Getting started with the X-CUBE-SOUNDTER1 sound terminal software expansion for STM32Cube

Introduction

X-CUBE-SOUNDTER1 is designed to allow the user to output multi-channel audio data with the STA350BW Sound Terminal® device mounted on the X-NUCLEO-CCA01M1 expansion board.

X-CUBE-SOUNDTER1 provides the complete middleware for STM32 to simplify application development with the STA350BW integrated solution for digital audio processing, digital amplifier control and FFX-power output stage of the Sound Terminal® device family. It is highly portable across different MCU families, thanks to STM32Cube. This package contains device drivers for full component control, dedicated middleware to simplify the design of biquadratic filter functions of the Sound Terminal®, and sample applications to stream standard sample PCM data to the component via the I²S or SAI buses. The audio streams are then heard through the speakers connected to the X-NUCLEO-CCA01M1 expansion board.

The software provides sample applications for STM32 Nucleo platforms stacked with the X-NUCLEO-CCA01M1 expansion board with an embedded STA350BW Sound Terminal® device. A single expansion board allows the output of a standard 2-channel stereo signal, while two X-NUCLEO-CCA01M1 boards on the same STM32 Nucleo allows 4-channel output with two independent I²S signals.

The software is based on STM32Cube technology and expands the range of STM32Cube-based packages.

1 Acronyms and abbreviations

Acronym	Description
PCM	Pulse Code Modulation
I²C	Inter-Integrated Circuit
I²S	Integrated Interchip Sound
BSP	Board Support Package
HAL	Hardware Abstraction Layer
IDE	Integrated Development Environment

2 What is STM32Cube?

■

What is STM32Cube?

STMCube™ is an STMicroelectronics initiative that helps you reduce development effort, time and cost. STM32Cube covers the STM32 portfolio.

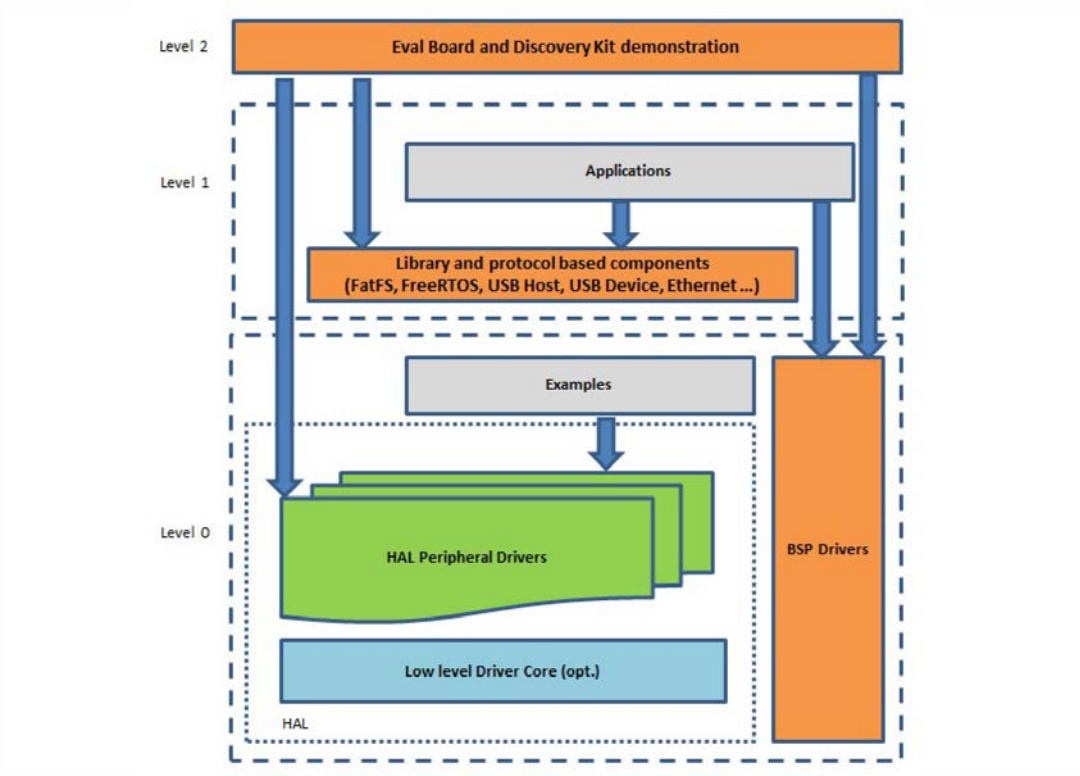
STM32Cube version 1.x includes:

- STM32CubeMX, a graphical software configuration tool that allows the generation of C initialization code using graphical wizards.
- A comprehensive embedded software platform specific to each series (such as the STM32CubeF4 for the STM32F4 series), which includes:
  - the STM32Cube HAL embedded abstraction-layer software, ensuring maximized portability across the STM32 portfolio
  - a consistent set of middleware components such as RTOS, USB, TCP/IP and graphics
  - all embedded software utilities with a full set of examples

2.1 STM32Cube architecture

The STM32Cube firmware solution is built around three independent levels that can easily interact with one another, as described in the diagram below.

Figure 1. Firmware architecture



Level 0: This level is divided into three sub-layers:

- Board Support Package (BSP): this layer offers a set of APIs relative to the hardware components in the hardware boards (Audio codec, IO expander, Touchscreen, SRAM driver, LCD drivers. etc…); it is based on

modular architecture allowing it to be easily ported on any hardware by just implementing the low level routines. It is composed of two parts:

  – Component: is the driver relative to the external device on the board and not related to the STM32, the component driver provides specific APIs to the external components of the BSP driver, and can be ported on any other board.
  – BSP driver: links the component driver to a specific board and provides a set of easy to use APIs. The API naming convention is BSP_FUNCT_Action(): e.g., BSP_LED_Init(), BSP_LED_On().

- Hardware Abstraction Layer (HAL): this layer provides the low level drivers and the hardware interfacing methods to interact with the upper layers (application, libraries and stacks). It provides generic, multi-instance and function-oriented APIs to help offload user application development time by providing ready to use processes. For example, for the communication peripherals (I²C, UART, etc.) it provides APIs for peripheral initialization and configuration, data transfer management based on polling, interrupt or DMA processes, and communication error management. The HAL Drivers APIs are split in two categories: generic APIs providing common, generic functions to all the STM32 series and extension APIs which provide special, customized functions for a specific family or a specific part number.
- Basic peripheral usage examples: this layer houses the examples built around the STM32 peripherals using the HAL and BSP resources only.

Level 1: This level is divided into two sub-layers:

- Middleware components: set of libraries covering USB Host and Device Libraries, STemWin, FreeRTOS, FatFS, LwIP, and PolarSSL. Horizontal interaction among the components in this layer is performed directly by calling the feature APIs, while vertical interaction with low-level drivers is managed by specific callbacks and static macros implemented in the library system call interface. For example, FatFs implements the disk I/O driver to access a microSD drive or USB Mass Storage Class.
- Examples based on the middleware components: each middleware component comes with one or more examples (or applications) showing how to use it. Integration examples that use several middleware components are provided as well.

Level 2: This level is a single layer with a global, real-time and graphical demonstration based on the middleware service layer, the low level abstraction layer and basic peripheral usage applications for board-based functions.

3 X-CUBE-SOUNDTER1 software expansion for STM32Cube

3.1 Overview

X-CUBE-SOUNDTER1 is a software package that expands the functionality provided by STM32Cube.

The key features of the package are:

- Complete driver and middleware to build applications using STA350BW Sound Terminal® device
- Control of STA350BW Sound Terminal to implement basic functions and advanced DSP management
- User-friendly BSP interface for an easy configuration of the device functions (initialization, audio playback, volume and mute control) and biquadratic filter management
- Dedicated middleware to facilitate biquadratic filter design based on standard filter typologies and parameters
- Easy portability across different MCU families thanks to STM32Cube
- Free, user-friendly license terms
- Sample implementation available on X-NUCLEO-CCA01M1 board when connected to a NUCLEO-F401RE, NUCLEO-F072RB, NUCLEO-L053R8, NUCLEO-L476RG or NUCLEO-F746ZG board.

This software includes all the elements needed for the development of advanced applications based on the X-NUCLEO-CCA01M1 expansion board, simplifying all the steps involved in getting the most out of the the Sound Terminal® device, including:

- device initialization through I²C bus
- microcontroller audio bus initialization (I²S or SAI)
- basic audio control functions (play, stop, pause, resume, etc.)
- configuration and control of the advanced embedded features (tone, biquadratic filter setup, etc.)

For all this to be possible, several firmware components have been developed, such as special Middleware to compute filter coefficients for Sound Terminal® devices, a dedicated component driver for the STA350BW chip and BSP layers to support the STM32 Nucleo boards.

3.2 Architecture

This software is a fully-compliant STM32Cube expansion which enables the development of applications using the digital STA350BW Sound Terminal® device.

The software interacts with the STM32CubeHAL hardware abstraction layer for the STM32 microcontroller and extends the STM32Cube suite by providing a board support package (BSP) for the Sound Terminal® expansion board and special middleware components for advanced signal filtering.

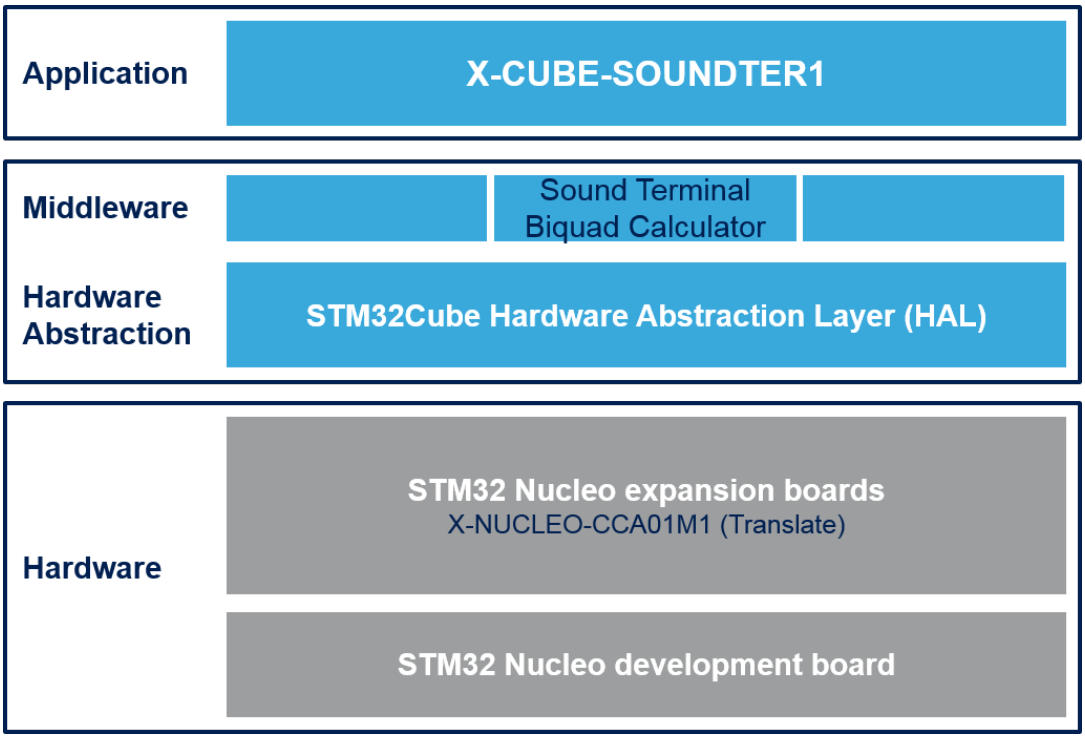
The software accesses and manipulates the expansion board functions via the following firmware layers:

STM32Cube HAL layer: consists of a set of simple, generic, multi-instance APIs (application programming interfaces) which interact with the upper layer applications, libraries and stacks. These generic and extension APIs are based on a common framework which allows any layers they built on, such as the middleware layer, to implement their functions without requiring specific hardware information for a given microcontroller unit (MCU). This structure improves library code reusability and guarantees easy portability across other devices.

Board Support Package (BSP) layer: provides software support for the STM32 Nucleo board peripherals, excluding the MCU. These specific APIs provide a programming interface for certain board specific peripherals like LEDs, user buttons, etc and can also be used to fetch individual board version information.

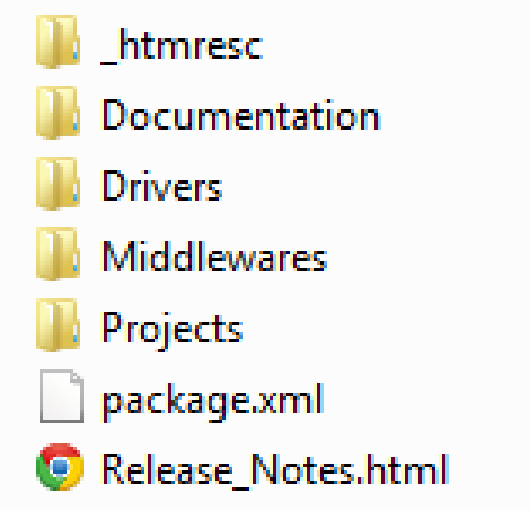
For the Sound Terminal® expansion board, it provides the programming interface for the STA350BW Sound Terminal® device and provides support for initializing the required MCU peripherals and writing data to the device.

Figure 2. **X-CUBE-SOUNDTER1 software architecture**



3.3 Folder structure

Figure 3. **X-CUBE-SOUNDTER1 package folder structure**



The following folders are included in the software package:

Documentation: contains a compiled HTML file generated from the source code with software component and API details.

Drivers: contains the HAL drivers, the board specific drivers for each supported board or hardware platform, including the onboard components and the CMSIS vendor-independent hardware abstraction layer for the Cortex-M processor series.

Middlewares: contains the library resources to facilitate STA350BW biquadratic filter coefficient computation.

Projects: contains a sample microphone data access application for the NUCLEO-F401RE, NUCLEO-F072RB, NUCLEO-L053R8, NUCLEO-L476RG or NUCLEO-F746ZG platforms, with three development environments (IAR Embedded Workbench for ARM, RealView Microcontroller Development Kit (MDK-ARM) and System Workbench for STM32 (SW4STM32)).

3.4 APIs

Detailed technical information regarding the user APIs can be found in a compiled HTML file inside the package Documentation folder, with full function and parameter descriptions.

3.5 Sample application information

A sample application for the X-NUCLEO-CCA01M1 expansion board coupled with either the NUCLEO-F401RE, NUCLEO-F072RB, NUCLEO-L053R8, NUCLEO-L476RG or NUCLEO-F746ZG board is provided in the Projects directory. Ready-to-build projects are available for multiple IDEs.

3.5.1 Application details

The application is designed to play a pre-recorded audio frame from the MCU flash memory through the STA350BW device mounted on the X-NUCLEO-CCA01M1 expansion board. The application implements basic audio functions like device initialization, play, stop and pause, and advanced Sound Terminal® features like biquadratic filtering and equalization management.

The application performs the following steps:

1. Initializes communication peripherals such as I²C and I²S or SAI buses.
2. Initializes the STA350BW audio device through the I²C bus.
3. Starts playing back the audio frame from the STM32 flash memory.
4. Computes certain sets of biquadratic filter coefficients using the relevant Sound Terminal® Biquad Calculator middleware. Alternatively, equalization presets are available in the code and can be used to facilitate audio equalization..

   Detailed technical information regarding the Sound Terminal® Biquad Calculator APIs can be found in a compiled HTML file in the package Documentation folder with full function and parameter descriptions.
5. Sets up the filter configuration. It is possible to switch between different equalizer configurations and setups using the user button on the STM32 Nucleo board.

BSP and component drivers are provided to ease application development. They can manage up to two X-NUCLEO-CCA01M1 expansion boards receiving data from two different I²S buses of the microcontroller. In this way, you can build an audio system capable of handling up to four independent channels.

Note that solder bridges on the expansion board must be set up according to desired system configuration (for further information refer to Section 4.3.2 Jumper configurations).

The audio-related components of the application are mainly in the audio_application.c file which uses the dedicated X-NUCLEO-CCA01M1 BSP layer, the Sound Terminal® Biquad Calculator library middleware and the STA350BW component driver.

Setting up the system for audio streaming requires the following steps, which you can trace in the application:

1. Check the hardware configuration for solder bridges and jumpers (referring to Section 4.3.2 Jumper configurations).
2. All the required peripherals must be configured according to the desired sampling frequency and the desired master volume. On further examination, we can see that the following BSP functions are used:
   a. BSP initialization: initializes the hardware peripherals with the desired streaming settings.
   b. BSP play: starts the audio streaming to the device. A double buffer mechanism is implemented and interrupts are generated to allow the user to fill the buffer with the desired audio data at each iteration. The source data buffer and its size can be determined inside the function call.
   c. BSP stop: stops the data acquisition.
3. The interrupt service routine related to audio data acquisition is implemented in a callback in the audio_application.c file (`BSP_AUDIO_OUT_Transfer_CallBack(…)`) in which the new audio data is used to fill the buffer.

a. The user can change the implementation of this function to apply, for example, DSP functions to the audio before sending it to the STA350BW device.

4. In the application, one of the following operations is performed each time the user button is pressed:
   a. A set of specific biquadratic coefficients is computed by the relevant Sound Terminal® Biquad Calculator middleware. The STA350BW registers are then written in order to store filter coefficients and to activate equalization via the BSP SetEq function.
   b. Tone value is managed
   c. Volume / mute controls on each channel is performed
5. The LEDs mounted on STM32 Nucleo blink according to the current setting.

In order to support two different STA350BW audio devices, a unique id number is associated with each one and must be passed as a parameter to the BSP functions in order to ensure control of the right device.

Hardware and solder bridge configurations differ according to the id number associated with the specific X-NUCLEO-CCA01M1 board, due to the necessity to communicate via different I²C and I²S or SAI buses and I²C addresses. Refer to the tables in Section 4.3.2 Jumper configurations to determine the correct configuration for the corresponding board id number.

3.6 Notes on Sound Terminal® Biquad Calculator middleware

The Biquad Calculator middleware allows the user to compute equalization coefficients that can be used to configure the biquadratic filters on the Sound Terminal® devices. It is designed to facilitate the application of the equalization capacity of the STA350BW and can be used to obtain the exact values to be stored in the device RAM in order to achieve the desired frequency response.

Precise coefficients can be obtained based on common filter parameters such as:

- sampling frequency
- filter type (first or second order low/high pass, band pass, notch, low/high shelf)
- cut frequency
- gain
- quality factor
- slope

Detailed mathematical description of the filter design phases can be found in application note AN3984, available at: www.st.com. The principles described are also exploited in the APWorkbench software utility (www.st.com) that, in simulation mode, represents a valid alternative for the computation of filter parameters and allows the evaluation of the frequency response with a graphical tool.

Detailed technical information regarding the Sound Terminal® Biquad Calculator APIs can be found in a compiled HTML file located in the package Documentation folder with full function and parameter descriptions.

4 System setup guide

4.1 Hardware description

This section describes the hardware components required for developing applications based on STA350BW Sound Terminal® device.

4.1.1 STM32 Nucleo platform

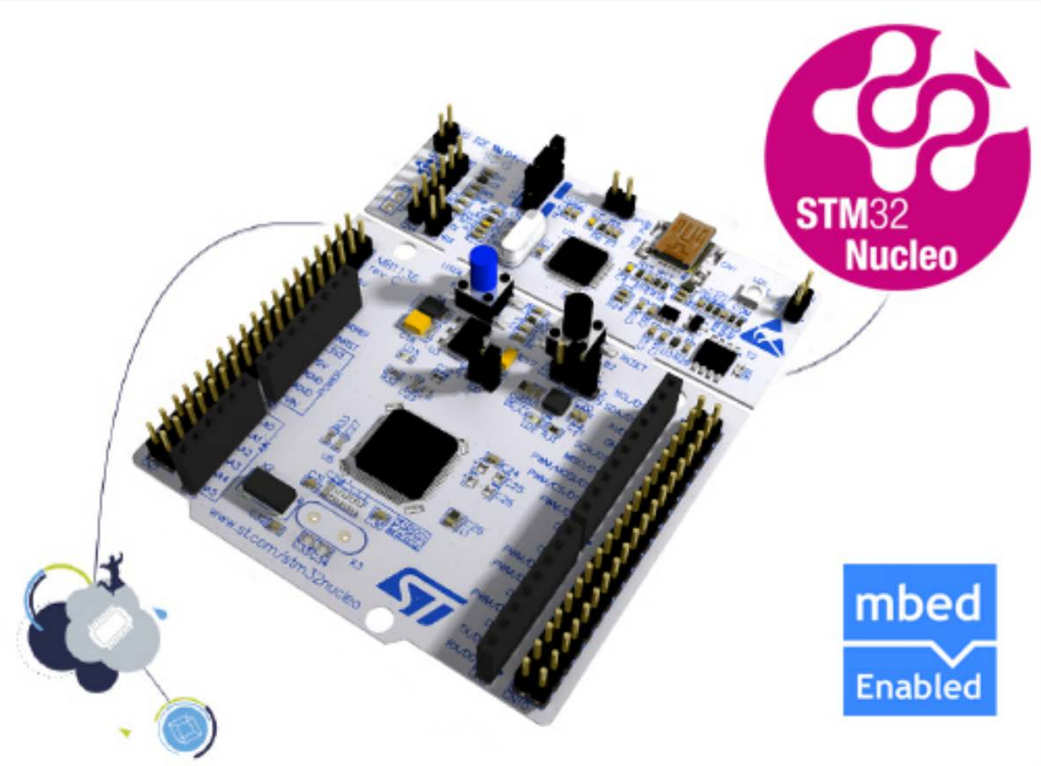
STM32 Nucleo development boards provide an affordable and flexible way for users to test solutions and build prototypes with any STM32 microcontroller line.

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.

Figure 4. STM32 Nucleo board



Information regarding the STM32 Nucleo board is available at www.st.com/stm32nucleo

4.1.2 X-NUCLEO-CCA01M1 expansion board

The X-NUCLEO-CCA01M1 is an Audio Out evaluation board based on the STA350BW device, which is part of the Sound Terminal® family that provides full digital audio streaming to speaker outputs in solutions that feature cost effectiveness, low power dissipation and sound enrichment. The X-NUCLEO-CCA01M1 expansion board is compatible with the ST morpho connector layout and is designed to enable communication between the audio device and the STM32 via the I²C bus for control and I²S or SAI buses for data streaming.

Figure 5. **X-NUCLEO-CCA01M1 expansion board**

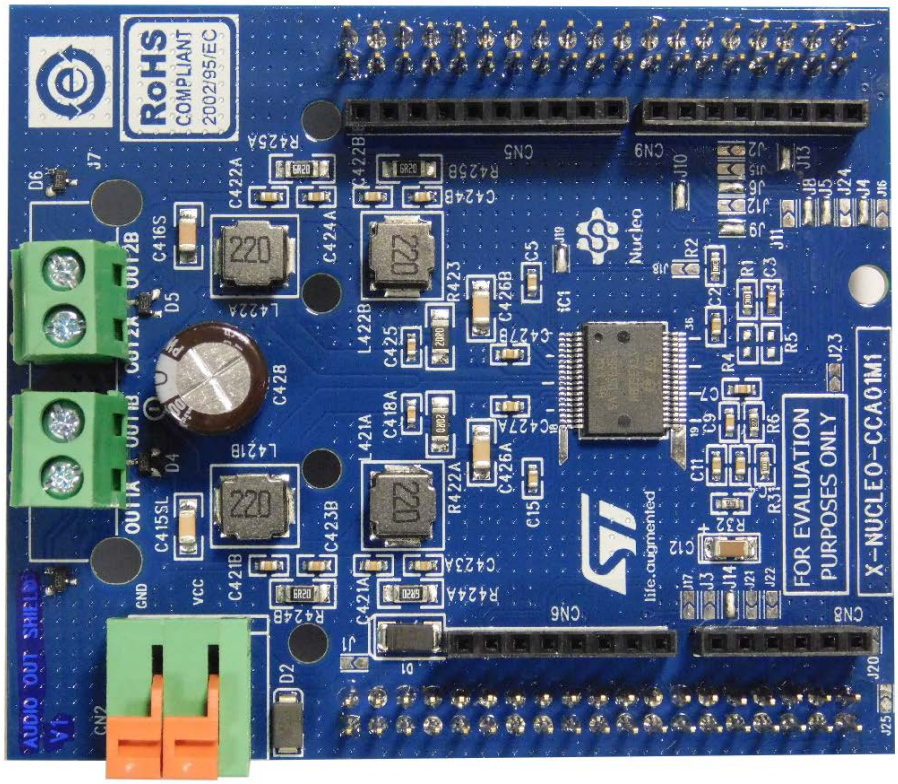
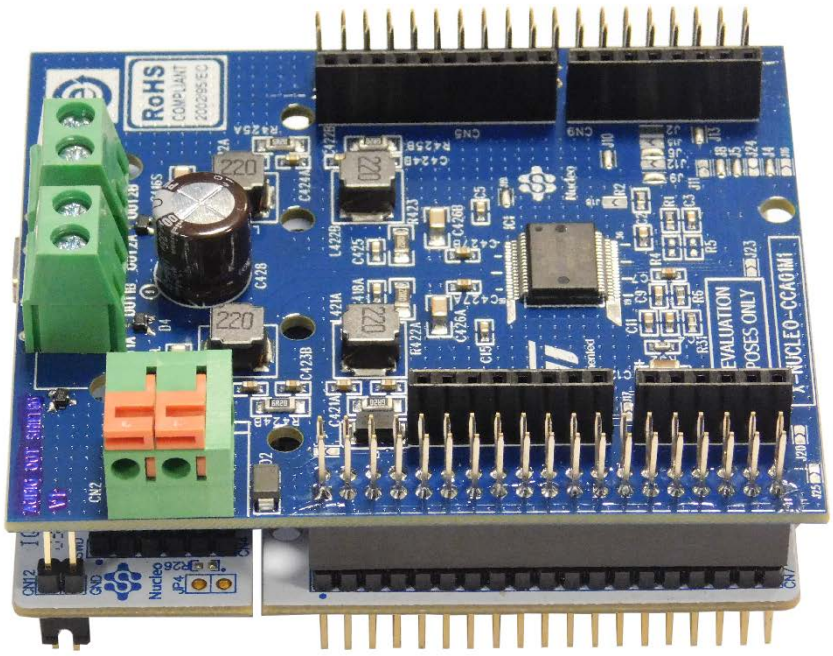


Figure 6. **Sound terminal expansion board connected to STM32 Nucleo board**



4.2 Software description

The following software components are required in order to set up a suitable development environment for creating applications using the STM32 Nucleo equipped with a Sound Terminal® expansion board:

- X-CUBE-SOUNDTER1: an expansion for STM32Cube dedicated to audio output applications development. The X-CUBE-SOUNDTER1 firmware and related documentation is available on st.com.
- Development tool-chain and Compiler: The STM32Cube expansion software supports the following environments:
  1. IAR Embedded Workbench for ARM® (IAR-EWARM) toolchain + ST-LINK
  2. RealView Microcontroller Development Kit (MDK-ARM-STM32) toolchain + ST-LINK
  3. System Workbench for STM32 (SW4STM32) + ST-LINK

4.3 Hardware and software setup

This section the initial installation of the abovementioned hardware and software components and the subsequent setup procedures.

4.3.1 Hardware setup

The following hardware components are required:

1. One Sound Terminal® expansion board (order code: X-NUCLEO-CCA01M1), designed to allow up to two X-NUCLEO-CCA01M1 expansion boards plugged on the one STM32 Nucleo host, configuring the solder bridges accordingly.
2. One STM32 Nucleo development platform (suggested order code: NUCLEO-F401RE). This software expansion also supports the NUCLEO-F072RB, NUCLEO-L476RG, NUCLEO-F746ZG and NUCLEO-L053R8 boards. In the latter case, only one X-NUCLEO-CCA01M1 expansion board can be fitted due to the lack of a second I²S interface on the STM32L053R8 MCU.
3. One USB type A to Mini-B USB cable to connect the STM32 Nucleo to the PC.
4. At least one 8 Ω passive speaker connected to the X-NUCLEO-CCA01M1 expansion board. Two speakers are obviously required in order to appreciate stereo audio reproduction.
5. An external power supply from +5 V to +26 V. For powering options, please refer to the X-NUCLEO-CCA01M1 user manual available at http://www.st.com/x-nucleo.

4.3.2 Jumper configurations

You can connect two X-NUCLEO-CCA01M1 expansion boards to the same STM32 Nucleo host in order to build a 4-channel audio system. In this scenario, each sound terminal expansion board must act as either “Device number one” or “Device number two”. This differentiation and the STM32 Nucleo board used determine the correct solder bridge configuration.

Note that for each hardware configuration, the correct firmware initialization parameters must be used. The respective settings for both use cases are described below.

Table 1. Board acting as first device

	STM32F401RE or STM32F746ZG	STM32F072RB	STM32L053R8	NUCLEO-L476RG
J1	OPEN	OPEN	OPEN	OPEN
J2	OPEN	OPEN	OPEN	OPEN
J3	OPEN	OPEN	OPEN	OPEN
J4	CLOSE	OPEN	OPEN	CLOSE
J5	CLOSE	CLOSE	CLOSE	CLOSE
J6	CLOSE	CLOSE	CLOSE	CLOSE
J8	CLOSE	CLOSE	CLOSE	CLOSE

	STM32F401RE or STM32F746ZG	STM32F072RB	STM32L053R8	NUCLEO-L476RG
J9	CLOSE	CLOSE	CLOSE	CLOSE
J10	CLOSE	CLOSE	CLOSE	CLOSE
J11	OPEN	OPEN	OPEN	OPEN
J12	OPEN	OPEN	OPEN	OPEN
J13	CLOSE	CLOSE	CLOSE	CLOSE
J14	CLOSE	CLOSE	CLOSE	OPEN
J15	OPEN	OPEN	OPEN	OPEN
J16	OPEN	OPEN	OPEN	OPEN
J17	OPEN	OPEN	OPEN	OPEN
J18	OPEN	OPEN	OPEN	OPEN
J19	CLOSE	CLOSE	CLOSE	CLOSE
J20	OPEN	OPEN	OPEN	OPEN
J21	OPEN	OPEN	OPEN	OPEN
J22	OPEN	OPEN	OPEN	OPEN
J23	OPEN	CLOSE	CLOSE	OPEN
J24	OPEN	OPEN	OPEN	OPEN
J25	OPEN	OPEN	OPEN	OPEN

Table 2. **Board acting as second device**

	STM32F401RE, STM32F746ZG	STM32F072RB	STM32L053R8	NUCLEO-L476RG
J1	OPEN	OPEN	NOT SUPPORTED	OPEN
J2	CLOSE	CLOSE		CLOSE
J3	CLOSE	OPEN		CLOSE
J4	OPEN	OPEN		OPEN
J5	OPEN	OPEN		OPEN
J6	OPEN	OPEN		OPEN
J8	OPEN	OPEN		OPEN
J9	CLOSE	CLOSE		CLOSE
J10	CLOSE	CLOSE		CLOSE
J11	OPEN	CLOSE		OPEN
J12	OPEN	CLOSE		OPEN
J13	OPEN	OPEN		OPEN
J14	OPEN	OPEN		OPEN
J15	OPEN	OPEN		OPEN
J16	CLOSE	OPEN		CLOSE
J17	OPEN	OPEN		CLOSE
J18	CLOSE	CLOSE		CLOSE
J19	OPEN	OPEN		OPEN
J20	OPEN	OPEN		OPEN
J21	CLOSE	CLOSE		OPEN
J22	CLOSE	CLOSE		CLOSE
J23	OPEN	OPEN		OPEN
J24	OPEN	CLOSE		OPEN
J25	CLOSE	OPEN		OPEN

4.3.3 Software setup

This section lists the minimum requirements for the developer to setup the SDK, run the sample testing scenario based on previous description and customize applications.

Development Tool-chains and Compilers

Please select one of the Integrated Development Environments supported by the STM32Cube expansion software and read the system requirements and setup information provided by the same IDE provider.

4.3.4 System setup guide

This section describes how to setup different hardware parts before writing and executing an application on the STM32 Nucleo board with the sound terminal expansion board.

STM32 Nucleo and sound terminal expansion boards setup

The STM32 Nucleo board integrates the ST-LINK/V2-1 debugger/programmer. You can download the relevant version of the ST-LINK/V2-1 USB driver by searching for STSW-LINK009 on www.st.com (according to your version of Microsoft Windows).

The X-NUCLEO-CCA01M1 microphone expansion board is easily connected to the STM32 Nucleo motherboard through the ST morpho extension connector, see Figure 4. The sound terminal expansion board is capable of interfacing with the external STM32 microcontroller on STM32 Nucleo via the I²C and I²S or SAI communication interfaces.

Revision history

Table 3. **Document revision history**

Date	Version	Changes
09-Nov-2015	1	Initial release.
15-Jul-2016	2	Added new board compatibility; Added column to Table 1. Board acting as first device and to Table 2. Board acting as second device.
18-Apr-2018	3	Updated Figure 2. X-CUBE-SOUNDTER1 software architecture.

Contents

List of tables

List of figures