode it using the JPEG hardware decoder in Polling mode, and display the final ARGB8888 image on the KoD DSI-LCD mounted on the board or on an HDMI monitor connected through the DSI-HDMI bridge board (MB1232.A).	-	-	-	X	X	-	X	-	-	-	-	-
		JPEG_EncodingFromFLASH_DMA	This example demonstrates how to read an RGB image stored in the internal Flash memory, encode it using the JPEG hardware encoder in DMA mode, and save it into an SD card.	X	X	-	X	X	-	X	-	-	-	-	-
		JPEG_EncodingUsingFs_DMA	This example demonstrates how to read a BMP file from an SD card memory using FatFs, encode it using the JPEG hardware encoder in DMA mode, and save it into the SD card.	-	-	-	X	X	-	X	-	-	-	-	-
		JPEG_MJPEG_VideoDecoding	This example demonstrates how to use the hardware JPEG decoder to decode an MJPEG video file located on the microSD and display the final ARGB8888 video on the KoD DSI-LCD mounted on the board or on an HDMI monitor connected through the DSI-HDMI bridge board (MB1232.A).	-	-	-	X	X	-	X	-	-	-	-	-
		JPEG_MJPEG_VideoDecodingFromOSPI	This example demonstrates how to use the hardware JPEG decoder to decode an MJPEG video file located in the external Octo-SPI Flash memory and display it on the LCD-TFT screen.	X	X	-	-	-	-	-	-	-	-	-	-
		JPEG_MJPEG_VideoDecodingFromQSPI	This example demonstrates how to use the hardware JPEG decoder to decode an MJPEG video file located in the external Quad-SPI Flash memory and display the final ARGB8888 video on the KoD DSI-LCD mounted on board or on an HDMI monitor connected through the DSI-HDMI bridge board (MB1232.A).	-	-	-	X	X	-	X	-	-	-	-	-
	LCD_DSI	LCD_DSI_CmdMode_DoubleBuffer	This example provides a description of how to use the embedded LCD DSI controller (using LTDC and DSI Host peripherals) to drive the KoD LCD mounted on the board in DSI Command mode with dual buffer display.	-	-	-	X	X	-	-	-	-	-	-	-
		LCD_DSI_CmdMode_PartialRefresh	This example provides a description of how to use the embedded LCD DSI controller (using LTDC and DSI Host peripherals) to drive the KoD LCD mounted on the board in DSI Command mode with partial refresh display.	-	-	-	X	X	-	-	-	-	-	-	-
		LCD_DSI_CmdMode_SingleBuffer	This example provides a description of how to use the embedded LCD DSI controller (using LTDC and DSI Host peripherals) to drive the KoD LCD mounted on the board in DSI Command mode with single buffer display.	-	-	-	X	X	-	-	-	-	-	-	-
		LCD_DSI_CmdMode_TearingEffect	This example provides a description of how to use the embedded LCD DSI controller (using LTDC and DSI Host peripherals) to drive the KoD LCD mounted on the board in DSI Command mode with tearing effect management based on DSI link.	-	-	-	X	X	-	-	-	-	-	-	-
		LCD_DSI_CmdMode_TearingEffect_ExtPin	This example provides a description of how to use the embedded LCD DSI controller (using LTDC and DSI Host peripherals) to drive the KoD LCD mounted on the board in DSI Command mode with tearing effect management based on DSI TE pin.	-	-	-	X	X	-	-	-	-	-	-	-

Level	Module name	Project name	Description	STM32H7B3I-EVAL	STM32H7B3I-DK	STM32H750B-DK	STM32H747I-EVAL	STM32H747I-DISCO	STM32H745I-DISCO	STM32H743I-EVAL	STM32H735G-DK	NUCLEO-H7A3ZI-Q	NUCLEO-H745ZI-Q	NUCLEO-H743ZI	NUCLEO-H723ZG
(Continued) **Examples**	(Continued) LCD_DSI	LCD_DSI_ULPM_Data	This example provides a description of how to use the embedded LCD DSI controller (using LTDC and DSI Host peripherals) to drive the KoD LCD mounted on the board and manage entry and exit in DSI ULPM mode on data lane only. In this mode, the DSI PHY state machine enters in low-power state on data lane thus allowing to save power when the LCD does not need to display any image.	-	-	-	X	X	-	-	-	-	-	-	-
		LCD_DSI_ULPM_DataClock	This example provides a description of how to use the embedded LCD DSI controller (using LTDC and DSI Host peripherals) to drive the KoD LCD mounted on the board and manage entry and exit in DSI ULPM mode on data lane and clock lane.	-	-	-	X	X	-	-	-	-	-	-	-
		LCD_DSI_VideoMode_DoubleBuffering	This example provides a description of how to use the embedded LCD DSI controller (using LTDC and DSI Host peripherals) to drive, in DSI Video mode, the KoD LCD mounted on the board or an HDMI monitor connected through the DSI-HDMI bridge board (MB1232.A). The display is based on two buffers.	-	-	-	X	X	-	-	-	-	-	-	-
		LCD_DSI_VideoMode_SingleBuffer	This example provides a description of how to use the embedded LCD DSI controller (using LTDC and DSI Host peripherals) to drive, in DSI Video mode, the KoD LCD mounted on the board or an HDMI monitor connected through the DSI-HDMI bridge board (MB1232.A). The display is based on a single buffer.	-	-	-	X	X	-	-	-	-	-	-	-
	LPTIM	LPTIM_Encoder	This example shows how to configure the LPTIM peripheral in Encoder mode.	-	-	-	-	-	-	X	-	-	-	X	-
		LPTIM_PWMExternalClock	This example describes how to configure and use LPTIM to generate a PWM at the lowest power consumption, using an external counter clock and the HAL LPTIM API.	-	-	-	-	-	-	X	-	X	-	X	-
		LPTIM_PWM_LSE	This example describes how to configure and use LPTIM to generate a PWM in low-power mode using the LSE as a counter clock and the HAL LPTIM API.	X	-	-	-	-	-	X	-	X	-	X	-
		LPTIM_PulseCounter	This example describes how to configure and use, through the LPTIM HAL API, the LPTIM peripheral to count pulses.	-	X	-	X	X	X	X	-	X	X	X	X
		LPTIM_Timeout	This example describes how to implement a low-power timeout to wake up the system using the LPTIM, through the HAL LPTIM API.	-	-	-	-	-	-	X	-	X	-	X	-
	LTDC	LTDC_ColorKeying	This example describes how to enable and use the LTDC color keying functionality.	X	-	-	-	-	-	X	-	-	-	-	-
		LTDC_ColorKeying_FromOSPI	This example describes how to enable and use the LTDC color keying functionality and use the Octo-SPI memory.	X	X	-	-	-	-	-	X	-	-	-	-
		LTDC_Display_1Layer	This example provides a description of how to configure the LTDC peripheral to display an RGB image of size 480x272 and format RGB565 (16 bits/pixel) on the LCD using only one layer.	-	-	X	-	-	X	X	-	-	-	-	-
		LTDC_Display_2Layers	This example describes how to configure the LTDC peripheral to display two layers at the same time.	-	X	X	-	-	X	X	X	-	-	-	-

Level	Module name	Project name	Description	STM32H7B3I-EVAL	STM32H7B3I-DK	STM32H750B-DK	STM32H747I-EVAL	STM32H747I-DISCO	STM32H745I-DISCO	STM32H743I-EVAL	STM32H735G-DK	NUCLEO-H7A3ZI-Q	NUCLEO-H745ZI-Q	NUCLEO-H743ZI	NUCLEO-H723ZG
(Continued) **Examples**	MDMA	MDMA_DMA2D_Triggering	This example describes how to use the MDMA with hardware trigger set to the DMA2D transfer complete flag.	-	-	-	X	X	-	X	-	-	-	-	-
		MDMA_GPDMA_Triggering	This example describes how to use the MDMA with hardware trigger set to D2 Domain GP-DMA transfer complete flag.	-	-	-	-	-	-	X	-	-	-	-	-
		MDMA_LTDC_Triggering	This example describes how to use the MDMA with hardware trigger set to the LTDC line interrupt Ffag.	X	-	-	-	-	-	X	X	-	-	-	-
		MDMA_LinkedList	This example describes how to use the MDMA to perform a list of transfers. The transfer list is organized as a linked-list. Each time the current transfer completes, the MDMA automatically reloads the next transfer parameters and starts the transfer without CPU intervention.	X	-	X	-	-	-	X	X	X	-	X	X
		MDMA_LinkedList_ColorsComp	This example demonstrates how to use the MDMA in Linked-list mode to extract RGB colors from an ARGB8888 image, resize each subimage (with a decimation factor /2) and display the resulting RGB decimated subimages on the LCD.	-	-	-	X	X	-	X	-	-	-	-	-
		MDMA_RepeatBlock_Rotation	This example provides a description of how to use the MDMA in Repeat block trigger mode to copy an RGB565 image to the LCD frame buffer.	X	X	-	X	X	X	X	X	-	-	-	-
		MDMA_RepeatBlock_ZoomOut	This example provides a description of how to use the MDMA in Repeat block trigger mode to decimate an RGB565 image and copy it to the LCD frame buffer.	-	-	-	X	X	X	X	-	-	-	-	-
	MMC	MMC_ReadWrite_DMA	This example performs write and read transfers to the MMC card in SDMMC internal DMA mode, and calculates write and read transfer speed.	-	-	-	-	-	X	-	-	-	-	-	-
		MMC_ReadWrite_IT	This example performs write and read transfers to the MMC card in Interrupt mode and calculates write and read transfer speed.	-	-	-	-	-	X	-	-	-	-	-	-
	OPAMP	OPAMP_Calibration	This example shows how to calibrate the OPAMP peripheral.	-	-	-	-	-	-	-	X	X	-	X	-
		OPAMP_Follower	This example shows how to configure the OPAMP peripheral in Follower mode interconnected with DAC and COMP.	X	-	-	-	-	-	X	-	X	-	X	X
		OPAMP_PGA_ExternalBias	This example shows how to configure the OPAMP peripheral in PGA mode with bias voltage for the Non-inverting mode.	-	-	-	X	-	-	X	-	-	-	X	-
	OSPI	OSPI_HyperRAM_MemoryMapped	This example describes how to write and read data in Memory-mapped mode to/from the Octo-SPI HyperRAM memory and compare the result with an intensive access.	-	-	-	-	-	-	-	X	-	-	-	-
		OSPI_NOR_MemoryMapped_DTR	This example describes how to erase part of the Octo-SPI NOR memory, write data in Interrupt mode, and access the Octo-SPI NOR memory in Memory-mapped mode to check the data in a forever loop. The memory is configured in Octal DTR mode.	X	X	-	-	-	-	-	X	-	-	-	-
		OSPI_NOR_ReadWrite_DMA	This example describes how to erase part of the Octo-SPI NOR memory, write data in DMA mode, read data in DMA mode, and compare the result in a forever loop.	X	X	-	-	-	-	-	-	-	-	-	-

Level	Module name	Project name	Description	STM32H7B3I-EVAL	STM32H7B3I-DK	STM32H750B-DK	STM32H747I-EVAL	STM32H747I-DISCO	STM32H745I-DISCO	STM32H743I-EVAL	STM32H735G-DK	NUCLEO-H7A3ZI-Q	NUCLEO-H745ZI-Q	NUCLEO-H743ZI	NUCLEO-H723ZG
(Continued) **Examples**	(Continued) OSPI	OSPI_RAM_MemoryMapped	This example describes how to write and read data in Memory-mapped mode to/from the Octo-SPI HyperRAM memory, and compare the result in a forever loop.	X	-	-	-	-	-	-	-	-	-	-	-
	OTFDEC	OTFDEC_DataDecrypt	This example describes how to decrypt data located in the Octo-SPI external Flash memory using the OTFDEC peripheral.	X	X	-	-	-	-	-	X	-	-	-	-
		OTFDEC_EncryptionDecryption	This example shows how to use OTFDEC to encrypt, decrypt and execute PI calculation algorithm stored in external NOR Flash memory. USART1 is used to verify that decryption and executing instructions are done correctly.	X	X	-	-	-	-	-	-	-	-	-	-
		OTFDEC_ExecutingAesInstruction	This example shows how to use the OTFDEC to decrypt and execute PI calculation algorithm stored in external NOR Flash memory using the embedded cryptographic IP standard AES-128 counter mode to encrypt the binary image. USART1 is used to verify that decryption and executing instructions are done correctly.	X	X	-	-	-	-	-	-	-	-	-	-
		OTFDEC_ExecutingCryptedInstruction	This example shows how to use the OTFDEC to decrypt and execute crypted instructions stored in external NOR Flash memory.	X	-	-	-	-	-	-	X	-	-	-	-
	PSSI	PSSI_Transmit_Receive_DMA	This example describes how to perform PSSI data buffer transmission/reception between the on-board PSSI configured as a slave and a master simulated by another board. This project is configured for STM32H7A3xxQ devices using STM32CubeH7 HAL and running on an STMicroelectronics NUCLEO-H723ZG board.	-	-	-	-	-	-	-	-	X	-	-	X
	PWR	PWR_D1ON_D2OFF	This example shows how to run the system with only D1 domain in Run mode while D2 domain is in Standby mode.	-	-	-	X	X	X	-	-	-	X	-	-
		PWR_D2ON_D1OFF	This example shows how to run the system with only D2 domain in Run mode while D1 domain is in Standby mode.	-	-	-	X	X	X	-	-	-	X	-	-
		PWR_Domain3SystemControl	This example shows how to maintain a basic system activity in low-power mode with D3 Domain only, by ensuring the communication between the D3SRAM, the BDMA and the LPUART when the system is in Stop mode.	-	-	-	X	X	X	X	-	-	X	X	-
		PWR_Hold_Mechanism	This example shows how to use the hold mechanism to allow the system to be re-initialized by a master CPU. This enables the master CPU to be woken up both by its own wakeup sources and by the wakeup sources of the slave CPU. The slave CPU remains on hold until it is released by the master CPU.	-	-	-	X	X	X	-	-	-	X	-	-
		PWR_STANDBY	This example shows how to enter Standby mode and wake up from this mode using an external reset or a WKUP pin.	-	X	-	X	-	X	X	-	X	-	X	-
		PWR_STANDBY_RTC	This example shows how to enter Standby mode and wake up from this mode using an external reset or the RTC wakeup timer.	X	X	-	X	X	X	X	X	X	X	X	X
		PWR_STOP2_RTC	This example shows how to enter Stop mode with main domain in DStop2 and wake up from this mode using the RTC wakeup timer with memory shut-off option enabled.	X	X	-	-	-	-	-	-	X	-	-	-

Level	Module name	Project name	Description	STM32H7B3I-EVAL	STM32H7B3I-DK	STM32H750B-DK	STM32H747I-EVAL	STM32H747I-DISCO	STM32H745I-DISCO	STM32H743I-EVAL	STM32H735G-DK	NUCLEO-H7A3ZI-Q	NUCLEO-H745ZI-Q	NUCLEO-H743ZI	NUCLEO-H723ZG
(Continued) **Examples**	(Continued) PWR	PWR_STOP_DataRetain	This example shows how to retain data in D3SRAM when the system enters Stop mode with D1 domain in Standby mode, to guarantee low-power consumption.	-	-	-	-	-	-	X	-	-	-	X	-
		PWR_STOP_RTC	This example shows how to enter Stop mode and wake up from this mode using the RTC wakeup timer as wakeup source event. This event is connected to EXTI_Line19.	-	-	-	X	X	X	X	X	-	X	X	-
		PWR_STOP_STANDBY	This example shows how to enter Stop/Standby mode and wake up from this mode using an external reset or a WKUP pin connected to the user button.	-	-	X	-	-	-	-	-	-	-	-	-
		PWR_VOS0_480MHZ	This example shows how to over-clock the system to 480 MHz with VOS0. In this example, when the USE_VOS0_480MHZ_OVERCLOCK define (located in main.h) is set to zero, the SystemClock_Config_400MHz() function is used to set the Flash latency and configure the system clock as well as the Arm®Cortex®-M7 and Arm®Cortex®-M4 frequencies at 400 MHz and 200 MHz, respectively.	-	-	-	-	-	-	-	-	-	X	X	-
	QSPI	QSPI_ExecuteInPlace	This example describes how to execute part of the code from the Quad-SPI memory. To do this, a section is created where the function is stored.	-	-	-	-	-	-	X	-	-	-	-	-
		QSPI_MemoryMapped	This example describes how to erase part of the Quad-SPI memory, write data in DMA mode, and access to Quad-SPI memory in Memory-mapped mode to check the data in a forever loop.	-	-	-	-	-	-	X	-	-	-	-	-
		QSPI_MemoryMappedDual	This example describes how to erase part of the Quad-SPI memory, write data in Interrupt mode, and access to Quad-SPI memory in Memory-mapped dual mode to check the data in a forever loop.	-	-	X	X	-	X	X	-	-	-	-	-
		QSPI_ReadWriteDual_DMA	This example describes how to use Quad-SPI interface in Dual mode. It erases part of the Quad-SPI memory, writes data in DMA mode, reads data in DMA mode, and compares the result in a forever loop.	-	-	-	-	-	-	X	-	-	-	-	-
		QSPI_ReadWrite_DMA	This example describes how to erase part of the Quad-SPI memory, read and write data in DMA mode.	-	-	-	-	-	-	X	-	-	-	-	-
		QSPI_ReadWrite_IT	This example describes how to erase part of the Quad-SPI memory, write data in Interrupt mode, read data in Interrupt mode, and compare the result in a forever loop.	-	-	-	-	-	-	X	-	-	-	-	-
	RAMECC	RAMECC_ErrorCount	This example describes how to enable and activate notifications for RAMECC RAMs.	-	-	-	-	-	-	-	-	-	X	-	-
	RCC	RCC_ClockConfig	This example demonstrates how to configure the system clock (SYSCLK) and modify the clock settings in Run mode, using the RCC HAL API.	X	X	-	X	X	X	X	X	X	X	X	X
	RNG	RNG_MultiRNG	This example demonstrates how to configure the RNG through the HAL API. This example uses the RNG to generate 32-bit long random numbers.	X	X	-	X	-	X	X	X	X	-	X	X

Level	Module name	Project name	Description	STM32H7B3I-EVAL	STM32H7B3I-DK	STM32H750B-DK	STM32H747I-EVAL	STM32H747I-DISCO	STM32H745I-DISCO	STM32H743I-EVAL	STM32H735G-DK	NUCLEO-H7A3ZI-Q	NUCLEO-H745ZI-Q	NUCLEO-H743ZI	NUCLEO-H723ZG
(Continued) **Examples**	RTC	RTC_ActiveTamper	This example describes how to program active tampers.	X	-	-	-	-	-	-	-	-	-	-	-
		RTC_Alarm	This example describes how to configure and generate an RTC alarm using the RTC HAL API.	X	X	X	X	X	X	X	X	X	X	X	X
		RTC_Tamper	This example describes how to configure the RTC to write/read data to/from RTC Backup registers. It also demonstrates the tamper detection feature.	-	X	-	-	-	-	X	X	X	-	X	X
		RTC_TimeStamp	This example demonstrates how to configure the RTC HAL API to use the timestamp feature.	-	X	-	-	-	-	X	X	X	-	X	X
	SAI	SAI_AudioPlay	This example shows how to use the SAI HAL API to play an audio file in DMA circular mode and handle the buffer update.	X	-	-	-	-	-	X	X	-	-	-	-
		SAI_AudioPlayback	This example shows how to use the SAI to play back audio data coming from two microphones.	-	-	-	-	-	X	X	-	-	-	-	-
	SD	SD_ReadWrite_DMA	This example shows how to support DMA mode for a microSD card.	X	-	-	-	-	-	X	-	-	-	-	-
		SD_ReadWrite_DMADoubleBuffer	This example performs write and read transfers to/from an SD card in SDMMC internal DMA mode, and calculates write and read transfer speeds.	X	-	-	-	-	-	X	X	-	-	-	-
		SD_ReadWrite_DMA_HS	This example shows how to support DMA mode for a microSD card.	X	-	-	-	-	-	X	-	-	-	-	-
		SD_ReadWrite_IT	This example performs write and read transfers to/from an SD card in SDMMC internal DMA mode, and calculates write and read transfer speeds.	X	-	-	-	-	-	X	X	-	-	-	-
	SPDIFRX	SPDIFRX_AudioPlay	This example shows how to use the SPDIFRX HAL API to receive audio data and then play them through the Codec by using the SAI interface.	-	-	-	-	X	-	-	-	-	-	-	-
	SPI	SPI_FullDuplex_ComDMA	This example shows how to perform data buffer transmission/ reception between two boards via SPI using DMA.	-	-	-	-	-	-	-	-	X	X	X	X
		SPI_FullDuplex_ComIT	This example shows how to perform data buffer transmission/ reception between two boards via SPI in Interrupt mode.	-	-	-	-	-	-	-	-	X	X	X	X
		SPI_FullDuplex_ComPolling	This example shows how to perform data buffer transmission/ reception between two boards via SPI in Polling mode.	-	-	-	-	-	-	-	-	X	-	X	X
	TIM	TIM_6Steps	This example shows how to configure the TIM1 peripheral to generate six steps.	X	-	-	-	-	-	X	-	-	-	X	-
		TIM_Asymetric	This example shows how to configure the TIM peripheral to generate an asymmetric signal.	-	-	-	-	-	-	X	-	X	-	X	-
		TIM_Combined	This example shows how to configure the TIM1 peripheral to generate three PWM combined signals with TIM1 channel 5.	-	-	-	-	-	-	X	-	-	-	X	-
		TIM_ComplementarySignals	This example shows how to configure the TIM1 peripheral to generate three complementary TIM1 signals, insert a defined dead time value, use the break feature, and lock the desired parameters.	-	-	-	-	-	-	X	-	-	-	X	-

Level	Module name	Project name	Description	STM32H7B3I-EVAL	STM32H7B3I-DK	STM32H750B-DK	STM32H747I-EVAL	STM32H747I-DISCO	STM32H745I-DISCO	STM32H743I-EVAL	STM32H735G-DK	NUCLEO-H7A3ZI-Q	NUCLEO-H745ZI-Q	NUCLEO-H743ZI	NUCLEO-H723ZG
(Continued) **Examples**	(Continued) TIM	TIM_DMA	This example shows how to use the DMA with TIMER update request to transfer data from memory to TIMER capture compare register 3 (TIMx_CCR3).	-	X	-	-	X	X	X	-	X	-	X	X
		TIM_DMABurst	This example shows how to update the TIMER channel 1 period and the duty cycle using the TIMER DMA burst feature.	-	-	-	-	-	-	X	X	-	-	X	-
		TIM_InputCapture	This example shows how to use the TIM peripheral to measure the frequency of an external signal.	-	-	-	-	-	-	X	-	-	-	X	-
		TIM_OCToggle	This example shows how to configure the TIMER peripheral to generate four different signals with four different frequencies.	-	-	-	-	-	-	X	-	-	-	X	-
		TIM_OnePulse	This example shows how to use the TIMER peripheral to generate a single pulse when a rising edge of an external signal is received on the TIMER input pin.	-	-	-	-	-	-	X	-	-	-	X	-
		TIM_PWMOutput	This example shows how to configure the TIMER peripheral in PWM (pulse width modulation) mode.	X	-	-	X	-	-	X	-	-	X	X	-
		TIM_Synchronization	This example shows how to synchronize TIM1 with TIM3 and TIM4 timers in Parallel mode.	-	-	-	-	-	-	X	-	X	-	X	-
		TIM_TimeBase	This example shows how to configure the TIMER peripheral to generate a 1-second timebase with the corresponding interrupt request.	-	-	X	-	-	-	X	-	X	-	X	-
	UART	LPUART_WakeUpFromStop	This example shows how to configure an LPUART to wake up the MCU from Stop mode when a given stimulus is received.	X	-	-	-	-	-	X	-	-	-	-	-
		UART_HyperTerminal_DMA	This example describes an UART transmission (transmit/receive) in DMA mode between a board and an HyperTerminal PC application.	X	-	-	-	-	-	X	-	-	-	-	-
		UART_HyperTerminal_IT	This example describes an UART transmission (transmit/receive) between a board and an HyperTerminal PC application by using an interrupt.	X	-	-	-	-	-	X	-	-	-	-	-
		UART_Printf	This example shows how to re-route the C library printf function to the UART.	X	X	-	-	-	-	X	X	-	-	-	-
		UART_TwoBoards_ComDMA	This example describes UART transmission (transmit/receive) in DMA mode between two boards.	-	-	-	-	-	-	-	-	X	-	X	-
		UART_TwoBoards_ComIT	This example describes UART transmission (transmit/receive) in Interrupt mode between two boards.	-	-	-	-	-	-	-	-	X	-	X	-
		UART_TwoBoards_ComPolling	This example describes UART transmission (transmit/receive) in Polling mode between two boards.	-	-	-	-	-	-	-	-	X	-	X	X
		UART_WakeUpFromStopUsingFIFO	This example shows how to use UART HAL API to wake up the MCU from Stop mode using the UART FIFO level.	X	-	-	X	X	X	X	-	-	X	-	-
	USART	USART_SlaveMode	This example describes an USART-SPI communication (transmit/receive) between two boards with the USART configured as a slave.	-	-	-	-	-	-	-	-	X	-	X	X

Level	Module name	Project name	Description	STM32H7B3I-EVAL	STM32H7B3I-DK	STM32H750B-DK	STM32H747I-EVAL	STM32H747I-DISCO	STM32H745I-DISCO	STM32H743I-EVAL	STM32H735G-DK	NUCLEO-H7A3ZI-Q	NUCLEO-H745ZI-Q	NUCLEO-H743ZI	NUCLEO-H723ZG
(Continued) **Examples**	WWDG	WWDG_Example	This example describes how to periodically update the WWDG counter and simulate a software fault that generates an MCU WWDG reset when a predefined time period has elapsed.	-	-	-	X	X	X	X	-	-	X	X	-
		WWDG_ResetAfterSwFailure	This example describes how to periodically update the WWDG counter and simulate a software fault that generates an MCU WWDG reset when a predefined time period has elapsed.	X	X	-	-	-	-	-	X	X	-	-	X
	Total number of examples: 647			**70**	**34**	**13**	**61**	**51**	**40**	**135**	**52**	**53**	**21**	**79**	**38**
Examples_LL	ADC	ADC_AnalogWatchdog	This example shows how to use an ADC peripheral with an ADC analog watchdog to monitor a channel and detect when the corresponding converted data are outside the window thresholds.	-	-	-	-	-	-	-	-	-	-	-	X
	COMP	COMP_CompareGpioVsVrefInt_IT	This example shows how to use a comparator peripheral to compare, in Interrupt mode, a voltage level applied on a GPIO pin to the internal voltage reference (V_{REFINT}). This example is based on the STM32H7xx COMP LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	-	-	-	-	-	-	-	-	X	-	X	X
		COMP_CompareGpioVsVrefInt_IT_Init	This example shows how to use a comparator peripheral to compare, in Interrupt mode, a voltage level applied on a GPIO pin to the internal voltage reference (V_{REFINT}). This example is based on the STM32H7xx COMP LL API. The peripheral initialization uses the LL initialization function to demonstrate LL init usage.	-	-	-	-	-	-	-	-	-	-	-	X
	CORDIC	CORDIC_CosSin	This example shows how to use the CORDIC peripheral to calculate cosine and sine.	-	-	-	-	-	-	-	-	-	-	-	X
	CORTEX	CORTEX_MPU	This example introduces the MPU feature. It configures a memory area as privileged read-only and attempts to perform read and write operations in different modes.	-	-	-	-	-	-	-	-	-	-	-	X
	CRC	CRC_CalculateAndCheck	This example shows 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).	-	-	-	-	-	-	-	-	X	-	-	X
		CRC_UserDefinedPolynomial	This example shows 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).	-	-	-	-	-	-	-	-	-	-	-	X
	CRS	CRS_Synchronization_IT	This example shows how to configure the clock recovery service in Interrupt mode through the STM32H7xx CRS LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	-	-	-	-	-	-	-	-	-	-	-	X
	DAC	DAC_GenerateWaveform_TriggerHW	This example shows how to use the DAC peripheral to generate a voltage waveform from a digital data stream transferred by DMA. It is based on the STM32H7xx DAC LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	-	-	-	-	-	-	-	-	X	-	X	X

Level	Module name	Project name	Description	STM32H7B3I-EVAL	STM32H7B3I-DK	STM32H750B-DK	STM32H747I-EVAL	STM32H747I-DISCO	STM32H745I-DISCO	STM32H743I-EVAL	STM32H735G-DK	NUCLEO-H7A3ZI-Q	NUCLEO-H745ZI-Q	NUCLEO-H743ZI	NUCLEO-H723ZG
(Continued) **Examples_LL**	(Continued) DAC	DAC_GenerateWaveform_TriggerHW_Init	This example shows how to use the DAC peripheral to generate a voltage waveform from a digital data stream transferred by DMA. It is based on the STM32H7xx DAC LL API. The peripheral initialization uses LL initialization functions to demonstrate LL init usage.	-	-	-	-	-	-	-	-	-	-	-	X
	DMA	DMA_CopyFromFlashToMemory	This example shows how to use a DMA to transfer a word data buffer from Flash memory to embedded SRAM. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	-	-	-	-	-	-	-	-	X	-	-	X
		DMA_CopyFromFlashToMemory_Init	This example shows how to use a DMA to transfer a word data buffer from Flash memory to embedded SRAM. The peripheral initialization uses LL initialization functions to demonstrate LL init usage.	-	-	-	-	-	-	-	-	-	-	-	X
	DMA2D	DMA2D_MemoryToMemory	This example shows how to configure the DMA2D peripheral in Memory-to-memory transfer mode. The example is based on the STM32H7xx DMA2D LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	-	X	-	-	-	-	-	X	-	-	-	-
	EXTI	EXTI_ToggleLedOnIT	This example shows how to configure the EXTI and use GPIOs to toggle the user LEDs available on the board when a user button is pressed. It is based on the STM32H7xx LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	-	-	-	-	-	-	-	-	X	-	-	X
		EXTI_ToggleLedOnIT_Init	This example shows 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 STM32H7xx LL API. The peripheral initialization uses LL initialization functions to demonstrate LL init usage.	-	-	-	-	-	-	-	-	-	-	-	X
	GPIO	GPIO_InfiniteLedToggling	This example shows how to configure and use GPIOs to toggle the on-board user LEDs every 250 ms. This example is based on the STM32H7xx LL API. The peripheral is initialized with LL unitary service functions to optimize for performance and size.	-	-	-	-	-	-	-	-	X	-	-	X
		GPIO_InfiniteLedToggling_Init	This example shows how to configure and use GPIOs to toggle the on-board user LEDs every 250 ms. This example is based on the STM32H7xx LL API. The peripheral is initialized with LL initialization function to demonstrate LL init usage.	-	-	-	-	-	-	-	-	-	-	-	X
	I2C	I2C_OneBoard_Communication_IT	This example shows how to handle the reception of one data byte from an I2C slave device by an I2C master device. Both devices operate in Interrupt mode. The peripheral is initialized with LL unitary service functions to optimize for performance and size.	-	-	-	-	-	-	-	-	-	-	-	X
		I2C_OneBoard_Communication_IT_Init	This example shows how to handle the reception of one data byte from an I2C slave device by an I2C master device. Both devices operate in Interrupt mode. The peripheral is initialized with LL initialization function to demonstrate LL init usage.	-	-	-	-	-	-	-	-	-	-	-	X

Level	Module name	Project name	Description	STM32H7B3I-EVAL	STM32H7B3I-DK	STM32H750B-DK	STM32H747I-EVAL	STM32H747I-DISCO	STM32H745I-DISCO	STM32H743I-EVAL	STM32H735G-DK	NUCLEO-H7A3ZI-Q	NUCLEO-H745ZI-Q	NUCLEO-H743ZI	NUCLEO-H723ZG
(Continued) **Examples_LL**	IWDG	IWDG_RefreshUntilUserEvent	This example shows 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.	-	-	-	-	-	-	-	-	X	-	-	X
	PWR	PWR_EnterStandbyMode	This example shows how to enter Standby mode and wake up from this mode by using an external reset or a wakeup interrupt.	-	-	-	-	-	-	-	-	-	-	-	X
		PWR_EnterStopMode	This example shows how to enter Stop mode.	-	-	-	-	-	-	-	-	-	-	-	X
	RCC	RCC_OutputSystemClockOnMCO	This example shows how to configure MCO pins (PA8 and PC9) to output the system clock.	-	-	-	-	-	-	-	-	X	-	-	X
	RNG	RNG_GenerateRandomNumbers	This example shows how to configure the RNG to generate 32-bit long random numbers. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	-	-	-	-	-	-	-	-	X	-	-	X
		RNG_GenerateRandomNumbers_IT	This example shows how to configure the RNG to generate 32-bit long random numbers using interrupts. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	-	-	-	-	-	-	-	-	-	-	-	X
	RTC	RTC_Alarm	This example shows how to configure the RTC LL API to configure and generate an alarm using the RTC peripheral. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	-	-	-	-	-	-	-	-	X	-	X	X
		RTC_Alarm_Init	This example shows how to configure the RTC LL API to configure and generate an alarm using the RTC peripheral. The peripheral initialization uses the LL initialization function.	-	-	-	-	-	-	-	-	-	-	-	X
	SPI	SPI_FullDuplex_ComIT	This example shows how to perform SPI data buffer transmission/reception between two instances in the same board by using interrupts.	-	-	-	-	-	-	-	-	-	-	X	X
		SPI_OneBoard_FullDuplex_IT	This example shows how to perform SPI data buffer transmission/reception between two instances in the same board by using interrupts. This example is based on the STM32H7xx SPI LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	-	-	-	-	-	-	-	-	X	-	-	-
		SPI_OneBoard_HalfDuplex_DMA_Init	This example shows how to configure GPIO and SPI peripherals to transmit bytes from an SPI master device to an SPI slave device in DMA mode. This example is based on the STM32H7xx SPI LL API. The peripheral initialization uses the LL initialization function to demonstrate LL init usage.	-	-	-	-	-	-	-	-	-	-	-	X
	TIM	TIM_PWMOutput	This example shows how to use a timer peripheral to generate a PWM output signal and update the PWM duty cycle. This example is based on the STM32H7xx TIM LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	-	-	-	-	-	-	-	-	X	-	-	X

Level	Module name	Project name	Description	STM32H7B3I-EVAL	STM32H7B3I-DK	STM32H750B-DK	STM32H747I-EVAL	STM32H747I-DISCO	STM32H745I-DISCO	STM32H743I-EVAL	STM32H735G-DK	NUCLEO-H7A3ZI-Q	NUCLEO-H745ZI-Q	NUCLEO-H743ZI	NUCLEO-H723ZG
(Continued) **Examples_LL**	(Continued) TIM	TIM_PWMOutput_Init	This example shows how to use a timer peripheral to generate a PWM output signal and update the PWM duty cycle. This example is based on the STM32H7xx TIM LL API. The peripheral initialization uses LL initialization function to demonstrate LL init.	-	-	-	-	-	-	-	-	-	-	-	X
	USART	USART_Communication_Rx_IT	This example shows how to configure GPIO and USART peripherals to receive characters from an HyperTerminal (PC) in Asynchronous mode using an interrupt. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	-	-	-	-	-	-	-	-	X	-	-	X
		USART_Communication_Rx_IT_Init	This example shows how to configure GPIO and USART peripherals to receive characters from an HyperTerminal (PC) in Asynchronous mode using an interrupt. The peripheral initialization uses the LL initialization function to demonstrate LL init.	-	-	-	-	-	-	-	-	-	-	-	X
	WWDG	WWDG_RefreshUntilUserEvent	This example shows how to configure 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).	-	-	-	-	-	-	-	-	-	-	-	X
	Total number of examples_ll: 52			**0**	**1**	**0**	**0**	**0**	**0**	**0**	**1**	**13**	**0**	**4**	**33**
Examples_MIX	CRC	CRC_PolynomialUpdate	This example shows how to use the CRC peripheral through the STM32H7xx CRC HAL and LL API.	-	-	-	-	-	-	-	-	X	-	-	X
	DMA	DMA_FLASHToRAM	This example shows how to use a DMA to transfer a word data buffer from Flash memory to embedded SRAM through the STM32H7xx DMA HAL and LL API. The LL API is used for performance improvement.	-	-	-	-	-	-	-	-	X	-	-	X
	DMA2D	DMA2D_MemToMemWithLCD	This example shows how to configure the DMA2D peripheral in Memory-to-memory transfer mode and display the result on the LCD. The DMA2D LL APIs are used for performance improvement.	-	X	-	-	-	-	-	X	X	-	-	X
	Total number of examples_mix: 8			**0**	**1**	**0**	**0**	**0**	**0**	**0**	**1**	**3**	**0**	**0**	**3**
Applications	Audio	Audio_playback_and_record	This application shows how to use the different functionalities of the SAI (Serial Audio Interface) to ensure audio record and playback via ST MEMS microphones (MP34DT01).	-	-	-	-	-	-	X	-	-	-	-	-
	Display	LTDC_AnimatedPictureFromSDCard	This example shows how to display an animated picture stored on the LCD on the microSD card.	-	X	-	-	-	-	X	-	-	-	-	-
		LTDC_Paint	This application describes how to configure the LCD touchscreen, attribute an action related to a configured touch zone, and save a BMP picture on the USB disk.	-	-	X	-	-	-	X	-	-	-	-	-
		LTDC_PicturesFromSDCard	This example shows how to use LTDC layers to display pictures stored on the SD card on the LCD.	X	X	-	-	-	-	X	-	-	-	-	-
	EEPROM	EEPROM_Emulation	This application describes the software solution for substituting standalone EEPROM by emulating the EEPROM mechanism using the STM32H743x on-chip Flash memory.	-	-	-	-	-	-	X	-	X	-	X	-

Level	Module name	Project name	Description	STM32H7B3I-EVAL	STM32H7B3I-DK	STM32H750B-DK	STM32H747I-EVAL	STM32H747I-DISCO	STM32H745I-DISCO	STM32H743I-EVAL	STM32H735G-DK	NUCLEO-H7A3ZI-Q	NUCLEO-H745ZI-Q	NUCLEO-H743ZI	NUCLEO-H723ZG
(Continued) **Applications**	ExtMem_Code Execution	ExtMem_Boot	This directory contains a set of sources files and pre-configured projects that describe how to build an application for execution from external memory using the ExtMem_Boot firmware.	X	X	-	-	-	-	X	X	-	-	-	-
		ExtMem_Application\FreeRTOS	This application shows how to implement thread creation using CMSIS RTOS API with execution from external memory.	X	X	-	-	-	-	X	X	-	-	-	-
		ExtMem_Application\LedToggling	This application provides a sample LED toggling program with execution from external memory.	X	X	-	-	-	-	X	X	-	-	-	-
	FPU	FPU_Fractal	This application explains how to use the STM32H7 floating-point units (FPU), and demonstrates the benefits it brings. The CortexM7 FPU is an implementation of the ARM FPv5-SP double-precision FPU.	-	-	-	X	X	-	X	-	-	-	-	-
	FatFs	FatFs_uSD_Standalone	This application describes how to use STM32Cube firmware with FatFs middleware component as a generic FAT file system module. The objective is to develop an application making the most of the features offered by FatFs to configure a microSD drive.	-	-	-	-	-	-	-	-	-	-	X	-
	FatFs	FatFs_CopyFiles	This application describes how to use STM32Cube firmware with FatFs middleware component as a generic FAT file system module. This example develops an application that exploits FatFs features to configure two microSD drives and copy files from the first instance to the second.	X	-	-	-	-	-	-	-	-	-	-	-
		FatFs_Dual_Instance	This application describes how to use STM32Cube firmware with FatFs middleware component as a generic FAT file system module. The objective is to develop an application making the most of the features offered by FatFs to configure a microSD and an USB drive.	-	-	-	X	-	-	-	-	-	-	-	-
		FatFs_MultiAccess_RTOS	This application describes how to use STM32Cube firmware with FatFs middleware component as a generic FAT file system module, FreeRTOS as an RTOS module based on using CMSIS-OS V2 wrapping layer common APIs.	-	X	-	-	-	-	-	-	-	-	-	-
		FatFs_MultiDrives	This application describes how to use STM32Cube firmware with FatFs middleware component as a generic FAT file system module. This example develops an application that exploits FatFs features, with multidrive (SDRAM, microSD) configurations.	-	X	-	-	-	-	X	-	-	-	-	-
		FatFs_RAMDisk	This application describes how to use STM32Cube firmware with FatFs middleware component as a generic FAT file system module, to develop an application exploiting FatFs offered features with RAM disk (SRAM) drive configuration.	-	-	-	-	-	-	X	-	X	-	-	-
		FatFs_SDRAMDisk	This application describes how to use STM32Cube firmware with FatFs middleware component as a generic FAT file system module. The objective is to develop an application making the most of the features offered by FatFs to configure a RAMDisk drive.	-	-	-	-	-	-	X	-	-	-	-	-

Level	Module name	Project name	Description	STM32H7B3I-EVAL	STM32H7B3I-DK	STM32H750B-DK	STM32H747I-EVAL	STM32H747I-DISCO	STM32H745I-DISCO	STM32H743I-EVAL	STM32H735G-DK	NUCLEO-H7A3ZI-Q	NUCLEO-H745ZI-Q	NUCLEO-H743ZI	NUCLEO-H723ZG
(Continued) **Applications**	(Continued) FatFs	FatFs_Shared_Device	This application describes how to use STM32Cube firmware with FatFs middleware component as a generic FAT file system module. The objective is to develop an application making the most of the features offered by FatFs to access the eMMC card.	-	-	-	-	-	X	-	-	-	-	-	-
		FatFs_USBDisk_RTOS	This application describes how to use STM32Cube firmware with FatFs middleware component as a generic FAT file system module and STM32 USB On-The-Go (OTG) host library, in Full-speed (FS) and High-speed (HS) modes, to develop an application exploiting FatFs offered features with USB disk drive configuration.	-	-	-	-	-	-	X	-	-	-	-	-
		FatFs_USBDisk_Standalone	This application describes how to use STM32Cube firmware with FatFs middleware component as a generic FAT file system module and STM32 USB On-The-Go (OTG) host library, in Full-speed (FS) and High-speed (HS) modes, to develop an application exploiting FatFs offered features with USB disk drive configuration.	-	-	-	-	-	-	X	-	-	-	-	-
		FatFs_uSD_DMA_RTOS	This application describes how to use STM32Cube firmware with FatFs middleware component as a generic FAT file system module, to develop an application exploiting FatFs offered features with microSD drive in RTOS mode configuration.	-	-	-	-	-	-	X	-	-	-	-	-
		FatFs_uSD_DMA_Standalone	This application describes how to use STM32Cube firmware with FatFs middleware component as a generic FAT file system module. This example develops an application that exploits FatFs features to configure a microSD drive.	-	X	-	-	-	-	X	-	-	-	-	-
		FatFs_uSD_RTOS	This application describes how to use STM32Cube firmware with FatFs middleware component as a generic FAT file system module, to develop an application exploiting FatFs offered features with microSD disk drive configuration.	-	-	-	-	-	-	-	X	-	-	-	-
		FatFs_uSD_Standalone	This application describes how to use STM32Cube firmware with FatFs middleware component as a generic FAT file system module. The objective is to develop an application making the most of the features offered by FatFs to configure a microSD drive.	-	-	-	-	-	-	X	X	-	-	-	-
	FreeRTOS	FreeRTOS_AMP_Dual_RTOS	This application shows how to use FreeRTOS message buffers to exchange data between the two cores. Each core has his own FreeRTOS instance.	-	-	-	-	-	X	-	-	-	X	-	-
		FreeRTOS_AMP_RTOS_BareMetal	This application shows how to use FreeRTOS message buffers to exchange data between the two cores.	-	-	-	-	-	X	-	-	-	X	-	-
		FreeRTOS_HwSemaphoreCoreSync	This application shows how to use an embedded hardware semaphore to send notifications between the two cores.	-	-	-	X	X	-	-	-	-	-	-	-
		FreeRTOS_LowPower	This application shows how to enter and exit low-power mode with CMSIS RTOS API.	-	X	-	-	-	-	-	-	X	-	-	-
		FreeRTOS_MPU	This application aims at describing how to use the MPU feature of FreeRTOS.	-	-	-	-	-	-	X	-	-	-	X	-

Level	Module name	Project name	Description	STM32H7B3I-EVAL	STM32H7B3I-DK	STM32H750B-DK	STM32H747I-EVAL	STM32H747I-DISCO	STM32H745I-DISCO	STM32H743I-EVAL	STM32H735G-DK	NUCLEO-H7A3ZI-Q	NUCLEO-H745ZI-Q	NUCLEO-H743ZI	NUCLEO-H723ZG
(Continued) **Applications**	(Continued) FreeRTOS	FreeRTOS_Mail	This application shows how to use mail queues with CMSIS RTOS API.	-	-	X	-	-	-	-	-	-	-	-	-
		FreeRTOS_Mutexes	This application shows how to use mutexes with CMSIS RTOS API.	X	-	-	-	-	-	-	-	X	-	-	-
		FreeRTOS_Queues	This application shows how to use message queues with CMSIS RTOS API.	-	-	-	-	-	-	-	-	X	-	-	-
		FreeRTOS_Semaphore	This application shows how to use semaphores with CMSIS RTOS API.	-	X	-	-	-	-	-	X	X	-	-	X
		FreeRTOS_SemaphoreFromISR	This application shows how to use semaphores from ISR with CMSIS RTOS API .	-	-	-	-	-	-	X	-	X	-	X	-
		FreeRTOS_ThreadCreation	This application shows how to implement thread creation using CMSIS RTOS API.	-	X	X	-	-	-	X	X	X	-	X	X
		FreeRTOS_Timers	This application shows how to use timers of CMSIS RTOS API.	X	-	-	-	-	-	-	-	X	-	-	-
	IAP	IAP_Binary_Template	This directory contains a set of sources files to build the application to be loaded into Flash memory using in-application programming (IAP) through USART.	-	-	-	-	-	-	X	-	-	-	-	-
		IAP_Main	This directory contains a set of sources files and pre-configured projects that describe how to build an in-application programming (IAP) that uses an USART interface to load an application binary.	-	-	-	-	-	-	X	-	-	-	-	-
	LibJPEG	LibJPEG_Decoding	This application demonstrates how to use the libjpeg API to decode a JPEG file.	-	-	-	-	-	-	X	X	-	-	-	-
		LibJPEG_Encoding	This application demonstrates how to use the libjpeg API to encode and decode a BMP image into a JPEG file.	-	-	-	-	-	-	X	X	-	-	-	-
	LwIP	LwIP_HTTP_Server_Netconn_RTOS	This application guides STM32Cube HAL API users to run an http server application based on Netconn API of LwIP TCP/IP stack The communication is done with a web browser application in a remote PC.	-	-	-	-	-	-	X	X	-	-	X	X
		LwIP_HTTP_Server_Raw	This application guides STM32Cube HAL API users to run a http server application based on Raw API of LwIP TCP/IP stack. The communication is done with a web browser application in a remote PC.	-	-	-	-	-	-	X	-	-	-	-	-
		LwIP_HTTP_Server_Socket_RTOS	This application guides STM32Cube HAL API users to run a http server application based on Socket API of LwIP TCP/IP stack. The communication is done with a web browser application in a remote PC.	-	-	-	-	-	-	X	X	-	-	-	X
		LwIP_TCP_Echo_Client	This application guides STM32Cube HAL API users to run TCP Echo Client application based on Raw API of LwIP TCP/IP stack. To run this application, open a command prompt window on the remote PC.	-	-	-	-	-	-	X	-	-	-	-	-
		LwIP_TCP_Echo_Server	This application guides STM32Cube HAL API users to run TCP Echo Server application based on Raw API of LwIP TCP/IP stack. To run this application, Open a command prompt window on the remote PC.	-	-	-	-	-	-	X	X	-	-	-	X

Level	Module name	Project name	Description	STM32H7B3I-EVAL	STM32H7B3I-DK	STM32H750B-DK	STM32H747I-EVAL	STM32H747I-DISCO	STM32H745I-DISCO	STM32H743I-EVAL	STM32H735G-DK	NUCLEO-H7A3ZI-Q	NUCLEO-H745ZI-Q	NUCLEO-H743ZI	NUCLEO-H723ZG
(Continued) **Applications**	(Continued) LwIP	LwIP_TFTP_Server	This application guides STM32Cube HAL API users to run a tftp server application for STM32H7xx devices.	-	-	-	-	-	-	X	-	-	-	-	-
		LwIP_UDPTCP_Echo_Server_Netconn_RTOS	This application guides STM32Cube HAL API users to run a UDP/TCP Echo Server application based on Netconn API of LwIP TCP/IP stack. To run this application, open a command prompt window on the remote PC.	-	-	-	-	-	-	X	-	-	-	-	-
		LwIP_UDP_Echo_Client	This application guides STM32Cube HAL API users to run a UDP Echo Client application based on Raw API of LwIP TCP/IP stack. To run this application, Open a command prompt window on the remote PC.	-	-	-	-	-	-	X	-	-	-	-	-
		LwIP_UDP_Echo_Server	This application guides STM32Cube HAL API users to run UDP Echo Server application based on Raw API of LwIP TCP/IP stack. To run this application, Open a command prompt window on the remote PC.	-	-	-	-	-	-	X	X	-	-	-	X
	OpenAMP	OpenAMP_PingPong	This application shows how to use OpenAMP middleware to create a communication channel (called rpmsg channel) between cores and send bidirectional messages.	-	-	-	X	X	X	-	-	-	-	-	-
		OpenAMP_RTOS_PingPong	This application shows how to use OpenAMP MW to create a communication channel (called rpmsg channel) between cores and send bidirectional messages.	-	-	-	X	X	X	-	-	-	-	-	-
	ResourcesManager	ResourcesManager_SharedResources	This application shows how to use the ResourcesManager to share resources between cores.	-	-	-	X	X	X	-	-	-	-	-	-
		ResourcesManager_UsageWithNotification	This application shows how to use the ResourcesManager to share resources between cores.	-	-	-	X	X	X	-	-	-	-	-	-
	STemWin	STemWin_HelloWorld	This directory contains a set of source files that implement a simple "Hello World" application based on STemWin for STM32H7B3xxQ devices.	X	X	X	X	X	X	X	-	-	-	-	-
		STemWin_SampleDemo	This directory contains a set of source files that implement a sample demonstration application allowing to demonstrate some of the STemWin Library capabilities on STM32H7B3xxQ devices.	X	X	X	-	-	X	X	-	-	-	-	-
		STemWin_acceleration	This directory contains a set of source files that implement a simple "acceleration" application based on STemWin for STM32H7xx devices.	-	-	-	X	X	-	-	-	-	-	-	-
		STemWin_animation	This directory contains a set of source files that implement a simple "animation" application based on STemWin for STM32H7xx devices.	-	-	-	X	X	-	-	-	-	-	-	-
		STemWin_fonts	This directory contains a set of source files that implement a simple "fonts" application based on STemWin for STM32H7xx devices.	-	-	-	X	X	-	-	-	-	-	-	-
		STemWin_memory_device	This directory contains a set of source files that implement a simple "memory device" application based on STemWin for STM32H7xx devices.	-	-	-	X	X	-	-	-	-	-	-	-

Level	Module name	Project name	Description	STM32H7B3I-EVAL	STM32H7B3I-DK	STM32H750B-DK	STM32H747I-EVAL	STM32H747I-DISCO	STM32H745I-DISCO	STM32H743I-EVAL	STM32H735G-DK	NUCLEO-H7A3ZI-Q	NUCLEO-H745ZI-Q	NUCLEO-H743ZI	NUCLEO-H723ZG
(Continued) **Applications**	USB_ Device	Audio_Standalone	This application is a part of the USB device library package using STM32Cube firmware. It describes how to use the USB device application based on the AUDIO Class implementation of an audio streaming (the output is a speaker/headset) capability on STM32H7xx devices.	-	-	-	X	-	-	X	-	-	-	-	-
		CDC_Standalone	This application is a part of the USB device library package using STM32Cube firmware. It describes how to use the USB device application based on the Device Communication Class (CDC) following the PSTN subprotocol on STM32H747xx devices. It uses the OTG-USB and UART peripherals.	-	-	-	X	-	-	X	-	-	-	-	-
		CustomHID_Standalone	This application is a part of the USB device library package using STM32Cube firmware. It describes how to use the USB device application based on the Custom HID Class on STM32H747xx devices.	-	-	-	X	X	-	X	-	-	-	-	-
		DFU_Standalone	This application is a compliant implementation of the Device Firmware Upgrade (DFU) capability to program the embedded Flash memory through the USB peripheral.	X	X	-	X	X	X	X	X	X	-	-	X
		DualCore_Standalone	This application is a part of the USB device library package using STM32Cube firmware. It describes how to use the USB device application based on the STM32H7xx multicore support feature integrating the Device Communication Class (CDC) and Human Interface (HID) in the same project.	-	-	-	X	-	-	X	-	-	-	-	-
		HID-CM4_MSC-CM7	This application is a part of the USB device library package using STM32Cube firmware. It describes how to use the USB device application based on the STM32H7xx multicore support feature integrating the Mass Storage Class (MSC) and Human Interface (HID) in the same project.	-	-	-	X	-	-	-	-	-	-	-	-
		HID_LPM_Standalone	The STM32H7xx devices support the USB Link Power Management Protocol (LPM-L1) and complies with the USB 2.0 LPM-L1 ECN. The hpcd.Init.lpm_enable in the usbd_conf.c must be set to 1 to enable the support for LPM-L1 protocol in the USB stack.	-	-	-	-	-	-	X	-	-	-	-	-
		HID_Standalone	This application shows how to use of the USB device application based on the Human Interface (HID).	X	X	X	X	X	X	X	X	X	-	-	X
		MSC_Standalone	This application is a part of the USB device library package using STM32Cube firmware. It describes how to use the USB device application based on the Mass Storage Class (MSC) on STM32H7xx devices.	-	-	-	X	X	-	X	-	-	-	-	-
	USB_Host	AUDIO_Standalone	This application is a part of the USB host library package using STM32Cube firmware. It describes how to use the USB host application based on the Audio OUT class on STM32H7xx devices.	-	-	-	-	-	-	X	-	-	-	-	-
		CDC_Standalone	This application is a part of the USB host library package using STM32Cube firmware. It describes how to use the USB host application based on the Communication Class (CDC) on STM32H7xx devices.	-	-	-	X	-	-	X	-	-	-	-	-

Level	Module name	Project name	Description	STM32H7B3I-EVAL	STM32H7B3I-DK	STM32H750B-DK	STM32H747I-EVAL	STM32H747I-DISCO	STM32H745I-DISCO	STM32H743I-EVAL	STM32H735G-DK	NUCLEO-H7A3ZI-Q	NUCLEO-H745ZI-Q	NUCLEO-H743ZI	NUCLEO-H723ZG
(Continued) **Applications**	(Continued) USB_Host	DualCore_Standalone	This application is a part of the USB host library package using STM32Cube firmware. It describes how to use the USB host application based on the STM32H7xx multicore support feature integrating Mass Storage (MSC) and Human Interface (HID) in the same project.	-	-	-	X	-	-	X	-	-	-	-	-
		DynamicSwitch_Standalone	This application is a part of the USB host library package using STM32Cube firmware. It describes how to use dynamically switch, on the same port, between available USB host applications on STM32H7xx devices.	-	-	-	-	-	-	X	-	-	-	-	-
		FWupgrade_Standalone	This application is a part of the USB host library package using STM32Cube firmware. It describes how to use USB host application based on the In-Application programming (IAP) on STM32H7xx devices.	-	-	-	X	-	-	X	-	-	-	-	-
		HID_RTOS	This application is a part of the USB host library package using STM32Cube firmware. It describes how to use USB host application based on the Human Interface Class (HID) on STM32H7xx devices.	-	-	-	X	-	-	X	-	-	-	-	-
		HID_Standalone	This application is a part of the USB host library package using STM32Cube firmware. It describes how to use the USB host application based on the Human Interface Class (HID) on STM32H7B3XXQ devices.	X	X	-	X	X	X	X	X	X	-	-	X
		MSC_RTOS	This application is a part of the USB host library package using STM32Cube firmware. It describes how to use the USB host application based on the Mass Storage Class (MSC) on STM32H7x devices.	-	-	-	X	-	-	X	-	-	-	-	-
		MSC_Standalone	This application is a part of the USB host library package using STM32Cube firmware. It describes how to use the USB host application based on the Mass Storage Class (MSC) on STM32H7XXQ devices.	X	X	X	X	X	X	X	X	X	-	-	X
	Wi-Fi	ClockAndWeather	This application provides a description on how to use the MXCHIP EMW3080B Wi-Fi module in a clock and weather porogram based on TouchGFX graphical library.	-	-	-	-	-	-	-	X	-	-	-	-
	mbedTLS	Crypto_Selftest	This application implements a set of cryptographic features on the STM32H753I-EVAL board. It is based on the mbedTLS self-test functions of individual mbedTLS components selectively chosen in a single configuration file "mbedtls_config.h".	-	-	-	-	-	-	X	X	-	-	-	-
		SSL_Client	This application describes how to run an SSL client application based on mbedTLS cryptographic library and LwIP TCP/IP stack.	-	-	-	-	-	-	X	-	-	-	-	-
		SSL_Server	This application describes how to run an SSL server application based on mbedTLS cryptographic library and LwIP TCP/IP stack.	-	-	-	-	-	-	X	-	-	-	-	-
	Total number of applications: 199			**13**	**17**	**7**	**27**	**17**	**13**	**55**	**19**	**13**	**2**	**6**	**10**
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.	X	X	X	X	X	X	X	X	X	X	X	X

Level	Module name	Project name	Description	STM32H7B3I-EVAL	STM32H7B3I-DK	STM32H750B-DK	STM32H747I-EVAL	STM32H747I-DISCO	STM32H745I-DISCO	STM32H743I-EVAL	STM32H735G-DK	NUCLEO-H7A3ZI-Q	NUCLEO-H745ZI-Q	NUCLEO-H743ZI	NUCLEO-H723ZG
(Continued) **Demonstrations**	**Total number of demonstrations: 12**			**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
Total number of projects: 954				**86**	**56**	**23**	**94**	**74**	**59**	**193**	**76**	**85**	**29**	**92**	**87**

Revision history

Table 2. **Document revision history**

Date	Version	Changes
12-May-2017	1	Initial release.
05-Sep-2017	2	Updated applications and demonstrations in Table 1: STM32CubeH7 firmware examples.
02-Jan-2018	3	Updated Section 1: Reference documents. Updated Table 1: STM32CubeH7 firmware examples.
27-Jun-2018	4	Added Arm wordmark notice in Section 1 Reference documents. Replaced STM32Cube embedded firmware by STM32Cube MCU Package in the whole document. Updated Table *STM32CubeH7 firmware examples*.
27-Mar-2019	5	• Single-core architecture examples running on STM32H743I-EVAL anrd NUCLEO-H743ZI: no changes compare to previous document revision. • Added dual-core architecture examples running on STM32H747I-EVAL STM32H745I-DISCO, STM32H747I-DISCO, STM32H750B-DK and NUCLEO-H745ZI-Q. • Added support for LL drivers.
19-Dec-2019	6	Added STM32H7B3I-EVAL, STM32H7B3I-DK and NUCLEO-H7A3ZI-Q. Added UM2298 in Section 1 Reference documents. Added Examples_MIX in Section 2 STM32CubeH7 examples.
30-Jun-2020	7	Added STM32H735G-DK and NUCLEO-H723ZG boards.

Contents