Getting started with the USB Type-C™ software expansion for STM32Cube

Introduction

The X-CUBE-TCPP software package contains the demo application examples for the USB Type-C™ expansion boards for STM32 Nucleo (X-NUCLEO-SNK1M1, X-NUCLEO-SRC1M1, and X-NUCLEO-DRP1M1) featuring the TCPP01-M12 USB Type-C™ port protection device for sink applications, the TCPP02-M18 USB Type-C™ port protection device for source applications, and the TCPP03-M20 USB Type-C™ Power Delivery device for dual role power (DRP) applications.

For sink applications, the expansion board is plugged onto an STM32 Nucleo development board (any STM32 Nucleo-64 development board, NUCLEO-G071RB, NUCLEO-G474RE, or NUCLEO-L412RB-P) with an STM32 microcontroller that executes the code.

When acting in the sink role, the X-CUBE-TCPP selects the highest and closest power profile to the value indicated by the binary file from the power profiles available on the source.

For source applications, the expansion board is plugged onto an STM32 Nucleo development board (NUCLEO-G071RB, NUCLEO-G474RE or NUCLEO-F446RE) with an STM32 microcontroller that executes the code.

For DRP applications, the expansion board is plugged onto an STM32 Nucleo development board with an STM32 microcontroller that features a USB Type-C™ Power Delivery controller (STM32G0, STM32G4, STM32L5, STM32U5).

The X-CUBE-TCPP can be downloaded from www.st.com or GitHub.

Related links

Visit the STM32Cube ecosystem web page on www.st.com for further information

1 Acronyms and abbreviations

Table 1. List of acronyms and abbreviations

Term	Definition
AMS	Atomic message sequence
APDO	Augmented power data object
BMC	Bi-phase mark coding
BSP	Board support package
CAD	Cable detection module
DFP	Downstream facing port
DPM	Device policy manager
DRD	Dual role data (ability for a product to act as either Host or Device)
DRP	Dual role power (ability for a product to be either Source or Sink)
DRS	Data role swap
GUI	Graphical user interface
HAL	Hardware abstraction layer
HW	Hardware
LL	Low layer
MSC	Message sequence chart
PDO	Power data object
PE	Policy engine
PHY	Physical layer
PPS	Programmable power supply
PRL	Physical protocol layer
PRS	Power role swap
SNK	Power sink capability
SRC	Power source capability
UCPD	USB Type-C™ power delivery
UFP	Upstream facing port

2 X-CUBE-TCPP software expansion for STM32Cube

2.1 Overview

The X-CUBE-TCPP software package expands STM32Cube functionality and provides middleware extensions, applicative layers and examples for USB Type-C™ sink expansion boards for STM32 Nucleo featuring the TCPP01-M12 USB Type-C™ port protection device for sink applications, the TCPP02-M18 USB Type-C™ port protection device for source applications, and the TCPP03-M20 USB Type-C™ Power Delivery device for dual role power (DRP) applications.

The key features are:

- Demo application example files for sink applications, using:
  - the X-NUCLEO-SNK1M1 USB Type-C™ Power Delivery expansion board connected to any STM32 Nucleo-64 development board (for USB Type-C™ sink at 5 V only without Power Delivery). Example provided for the NUCLEO-L412RB-P
  - the X-NUCLEO-SNK1M1 USB Type-C™ Power Delivery expansion board connected to a NUCLEO-G071RB or NUCLEO-G474RE development board (for USB Type-C™ sink with Power Delivery up to 100 W)
- Demo application example files for source applications using:
  - the X-NUCLEO-SRC1M1 USB Type-C™ Power Delivery expansion board connected to any STM32 Nucleo-64 development board (for USB Type-C™ source without Power Delivery). Example provided for NUCLEO-F446RE
  - the X-NUCLEO-SRC1M1 USB Type-C™ Power Delivery expansion board connected to a NUCLEO-G071RB or NUCLEO-G474RE development board (for USB Type-C™ source with Power Delivery)
- Demo application example files for dual role power applications using:
  - the X-NUCLEO-DRP1M1 USB Type-C Power Delivery expansion board connected to a NUCLEO-G071RB or NUCLEO-G474RE development board for USB Type-C™ DRP with Power Delivery
- Package compatible with STM32CubeMX
- Easy portability across different MCU families, thanks to STM32Cube
- Free user-friendly license terms

2.1.1 USB Power Delivery

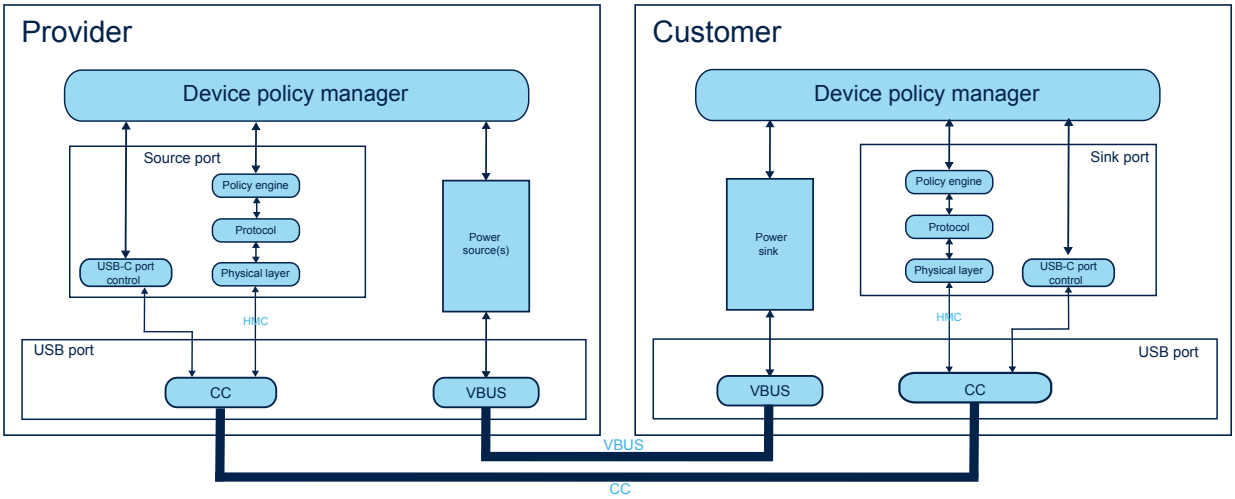
As stated in the USB-IF website, the USB Power Delivery Specification enables the maximum functionality of USB by providing more flexible power delivery along with data over a single cable. Its aim is to operate with and build on the existing USB ecosystem.

The USB power delivery specification defines a power delivery system covering all elements of a USB system including hosts, devices, hubs, chargers and cable assemblies. The specification describes the architecture, protocols, power supply behavior, connectors and cabling necessary to manage power delivery over USB at up to 100 W.

The figure below shows the communication stack, consisting in:

- a device policy manager that exists in all devices and manages USB power delivery resources within the device across one or more ports according to the device local policy;
- a policy engine that exists in each USB power delivery port and implements the local policy for that port;
- a protocol layer that enables messages to be exchanged between a source port and a sink port;
- a physical layer that handles transmission and reception of bits on the wire and handles data transmission;
- the USB-C port control that handles the Type-C detection state machines.

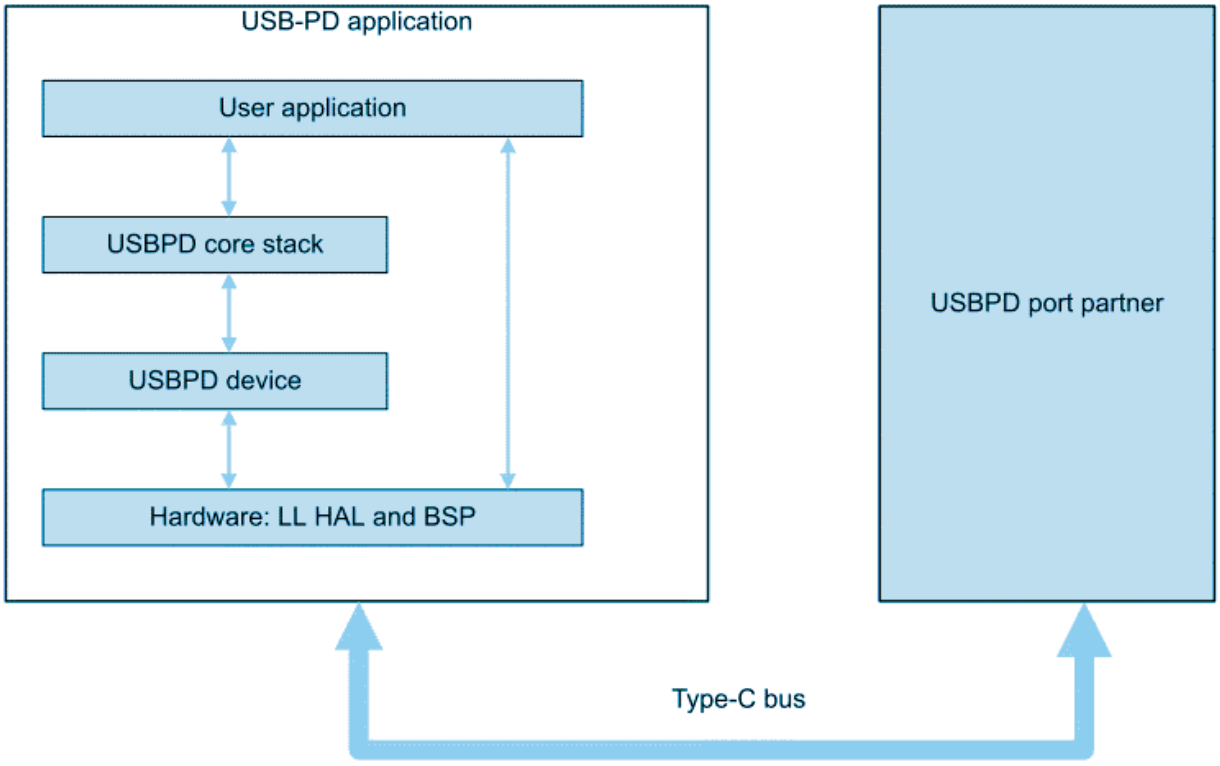
Figure 1. **USB PD protocol stack diagram**



2.1.2 USB PD software applications for STM32

In the STM32Cube firmware packages, the top level architecture of any STM32 USB Power Delivery application can be described as per the figure below.

Figure 2. **STM32 USBPD software application architecture**



The software components used to build a USBPD application are:

- User application: corresponding to the Device Policy Manager plus Power management;
- USBPD core stack: corresponding to Policy engine plus Protocol Layer, as described in USB-IF specifications;
- USBPD device: driver interface for physical layer plus USB-C port control functions;
- Hardware layer: low-level resources used to interact with the hardware HAL, LL and BSP.

2.2 Architecture

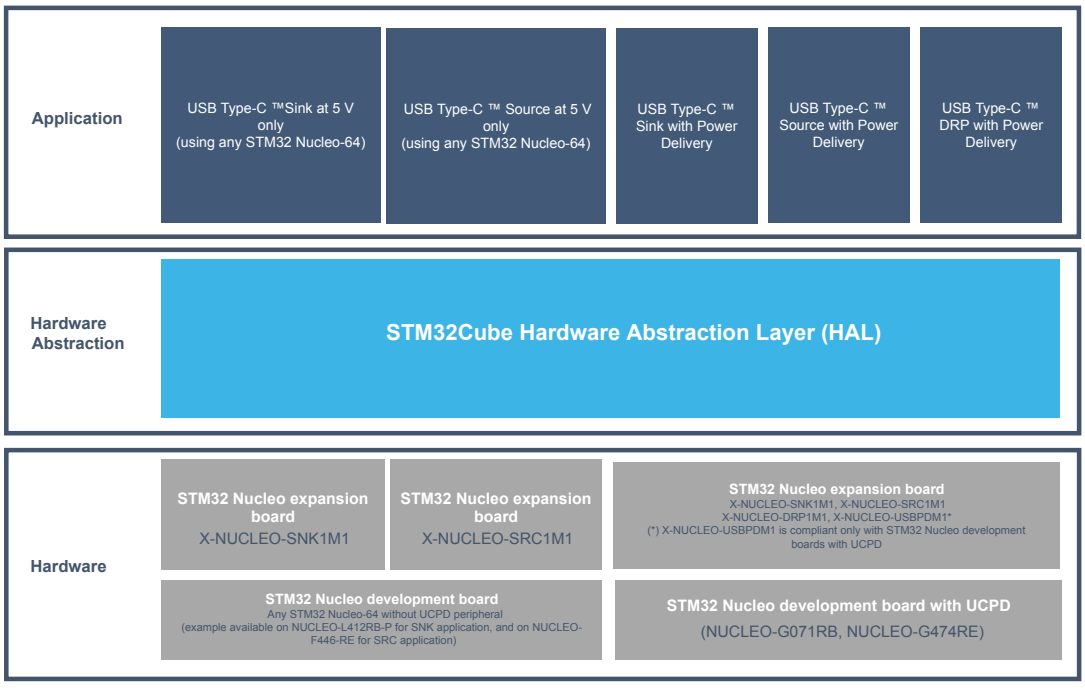
This software is a fully compliant expansion for STM32Cube enabling development of USB Power Delivery and dual role power applications.

The software is based on the hardware abstraction layer for the STM32 microcontroller, STM32CubeHAL. The package extends STM32Cube by providing complete middleware for the Bluetooth low energy expansion board and some middleware components for serial communication with a PC.

The software layers used by the application software to access and use the TCPP01-M12/TCPP02-M18/TCPP03-M20 expansion boards are:

- The STM32Cube HAL driver layer provides a simple, generic, and multi-instance set of APIs (application programming interfaces) to interact with the upper layers (application, libraries, and stacks). It includes generic and extension APIs and is based on a generic architecture, which allows the layers built on it (such as the middleware layer) to implement their functionalities without dependence on the specific hardware configuration of a given microcontroller unit (MCU). This structure improves library code reusability and guarantees high portability across other devices.
- USB-PD middleware components (USBPD Core Stack and USBPD Device drivers)
- USBPD applications and demonstrations
- The board support package (BSP) layer provides supporting software for the peripherals on the STM32 Nucleo board, except for the MCU. It has a set of APIs to provide a programming interface for certain board-specific peripherals (the LED, the user button etc.) and allow identification of the specific board version.

Figure 3. X-CUBE-TCPP software architecture



2.3 BSP for USB Power Delivery

In the STM32 USB Power Delivery application architecture, as delivered within X-CUBE-TCPP software package (see Figure 3), a BSP driver is dedicated to USBPD feature. Goal of this USBPD PWR BSP driver is to provide a common API to Application (user code) + USBPD middleware in order to achieve hardware operations related to USB PD, in a board independent manner.

This software component is identified as USBPD PWR BSP driver and is located in the dedicated expansion board BSP driver (for example in Drivers\BSP\X-NUCLEO-DRP1M1 directory for the X-NUCLEO-DRP1M1 expansion board).

Note: No BSP USBPD PWR is provided for X-NUCLEO-SNK1M1, as its applications use default functions provided in the generic application code (weak functions apply, no overloading with real BSP functions is needed).

The following sections provide details on USBPD PWR BSP component APIs that could be used by any application supporting the USBPD feature on a given hardware. As several Type-C ports could be handled simultaneously, all APIs accept a Type-C port number as entry parameter which identifies the port interested by the API call. The descriptions refer to the Type-C port identified by the port number value provided as input parameter to the API call.

The detailed APIs handle the various types of USBPD applications types (SRC, SNK or DRP), on the various identified software. For a given configuration and hardware, all APIs might not be required or relevant.

2.3.1 Initialisation/De-initialization

The USBPD PWR BSP component driver provides the API for initializing/de-initializing global USBPD power resources for a given Type-C port on a given board:

- `BSP_USBPD_PWR_Init()` - performs global initialization of BSP in charge of managing the USBPD power resource for the given port
- `BSP_USBPD_PWR_Deinit()` - performs de-initialization of BSP in charge of managing the USBPD power resource for the given port

2.3.2 Role management

The USBPD PWR BSP component driver provides the API to the application and to the USBPD stack to reflect the current role value on the PD interface into the BSP USBPD PWR module:

- `BSP_USBPD_PWR_SetRole()`: informs the BSP USBPD PWR of PD role (SRC or SNK) currently active on the applicative (User application or USBPD Device middleware) side. According to the current role, some actions could be taken in the BSP layer to properly handle the configuration

2.3.3 Power mode management

The USBPD PWR BSP component driver provides the API and the functionality to be able to configure power saving mode (if applicable):

- `BSP_USBPD_PWR_SetPowerMode()` - sets the BSP USBPD PWR power saving mode to the requested value
- `BSP_USBPD_PWR_GetPowerMode()` - gets the current BSP USBPD PWR power saving mode

2.3.4 V_{BUS} management

All V_{BUS} related APIs are common to all BSP components supporting PD feature for V_{BUS} management.

- `BSP_USBPD_PWR_VBUSInit()` - initializes hardware resources involved in V_{BUS} management and measurement. This initialization is required to allow the application to properly measure and check V_{BUS} level during PD session
- `BSP_USBPD_PWR_VBUSDeInit()` - de-initializes hardware resources involved in V_{BUS} management and measurement
- `BSP_USBPD_PWR_VBUSOn()` - enables V_{BUS} signal delivery (only to be used when port acts as SRC)
- `BSP_USBPD_PWR_VBUSOff()` - disables V_{BUS} signal delivery (only to be used when port acts as SRC)
- `BSP_USBPD_PWR_VBUSIsOn()` - provides V_{BUS} status (On or Off)
- `BSP_USBPD_PWR_VBUSSetVoltage_Fixed()` - configures power source according to requirements extracted from a fixed PDO in terms of voltage and current (only to be used when port acts as SRC)
- `BSP_USBPD_PWR_VBUSSetVoltage_Variable()` - configures power source according to requirements extracted from a variable PDO in terms of voltage and current (only to be used when port acts as SRC)
- `BSP_USBPD_PWR_VBUSSetVoltage_Battery()` - configures power source according to requirements extracted from a battery PDO in terms of voltage and current (only to be used when port acts as SRC)
- `BSP_USBPD_PWR_VBUSSetVoltage_APDO()` - configures power source according to requirements extracted from an APDO in terms of voltage and current (only to be used when port acts as SRC)
- `BSP_USBPD_PWR_SetVBUSDisconnectionThreshold()` - configures the V_{BUS} level that triggers a disconnection event. If V_{BUS} goes below the programmed level, a callback function could be registered
- `BSP_USBPD_PWR_VBUSGetVoltage()` - gets actual voltage level measured on the V_{BUS} line
- `BSP_USBPD_PWR_VBUSGetCurrent()` - gets actual current level measured on the V_{BUS} line
- `BSP_USBPD_PWR_VBUSDischargeOn()` - activates discharge on V_{BUS} line
- `BSP_USBPD_PWR_VBUSDischargeOff()` - de-activates discharge on V_{BUS} line

2.3.5 V_{CONN} management

- `BSP_USBPD_PWR_VCONNInit()` - initializes hardware resources involved in V_{CONN} management. This initialization is required to allow the application to properly manage V_{CONN} level during PD session
- `BSP_USBPD_PWR_VCONNDeInit()` - de-initializes hardware resources involved in V_{CONN} management
- `BSP_USBPD_PWR_VCONNOn()` - enables V_{CONN} sourcing on the selected CC line
- `BSP_USBPD_PWR_VCONNOff()` - disables V_{CONN} sourcing on the selected CC line
- `BSP_USBPD_PWR_VCONNIsOn()` - provides V_{CONN} status (on or off)
- `BSP_USBPD_PWR_VCONNDischargeOn()` - activates discharge on V_{CONN} line
- `BSP_USBPD_PWR_VCONNDischargeOff()` - de-activates discharge on V_{CONN} line

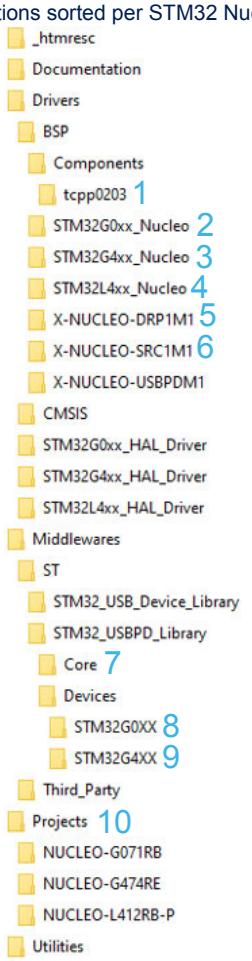
2.3.6 Callback registration

- `BSP_USBPD_PWR_RegisterVBUSDetectCallback()` - registers callback function to be invoked when V_{BUS} is considered as provided (V_{BUS} present) or when V_{BUS} falls below programmed threshold (V_{BUS} absent). Usually, a callback function can be provided by the USBPD Device component
- `BSP_USBPD_PWR_EventCallback()` - default callback implemented in BSP USBPD PWR component that is executed when an event is reported by lower layers (optional)

2.4 Folder structure

Figure 4. X-CUBE-TCPP package folder structure

1. TCPP0203 component driver (needed for X-NUCLEO-DRP1M1 expansion board)
2. Board support package (BSP) for STM32G0xx development boards supported in the STM32Cube firmware expansion package
3. Board support package (BSP) for STM32G4xx development boards supported in the STM32Cube firmware expansion package
4. Board support package (BSP) for STM32L4xx development boards supported in the STM32Cube firmware expansion package
5. Additional BSP component dedicated to TCPP0203 component management, also including I²C bus management feature for TCPP03 driving
6. Additional BSP component dedicated to TCPP0203 component management, also including I²C bus management feature for TCPP03 driving
7. USBPD core stack library
8. USBPD middleware device driver for STM32G0xx series
9. USBPD middleware device driver for STM32G4xx series
10. Project directory containing USBPD applications sorted per STM32 Nucleo development board



The main folders included in the software package are:

- **Documentation**: contains a compiled HTML file generated from the source code, detailing the software components and APIs.
- **Drivers**: contains the HAL drivers, the specific drivers for each supported board or hardware platform, including the on-board components and the CMSIS layer (vendor-independent hardware abstraction layer for the Cortex-M processor series).
- **Middlewares**: contains the USBPD core stack library, drivers and protocols related to host software and applications for USBPD.
- **Projects**: contains the USBPD sample applications for the NUCLEO-G071RB, NUCLEO-G474RE, NUCLEO-L412RB-P platforms with three development environments (IAR Embedded Workbench for ARM, RealView Microcontroller Development Kit (MDK-ARM) and STM32CubeIDE).
- **Utilities**: contains software components as Embedded Trace system (TRACER_EMB), or embedded module used in board connection with STM32CubeMonUCPD (GUI_INTERFACE).

3 System setup

3.1 Hardware description

3.1.1 STM32 Nucleo

STM32 Nucleo development boards provide an affordable and flexible way for users to test solutions and build prototypes with any STM32 microcontroller line.

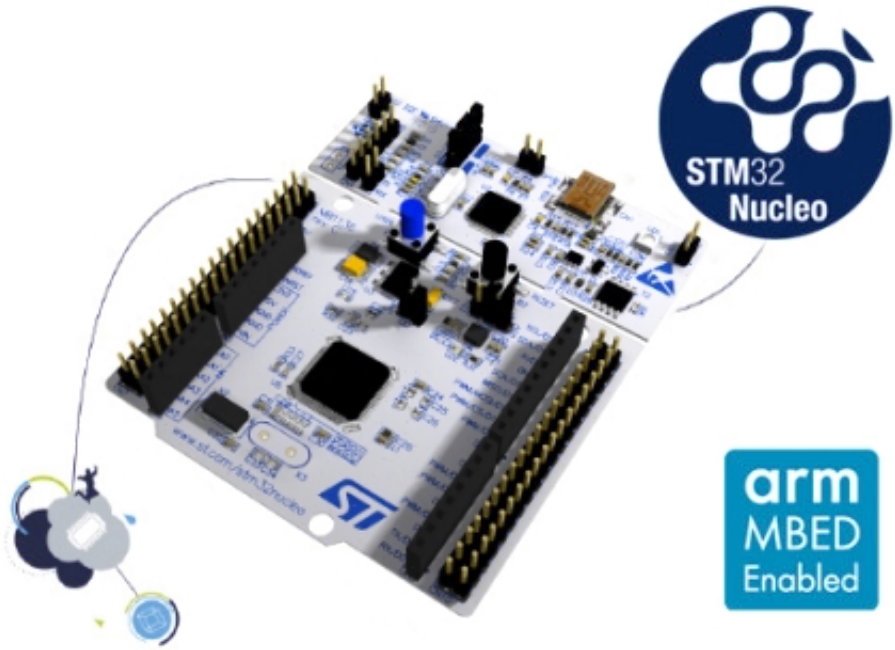
The Arduino connectivity support and ST morpho connectors make it easy to expand the functionality of the STM32 Nucleo open development platform with a wide range of specialized expansion boards to choose from.

The STM32 Nucleo board does not require separate probes as it integrates the ST-LINK/V2-1 debugger/programmer.

The STM32 Nucleo board comes with the comprehensive STM32 software HAL library together with various packaged software examples for different IDEs (IAR EWARM, Keil MDK-ARM, STM32CubeIDE, mbed and GCC/LLVM).

All STM32 Nucleo users have free access to the mbed online resources (compiler, C/C++ SDK and developer community) at www.mbed.org to easily build complete applications.

Figure 5. STM32 Nucleo board



3.1.2 X-NUCLEO-DRP1M1 expansion board

The X-NUCLEO-DRP1M1 expansion board allows evaluating the features of TCPP03-M20 and the USB Type-C™ features and protections required for V_{BUS} and CC lines suitable for dual role power (DRP) applications.

The expansion board can be stacked on top of any STM32 Nucleo-64 with Power Delivery (UCPD) peripheral embedded in their microcontrollers.

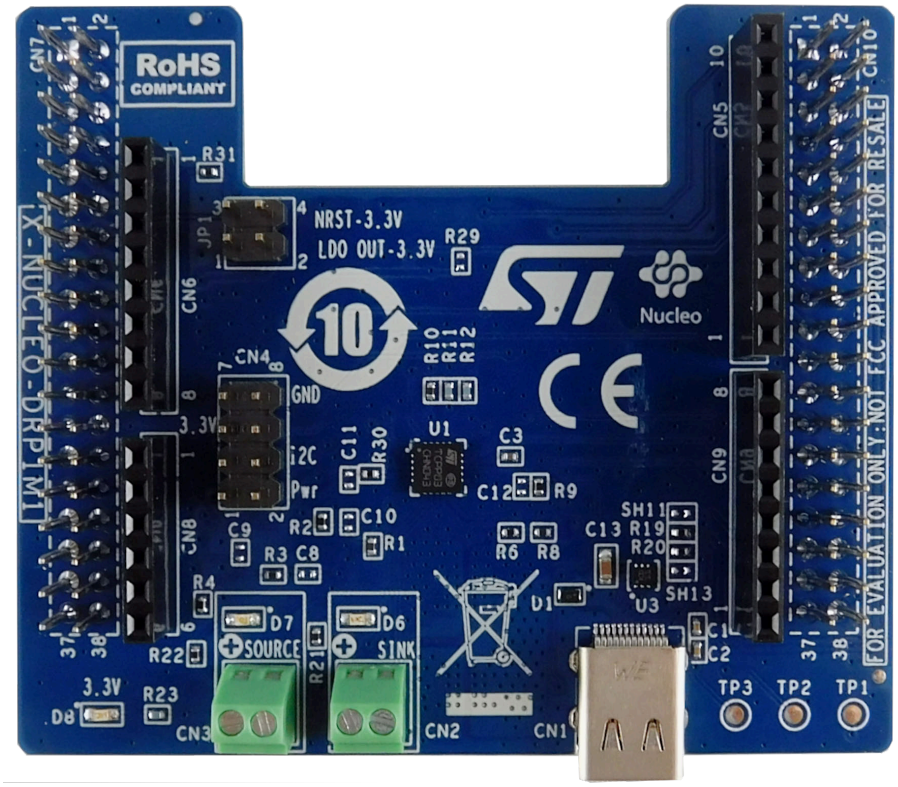
The X-NUCLEO-DRP1M1 effectively demonstrates the dead battery and Sink operation, thanks to the integrated ST715PU33R LDO linear regulator that supplies the connected STM32 Nucleo development board. It also demonstrates USB Type-C™ Source operation when a compatible external Source is connected to the board.

Moreover, the expansion board allows Dual Role Data functionalities for sourcing devices.

The X-NUCLEO-DRP1M1 is compliant with the USB Type-C™ and Power Delivery specifications 3.1 standard power range (SPR) and is USB-IF certified as a 100 W DRP solution supporting programmable power supply (PPS).

The companion software package (X-CUBE-TCPP) contains the application examples for development boards embedding UCPD-based microcontrollers (NUCLEO-G071RB and NUCLEO-G474RE) that can be ported to other development boards embedding UCPD-based microcontrollers (for example, NUCLEO-G0B1RE).

Figure 6. X-NUCLEO-DRP1M1 expansion board



3.1.3 X-NUCLEO-SRC1M1 expansion board

The X-NUCLEO-SRC1M1 expansion board allows evaluating the features of the TCPP02-M18 as well as the USB Type-C™ features and protections required for VBUS and CC lines suitable for source applications.

The expansion board can be stacked on top of any STM32 Nucleo-64 with Power Delivery (UCPD) peripheral embedded in their microcontrollers.

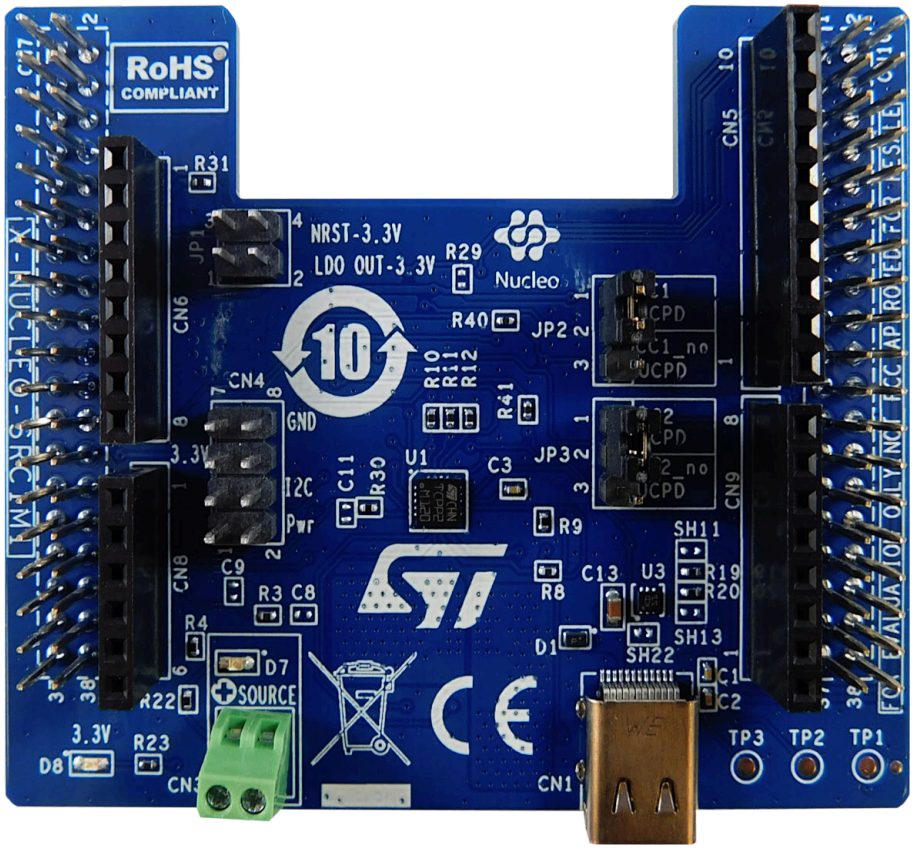
It can also be stacked on other STM32 Nucleo development boards not supporting the UCPD peripheral to demonstrate the USB Type-C™ basic operations (attach, detach, and power supply current capability recognition).

The X-NUCLEO-SRC1M1 effectively demonstrates the USB Type-C™ source basic operations (attach, detach, power supply control, and current capability information) when a compatible external sink is connected to the board.

The X-NUCLEO-SRC1M1 is compliant with the USB Type-C™ and Power Delivery specifications 3.1 standard power range (SPR). It also supports programmable power supply (PPS).

The companion software package (X-CUBE-TCPP) contains the application examples for development boards that embed a UCPD-based microcontroller (NUCLEO-G071RB, NUCLEO-G474RE) and for non-UCPD ones (NUCLEO-F446RE).

Figure 7. X-NUCLEO-SRC1M1 expansion board



3.1.4 X-NUCLEO-SNK1M1 expansion board

The X-NUCLEO-SNK1M1 expansion board allows evaluating the features of TCPP01-M12 and the USB Type-C overvoltage protection for V_{BUS} and CC lines suitable for Sink applications.

The expansion board is designed to be stacked on top of any STM32 Nucleo-64 development board exploiting the characteristics of the USB Type-C and Power Delivery (UCPD) peripheral embedded in their microcontrollers.

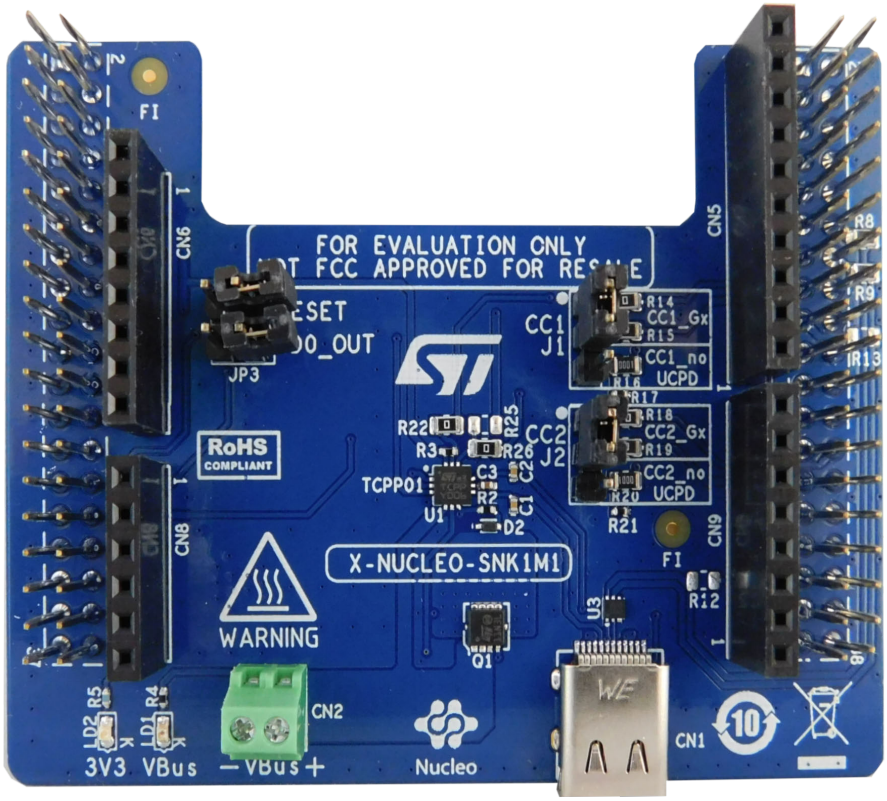
It can also be stacked on other STM32 Nucleo development boards not supporting the UCPD peripheral to demonstrate the Type-C basic operations (attach, detach and power supply current capability recognition).

The X-NUCLEO-SNK1M1 provides an effective demonstration of the dead battery operation, thanks to the integrated ST715PU33R LDO linear regulator that supplies the connected STM32 Nucleo development board when a Source is attached via a USB Type-C connector.

The X-NUCLEO-SNK1M1 is compliant with the latest USB Type-C and Power Delivery specifications and is also USB-IF certified as a 100 W solution supporting Programmable Power Supply (PPS) function.

The companion software package (X-CUBE-TCPP) contains the application examples for development boards embedding UCPD-based microcontrollers (NUCLEO-G071RB, NUCLEO-G474RE) and for non-UCPD ones (NUCLEO-L412RB-P).

Figure 8. X-NUCLEO-SNK1M1 expansion board



Note: Before running any demo, set CC1 J1, CC2 J2, JP3 and JP4 jumpers according to the configuration described in Demo application setup.

3.2 Software setup

The following software components are needed to set up a suitable development environment for creating applications for the STM32 Nucleo equipped with the X-NUCLEO-SNK1M1, X-NUCLEO-SRC1M1 or X-NUCLEO-DRP1M1 expansion board:

- X-CUBE-TCPP expansion software
- Development toolchain and compiler. The STM32Cube expansion software supports the following environments:
  - IAR Embedded Workbench for Arm® (EWARM) toolchain + ST-LINK/V2
  - RealView Microcontroller Development Kit (MDK-ARM) toolchain + ST-LINK/V2
  - STM32CubeIDE + ST-LINK/V2

Once X-CUBE-TCPP expansion firmware package has been downloaded, unzipping the archive will install the folder structure as described in Figure 4.

The Projects directory contains several software examples that illustrates USBPD application building on top of each USB Type-C expansion boards for STM32 Nucleo. Projects are sorted per STM32 Nucleo board hosting the device (STM32G0, STM32F4, STMG2G4 or STM32L4).

Project names indicate the type of USB Type-C expansion board that is targeted and the type of the corresponding application. For example, the project located in Projects\NUCLEO-G071RB\Applications\USB_PD\DRP1M1_DRP contains the application code for a DRP project that runs on a NUCLEO-G071RB development board stacked with an X-NUCLEO-DRP1M1 expansion board.

4 Demo application examples

4.1 X-NUCLEO-DRP1M1 expansion board use with STM32G071 or STM32G474 STM32 Nucleo-64 boards

The X-NUCLEO-DRP1M1 expansion board allows using the TCPP03-M20 device with STM32 devices supporting the USBPD feature on boards not including the Type-C connector.

The board permits you to add the Type-C connector, protected by the TCPP03-M20, on top of existing STM32 Nucleo development boards (for example, either a NUCLEO-G071RB or NUCLEO-G474RE development boards).

4.1.1 NUCLEO-G071RB setup

The NUCLEO-G071RB STM32 Nucleo development board has to be configured with the following jumper positions:

- JP2 on STLK for firmware flashing
- JP3 closed (IDD)
- CN4 closed 1-2, 3-4 for connection with STM32CubeMonUCPD

4.1.2 NUCLEO-G474RE setup

The NUCLEO-G474RE STM32 Nucleo development board has to be configured with the following jumper positions:

- 5V_SEL on 5V_STLK for firmware flashing
- JP6 closed (IDD)
- JP8 closed on 1-2 (VREF)
- JP3 closed

4.1.3 X-NUCLEO-DRP1M1 setup

The X-NUCLEO-DRP1M1 expansion board must be configured in the following way:

- JP1 1-2 open (LDO_OUT)
- JP1 3-4 open (NRST)

In order to be able to act as SRC (Provider Role), a power supply in line with SRC PDO characteristics should be connected to CN3 connector.

Note: An alternative configuration is possible (for example, to supply V_{CC} to MCU using power source connected to X-NUCLEO-DRP1M1 CN3). In this configuration, you have to:

- *close JP1 1-2 and JP1 3-4*
- *remove STLK jumper connected on the STM32 Nucleo development board*
- *close JP8 2-3 on NUCLEO-G474RE*

4.1.4 Configuration settings

To stack the expansion board on top of the STM32 Nucleo board, all configuration settings related to the board specific hardware are gathered in a dedicated configuration file named “drp1m1_conf.h” and located in the Inc directory of the application project.

4.1.4.1 GPIO/EXTI settings

The following settings are related to STM32 GPIOs connected to TCPP03-M20 through the X-NUCLEO-DRP1M1 expansion board:

- GPIO for TCPP03-M20 FLGn pin management (port and number, initialization parameters, port clock macros)
- EXTI line associated to FLGn signal falling edge detection and the corresponding IRQ handler
- GPIO dedicated to the TCPP03-M20 EN pin management

4.1.4.2 *I2C connection*

The settings related to I2C bus communication define:

- I2C instance to be used for TCPP03-M20 communication (I2C1, I2C2, etc.);
- the corresponding I2C SDA and SCL pin ports, and numbers.

4.1.4.3 *V_{BUS} measurement using ADC*

The following settings apply to ADC used in V_{BUS} measurement:

- ADC instance
- ADC channel
- GPIO port and number to be measured by ADC

4.2 X-NUCLEO-SRC1M1 expansion board use with STM32G071, STM32G474 or STM32F446 STM32 Nucleo-64 boards

The X-NUCLEO-SRC1M1 expansion board allows using the TCPP02-M18 device with STM32 devices supporting the USBPD feature on boards not including the USB Type-C™ connector.

The board permits you to add the USB Type-C™ connector, protected by the TCPP02-M18, on top of existing STM32 Nucleo development boards (for example, either a NUCLEO-G071RB, NUCLEO-G474RE or NUCLEO-F446RE development board).

4.2.1 NUCLEO-G071RB setup

The NUCLEO-G071RB STM32 Nucleo development board has to be configured with the following jumper positions:

- JP2 on STLK for firmware flashing
- JP3 closed (IDD)
- CN4 closed 1-2, 3-4 for connection with STM32CubeMonUCPD

4.2.2 NUCLEO-G474RE setup

The NUCLEO-G474RE STM32 Nucleo development board has to be configured with the following jumper positions:

- 5V_SEL on 5V_STLK for firmware flashing
- JP6 closed (IDD)
- JP8 closed on 1-2 (VREF)
- JP3 closed

4.2.3 NUCLEO-F446RE setup

The NUCLEO-F446RE STM32 Nucleo development board has to be configured with the following jumper positions:

- JP6 closed (IDD)
- JP5 closed on 1-2 (U5V)

4.2.4 X-NUCLEO-SRC1M1 setup

When connected to an STM32 Nucleo development board supporting UCPD IP (STM32G0xx or STM32G4xx), the X-NUCLEO-SRC1M1 expansion board must be configured in the following way:

- JP1 1-2 open (LDO_OUT)
- JP1 3-4 open (NRST)

In order to be able to act as an SRC (provider role), a power supply in line with SRC PDO characteristics should be connected to CN3 connector.

Note: An alternative configuration is possible (for example, to supply VCC to MCU by using power source connected to X-NUCLEO-SRC1M1 CN3). In this configuration, you have to:

- *close JP1 1-2 and JP1 3-4*
- *remove STLK jumper connected on the STM32 Nucleo development board*
- *close JP8 2-3 on NUCLEO-G474RE*

When connected to an STM32 Nucleo development board not supporting UCPD IP (STM32F4xx for instance), the X-NUCLEO-SRC1M1 expansion board must be configured in the following way:

- Move CC1 (J2) jumper to the CC1 no UCPD position 2/3
- Move CC2 (J3) jumper to the CC2 no UCPD position 2/3
- JP1 1-2 [LDO_OUT] jumper on
- JP1 3-4 [RESET] jumper on
- Supply CN3 with a 5 V power supply
- Connect PA3 (CN10 - 37) and PC4 (CN10 - 34) together
- Optional, for up to 3.0 A
  - Remove R35 and place it on SH19
  - Remove R39 and place it on SH21
  - Change R5 shunt resistor to a 10 mOhms one

4.2.5 Configuration settings

To stack the expansion board on top of the STM32 Nucleo development board, all configuration settings related to the board specific hardware are gathered in a dedicated configuration file named “src1m1_conf.h” and located in the Inc directory of the application project.

4.2.5.1 GPIO/EXTI settings

The following settings are related to STM32 GPIOs connected to TCPP02-M18 through the X-NUCLEO-SRC1M1 expansion board:

- GPIO for TCPP02-M18 FLGn pin management (port and number, initialization parameters, port clock macros)
- EXTI line associated to FLGn signal falling edge detection and the corresponding IRQ handler
- GPIO dedicated to the TCPP02-M18 EN pin management

4.2.5.2 I2C connection

The settings related to I2C bus communication define:

- I2C instance to be used for TCPP02-M18 communication (I2C1, I2C2, etc.);
- the corresponding I2C SDA and SCL pin ports, and numbers.

4.2.5.3 V_{BUS} measurement using ADC

The following settings apply to ADC used in V_{BUS} measurement:

- ADC instance
- ADC channel
- GPIO port and number to be measured by ADC

4.3 X-NUCLEO-SNK1M1 expansion board use with STM32G071, STM32G474 or STM32L412RB STM32 Nucleo-64 boards

The X-NUCLEO-SNK1M1 expansion board allows using the TCPP01-M12 device with STM32 devices supporting the USBPD feature on boards not including the USB Type-C™ connector.

The board permits you to add the USB Type-C™ connector, protected by the TCPP01-M12, on top of existing STM32 Nucleo development boards (for example, either a NUCLEO-G071RB or NUCLEO-G474RE development boards).

The X-NUCLEO-SNK1M1 can also be used with STM32 devices not supporting USBPD feature in order to implement USB Type-C™ sink applications only (no Power Delivery protocol support, but anyway capable of connecting to a USB Type-C™ only power supply).

4.3.1 NUCLEO-G071RB setup

The NUCLEO-G071RB STM32 Nucleo development board has to be configured with the following jumper positions:

- JP2 on STLK for firmware flashing
- JP3 closed (IDD)
- CN4 closed 1-2, 3-4 for connection with STM32CubeMonUCPD

4.3.2 NUCLEO-G474RE setup

The NUCLEO-G474RE STM32 Nucleo development board has to be configured with the following jumper positions:

- 5V_SEL on 5V_STLK for firmware flashing
- JP6 closed (IDD)
- JP8 closed on 1-2 (VREF)
- JP3 closed

4.3.3 NUCLEO-L412RB setup

The NUCLEO-L412RB STM32 Nucleo development board can be used in its default configuration.

4.3.4 X-NUCLEO-SNK1M1 setup

The X-NUCLEO-SNK1M1 expansion board has to be configured in the following way:

- JP3 open (LDO_OUT)
- JP4 open (NRST)
- J1 and J2 closed on 1-2 when connected to an STM32 Nucleo development board supporting UCPD IP (STM32G0xx or STM32G4xx)
- J1 and J2 closed on 2-3 when connected to an STM32 Nucleo development board not supporting UCPD IP (STM32L4xx for instance)

4.3.5 Configuration settings

To stack the expansion board on top of the STM32 Nucleo board, all configuration settings related to the board specific hardware are gathered in the “main.h”/”main.c” files located in Inc and Src directories of the application project.

4.3.5.1 GPIO settings

The following settings are related to STM32 GPIOs connected to TCPP01-M12 through the X-NUCLEO-SNK1M1 expansion board:

- GPIO for TCPP01-M12 DB pin management
- GPIO for TCPP01-M12 V_{CC} pin management

4.3.5.2 V_{BUS} measurement using ADC

The following settings apply to ADC used in V_{BUS} measurement:

- ADC instance
- ADC channel
- GPIO port and number to be measured by ADC

Appendix A References

USB specifications:

- USB2.0 Universal Serial Bus Revision 2.0 Specification
- USB3.1 Universal Serial Bus Revision 3.2 Specification
- Universal Serial Bus Type-C Cable and Connector Specification 2.0, August 2019
- Universal Serial Bus Power Delivery Specification, Revision 3.0, Version 2.0, August 28, 2019

Freely available resources on www.st.com:

- AN5225: "USB Type-C Power Delivery using STM32 MCUs and MPUs"
- TA0357: "Overview of USB Type-C and Power Delivery technologies"
- TCPP01-M12 datasheet - DS12900: "USB-C overvoltage protection for VBUS and CC lines"
- TCPP02-M18 datasheet - DS13787: "USB type-C protection for source application"
- TCPP03-M20 datasheet - DS13618: "USB-C protection for dual role power (DRP)"
- UM2552: "Managing USB power delivery systems with STM32 microcontrollers"
- UM2891: "Getting started with the X-NUCLEO-DRP1M1 USB Type-C™ Power Delivery"
- UM2773: "Getting started with the X-NUCLEO-SNK1M1 USB Type-C™ Power Delivery Sink expansion board based on TCPP01-M12 for STM32 Nucleo"

Revision history

Table 2. **Document revision history**

Date	Revision	Changes
04-Oct-2021	1	Initial release.
09-Dec-2021	2	Updated Introduction, Section 2.1 Overview, Section 2.2 Architecture, Section 2.4 Folder structure, Section 3.2 Software setup, Section 4.1.4.2 I2C connection, and Section Appendix A References. Added Section 3.1.3 X-NUCLEO-SRC1M1 expansion board, Section 4.2 X-CUBESRC1M1 expansion board use with STM32G071 or STM32G474 STM32 Nucleo-64 boards, Section 4.1.1 NUCLEO-G071RB setup, Section 4.1.2 NUCLEO-G474RE setup, Section 4.2.3 XNUCLEO-SRC1M1 setup, Section 4.2.4 Configuration settings, Section 4.2.4.1 GPIO/EXTI settings, Section 4.2.4.2 I2C connection, and Section 4.1.4.3 VBUS measurement using ADC.
09-May-2022	3	Updated introduction, Section 2.1 Overview, Section 2.2 Architecture, Section 3.2 Software setup, and Section Appendix A References. Removed Section 3.1.5 X-NUCLEO-USBPDM1 expansion board, Section 4.4 X-NUCLEOUSBPDM1 expansion board use with STM32G071 or STM32G474 STM32 Nucleo-64 boards, Section 4.4.1 NUCLEO-G071RB setup, Section 4.4.2 NUCLEO-G474RE setup, Section 4.4.3 XNUCLEO-USBPDM1 setup, Section 4.4.4 Configuration settings, Section 4.4.4.1 GPIO settings, and Section 4.4.4.2 VBUS measurement using ADC. Removed all references to the X-NUCLEO-USBPDM1 expansion board.
29-Jun-2022	4	Updated introduction, Section 2.1 Overview, Section 2.2 Architecture, Section 3.1.3 X-NUCLEO-SRC1M1 expansion board, Section 3.1.4 X-NUCLEO-SNK1M1 expansion board, Section 3.2 Software setup, Section 4.2 X-NUCLEO-SRC1M1 expansion board use with STM32G071, STM32G474 or STM32F446 STM32 Nucleo-64 boards, Section 4.2.4 X-NUCLEO-SRC1M1 setup, and Section 4.2.5 Configuration settings. Added Section 4.2.3 NUCLEO-F446RE setup.

Contents

List of tables

List of figures