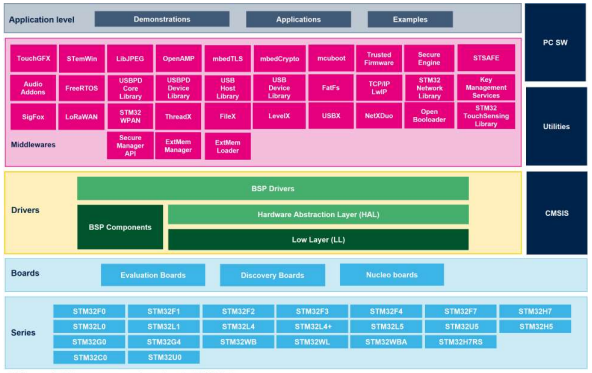
STM32CubeWB Firmware Examples for STM32WBxx Series

Copyright 2019 STMicroelectronics

The STM32CubeWB Firmware package comes with a rich set of examples running on STMicroelectronics boards, organized by board and provided with preconfigured projects for the main supported toolchains.



(1) The set of middleware components depends on the STM32 Series.

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

- **Examples** uses only the HAL and BSP drivers (Middleware not used), having as objective to demonstrate the product/peripherals features and usage, The examples are organized per peripheral (a folder for each peripheral, ex, TIM) and offers different complexity level from basic usage of a given peripheral (ex, PWM generation using timer) till integration of several peripherals(use DAC for signals generation with synchronization from TIM6 and DMA), Board resources usage is reduced to the strict minimum.
- **Examples_LL** uses only the LL drivers (HAL and Middleware not used), offering optimum implementation of typical use cases of the peripheral features and configuration procedures, The examples are organized per peripheral (a folder for each peripheral, ex, TIM) and runs exclusively on Nucleo board,
- **Examples_MIX** uses only HAL, BSP and LL drivers (Middleware are not used), having as objective to demonstrate how to use both HAL and LL APIs in the same application, to combine the advantages of both APIs (HAL offers high level and functionalities oriented APIs, with high portability level and hide product or IPs complexity to end user. While LL offers low level APIs at registers level with better optimization). The examples are organized per peripheral (a folder for each peripheral, ex, TIM) and runs exclusively on Nucleo board.
- **Applications** intends to demonstrate the product performance and how to use the different Middleware stacks available, The Applications are organized per Middleware (a folder for each Middleware, ex, USB Host) or product feature that need high level firmware bricks (ex, Audio), Integration Applications that use several Middleware stacks are provided as well.
- A Template project is provided to allow user to quickly build any firmware application on a given board.

The examples are located under STM32Cube_FW_WB_VX.Y.Z\Projects\, and all of them have the same structure:

- \Inc folder that contains all header files.
- \Src folder for the sources code.
- \EWARM, \MDK-ARM and \STM32CubeIDE folders contain the preconfigured project for each toolchain.
- A readme describing the example behavior and the environment required to run the example.

To run the example, you have to do the following:

- Open the example using your preferred toolchain.
- Rebuild all files and load the image into target memory.
- Run the example by following the readme instructions.
- *Note: refer to section "Development Toolchains and Compilers" and "Supported Devices and EVAL, Nucleo and Discovery boards" of the Firmware package release notes to know about the SW/HW environment used for the Firmware development and validation, The correct operation of the provided examples is not guaranteed on some environments, for example when using different compiler or board versions.*

The provided examples can be tailored to run on any compatible hardware; user simply need to update the BSP drivers for his board, if it has the same hardware functions (LED, LCD display, pushbuttons...etc.). The BSP is based on a modular architecture that allows it to be ported easily to any hardware by just implementing the low level routines.

The table below contains the list of examples provided within STM32Cube_FW_WB Firmware package.
In this table, the label **CubeMX** means the projects have been created using STM32CubeMX, the STM32Cube initialization code generator. Those projects can be opened with this tools to modify the projects itself. The others projects are manually created to demonstrate the product features.

Reference materials available on www.st.com/stm32cubefw

- UM2550 : Getting started with STM32CubeWB for STM32WBxx Series.
- UM2551 : STM32CubeWB Nucleo demonstration firmware.
- UM2442 : Description of STM32WBxx HAL drivers.
- UM1721 : Developing Applications on STM32Cube with FatFs.
- UM1722 : Developing Applications on STM32Cube with RTOS.
- AN5289 : Building a Wireless application

Level	Module Name	Project Name	Description	STM32WB5MM-DK	P-NUCLEO-WB55.USBDongle	P-NUCLEO-WB55.Nucleo	NUCLEO-WB15CC	B-WB1M-WPAN1
Templates	-	Starter project	This projects provides a reference template that can be used to build any firmware application.	CubeMx	-	CubeMx	CubeMx	X
			Total number of templates: 4	1	0	1	1	1
Templates_LL	-	Starter project	This projects provides a reference template through the LL API that can be used to build any firmware application.	CubeMx	-	CubeMx	CubeMx	X
			Total number of templates_ll: 4	1	0	1	1	1
Examples	-	BSP	How to use the different BSP drivers of external devices mounted on : STM32WB5MM-DK board.	X	X	-	-	-
	ADC	ADC_AnalogWatchdog	How to use the ADC peripheral to perform conversions with an analog watchdog and out-of-window interrupts enabled.	-	-	CubeMx	-	-
		ADC_MultiChannelSingleConversion	Use ADC to convert a several channels using sequencer in discontinuous mode, conversion data are transferred by DMA into an array, indefinitely (circular mode).	-	-	CubeMx	CubeMx	-
		ADC_Oversampling	Use ADC to convert a single channel but using oversampling feature to increase resolution.	-	-	CubeMx	-	-
		ADC_SingleConversion_TriggerSW_IT	How to use the ADC to convert a single channel at each software start. This example uses the interrupt programming model.	-	-	CubeMx	CubeMx	-
		ADC_SingleConversion_TriggerTimer_DMA	Use ADC to convert a single channel at each trig from timer, conversion data are transferred by DMA into an array, indefinitely (circular mode).	-	-	CubeMx	CubeMx	-
	BSP	BSP_Example	This example describes how to use the bsp API.	-	-	CubeMx	CubeMx	X
	COMP	COMP_CompareGpioVsVrefInt_IT	How to configure the COMP peripheral to compare the external voltage applied on a specific pin with the Internal Voltage Reference.	-	-	CubeMx	CubeMx	-
		COMP_CompareGpioVsVrefInt_Window_IT	This example shows how to make an analog watchdog using the COMP peripherals in window mode.	-	-	CubeMx	-	-
	CORTEX	CORTEXM_SysTick	How to use the default SysTick configuration with a 1 ms timebase to toggle LEDs.	-	-	-	CubeMx	-
	CRC	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).	-	-	CubeMx	CubeMx	-
		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.	-	-	CubeMx	-	-
	CRYP	CRYP_AESModes	How to use the CRYP peripheral to encrypt and decrypt data using AES in chaining modes (ECB, CBC, CTR).	-	-	CubeMx	-	-
		CRYP_DMA	How to use the AES1 peripheral to encrypt and decrypt data using AES 128 Algorithm with ECB chaining mode in DMA mode.	-	-	CubeMx	CubeMx	-
	Cortex	CORTEXM_MPU	Presentation of the MPU feature. This example configures a memory area as privileged read-only, and attempts to perform read and write operations in different modes.	-	-	CubeMx	-	-
		CORTEXM_ModePrivilege	How to modify the Thread mode privilege access and stack. Thread mode is entered on reset or when returning from an exception.	-	-	CubeMx	-	-
		CORTEXM_SysTick	How to use the default SysTick configuration with a 1 ms timebase to toggle LEDs.	-	-	CubeMx	-	-

DMA	DMA_FLASHToRAM	How to use a DMA to transfer a word data buffer from Flash memory to embedded SRAM through the HAL API.	-	-	CubeMx	CubeMx	-
	DMA_MUXSYNC	How to use the DMA with the DMAMUX to synchronize a transfer with the LPTIM1 output signal. USART1 is used in DMA synchronized mode to send a countdown from 10 to 00, with a period of 2 seconds.	-	-	CubeMx	CubeMx	-
	DMA_MUX_RequestGen	How to use the DMA with the DMAMUX request generator to generate DMA transfer requests upon an External line 4 rising edge signal.	-	-	CubeMx	CubeMx	-
FLASH	FLASH_EraseProgram	How to configure and use the FLASH HAL API to erase and program the internal Flash memory.	-	-	CubeMx	CubeMx	-
	FLASH_WriteProtection	How to configure and use the FLASH HAL API to enable and disable the write protection of the internal Flash memory.	-	-	CubeMx	CubeMx	-
GPIO	GPIO_EXTI	How to configure external interrupt lines.	-	-	CubeMx	CubeMx	-
	GPIO_IOToggle	How to configure and use GPIOs through the HAL API.	-	-	CubeMx	CubeMx	-
HAL	HAL_TimeBase	How to customize HAL using a general-purpose timer as main source of time base, instead of Systick.	-	-	CubeMx	CubeMx	-
	HAL_TimeBase_RTC_ALARM	How to customize HAL using RTC alarm as main source of time base, instead of Systick.	-	-	CubeMx	-	-
	HAL_TimeBase_RTC_WKUP	How to customize HAL using RTC wakeup as main source of time base, instead of Systick.	-	-	CubeMx	CubeMx	-
	HAL_TimeBase_TIM	How to customize HAL using a general-purpose timer as main source of time base instead of Systick.	-	-	CubeMx	CubeMx	-
HSEM	HSEM_ProcessSync	How to use a HW semaphore to synchronize 2 process.	-	-	CubeMx	-	-
	HSEM_ReadLock	How to enable, take then release semaphore using 2 different Process.	-	-	CubeMx	-	-
I2C	I2C_TwoBoards_AdvCom1T	How to handle several I2C data buffer transmission/reception between a master and a slave device, using an interrupt.	-	-	CubeMx	-	-
	I2C_TwoBoards_ComDMA	How to handle I2C data buffer transmission/reception between two boards, via DMA.	-	-	CubeMx	CubeMx	-
	I2C_TwoBoards_ComIT	How to handle I2C data buffer transmission/reception between two boards, using an interrupt.	-	-	CubeMx	CubeMx	-
	I2C_TwoBoards_ComPolling	How to handle I2C data buffer transmission/reception between two boards, in polling mode.	-	-	CubeMx	-	-
	I2C_TwoBoards_RestartAdvCom1T	How to perform multiple I2C data buffer transmission/reception between two boards, in interrupt mode and with restart condition.	-	-	CubeMx	-	-
	I2C_TwoBoards_RestartComIT	How to handle single I2C data buffer transmission/reception between two boards, in interrupt mode and with restart condition.	-	-	CubeMx	-	-
	I2C_WakeUpFromStop	How to handle I2C data buffer transmission/reception between two boards, using an interrupt when the device is in Stop mode.	-	-	CubeMx	-	-
	I2C_WakeUpFromStop2	How to handle I2C data buffer transmission/reception between two boards, using an interrupt when the device is in Stop 2 mode.	-	-	CubeMx	-	-
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.	-	-	CubeMx	-	-
	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.	-	-	CubeMx	CubeMx	-
LCD	LCD_SegmentsDrive	How to drive a LCD Glass using STM32WBxx hal driver.	-	-	CubeMx	-	-
LPTIM	LPTIM_PWMExternalClock	How to configure and use, through the HAL LPTIM API, the LPTIM peripheral using an external counter clock, to generate a PWM signal at the lowest power consumption.	-	-	CubeMx	CubeMx	-
	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.	-	-	CubeMx	-	-
	LPTIM_PulseCounter	How to configure and use, through the LPTIM HAL API, the LPTIM peripheral to count pulses.	-	-	CubeMx	CubeMx	-
	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.	-	-	CubeMx	-	-
PKA	PKA_ECCscalarMultiplication	How to use the PKA peripheral to execute ECC scalar multiplication. This allows generating a public key from a private key.	-	-	CubeMx	-	-
	PKA_ECCscalarMultiplication_IT	How to use the PKA peripheral to execute ECC scalar multiplication. This allows generating a public key from a private key in interrupt mode.	-	-	CubeMx	-	-
	PKA_ECDSA_Sign	How to compute a signed message regarding the Elliptic curve digital signature algorithm (ECDSA).	-	-	CubeMx	CubeMx	-
	PKA_ECDSA_Sign_IT	How to compute a signed message regarding the Elliptic curve digital signature algorithm (ECDSA) in interrupt mode.	-	-	CubeMx	-	-
	PKA_ECDSA_Verify	How to determine if a given signature is valid regarding the Elliptic curve digital signature algorithm (ECDSA).	-	-	CubeMx	-	-
	PKA_ECDSA_Verify_IT	How to determine if a given signature is valid regarding the Elliptic curve digital signature algorithm (ECDSA) in interrupt mode.	-	-	CubeMx	CubeMx	-
	PKA_ModularExponentiation	How to use the PKA peripheral to execute modular exponentiation. This allows ciphering/deciphering a text.	-	-	CubeMx	-	-
	PKA_ModularExponentiationCRT	How to compute the Chinese Remainder Theorem (CRT) optimization.	-	-	CubeMx	-	-
	PKA_ModularExponentiationCRT_IT	How to compute the Chinese Remainder Theorem (CRT) optimization in interrupt mode.	-	-	CubeMx	-	-
	PKA_ModularExponentiation_IT	How to use the PKA peripheral to execute modular exponentiation. This allows ciphering/deciphering a text in interrupt mode.	-	-	CubeMx	-	-
	PKA_PointCheck	How to use the PKA peripheral to determine if a point is on a curve. This allows validating an external public key.	-	-	CubeMx	-	-
	PKA_PointCheck_IT	How to use the PKA peripheral using interrupt mode to determine if a point is on a curve. This allows validating an external public key.	-	-	CubeMx	-	-
PWR	PWR_LPRUN	How to enter and exit the Low-power run mode.	-	-	CubeMx	CubeMx	-
	PWR_LPSLEEP	How to enter the Low-power sleep mode and wake up from this mode by using an interrupt.	-	-	CubeMx	CubeMx	-
	PWR_PVD	How to configure the programmable voltage detector by using an external interrupt line. External DC supply must be used to supply Vdd.	-	-	CubeMx	CubeMx	-
	PWR_STANDBY_RTC	How to enter the Standby mode and wake-up from this mode by using an external reset or the RTC wakeup timer.	-	-	CubeMx	CubeMx	-
	PWR_STOP2_RTC	How to enter the Stop 2 mode and wake-up from this mode using an external reset or RTC wakeup timer.	-	-	CubeMx	-	-
QSPI	QSPI_ExecuteInPlace	This example describes how to execute a part of the code from the QSPI memory. To do this, a section is created where the function is stored.	CubeMx	-	CubeMx	-	-
	QSPI_MemoryMapped	This example describes how to erase part of the QSPI memory, write data in DMA mode and access to QSPI memory in memory-mapped mode to check the data in a forever loop.	CubeMx	-	CubeMx	-	-
	QSPI_ReadWrite_DMA	This example describes how to erase part of the QSPI memory, write data in DMA mode, read data in DMA mode and compare the result in a forever loop.	CubeMx	-	CubeMx	-	-
	QSPI_ReadWrite_IT	This example describes how to erase part of the QSPI memory, write data in IT mode, read data in IT mode and compare the result in a forever loop.	CubeMx	-	CubeMx	-	-
RCC	RCC_CRS_Synchronization_IT	Configuration of the clock recovery service (CRS) in interrupt mode, using the RCC HAL API.	-	-	CubeMx	-	-

		RCC_CRS_Synchronization_Polling	Configuration of the clock recovery service (CRS) in Polling mode, using the RCC HAL API.	-	-	CubeMx	-	-
		RCC_ClockConfig	The main purpose of this example is to serve as a reference for clock configuration operation needed by most of the BLE applications.	CubeMx	-	CubeMx	CubeMx	-
	RNG	RNG_MultiRNG	Configuration of the RNG using the HAL API. This example uses the RNG to generate 32-bit long random numbers.	-	-	CubeMx	CubeMx	-
		RNG_MultiRNG_IT	Configuration of the RNG using the HAL API. This example uses RNG interrupts to generate 32-bit long random numbers.	-	-	CubeMx	-	-
	RTC	RTC_Alarm	Configuration and generation of an RTC alarm using the RTC HAL API.	-	-	CubeMx	CubeMx	-
		RTC_Calendar	Configuration of the calendar using the RTC HAL API.	-	-	CubeMx	-	-
		RTC_LSI	Use of the LSI clock source autocalibration to get a precise RTC clock.	-	-	CubeMx	-	-
		RTC_Tamper	Configuration of the RTC HAL API to write/read data to/from RTC Backup registers.	-	-	CubeMx	CubeMx	-
		RTC_TimeStamp	Configuration of the RTC HAL API to demonstrate the timestamp feature.	-	-	CubeMx	-	-
	SAI	SAI_AudioPlay	Use of the SAI HAL API to play an audio file in DMA circular mode and handle the buffer update. Refer to Projects\STM32WB5MM-DK\Examples\BSP.	X	-	-	-	-
	SPI	SPI_FullDuplex_ComDMA_Master	Data buffer transmission/reception between two boards via SPI using DMA. This example is for the Master board.	-	-	CubeMx	CubeMx	-
		SPI_FullDuplex_ComDMA_Slave	Data buffer transmission/reception between two boards via SPI using DMA. This example is for the Slave board.	-	-	CubeMx	CubeMx	-
		SPI_FullDuplex_ComIT_Master	Data buffer transmission/reception between two boards via SPI using Interrupt mode. This example is for the Master board.	-	-	CubeMx	-	-
		SPI_FullDuplex_ComIT_Slave	Data buffer transmission/reception between two boards via SPI using Interrupt mode. This example is for the Slave board.	-	-	CubeMx	-	-
		SPI_FullDuplex_ComPolling_Master	Data buffer transmission/reception between two boards via SPI using Polling mode. This example is for the Master board.	-	-	CubeMx	-	-
		SPI_FullDuplex_ComPolling_Slave	Data buffer transmission/reception between two boards via SPI using Polling mode. This example is for the Slave board.	-	-	CubeMx	-	-
	TIM	TIM_DMA	Use of the DMA with TIMER Update request to transfer data from memory to TIMER Capture Compare Register 3 (TIMx_CCR3).	-	-	CubeMx	-	-
		TIM_DMABurst	How to update the TIMER channel 1 period and duty cycle using the TIMER DMA burst feature.	-	-	CubeMx	-	-
		TIM_InputCapture	How to use the TIM peripheral to measure an external signal frequency.	-	-	CubeMx	-	-
		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).	-	-	CubeMx	CubeMx	-
		TIM_OCInactive	Configuration of the TIM peripheral in Output Compare Inactive mode with the corresponding Interrupt requests for each channel.	-	-	CubeMx	-	-
		TIM_OCToggle	Configuration of the TIM peripheral to generate four different signals at four different frequencies.	-	-	CubeMx	-	-
		TIM_OnePulse	Use of the TIM peripheral to generate a single pulse when an external signal rising edge is received on the timer input pin.	-	-	X	-	-
		TIM_PWMInput	How to use the TIM peripheral to measure the frequency and duty cycle of an external signal.	-	-	CubeMx	CubeMx	-
		TIM_PWMOutput	This example shows how to configure the TIM peripheral in PWM (Pulse Width Modulation) mode.	-	-	CubeMx	CubeMx	-
		TIM_TimeBase	This example shows how to configure the TIM peripheral to generate a time base of one second with the corresponding Interrupt request.	-	-	CubeMx	-	-
	TSC	TSC_BasicAcquisition_Interrupt	Use of the TSC HAL API to perform continuous acquisitions of one channel in Interrupt mode.	-	-	X	-	-
	UART	UART_Console	UART transmission (printf/getchar) via console with user interaction.	CubeMx	-	CubeMx	-	-
		UART_HyperTerminal_DMA	UART transmission (transmit/receive) in DMA mode between a board and an HyperTerminal PC application.	-	-	CubeMx	CubeMx	-
		UART_HyperTerminal_IT	UART transmission (transmit/receive) in Interrupt mode between a board and an HyperTerminal PC application.	-	-	CubeMx	CubeMx	-
		UART_Printf	Re-routing of the C library printf function to the UART.	-	-	CubeMx	CubeMx	-
		UART_ReceptionToIdle_CircularDMA	How to use the HAL UART API for reception to IDLE event in circular DMA mode.	-	-	CubeMx	-	-
		UART_TwoBoards_ComDMA	UART transmission (transmit/receive) in DMA mode between two boards.	-	-	CubeMx	-	-
		UART_TwoBoards_ComIT	UART transmission (transmit/receive) in Interrupt mode between two boards.	-	-	CubeMx	-	-
		UART_TwoBoards_ComPolling	UART transmission (transmit/receive) in Polling mode between two boards.	-	-	CubeMx	-	-
	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.	-	-	CubeMx	CubeMx	-
			Total number of examples: 152	8	1	100	42	1
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.	-	-	CubeMx	-	-
		ADC_ContinuousConversion_TriggerSW	How to use an ADC peripheral to perform continuous ADC conversions on a channel, from a software start. This example is based on the STM32WBxx ADC LL API. The peripheral initialization is done using LL unitary service functions for optimization purposes (performance and size).	-	-	X	-	-
		ADC_ContinuousConversion_TriggerSW_Init	How to use an ADC peripheral to perform continuous ADC conversions on a channel, from a software start. This example is based on the STM32WBxx ADC LL API. The peripheral initialization is done using LL initialization function to demonstrate LL init usage.	-	-	CubeMx	-	-
		ADC_ContinuousConversion_TriggerSW_LowPower_Init	How to use an ADC peripheral with ADC low-power features. This example is based on the STM32WBxx ADC LL API. The peripheral initialization is done using LL unitary service functions for optimization purposes (performance and size).	-	-	CubeMx	-	-
		ADC_GroupsRegularInjected_Init	How to use an ADC peripheral with both ADC groups (regular and injected) in their intended use cases.	-	-	CubeMx	-	-
		ADC_MultiChannelSingleConversion_Init	How to use an ADC peripheral to convert several channels. ADC conversions are performed successively in a scan sequence.	-	-	-	CubeMx	-
		ADC_Oversampling_Init	How to use an ADC peripheral with ADC oversampling.	-	-	CubeMx	-	-
		ADC_SingleConversion_TriggerSW_DMA_Init	How to use an ADC peripheral to perform a single ADC conversion on a channel, at each software start. This example uses the DMA programming model (for polling or interrupt programming models, refer to other examples).	-	-	CubeMx	-	-
		ADC_SingleConversion_TriggerSW_IT_Init	How to use an ADC peripheral to perform a single ADC conversion on a channel, at each software start. This example uses the interrupt programming model (for polling or DMA programming models, please refer to other examples).	-	-	CubeMx	CubeMx	-
		ADC_SingleConversion_TriggerSW_Init	How to use an ADC peripheral to perform a single ADC conversion on a channel at each software start. This example uses the polling programming model (for interrupt or DMA programming models, please refer to other examples).	-	-	CubeMx	-	-
		ADC_SingleConversion_TriggerTimer_DMA_Init	How to use an ADC peripheral to perform a single ADC conversion on a channel at each trigger event from a timer. Converted data is indefinitely transferred by DMA into a table (circular mode).	-	-	CubeMx	CubeMx	-
		ADC_TemperatureSensor	How to use an ADC peripheral to perform a single ADC conversion on the internal temperature sensor and calculate the temperature in degrees Celsius.	-	-	X	-	-
	COMP	COMP_CompareGpioVsVrefInt_IT	How to use a comparator peripheral to compare a voltage level applied on a GPIO pin to the internal voltage reference (VREFINT), in interrupt mode. This example is based on the STM32WBxx COMP LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	-	-	X	-	-
		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. This example is based on the STM32WBxx COMP LL API. The peripheral initialization uses the LL initialization function to demonstrate LL Init usage.	-	-	CubeMx	-	-

	COMP_CompareGpioVsVrefInt_OutputGpio_Init	How to use a comparator peripheral to compare a voltage level applied on a GPIO pin to the internal voltage reference (VREFINT). The comparator output is connected to a GPIO. This example is based on the STM32WBxx COMP LL API.	-	-	CubeMx	-	-
	COMP_CompareGpioVsVrefInt_Window_IT_Init	How to use a pair of comparator peripherals to compare a voltage level applied on a GPIO pin to two thresholds: the internal voltage reference (VREFINT) and a fraction of the internal voltage reference (VREFINT/2), in interrupt mode. This example is based on the STM32WBxx COMP LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	-	-	CubeMx	-	-
CORTEX	CORTEX_MPU	Presentation of the MPU feature. This example configures a memory area as privileged read-only, and attempts to perform read and write operations in different modes.	-	-	CubeMx	-	-
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).	-	-	CubeMx	CubeMx	-
	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).	-	-	CubeMx	-	-
CRS	CRS_Synchronization_IT	How to configure the clock recovery service in IT mode through the STM32WBxx CRS LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	-	-	CubeMx	-	-
	CRS_Synchronization_Polling	How to configure the clock recovery service in polling mode through the STM32WBxx CRS LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	-	-	CubeMx	-	-
DMA	DMA_CopyFromFlashToMemory	How to use a DMA channel 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	-	-
	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.	-	-	CubeMx	CubeMx	-
EXTI	EXTI_ToggleLedOnIT	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 STM32WBxx LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	-	-	X	-	-
	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 STM32WBxx LL API. Peripheral initialization is done using LL initialization function to demonstrate LL init usage.	-	-	CubeMx	CubeMx	-
GPIO	GPIO_InfiniteLedToggling	How to configure and use GPIOs to toggle the on-board user LEDs every 250 ms. This example is based on the STM32WBxx LL API. The peripheral is initialized with LL unitary service functions to optimize for performance and size.	-	-	X	-	-
	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 STM32WBxx LL API. The peripheral is initialized with LL initialization function to demonstrate LL init usage.	-	-	CubeMx	CubeMx	-
HSEM	HSEM_DualProcess	How to use the low-layer HSEM API to initialize, lock, and unlock hardware semaphore in the context of two processes accessing the same resource.	-	-	CubeMx	-	-
	HSEM_DualProcess_IT	How to use the low-layer HSEM API to initialize, lock, and unlock hardware semaphore in the context of two processes accessing the same resource. This example configures HSEM in interrupt mode to trigger an interrupt when a process takes the semaphore.	-	-	CubeMx	CubeMx	-
I2C	I2C_OneBoard_AdvCommunication_DMAAndIT_Init	How to exchange data between an I2C master device in DMA mode and an I2C slave device in interrupt mode. The peripheral is initialized with LL unitary service functions to optimize for performance and size.	-	-	CubeMx	-	-
	I2C_OneBoard_Communication_DMAAndIT_Init	How to transmit data bytes from an I2C master device using DMA mode to an I2C slave device using interrupt mode. The peripheral is initialized with LL unitary service functions to optimize for performance and size.	-	-	CubeMx	-	-
	I2C_OneBoard_Communication_IT	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	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.	-	-	CubeMx	-	-
	I2C_OneBoard_Communication_PollingAndIT_Init	How to transmit data bytes from an I2C master device using polling mode to an I2C slave device using interrupt mode. The peripheral is initialized with LL unitary service functions to optimize for performance and size.	-	-	CubeMx	-	-
	I2C_TwoBoards_MasterRx_SlaveTx_IT_Init	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.	-	-	CubeMx	CubeMx	-
	I2C_TwoBoards_MasterTx_SlaveRx_DMA_Init	How to transmit data bytes from an I2C master device using DMA mode to an I2C slave device using DMA mode. The peripheral is initialized with LL unitary service functions to optimize for performance and size.	-	-	CubeMx	CubeMx	-
	I2C_TwoBoards_MasterTx_SlaveRx_Init	How to transmit data bytes from an I2C master device using polling mode to an I2C slave device using interrupt mode. The peripheral is initialized with LL unitary service functions to optimize for performance and size.	-	-	CubeMx	-	-
	I2C_TwoBoards_WakeUpFromStop2_IT_Init	How to handle the reception of a data byte from an I2C slave device in Stop 2 mode by an I2C master device, both using interrupt mode. The peripheral is initialized with LL unitary service functions to optimize for performance and size.	-	-	CubeMx	-	-
	I2C_TwoBoards_WakeUpFromStop_IT_Init	How to handle the reception of a data byte from an I2C slave device in Stop 1 mode by an I2C master device, both using interrupt mode. The peripheral is initialized with LL unitary service functions to optimize for performance and size.	-	-	CubeMx	-	-
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 (SW1) is pressed, The peripheral is initialized with LL unitary service functions to optimize for performance and size.	-	-	CubeMx	CubeMx	-
LPTIM	LPTIM_PulseCounter	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 STM32WBxx LPTIM LL API. The peripheral is initialized with LL unitary service functions to optimize for performance and size.	-	-	X	-	-
	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 STM32WBxx LPTIM LL API. The peripheral is initialized with LL initialization function to demonstrate LL init usage.	-	-	CubeMx	CubeMx	-
LPUART	LPUART_WakeUpFromStop2_Init	Configuration of GPIO and LPUART peripherals to allow characters received on LPUART_RX pin to wake up the MCU from low-power "STOP2" mode. This example is based on the LPUART LL API. The peripheral initialization uses LL initialization function to demonstrate LL init usage.	-	-	CubeMx	-	-
	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 "STOP0" mode. This example is based on the LPUART LL API. The peripheral initialization uses LL initialization function to demonstrate LL init usage.	-	-	CubeMx	CubeMx	-
PKA	PKA_ECDSA_Sign	How to use the low-layer PKA API to generate an ECDSA signature.	-	-	CubeMx	CubeMx	-
	PKA_ModularExponentiation	How to use the low-layer PKA API to execute RSA modular exponentiation.	-	-	CubeMx	-	-
PWR	PWR_EnterStandbyMode	How to enter the Standby mode and wake up from this mode by using an external reset or a wakeup interrupt.	-	-	CubeMx	CubeMx	-
	PWR_EnterStopMode	How to enter the Stop 2 mode.	-	-	CubeMx	CubeMx	-
	PWR_OptimizedRunMode	How to increase/decrease frequency and VCORE and how to enter/exit the Low-power run mode.	-	-	CubeMx	CubeMx	-
	PWR_SMPS_16MHZ_HSI	This example shows how to use power converters of STM32WB (SMPS, LDO and LP-LDO) depending on Vdd voltage and low-power mode. In this example, the system clock source is the HSI at 16MHz.	-	-	CubeMx	-	-
	PWR_SMPS_64MHZ_MSI_PLL	This example shows how to use power converters of STM32WB (SMPS, LDO and LP-LDO) depending on Vdd voltage and low-power mode. In this example, the system clock source is the MSI PLL at 64MHz.	-	-	CubeMx	-	-
RCC	RCC_HWAutoMSICalibration	Use of the MSI clock source hardware autocalibration and LSE clock (PLL mode) to obtain a precise MSI clock.	-	-	CubeMx	-	-
	RCC_OutputSystemClockOnMCO	Configuration of MCO pin (PA8) to output the system clock.	-	-	CubeMx	-	-
	RCC_UseHSEasSystemClock	Use of the RCC LL API to start the HSE and use it as system clock.	-	-	CubeMx	-	-
	RCC_UseHSI_PLLasSystemClock	Modification of the PLL parameters in run time.	-	-	CubeMx	CubeMx	-
RNG	RNG_GenerateRandomNumbers	Configuration of the RNG to generate 32-bit long random numbers. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	-	-	CubeMx	-	-
	RNG_GenerateRandomNumbers_IT	Configuration of 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).	-	-	CubeMx	CubeMx	-
RTC	RTC_Alarm	Configuration of 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	-	-
	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.	-	-	CubeMx	-	-
	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).	-	-	CubeMx	-	-
	RTC_ExitStandbyWithWakeUpTimer_Init	Configuration of the RTC to wake up from Standby mode using the RTC Wakeup timer. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	-	-	CubeMx	-	-
	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).	-	-	CubeMx	-	-
	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).	-	-	CubeMx	-	-
SPI	SPI_OneBoard_HalfDuplex_DMA	Configuration of 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 STM32WBxx SPI LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	-	-	X	-	-

		SPI_OneBoard_HalfDuplex_DMA_Init	Configuration of 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 STM32WBxx SPI LL API. The peripheral initialization uses the LL initialization function to demonstrate LL Init usage.	-	-	CubeMx	-	-
		SPI_OneBoard_HalfDuplex_IT_Init	Configuration of GPIO and SPI peripherals to transmit bytes from an SPI Master device to an SPI Slave device in Interrupt mode. This example is based on the STM32WBxx SPI LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	-	-	CubeMx	-	-
		SPI_TwoBoards_FullDuplex_DMA_Master_Init	Data buffer transmission and reception via SPI using DMA mode. This example is based on the STM32WBxx SPI LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size). This example is for the Master board.	-	-	CubeMx	CubeMx	-
		SPI_TwoBoards_FullDuplex_DMA_Slave_Init	Data buffer transmission and reception via SPI using DMA mode. This example is based on the STM32WBxx SPI LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size). This example is for the Slave board.	-	-	CubeMx	CubeMx	-
		SPI_TwoBoards_FullDuplex_IT_Master_Init	Data buffer transmission and reception via SPI using Interrupt mode. This example is based on the STM32WBxx SPI LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size). This example is for the Master board.	-	-	CubeMx	-	-
		SPI_TwoBoards_FullDuplex_IT_Slave_Init	Data buffer transmission and reception via SPI using Interrupt mode. This example is based on the STM32WBxx SPI LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size). This example is for the Slave board.	-	-	CubeMx	-	-
	TIM	TIM_BreakAndDeadtime	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.	-	-	X	-	-
		TIM_DMA_Init	Use of the DMA with a timer update request to transfer data from memory to Timer Capture Compare Register 3 (TIMx_CCR3). This example is based on the STM32WBxx TIM LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	-	-	CubeMx	-	-
		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 STM32WBxx TIM LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	-	-	CubeMx	-	-
		TIM_OnePulse	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 STM32WBxx TIM LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	-	-	X	-	-
		TIM_OutputCompare_Init	Configuration of the TIM peripheral to generate an output waveform in different output compare modes. This example is based on the STM32WBxx TIM LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	-	-	CubeMx	-	-
		TIM_PWMOutput	Use of a timer peripheral to generate a PWM output signal and update the PWM duty cycle. This example is based on the STM32WBxx TIM LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	-	-	X	-	-
		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 STM32WBxx TIM LL API. The peripheral initialization uses LL initialization function to demonstrate LL Init.	-	-	CubeMx	-	-
		TIM_TimeBase_Init	Configuration of the TIM peripheral to generate a timebase. This example is based on the STM32WBxx TIM LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	-	-	CubeMx	CubeMx	-
	USART	USART_Communication_Rx_IT	Configuration of 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	-	-
		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).	-	-	CubeMx	-	-
		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). Virtual Com Port feature of STLINK is used for UART communication between board and PC.	-	-	CubeMx	-	-
		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.	-	-	CubeMx	CubeMx	-
		USART_Communication_Rx_IT_VCP_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. Virtual Com Port (VCP) feature of STLINK is used for UART communication between board and PC.	-	-	CubeMx	-	-
		USART_Communication_TxRx_DMA_Init	This example shows how to configure GPIO and USART peripheral to send characters asynchronously to/from an HyperTerminal (PC) in DMA mode. This example is based on STM32WBxx USART LL API. Peripheral initialization is done using LL unitary services functions for optimization purpose (performance and size).	-	-	CubeMx	-	-
		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 STM32WBxx USART LL API. Peripheral initialization is done using LL unitary services functions for optimization purpose (performance and size).	-	-	CubeMx	-	-
		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 STM32WBxx USART LL API. Peripheral initialization is done using LL unitary services functions for optimization purpose (performance and size). Virtual Com Port (VCP) feature of STLINK is used for UART communication between board and PC.	-	-	CubeMx	-	-
		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 STM32WBxx USART LL API. Peripheral initialization is done using LL unitary services functions for optimization purpose (performance and size).	-	-	CubeMx	CubeMx	-
		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 STM32WBxx USART LL API. Peripheral initialization is done using LL unitary services functions for optimization purpose (performance and size). Virtual Com Port (VCP) feature of STLINK is used for UART communication between board and PC.	-	-	CubeMx	-	-
		USART_WakeUpFromStop1_Init	Configuration of GPIO and USART1 peripherals to allow the characters received on USART_RX pin to wake up the MCU from low-power "STOP1" mode.	-	-	CubeMx	-	-
		USART_WakeUpFromStop_Init	Configuration of GPIO and USART1 peripherals to allow the characters received on USART_RX pin to wake up the MCU from low-power "STOP0" mode.	-	-	CubeMx	-	-
	UTILS	UTILS_ConfigureSystemClock	Use of UTILS LL API to configure the system clock using PLL with HSI as source clock.	-	-	CubeMx	-	-
		UTILS_ReadDeviceInfo	This example reads the UID, Device ID and Revision ID and saves them into a global information buffer.	-	-	CubeMx	CubeMx	-
	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).	-	-	CubeMx	CubeMx	-
		Total number of examples_ll: 118		0	0	92	26	0
Examples_MIX	ADC	ADC_SingleConversion_TriggerSW_IT	How to use the ADC to perform a single ADC channel conversion at each software start. This example uses the interrupt programming model (for polling and DMA programming models, please refer to other examples). It is based on the STM32WBxx ADC HAL and LL API. The LL API is used for performance improvement.	-	-	CubeMx	-	-
	CRC	CRC_PolynomialUpdate	How to use the CRC peripheral through the STM32WBxx CRC HAL and LL API.	-	-	CubeMx	-	-
	DMA	DMA_FLASHToRAM	How to use a DMA to transfer a word data buffer from Flash memory to embedded SRAM through the STM32WBxx DMA HAL and LL API. The LL API is used for performance improvement.	-	-	CubeMx	CubeMx	-
	I2C	I2C_OneBoard_ComSlave7_10bits_IT	How to perform I2C data buffer transmission/reception between one master and two slaves with different address sizes (7-bit or 10-bit). This example uses the STM32WBxx I2C HAL and LL API (LL API usage for performance improvement) and an interrupt.	-	-	CubeMx	-	-
	PWR	PWR_STOP1	How to enter the STOP 1 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).	-	-	CubeMx	CubeMx	-
	SPI	SPI_FullDuplex_ComPolling_Master	Data buffer transmission/reception between two boards via SPI using Polling mode. This example is for the Master board.	-	-	CubeMx	-	-
		SPI_FullDuplex_ComPolling_Slave	Data buffer transmission/reception between two boards via SPI using Polling mode. This example is for the Slave board.	-	-	CubeMx	-	-
		SPI_HalfDuplex_ComPollingIT_Master	Data buffer transmission/reception between two boards via SPI using Polling (LL driver) and Interrupt modes (HAL driver). This example is for the Master board.	-	-	CubeMx	CubeMx	-
		SPI_HalfDuplex_ComPollingIT_Slave	Data buffer transmission/reception between two boards via SPI using Polling (LL driver) and Interrupt modes (HAL driver). This example is for the Slave board.	-	-	CubeMx	CubeMx	-
	TIM	TIM_PWMInput	Use of the TIM peripheral to measure an external signal frequency and duty cycle.	-	-	CubeMx	CubeMx	-
	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 STM32WBxx UART HAL and LL API, the LL API being used for performance improvement.	-	-	CubeMx	CubeMx	-
		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 STM32WBxx UART HAL and LL API, the LL API being used for performance improvement.	-	-	CubeMx	-	-
		Total number of examples_mix: 18		0	0	12	6	0
Applications	BLE	BLE_AT_Server	How to demonstrate Point-to-Point communication using BLE component (as GATT server).	-	-	X	-	-
		BLE_Beacon	How to advertise 3 types of beacon (tlm, uuid, url).	-	-	CubeMx	-	-
		BLE_BloodPressure	How to use the Blood Pressure profile as specified by the BLE SIG.	-	-	CubeMx	-	-
		BLE_CableReplacement	How to use the Point-to-Point communication using BLE component.	-	-	X	-	-
		BLE_Custom	@note This application is to demonstrate that a BLE_Custom application can be created using CubeMX.	-	-	CubeMx	-	-
		BLE_DataThroughput	How to use data throughput via notification from server to client using BLE component.	-	-	X	X	-
		BLE_HR_p2pServer	@note This application includes two BLE services, the first one is a BLE_P2P_Server like, including two characteristics; the second one is a BLE_Heart_Rate like with three characteristics.	-	-	-	-	CubeMx
		BLE_HR_p2p_Sensor	@note This application includes three BLE services, the first one is a BLE_P2P_Server like, including two characteristics; the second one is a BLE_Heart_Rate like with three characteristics.	-	-	-	-	CubeMx

	BLE_HealthThermometer	How to use the Health Thermometer profile as specified by the BLE SIG.	-	-	CubeMx	-	-
	BLE_HeartRate	How to use the Heart Rate profile as specified by the BLE SIG.	CubeMx	X	CubeMx	CubeMx	-
	BLE_HeartRateFreeRTOS	How to use the Heart Rate profile as specified by the BLE SIG with FreeRTOS.	-	-	CubeMx	-	-
	BLE_HeartRateFreeRTOS_ANCS	How to read notifications from Apple Notification Center Service (ANCS) as specified by Apple specification and use the Heart Rate profile as specified by the BLE SIG with FreeRTOS.	-	-	CubeMx	-	-
	BLE_HeartRateFreeRTOS_PLL	How to use the Heart Rate profile as specified by the BLE SIG with FreeRTOS.	-	-	X	-	-
	BLE_HeartRateThreadX	How to use the Heart Rate profile as specified by the BLE SIG using ThreadX OS.	-	-	X	-	-
	BLE_HeartRate_ANCS	How to read notifications from Apple Notification Center Service (ANCS) as specified by Apple specification and use Heart Rate profile as specified by the BLE SIG.	-	-	CubeMx	-	-
	BLE_HeartRate_PLL	How to use the Heart Rate profile as specified by the BLE SIG.	-	-	X	-	-
	BLE_HeartRate_ota	How to use the Heart Rate profile as specified by the BLE SIG to be downloaded with BLE OTA application.	-	-	X	X	-
	BLE_Hid	How to use the Human Interface Device profile as specified by the BLE SIG.	-	-	X	-	-
	BLE_MeshLightingLPN	This is the implementation of the BLE Mesh Low Power Node profile as specified by the BLE SIG, a Low Power Node with the capacity to be associated to a Proxy-Relay-Friend Node for Friendship.	-	X	X	-	-
	BLE_MeshLightingPRFNode	This is the implementation of the BLE Mesh Lighting profile as specified by the BLE SIG, a Proxy-Relay-Friend Node with the capacity to handle a Friendship with a Low Power Node.	-	X	X	-	-
	BLE_MeshLightingProvisioner	This is the implementation of the BLE Mesh Lighting Embedded Provisioner as specified by the BLE SIG, a Node with the capacity of creating MESH network from unprovisionned Nodes, like Proxy-Relay-Friend Nodes or Low Power Nodes.	-	-	X	-	-
	BLE_Mesh_Model_Sensor	This is the implementation of a BLE Mesh Sensor Model (Client and Server) as specified by the BLE SIG.	X	-	-	-	-
	BLE_Mesh_ThermometerSensor	This is the implementation of a BLE Mesh Vendor model as specified by the BLE SIG.	X	-	-	-	-
	BLE_Ota	OTA implementation to download a new image into the user flash.	-	-	X	X	-
	BLE_Peripheral_Lite	How to communicate with simple BLE peripheral with minimum activated features.	-	-	X	-	-
	BLE_Peripheral_Lite_EventCallbacks	How to communicate with simple BLE peripheral with minimum activated features with implemented BLE API event callbacks functions.	-	-	X	-	-
	BLE_Power_Peripheral	@note This application is to demonstrate that a BLE_Power_Peripheral application can be created using CubeMX.	-	-	CubeMx	-	-
	BLE_Proximity	How to use the Proximity profile as specified by the BLE SIG.	-	-	X	-	-
	BLE_RfWithFlash	How to demonstrate the capability to erase/write the flash while a Point-to-Point communication using BLE component is active.	-	-	X	-	-
	BLE_Sensor	This example is to demonstrate capabilities of STM32WB5MM Discovery Kit with the use of board sensors.	CubeMx	-	-	-	-
	BLE_TransparentMode	How to communicate with the STM32CubeMonitor-RF Tool using the transparent mode.	CubeMx	-	CubeMx	CubeMx	-
	BLE_TransparentModeVCP	How to communicate with the STM32CubeMonitor-RF Tool using the transparent mode through USB Virtual COM Port.	-	X	-	-	-
	BLE_p2pClient	How to demonstrate Point-to-Point communication using BLE component (as GATT client).	-	X	CubeMx	CubeMx	-
	BLE_p2pClient_Ext	Demontrates a BLE scanner with connections from an extended and a legacy advertising Two Nucleo STM32WB55xx boards (MB1355C) are used, one acting as GATT client, and one as GATT server.	-	-	X	X	-
	BLE_p2pRouteur	This example is to demonstrate Multipoint communication using BLE component.	-	X	CubeMx	-	-
	BLE_p2pServer	How to demonstrate Point-to-Point communication using BLE (as GATT server).	CubeMx	X	CubeMx	CubeMx	-
	BLE_p2pServerThreadX	How to demonstrate Point-to-Point communication using BLE component (as GATT server).	-	-	X	-	-
	BLE_p2pServer_Ext	Demonstrate multiple extended advertising sets.	-	-	X	X	-
	BLE_p2pServer_ota	How to demonstrate Point-to-Point communication using BLE component (peripherical as GATT server) to be downloaded with BLE OTA application.	-	-	X	X	-
BLE_LLD	BLE_LLD_Chat	How to create a "Chat" talk between 2 boards using terminals.	X	-	X	X	-
	BLE_LLD_Datarate	How to send BLE LLD packets in high data rate.	-	X	X	-	-
	BLE_LLD_Lowpower	How to send BLE LLD packets while using low power mode.	-	-	X	X	-
	BLE_LLD_Pressbutton	How to control remote LEDs with BLE LLD.	-	-	X	X	-
	BLE_LLD_Proximity	How to use BLE LLD to detect nearby boards? This solution can be used in contact tracing for Covid-19 since it provides an estimation of the distance with other boards.	-	-	X	-	-
BLE_Mac	BLE_Mac_Static	How to use BLE application and 802_15_4 Mac application in static concurrent mode.	-	-	X	-	-
BLE_Thread	Ble_Thread_Dyn	How to use BLE application and Thread application in dynamic concurrent mode.	-	-	X	-	-
	Ble_Thread_Dyn_SED	How to use BLE application and Thread application (acting as sleep end device) in dynamic concurrent mode.	-	-	X	-	-
	Ble_Thread_Dyn_SED_FreeRTOS	How to use BLE application and Thread application (acting as sleep end device) in dynamic concurrent mode.	-	-	New	-	-
	Ble_Thread_Static	How to use BLE application and Thread application in static concurrent mode.	-	-	X	-	-
BLE_Zigbee	BLE_Zigbee_Dyn	How to use BLE application and Zigbee application (acting as router) in dynamic concurrent mode.	-	-	X	-	-
	BLE_Zigbee_Dyn_SED	How to use BLE application and Zigbee application (acting as sleep end device) in dynamic concurrent mode.	-	-	X	-	-
	BLE_Zigbee_Static	How to use BLE application and Zigbee application in static concurrent mode.	-	-	X	-	-
CKS	CKS_Crypt	How to use CKS feature to store AES crypto keys in secure area.	-	-	X	-	-
Demontrations	Audio_BVLINKWB	This demonstration firmware is based on STM32Cube Function pack for STM32wb MCU featuring full-duplex audio streaming over Bluetooth 5.0 using Opus codec.	X	-	-	-	-
FreeRTOS	FreeRTOS_Mail	How to use mail queues with CMSIS RTOS API.	-	-	CubeMx	-	-
	FreeRTOS_Mutexes	How to use mutexes with CMSIS RTOS API.	-	-	CubeMx	CubeMx	-
	FreeRTOS_Queues	How to use message queues with CMSIS RTOS API.	-	-	CubeMx	-	-
	FreeRTOS_Semaphore	How to use semaphores with CMSIS RTOS API.	-	-	CubeMx	-	-

	FreeRTOS_SemaphoreFromISR	How to use semaphore from ISR with CMSIS RTOS API.	-	-	CubeMx	-	-
	FreeRTOS_Signal	How to perform thread signaling using CMSIS RTOS API.	-	-	CubeMx	-	-
	FreeRTOS_SignalFromISR	This application shows the usage of CMSIS-OS Signal API from ISR context.	-	-	CubeMx	-	-
	FreeRTOS_ThreadCreation	How to implement thread creation using CMSIS RTOS API.	-	-	CubeMx	-	-
	FreeRTOS_Timers	How to use timers of CMSIS RTOS API.	-	-	CubeMx	-	-
Mac_802_15_4	Mac_802_15_4_Coordinator	How to use MAC 802.15.4 Association and Data exchange.	-	-	New	-	-
	Mac_802_15_4_FFD	How to use MAC 802.15.4 Association and Data exchange.	-	-	X	-	-
	Mac_802_15_4_LPM_Periodic_Transmit	How to use MAC 802.15.4 data transmission with STOP1 low power mode enabled.	-	-	X	-	-
	Mac_802_15_4_Node	How to use MAC 802.15.4 Association and Data exchange.	-	-	New	-	-
	Mac_802_15_4_RFD	How to use MAC 802.15.4 Association and Data exchange.	-	-	X	-	-
Phy_802_15_4	Phy_802_15_4_Cli	How to create a "PHY_802.15.4 command line interface" application on STM32WB55xx boards using terminals.	-	-	X	-	-
Thread	Thread_Cli_Cmd	How to control the Thread stack via Cli commands.	-	X	CubeMx	-	-
	Thread_Coap_DataTransfer	How to transfer large blocks of data through the CoAP messaging protocol.	-	X	CubeMx	-	-
	Thread_Coap_Generic	How to build Thread application based on Coap messages.	X	X	CubeMx	-	-
	Thread_Coap_Generic_Ota	How to build Thread application based on Coap messages (OTA mode).	-	-	X	-	-
	Thread_Coap_Generic_ThreadX	How to build Thread application based on Coap messages. (using ThreadX) This application requires two STM32WB55xx boards.	-	-	X	-	-
	Thread_Coap_MultiBoard	How to use Coap for sending message to multiple boards.	-	-	CubeMx	-	-
	Thread_Coap_Secure	How to build Thread application based on Coap Secure messages.	-	-	X	-	-
	Thread_Commissioning	How to use Thread commissioning process.	-	-	CubeMx	-	-
	Thread_FTD_Coap_Multicast	How to exchange multicast Coap messages.	-	X	CubeMx	-	-
	Thread_NVM	How to configure NVM for Thread applications.	-	-	X	-	-
	Thread_Ota	How to update Over The Air (OTA) FW application and Copro Wireless binary using Thread (Client side).	-	-	X	-	-
	Thread_Ota_Server	How to update Over The Air (OTA) FW application and Copro Wireless binary using Thread (Server side).	-	-	X	-	-
	Thread_RCP	This application is used to demonstrate the OpenThread Border router feature using an STM32WB device.	-	-	X	-	-
	Thread_RCP_Cli_Cmd	How to control the Thread stack via Cli commands and trigger an automatic commissioning joiner sequence from a press button.	-	-	X	-	-
	Thread_SED_Coap_FreeRTOS	How to exchange a Coap message using the Thread protocol (using FreeRTOS porting).	-	-	CubeMx	-	-
	Thread_SED_Coap_Multicast	How to exchange a Coap message using the Thread protocol.	-	X	CubeMx	-	-
	Thread_Udp	How to transfer data using UDP.	-	-	X	-	-
TouchSensing	TouchSensing_1touchKey	Use of the STMTouch driver with 1 touchkey sensor.	-	-	X	-	-
	TouchSensing_1touchkey	Use of the STMTouch driver with 1 touchkey sensor.	CubeMx	-	-	-	-
USB_Device	CDC_Standalone	This application describes how to use USB device application based on the Device Communication Class (CDC) following the PSTN sub-protocol on the STM32WBxx devices.	-	-	CubeMx	-	-
	DFU_Standalone	Compliant implementation of the Device Firmware Upgrade (DFU).	-	CubeMx	CubeMx	-	-
	HID_Standalone	Use of the USB device application based on the Human Interface (HID).	-	CubeMx	CubeMx	-	-
Zigbee	Zigbee_APS_Coord	How to use the APS layer in an application acting as a Zigbee Coordinator within a centralized network.	-	-	X	-	-
	Zigbee_APS_Router	How to use the APS layer in an application acting as a Router within a centralized Zigbee network.	-	-	X	-	-
	Zigbee_Commissioning_Client_Coord	How to use the Commissioning cluster on a device acting as a Client with Coordinator role within a Centralized Zigbee network.	-	-	X	-	-
	Zigbee_Commissioning_Server_Router	How to use the Commissioning cluster on a device acting as a Server with Router role within a Centralized Zigbee network.	-	-	X	-	-
	Zigbee_DevTemp_Client_Router	How to use the Device Temperature cluster on a device acting as a Client with Router role within a Centralized Zigbee network.	-	-	X	-	-
	Zigbee_DevTemp_Server_Coord	How to use the Device Temperature cluster on a device acting as a Server with Coordinator role on a Centralized Zigbee network.	-	-	X	-	-
	Zigbee_Diagnostic_Client_Router	How to use the Diagnostic cluster on a device acting as a Client with Router role within a Centralized Zigbee network.	-	-	X	-	-
	Zigbee_Diagnostic_Server_Coord	How to use the Diagnostic cluster as a device acting as a Server with Coordinator role within a Centralized Zigbee network.	-	-	X	-	-
	Zigbee_DoorLock_Client_Router	How to use the Door Lock cluster on a device acting as a Client with Router role within a Centralized Zigbee network.	-	-	X	-	-
	Zigbee_DoorLock_Server_Coord	ow to use the Door Lock cluster on a device acting as a Server with Coordinator role within a Centralized Zigbee network.	-	-	X	-	-
	Zigbee_Find_Bind_Coord	How to use the Finding and Binding feature on a device acting as a Server with Coordinator role within a Centralized Zigbee network.	-	-	X	-	-
	Zigbee_Find_Bind_IAS_Router2	How to use the Finding and Binding feature on a device using IAS cluster, acting as a Client with Router role within a Centralized Zigbee network.	-	-	X	-	-
	Zigbee_Find_Bind_OnOff_Router1	How to use the Finding and Binding feature on a device using OnOff cluster, acting as a Client with Router role within a Centralized Zigbee network.	-	-	X	-	-
	Zigbee_IAS_WD_Client_Router	How to use the IAS WD (Intruder Alarm System Warning Device) cluster on a device acting as a Client with Router role within a Centralized Zigbee network.	-	-	X	-	-
	Zigbee_IAS_WD_Server_Coord	How to use the IAS WD (Intruder Alarm System Warning Device) cluster on a device acting as a Server with Coordinator role within a Centralized Zigbee network.	-	-	X	-	-
	Zigbee_MeterId_Client_Router	How to use the Meter Identification cluster on a device acting as a client with Router role within a Centralized Zigbee network.	-	-	CubeMx	-	-
	Zigbee_MeterId_Server_Coord	How to use the Meter Identification cluster on a device acting as a Server with Coordinator role within a Centralized Zigbee network.	-	X	CubeMx	-	-

	Zigbee_OTA_Client_Router	How to use the OTA cluster on multiple devices acting as a Client with Router role receiving and updating on a parallel way, the same OTA image from the ZC.	-	-	X	-	-
	Zigbee_OTA_Server_Coord	How to use the OTA cluster on one OR multiple devices. Router(s) is/are receiving and updating on a parallel way, the same OTA New image sent by the ZC.	-	-	X	-	-
	Zigbee_OnOff_ChannelsAgility_SED	How to use the OnOff cluster on a device on a Sleepy End Device (SED) acting as a Client within a Centralized Zigbee network.	-	-	X	-	-
	Zigbee_OnOff_ChannelsAgility_ZC	How to use the OnOff cluster on a device acting acting as a Server with Coordinator role within a Centralized Zigbee network.	-	-	X	-	-
	Zigbee_OnOff_ChannelsAgility_ZR	How to use the OnOff cluster on a device acting as a Client with Router role within a Centralized Zigbee network.	-	-	X	-	-
	Zigbee_OnOff_Client_Distrib	How to use the OnOff cluster on a device acting as a Client within a distributed Zigbee network.	-	-	CubeMx	-	-
	Zigbee_OnOff_Client_Router	How to use the OnOff cluster on a device acting as a Client with Router role within a Centralized Zigbee network.	X	X	CubeMx	-	-
	Zigbee_OnOff_Client_Router_FreeRTOS	How to use the OnOff cluster on a device acting as a Client with Router role within a Centralized Zigbee network using FreeRTOS.	-	-	X	-	-
	Zigbee_OnOff_Client_Router_Ota	How to use an updated OnOff cluster Zigbee application previously downloaded via OTA.	-	-	X	-	-
	Zigbee_OnOff_Client_Router_ThreadX	How to use the OnOff cluster on a device acting as a Client with Router role within a Centralized Zigbee network.	-	-	X	-	-
	Zigbee_OnOff_Client_SED	How to use the OnOff cluster on a Sleepy End Device acting as a Client within a Centralized Zigbee network.	-	-	CubeMx	-	-
	Zigbee_OnOff_Coord_NVM	How to use the OnOff cluster with persistent data activated on a device acting as Coordinator within a Centralized Zigbee network.	-	-	X	-	-
	Zigbee_OnOff_Router_NVM	How to use the OnOff cluster with persistent data activated on a device acting as Router within a Centralized Zigbee network.	-	-	X	-	-
	Zigbee_OnOff_Server_Coord	How to use the OnOff cluster on a device acting as a Server with Coordinator role within a Centralized Zigbee network.	X	X	CubeMx	-	-
	Zigbee_OnOff_Server_Coord_FreeRTOS	How to use the OnOff cluster on a device acting as a Server with Coordinator role within a Centralized Zigbee network using FreeRTOS.	-	-	X	-	-
	Zigbee_OnOff_Server_Coord_ThreadX	How to use the OnOff cluster on a device acting as a Server with Coordinator role within a Centralized Zigbee network using ThreadX.	-	-	X	-	-
	Zigbee_OnOff_Server_Distrib	How to use the OnOff cluster on a device acting as a Server within a distributed Zigbee network.	-	-	CubeMx	-	-
	Zigbee_PollControl_Client_Coord	How to use the Poll Control cluster on a device acting as a client with Coordinator role within a Centralized Zigbee network.	-	-	X	-	-
	Zigbee_PollControl_Server_SED	How to use the Poll Control cluster on a Sleepy End Device (SED) acting as a Server within a Centralized Zigbee network.	-	-	X	-	-
	Zigbee_PowerProfile_Client_Coord	How to use the Power Profile cluster on a device acting as a Client with Coordinator role within a Centralized Zigbee network.	-	-	X	-	-
	Zigbee_PowerProfile_Server_Router	How to use the Power Profile cluster on a device acting as a Server with Router role within a Centralized Zigbee network.	-	X	X	-	-
	Zigbee_PressMeas_Client_Router	How to use the Pressure Measurement cluster on a device acting as a client with Router role within a Centralized Zigbee network.	-	-	X	-	-
	Zigbee_PressMeas_Server_Coord	How to use the Pressure Measurement cluster on a device acting as a Server with Coordinator role within a Centralized Zigbee network.	-	-	X	-	-
	Zigbee_SE_Msg_Client_Coord	How to use the Smart Energy Messaging cluster on a device acting as a client with Coordinator role within a Centralized Zigbee network.	-	-	X	-	-
	Zigbee_SE_Msg_Server_Router	How to use the Smart Energy Messaging cluster on a device acting as a Server with Router role within a Centralized Zigbee network.	-	-	X	-	-
	Zigbee_TempMeas_Client_Router	How to use the Device Temperature cluster on a device acting as a Client with Router role within a Centralized Zigbee network.	X	-	-	-	-
	Zigbee_TempMeas_Server_Coord	How to use the Device Temperature cluster on a device acting as a Server with Coordinator role on a Centralized Zigbee network.	X	-	-	-	-
	Zigbee_custom_ls_Client_Router	How to use the Custom long string cluster on a device acting as a Client with Router role within a Centralized Zigbee network.	-	-	X	-	-
	Zigbee_custom_ls_Server_Coord	How to use the Custom long string cluster on a device acting as a Server with Coordinator role within a Centralized Zigbee network.	-	-	X	-	-
		Total number of applications: 176	14	19	127	14	2
		Total number of projects: 472	24	20	333	90	5