UM2912

User manual

STM32CubeU5 STM32U575I-EV demonstration firmware

Introduction

STM32Cube is an STMicroelectronics original initiative to significantly improve designer's productivity by reducing development effort, time, and cost. STM32Cube covers the whole STM32 portfolio.

STM32Cube includes:

- A set of user-friendly software development tools to cover project development from conception to realization, among which are:
  - STM32CubeMX, a graphical software configuration tool that allows the automatic generation of C initialization code using graphical wizards
  - STM32CubeIDE, an all-in-one development tool with peripheral configuration, code generation, code compilation, and debug features
  - STM32CubeProgrammer (STM32CubeProg), a programming tool available in graphical and command-line versions
  - STM32CubeMonitor (STM32CubeMonitor, STM32CubeMonPwr, STM32CubeMonRF, STM32CubeMonUCPD) powerful monitoring tools to fine-tune the behavior and performance of STM32 applications in real-time
- STM32Cube MCU and MPU Packages, comprehensive embedded-software platforms specific to each microcontroller and microprocessor series (such as STM32CubeU5 for the STM32U5 Series), which include:
  - STM32Cube hardware abstraction layer (HAL), ensuring maximized portability across the STM32 portfolio
  - STM32Cube low-layer APIs, ensuring the best performance and footprints with a high degree of user control over hardware
  - A consistent set of middleware components such as FAT file system, RTOS, OpenBootloader, USB Host, USB Device, and USB Power Delivery
  - All embedded software utilities with full sets of peripheral and applicative examples
- STM32Cube Expansion Packages, which contain embedded software components that complement the functionalities of the STM32Cube MCU and MPU Packages with:
  - Middleware extensions and applicative layers
  - Examples running on some specific STMicroelectronics development boards

The STM32CubeU5 Evaluation board demonstration firmware is built around almost the whole STM32 capability to offer a large scope of usage based on the STM32Cube HAL BSP and utility components (mainly the LPBAM).

The architecture was defined to make from the STM32CubeU5 demonstration core an independent central component that can be used with several RTOS and third-party firmware libraries through several abstraction layers inserted between the STM32CubeU5 demonstration core and the several modules and libraries working around.

The STM32CubeU5 Evaluation board demonstration firmware supports STM32U5xx devices and runs on the STM32U575I-EV Evaluation board.

1 General information

The STM32CubeU5 demonstration firmware runs on the STM32U575I-EV Evaluation board featuring the STM32U575AI microcontroller based on the Arm® Cortex®-M33 core with Arm® TrustZone®.

Table 1 lists the acronyms and abbreviations used in this document.

Table 1. Definition of terms

Term	Definition
ADF	Audio-digital filter
DMA	Direct memory access
GTZC	Global TrustZone® controller
LPBAM	Low-power background autonomous mode
LPDMA	Low-power direct memory access
LPTIM	Low-power timer
MDF	Multi-function digital filter
MPCBB	Block-based memory protection controller
TZSC	TrustZone® security controller
USB-PD	USB power delivery

Note: Arm and TrustZone are registered trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere.

2 STM32Cube overview

STM32Cube is an STMicroelectronics original initiative to significantly improve designer's productivity by reducing development effort, time, and cost. STM32Cube covers the whole STM32 portfolio.

STM32Cube includes:

- A set of user-friendly software development tools to cover project development from conception to realization, among which are:
  – STM32CubeMX, a graphical software configuration tool that allows the automatic generation of C initialization code using graphical wizards
  – STM32CubeIDE, an all-in-one development tool with peripheral configuration, code generation, code compilation, and debug features
  – STM32CubeProgrammer (STM32CubeProg), a programming tool available in graphical and command-line versions
  – STM32CubeMonitor (STM32CubeMonitor, STM32CubeMonPwr, STM32CubeMonRF, STM32CubeMonUCPD) powerful monitoring tools to fine-tune the behavior and performance of STM32 applications in real-time
- STM32Cube MCU and MPU Packages, comprehensive embedded-software platforms specific to each microcontroller and microprocessor series (such as STM32CubeU5 for the STM32U5 Series), which include:
  – STM32Cube hardware abstraction layer (HAL), ensuring maximized portability across the STM32 portfolio
  – STM32Cube low-layer APIs, ensuring the best performance and footprints with a high degree of user control over hardware
  – A consistent set of middleware components such as ThreadX, FileX / LevelX, USBX, NetX Duo, USB power delivery, TF-M, mbed-crypto, Touch library, Network library, OpenBootloader
  – All embedded software utilities with full sets of peripheral and applicative examples
- STM32Cube Expansion Packages, which contain embedded software components that complement the functionalities of the STM32Cube MCU and MPU Packages with:
  – Middleware extensions and applicative layers
  – Examples running on some specific STMicroelectronics development boards

Within STM32CubeU5, both the HAL and LL APIs are production-ready, checked with CodeSonar® static analysis tool, and developed in compliance with MISRA C® guidelines, following a process certified according to IEC 61508 systematic capability 2 level (SC2). Reports are available on demand.

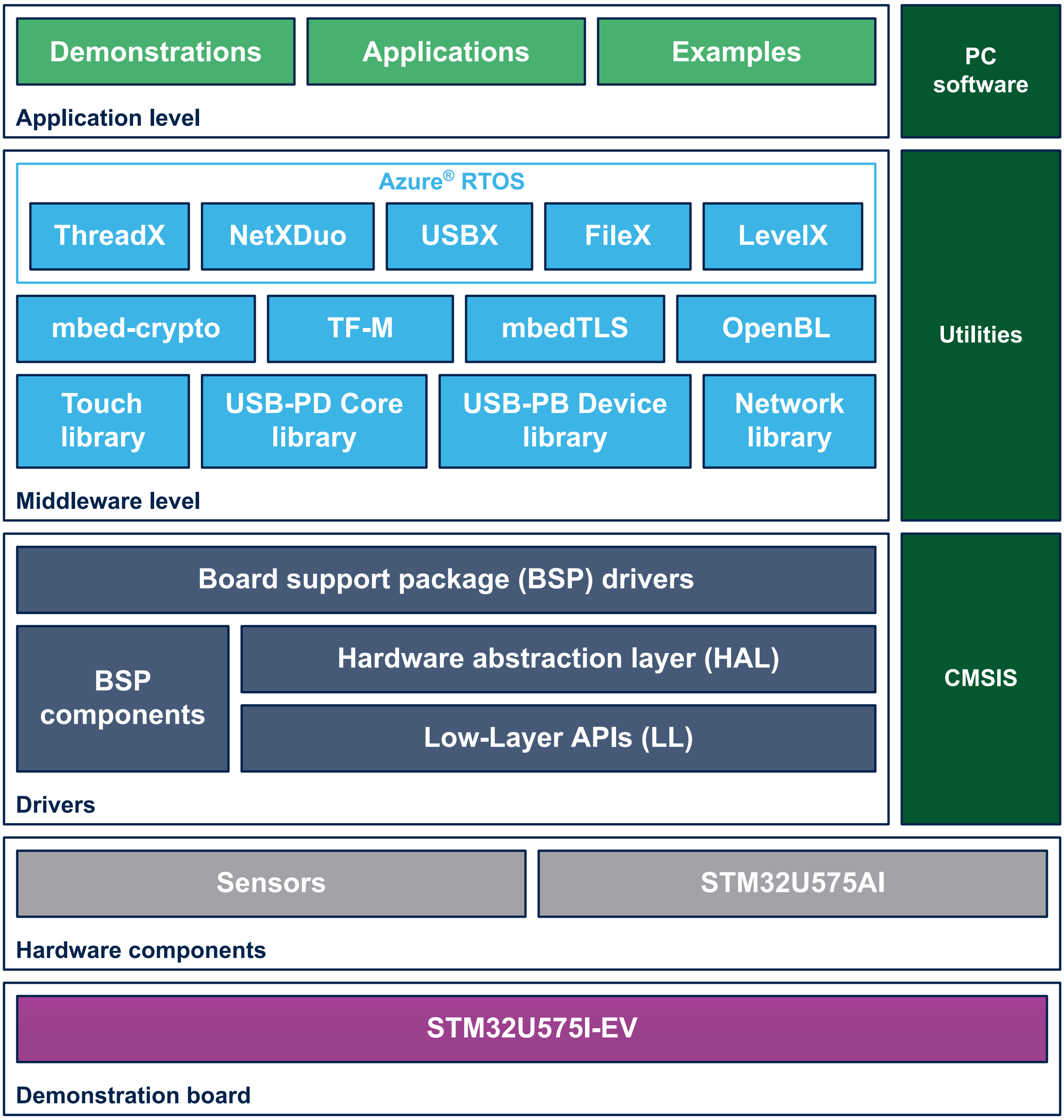


Figure 1. STM32CubeU5 MCU Package architecture

3 Global architecture

The STM32CubeU5 Evaluation board demonstration firmware is composed of a central kernel based on a set of firmware and hardware services offered by the STM32Cube MCU Package and a set of modules mounted on the kernel and built in a modular architecture. Each module can be reused separately in a standalone application. The full set of modules is managed by the kernel which provides access to all common resources and facilitates the addition of new modules.

Each module must provide the following functionalities and proprieties:

1. Icon and graphical aspect characteristics
2. Method to start up the module
3. Method to close safety the module
4. Method to manage low power mode
5. The module application core (Main module process)
6. Specific configuration
7. Error management

4 STM32CubeU5 STM32U575I-EV demonstration functional description

4.1 Main graphical interface

Overview

The main graphical interface presents a principal view of the demonstration.

It offers the user to select the application to be launch. There is the possibility to launch the “LPBAM”, “Trust Zone”,” Digital Filters”,” Low Power”,” Camera”,” Calendar” applications or display the system information.

Contents

The graphical interface contains eight buttons to select by the user:

- LPBAM (Low Power Background Autonomous Mode)
- Trust Zone (Security peripheral/memory control access)
- Digital Filters (Sound Activity detector, Audio filters, Power metering)
- USB-PB application (Delivered separately in the STM32Cube Firmware package)
- Low Power Module (Different low power modes)
- Camera Viewer Module
- Calendar Modules (Date/Time/ Alarm)
- System information

Figure 2. **Welcome screen**

Figure 3. **Main graphical interface**

Pictures are not contractual.

4.2 Demonstration modules

4.2.1 LPBAM modules

LPBAM (Low-power background autonomous mode) innovative autonomous power mode saves power by enabling direct memory access (LPDMA low-power direct memory access) and ensuring the peripherals keep working, while most of the device is in stop mode.

This part presents two examples of applications showing how it works.

- The first module concerns data acquisition from an I^2C motion sensor (Gyroscope and accelerometer).
- The second module is a low-frequency signal generator.

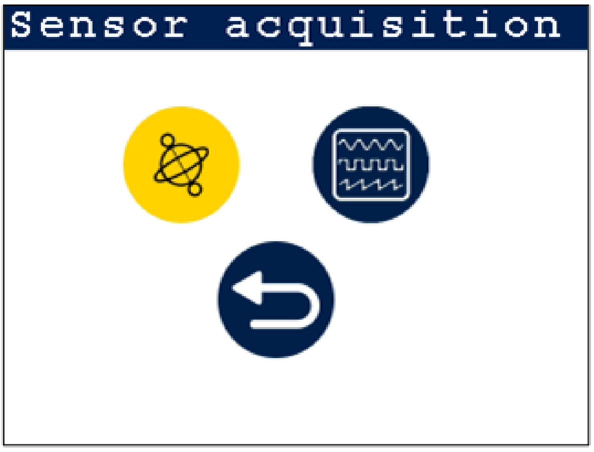


Figure 4. LPBAM modules (Sensor acquisition)

4.2.1.1 *Sensor acquisition*

Here is the example of an application based on LPBAM utilities that shows how data acquisition can be performed from an I^2C motion sensor (Gyroscope and accelerometer).

The application is organized as follows:

1. In Run mode, we should prepare the acquisition scenario step:
   a. Initialize the sensor device.
   b. Prepare DMA nodes to perform the configuration and data transfer.
   c. Configure the low-power timer (LPTIM) which presents the trigger for each acquisition (Acquisition frequency).
2. Start the low-power direct memory access (LPDMA) and enter Stop mode.
3. In Stop mode, the system acquires the sensor data and saves them in the internal SRAM.
4. After the wakeup using the joystick, the sensor data saved during the Stop mode is plotted in a graph as shown in Figure 6.

Figure 5. Sensor acquisition control

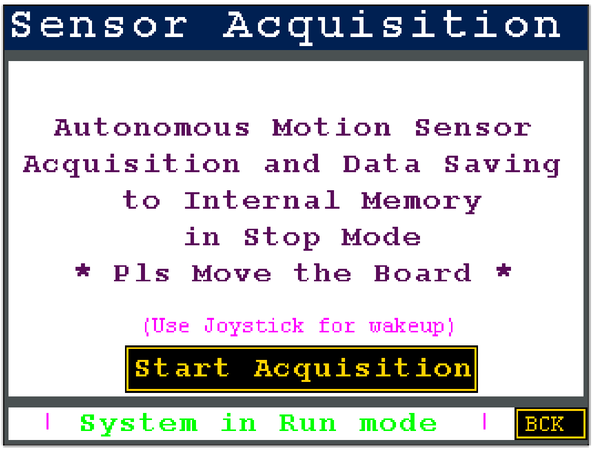
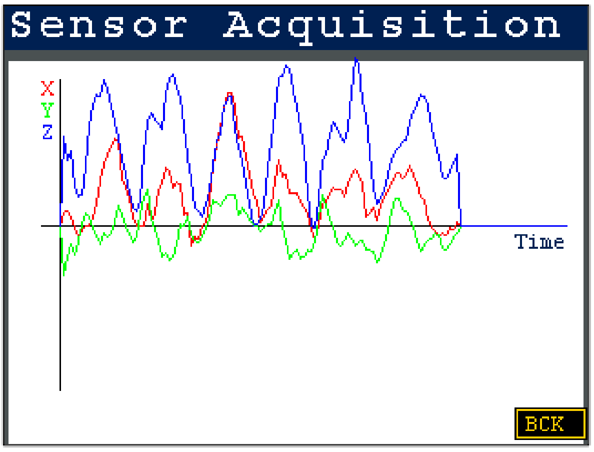


Figure 6. Sensor acquisition result graph



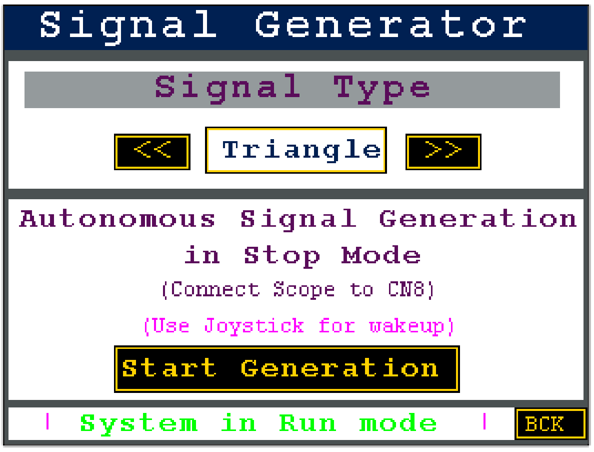
4.2.1.2 *Signal generator*

This section presents the example of an application based on LPBAM utilities that shows how a signal can be generated with a personalized waveform in Stop mode and without MCU intervention.

The application is organized as follows:

1. In Run mode, the signal generation scenario step is prepared:
   a. Initialize the DAC peripheral.
   b. Prepare DMA nodes to perform the configuration and data transfer.
   c. Configure the low-power timer (LPTIM) which provides the conversion frequency (Signal frequency).
2. Start the low-power direct memory access (LPDMA) and enter Stop mode.
3. In Stop mode, the system transfers signal waveform data to DAC peripheral.
4. After the wakeup using the joystick, the generation stops, and another type of signal is configured before being generated.

Figure 7. Signal generator configuration interface



4.2.2 TrustZone® controller modules

The global TrustZone® controller (GTZC) block contains the following sub-blocks:

1. TZSC (TrustZone® security controller): This sub-block defines the secure privileged state of slave peripherals. It also controls the sub-region area size and properties for the watermark memory peripheral controller (MPCWM). The TZSC informs some peripherals (such as RCC or GPIOs) about the security status of each securable peripheral, by sharing with RCC and I/O logic.
2. MPCBB (Block-based memory protection controller) This sub-block configures the internal RAM in a TrustZone®-system product having segmented SRAM (512-byte pages) with programmable security and privileged attributes.

In this context, two applications are offered in emulation mode to highlight these features (Peripheral access control and memory access control).

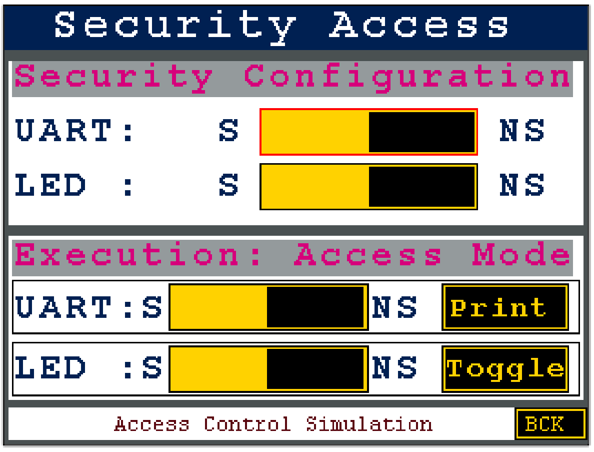
Figure 8. TrustZone® controller modules

4.2.2.1 *Peripheral access control*

- The TrustZone® security controller defines the secure privileged state of slave peripherals. Here is an example simulating the behavior of LED (GPIO) and UART peripherals in a security context controlled by the GTZC.

It shows how a peripheral can be configured and provides the expected results during access to the peripheral in the security configuration.

Figure 9. GTZC peripheral access interface



As shown in Figure 9, the user can configure in the configuration window the UART and the LED as secured or non-secured peripherals.

Then, in the execution window, the user sets the access type, that defines in which access mode the user performs the execution (Secure or non-secure access mode).

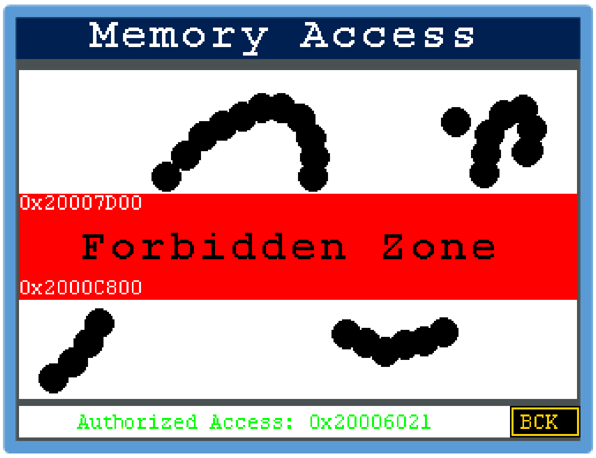
Finally, the user can check the result of the execution by pressing the button `Print` for the UART or `Toggle` for the LED.

4.2.2.2 *Memory access control*

The GTZC contains an important sub-block for memory protection.

MPCBBblock-based memory protection controller: This sub-block configures the internal RAM in a TrustZone®-system product having segmented SRAM (512-byte pages) with programmable security and privileged attributes.

Figure 10. GTZC memory access interface



Based on the MPCBB block, this example shows how the GTZC can protect a specific memory region. The example provides an on-screen drawing application using the finger. This is a frame buffer memory writing in the corresponding address in the memory.

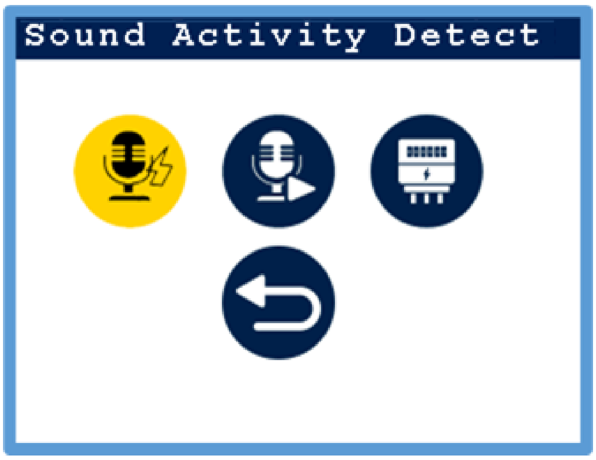
If the users touch with their fingers an area corresponding to the normal zone, a dot is drawn and a message displayed at the bottom of the screen mentioning that access to this address is authorized. Now, if the users touch with their fingers an area corresponding to the forbidden zone, nothing is drawn and a message is displayed at the bottom of the screen mentioning that access to this address is not allowed.

4.2.3 Multi-function digital filters modules

The multi-function digital filter (MDF) is a high-performance module dedicated to the connection of external sigma-delta modulators. It is mainly targeted for the following applications: audio, capture signals, motor control, and metering.

In this demonstration, three applications are provided: The first one is sound activity detection, the second one is an audio playback with different filtering modes, and the third one is a power metering module.

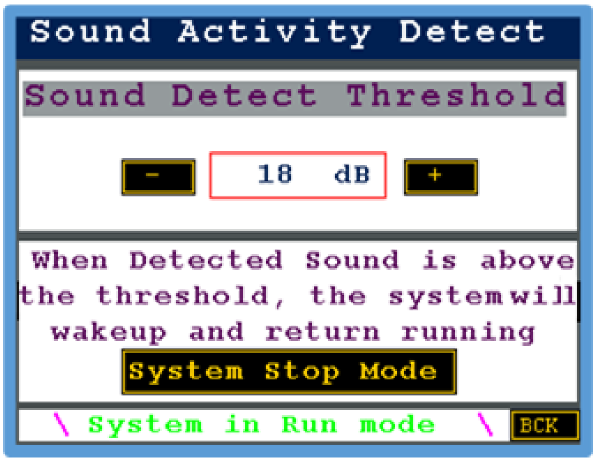
Figure 11. Multi-function digital filter module interface



4.2.3.1 *Sound activity detector*

A sound activity detector (SAD) is available for the detection of sounds or voice signals. The SAD is a feature of ADF peripheral. Several parameters can be programmed to adjust properly the SAD to the sound environment. The SAD strongly reduces the power consumption by preventing the storage of samples into the system memory, if the observed signal does not match the programmed criteria.

Figure 12. Sound activity detector



In this application, the SAD is configured to detect sound coming from the microphone, analyze the sound data after filtering, and compare them to the defined threshold to use for the system wakeup.

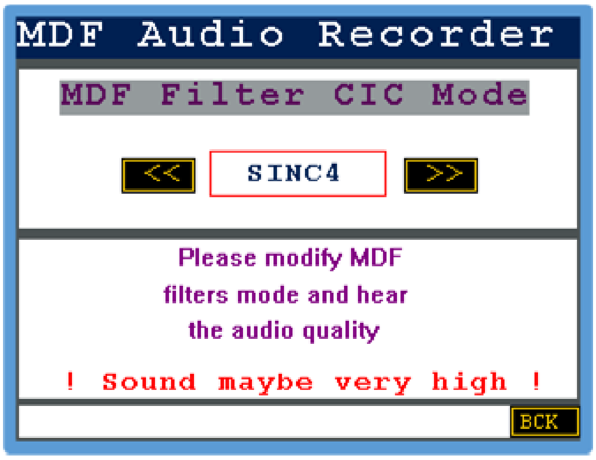
The user selects the desired threshold depending on the environment and the use case. Then he puts the system in Stop mode.

Once a detected sound is above the defined threshold, the system wakes up and returns to the Run mode.

4.2.3.2 *Audio filters*

The audio digital filter (ADF) is a high-performance module dedicated to the connection of external sigma-delta modulators. It targets mainly audio applications.

Figure 13. MDF audio recorder interface



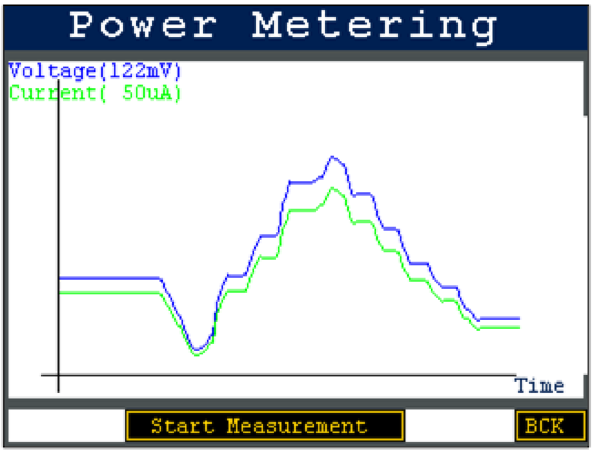
This module shows the capabilities of MDF and ADF peripherals to filter in different ways audio signals captured by a MEMS microphone. The audio is then played and heard via an earphone connected to the CN27 audio jack connector. The user can select the Filtering mode and hear the different audio quality. The different filtering modes must be used according to the user's use case.

4.2.3.3 *Power metering*

This application shows how the MDF can be used for power-metering purposes and gives the variation of the voltage and current for the user circuit. The example plots a power consumption real-time measurement (Voltage and current) of the circuit connected to the RV1 potentiometer.

The user must connect pin 2 of JP1 to pin 2 of CN24 (JP30 must be ON). To modify the power consumed by the circuit, the user can rotate the RV1 potentiometer and directly check the variation in the graph on the screen.

Figure 14. **Power metering graph**



4.2.4 USB-PD module

USB-PD is delivered in a separate application that shows the USB-PD features included in the STM32U5 Series microcontroller.

4.2.5 Low-power module

This application mainly shows the different low-power modes supported by the STM32U5 Series device (Sleep, Stop1, Stop2, Stop3, Standby, and Shutdown).

After entering the selected low-power mode, the user can measure the power consumption (the current) using the IDD jumper.

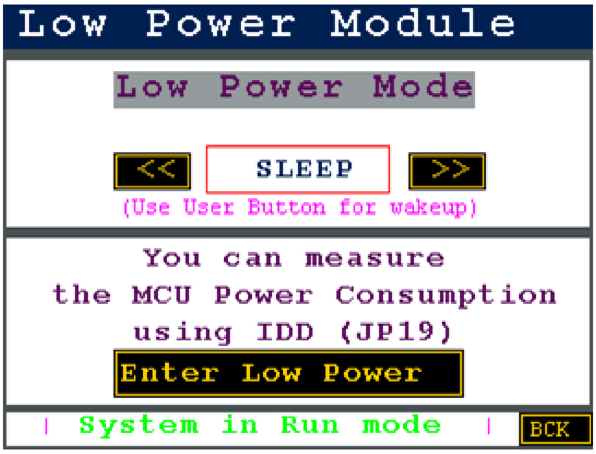


Figure 15. **Low-power interface in Sleep mode**

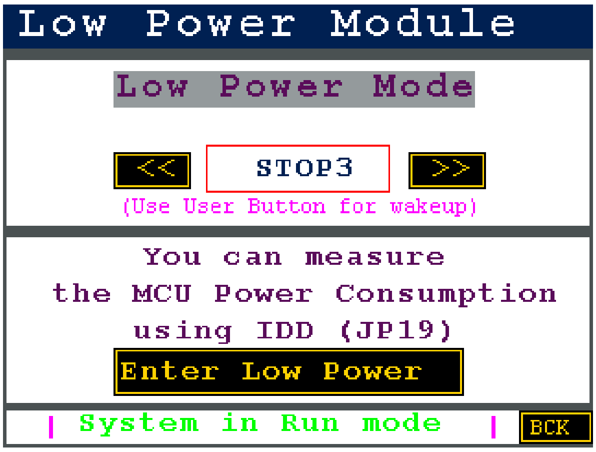


Figure 16. **Low-power interface in Stop3 mode**

4.2.6 Camera module

This module presents the capability of STM32U5 devices to interface with the camera sensor using the DCMI peripheral to do picture capture in Snapshot or Continuous mode. Then the captured pictures can be shown on the LCD screen and transferred using communication peripherals, such as SPI, USART, or USB.

Figure 17. Camera picture display

4.2.7 Calendar module

This module shows the capabilities of the RTC peripheral to provide calendar functionalities:

- Set and show the date.
- Set and show the time.
- Set the Alarm (Time and date)

Figure 18. **Calendar main menu**

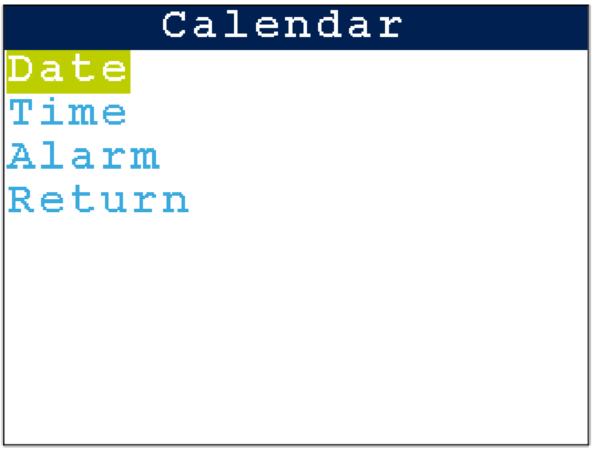


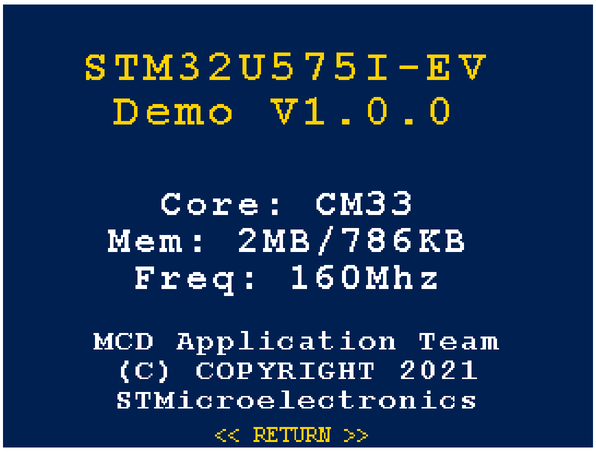
Figure 19. **Date configuration interface**

Figure 20. **Alarm configuration interface**

4.2.8 System information

The system information shows the main demonstration information such as the used board, the STM32CubeU5 demonstration firmware version, the core type, the Flash memory and SRAM sizes, and the MCU clock.

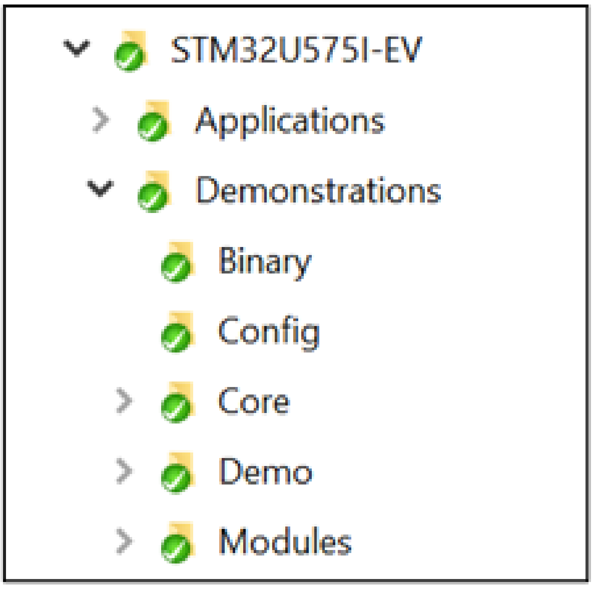
Figure 21. System information window



4.3 Demonstration repository

The STM32Cube is a component in the STM32Cube MCU Package. The following figure shows the demonstration folder organization:

Figure 22. Demonstration folder organization



The repository is composed as follows:

- Binaries folder contains the demonstration binaries to be programmed in the target internal Flash memory.
- Config folder contains the configuration header files.
- Core folder contains the source and header files for the demonstration kernel that manage the demonstration execution based on modules.
- Demo folder mainly contains the IDE project files.
- Modules folder contains the source and header files of demonstration modules.

Revision history

Table 2. **Document revision history**

Date	Revision	Changes
5-Oct-2021	1	Initial release.

Contents

List of tables

List of figures