STM32CubeH5 STM32H573I-DK demonstration firmware

Introduction

STM32Cube is an STMicroelectronics original initiative to improve designer productivity significantly 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
    - STM32CubeCLT, an all-in-one command-line development toolset with code compilation, board programming, 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 STM32CubeH5 for the STM32H5 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 STM32CubeH5 Discovery 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.

The architecture is defined to make from the STM32CubeH5 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 STM32CubeH5 demonstration core and the several modules and libraries working around.

The STM32CubeH5 Discovery board demonstration firmware supports STM32H5xx devices and runs on the STM32H573I-DK Discovery board.

UM3126 - Rev 1 - July 2023
For further information contact your local STMicroelectronics sales office.

www.st.com

Within STM32CubeH5, 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. STM32CubeH5 MCU Package architecture
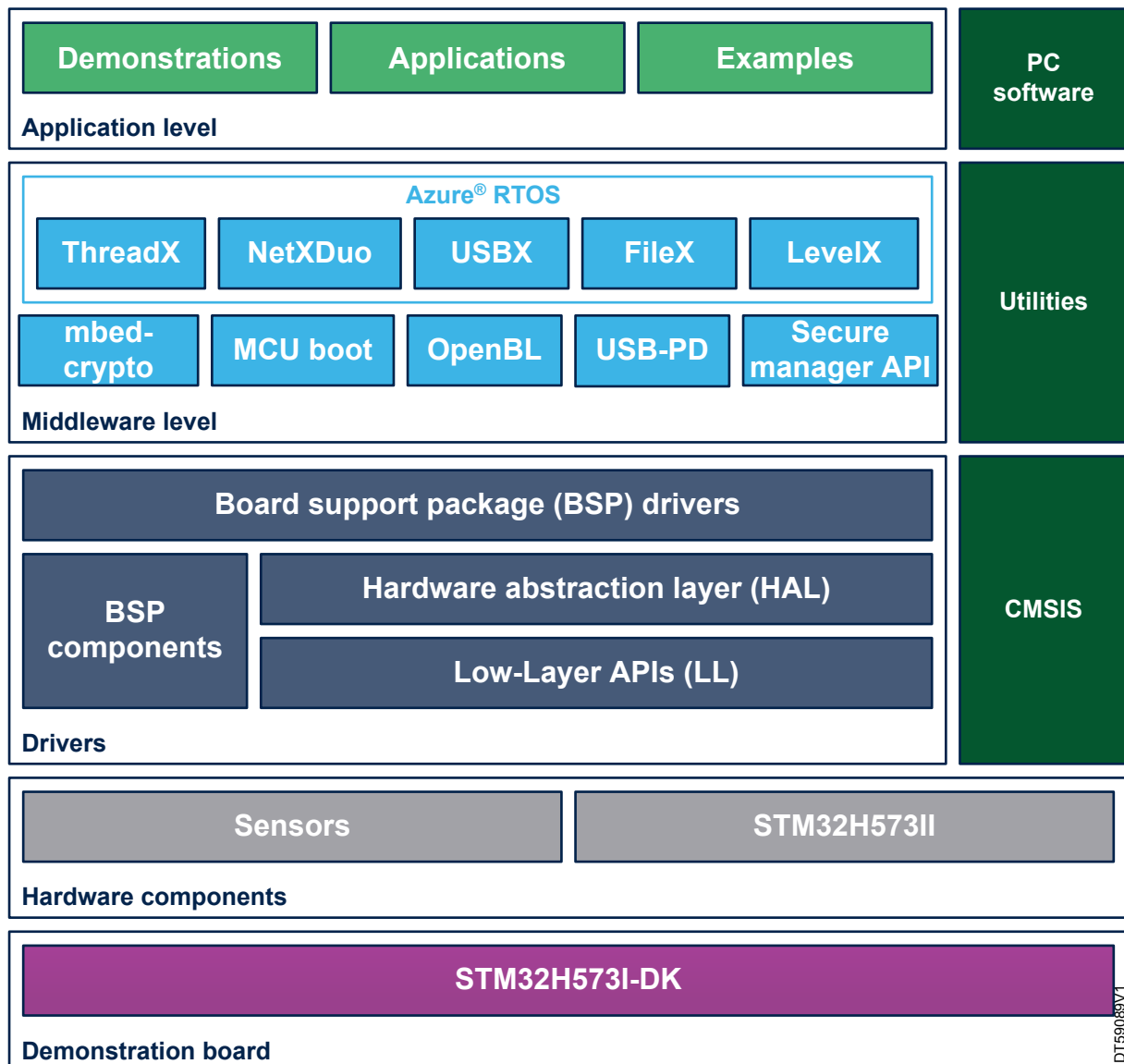
1   General information

The STM32CubeH5 demonstration firmware runs on the STM32H573I-DK Discovery board featuring the STM32H573II 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
DMA	Direct memory access
GTZC	Global TrustZone® controller
OCTOSPI	Octo-SPI interface
OTFDEC	On-the-fly decryption engine
RTC	Real-time clock
SAI	Serial audio interface
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.*

arm

2 Global architecture

The STM32CubeH5 demonstration firmware is composed of two subdemonstrations accessible from a main menu launcher:

1. The TouchGFX graphics demonstration showing the TouchGFX framework capabilities of STM32H573I-DK. The demo is running in 16-bit color mode, with double frame buffers in internal SRAM and all assets stored in internal flash memory. To access the project code and reflash the TGFX subdemo, the user must use TouchGFXDesigner, which is available as a free download on st.com, refer to as X-CUBE-TOUCHGFX.

2. STM32H573I demonstration: 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 kernel, which provides access to all common resources and facilitates the addition of new modules, manages the full set of modules.

   Azure® RTOS ThreadX is used for tasks scheduling and hence the switching between the different demonstration modules and submodules.

   Each module must provide the following functionalities and proprieties:

   a. Icon and graphical aspect characteristics

   b. Method to start up the module

   c. Method to safely close the module

   d. The module application core (main module process)

   e. Specific configuration

   f. Error management

   To build this demonstration separately, refer to the STM32Cube_FW_H5_V1.1.0 or later.

Note: *A touchscreen controls the demonstration navigation.*

Note: *Going back to the main menu the user needs to use the black reset button on the board.*

3 Demonstration functional description

3.1 Main graphical interface

The main graphical interface presents a menu launcher to select one of the two subdemonstrations: the TouchGFX demo and STM32H573I-DK demo.

Figure 2. Menu launcher

Picture is not contractual.

3.2 TouchGFX demo

Overview

Graphical menu to access four kinds of graphical user interface effects/applications.

Contents

The graphical interface contains four modules selected by the user:
- Basement control
- Light effect
- Blend effect
- Weather overview

Figure 3. TouchGFX main graphical menu

3.2.1 Basement control

UI shows a scroll list consisting of text and temperature values set in the schedule and temperature settings.

Figure 4. TouchGFX basement control

3.2.2 Light effect

UI animation shows the graphics capabilities of TouchGFX and STM32H573xx devices with a light effect moving smoothly around the screen.

Figure 5. TouchGFX light effect

3.2.3 Blend effect

UI shows the alpha blending of multiple layers with TouchGFX on STM32H573xx devices, which is all done with software rendering.

Figure 6. TouchGFX blend effect
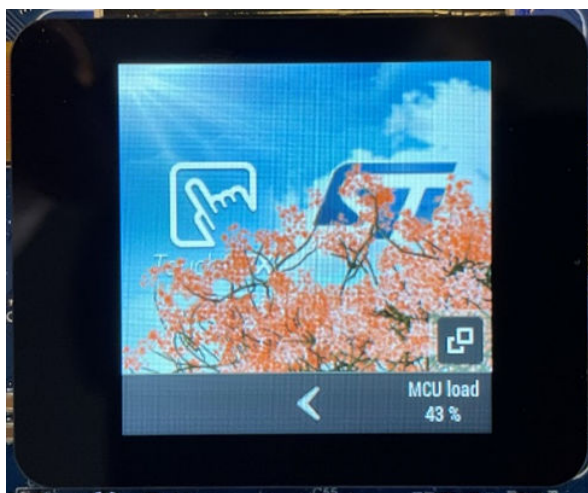
3.2.4 Weather overview

UI shows full-screen updates by sliding between different screens.

Figure 7. TouchGFX weather overview

3.3 STM32H573I-DK demo

Overview

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

It offers the user to select the application to launch through a touchscreen. There is the possibility to launch the *Trust Zone*, *Audio Rec/Play*, *USB-PD*, *Calendar*, and *Cryptography* applications or display the system information.

Contents

The graphical interface contains six modules selected by the user:

- Trust Zone (security peripheral/memory control access)
- USB-PB application (dual role power and dual role data)
- Calendar modules (Date/Time/Alarm)
- Cryptography module (encrypt and decrypt code from external memory)
- System information

Figure 8. STM32H573I-DK demo welcome screen

Figure 9. STM32H573I-DK demo main graphical interface
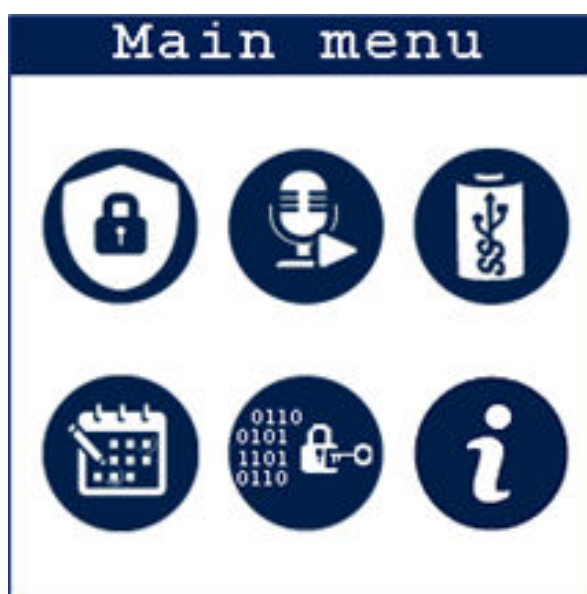
3.3.1 TrustZone® controller modules

Overview

TrustZone® technology within Arm® Cortex®-M33 is an efficient solution to protect data and code.

It divides the system into two worlds, secure(trusted) and nonsecure (nontrusted). The Trust Zone controller (GTZC) encompasses peripherals, memory, software, interrupts, and bus transactions.

Contents

The global TrustZone® controller (GTZC) block contains the following subblocks:

1. TZSC (TrustZone® security controller): This subblock defines the secure privileged state of the slave peripherals. It also controls the subregion area size and properties of 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 subblock 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 10. TrustZone® controller modules

3.3.1.1 Peripheral access control

- The TrustZone® security controller defines the secure privileged state of the slave peripherals. Here is an example that simulates 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 11. GTZC peripheral access interface
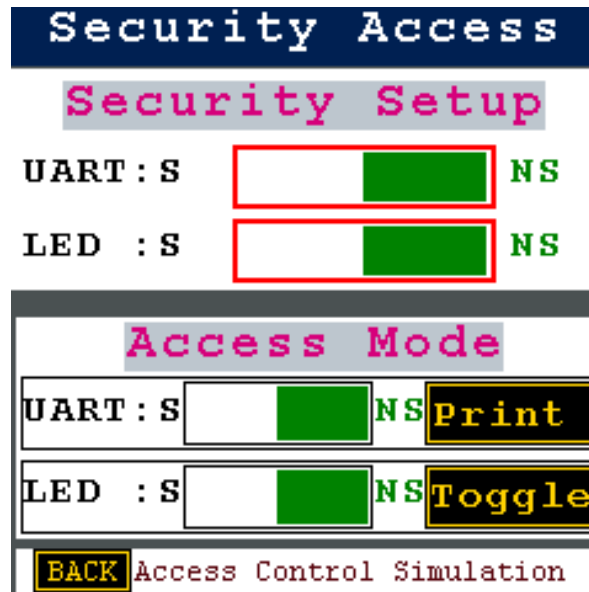
As shown in Figure 11. GTZC peripheral access interface, the user can configure in the configuration window the UART and the LED as secured or nonsecured peripherals.

Then, in the execution window, the user sets the access type that defines in which access mode the user performs the execution (secure or nonsecure 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.

3.3.1.2 Memory access control

The GTZC contains an important subblock for memory protection.

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

Figure 12. GTZC memory access interface
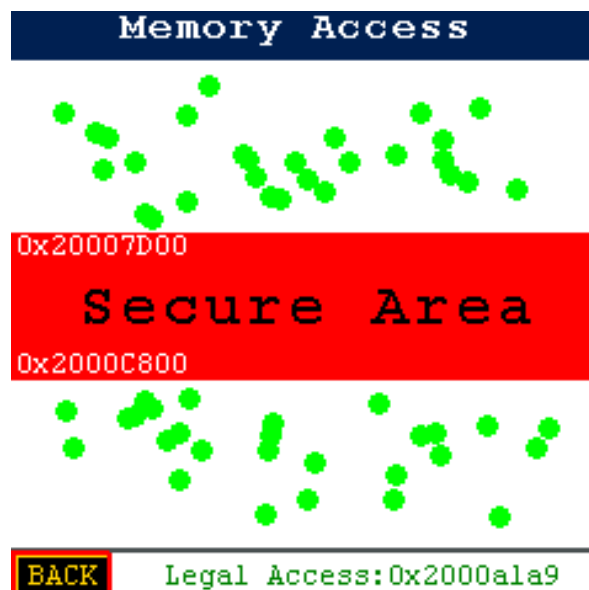
Based on the MPCBB block, this example shows how the GTZC can protect a specific memory region. The example provides an on-screen drawing application using the finger. This is a frame buffer memory writing in the corresponding address in the memory.

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.

3.3.2 Audio module

Overview

The serial audio interface (SAI) is a peripheral communicating with external audio devices supporting the most audio standards allowing an easy connection to any existing audio device.

Integrating a PDM interface, the SAI can connect up to eight digital microphones for beamforming or simple speech capture applications.

Contents

The SAI1 is configured to record PDM data from a digital microphone mounted on the board.

The STM32_Audio library is used to convert the PDM data into PCM audible data. The PCM data is played via audio codecs CS42L51 and SAI2.

Connected to an audio codec, the SAI is configured to:

- Receive and store data from a digital microphone in DMA mode.
- Transmit the received data for playback.

The module provides the options to start, stop, pause, and resume the recording and playback.

For audio playback, the user must plug a headset into the audio jack.

Figure 13. Audio recorder interface

3.3.3 USB-PD module

This module demonstrates the functionality of the USB-PD peripheral. The board can behave as a USB DeviceDevice or USB Host.

When a device is connected via a USB Type-C® cable, the board behaves as a mouse.

By tapping on the triangles, the mouse cursor moves.
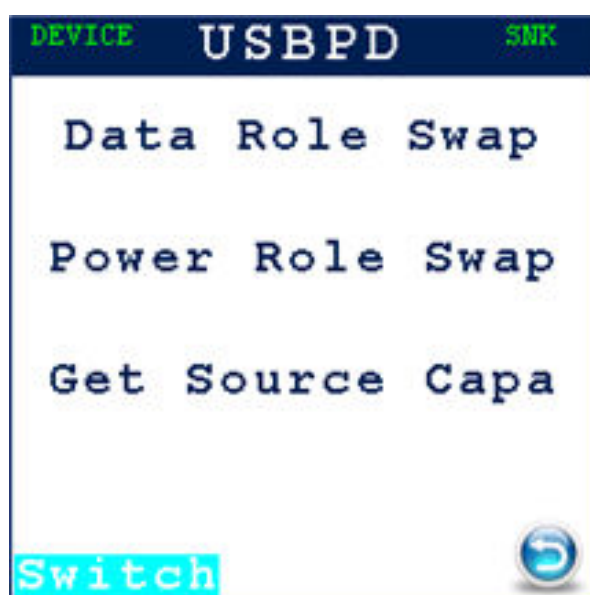
Figure 14. Mouse mode interface
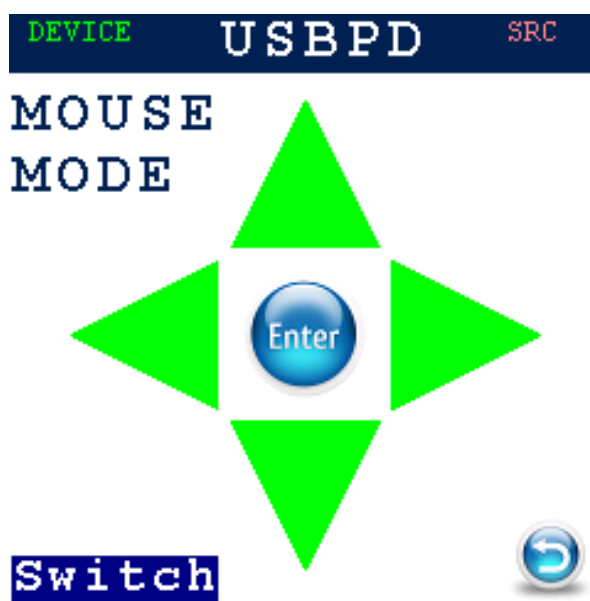
By clicking on the *Switch* button, and connecting a device via a USB Type-C®, the USBPD command menu and device name are displayed on the screen.

The USBPD command menu is as follows:

- Data Role Swap
- Power Role Swap
- Get Source/Sink capabilities

Figure 15. USBPD command menu

- Data Role Swap:
  USBPD command to switch the role between host and device (device acts as host and host acts as device).
  If the Data Role Swap request is rejected, a rejection message is displayed.

- Power Role Swap:
  USBPD command to switch power role between power provider and power consumer (source/provider power acts as sink/consumer power and sink/consumer power acts as source/provider power).

- Get Source capabilities:
  Can be triggered by pressing the *Get Source Capabilities* menu. If the request is rejected, a rejection message is displayed. If the request is accepted, the source capabilities are displayed on the screen.

- Get Sink capabilities:
  Can be triggered by pressing the *Get Sink Capabilities* menu. If the request is rejected, a rejection message is displayed. If the request is accepted, the capacities of the Sink are displayed on the screen.

Figure 16. USBPD Source capabilities   **Figure 17. USBPD Sink capabilities**

3.3.4 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 and disable the alarm.

When the alarm is triggered, the green LED is toggling for a while.

Figure 18. Calendar main menu
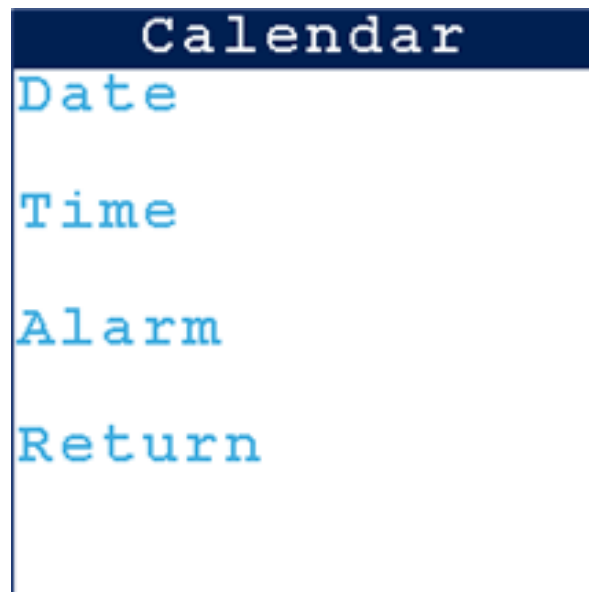
Figure 19. Date configuration interface

Figure 20. Alarm configuration interface

3.3.5 Cryptography module

An on-the-fly decryption engine (OTFDEC) is a peripheral that allows decryption during read operations without considerable losses in terms of execution time.

In this module, the OTFDEC is used with OCTOSPI to run an encrypted code from an external memory (cipher code).

The OCTOSPI is used to run the same code not encrypted from external memory (plain code).

To check OTFDEC performance, the execution time is calculated for both code running methods, and a comparison is done using a graph as shown in Figure 21.

For a better result display purpose, the execution time starts from three seconds.

Figure 21. Execution time comparison
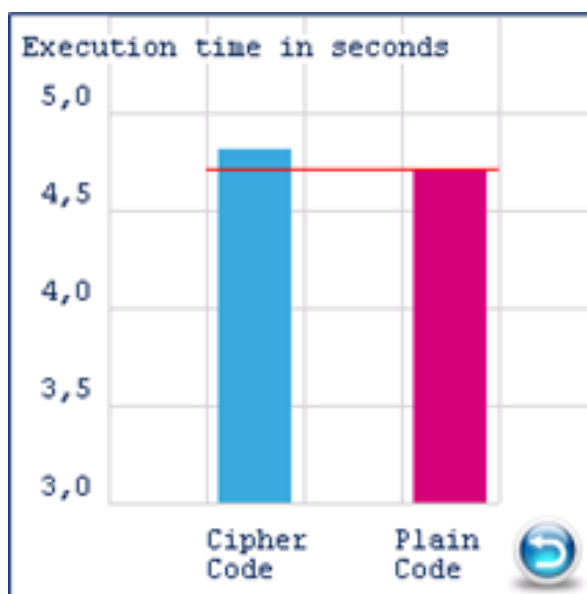
3.3.6 System information module

The system information shows the main demonstration information such as the used board, the STM32CubeH5 demonstration firmware version, the MCU name, the board name, and the CPU clock.

Figure 22. System information window

Revision history

Table 2. Document revision history

Date	Revision	Changes
05-Jul-2023	1	Initial release.

Contents

List of tables

List of figures

IMPORTANT NOTICE – READ CAREFULLY