STM32H7 dual-core demo game user guide

Introduction

This user guide describes a dual-core demonstration developed on the STM32H747I-DISCO Discovery kit featuring an STM32H747 dual-core microcontroller (STM32H747XI).

The demonstration consists of a game named *Tutti-Frutti* that uses the STM32H747 dual-core microcontroller to run simultaneously two applications in parallel. Its main purpose is to help the user to observe the benefit of:

- splitting an application in two and using both the Arm® Cortex®-M7 and Arm® Cortex®-M4 processing cores available on-chip
- core computation offload

The demonstration runs on the STM32H747I-DISCO without the need for any external accessory.

Figure 1. Demo game welcome screen

1 General information

The STM32H747I-DISCO Discovery kit features an STM32H747 microcontroller based on the Arm® Cortex®-M7 and Cortex®-M4 processing cores.

Table 1 defines the terms used in this document.

Table 1. Acronyms, abbreviations, and definitions

Term	Definition
AI	Artificial intelligence.
Chef	Character played by the application in the demo dual-core game. Refer to M7.
CM4	Name given to the Cortex®-M4 core in the application package.
CM7	Name given to the Cortex®-M7 core in the application package.
Customer	Character played by the user in the demo dual-core game.
M4	Arm® Cortex®-M4 core of the STM32H747 microcontroller. It represents the waiter in the demo dual-core game.
M7	Arm® Cortex®-M7 core of the STM32H747 microcontroller. It represents the chef in the demo dual-core game.
ML	Machine learning.
Waiter	Character played by the application in the demo dual-core game. Refer to M4.

Table 2 lists the reference documents used in this user manual.

Table 2. References

Reference	Documents
[KIT]	• *Discovery kit with STM32H747XI MCU* data brief (DB3608) • *Discovery kit with STM32H747XI MCU* user manual (UM2411)
[MCU]	• *Dual 32-bit Arm® Cortex®-M7 up to 480MHz and -M4 MCUs, up to 2MB Flash, 1MB RAM, 46 com. and analog interfaces, SMPS, DSI* datasheet (DS12930) • *STM32H745/755 and STM32H747/757 advanced Arm®-based 32-bit MCUs* reference manual (RM0399) • *STM32H745/755 and STM32H747/757 lines dual-core architecture* application note (AN5557)
[AI]	• *Artificial intelligence ecosystem for STM32* in the STMicroelectronics website at *www.st.com* with presentations, resources, and demonstration videos • *Artificial intelligence overview and examples* in the STMicroelectronics wiki site at wiki.st.com/stm32mcu

Note: Arm is a registered trademark of Arm Limited (or its subsidiaries) in the US and/or elsewhere.

2 Presentation

2.1 General description

The STM32H747 dual-core demonstration runs on the STM32H747I-DISCO Discovery kit. The demonstration is a game that combines graphical animation and artificial intelligence. It takes place in a juice bar.

The user interacts with the demonstration mainly by drawing on a blackboard. The two Cortex®-M cores are personified by two game characters (the chef and the waiter). The user plays the customer commanding a juice by drawing a fruit on the blackboard before the time runs out. The AI engine of the game is used to recognize the fruit drawn by the user. The output of the game is the production of the juice. When the expected juice is produced, the customer shows a happy face and pays for the juice.

2.2 Board used and ecosystem

The application is running on the STM32H747I-DISCO. This Discovery kit is a complete demonstration and development platform for the STM32H747 microcontrollers. It can be used as a reference design for user application development before porting to the final product.

STM32H747I-DISCO features the STM32H747XI dual-core microcontroller based on the Arm® Cortex®-M7 and the Arm® Cortex®-M4 with 2 Mbytes of flash memory. It has a 4” capacitive LCD touch-panel module with MIPI DSI® interface (refer to [MCU]).

Both cores are located in two separate power domains.

- The Cortex®-M7 is located in the D1 domain and operates at up to 480 MHz
- The Cortex®-M4 is located in the D2 domain and operates at up to 240 MHz

The full range of hardware features available on the Discovery kit helps users to improve their application development by an evaluation of all the peripherals such as USB OTG HS, SDRAM, Quad-SPI flash memory, DCMI connector, and others (refer to [KIT]).

The tools used for developing the demo are the following:

- STM32CubeMX, STM32Cube initialization code generator
- TouchGFX (X-CUBE-TOUCHGFX), graphical framework optimized for STM32 microcontrollers
- STM32Cube.AI (X-CUBE-AI), conversion of a trained neural network to an optimized STM32 microcontroller code (refer to [AI] for more information about the STMicroelectronics AI ecosystem)

The demo is based on the FreeRTOS™ middleware.

Note: FreeRTOS is a trademark of Amazon in the United States and/or other countries.

3 Game overview

3.1 Game description

The *Tutti-Frutti* demo game, which is presented as the dual-core game, takes place in a juice bar. In the juice bar, the dual-core feature of the STM32H747 is represented by two characters, the chef and the waiter, each being managed by one of the two cores.

The game is about a customer, played by the user, visiting the juice bar. The customer selects which fruit juice to order by drawing the fruit on the blackboard.

To launch the game and display its main screen, click on the [**START**] button of the welcome screen. Both screens are described in the Table 3, along with the game info screen, in the Section 3.2 Screen descriptions.

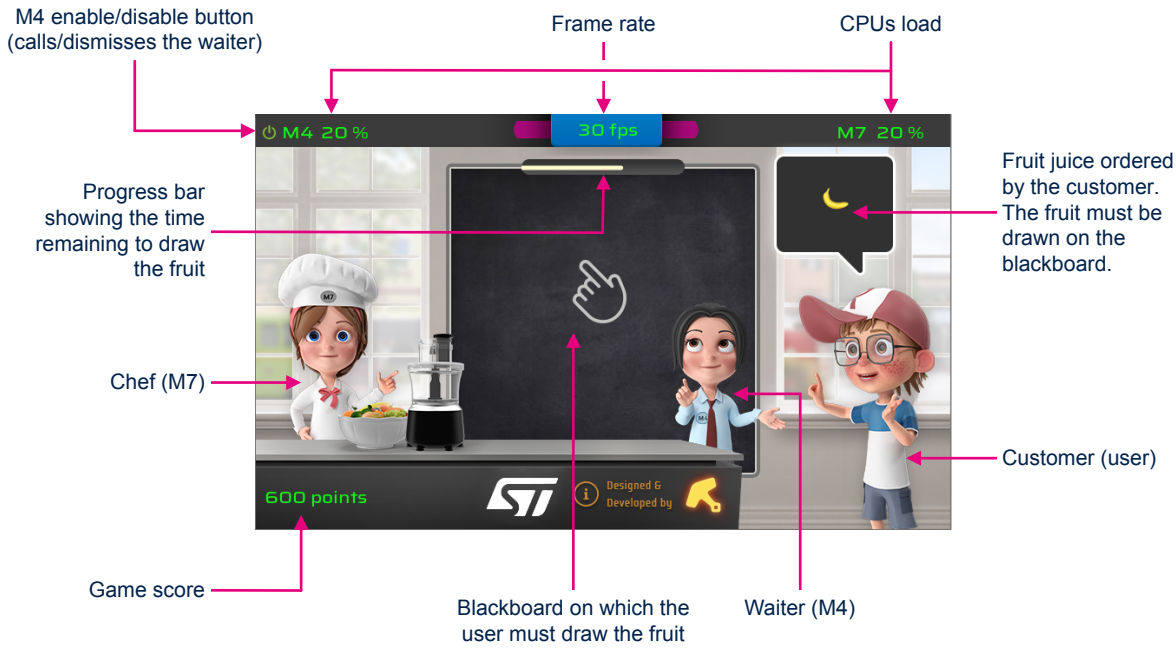
The waiter serves the juices for the customers entering the bar. When the customer decides which fruit juice to order, the user must simply draw the given fruit on the blackboard before time runs out. The time left is indicated by a progress bar located above the blackboard.

If the waiter recognizes the fruit drawn by the user, the customer is satisfied to receive the desired juice and pays for it. The waiter, in turn, gets points. The drawing of the fruit on the detection blackboard is detailed further in the Section 3.3 Fruit recognition examples.

If, on the contrary, an incorrect fruit is detected, the customer receives a juice that differs from the order. The customer is disappointed, does not pay for the order, and leaves immediately.

The different fields of the main screen of the game are detailed in the Figure 2.

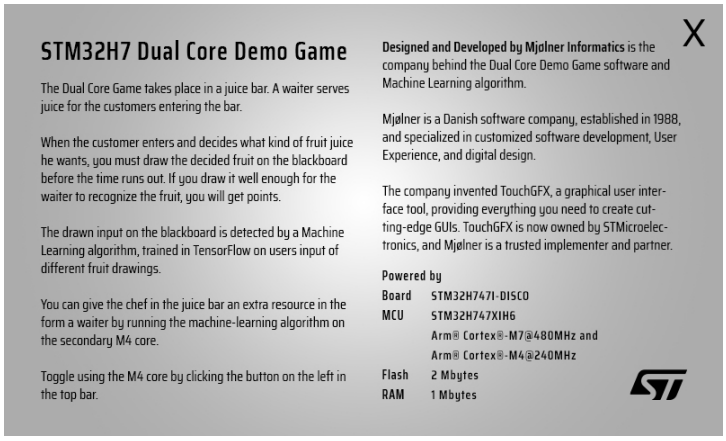
Figure 2. Main screen fields



Click on the button to launch the info screen. The info screen is described in the Section 3.2 Screen descriptions.

3.2 Screen descriptions

Table 3. **Screen descriptions**

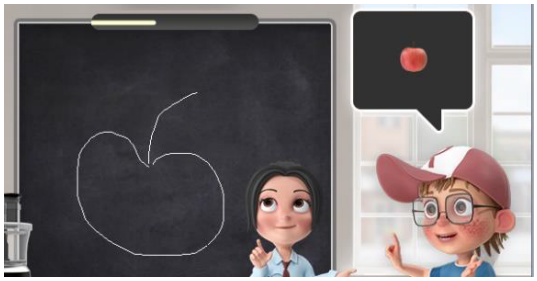
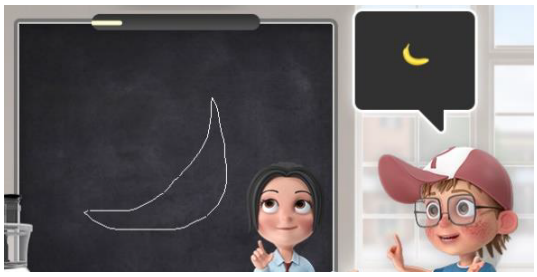
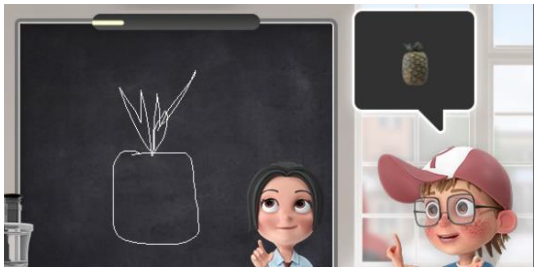
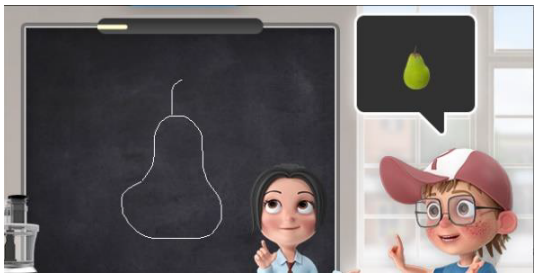
Screen	Description
	Welcome screen. This screen presents the name of the game and contains the [**START**] button.
	Main screen. In this screen, the application takes place with the different widgets and steps presented in the Section 3.1 Game description.
	Info screen. This screen describes the application, the development team, and the hardware.

3.3 Fruit recognition examples

The user must wait for the customer to choose a fruit juice (scrolling of fruits). When a fruit is selected, the user must draw the corresponding fruit on the blackboard accordingly before time elapses. The progress bar above the blackboard indicates the time remaining.

If the fruit is well drawn so that the waiter recognizes it, the customer is pleased, pays for the juice, and 100 points are added to the user score. Table 4 shows some real examples of fruit drawing recognition.

Table 4. Fruit drawing examples

4 Program the binary files to the STM32H7 board

To program the demo binary files to the STM32H747I-DISCO, use the STM32CubeProgrammer tool (STM32CubeProg) and follow the instructions below.

Step 1. Download the demo.

Download the *Tutti-Frutti* demo zipped binary from the *Documentation* section of the STM32H747I-DISCO webpage.

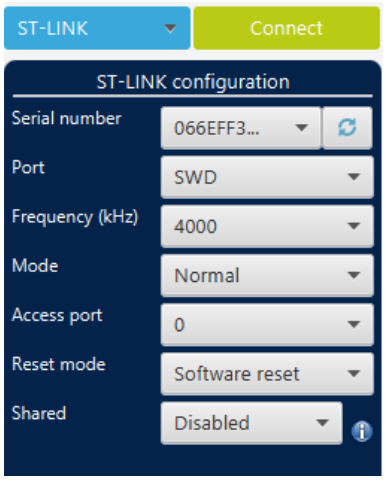
Step 2. Unzip the downloaded zipped file.

Step 3. Launch STM32CubeProgrammer.

Step 4. Connect the board.

Click on [**Connect**] once the board ST-LINK is detected as shown in Figure 3.

Figure 3. ST-LINK detection in STM32CubeProgrammer



Step 5. Open the [**External Loader**] menu.

Click on the button.

Step 6. Select the memory.

Select the Quad-SPI flash memory in the external loader as shown in Figure 4.

Figure 4. Q-SPI flash memory selection

Select	Name	Board	Start Address	Memory Size
☑	MT25TL01G_STM32H747I-DISCO	STM32H747I-D...	0x90000000	128M

Step 7. Open the [**Erasing & Programming**] menu.

Click on the button.

Step 8. Browse to the binaries already unzipped.

Step 9. Download the file `CM4/intflash.hex`, click on [**start program**], and wait for the operation completion.

Step 10. Download the file `CM7/target.hex`, click on [**start program**], and wait for the operation completion.

Once the steps above are completed, run the *Tutti-Frutti* dual-core demonstration on the STM32H747I-DISCO.

5 *Tutti-Frutti* demo

5.1 Demo in dual-core configuration

The input drawn on the blackboard is detected by a form detection machine learning algorithm. This algorithm is trained in TensorFlow™ on user drawings of different fruits. This training dataset is obtained from the Google Quick, Draw!™ online game.

During the drawing input, the progress bar indicates the active timer that represents the running machine learning detection.

In the interactive application, the dual-core feature of the STM32H747I-DISCO is represented by a personification of its two cores:

- The waiter is M4, the Arm® Cortex®-M4 core
- The chef is M7, the Arm® Cortex®-M7 core

When both cores are running, the tasks are divided between them:

- The Cortex®-M4 runs the machine learning algorithm, using the fruit drawing on the touch panel as an input
- The Cortex®-M7 manages the graphical application

The loads of both cores are shown on the top of the main screen. They illustrate the dual-core behavior of the application.

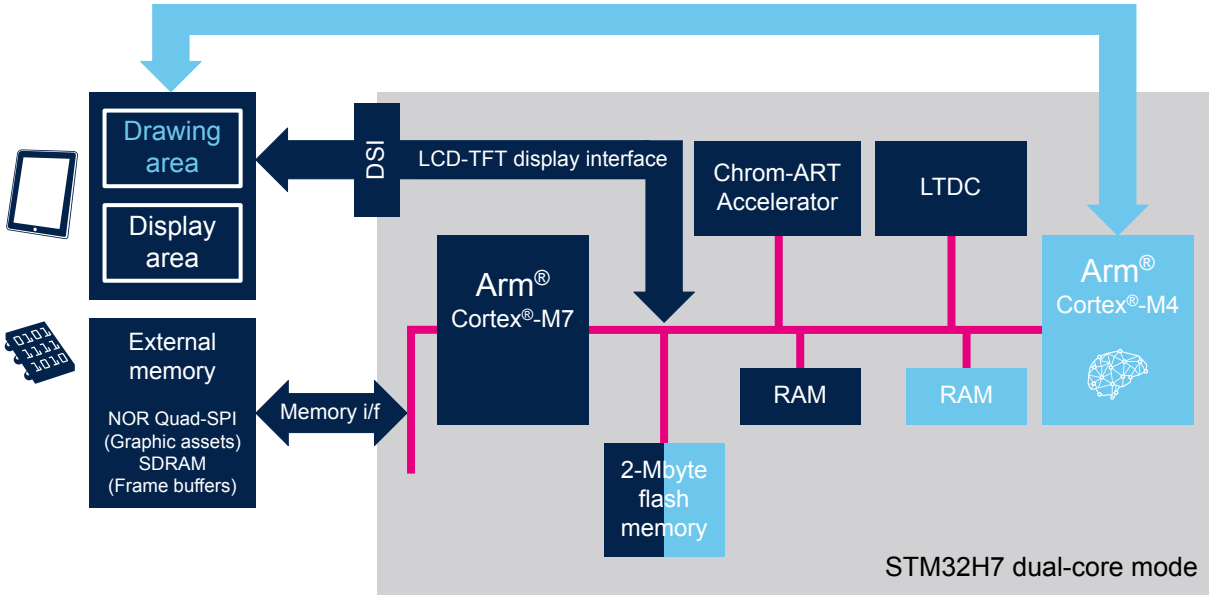
When the user draws the desired fruit, both cores are active:

- the Cortex®-M4, which can reach up to 300 DMIPS at 1.25 DMIPS/MHz, runs at a high load
- the Cortex®-M7, which can reach up to 1027 DMIPS at 2.14 DMIPS/MHz, executes high-performance graphical tasks in parallel and runs at a low load

The Cortex®-M4 offloads the Cortex®-M7 and offers optimal performance for real-time applications.

The configuration of the demo in the dual-core mode is detailed in the Figure 5.

Figure 5. Demo in dual-core configuration



Note: TensorFlow and Quick Draw! are trademarks of Google Inc.

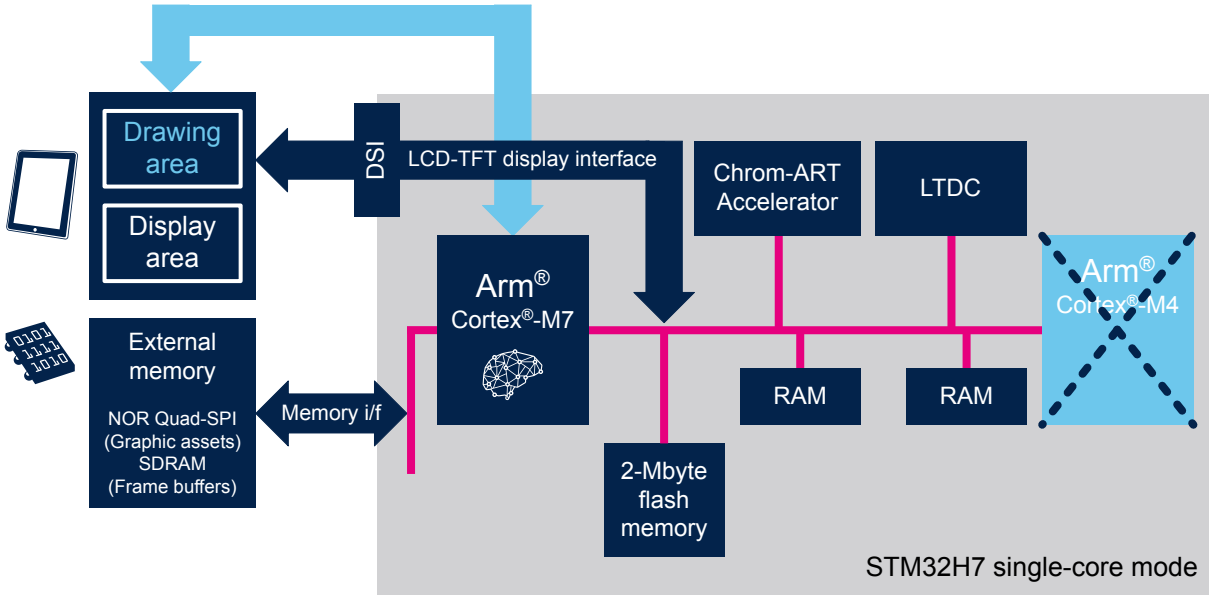
5.2 Demo in single-core configuration

It is possible to turn off the use of the Cortex®-M4 core by clicking the button on the left of the top bar in the main screen (refer to Figure 2). In this case, the game no longer runs as a dual-core application but as a single-core application.

With a click on the button, the Cortex®-M4 is disabled and the waiter disappears. This means that the Cortex®-M7 alone takes charge of both the waiter and the chef roles: the Cortex®-M7 runs the machine learning detection as well as the graphical application. In this case, the load of the Cortex®-M7 reaches a higher level.

The configuration of the demo in the single-core mode is detailed in the Figure 6.

Figure 6. Demo in single-core configuration



5.3 Demo memory footprint

Table 5 details the demo memory footprint in the dual-core configuration.

Table 5. Demo memory footprint

Memory region	Used
Arm® Cortex®-M7	
AXI-SRAM_D1	~90 Kbytes
Internal flash memory	~410 Kbytes
Quad-SPI NOR flash memory	12 623 Kbytes
SDRAM (TouchGFX frame buffer)	1 125 Kbytes
Arm® Cortex®-M4	
SRAM1_D2	~54 Kbytes
Internal flash memory	174 Kbytes

Revision history

Table 6. **Document revision history**

Date	Revision	Changes
18-May-2022	1	Initial release.

Contents

List of tables

List of figures