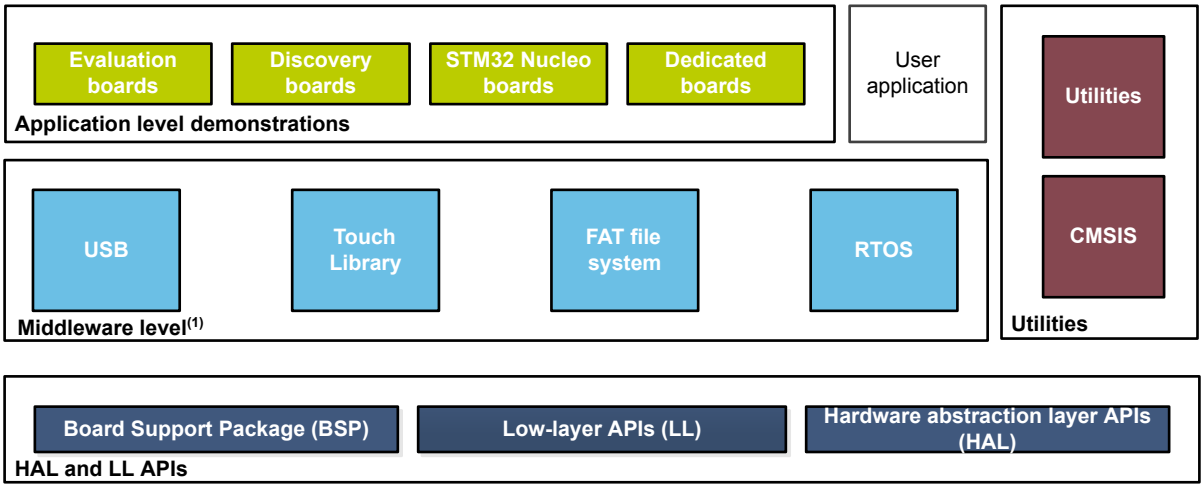
STM32Cube MCU Package examples for STM32L0 Series

Introduction

The STM32CubeL0 MCU Package comes 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. STM32CubeL0 firmware components



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

1 Reference documents

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

- Latest release of STM32CubeL0 MCU Package for the STM32L0 32-bit Arm® Cortex®-M0+ microcontrollers
- *Getting started with STM32CubeL0 for STM32L0 Series* (UM1754)
- *Description of STM32L0xx HAL drivers* (UM1749)
- *STM32Cube USB Device library* (UM1734)
- *Developing applications on STM32Cube with FatFS* (UM1721)
- *Developing applications on STM32Cube with RTOS* (UM1722)

The microcontrollers of the STM32L0 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 STM32CubeL0 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 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 drivers and middleware components not used). They offer an optimum implementation of typical use cases of the peripheral features and configuration sequences. The examples are organized per peripheral (one folder for each peripheral, e.g. TIM) and run exclusively on Nucleo boards.

- **Examples_MIX**

  These examples use only HAL, BSP and LL drivers (middleware components not used). They aim at demonstrating how to use both HAL and LL APIs in the same application to combine the advantages of both APIs:

  – HAL offers high-level function-oriented APIs with high portability level by hiding product/IPs complexity for end users.

  – LL provides low-level APIs at register level with better optimization.

  The examples are organized per peripheral (one folder for each peripheral, e.g. TIM) and run exclusively on Nucleo boards.

- **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 project**

  The template project is provided to allow the user to quickly build a firmware application using HAL and BSP drivers on a given board.

- **Template_LL project**

  The template LL project is provided to allow the user to quickly build a firmware application using LL drivers on a given board.

The examples are located under *STM32Cube_FW_L0_VX.Y.Z\Projects*\. They all have the same structure:

- *Inc* folder, containing all header files.
- *Src* folder, containing the sources code.
- *EWARM, \MDK-ARM and \SW4STM32* folders, containing the preconfigured project for each toolchain.
- readme.txt file, describing the example behavior and the environment required to run the example.

To run the example, proceed as follows:

1. Open the example using your preferred toolchain.
2. Rebuild all files and load the image into target memory.
3. Run the example by following the readme.txt instructions.

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

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

Section 2 contains the list of examples provided with STM32CubeL0 MCU Package.

Table 1. STM32CubeL0 firmware examples

Level	Module Name	Project Name	Description	NUCLEO-L073RZ	NUCLEO-L031K6	NUCLEO-L053R8	STM32L073Z-EVAL	NUCLEO-L011K4	NUCLEO-L010RB	32L0538DISCOVERY
Templates	-	Starter project	This project provides a reference template that can be used to build any firmware application.	X	X	X	X	X	New	X
	Total number of templates: 7			**1**	**1**	**1**	**1**	**1**	**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	New	X
	Total number of templates_ll: 7			1	1	1	1	1	1	1
Examples	-	BSP	This example provides a description of how to use the different BSP drivers.	-	-	X	X	-	-	-
	ADC	ADC_AnalogWatchdog	How to use the ADC peripheral to perform conversions with an analog watchdog and out-of-window interrupts enabled.	X	-	-	X	-	-	-
		ADC_DMA_Transfer	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	New	X
		ADC_LowPower	How to use the ADC peripheral to perform conversions with ADC low-power modes: auto-wait and auto-power off.	X	X	X	X	X	-	X
		ADC_OverSampler	How to configure and use the ADC to convert an external analog input, combined with the oversampling feature, to increase resolution through the HAL API.	-	X	X	-	-	-	X
		ADC_RegularConversion_Interrupt	How to use the ADC in Interrupt mode to convert data through the HAL API.	X	X	X	X	-	-	X
		ADC_RegularConversion_Polling	How to use the ADC in Polling mode to convert data through the HAL API.	X	X	X	X	-	-	-
		ADC_Sequencer	How to use the ADC peripheral with a sequencer to convert several channels.	-	-	X	-	-	-	-
	AES	AES_DMA	How to use the AES peripheral to encrypt and decrypt data using AES algorithm in ECB chaining mode.	-	-	X	-	-	-	-
		AES_Modes	How to configure the AES hardware accelerator to encrypt then decrypt text in ECB, CBC and CTR mode.	-	-	X	-	-	-	-

Level	Module Name	Project Name	Description	NUCLEO-L073RZ	NUCLEO-L031K6	NUCLEO-L053R8	STM32L073Z-EVAL	NUCLEO-L011K4	NUCLEO-L010RB	32L0538DISCOVERY
Examples	COMP	COMP_AnalogWatchdog	How to use a pair of comparator peripherals to compare, , in Interrupt mode, 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/4).	X	-	X	X	-	-	X
		COMP_Interrupt	How to use a comparator peripheral to compare, in Interrupt mode, a voltage level applied on a GPIO pin to the internal voltage reference (VREFINT).	-	-	X	-	-	-	-
		COMP_PWMSignalControl	How to configure a comparator peripheral to automatically hold the TIMER PWM output in the safe state (low level) as soon as the comparator output is set to a high level.	X	X	X	X	-	-	X
		COMP_PulseWidthMeasurement	How to configure a comparator peripheral to measure a pulse width. This method (measuring signal pulses using a comparator) is useful when an external signal does not respect the VIL and VIH levels.	X	-	X	X	-	-	X
	CRC	CRC_Data_Reversing_16bit_CRC	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 8-bit data (bytes). 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	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	New	X
		CRC_bytes_stream_7bit_CRC	How to configure the CRC using the HAL API. The CRC (cyclic redundancy check) calculation unit computes 7-bit CRC codes derived from buffers of 8-bit data (bytes). The user-defined generating polynomial is manually set to 0x65, that is, X^7 + X^6 + X^5 + X^2 + 1, as used in the Train Communication Network, IEC 60870-5[17].	X	-	-	-	-	-	-
	CRYP	CRYP_AESModes	How to use the CRYP peripheral to encrypt and decrypt data using AES in chaining modes (ECB, CBC, CTR).	X	X	-	-	X	-	-
		CRYP_DMA	How to use the CRYP peripheral to encrypt and decrypt data using the AES-128 algorithm with ECB chaining mode in DMA mode.	X	X	-	-	X	-	-
	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.	X	-	X	X	-	-	X
		CORTEXM_ModePrivilege	How to modify the Thread mode privilege access and stack. The Thread mode is entered on reset or when returning from an exception.	-	-	X	-	-	-	X
		CORTEXM_SysTick	How to use the default SysTick configuration with a 1 ms timebase to toggle LEDs.	X	-	X	X	-	New	X
	DAC	DAC_SignalsGeneration	How to use the DAC peripheral to generate several signals using the DMA controller.	X	-	X	X	-	-	X
		DAC_SimpleConversion	How to use the DAC peripheral to do a simple conversion.	X	-	X	X	-	-	X
	DMA	DMA_FLASHToRAM	How to use a DMA to transfer a word data buffer from Flash memory to embedded SRAM through the HAL API.	X	X	X	X	X	-	X
		DMA_RAMToDAC	How to use a DMA channel to transfer data buffer from RAM to DAC.	X	-	X	-	-	-	X

Level	Module Name	Project Name	Description	NUCLEO-L073RZ	NUCLEO-L031K6	NUCLEO-L053R8	STM32L073Z-EVAL	NUCLEO-L011K4	NUCLEO-L010RB	32L0538DISCOVERY
Examples	FIREWALL	FIREWALL_VolatileData_ Executable	How to use the Firewall peripheral to protect a volatile data segment and to define it as executable.	-	-	-	X	-	-	-
		FIREWALL_VolatileData_Shared	How to use the Firewall peripheral to protect a code segment as well as volatile and non-volatile data segments.	-	-	-	X	-	-	-
	FLASH	FLASH_DualBoot	Guide through the configuration steps to program internal Flash memory bank 1 and bank 2, and to swap between both of them by mean of the FLASH HAL API.	-	-	-	X	-	-	-
		FLASH_EraseProgram	How to configure and use the FLASH HAL API to erase and program the internal Flash memory.	X	X	X	X	X	-	X
		FLASH_WriteProtection	How to configure and use the FLASH HAL API to enable and disable the write protection on the internal Flash memory.	X	X	X	X	X	-	X
	GPIO	GPIO_EXTI	How to configure external interrupt lines.	X	-	X	X	-	-	X
		GPIO_IOToggle	How to configure and use GPIOs through the HAL API.	X	X	X	X	X	New	-
		GPIO_IOToggle_MaxFrequency	How to configure and use GPIOs.	X	X	X	X	X	-	X
		GPIO_IOToggle_VariableFreq	How to configure and use GPIOs to toggle the on-board user LEDs at different frequencies. This example is based on the STM32L0xx LL API. The peripheral is initialized with LL initialization function to demonstrate LL init usage.	-	X	-	-	-	-	-
	HAL	HAL_TimeBase	How to customize the HAL driver using a general-purpose timer as main timebase source , instead of the SysTick.	-	-	X	-	-	-	X
	I2C	I2C_TwoBoards_AdvComIT	How to handle I2C data buffer transmission/reception between two boards, using an interrupt and two buffers.	X	X	X	X	X	-	X
		I2C_TwoBoards_ComDMA	How to handle I2C data buffer transmission/reception between two boards, via DMA.	X	X	X	X	X	-	X
		I2C_TwoBoards_ComIT	How to handle I2C data buffer transmission/reception between two boards, using an interrupt and a common buffer.	-	X	X	-	X	-	X
		I2C_TwoBoards_ComPolling	How to handle I2C data buffer transmission/reception between two boards, in Polling mode.	-	X	X	-	X	-	-
		I2C_TwoBoards_RestartAdvComIT	How to perform multiple I2C data buffer transmission/reception between two boards, in Interrupt mode and with a restart condition.	X	-	-	-	-	-	-
		I2C_TwoBoards_RestartComIT	How to handle single I2C data buffer transmission/reception between two boards, in Interrupt mode and with a restart condition.	X	-	-	-	-	-	-
		I2C_WakeUpFromStop	How to handle I2C data buffer transmission/reception between two boards, using an interrupt when the device is in Stop mode.	X	X	-	X	X	-	-

Level	Module Name	Project Name	Description	NUCLEO-L073RZ	NUCLEO-L031K6	NUCLEO-L053R8	STM32L073Z-EVAL	NUCLEO-L011K4	NUCLEO-L010RB	32L0538DISCOVERY
Examples	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.	X	X	X	X	X	New	X
		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.	X	-	X	X	-	-	X
	LCD	LCD_Blink_Frequency	How to use the embedded LCD glass controller and to set the LCD blinking mode and blinking frequency.	-	-	X	X	-	-	-
		LCD_SegmentsDrive	How to use the embedded LCD glass controller to drive the on-board LCD glass by Pacific Display Devices.	-	-	X	-	-	-	-
	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.	-	-	X	X	-	-	X
		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.	-	-	X	X	-	-	X
		LPTIM_PulseCounter	How to configure and use, through the LPTIM HAL API, the LPTIM peripheral to count pulses.	X	X	X	X	X	-	X
		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.	X	X	X	X	-	New	X
	PWR	PWR_LPRUN	How to enter and exit the Low-power run mode.	X	X	X	X	X	-	X
		PWR_LPSLEEP	How to enter the Low-power sleep mode and wake up from this mode by using an interrupt.	X	X	X	X	X	-	X
		PWR_PVD	How to configure the programmable voltage detector by using an external interrupt line. External DC supply must be used to supply V_{DD}.	X	X	X	X	X	-	X
		PWR_SLEEP	How to enter Sleep mode and wake up from this mode by using an interrupt.	X	X	X	X	X	-	X
		PWR_STANDBY	How to enter Standby mode and wake up from this mode by using an external reset or the WKUP pin.	X	X	X	X	X	New	X
		PWR_STANDBY_RTC	How to enter Standby mode and wake up from this mode by using an external reset or the RTC wakeup timer.	X	X	X	X	X	-	X
		PWR_STOP	How to enter Stop mode and wake up from this mode by using the RTC wakeup timer event or an interrupt.	X	X	X	X	X	-	X
		PWR_STOP_RTC	How to enter Stop mode and wake up from this mode by using the RTC wakeup timer event connected to an interrupt.	X	X	X	X	X	-	X

Level	Module Name	Project Name	Description	NUCLEO-L073RZ	NUCLEO-L031K6	NUCLEO-L053R8	STM32L073Z-EVAL	NUCLEO-L011K4	NUCLEO-L010RB	32L0538DISCOVERY
Examples	RCC	RCC_CRS_Synchronization_IT	Configuration of the clock recovery service (CRS) in Interrupt mode, using the RCC HAL API.	-	-	-	X	-	-	-
		RCC_CRS_Synchronization_ Polling	Configuration of the clock recovery service (CRS) in Polling mode, using the RCC HAL API.	-	-	-	X	-	-	-
		RCC_ClockConfig	Configuration of the system clock (SYSCLK) and modification of the clock settings in Run mode, using the RCC HAL API.	-	-	X	X	-	-	-
		RCC_LSIConfig	Enabling/disabling of the low-speed internal (LSI) RC oscillator (about 40 KHz) at runtime, using the RCC HAL API.	X	X	-	X	X	New	-
	RNG	RNG_MultiRNG	Configuration of the RNG using the HAL API. This example uses the RNG to generate 32-bit long random numbers.	X	-	X	X	-	-	X
	RTC	RTC_Alarm	Configuration and generation of an RTC alarm using the RTC HAL API.	X	X	X	X	X	New	X
		RTC_Calendar	Configuration of the calendar using the RTC HAL API.	-	X	-	X	X	-	X
		RTC_LSI	Use of the LSI clock source autocalibration to get a precise RTC clock.	X	X	X	X	X	-	X
		RTC_LowPower_STANDBY	How to enter Standby mode and wake up from this mode using the RTC alarm event.	-	-	-	X	-	-	-
		RTC_Tamper	Configuration of the RTC HAL API to write/read data to/from RTC Backup registers.	X	X	X	X	X	-	X
		RTC_TimeStamp	Configuration of the RTC HAL API to demonstrate the timestamp feature.	X	X	X	X	X	-	X
	SMBUS	SMBUS_TSENSOR	SMBUS data buffer transmission/reception using an interrupt. The STM32 microcontroller communicates with an SMBUS temperature sensor.	-	-	-	X	-	-	-
	SPI	SPI_FullDuplex_AdvCom	Configuration of the HAL SPI API to transmit/receive a data buffer in Interrupt mode and in an advanced communication mode: the master board always sends the command to the slave before performing any transmission; the slave board sends back an acknowledgement before proceeding.	-	-	X	-	-	-	-
		SPI_FullDuplex_ComDMA	Data buffer transmission/reception between two boards via SPI using DMA.	X	X	X	X	X	-	X
		SPI_FullDuplex_ComIT	Data buffer transmission/reception between two boards via SPI using Interrupt mode.	X	X	X	X	X	-	X
		SPI_FullDuplex_ComPolling	Data buffer transmission/reception between two boards via SPI using Polling mode.	X	X	X	X	X	-	-

Level	Module Name	Project Name	Description	NUCLEO-L073RZ	NUCLEO-L031K6	NUCLEO-L053R8	STM32L073Z-EVAL	NUCLEO-L011K4	NUCLEO-L010RB	32L0538DISCOVERY
Examples	TIM	TIM_DMA	Use of 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
		TIM_DMABurst	Update of the TIMER channel 1 period and duty cycle using the TIMER DMA burst feature.	X	X	X	X	X	-	X
		TIM_ExtTriggerSynchro	Synchronization of TIM peripherals in Cascade mode with an external trigger.	-	-	X	-	-	-	-
		TIM_InputCapture	Use of the TIM peripheral to measure an external signal frequency.	X	X	X	X	X	-	X
		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).	X	X	X	X	X	-	X
		TIM_OCInactive	Configuration of the TIM peripheral in Output Compare Inactive mode with the corresponding Interrupt requests for each channel.	X	X	X	X	X	-	-
		TIM_OCToggle	Configuration of the TIM peripheral to generate four different signals at four different frequencies.	X	X	X	X	X	-	X
		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	X	X	X	X	-	-
		TIM_PWMInput	Use of the TIM peripheral to measure an external signal frequency and duty cycle.	X	-	X	-	-	-	-
		TIM_PWMOutput	Configuration of the TIM peripheral in PWM (pulse width modulation) mode.	X	X	X	X	X	-	X
		TIM_TimeBase	Configuration of the TIM peripheral to generate a timebase of one second with the corresponding interrupt request.	X	X	X	X	X	New	X

Level	Module Name	Project Name	Description	NUCLEO-L073RZ	NUCLEO-L031K6	NUCLEO-L053R8	STM32L073Z-EVAL	NUCLEO-L011K4	NUCLEO-L010RB	32L0538DISCOVERY
Examples	TSC	TSC_BasicAcquisition_Interrupt	Use of he TSC to perform continuous acquisitions of two channels in Interrupt mode.	-	-	-	X	-	-	-
		TSC_BasicAcquisition_Polling	Use of the TSC to perform continuous acquisitions of one channel in Polling mode.	-	-	-	X	-	-	-
	UART	LPUART_TwoBoards_ComIT	LPUART transmission (transmit/receive) in Interrupt mode between two boards.	-	-	-	-	-	-	X
		LPUART_WakeUpFromStop	Configuration of an LPUART to wake up the MCU from Stop mode when a given stimulus is received.	-	-	-	-	-	-	X
		UART_HyperTerminal_DMA	UART transmission (transmit/receive) in DMA mode between a board and an HyperTerminal PC application.	-	X	-	X	X	-	-
		UART_LowPower_HyperTerminal_DMA	LPUART transmission (transmit/receive) in DMA mode between a board and an HyperTerminal PC application.	-	-	-	X	-	-	-
		UART_Printf	Re-routing of the C library printf function to the UART.	-	X	-	X	X	-	-
		UART_TwoBoards_ComDMA	UART transmission (transmit/receive) in DMA mode between two boards.	X	X	X	X	X	-	X
		UART_TwoBoards_ComIT	UART transmission (transmit/receive) in Interrupt mode between two boards.	X	X	X	X	X	-	X
		UART_TwoBoards_ComPolling	UART transmission (transmit/receive) in Polling mode between two boards.	X	X	X	X	X	-	-
		UART_WakeUpFromStop	Configuration of an UART to wake up the MCU from Stop mode when a given stimulus is received.	-	X	-	X	X	-	-
	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.	X	X	X	X	X	New	X
	Total number of examples: 386			**64**	**56**	**72**	**75**	**50**	**11**	**57**

Level	Module Name	Project Name	Description	NUCLEO-L073RZ	NUCLEO-L031K6	NUCLEO-L053R8	STM32L073Z-EVAL	NUCLEO-L011K4	NUCLEO-L010RB	32L0538DISCOVERY
Examples_LL	ADC	ADC_AnalogWatchdog	How to use an ADC peripheral with an ADC analog watchdog to monitor a channel and detect when the corresponding conversion data is outside the window thresholds.	X	-	-	-	-	-	-
		ADC_ContinuousConversion_TriggerSW	How to use an ADC peripheral to perform continuous ADC conversions on a channel, from a software start.	X	-	-	-	-	-	-
		ADC_ContinuousConversion_TriggerSW_Init	How to use an ADC peripheral to perform continuous ADC conversions on a channel, from a software start.	X	-	-	-	-	-	-
		ADC_ContinuousConversion_TriggerSW_LowPower	How to use an ADC peripheral with ADC low-power features.	X	-	-	-	-	-	-
		ADC_MultiChannelSingleConversion	How to use an ADC peripheral to convert several channels. ADC conversions are performed successively in a scan sequence.	X	-	-	-	-	-	-
		ADC_Oversampling	How to use an ADC peripheral with ADC oversampling.	X	-	-	-	-	-	-
		ADC_SingleConversion_TriggerSW	How to use an ADC peripheral to perform a single ADC conversion on a given channel at each software start. This example uses the polling programming model (for interrupt or DMA programming models, please refer to other examples).	X	-	-	-	-	-	-
		ADC_SingleConversion_TriggerSW_DMA	How to use an ADC peripheral to perform a single ADC conversion on a given channel at each software start. This example uses the DMA programming model (for polling or interrupt programming models, refer to other examples).	X	-	-	-	-	-	-
		ADC_SingleConversion_TriggerSW_IT	How to use an ADC peripheral to perform a single ADC conversion on a given channel at each software start. This example uses the interrupt programming model (for polling or DMA programming models, please refer to other examples).	X	-	-	-	-	-	-
		ADC_SingleConversion_TriggerTimer_DMA	How to use an ADC peripheral to perform a single ADC conversion on a given channel at each trigger event from a timer. Converted data is indefinitely transferred by DMA into a table (circular mode).	X	-	-	-	-	-	-
		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_CompareWithInternalReference_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 STM32L0xx COMP LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	X	-	-	-	-	-	-
		COMP_CompareWithInternalReference_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 STM32L0xx COMP LL API. The peripheral initialization uses the LL initialization function to demonstrate LL init usage.	X	-	-	-	-	-	-
	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.	X	-	-	-	-	-	-

Level	Module Name	Project Name	Description	NUCLEO-L073RZ	NUCLEO-L031K6	NUCLEO-L053R8	STM32L073Z-EVAL	NUCLEO-L011K4	NUCLEO-L010RB	32L0538DISCOVERY
Examples_LL	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).	X	-	-	-	-	-	-
		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).	X	-	-	-	-	-	-
	CRS	CRS_Synchronization_IT	How to configure the clock recovery service in Interrupt mode through the STM32L0xx CRS LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	X	-	-	-	-	-	-
		CRS_Synchronization_Polling	How to configure the clock recovery service in Polling mode through the STM32L0xx CRS LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	X	-	-	-	-	-	-
	DAC	DAC_GenerateConstantSignal_ TriggerSW	How to use the DAC peripheral to generate a constant voltage signal. This example is based on the STM32L0xx DAC LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	X	-	-	-	-	-	-
		DAC_GenerateWaveform_ TriggerHW	How to use the DAC peripheral to generate a voltage waveform from a digital data stream transfered by DMA. This example is based on the STM32L0xx DAC LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	X	-	-	-	-	-	-
		DAC_GenerateWaveform_ TriggerHW_Init	How to use the DAC peripheral to generate a voltage waveform from a digital data stream transfered by DMA. This example is based on the STM32L0xx DAC LL API. The peripheral initialization uses LL initialization functions to demonstrate LL init usage.	X	-	-	-	-	-	-
	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.	X	-	-	-	-	-	-

Level	Module Name	Project Name	Description	NUCLEO-L073RZ	NUCLEO-L031K6	NUCLEO-L053R8	STM32L073Z-EVAL	NUCLEO-L011K4	NUCLEO-L010RB	32L0538DISCOVERY
Examples_LL	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 STM32L0xx LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	X	-	-	-	-	-	-
		EXTI_ToggleLedOnIT_Init	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 STM32L0xx LL API. The peripheral initialization uses LL initialization functions to demonstrate LL init usage.	X	-	-	-	-	-	-
	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 STM32L0xx 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 STM32L0xx LL API. The peripheral is initialized with LL initialization function to demonstrate LL init usage.	X	-	-	-	-	-	-
	I2C	I2C_OneBoard_Adv Communication_DMAAndIT	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.	X	-	-	-	-	-	-
		I2C_OneBoard_Communication_ DMAAndIT	How to transmit data bytes from an I2C master device using DMA mode to an I2C slave device in Interrupt mode. The peripheral is initialized with LL unitary service functions to optimize for performance and size.	X	-	-	-	-	-	-
		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.	X	-	-	-	-	-	-
		I2C_OneBoard_Communication_ PollingAndIT	How to transmit data bytes from an I2C master device opetating in Polling mode to an I2C slave device operating in Interrupt mode. The peripheral is initialized with LL unitary service functions to optimize for performance and size.	X	-	-	-	-	-	-
		I2C_TwoBoards_MasterRx_ SlaveTx_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_TwoBoards_MasterTx_ SlaveRx	How to transmit data bytes from an I2C master device operating in Polling mode to an I2C slave device operating in Interrupt mode. The peripheral is initialized with LL unitary service functions to optimize for performance and size.	X	-	-	-	-	-	-
		I2C_TwoBoards_MasterTx_ SlaveRx_DMA	How to transmit data bytes from an I2C master device to an I2C slave device. Both devices operate in DMA mode. The peripheral is initialized with LL unitary service functions to optimize for performance and size.	X	-	-	-	-	-	-
		I2C_TwoBoards_WakeUpFromStop _IT	How to handle the reception of a data byte from an I2C slave device in Stop mode by an I2C master device, both operating in Interrupt mode. The peripheral is initialized with LL unitary service functions to optimize for performance and size.	X	-	-	-	-	-	-

Level	Module Name	Project Name	Description	NUCLEO-L073RZ	NUCLEO-L031K6	NUCLEO-L053R8	STM32L073Z-EVAL	NUCLEO-L011K4	NUCLEO-L010RB	32L0538DISCOVERY
Examples_LL	IWDG	IWDG_RefreshUntilUserEvent	How to configure the IWDG peripheral to ensure periodical counter update and generate an MCU IWDG reset when a user button is pressed. The peripheral is initialized with LL unitary service functions to optimize for performance and size.	X	-	-	-	-	-	-
	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 STM32L0xx 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 STM32L0xx LPTIM LL API. The peripheral is initialized with LL initialization function to demonstrate LL init usage.	X	-	-	-	-	-	-
	LPUART	LPUART_WakeUpFromStop	Configuration of GPIO and LPUART peripherals to allow characters received on LPUART_RX pin to wake up the MCU from low-power mode. This example is based on the LPUART LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	X	-	-	-	-	-	-
		LPUART_WakeUpFromStop_Init	Configuration of GPIO and LPUART peripherals to allow characters received on LPUART_RX pin to wake up the MCU from low-power mode. This example is based on the LPUART LL API. The peripheral initialization uses LL initialization function to demonstrate LL init usage.	X	-	-	-	-	-	-
	PWR	PWR_EnterStandbyMode	How to enter Standby mode and wake up from this mode by using an external reset or a wakeup interrupt.	X	-	-	-	-	-	-
		PWR_EnterStopMode	How to enter Stop mode.	X	-	-	-	-	-	-
		PWR_LPRunMode_SRAM1	How to execute code in Low-power run mode from SRAM1.	X	-	-	-	-	-	-
		PWR_OptimizedRunMode	How to increase/decrease frequency and V_{CORE} and how to enter/exit the Low-power run mode.	X	-	-	-	-	-	-
	RCC	RCC_OutputSystemClockOnMCO	Configuration of MCO pin (PA8) to output the system clock.	X	-	-	-	-	-	-
		RCC_UseHSEasSystemClock	Use of the RCC LL API to start the HSE and use it as system clock.	X	-	-	-	-	-	-
		RCC_UseHSI_PLLasSystemClock	Modification of the PLL parameters in run time.	X	-	-	-	-	-	-
	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).	X	-	-	-	-	-	-
		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).	X	-	-	-	-	-	-

Level	Module Name	Project Name	Description	NUCLEO-L073RZ	NUCLEO-L031K6	NUCLEO-L053R8	STM32L073Z-EVAL	NUCLEO-L011K4	NUCLEO-L010RB	32L0538DISCOVERY
Examples_LL	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.	X	-	-	-	-	-	-
		RTC_Calendar	Configuration of the LL API to set the RTC calendar. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	X	-	-	-	-	-	-
		RTC_ExitStandbyWithWakeUp Timer	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).	X	-	-	-	-	-	-
		RTC_ProgrammingTheWakeUp Timer	Configuration of the RTC to use the WUT. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	X	-	-	-	-	-	-
		RTC_Tamper	Configuration of the Tamper using the RTC LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	X	-	-	-	-	-	-
		RTC_TimeStamp	Configuration of the Timestamp using the RTC LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	X	-	-	-	-	-	-
	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 STM32L0xx 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 STM32L0xx SPI LL API. The peripheral initialization uses the LL initialization function to demonstrate LL init usage.	X	-	-	-	-	-	-
		SPI_OneBoard_HalfDuplex_IT	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 STM32L0xx SPI LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	X	-	-	-	-	-	-
		SPI_TwoBoards_FullDuplex_DMA	Data buffer transmission and reception via SPI using DMA mode. This example is based on the STM32L0xx SPI LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	X	-	-	-	-	-	-
		SPI_TwoBoards_FullDuplex_IT	Data buffer transmission and receptionvia SPI in Interrupt mode. This example is based on the STM32L0xx SPI LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	X	-	-	-	-	-	-

Level	Module Name	Project Name	Description	NUCLEO-L073RZ	NUCLEO-L031K6	NUCLEO-L053R8	STM32L073Z-EVAL	NUCLEO-L011K4	NUCLEO-L010RB	32L0538DISCOVERY
Examples_LL	TIM	TIM_DMA	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 STM32L0xx TIM LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	X	-	-	-	-	-	-
		TIM_InputCapture	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 STM32L0xx TIM LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	X	-	-	-	-	-	-
		TIM_OnePulse	Configuration of the TIM peripheral to generate a positive pulse in Output Compare mode with a length of t_{PULSE} and after a delay of t_{DELAY}. This example is based on the STM32L0xx TIM LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	X	-	-	-	-	-	-
		TIM_OutputCompare	Configuration of the TIM peripheral to generate an output waveform in different output compare modes. This example is based on the STM32L0xx TIM LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	X	-	-	-	-	-	-
		TIM_PWMOutput	Use of the TIM peripheral to generate a PWM output signal and update the PWM duty cycle. This example is based on the STM32L0xx TIM LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	X	-	-	-	-	-	-
		TIM_PWMOutput_Init	Use of the TIM peripheral to generate a PWM output signal and update the PWM duty cycle. This example is based on the STM32L0xx TIM LL API. The peripheral initialization uses LL initialization function to demonstrate LL init.	X	-	-	-	-	-	-
		TIM_TimeBase	Configuration of the TIM peripheral to generate a timebase. This example is based on the STM32L0xx TIM LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	X	-	-	-	-	-	-

Level	Module Name	Project Name	Description	NUCLEO-L073RZ	NUCLEO-L031K6	NUCLEO-L053R8	STM32L073Z-EVAL	NUCLEO-L011K4	NUCLEO-L010RB	32L0538DISCOVERY
Examples_LL	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	Configuration of GPIO and USART peripherals to continuously 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_Init	Configuration of 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	-	-	-	-	-	-
		USART_Communication_Tx	Configuration of GPIO and USART peripherals to send characters asynchronously to an HyperTerminal (PC) in Polling mode. If the transfer could not be complete within the allocated time, a timeout allows to exit from the sequence with timeout error. This example is based on STM32L0xx USART LL API.	X	-	-	-	-	-	-
		USART_Communication_TxRx_DMA	Configuration of GPIO and USART peripherals to send characters asynchronously to/from an HyperTerminal (PC) in DMA mode.	X	-	-	-	-	-	-
		USART_Communication_Tx_IT	Configuration of GPIO and USART peripheral to send characters asynchronously to HyperTerminal (PC) in Interrupt mode. This example is based on the STM32L0xx USART LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	X	-	-	-	-	-	-
		USART_HardwareFlowControl	Configuration of GPIO and USART peripheral to receive characters asynchronously from an HyperTerminal (PC) in Interrupt mode with the Hardware Flow Control feature enabled. This example is based on STM32L0xx USART LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	X	-	-	-	-	-	-
		USART_SyncCommunication_FullDuplex_DMA	Configuration of GPIO, USART, DMA and SPI peripherals to transmit bytes between a USART and an SPI (in slave mode) in DMA mode. This example is based on the STM32L0xx USART LL API. The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	X	-	-	-	-	-	-
		USART_SyncCommunication_FullDuplex_IT	Configuration of GPIO, USART, DMA and SPI peripherals to transmit bytes between a USART and an SPI (in slave mode) in Interrupt mode. This example is based on the STM32L0xx USART LL API (the SPI uses the DMA to receive/transmit characters sent from/received by the USART). The peripheral initialization uses LL unitary service functions for optimization purposes (performance and size).	X	-	-	-	-	-	-
		USART_WakeUpFromStop	Configuration of GPIO and USART peripherals to allow the characters received on USART RX pin to wake up the MCU from low-power mode.	X	-	-	-	-	-	-

Level	Module Name	Project Name	Description	NUCLEO-L073RZ	NUCLEO-L031K6	NUCLEO-L053R8	STM32L073Z-EVAL	NUCLEO-L011K4	NUCLEO-L010RB	32L0538DISCOVERY
Examples_LL	UTILS	UTILS_ConfigureSystemClock	How to use UTILS LL API to configure the system clock using PLL with HSI as source clock.	X	-	-	-	-	-	-
		UTILS_ReadDeviceInfo	How to Read UID, Device ID and Revision ID and save them into a global information buffer.	X	-	-	-	-	-	-
	WWDG	WWDG_RefreshUntilUserEvent	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).	X	-	-	-	-	-	-
	Total number of examples_ll: 82			**82**	**0**	**0**	**0**	**0**	**0**	**0**

Level	Module Name	Project Name	Description	NUCLEO-L073RZ	NUCLEO-L031K6	NUCLEO-L053R8	STM32L073Z-EVAL	NUCLEO-L011K4	NUCLEO-L010RB	32L0538DISCOVERY
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 STM32L0xx ADC HAL and LL API. The LL API is used for performance improvement.	X	-	-	-	-	-	-
	CRC	CRC_PolynomialUpdate	How to use the CRC peripheral through the STM32L0xx CRC HAL and LL API.	X	-	-	-	-	-	-
	DMA	DMA_FLASHToRAM	How to use a DMA to transfer a word data buffer from Flash memory to embedded SRAM through the STM32L0xx DMA HAL and LL API. The LL API is used for performance improvement.	X	-	-	-	-	-	-
	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 STM32L0xx I2C HAL and LL API (LL API usage for performance improvement) and an interrupt.	X	-	-	-	-	-	-
	PWR	PWR_STANDBY_RTC	How to enter Standby mode and wake up from this mode by using an external reset or the RTC wakeup timer through the STM32L0xx RTC and RCC HAL, and LL API (LL API use for maximizing performance).	X	-	-	-	-	-	-
		PWR_STOP	How to enter Stop mode and wake up from this mode by using external reset or wakeup interrupt (all the RCC function calls use RCC LL API for minimizing footprint and maximizing performance).	X	-	-	-	-	-	-
	SPI	SPI_FullDuplex_ComPolling	Data buffer transmission/reception between two boards via SPI using Polling mode.	X	-	-	-	-	-	-
		SPI_HalfDuplex_ComPollingIT	Data buffer transmission/reception between two boards via SPI using Polling (LL driver) and Interrupt modes (HAL driver).	X	-	-	-	-	-	-
	TIM	TIM_PWMInput	Use of the TIM peripheral to measure an external signal frequency and duty cycle.	X	-	-	-	-	-	-
	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 STM32L0xx UART HAL and LL API, the LL API being used for performance improvement.	X	-	-	-	-	-	-
		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 STM32L0xx UART HAL and LL API, the LL API being used for performance improvement.	X	-	-	-	-	-	-
	Total number of examples_mix: 11			**11**	**0**	**0**	**0**	**0**	**0**	**0**

Level	Module Name	Project Name	Description	NUCLEO-L073RZ	NUCLEO-L031K6	NUCLEO-L053R8	STM32L073Z-EVAL	NUCLEO-L011K4	NUCLEO-L010RB	32L0538DISCOVERY
Applications	FatFs	FatFs_uSD_RTOS	How to use STM32Cube firmware with FatFs middleware component as a generic FAT file system module. This example develops an application exploiting FatFs features, with a microSD drive in RTOS mode configuration.	-	-	X	-	-	-	-
		FatFs_uSD_Standalone	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	-	-	-
	FreeRTOS	FreeRTOS_LowPower	How to enter and exit low-power mode with CMSIS RTOS API.	X	X	X	X	-	-	X
		FreeRTOS_LowPower_LPTIM	How to enter Stop mode when all RTOS tasks are suspended.	-	-	X	-	-	-	-
		FreeRTOS_Mail	How to use mail queues with CMSIS RTOS API.	X	X	X	X	-	-	X
		FreeRTOS_Mutexes	How to use mutexes with CMSIS RTOS API.	X	X	X	X	-	-	X
		FreeRTOS_Queues	How to use message queues with CMSIS RTOS API.	X	X	X	X	-	-	X
		FreeRTOS_Semaphore	How to use semaphores with CMSIS RTOS API.	X	X	X	X	-	-	X
		FreeRTOS_SemaphoreFromISR	How to use semaphore from ISR with CMSIS RTOS API.	X	X	X	X	-	-	X
		FreeRTOS_Signal	How to perform thread signaling using CMSIS RTOS API.	X	X	X	X	-	-	X
		FreeRTOS_SignalFromISR	How to perform thread signaling from an interrupt using CMSIS RTOS API..	X	X	X	X	-	-	X
		FreeRTOS_ThreadCreation	How to implement thread creation using CMSIS RTOS API.	X	X	X	X	-	-	X
		FreeRTOS_Timers	How to use timers of CMSIS RTOS API.	X	X	X	X	-	-	X
	IAP	IAP_Binary_Template	This directory contains a set of sources files that build the application to be loaded into Flash memory using In-Application Programming (IAP) using the USART.	-	-	-	X	-	-	-
		IAP_Main	How to build an application to be loaded into Flash memory using In-Application Programming (IAP, through USART).	-	-	-	X	-	-	-
	LCD	LCD_Display_Text	How to use and configure the embedded LCD glass controller to display a simple text and turn on icons.	-	-	-	X	-	-	-
	TouchSensing	TouchSensing_Linear	Use of the STMTouch driver with one linear sensor. This example also uses ECS and DTO.	-	-	-	X	-	-	X

Level	Module Name	Project Name	Description	NUCLEO-L073RZ	NUCLEO-L031K6	NUCLEO-L053R8	STM32L073Z-EVAL	NUCLEO-L011K4	NUCLEO-L010RB	32L0538DISCOVERY
Applications	USB_Device	CDC_Standalone	Use of the USB device application based on the Device Communication Class (CDC) and following the PSTN subprotocol. This application uses the USB Device and UART peripherals.	-	-	-	X	-	-	-
		DFU_Standalone	Compliant implementation of the Device Firmware Upgrade (DFU) capability to program the embedded Flash memory through the USB peripheral.	-	-	X	X	-	-	X
		HID_Standalone	Use of the USB device application based on the Human Interface (HID).	-	-	X	X	-	-	-
		HID_Standalone_BCD	Use of the USB device application (BCD feature) based on the Human Interface (HID).	-	-	X	X	-	-	X
		HID_Standalone_LPM	Use of the USB device application based on the Human Interface (HID) with Link Power Management Protocol (LPM).	-	-	X	X	-	-	-
		HID_TSL_Standalone	Use of the USB device application based on the Human Interface (HID).	-	-	-	-	-	-	X
		HID_TSL_Standalone_LPM	Use of the USB device application based on the Human Interface (HID) with Link Power Management Protocol (LPM).	-	-	-	-	-	-	X
		MSC_Standalone	Use of the USB device application based on the Mass Storage Class (MSC).	-	-	-	X	-	-	-
	Total number of applications: 73			**10**	**10**	**17**	**21**	**0**	**0**	**15**
Demonstrations	-	Adafruit_LCD_1_8_SD_Joystick	Demonstration firmware based on STM32Cube. This example helps you to discover STM32 Cortex-M devices that are plugged onto your STM32 Nucleo board.	X	-	-	-	-	-	-
		Demo	Demonstration firmware based on STM32Cube. This example helps you to discover STM32 Cortex-M devices that are plugged onto your STM32 Evaluation board.	-	-	X	X	-	-	X
		Gravitech_4digits	Demonstration of firmware based on STM32Cube. This example provides firmware to to help you to discover STM32 Cortex-M devices that are plugged onto an your STM32NUCLEO_32 board.	-	X	-	-	X	-	-
	Total number of demonstrations: 6			**1**	**1**	**1**	**1**	**1**	**0**	**1**
Total number of projects: 571				**170**	**69**	**92**	**99**	**53**	**13**	**75**

Revision history

Table 2. **Document revision history**

Date	Version	Changes
06-Jul-2015	1	Initial release.
30-Nov-2015	2	Added SW4STM32 firmware in Section 2: STM32CubeL0 examples. Added NUCLEO-L073RZ, NUCLEO-L011K4, NUCLEO-L031K6, STM32L073Z-EVAL and 32L0538DISCOVERY in Table 1:STM32CubeL0 firmware examples.
13-Apr-2016	3	Added support for Low-Level Driver (LL).
15-Nov-2016	4	Updated Section 2: STM32CubeL0 examples. Updated Table 1: STM32CubeL0 firmware examples.
09-Sep-2017	5	Updated Figure 1: STM32CubeL0 firmware components. Updated Table 1: STM32CubeL0 firmware examples.
08-Nov-2018	6	Replaced “firmware package” by “MCU Package”. Updated STM32Cube™ logo. Replace STM32Cube firmware by STM32Cube MCU Package in the whole document. Added Arm logo in Section 1 Reference documents. Removed reference to TrueStudio and added Arm wordmark notice in Section 2 STM32CubeL0 examples. Updated Table 1. STM32CubeL0 firmware examples to add NUCLEO-L010RB board and new examples related to STM32L0 Value line, and removed FLASH_DualBoot_Workaround example. Removed NUCLEO-L010RB from the whole document and updated Section 2 STM32CubeL0 examples.

Contents

1 Reference documents 2
2 STM32CubeL0 examples 3
Revision history 23