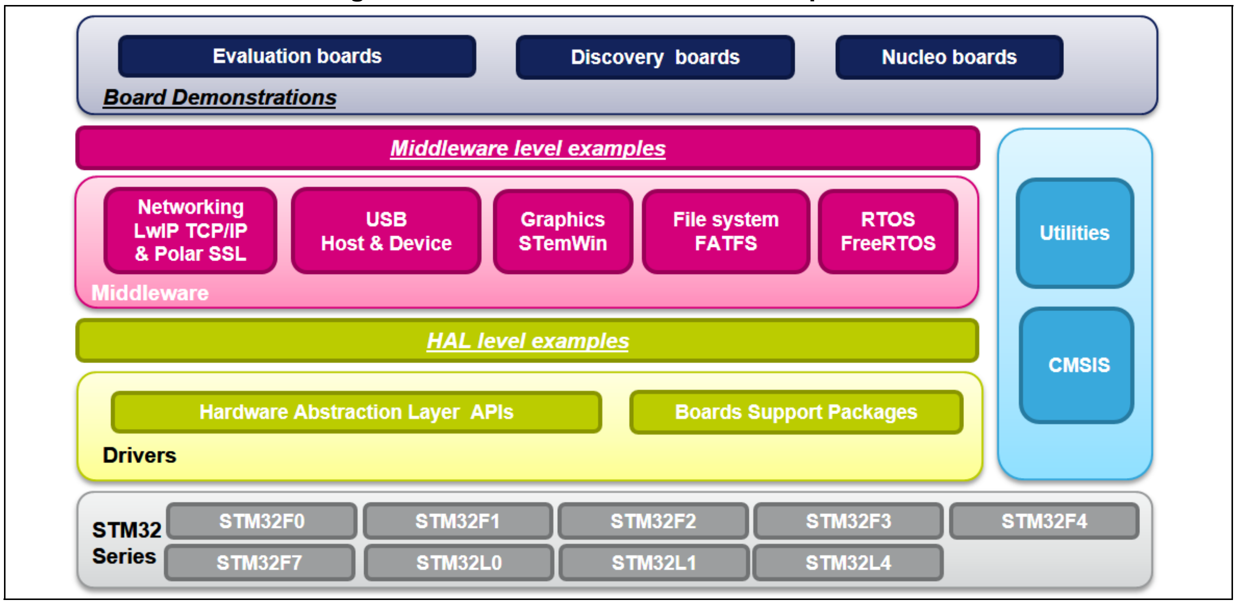
AN4733
Application note

STM32Cube firmware examples for STM32F2 Series

Introduction

The STM32CubeF2 firmware package comes with a rich set of examples running on STMicroelectronics boards. The examples are organized by board and provided with preconfigured projects for the main supported toolchains (see *Figure 1*).

Figure 1. STM32CubeF2 firmware components



Reference documents

The reference documents are available on www.st.com/stm32cube:

- Latest release of STM32CubeF2 firmware package
- *Getting started with the STM32CubeF2 firmware package for STM32F2 Series* user manual (UM1739)
- *STM32Cube USB Device library* user manual (UM1734)
- *STM32Cube USB host library* user manual (UM1720)
- *Developing Applications on STM32Cube with FatFs* user manual (UM1721)
- *Developing Applications on STM32Cube with RTOS* user manual (UM1722)
- *Developing applications on STM32Cube with LwIP TCP/IP stack* user manual (UM1713)
- *STM32Cube Ethernet IAP example* user manual (UM1709)

STM32CubeF2 examples

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

- **Examples:** the 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.
- **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 to integrate and run the maximum number of peripherals and middleware stacks to showcase the product features and performance.
- **Template project:** the template project is provided to allow to quickly build a firmware application on a given board.

The examples are located under *STM32Cube_FW_STM32CubeF2_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 *\TrueSTUDIO* 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 the 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 firmware 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.

Table 1 contains the list of examples provided within STM32CubeF2 firmware package.

Table 1. STM32CubeF2 firmware examples

Level	Module Name	Project Name	Description	STM32F207ZG -Nucleo	STM322xG -EVAL
Templates	-	Starter project	This directory provides a reference template project that can be used to build any firmware application.	X	X
Total number of templates: 2				1	1
Examples	-	BSP	The BSP examples detects the presence of Adafruit 1.8" TFT shield with joystick and uSD.	X	X
	ADC	ADC_DualModeInterleaved	This example provides a short description of how to use two ADC peripherals to perform conversions in interleaved dual-mode.	-	X
		ADC_InjectedConversion_Interrupt	This example describes how to use the ADC in interrupt mode to convert data through the HAL API.	-	X
		ADC_RegularConversion_DMA	This example describes how to use the ADC1 and DMA to transfer continuously converted data from ADC1 to memory.	X	X
		ADC_RegularConversion_Interrupt	This example describes how to use the ADC in interrupt mode to convert data through the HAL API.	X	X
		ADC_RegularConversion_Polling	This example describes how to use the ADC in Polling mode to convert data through the HAL API.	-	X
		ADC_TriggerMode	This example describes how to use the ADC and TIM8 to convert continuously data from ADC channel. Each time an external trigger is generated by TIM2 a new conversion is started by ADC.	-	X
		ADC_TripleModeInterleaved	This example provides a short description of how to use the ADC peripheral to convert a regular channel in Triple interleaved mode.	-	X
	CAN	CAN_LoopBack	This example provides a description of how to set a communication with the CAN in loopback mode.	-	X
		CAN_Networking	This example shows how to configure the CAN peripheral to send and receive CAN frames in the normal mode. The sent frames are used to control LEDs by pressing key pushbutton.	-	X
	CRC	CRC_Example	This example guides the user through the different configuration steps by means of 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

Table 1. STM32CubeF2 firmware examples (continued)

Level	Module Name	Project Name	Description	STM32F207ZG -Nucleo	STM322xG -EVAL
Examples	CRYP	CRYP_AESModes	This example provides a short description of how to use the CRYPTO peripheral to encrypt and decrypt data using AES in chaining modes (ECB, CBC, CTR) and all key size (128, 192, 256) algorithm.	-	X
		CRYP_AES_DMA	This example provides a short description of how to use the CRYPTO peripheral to encrypt and decrypt data using AES-128 Algorithm with ECB chaining mode.	-	X
		CRYP_DESTDESmodes	This example provides a short description of how to use the CRYPTO peripheral to encrypt and decrypt data using DES and TDES in all mode (ECB, CBC) algorithm.	-	X
		CRYP_TDES_DMA	This example provides a short description of how to use the CRYPTO peripheral to encrypt data using TDES Algorithm.	-	X
	Cortex	CORTEXM_MPU	This example presents the MPU feature. The example purpose is to configure a memory region as privileged read only region and tries to perform read and write operation in different mode.	X	X
		CORTEXM_ModePrivilege	This example shows how to modify Cortex-M3 Thread mode privilege access and stack.	-	X
		CORTEXM_SysTick	This example shows how to use the default SysTick configuration with a 1 ms timebase to toggle LEDs.	X	X
	DAC	DAC_SignalsGeneration	This example provides a short description of how to use the DAC peripheral to generate several signals using DMA controller.	-	X
		DAC_SimpleConversion	This example provides a short description of how to use the DAC peripheral to do a simple conversion.	-	X
	DCMI	DCMI_CaptureMode	This example provides a short description of how to use the DCMI to interface with camera module and display in continuous mode the picture on LCD.	-	X
		DCMI_SnapshotMode	This example provides a short description of how to use the DCMI to interface with camera module and display in snapshot mode the picture on LCD.	-	X
	DMA	DMA_FIFOMode	This example provides a description of how to use a DMA channel to transfer a word data buffer from the FLASH memory to an embedded SRAM memory with FIFO mode enabled through the STM32F2xx HAL API.	-	X
		DMA_FLASHToRAM	This example provides a description of how to use a DMA channel to transfer a word data buffer from the Flash memory to an embedded SRAM memory through the HAL API.	X	X

Table 1. STM32CubeF2 firmware examples (continued)

Level	Module Name	Project Name	Description	STM32F207ZG -Nucleo	STM322xG -EVAL
Examples	FLASH	FLASH_EraseProgram	This application describes how to configure and use the FLASH HAL API to erase and program the internal FLASH memory.	X	X
		FLASH_WriteProtection	This example describes how to configure and use the FLASH HAL API to enable and disable the write protection of the internal FLASH memory.	-	X
	FSMC	FSMC_SRAM	This example describes how to configure the FSMC controller to access the SRAM memory.	-	X
		FSMC_SRAM_DataMemory	This example describes how to configure the FSMC controller to access the SRAM memory including heap and stack.	-	X
	GPIO	GPIO_EXTI	This example shows how to configure external interrupt lines.	X	X
		GPIO_IOToggle	This example describes how to configure and use GPIOs through the HAL API.	X	X
	HAL	HAL_TimeBase	This example describes how to customize the HAL time base using a general purpose timer instead of Systick as main source of time base.	X	X
	HASH	HASH_HMAC_SHA1MD5	This example provides a short description of how to use the HASH peripheral to hash data using HMAC SHA-1 and HMAC MD5 Algorithms.	-	X
		HASH_SHA1MD5	This example provides a short description of how to use the HASH peripheral to hash data using SHA-1 and MD5 Algorithms.	-	X
		HASH_SHA1MD5_DMA	This example provides a short description of how to use the HASH peripheral to hash data using SHA-1 and MD5 Algorithms.	-	X
	I2C	I2C_TwoBoards_AdvComIT	This example describes how to perform I2C data buffer transmission/reception between two boards, using an interrupt.	-	X
	I2S	I2S_Audio	This example provides a basic implementation of audio features.	-	X
	IWDG	IWDG_Example	This example describes how to reload the IWDG counter and to simulate a software fault by generating an MCU IWDG reset when a programmed time period has elapsed.	X	X

Table 1. STM32CubeF2 firmware examples (continued)

Level	Module Name	Project Name	Description	STM32F207ZG -Nucleo	STM322xG -EVAL
Examples	PWR	PWR_BOR	This example shows how to configure the programmable BOR thresholds using the FLASH option bytes.	-	X
		PWR_CurrentConsumption	This example shows how to configure the STM32F2xx system to measure the different Low-power mode current consumption. The Low-power modes are: - Sleep Mode - Stop mode with RTC - Standby mode without RTC and BKPSRAM - Standby mode with RTC - Standby mode with RTC and BKPSRAM. To run this example, the user has to follow this step: 1. Select the Low-power modes to be measured by uncommenting the corresponding line inside the stm32f2xx_lp_modes.h file.	X	X
		PWR_PVD	This example shows how to configure the programmable voltage detector using an external interrupt line. In this example, EXTI line 16 is configured to generate an interrupt on each rising or falling edge of the PVD output signal (which indicates that the Vdd voltage is below the PVD threshold).	-	X
		PWR_STANDBY	This example shows how to enter the system to Standby mode and wake-up from this mode using: external RESET, RTC Alarm A or WKUP pin.	-	X
		PWR_STOP	This example shows how to enter Stop mode and wake up from this mode by using the RTC Wakeup timer event or an interrupt.	-	X
	RCC	RCC_ClockConfig	This example describes how to use the RCC HAL API to configure the system clock (SYSCLK) and modify the clock settings on run time.	X	X
	RNG	RNG_MultiRNG	This example guides the user through the different configuration steps by means of the HAL API to ensure RNG random 32-bit numbers generation.	-	X
	RTC	RTC_Alarm	This example guides the user through the different configuration steps by means of the HAL API to ensure Alarm configuration and generation using the RTC peripheral.	-	X
		RTC_Calendar	This example guides the user through the different configuration steps by means of the HAL API to ensure Calendar configuration using the RTC peripheral.	X	X
		RTC_Tamper	This example guides the user through the different configuration steps by means of the RTC HAL API to write/read data to/from RTC Backup registers and demonstrate the Tamper detection feature.	X	X
		RTC_TimeStamp	This example guides the user through the different configuration steps by means of the HAL API to ensure Time Stamp configuration using the RTC peripheral.	-	X

Table 1. STM32CubeF2 firmware examples (continued)

Level	Module Name	Project Name	Description	STM32F207ZG -Nucleo	STM322xG -EVAL
Examples	SMARTCARD	SMARTCARD_T0	This example describes a firmware Smartcard Interface based on the USART peripheral. The main purpose of this firmware example is to provide resources facilitating the development of an application using the USART peripheral in smartcard mode.	-	X
	SPI	SPI_FullDuplex_AdvComIT	This example guides the user through the different configuration steps by means of the HAL API to ensure the SPI Data buffer transmission and reception using Interrupt, in an advance communication mode: the Master board is always sending a command to the slave before any transmission and the slave board is sending an acknowledge before going further.	-	X
		SPI_FullDuplex_ComDMA	This example shows how to perform the SPI data buffer transmission/reception between two boards via DMA.	-	X
		SPI_FullDuplex_ComIT	This example shows how to ensure SPI data buffer transmission/reception between two boards by using an interrupt.	-	X
		SPI_FullDuplex_ComPolling	This example shows how to ensure SPI data buffer transmission/reception in Polling mode between two boards.	-	X
	TIM	TIM_6Steps	This example shows how to configure the TIM1 peripheral to generate 6 Steps.	-	X
		TIM_7PWMOutput	This example shows how to configure the TIM1 peripheral to generate 7 PWM signals with 4 different duty cycles (50%, 37.5%, 25% and 12.5%).	-	X
		TIM_CascadeSynchro	This example shows how to synchronize the TIM peripherals in cascade mode.	-	X
		TIM_ComplementarySignals	This example shows how to configure the TIM1 peripheral to generate three complementary TIM1 signals, to insert a defined dead time value, to use the break feature and to lock the desired parameters.	-	X
		TIM_DMA	This example provides a description of how to use DMA with TIMER update request to transfer data from memory to TIMER Capture Compare Register 3 (CCR3).	X	X
		TIM_DMABurst	This example shows how to update the TIM1 channel1 period and the duty cycle using the TIM1 DMA burst feature.	-	X
		TIM_Encoder	This example shows how to configure the TIM1 peripheral in encoder mode to determinate the rotation direction.	-	X
		TIM_ExtTriggerSynchro	This example shows how to synchronize the TIM peripheral in cascade mode with an external trigger.	-	X

Table 1. STM32CubeF2 firmware examples (continued)

Level	Module Name	Project Name	Description	STM32F207ZG -Nucleo	STM322xG -EVAL
Examples	TIM	TIM_InputCapture	This example shows how to use the TIM peripheral to measure the frequency of an external signal.	X	X
		TIM_OCActive	This example shows how to configure the TIM peripheral in Output Compare Active mode (when the counter matches the capture/compare register, the concerned output pin is set to its active state).	X	X
		TIM_OCInactive	This example shows how to configure the TIM peripheral in Output Compare Inactive mode with the corresponding Interrupt requests for each channel.	-	X
		TIM_OCToggle	This example shows how to configure the TIM peripheral to generate four different signals with four different frequencies.	X	X
		TIM_OnePulse	This example shows how to use the TIM peripheral to generate a One pulse Mode after a rising edge of an external signal is received in timer input pin.	X	X
		TIM_PWMInput	This example shows how to use the TIM peripheral to measure the frequency and duty cycle of an external signal.	X	X
		TIM_PWMOutput	This example shows how to configure the TIM peripheral in PWM (Pulse Width Modulation) mode.	X	X
		TIM_ParallelSynchro	This example shows how to synchronize TIM2 and the timers (TIM3 and TIM4) in parallel mode.	-	X
		TIM_Synchronization	This example shows how to synchronize TIM1 and the timers (TIM3 and TIM4) in parallel mode.	-	X
		TIM_TimeBase	This example shows how to configure the TIM peripheral to generate a time base of one second with the corresponding interrupt request.	-	X
	UART	UART_Hyperterminal_DMA	This example describes an UART transmission (transmit/receive) in DMA mode between a board and an Hyperterminal PC application.	-	X
		UART_Hyperterminal_IT	This example describes an UART transmission (transmit/receive) between a board and an Hyperterminal PC application by using an interrupt.	-	X
		UART_Printf	This example shows how to reroute the C library printf function to the UART. It outputs a message sent by the UART on the HyperTerminal.	X	X
	WWDG	WWDG_Example	This example guides the user through the different configuration steps by means of the HAL API to perform periodic WWDG counter update and simulate a software fault that generates an MCU WWDG reset when a predefined time period has elapsed.	X	X
	Total number of examples: 100			**25**	**75**

Table 1. STM32CubeF2 firmware examples (continued)

Level	Module Name	Project Name	Description	STM32F207ZG -Nucleo	STM322xG -EVAL
Applications	Camera	Camera_To_USBDisk	This application provides a short description of how to use the DCMI to interface with camera module and display in continuous mode the picture on LCD and to save a picture in USB device.	-	X
	Display	LCD_Paint	This application describes how to configure LCD touch screen and attribute an action related to configured touch zone and how to save BMP picture in SD Card.	-	X
	EEPROM	EEPROM_Emulation	This application describes the software solution for substituting a standalone EEPROM by emulating the EEPROM mechanism using the on-chip Flash of STM32F207xx devices.	X	-
	FatFs	FatFs_MultiDrives	This application provides a description on how to use STM32Cube firmware with FatFs middleware component as a generic FAT file system module, in order to develop an application exploiting FatFs offered features with multidrive (RAMDisk, uSD) configuration.	-	X
		FatFs_RAMDisk	This application provides a description on how to use STM32Cube firmware with FatFs middleware component as a generic FAT file system module, in order to develop an application exploiting FatFs offered features with RAM disk (SRAM) drive configuration.	-	X
		FatFs_RAMDisk_RTOS	This application provides a description on how to use STM32Cube firmware with FatFs middleware component as a generic FAT file system module, in order to develop an application exploiting FatFs offered features with RAM disk (SRAM) drive in RTOS mode configuration.	-	X
		FatFs_USBDisk	This application provides a description on how to use STM32Cube firmware with FatFs middleware component as a generic FAT file system module and STM32 USB On-The-Go (OTG) host library, in High Speed (HS) modes (configured in FS), in order to develop an application exploiting FatFs offered features with USB disk drive configuration.	X	X
		FatFs_USBDisk_MultipleAccess_ RTOS	This application provides a description on how to use STM32Cube firmware with FatFs middleware component as a generic FAT file system module, FreeRTOS as an RTOS module based on using CMSIS-OS wrapping layer common APIs, and also STM32 USB On-The-Go (OTG) host library, in both Full Speed (FS) and High Speed (HS) modes, in order to develop an application exploiting FatFs offered features with USB disk drive in RTOS mode configuration.	-	X

Table 1. STM32CubeF2 firmware examples (continued)

Level	Module Name	Project Name	Description	STM32F207ZG -Nucleo	STM322xG -EVAL
Applications	FatFs	FatFs_USBDisk_RTOS	This application provides a description on how to use STM32Cube firmware with FatFs middleware component as a generic FAT file system module, FreeRTOS as an RTOS module based on using CMSIS-OS wrapping layer common APIs, and also STM32 USB On-The-Go (OTG) host library, in both Full Speed (FS) and High Speed (HS) modes, in order to develop an application exploiting FatFs offered features with USB disk drive in RTOS mode configuration.	-	X
		FatFs_uSD	This application provides a description on how to use STM32Cube firmware with FatFs middleware component as a generic FAT file system module, in order to develop an application exploiting FatFs offered features with micro SD drive configuration.	-	X
		FatFs_uSD_RTOS	This application provides a description on how to use STM32Cube firmware with FatFs middleware component as a generic FAT file system module, in order to develop an application exploiting FatFs offered features with micro SD drive in RTOS mode configuration.	-	X
	FreeRTOS	FreeRTOS_LowPower	This directory contains a set of source files that implement an application that uses message queues with CMSIS RTOS API. This application creates two threads.	-	X
		FreeRTOS_Mutexes	This directory contains a set of source files that implement an application that uses mutexes with CMSIS RTOS API. This application creates three threads with different priorities and an access at the same mutex MutexHighPriorityThread() which has the highest priority, so executed first, then it grabs the mutex and sleeps for a short period to let the lower priority threads execution. When it has completed its demonstration functionality, it gives the mutex back before suspending itself.	-	X
		FreeRTOS_Queues	This directory contains a set of source files that implement an application that uses message queues with CMSIS RTOS API. This application creates two threads that send and receive an incrementing number to/from a queue.	-	X
		FreeRTOS_Semaphore	This directory contains a set of source files that implement an application that uses semaphores with CMSIS RTOS API. This application creates two threads that toggle LEDs through a shared semaphore.	-	X
		FreeRTOS_SemaphoreFromISR	This directory contains a set of source files that implement an application that uses semaphore from ISR with CMSIS RTOS API. This application creates a thread that toggles LED through semaphore given from ISR.	-	X

Table 1. STM32CubeF2 firmware examples (continued)

Level	Module Name	Project Name	Description	STM32F207ZG -Nucleo	STM322xG -EVAL
Applications	FreeRTOS	FreeRTOS_ThreadCreation	This directory contains a set of source files that implement a thread creation application using CMSIS RTOS API. This application creates two threads with the same priority, which executes in a periodic cycle of 15 seconds.	-	X
		FreeRTOS_Timers	This directory contains a set of source files that implement an application that uses timers of CMSIS RTOS API This application creates a thread that toggles LED2 every 400 ms, and a periodic timer that calls a callback function every 200 ms to toggle the LED1.	-	X
	LibJPEG	LibJPEG_Decoding	This application demonstrates how to read a jpeg file from the SDCard memory, decode it and display the final BMP image on the LCD.	-	X
		LibJPEG_Encoding	This application demonstrates how to read BMP file from the micro SD, encode it, save the jpeg file in uSD Card then decode the jpeg file and display the final BMP image on the LCD.	-	X
	LwIP	LwIP_HTTP_Server_Netconn_ RTOS	This application guides STM32Cube HAL API users to run a http server application based on Netconn API of LwIP TCP/IP stack. The communication is done with a web browser application in a remote PC.	X	X
		LwIP_HTTP_Server_Raw	This application guides STM32Cube HAL API users to run a http server application based on Raw API of LwIP TCP/IP stack. The communication is done with a web browser application in a remote PC.	-	X
		LwIP_HTTP_Server_Socket_RTOS	This application guides STM32Cube HAL API users to run a http server application based on Socket API of LwIP TCP/IP stack. The communication is done with a web browser application in a remote PC.	-	X
		LwIP_IAP	This application guides STM32Cube HAL API users to run In-Application Programming (IAP) over Ethernet.	-	X
		LwIP_TCP_Echo_Client	This application guides STM32Cube HAL API users to run TCP Echo Client application based on Raw API of LwIP TCP/IP stack. To run this application, on the remote PC, open a command prompt window.	-	X
		LwIP_TCP_Echo_Server	This application guides STM32Cube HAL API users to run TCP Echo Server application based on Raw API of LwIP TCP/IP stack. To run this application, on the remote PC, open a command prompt window.	-	X
		LwIP_TFTP_Server	This application guides STM32Cube HAL API users to run a tftp server demonstration for STM32F2xx devices.	-	X

Table 1. STM32CubeF2 firmware examples (continued)

Level	Module Name	Project Name	Description	STM32F207ZG -Nucleo	STM322xG -EVAL
Applications	LwIP	LwIP_UDPTCP_Echo_Server_ Netconn_RTOS	This application guides STM32Cube HAL API users to run a UDP/TCP Echo Server application based on Netconn API of LwIP TCP/IP stack. To run this application, on the remote PC, open a command prompt window.	-	X
		LwIP_UDP_Echo_Client	This application guides STM32Cube HAL API users to run a UDP Echo Client application based on Raw API of LwIP TCP/IP stack. To run this application, on the remote PC, open a command prompt window.	-	X
		LwIP_UDP_Echo_Server	This application guides STM32Cube HAL API users to run UDP Echo Server application based on Raw API of LwIP TCP/IP stack. To run this application, on the remote PC, open a command prompt window.	-	X
	PolarSSL	SSL_Client	This application guides STM32Cube HAL API users to run an SSL client application based on PolarSSL crypto library and LwIP TCP/IP stack. To off-load the CPU from encryption/decryption, hash and RNG, all these algorithms are implemented using the hardware acceleration AES 128/192/256, Triple DES, MD5, SHA-1 and analog RNG through the STM32Cube HAL APIs. In this application the client (STM3221G-EVAL) sends a crypted message to the server (test PC), which will decrypt the message then reply to the client.	-	X
		SSL_Server	This application guides STM32Cube HAL API users to run an SSL Server application based on PolarSSL crypto library and LwIP TCP/IP stack. To off-load the CPU from encryption/decryption, hash and RNG, all these algorithms are implemented using the hardware acceleration AES 128/192/256, Triple DES, MD5, SHA-1, SHA2-2 and analog RNG through the STM32Cube HAL APIs. The HTTP server (STM3221G-EVAL) contains a html page dynamically refreshed (every 1 s), it shows the RTOS statistics in runtime. The HyperTerminal can be used to debug messages exchanged between the client and the server.	-	X
	STemWin	STemWin_HelloWorld	This directory contains a set of source files that implements a simple "Hello World" application based on STemWin for STM32F2xx devices.	-	X
		STemWin_SampleDemo	This directory contains a set of source files that implements a sample demonstration application allowing to show some of the STemWin Library capabilities on STM32F2xx devices.	-	X

Table 1. STM32CubeF2 firmware examples (continued)

Level	Module Name	Project Name	Description	STM32F207ZG -Nucleo	STM322xG -EVAL
Applications	USB_ Device	AUDIO_Standalone	This application is a part of the USB Device Library package using STM32Cube firmware. It describes how to use the USB device application based on the AUDIO class implementation of an audio streaming (Out: Speaker/Headset) capability on the STM32F2xx devices.	-	X
		CDC_Standalone	This application is a part of the USB Device Library package using STM32Cube firmware. It describes how to use the USB device application based on the Device Communication class (CDC) following the PSTN sub protocol in the STM32F2xx devices using the OTG-USB and UART peripherals.	-	X
		CustomHID_Standalone	This application is a part of the USB Device Library package using STM32Cube firmware. It describes how to use USB device application based on the Custom HID Class on the STM32F2xx devices.	-	X
		DFU_Standalone	This application is a part of the USB Device Library package using STM32Cube firmware. It describes how to use the USB device application based on the Device Firmware Upgrade (DFU) on the STM32F2xx devices.	X	X
		DualCore_Standalone	This application is a part of the USB Device Library package using STM32Cube firmware. It describes how to use USB device application based on the STM32F2xx multi core support feature integrating Mass Storage (MSC) and Human Interface (HID) in the same project.	-	X
		HID_Standalone	This application is a part of the USB Device Library package using STM32Cube firmware. It describes how to use the USB device application based on the Human Interface (HID) on the STM32F2xx devices.	X	X
		MSC_Standalone	This application is a part of the USB Device Library package using STM32Cube firmware. It describes how to use the USB device application based on the Mass Storage Class (MSC) on the STM32F2xx devices.	-	X
	USB_Host	AUDIO_Standalone	This application is a part of the USB Host Library package using STM32Cube firmware. It describes how to use the USB host application based on the Audio OUT class on the STM32F2xx devices.	-	X
		CDC_Standalone	This application is a part of the USB Host Library package using STM32Cube firmware. It describes how to use the USB host application based on the Communication Class (CDC) on the STM32F2xx devices.	-	X
		DualCore_Standalone	This application is a part of the USB Host Library package using STM32Cube firmware. It describes how to use the USB host application based on the STM32F2xx multi core support feature integrating Mass Storage (MSC) and Human Interface (HID) in the same project.	-	X

Table 1. STM32CubeF2 firmware examples (continued)

Level	Module Name	Project Name	Description	STM32F207ZG -Nucleo	STM322xG -EVAL
Applications	USB_Host	DynamicSwitch_Standalone	This application is a part of the USB Host Library package using STM32Cube firmware. It describes how to use dynamically a switch, on the same port, between available USB host applications on the STM32F2xx devices.	-	X
		FWupgrade_Standalone	This application is a part of the USB Host Library package using STM32Cube firmware. It describes how to use the USB host application based on the In-Application programming (IAP) on the STM32F2xx devices.	-	X
		HID_RTOS	This application is a part of the USB Host Library package using STM32Cube firmware. It describes how to use the USB host application based on the Human Interface Class (HID) on the STM32F2xx devices.	-	X
		HID_Standalone	This application is a part of the USB Host Library package using STM32Cube firmware. It describes how to use the USB host application based on the Human Interface Class (HID) on the STM32F2xx devices.	X	X
		MSC_RTOS	This application is a part of the USB Host Library package using STM32Cube firmware. It describes how to use the USB host application based on the Mass Storage Class (MSC) on the STM32F2xx devices in RTOS mode configuration.	-	X
		MSC_Standalone	This application is a part of the USB Host Library package using STM32Cube firmware. It describes how to use the USB host application based on the Mass Storage Class (MSC) on the STM32F2xx devices.	X	X
	Total number of applications: 56			**7**	**49**
Demonstration	-	Demo	The provided demonstration firmware based on STM32Cube helps the user to discover STM32 Cortex-M devices that can be plugged on a STM32 Nucleo board.	X	-
	Total number of demonstration: 1			**1**	**0**
Total number of projects: 159				**34**	**125**

1 Revision history

Table 2. Document revision history

Date	Revision	Changes
23-Jul-2015	1	Initial release.
26-Nov-2015	2	Updated *Table 1: STM32CubeF2 firmware examples* adding the list of examples and applications provided with the STM32F207ZG-Nucleo board.