

Embedded Machine Learning for Edge Computing

Introduction to STM32 microcontrollers

Kithmin Wickremasinghe

About Myself

- MSc Student in Electrical and Comp. Eng., UBC, Canada
- B.Sc. in Electronic & Telecom. Eng. from the University of Moratuwa, Sri Lanka (2020)
- Former Intern at Ansys Canada Ltd.
- Former R&D Intern at Telexistence Inc. (Tokyo, Japan)
- www.kithminrw.com



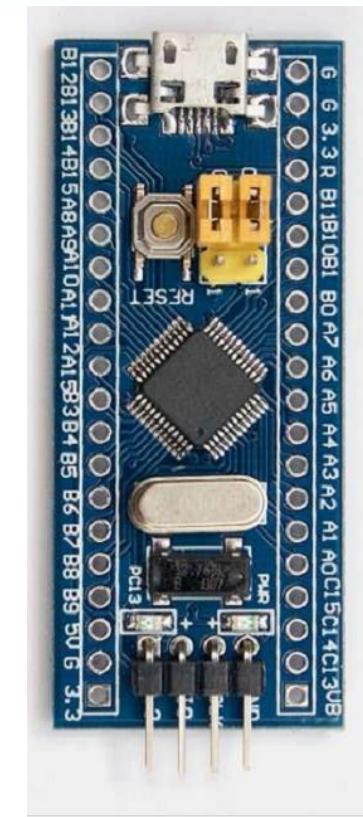
Learning outcomes

- Introduction to STM32.
- How to make a simple LED blink with STM32 and simulating in Wokwi.
- Advanced topics with STM32Cube.
 - STM32Cube Ecosystem
 - STM32CubeMX keysteps
- Overview of STM32 CUBE.AI.
- Case studies with STM32 CUBE.AI.

Why STM32 for Embedded AI

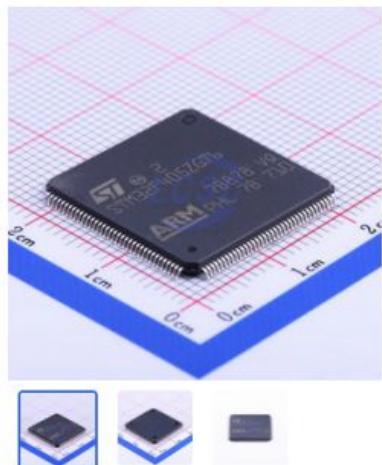


		Uno	STM32 Blue Pill
General	Dimensions	2.08" x 0.9"	4" x 2.1"
	Pricing	\$20-23	\$3-5
Connectivity	I/O Pins	14	37
	PWM Pins	6	15
	Analog In Pins	6	10
	Analog Out Pins (DAC)	-	-
Computing	Processor	ATMega328P	STM32F103C8T6
	Flash Memory	32 kB	64 kB
	SRAM	2 kB	20 kB
	EEPROM	1 kB	-
	Clock speed	16 MHz	72 MHz
	Voltage Level	5V	3.3V
Communication	USB Connectivity	Standard A/B USB	Micro-USB
	Hardware Serial Ports	1	3
	SPI Support	Yes (1x)	Yes (2x)
	CAN Support	No	Yes
	I2C Support	Yes (1x)	Yes (2x)



- Easy to learn
- Ready made libraries
- Huge community
- Resource availability
- Complexity of initial setup
- Higher costs for purchasing
- Need to write code from scratch.

Why STM32 for Embedded AI



STMicroelectronics STM32F405ZGT6

Manufacturer	STMicroelectronics
Mfr. Part #	STM32F405ZGT6
LCSC Part #	C92114
Package	LQFP-144_20x20x05P
Customer #	<input type="text"/> Customer Number
Datasheet	STMicroelectronics STM32F405ZGT6
EasyEDA	EasyEDA Model
Description	192KB 1MB 1.8V~3.6V -40°C~+85°C ARM Cortex-M4 1@x24ch/12bit 1@x2ch/12bit

Images are for reference only

 Add to Favourites.

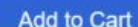
In Stock: 61

 Stock Notification ?

61 Can Ship in 3 Days

 Inquire for More Quantity

1 

 Add to Cart

Minimum : 1 Multiples : 1

Unit Price US\$ 15.7341

Ext. Price US\$ 15.73



 Click to expand

Arduino Nano 33 BLE Sense

Code: ABX00031 / Barcode: 7630049201507

\$40.50

Quantity 
1

 SOLD OUT

 ADD TO WISHLIST

Bring the power of AI to your pocket with Arduino's tiniest form factor.

Enhance your experience, add to your cart:

<input type="checkbox"/> Nano Screw terminal adapter (3 boards pack)	\$0.00
<input type="checkbox"/> Arduino MKR IoT Bundle	\$35.97
<input type="checkbox"/> Arduino MKR RGB Shield	\$92.60
<input type="checkbox"/> Gravity: 27 Pcs Sensor Set for Arduino	\$53.90
<input type="checkbox"/> USB Cable Type A Male to Micro Type B Male	\$101.40
	\$6.90

Share this:    

Basic setup with STM32 MCU

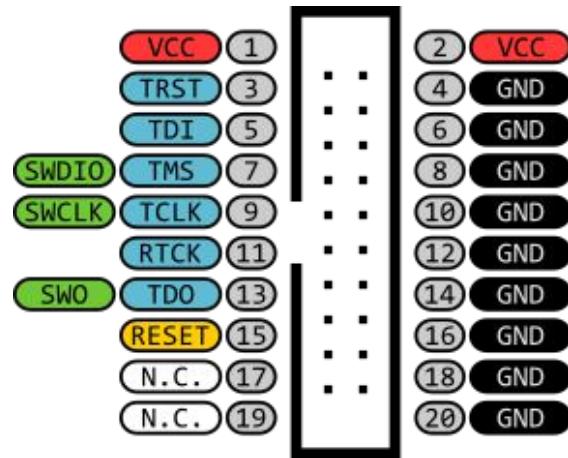
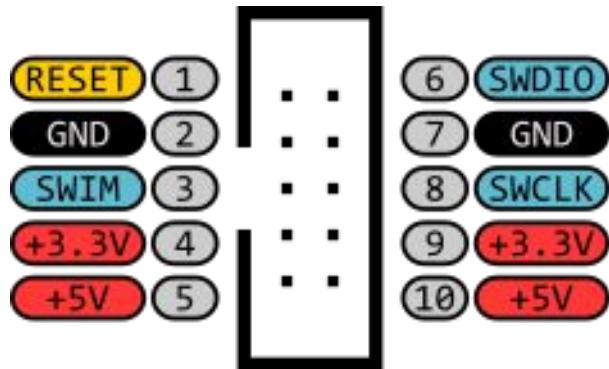
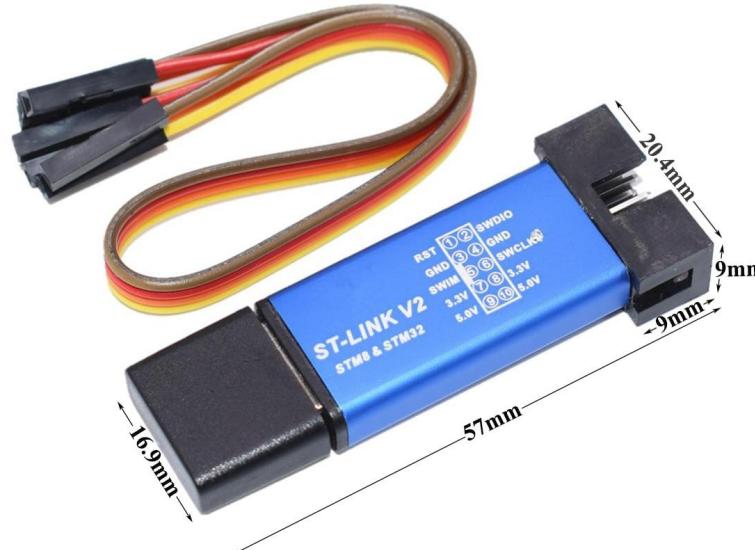


- Things to consider,
 - STM32 is a 3.3V MCU,
 - Supply should be 3.3V.
 - There are 5V tolerant pins therefore be careful only to use them for 5V.
 - Have an ST-link programmer,
 - Can debug and program.
 - Can flash the MCU with UART but no debugging is available in that approach.
 - Select proper memory with Boot 0.
 - Select clock source and frequency.
 - Select peripherals to activate.
 - Select middleware.
 - Write C++ code w/ HAL library from scratch

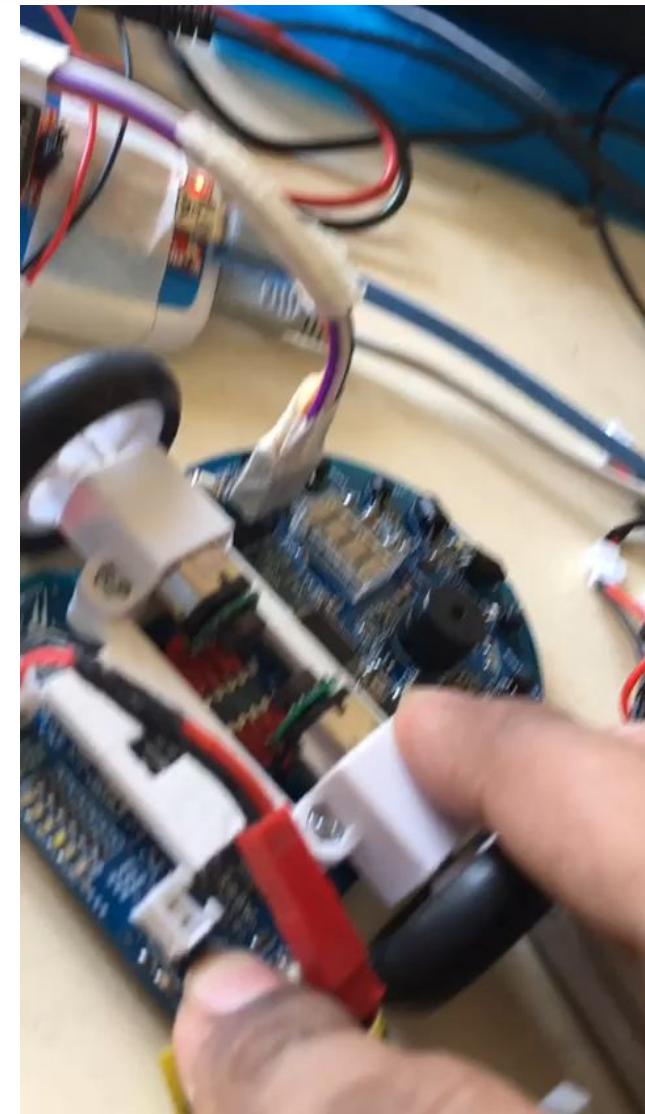
STM32 MCUs 32-bit Arm® Cortex®-M		10 YEARS COMMITMENT	
	High Performance	STM32F7 1082 CoreMark 216 MHz Cortex-M7	STM32H7 Up to 3224 CoreMark Up to 600 MHz Cortex-M7 240 MHz Cortex-M4
	STM32G0 142 CoreMark 64 MHz Cortex-M0+	STM32G4 569 CoreMark 170 MHz Cortex-M4	STM32GO 114 CoreMark 48 MHz Cortex-M0+ STM32FO 106 CoreMark 48 MHz Cortex-M0 STM32F1 177 CoreMark 72 MHz Cortex-M3 STM32F3 245 CoreMark 72 MHz Cortex-M4
	STM32L0 75 CoreMark 32 MHz Cortex-M0+	STM32U4+ 409 CoreMark 120 MHz Cortex-M4	STM32U5 651 CoreMark 160 MHz Cortex-M33
	STM32WL 162 CoreMark 48 MHz Cortex-M4 48 MHz Cortex-M0+	STM32WB0 64 MHz Cortex-M0+ STM32WB 216 CoreMark 64 MHz Cortex-M4 32 MHz Cortex-M0+	STM32WBA 407 CoreMark 100 MHz Cortex-M33

Cortex-M0+
Radio co-processor

STM32 Microcontroller Programmers

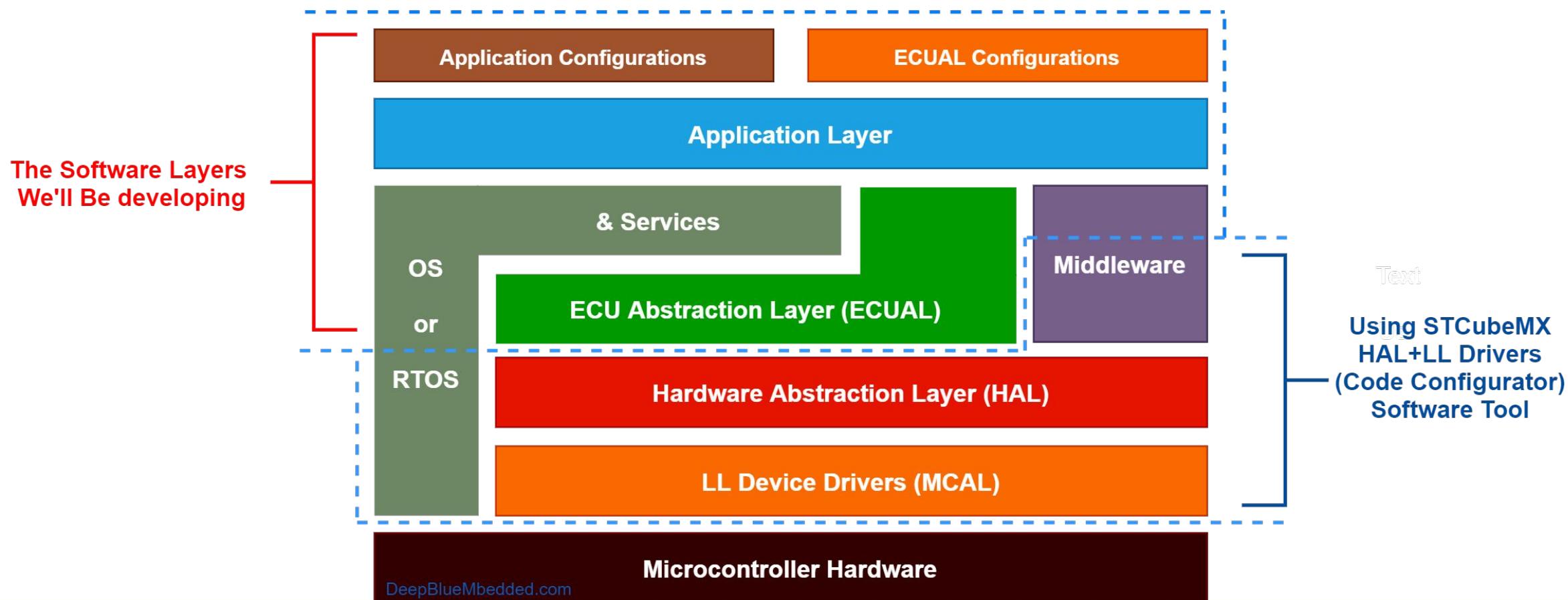


<https://stm32-base.org/guides/connecting-your-debugger.html>



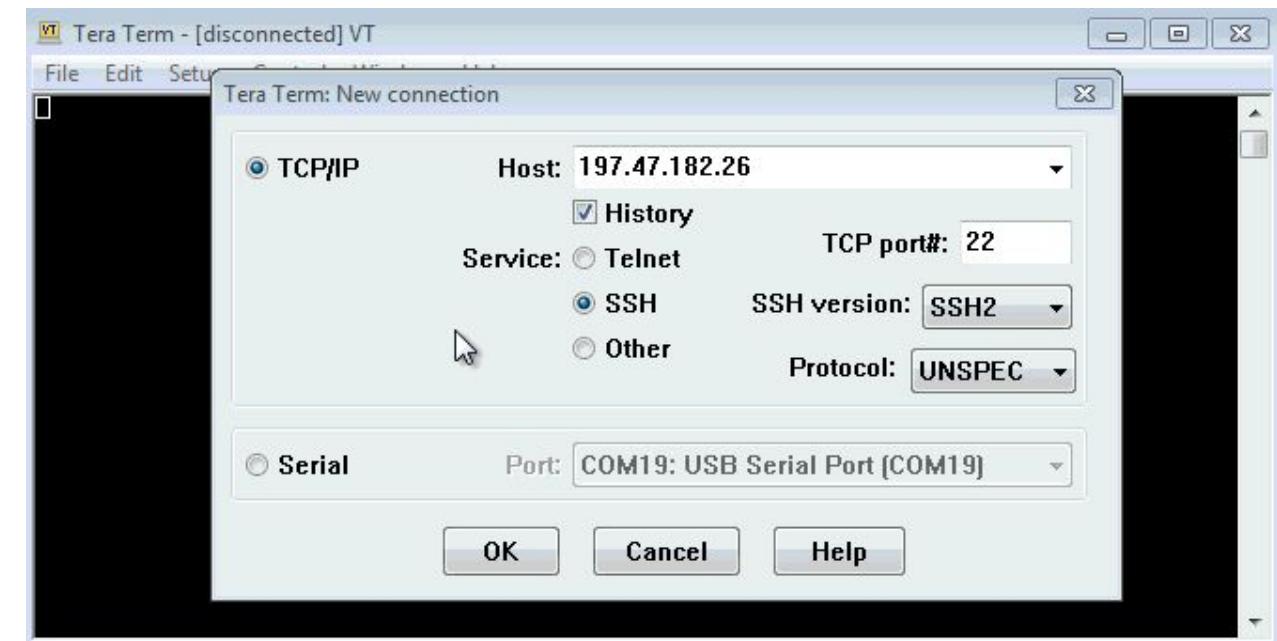
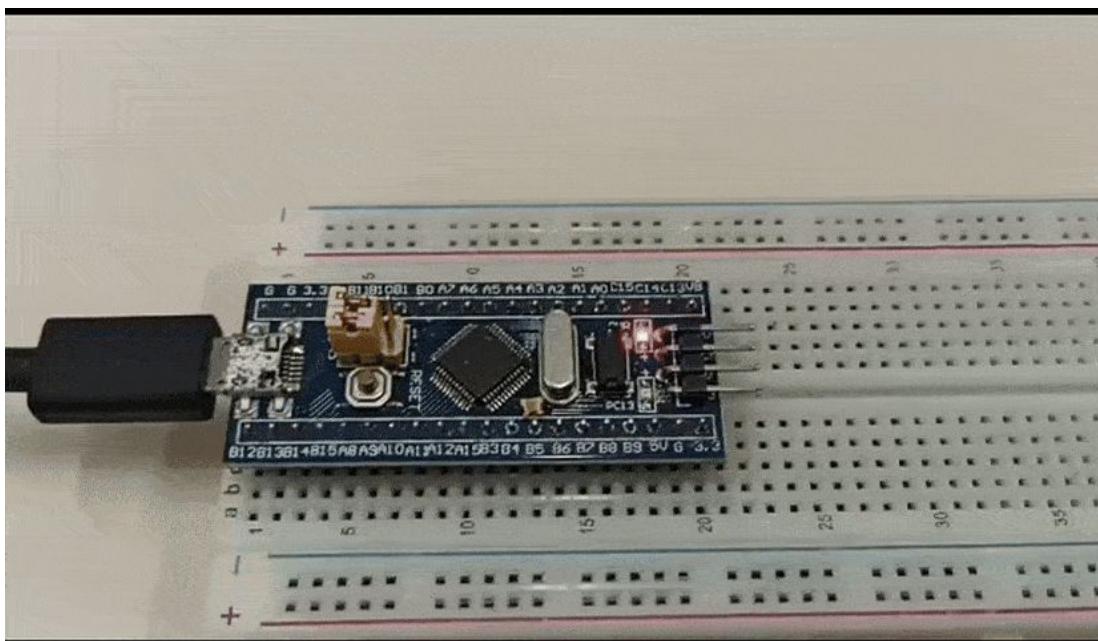
HAL

Hardware abstraction layer (HAL) is a layer of programming in an operating system that allows to interact with hardware at a general or abstract level rather than a detailed hardware level. **Hardware abstractions** are sets of routines in software that provide programs with access to hardware resources through programming interfaces. The programming interface allows all devices in a particular class C of hardware devices to be accessed through identical interfaces even though C may contain different subclasses of devices that each provide a different hardware interface.



- Some Examples for STM32 cube HAL,

- `HAL_GPIO_WritePin(DEBUG_LED_GPIO_Port, DEBUG_LED_Pin, GPIO_PIN_SET);`
- `HAL_GPIO_TogglePin(DEBUG_LED_GPIO_Port, DEBUG_LED_Pin);`
- `HAL_UART_Transmit(&huart3, (uint8_t *)buf, buf_len, 100);`



Getting Started with STM32

1

Setting up the environment needed to begin with STM32

Shalutha @ ENTC 17

Hey guys,

So how are you guys? Hope you all are doing well.

From the last article, we told you that we will be looking into the programming side of STM32. But, we thought it would be much better for you guys, if we can give an idea about the hardware you need to do the programming with STM32 first. So today's article will be about the hardware you needed to program a STM32, and the step by step guide to settle them with your PC.

So the topics for today are as follows.

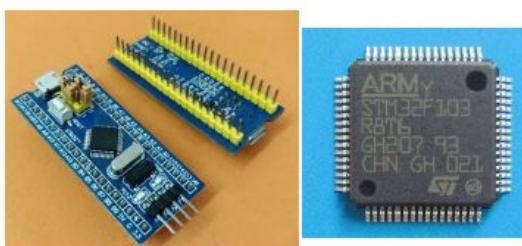
- 1 - Choosing a STM32 chip/Development Board for learning purposes
- 2 - Methods of programming the STM32 chip/Development Board
- 3 - Installing STM32CubeIDE

1 - Choosing a STM32 chip/Development Board for learning purposes

As I mentioned in the previous post, the STM32F1 series and the STMF4 series are the most suitable series for a beginner. For a price of RS 700-1000 range, you can easily buy a STM32F103C8 Development Board in Sri Lanka. F4 series's Nucleo Boards (having chips like STM32F401, STM32F411, etc) are a little bit high in cost in Sri Lanka. So because of the cost issue, we would recommend a STM32F103C8 board for testing purposes.

The chip we are using in our micromouse is STM32F405. The F4 series has a greater processing power and many other advantages (higher flash memory, SRAM, etc) than the F1 series, but it should be mentioned that we can use the STM32F103C8 board to check every sensor, actuator, etc. that we are using in our mouse. So that you don't need to wait until you completely build your mouse, you can check every function you are planning to use in your mouse, using the STM32F103C8 board (Bluepill).

Following shows a STM32F103C8 Development Board (Bluepill) & the chip (STM32F103C8) used in it



This one here is a STM32 Nucleo F411RE Development Board

Apr '20

Apr
2020
1 / 1
Apr
2020

2

Programming using CubeIDE #1- Blinking a LED

Senura @ ENTC 16

Apr '20

Hi guys!

Hope you are now able to set up the environment to start programming with STM32. If you have any questions or problems regarding our previous post (setting up the environment needed to begin with STM32) feel free to ask, because if you want to check your code in the real world that knowledge is a must.

Today, we will be looking at the key features of STM32CubeIDE and how a new project is created

Key Features

1. Integration of STM32CubeMX that provides services for:

- Project creation and STM32 microcontroller and microprocessor selection
- Pinout, clock, peripheral, and middleware configuration
- Generation of the initialization code

2. Based on ECLIPSE™/CDT, with support of ECLIPSE™ add-ons, GNU C/C++ for Arm® toolchain and GDB debugger

3. Additional advanced debug features including:

- CPU core, peripheral register, and memory views
- Live variable watch view
- System analysis and real-time tracing (SWV)
- CPU fault analysis tool

4. Support of ST-LINK (STMicroelectronics) and J-Link (SEGGER) debug probes

5. Import project from Atollic® TrueSTUDIO® and AC6 System Workbench for STM32 (SW4STM32)

6. Multi-OS support: Windows®, Linux®, and macOS®, 64-bit versions only

Project creation and STM32 microcontroller and microprocessor selection

Workspace and project creation :

The first thing to be done when using CubeIDE is to create a project. Before that we will introduce you, the concept of workspace.

A workspace is a container that includes project folders or information about project folders, and a .metadata folder that contains information about the projects. A workspace is simply a folder on a hard drive, which can be located anywhere in the storage media. When STM32CubeIDE starts up, it asks which workspace must be used. This may be changed at any time by selecting [File]>[Switch Workspace] and navigating to another folder.

The easiest way to create a new embedded project is to use the STM32 Project wizard. It is selected through the [File]>[New]>[STM32 Project] menu command, and launches the embedded MCUFinder:

1 - Select the target MCU or board and click next (We have chosen STM32F103C8 MCU).

- This image shows you the target selection window. In light blue color in the upper left corner of this image, you can see now we are at the MCU/MPU selector tab.

Apr
2020
1 / 2
Apr
2020

Apr
2020

Wokwi STM32 Simulator



WOKWI SAVE SHARE

Docs

main.c diagram.json Library Manager

```
1 // STM32 Nucleo-L031K6 HAL Blink + printf() example
2 // Simulation: https://wokwi.com/projects/367244067477216257
3
4 #include <stdio.h>
5 #include <stdint.h>
6 #include <string.h>
7 #include <stm32l0xx_hal.h>
8
9 // ST Nucleo Green user LED (PB3)
10#define LED_PORT          GPIOB
11#define LED_PIN           GPIO_PIN_3
12#define LED_PORT_CLK_ENABLE __HAL_RCC_GPIOB_CLK_ENABLE
13#define VCP_TX_Pin GPIO_PIN_2
14#define VCP_RX_Pin GPIO_PIN_15
15
16 UART_HandleTypeDef huart2;
17
18 void SystemClock_Config(void);
19 static void MX_USART2_UART_Init(void);
20
21 void osSystickHandler(void)
22 {
23     // 1 Hz blinking:
24     if ((HAL_GetTick() % 500) == 0)
25     {
26         HAL_GPIO_TogglePin(LED_PORT, LED_PIN);
27     }
28 }
29
30 void initGPIO()
31 {
32     GPIO_InitTypeDef GPIO_Config;
```

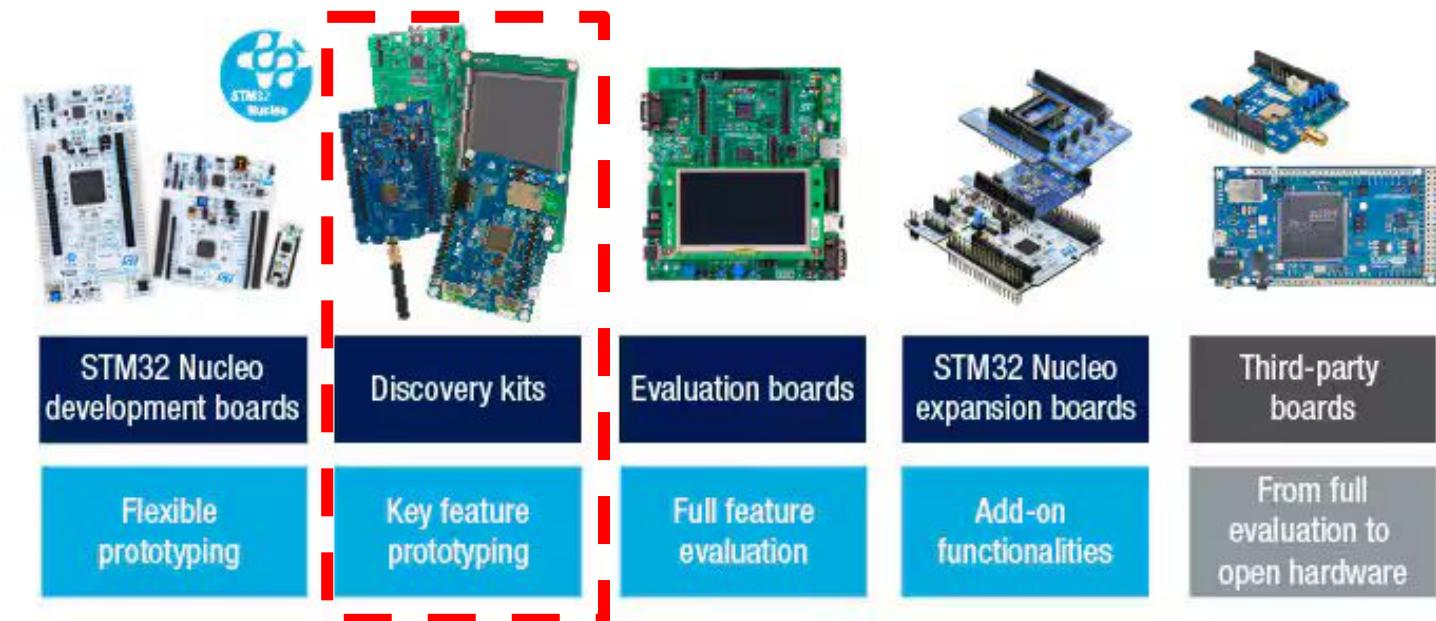
Simulation

00:14.496 99%

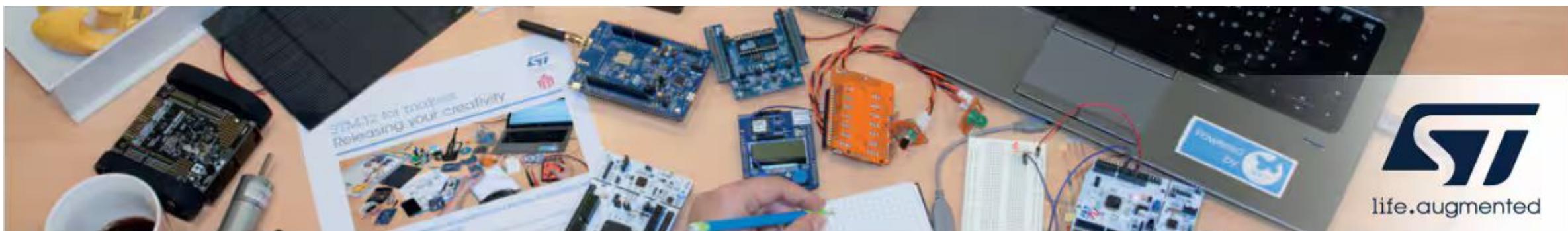
<https://wokwi.com/projects/392812608178590721>

Hello, World!
Hello Students! Tracing X = 11
Hello, World!
Hello Students! Tracing X = 12
Hello, World!
Hello Students! Tracing X = 13

Getting started with STM32



[Discovery kit with STM32F407VG MCU * New order code STM32F407G-DISC1](#)
(replaces STM32F4DISCOVERY) - STMicroelectronics



STM32 Series

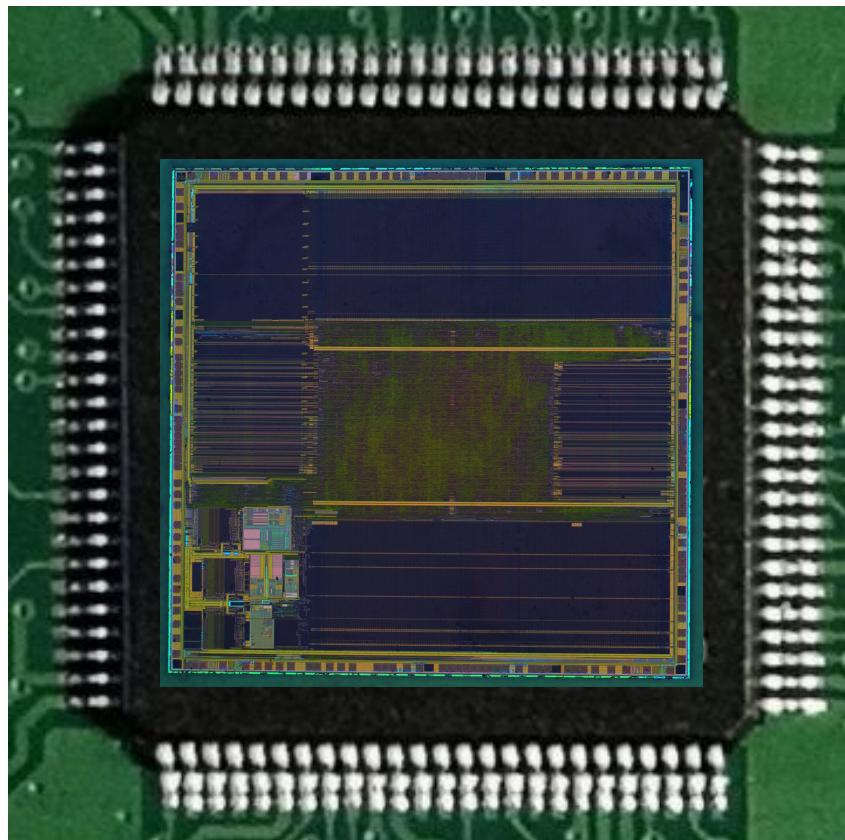
STM32
CubeMCU Packages

Dedicated to each STM32 Series

Mainstream MCU	High Performance MCU	MPU	Ultra-Low Power MCU	Wireless MCU
 STM32 CubeG4	 STM32 CubeH7	 STM32 CubeMP1	 STM32 CubeL0	 STM32 CubeWB
 STM32 CubeF3	 STM32 CubeF7		 STM32 CubeL1	 STM32 CubeWL
 STM32 CubeF1	 STM32 CubeF4		 STM32 CubeL4	
 STM32 CubeG0	 STM32 CubeF2		 STM32 CubeL5	
 STM32 CubeF0				

STM32 Datasheets

- Refer datasheet to get all the necessary information about the chip.
- Discuss important parts of the datasheet



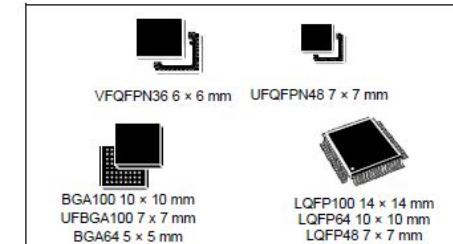
STM32F103x8
STM32F103xB

Medium-density performance line Arm®-based 32-bit MCU with 64 or 128 KB Flash, USB, CAN, 7 timers, 2 ADCs, 9 com. interfaces

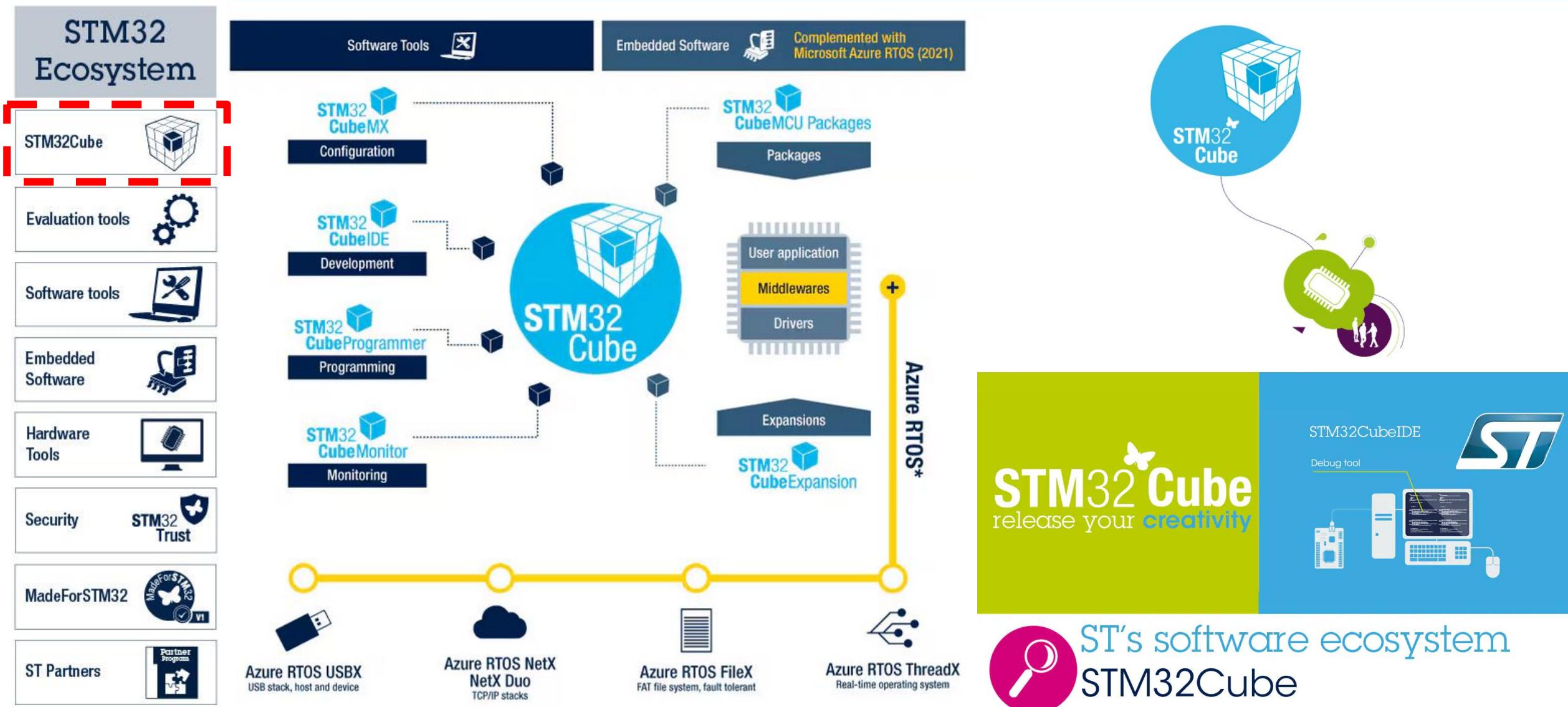
Datasheet - production data

Features

- Arm® 32-bit Cortex®-M3 CPU core
 - 72 MHz maximum frequency, 1.25 DMIPS / MHz (Dhrystone 2.1) performance at 0 wait state memory access
 - Single-cycle multiplication and hardware division
- Memories
 - 64 or 128 Kbytes of Flash memory
 - 20 Kbytes of SRAM
- Clock, reset and supply management
 - 2.0 to 3.6 V application supply and I/Os
 - POR, PDR, and programmable voltage detector (PVD)
 - 4 to 16 MHz crystal oscillator
 - Internal 8 MHz factory-trimmed RC
 - Internal 40 kHz RC
 - PLL for CPU clock
 - 32 kHz oscillator for RTC with calibration
- Low-power
 - Sleep, Stop and Standby modes
 - V_{BAT} supply for RTC and backup registers
- 2x 12-bit, 1 μ s A/D converters (up to 16 channels)
 - Conversion range: 0 to 3.6 V
 - Dual-sample and hold capability
 - Temperature sensor
- Debug mode
 - Serial wire debug (SWD) and JTAG interfaces
- Seven timers
 - Three 16-bit timers, each with up to 4 IC/OC/PWM or pulse counter and quadrature (incremental) encoder input
 - 16-bit, motor control PWM timer with dead-time generation and emergency stop
 - Two watchdog timers (independent and window)
 - SysTick timer 24-bit downcounter
- Up to nine communication interfaces
 - Up to two I²C interfaces (SMBus/PMBus®)
 - Up to three USARTs (ISO 7816 interface, LIN, IrDA capability, modem control)
 - Up to two SPIs (18 Mbit/s)
 - CAN interface (2.0B Active)
 - USB 2.0 full-speed interface
- CRC calculation unit, 96-bit unique ID



STM32Cube Ecosystem



STM32Cube Ecosystem

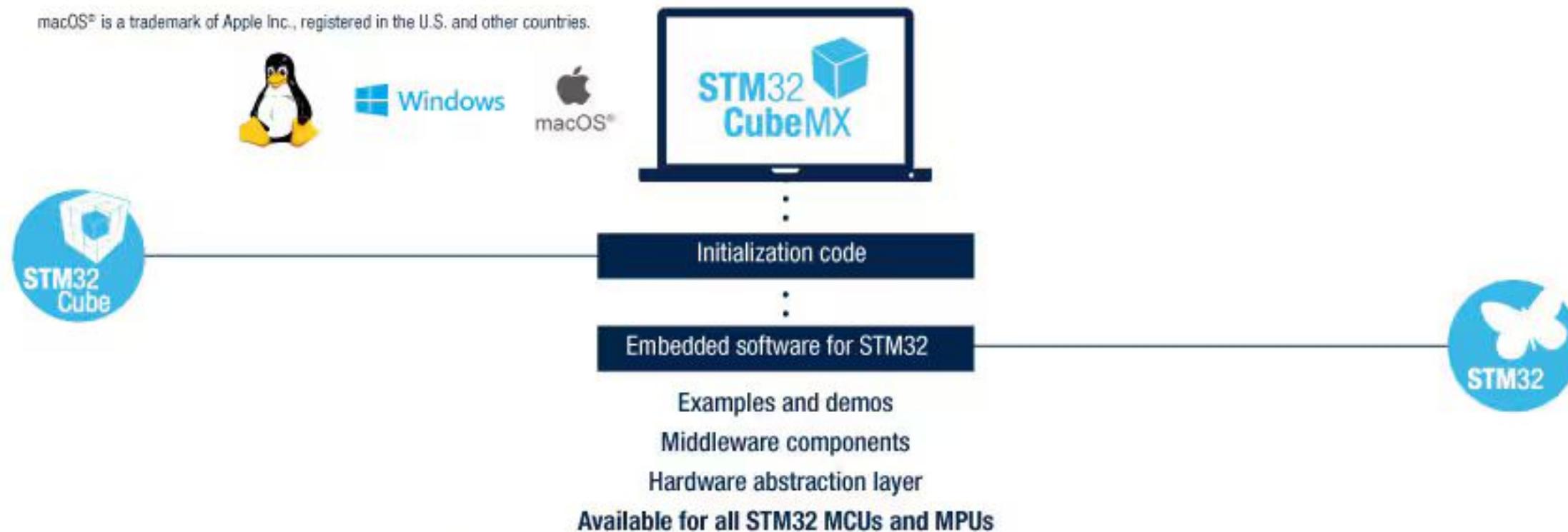


Part Number	Configuration	Initialization code Generation	Code Edition	Code Building	Debugging	Binary Programming	Monitoring
STM32CubeMX	✓	✓					
STM32CubeIDE	✓ (Integrates STM32CubeMX)	✓ (Integrates STM32CubeMX)	✓	✓	✓	✓	✓
STM32CubeProgrammer						✓	
STM32CubeMonitor							✓

Reference - [Discover the STM32Cube Ecosystem - STMicroelectronics](#)

STM32CubeMX

macOS® is a trademark of Apple Inc., registered in the U.S. and other countries.

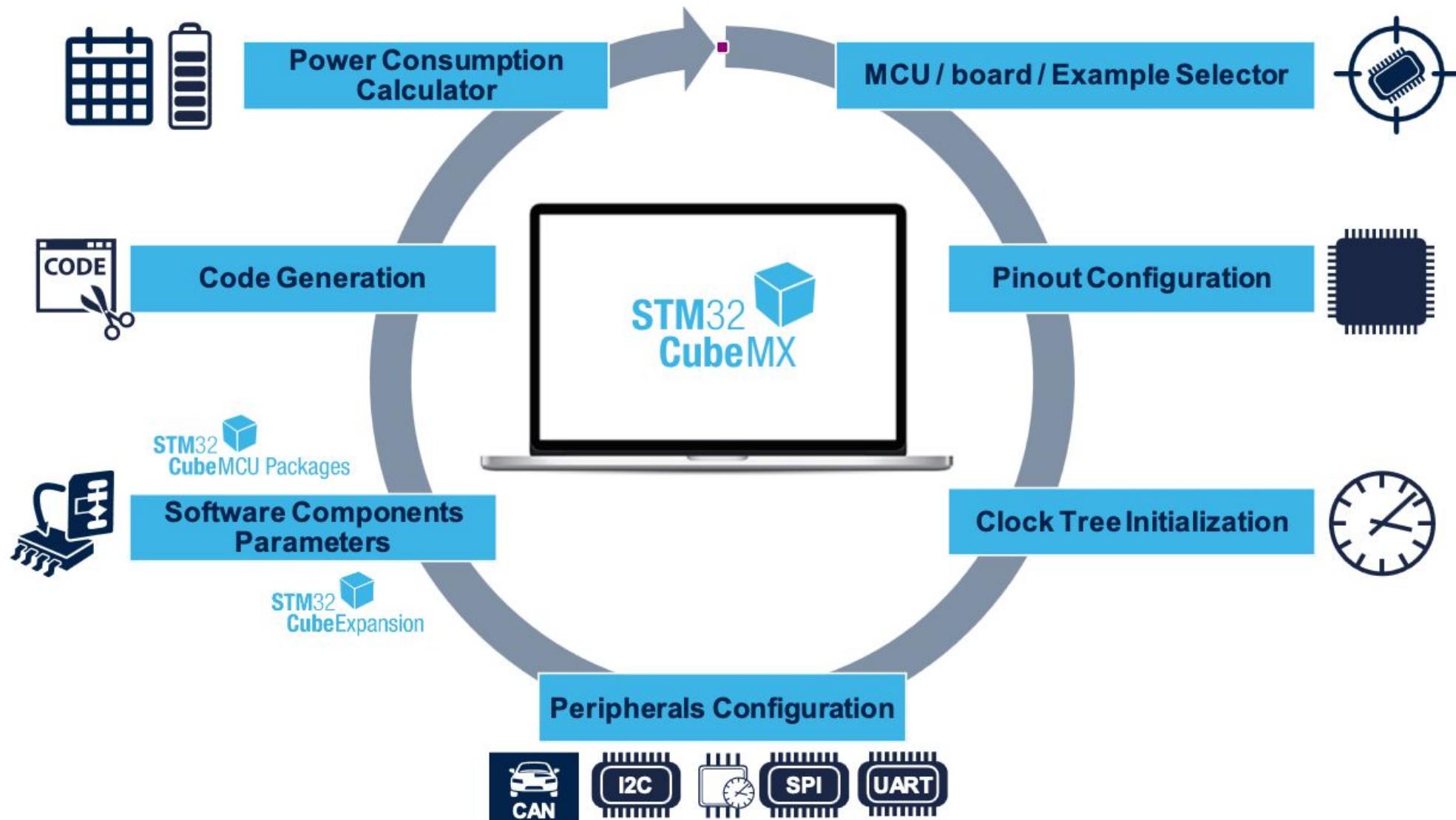


	High Performance MCUs					Mainstream MCUs					Ultra-low-power MCUs					Wireless MCUs			MPU		
STM32	F2	F4	F7	H5	H7	C0	F0	F1	F3	G0	G4	L0	L1	L4	L4+	L5	U5	WB	WBA	WL	MP1*

Note :* available for Cortex-M4 side only

[STM32CubeMX - STM32Cube initialization code generator - STMicroelectronics](#)

STM32CubeMX key steps



CubeMX IDE



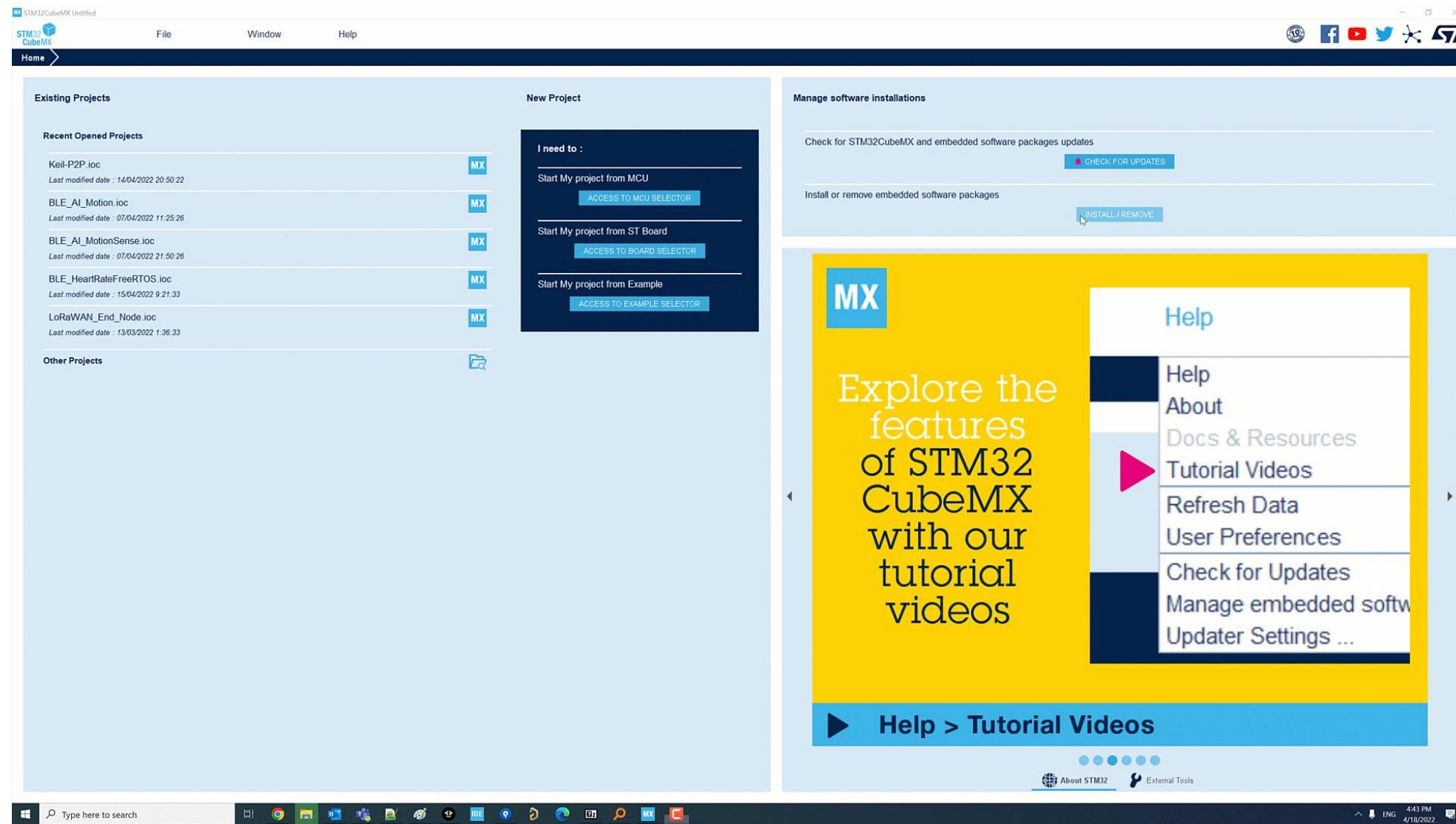
The screenshot shows the STM32CubeMX IDE interface. At the top, there's a toolbar with red, yellow, and green buttons, followed by the title "STM32CubeMX Untitled". Below the title is a navigation bar with "File", "Window", "Help", and a user icon labeled "myST". A "Home" button is also present. The main area is divided into three sections: "Existing Projects" (with a "Open Existing Projects" button and a folder icon), "New Project" (with three options: "Start My project from MCU" (with "ACCESS TO MCU SELECTOR" button), "Start My project from ST B..." (with "ACCESS TO BOARD SELECTOR" button), and "Start My project from Exa..." (with "ACCESS TO EXAMPLE SELECTOR" button)), and "Manage software installations" (with "Check for STM32CubeMX and e..." (with "CHECK FOR UPDATES" button) and "Install or remove embedded sof..." (with "INSTALL / REMOVE" button)). At the bottom, there's a footer with a "Data collection information notice" link, a "About STM32" link, and an "External Tools" link.

MCU / MPU
SELECTOR

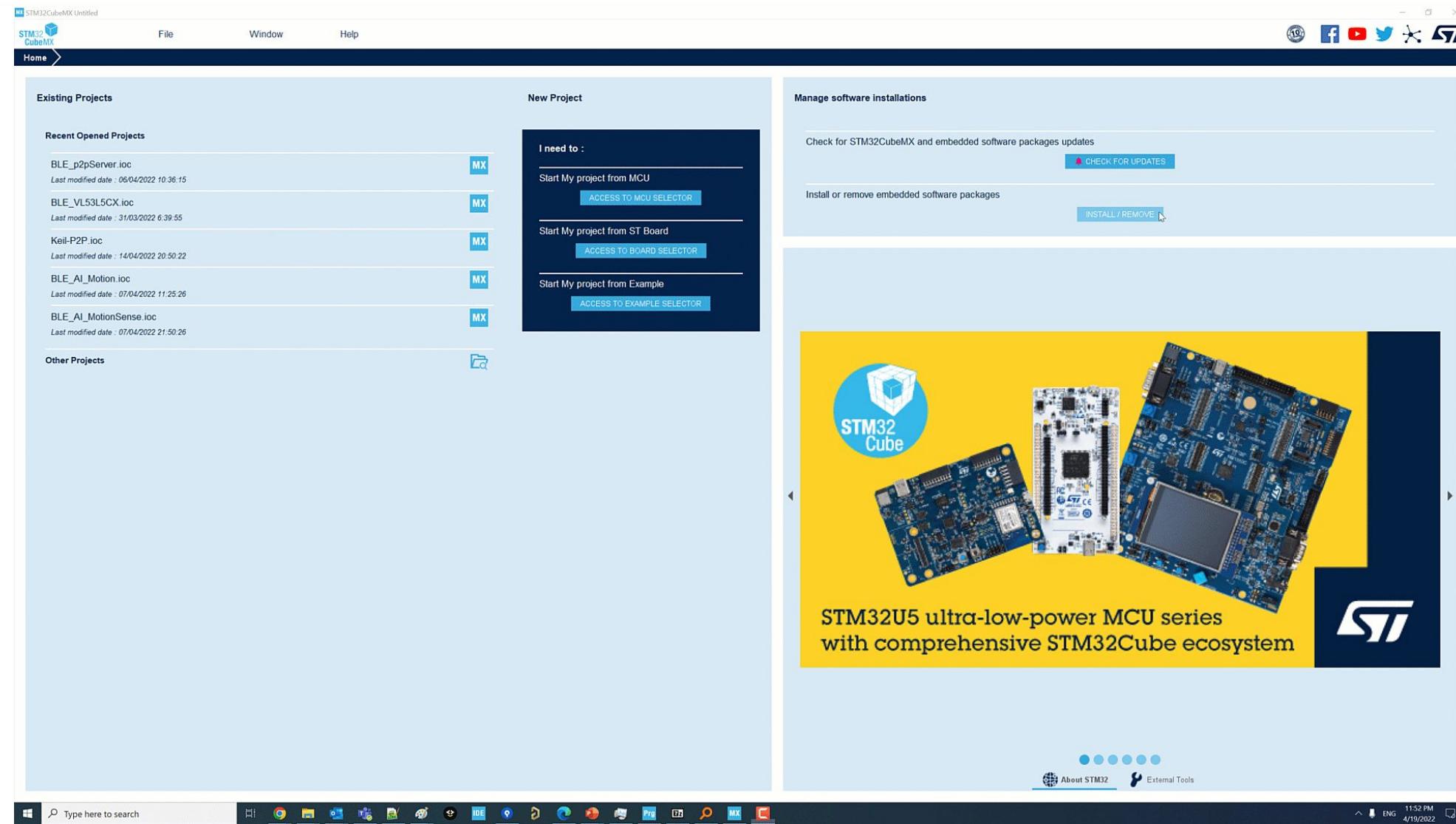
BOARD
SELECTOR

EXAMPLE
SELECTOR

Install Expansion Middleware - X.CUBE-AI



Install Expansion Middleware - X.CUBE-MEMS1



MCU/MPU or Board selector

MCU / MPU SELECTOR

BOARD SELECTOR

EXAMPLE SELECTOR

New Project from a Board

MCU/MPU Selector | Board Selector | Example Selector | Cross Selector

Board Filters

Commercial Part Number: stm32f407

SEARCH + -

PRODUCT INFO

- Type
- Supplier
- MCU / MPU Series
- Marketing Status
- Price

MEMORY

- Ext. Flash = 0 (MBit)
- Ext. EEPROM = 0 (kBytes)
- Ext. RAM = 0 (MBit)

FEATURES

- Embedded Sensor
- User Button
- Camera
- CAN

STM32F4 Series

STM32F407G-DISC1

Discovery kit with STM32F407VG MCU * New order code STM32F407G-DISC1 (replaces STM32F4DISCOVERY)

ACTIVE
Product is in mass production

Part Number : STM32F4DISCOVERY
Commercial Part Number : STM32F407G-DISC1

Unit Price (US\$) : 19.9
Mounted Device : STM32F407VGT6

The STM32F4DISCOVERY Discovery kit leverages the capabilities of the STM32F407 high-performance microcontrollers, to allow users to develop audio applications easily. It includes an ST-LINK/V2-A embedded debug tool, one ST-MEMS digital accelerometer, one digital microphone, one audio DAC with integrated class D speaker driver, LEDs, push-buttons, and a USB OTG Micro-AB connector. Specialized add-on boards can be connected by means of the

Boards List: 1 item

Overview	Commercial Part No.	Type	Marketing Status	Unit Price (US\$)	Mounted Device
	STM32F407G-DISC1	Discovery Kit	Active	19.9	STM32F407VGT6

DEDICATED
FILTERS

DESCRIPTION
&
INFORMATION

PRODUCT
LIST

Board selector

New Project from a Board

MCU/MPU Selector | **Board Selector** | Example Selector | Cross Selector

Board Filters

- Commercial Part Number:
-
- LED
- Audio Line Input
- Audio Line Output
- Audio Processor
- Microphone
- ST-LINK
- Potentiometer
- Power Supply
- RS-232
- RS-485
- Speaker
- Touch Feature
- USB Port
- Cryptography
- SMPs
- Debug Connector
- Wireless Interface
- Other on-board Feature

Features Large Picture Docs & Resources [Datasheet](#) [Buy](#) [Start Project](#)

STM32F4 Series

STM32F407G-DISC1 ACTIVE Product is in mass production

Part Number : STM32F4DISCOVERY
Commercial Part Number : STM32F407G-DISC1
Unit Price (US\$) : 19.9
Mounted Device : STM32F407VGT6

The STM32F4DISCOVERY Discovery kit leverages the capabilities of the STM32F407 high-performance microcontrollers, to allow users to develop audio applications easily. It includes an embedded debug tool, one ST-MEMS digital microphone, one digital microphone, one audio DAC with a speaker driver, LEDs, push-buttons, and a USB connector. Add-on boards can be connected by means of the header connectors.

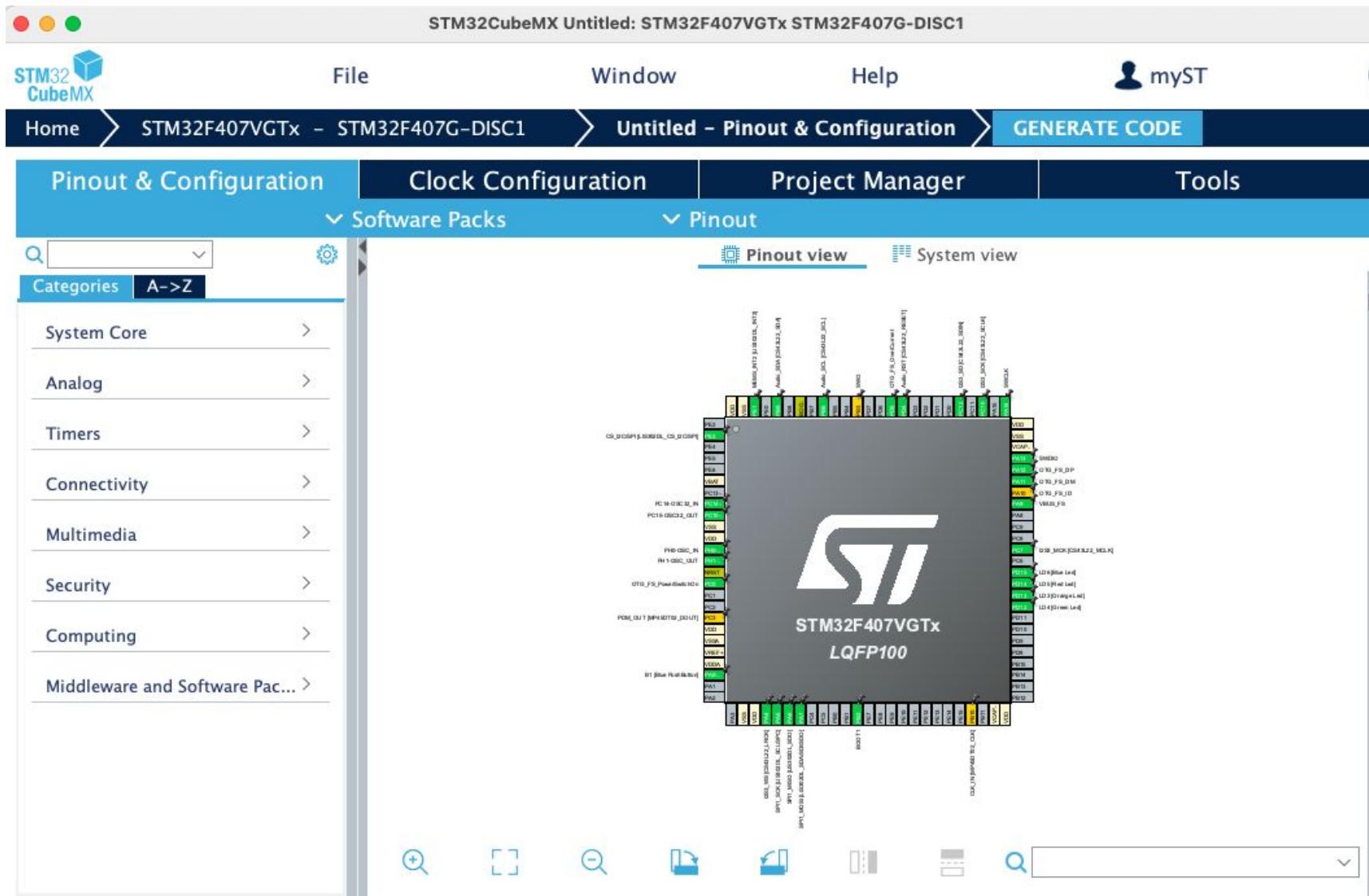
Board Project Options: STM32F407G-DISC1

Initialize all peripherals with their default Mode ?

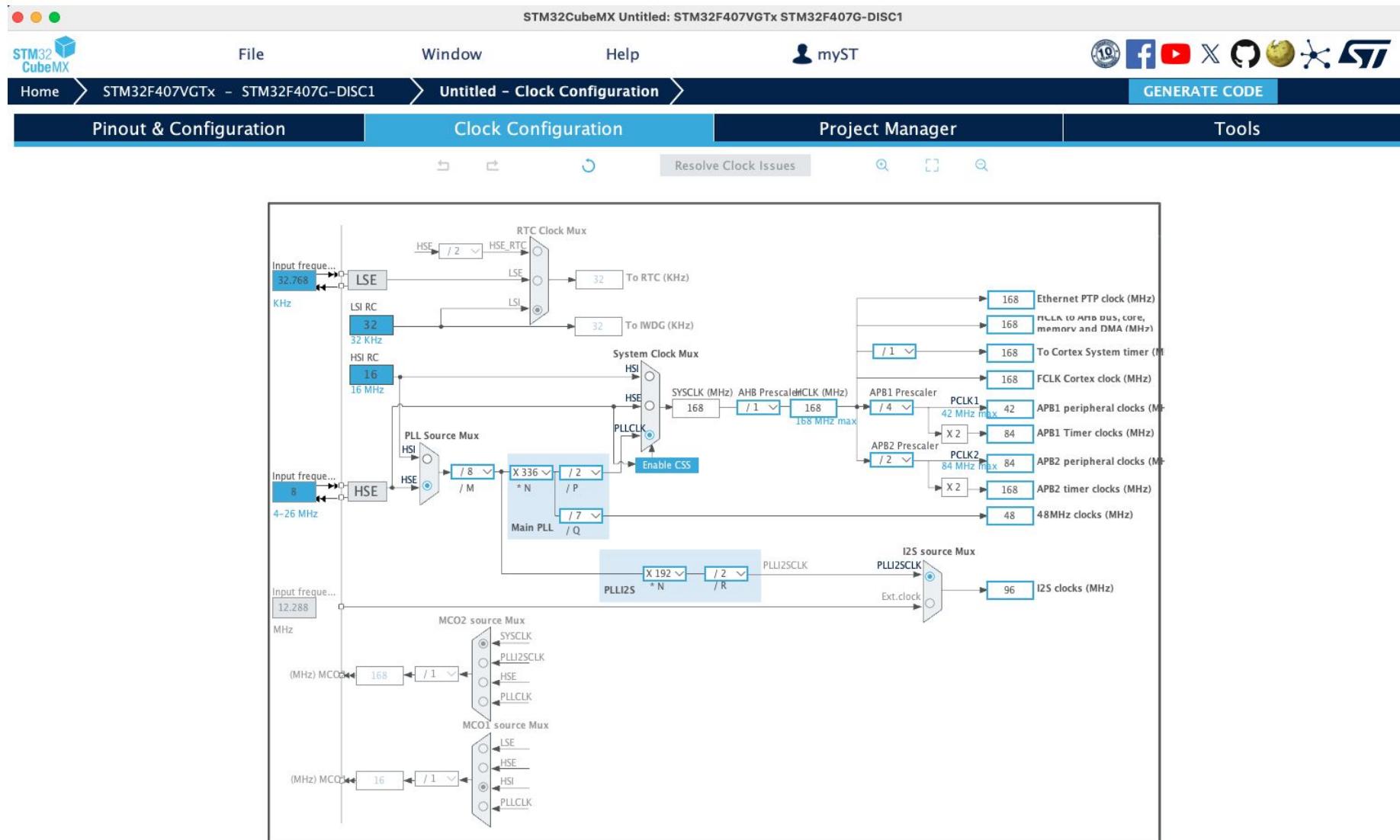
Boards List: 1 item [Export](#)

*	Overview	Commercial Part No	Type	Marketing Status	Unit Price (US\$)	Mounted Device
★		STM32F407G-DISC1	Discovery Kit	Active	19.9	STM32F407VGT6

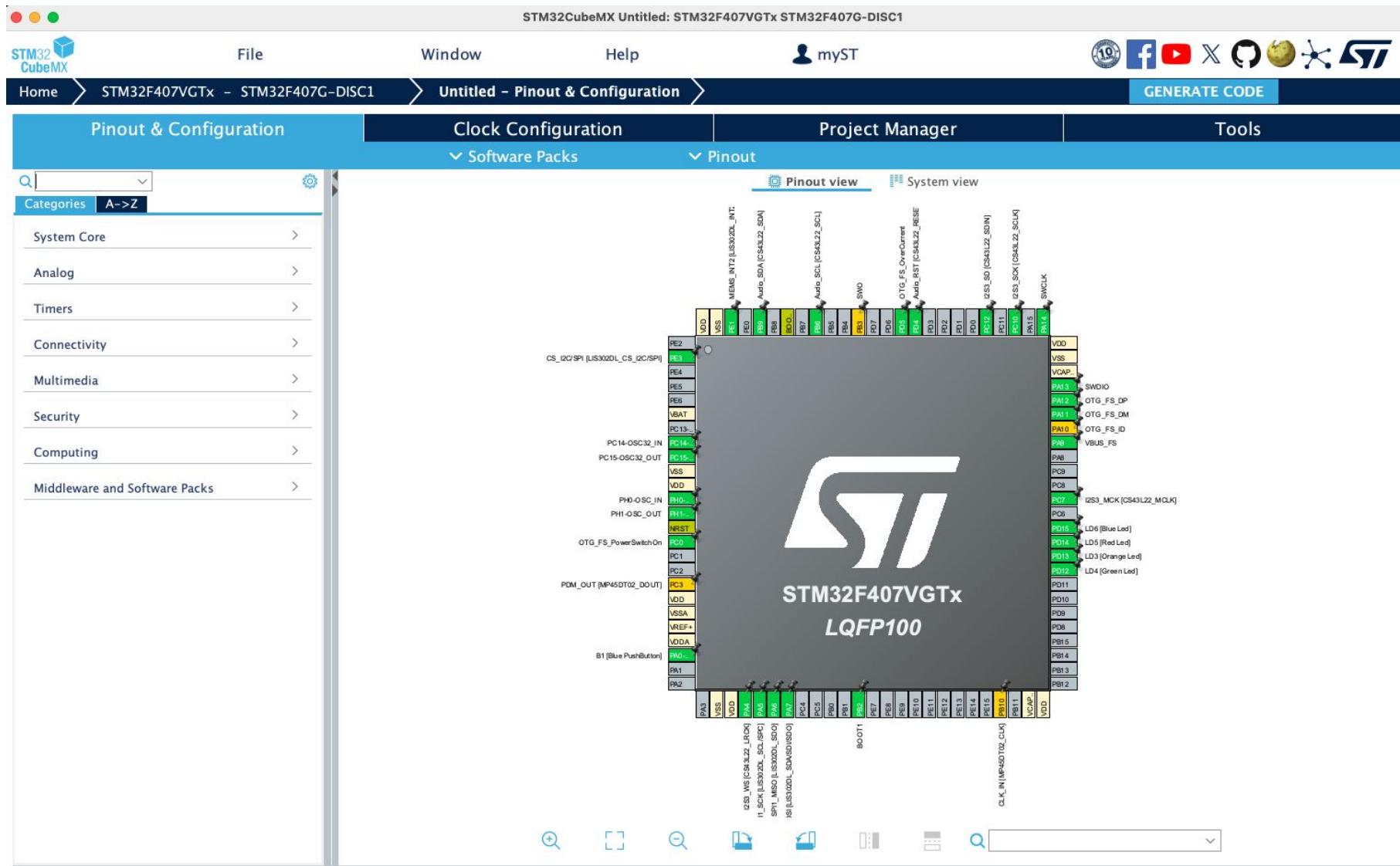
Pinout configuration



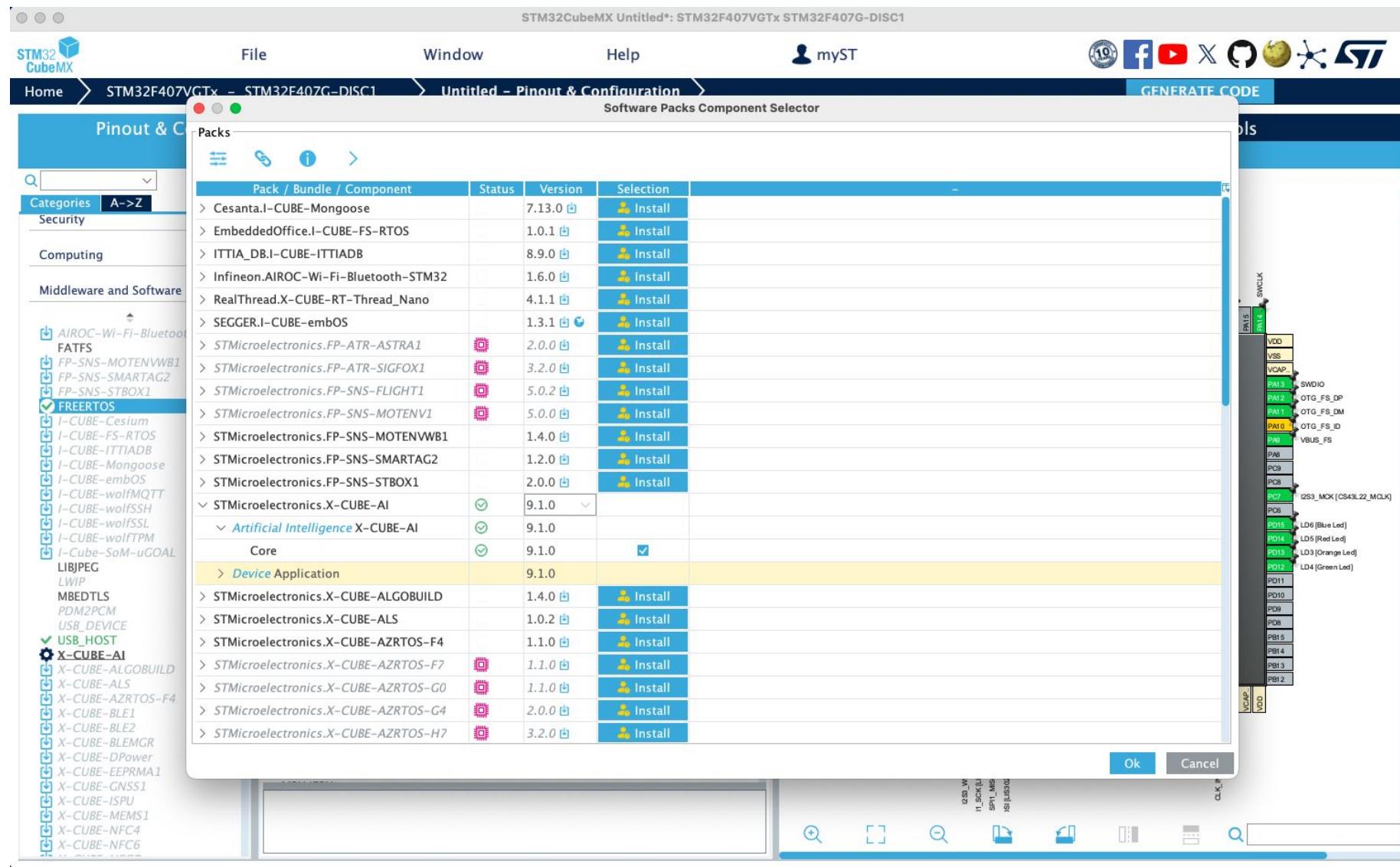
Clock-tree configuration



Peripheral parameters



Add Expansion Middleware



MCU/MPU selection for AI application



The screenshot displays the STM32CubeMX software interface for the STM32F407VGTx. The top menu includes File, Window, Help, myST, GENERATE CODE, and various icons. The main tabs are Pinout & Configuration, Clock Configuration, Project Manager, and Tools. The Pinout & Configuration tab is active, showing the STMMicroelectronics.X-CUBE-AI.9.1.0 Mode and Configuration. The Pinout view on the right shows the physical pin layout for the STM32F407VGTx LQFP100 package, with pins labeled from PA0 to PA15, PB0 to PB15, PC0 to PC15, PD0 to PD15, PE0 to PE15, and PF0 to PF15. The Project Manager tab shows a list of categories including Categories, A-Z, and various software packs like FP-SNS-STBOX1, FREERTOS, I-CUBE-Cesium, I-CUBE-FS-RTOS, I-CUBE-ITTIADB, I-CUBE-Mongoose, I-CUBE-embOS, I-CUBE-wolfMQTT, I-CUBE-wolfSSH, I-CUBE-wolfSSL, I-CUBE-wolfTPM, I-CUBE-SoM-uGOAL, LIBJPEG, LWIP, MBEDTLS, PDMPCM, USB_DEVICE, and USB_HOST (which is checked). The Clock Configuration tab shows the mode as Artificial Intelligence X-CUBE-AI. The Tools tab is also visible.

Visualize Model



STM32CubeMX Untitled*: STM32F407VGTx STM32F407G-DISC1

File Window Help myST GENERATE CODE

Home > STM32F407VGTx - STM32F407G-DISC1 > Untitled - Pinout & Configuration >

Pinout & Configuration Clock Configuration Project Manager Tools

STM32CubeMX - STM32F407VGTx - STM32F407G-DISC1

Categories A-Z

- Timers
- Connectivity
- Multimedia
- Security
- Computing
- Middleware and Software Packs
 - AIROC-Wi-Fi-Bluetooth-STM32 FATFS
 - FP-SNS-MOTENVB1
 - FP-SNS-SMARTAG2
 - FP-SNS-STBOX1
 - FREERTOS
 - I-CUBE-Cesium
 - I-CUBE-FS-RTOS
 - I-CUBE-ITTIADB
 - I-CUBE-Mongoose
 - I-CUBE-embOS
 - I-CUBE-wolMQTT
 - I-CUBE-wolfSSH
 - I-CUBE-wolfSSL
 - I-CUBE-wolFTP
 - I-Cube-SaM-uGOAL LIBJPEG
 - LWIP
 - MBEDTLS
 - PDM2PCM
 - USB_DEVICE
 - USB_HOST
 - X-CUBE-AI
 - X-CUBE-ALGOBUILD
 - X-CUBE-ALS
 - X-CUBE-AZRTOS-F4
 - X-CUBE-BLE1

Artificial Intelligence X-CUBE-AI.9.1.0 Mode and Configuration

Mode sound_classifier

LUTZ ROEDER'S NETRON

Open Model...

Reset Configuration Develop

Main sound_classifier +

Compression: None

Validation inputs: Random number

Validation outputs: None

Complexity: -
Used Flash: - (.00 B over 1024.00 KiB Internal)
Used Ram: - (.00 B over 192.00 KiB Internal)
Achieved compression: -
Analysis status: -

Analyze Validate on desk... Validate on target

Pinout view System view

STM32F407VGTx LQFP100

Visualize Model

The screenshot shows the STM32CubeMX software interface. At the top, the title bar reads "STM32CubeMX Untitled*: STM32F407VGTx STM32F407G-DISC1". The menu bar includes "File", "Window", "Help", and a user account icon. To the right are social media links for Facebook, YouTube, and X, along with the ST logo. Below the menu is a breadcrumb navigation: "Home > STM32F407VGTx - STM32F407G-DISC1 > Untitled - Pinout & Configuration >". A "GENERATE CODE" button is located in the top right corner. The main workspace is divided into four tabs: "Pinout & Configuration" (selected), "Clock Configuration", "Project Manager", and "Tools". The "Pinout & Configuration" tab shows a detailed pinout view for the STM32F407VGTx LQFP100 package, listing pins like PB0, PB1, PB2, etc., with their corresponding functions such as "I2C_1_SDA", "VDD", "VSS", "VCAP", etc. The central area displays a "sound_classifier" neural network model graph with nodes: input, Dense (kernel 512x10, bias 10), ReLU, Dense (kernel 10x3, bias 3), Softmax, and dense_1. A "NODE PROPERTIES" panel shows settings for the selected Dense layer, including type (Dense), module (tensorflow.keras.layers), name (dense), activation (relu), and units (10). The "ATTRIBUTES" panel lists attributes like kernel_initializer (RandomUniform(minval: -0.05, maxval: 0.05, seed: null)) and trainable (true). The "INPUTS" panel shows validation inputs and outputs. The "Clock Configuration" and "Project Manager" tabs are also visible in the header.

Code generation

STM32CubeMX EMLEC_v1.ioc*: STM32F407VGTx STM32F407G-DISC1

File Window Help myST

Home > STM32F407VGTx - STM32F407G-DISC1 > EMLEC_v1.ioc - Project Manager >

GENERATE CODE

Pinout & Configuration **Clock Configuration** **Project Manager** **Tools**

Project

- Project Settings
 - Project Name: EMLEC_v1
 - Project Location: /Users/kithminrw/STM32Cube
 - Application Structure: Advanced Do not generate the main()

Code Generator

- Toolchain Folder Location: /Users/kithminrw/STM32Cube/EMLEC_v1/
 - Toolchain / IDE: STM32CubeIDE Generate Under Root

Advanced Settings

- Linker Settings
 - Minimum Heap Size: 0x800
 - Minimum Stack Size: 0x800
- Thread-safe Settings
 - Cortex-M4NS
 - Enable multi-threaded support
 - Thread-safe Locking Strategy: Default – Mapping suitable strategy depending on RTOS selection.
- Mcu and Firmware Package
 - Mcu Reference: STM32F407VGTx
 - Firmware Package Name and Version: STM32Cube FW_F4 V1.28.0
 - Use Default Firmware Location
 - Firmware Relative Path: /Users/kithminrw/STM32Cube/Repository/STM32Cube_FW_F4_V1.28.0



Power consumption calculator



STM32CubeMX EMLEC_v1.ioc: STM32F407VGTX STM32F407G-DISC1

File Window Help myST GENERATE CODE

Home > STM32F407VGTX - STM32F407G-DISC1 > EMLEC_v1.ioc - Tools >

Pinout & Configuration | Clock Configuration | Project Manager | Tools

PCC

STM32F407VGTX
T_A 25°C / V_{DD} 3.3V
Sequence Generator
Why Use Sequence Generator ?
Sequence Generator allows to quickly generate 2 steps in high and low power modes in typical conditions without creating each step manually using New Step button.
What is the Default Sequence ?
Default Sequence gathers a 0.1 ms step in RUN at max CPU frequency getting highest performance and consumption, and a 0.9 ms step in STOP with the lowest consumption in typical conditions with V_{DD} 3.3V at T_A 25°C.
Generator
Generate RUN + STOP
Back To Default Sequence
Disable Auto Refr... Auto Refresh ON
.Results at T_A 25°C / 3.3V
Sequence Type Default
Typ. Average Current 4.85 mA
DMIPS 210
Sequence Configuration
RUN Consumption 46 mA
High Power Mode RUN
CPU Frequency 168 MHz
STOP Consumption 280 µA
Low Power Mode STOP
RUN Step 0.1 ms / STOP Step 0.9 ms

CAD

New Step Step Sequence Default Sequence Table

Step	Mode	Vdd	Range/Scale	Memory	CPU/Bus Freq	Clock Config	Peripherals	Step Current	Duration
1	RUN	3.3	Scale1-High	FLASH	168 MHz	HSE PLL		46 mA	0.1 ms
2	STOP	3.3	No Scale	n/a	0 Hz	Regulator_LP Flash...		280 µA	0.9 ms

Sequence Information Notes

- 1/ Manual change in Sequence Table will disable the Auto Refresh of the Sequence Generator
- 2/ Default sequence = RUN at max CPU frequency + STOP with the lowest consumption
- 3/ Default sequence is an example which does not match any pinout, configuration nor clock settings and can be directly edited for reuse or removed
- 4/ PCC sequence has no impact on code generation

Display Selection Select your Preferred Display Plot: All Steps

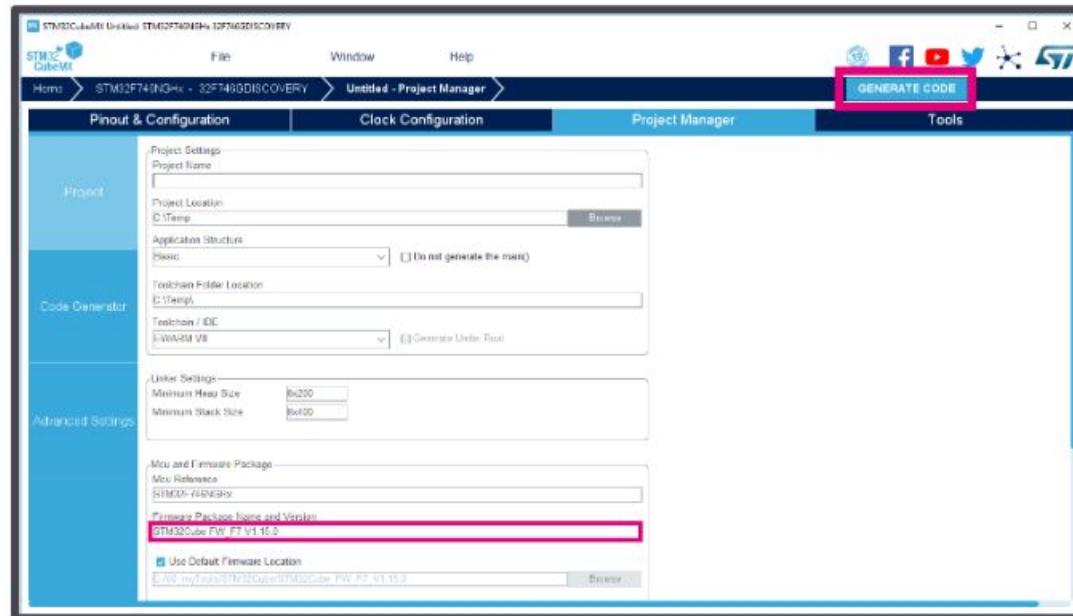
Consumption Profile by Step

The graph plots Consumption (mA) on the Y-axis (0 to 50) against Time (ms) on the X-axis (0.00 to 1.05). A black step function shows a high current level (around 46 mA) during the RUN step (from ~0.05 ms to ~0.55 ms) and a much lower current level (around 280 µA) during the STOP step (from ~0.55 ms to ~1.05 ms). A legend indicates the black line represents Idd by Step and the blue line represents Average Current.

Sequence Time / Ta Max 1 ms / 98.47 °C
Battery Life Estimation 29 days, 4 hours

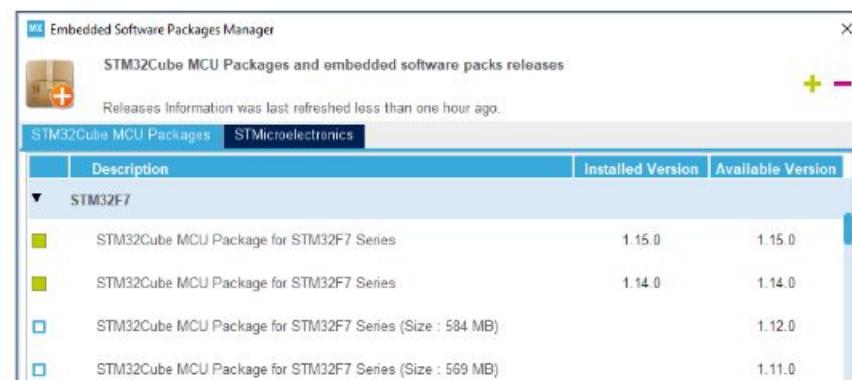
Average Consumption 4.85 mA
Average DMIPS 210 DMIPS

STM32CubeMX repository



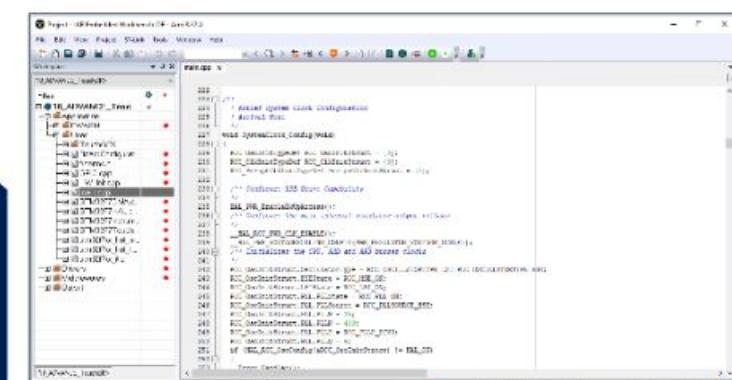
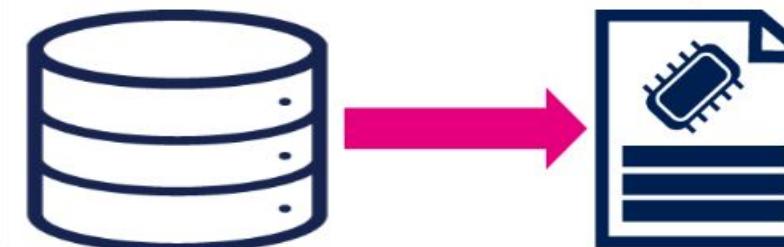
After pressing “GENERATE CODE”:

1. STM32CubeMX grabs necessary peripheral drivers based on your pinout/peripheral configuration from STM32Cube MCU Package in STM32CubeMX Repository
2. STM32CubeMX grabs necessary middleware based on your middleware configuration from STM32Cube MCU Package in STM32CubeMX repository
3. Generate IDE project



Description	Installed Version	Available Version
STM32F7		
STM32Cube MCU Package for STM32F7 Series	1.15.0	1.15.0
STM32Cube MCU Package for STM32F7 Series	1.14.0	1.14.0
STM32Cube MCU Package for STM32F7 Series (Size : 584 MB)	1.12.0	
STM32Cube MCU Package for STM32F7 Series (Size : 569 MB)	1.11.0	

STM32CubeMX
Repository



```
/* File: main.c
 * Description: Main application code configuration
 */
#include "main.h"

void main(void)
{
    /* Initialize system clock configuration
    * ...
    */

    /* Initialize USART1
    * ...
    */

    /* Configure USART1 baud rate
    * ...
    */

    /* Start USART1 transmission
    * ...
    */
}
```

STM32 CubeIDE



The screenshot shows the STM32CubeIDE interface with the following details:

- Project Explorer:** Displays the project structure for "EMLEC_v1".
- Code Editor (main.c):** Shows the main application code. Key snippets include:
 - /* USER CODE BEGIN 1 */
 - /* MCU Configuration--
 - /* Reset of all peripherals, Initializes the Flash interface and the Systick. */ HAL_Init();
 - /* USER CODE BEGIN Init */
 - /* USER CODE END Init */
 - /* Configure the system clock */ SystemClock_Config();
 - /* USER CODE BEGIN SysInit */
 - /* USER CODE END SysInit */
 - /* Initialize all configured peripherals */ MX_GPIO_Init(); MX_I2C1_Init(); MX_I2S3_Init(); MX_SPI1_Init(); MX_USB_HOST_Init();
 - /* USER CODE BEGIN 2 */
 - /* Infinite loop */
 - /* USER CODE BEGIN WHILE */ while (1)
 - /* USER CODE END WHILE */ MX_USB_HOST_Process();
 - /* USER CODE BEGIN 3 */
 - /* USER CODE END 3 */
- Outline:** Lists symbols and functions defined in the code.
- Problems:** Shows 0 errors, 1 warning, 0 others.
- Build Analyzer:** Provides static stack analysis and cyclomatic complexity information.
- Memory Regions:** Displays memory usage details.

STM32CubeIDE - STMicroelectronics (<https://www.st.com>)

History of STM32 CubeIDE



History

a atollic
TrueSTUDIO®

ST a atollic
TrueSTUDIO® for STM32

STM32
CubeMX



STM32 Cube.AI



Load your trained neural network model

or pick one from STM32 model zoo



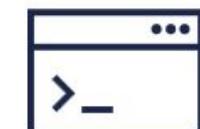
Optimize and validate your NN model



STM32Cube.AI for desktop



STM32Cube ecosystem



Command Line Interface

STM32Cube.AI Developer Cloud



Online platform



REST API



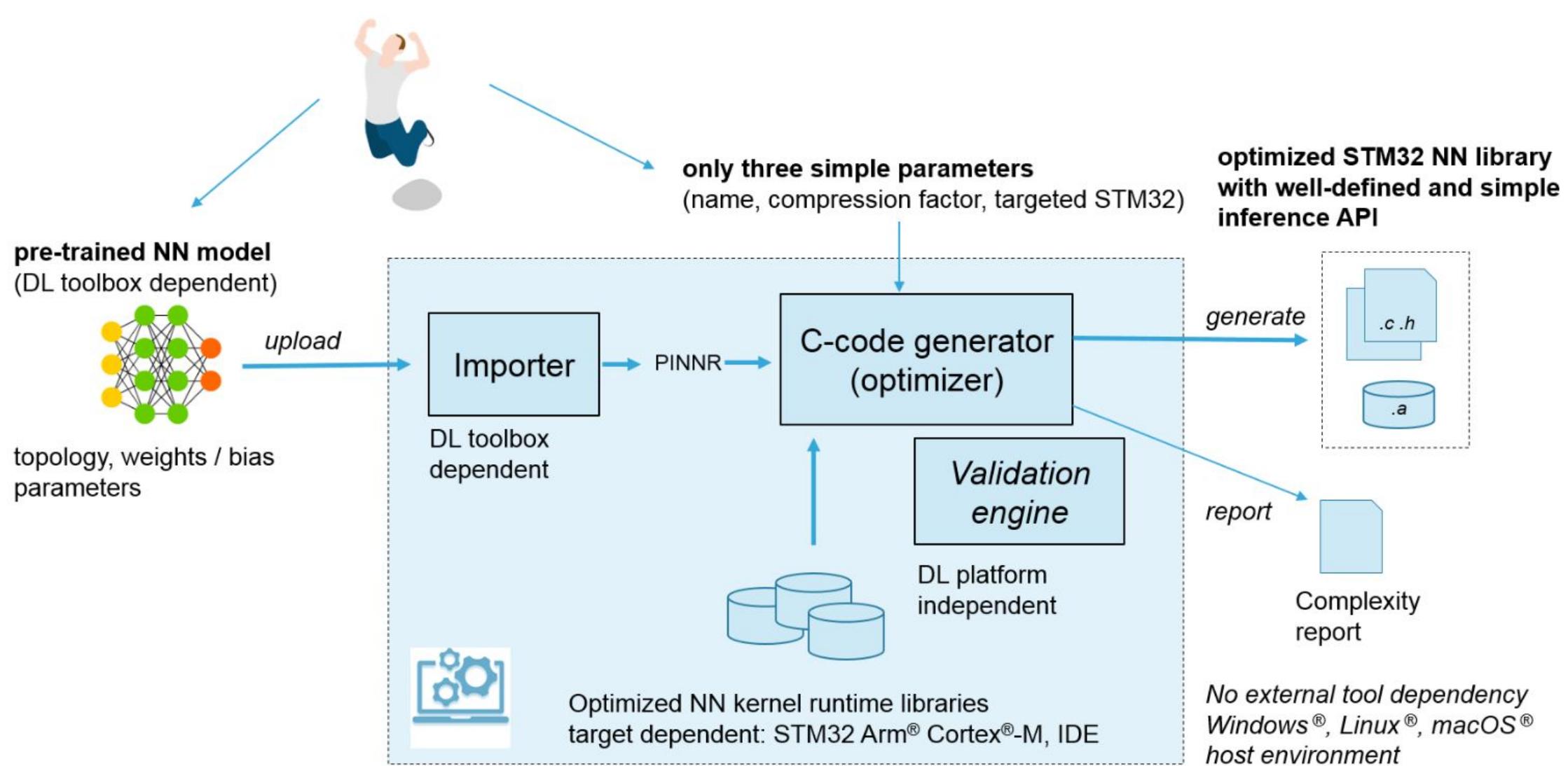
Benchmarking tool

Generate optimized code for STM32

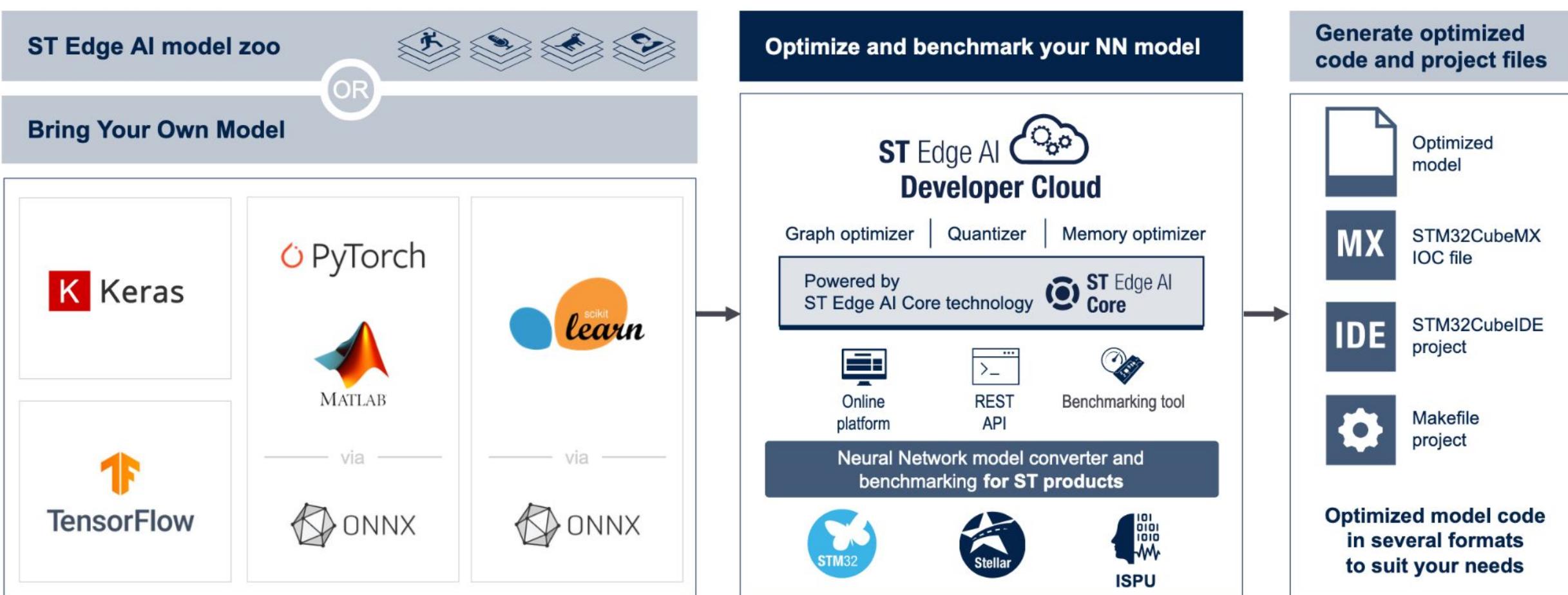


Optimized model
code for STM32

X-CUBE-AI core engine



STM32 Cube.AI

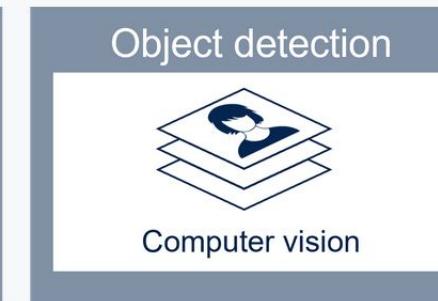
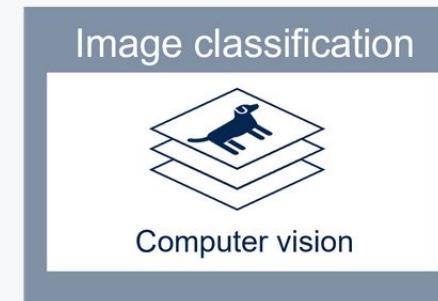
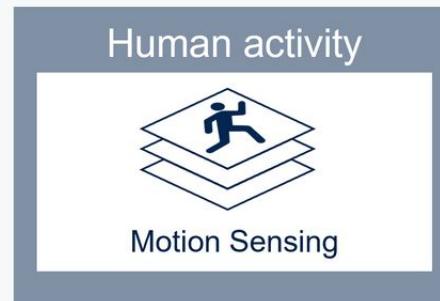


[AI:X-CUBE-AI documentation - stm32mcu](#)

STM32 model zoo

Find the best edge AI model

- A large collection of application-oriented models ready for re-training
- Scripts to easily retrain any model from user datasets
- Application code examples automatically generated from user AI model



- ▶ **Image classification (IC):** Models: EfficientNet, MobileNet v1, MobileNet v2, Resnet v1 including with hybrid quantization, SqueezeNet v1.1, STMNIST.
- ▶ **Object detection (OD):** Models: ST SSD MobileNet v1, Tiny YOLO v2, SSD MobileNet v2 fpn lite, ST Yolo LC v1.
- ▶ **Human activity recognition (HAR):** Models: CNN IGN, and CNN GMP for different settings.
- ▶ **Audio event detection (AED):** Models: Yamnet, MiniResnet, MiniResnet v2.
- ▶ **Hand posture recognition (HPR):** Model: ST CNN 2D Hand Posture.

STMicroelectronics – STM32 model zoo

Welcome to STM32 model zoo!

The STM32 AI model zoo is a collection of reference machine learning models that are optimized to run on STM32 microcontrollers. Available on GitHub, this is a valuable resource for anyone looking to add AI capabilities to their STM32-based projects.

[STMicroelectronics/stm32ai-modelzoo: AI Model Zoo for STM32 devices](https://github.com/STMicroelectronics/stm32ai-modelzoo)

ST Edge AI Developer Cloud



The screenshot shows the ST Edge AI Developer Cloud interface. On the left, the "Your model library" section allows users to upload new models via a yellow "New model" button or a file browser. It specifies supported formats (Keras, ONNX, TFLite) and a max file size of 256 MiB. On the right, the "Pick a model from ST Model Zoo" section lists three pre-trained models: CNN2D_ST_HandPosture_8classes_hand_posture_ST_VL53L5CX_ha..., CNN2D_ST_HandPosture_8classes_hand_posture_ST_VL53L8CX_ha..., and cnn_8x8x8_ispu_wand_ST_LSM6DSO16IS_ispu_wand_dataset.h5. Each entry includes a red "K" icon, a description, a dataset used, and an "Import" button.

ST Edge AI Developer Cloud

Your model library

New model

Drop your model or click here to open a file browser.
Supported models are Keras, ONNX and TFLite (.h5, .hdf5, .keras, .onnx, .tflite)
Max file size: 256 MiB

Show previously quantized models

Date ↓ Name Size

Pick a model from ST Model Zoo

Target: 2 selected | Use case: 6 selected | Neural Network: 42 selected

K CNN2D_ST_HandPosture_8classes_hand_posture_ST_VL53L5CX_ha...
Content Length: 31.09 KiB
Description: CNN2D_ST_HandPosture 8 postures trained on ST_VL53L5CX_handposture_dataset with 8 classes
Use case: hand_posture
Dataset used: ST_VL53L5CX_handposture_dataset

K CNN2D_ST_HandPosture_8classes_hand_posture_ST_VL53L8CX_ha...
Content Length: 72.52 KiB
Description: CNN2D_ST_HandPosture 8 postures trained on ST_VL53L8CX_handposture_dataset with 8 classes
Use case: hand_posture
Dataset used: ST_VL53L8CX_handposture_dataset

K cnn_8x8x8_ispu_wand_ST_LSM6DSO16IS_ispu_wand_dataset.h5
Content Length: 90.02 KiB
Description: CNN1D for air writing recognition that detects the letters I, S, P, and U
Use case: ispu_wand
Dataset used: ST_LSM6DSO16IS_ispu_wand_dataset

<https://stm32ai-cs.st.com/home>

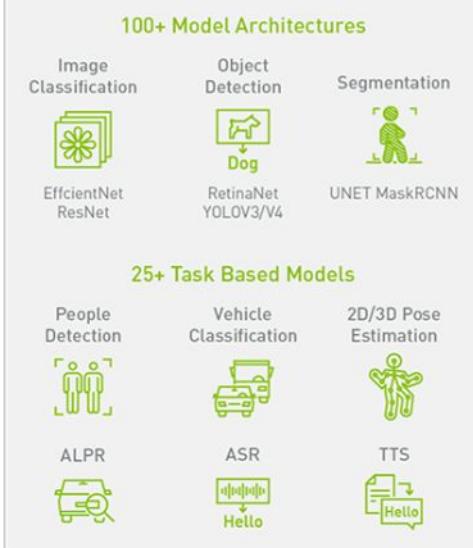
STM32AI - Train, Adapt and Optimize!



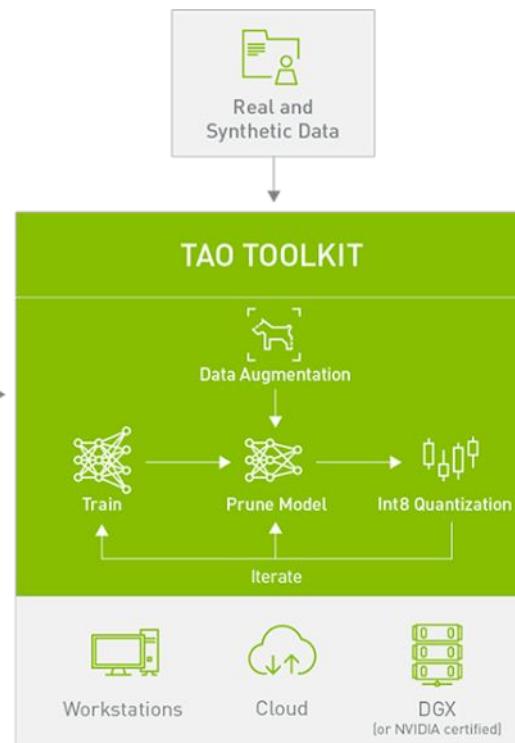
- 1** Bring your own model weights or choose from NVIDIA's library of model architectures or task-based models



OR
Start with NVIDIA Pretrained Models

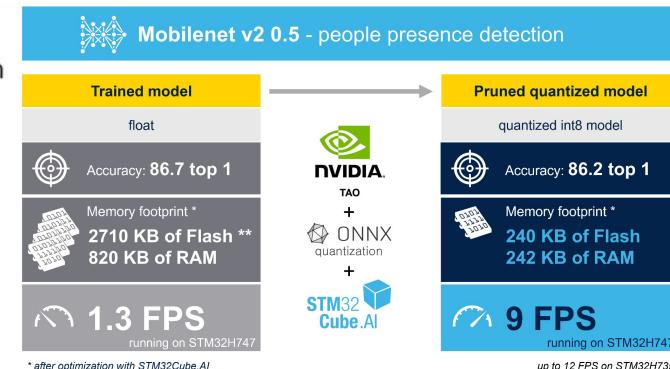
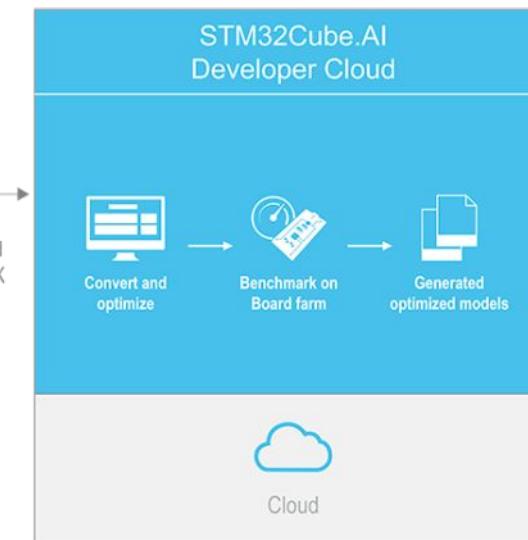


- 2** Quickly train, adapt, and optimize models with your real or synthetic data



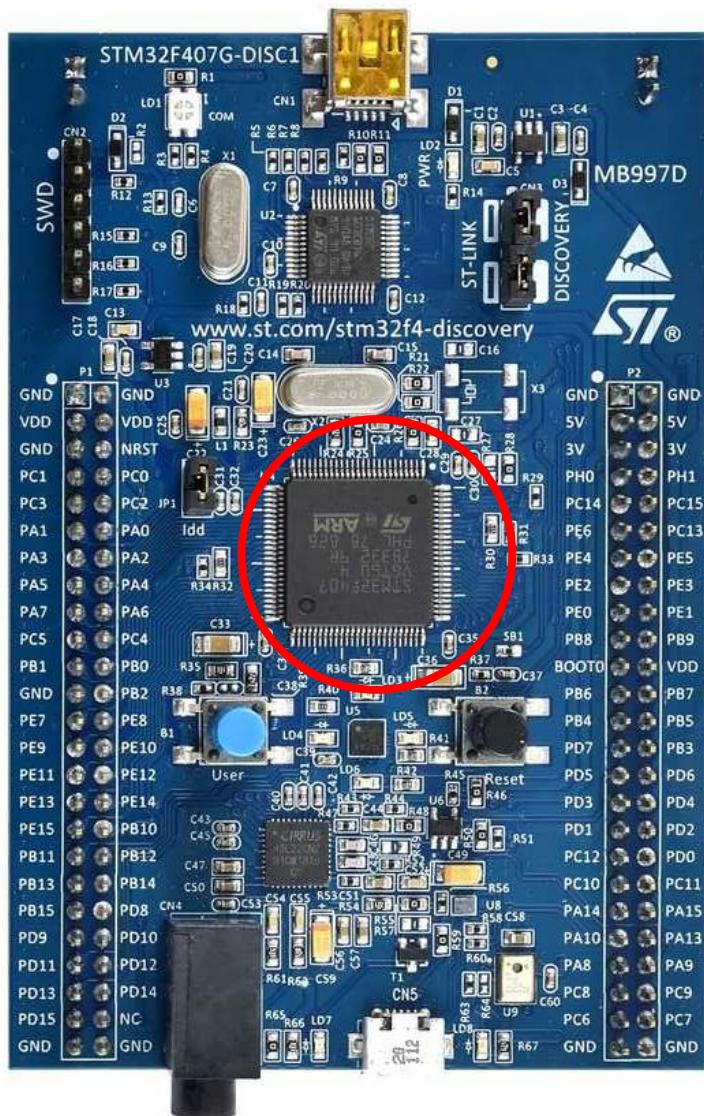
- 3** Integrate your customized models into your application and deploy

NVIDIA TAO Toolkit

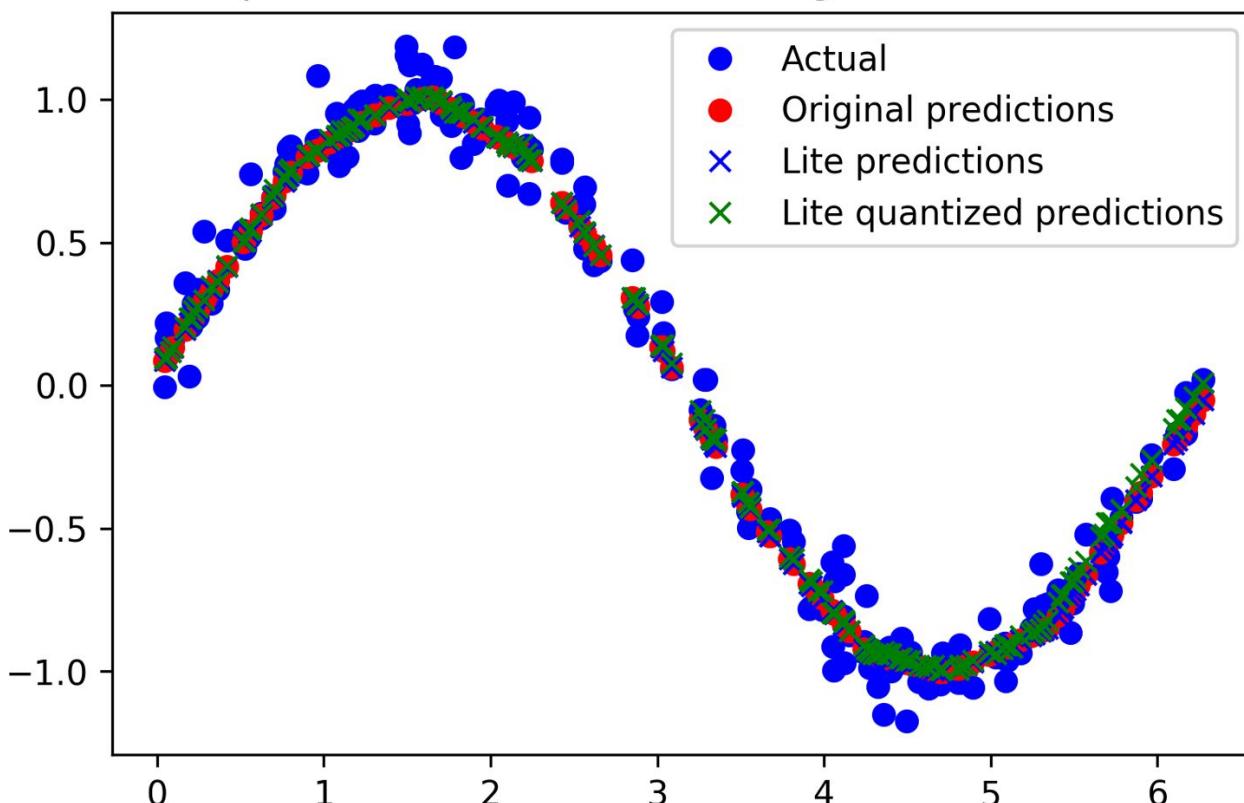


[GitHub - STMicroelectronics/stm32ai-tao: Nvidia TAO \(Train, Adapt, Optimize\) with STM32Cube.AI Developer Cloud](https://github.com/STMicroelectronics/stm32ai-tao)

Case Study 1 - tflite_SineWave

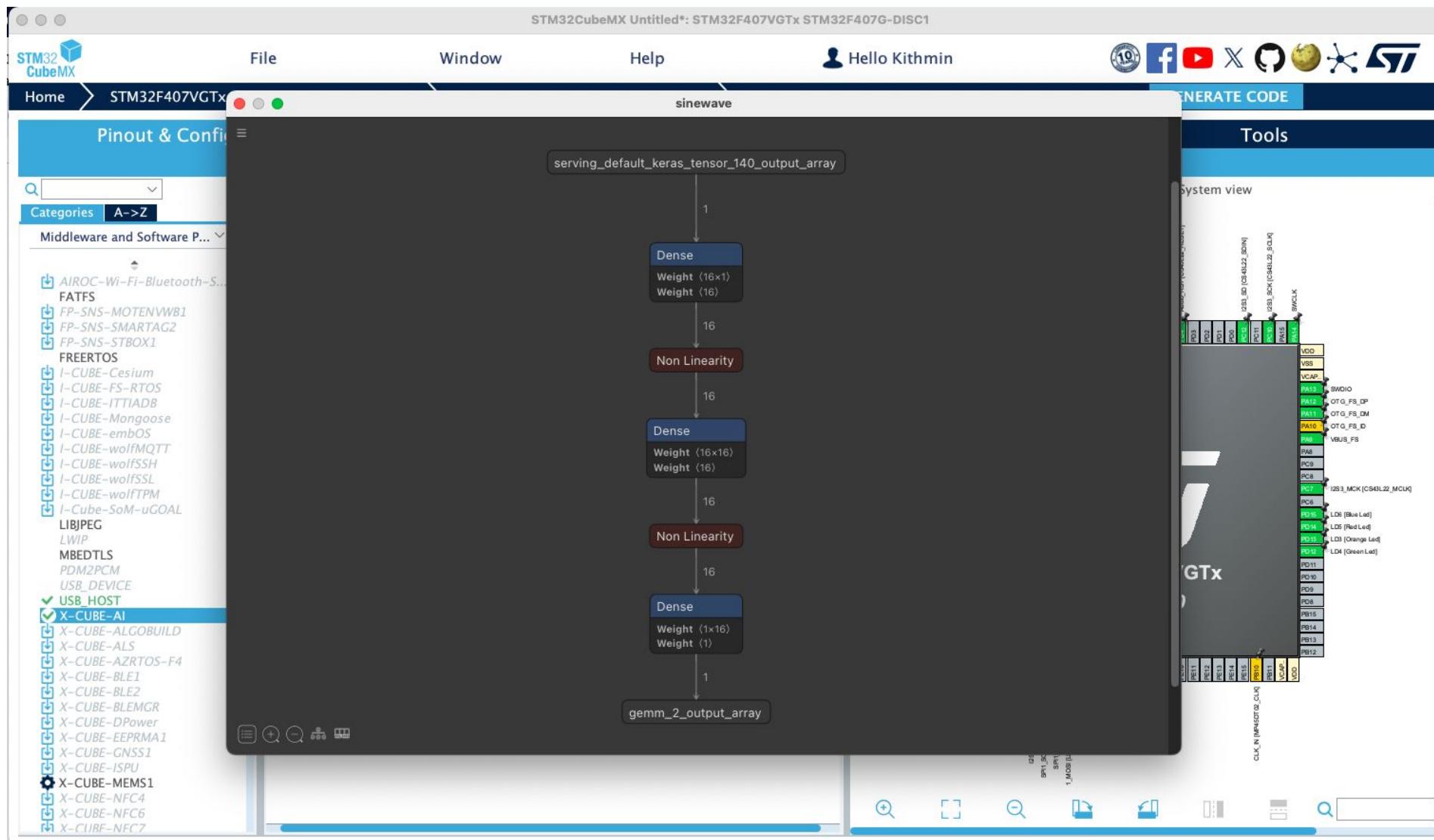


Comparison of various models against actual values

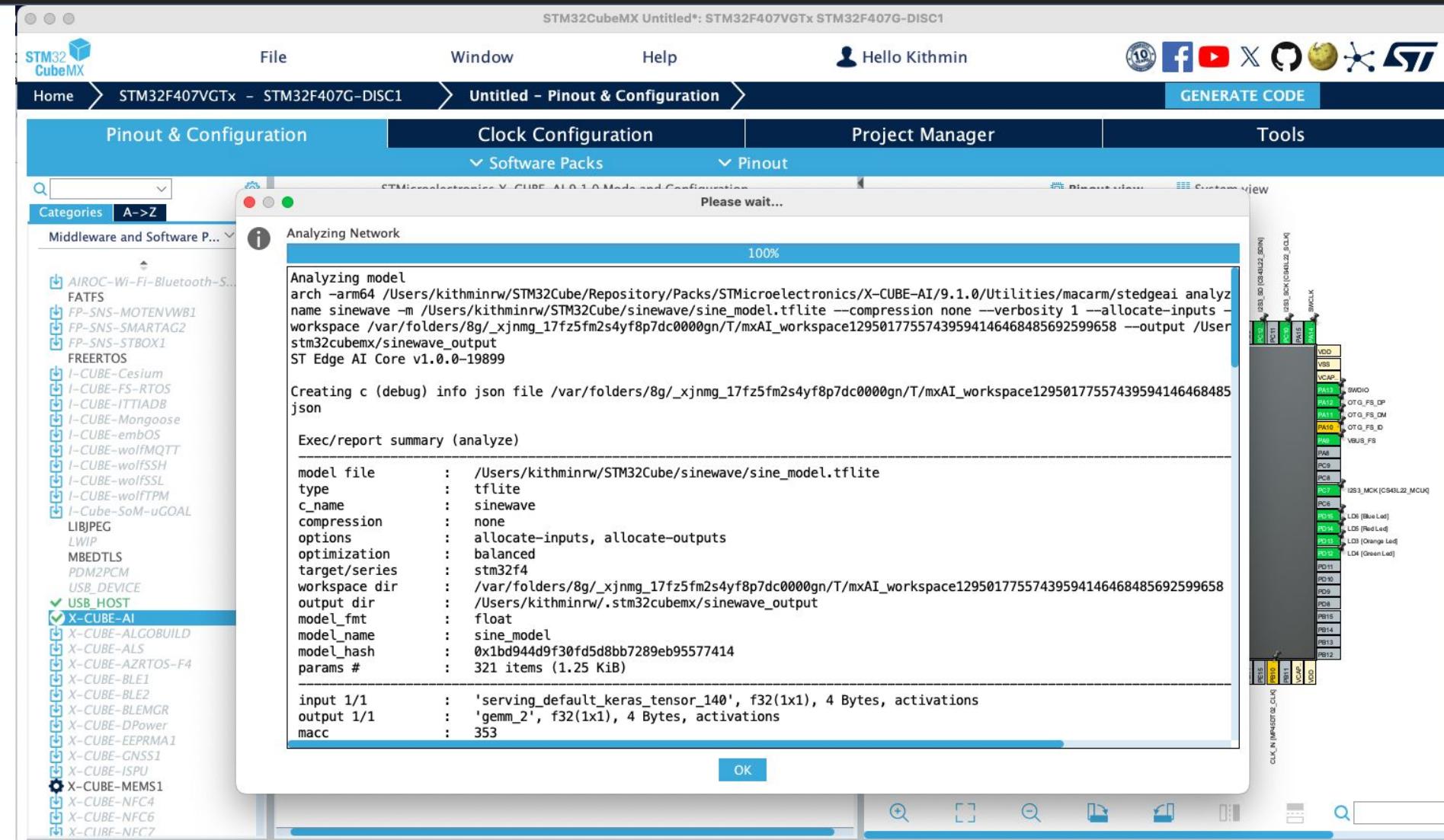


Read - [Chapter 4. The “Hello World” of TinyML: Building and Training a Model](#)
Video - [TinyML: Getting Started with STM32 X-CUBE-AI | Digi-Key Electronics](#)
Training Notebook - [TensorFlow Lite Sinewave Regression Training and Conversion](#) and [Intro to TinyML Part 1: Training a Neural Network for Arduino in TensorFlow | Digi-Key Electronics](#)

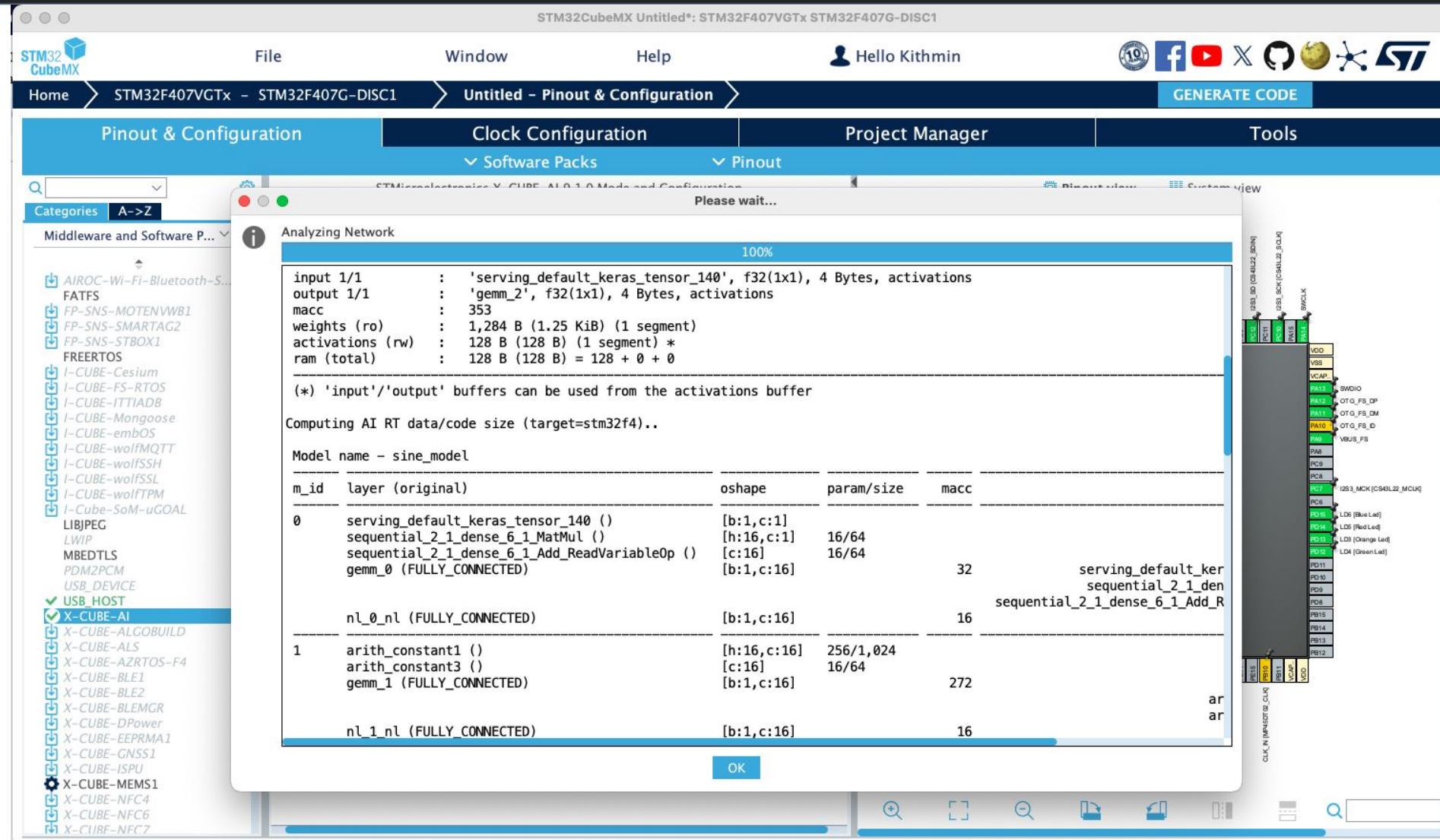
Case Study 1 - tflite_SineWave



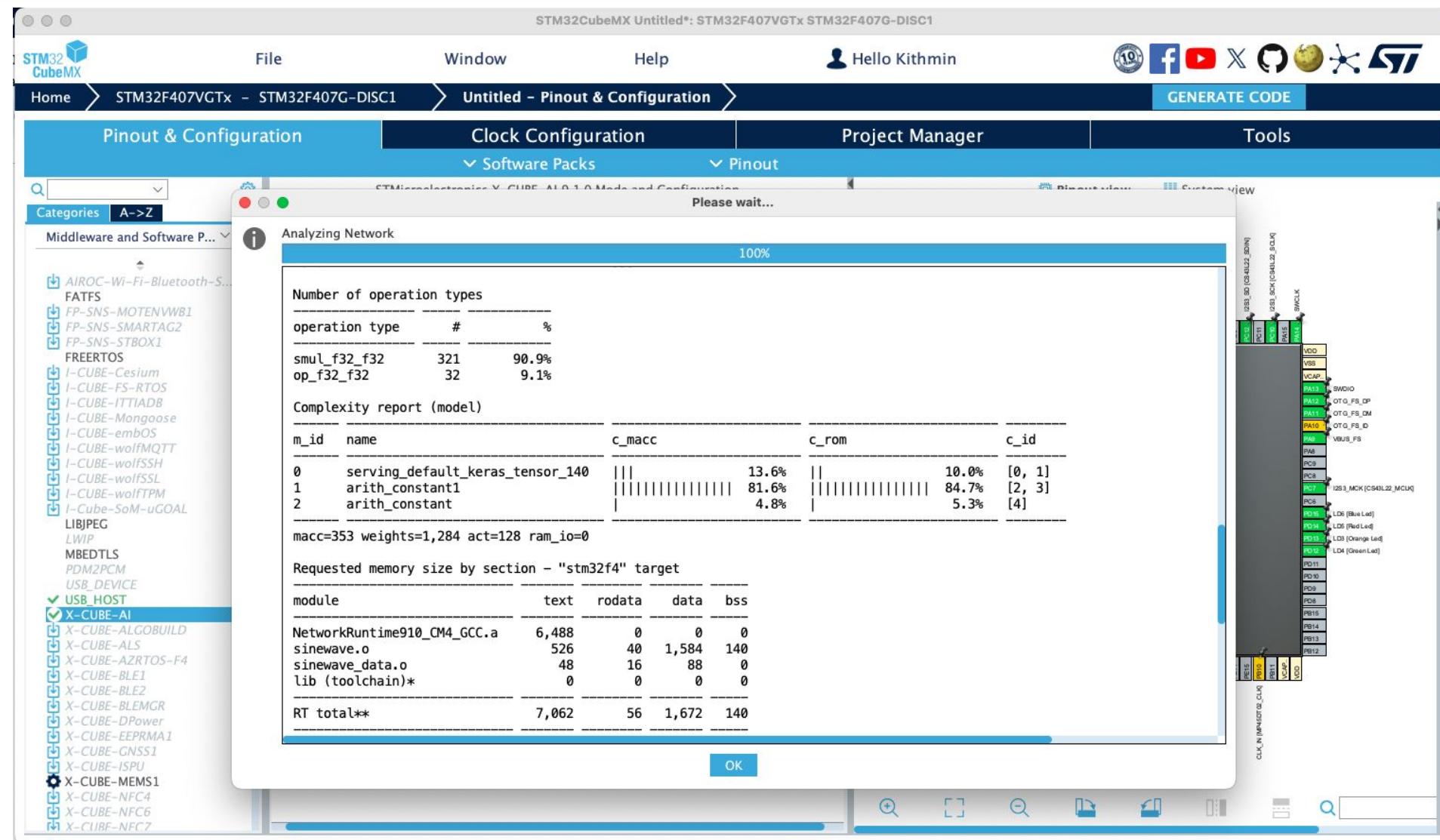
Case Study 1 - tflite_SineWave



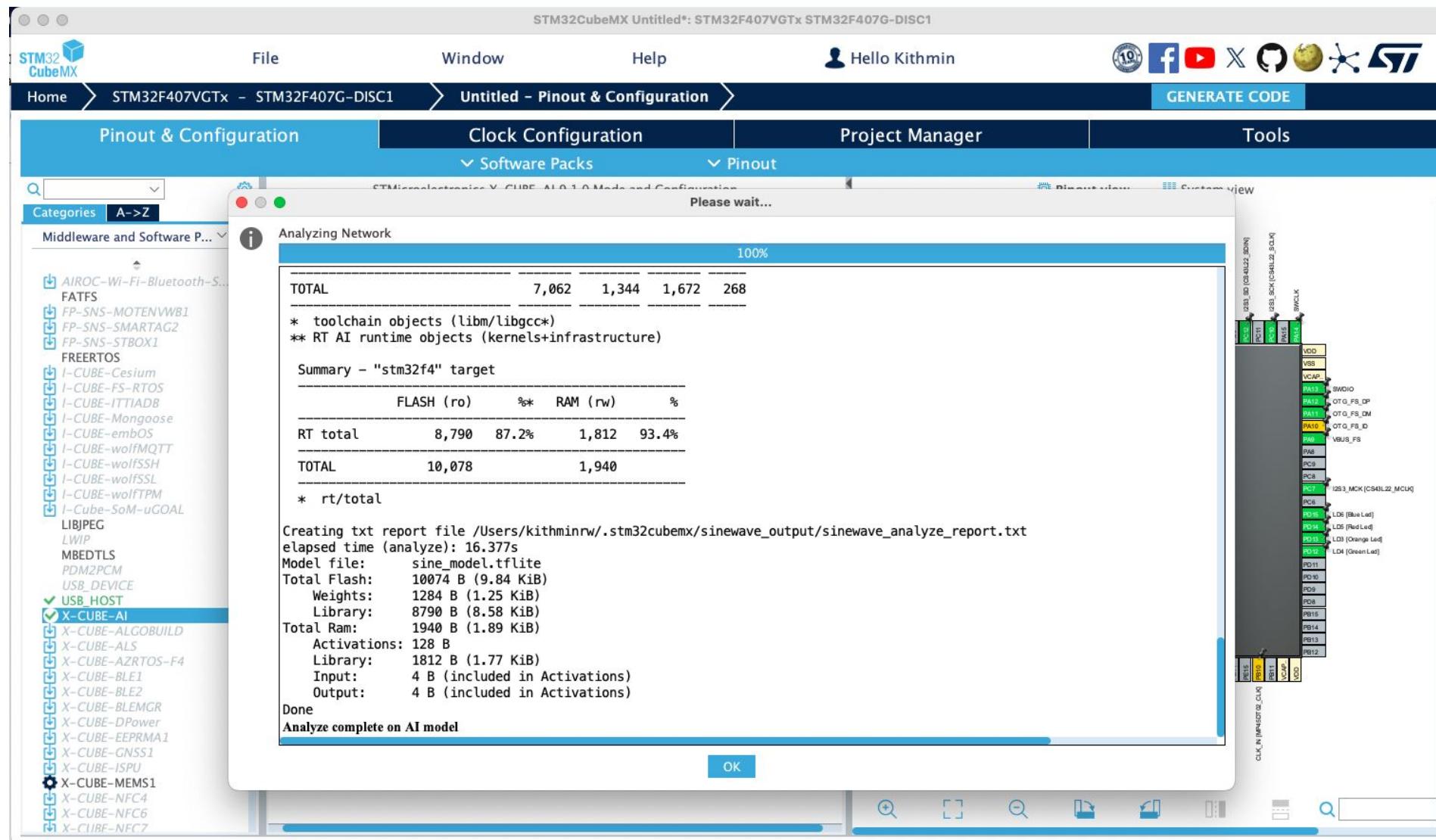
Case Study 1 - tflite_SineWave



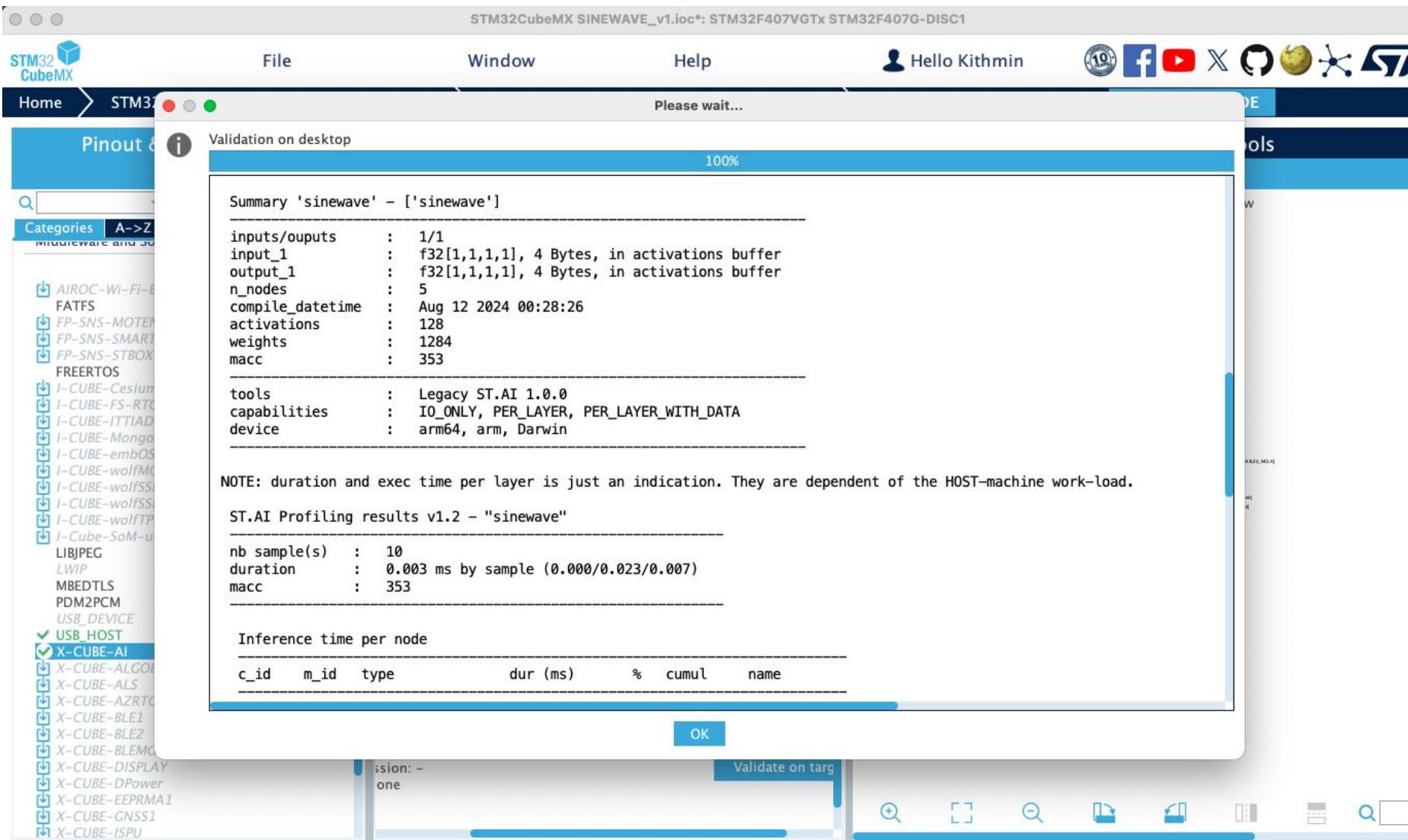
Case Study 1 - tflite_SineWave



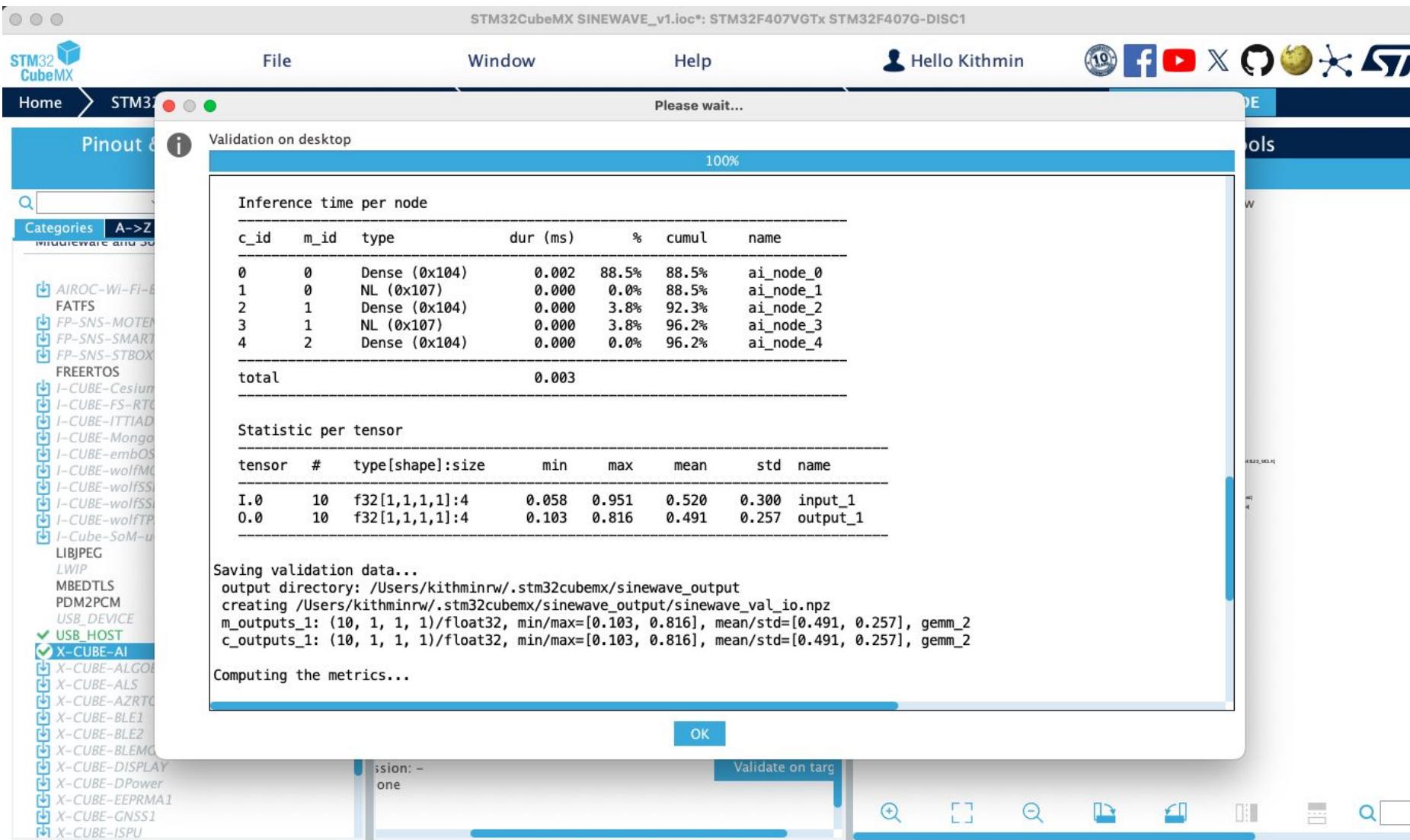
Case Study 1 - tflite_SineWave



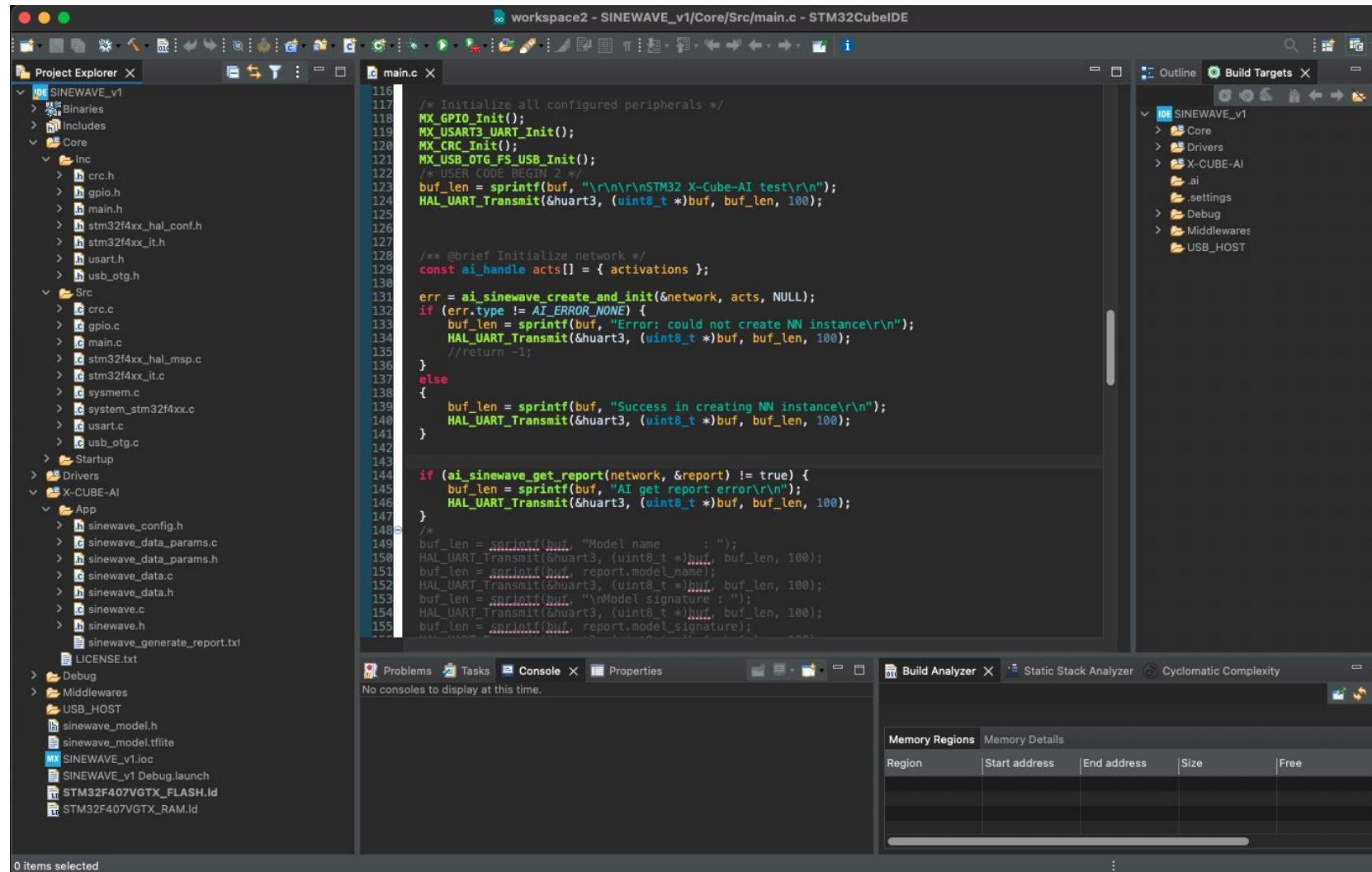
Case Study 1 - tflite_SineWave



Case Study 1 - tflite_SineWave



Case Study 1 - tflite_SineWave

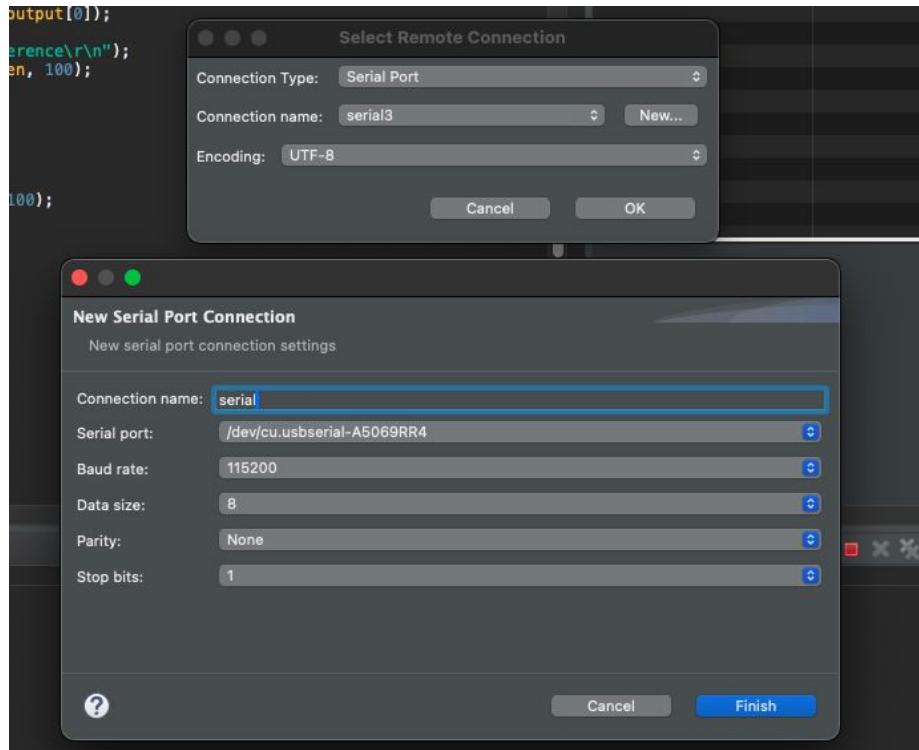
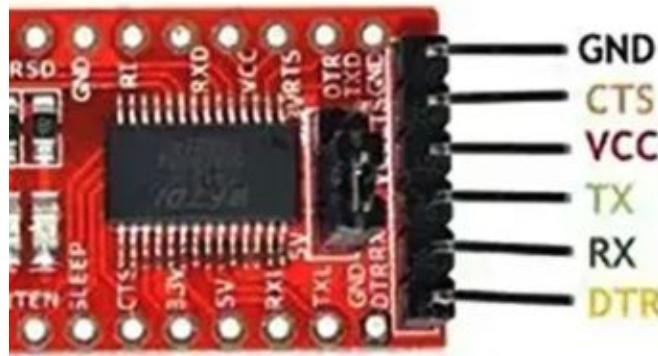


The screenshot shows the STM32CubeIDE interface with the following details:

- Project Explorer:** Displays the project structure for "SINEWAVE_v1". It includes sub-folders like Binaries, Includes, Core (with sub-folders Inc and Src), Drivers, X-CUBE-AI (with sub-folders .ai and .settings), App (with sub-files sinewave_config.h, sinewave_data_params.c, sinewave_data_params.h, sinewave_data.c, sinewave_data.h, sinewave.c, sinewave.h, and sinewave_generate_report.txt), and files like LICENSE.txt, Debug, Middlewares, USB_HOST, sinewave_model.h, sinewave_model.tflite, MX_SINEWAVE_v1.ioc, SINEWAVE_v1.Debug.launch, STM32F407VGTX_FLASH.Id, and STM32F407VGTX_RAM.Id.
- Code Editor:** The main.c file is open, showing C code for initializing peripherals (MX_GPIO_Init(), MX_USART3_UART_Init(), MX_CRC_Init(), MX_USB_OTG_FS_USB_Init()), creating an AI instance (ai_sinewave_create_and_init()), and transmitting reports via UART (HAL_UART_Transmit()). The code also handles errors and prints messages to the serial port.
- Build Analyzer:** Shows memory regions and stack details for the project.

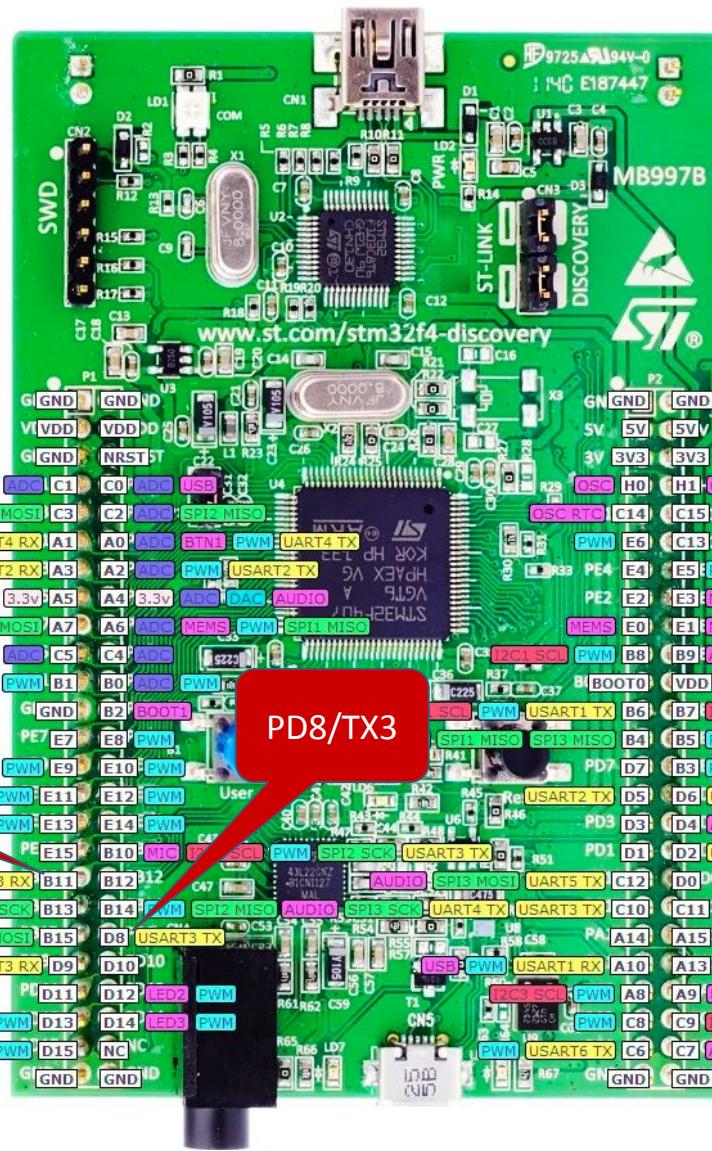
https://github.com/SkillSurf/ML4_edgeAI/tree/main/STM32_cubeAI-sinewave

Case Study 1 - tflite_SineWave



P1

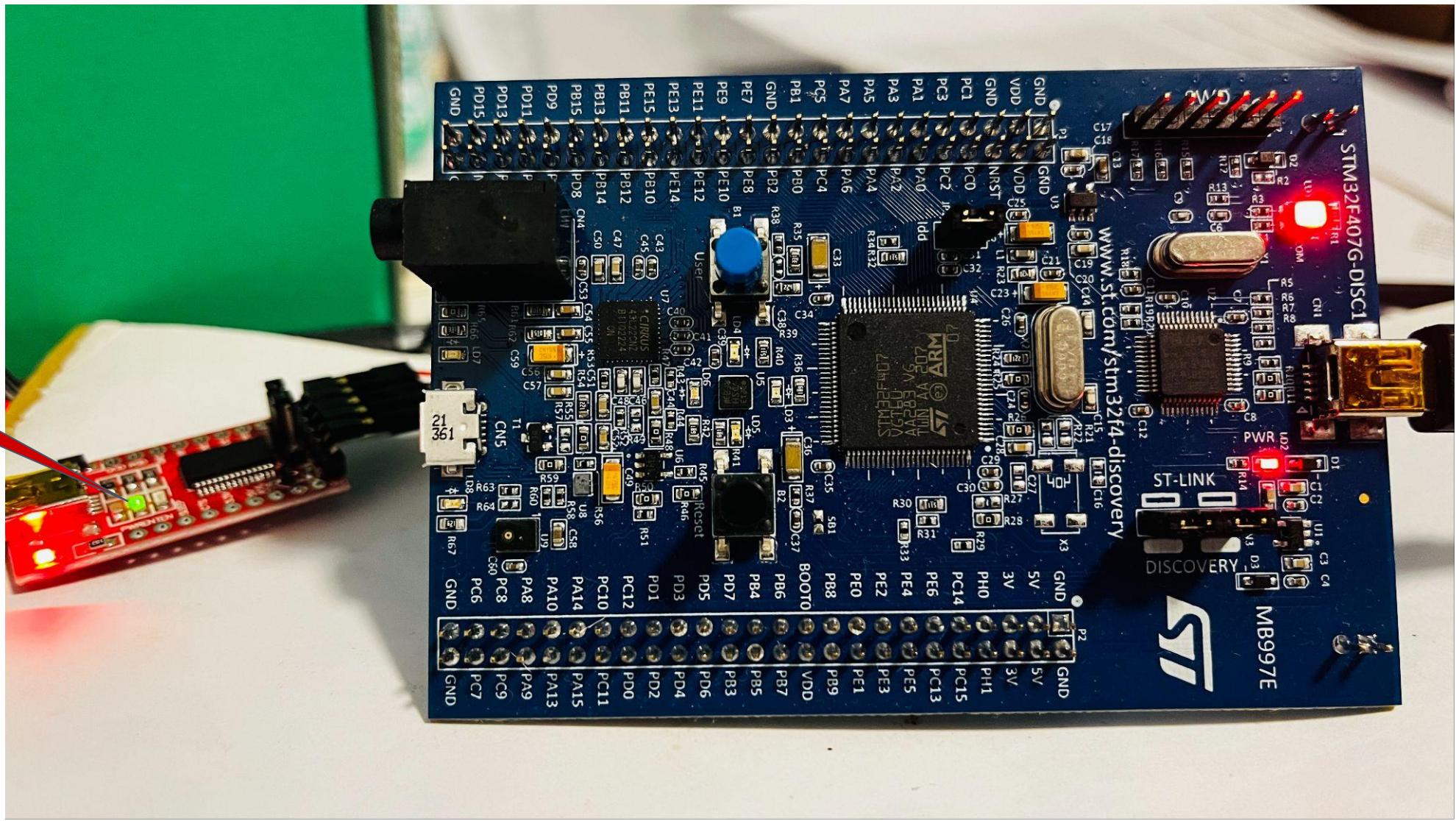
1	2
3	4
5	6
7	8
9	10
11	12
13	14
15	16
17	18
19	20
21	22
23	24
25	26
27	28
29	30
31	32
33	34
35	36
37	38
39	40
41	42
43	44
45	46
47	48
49	50



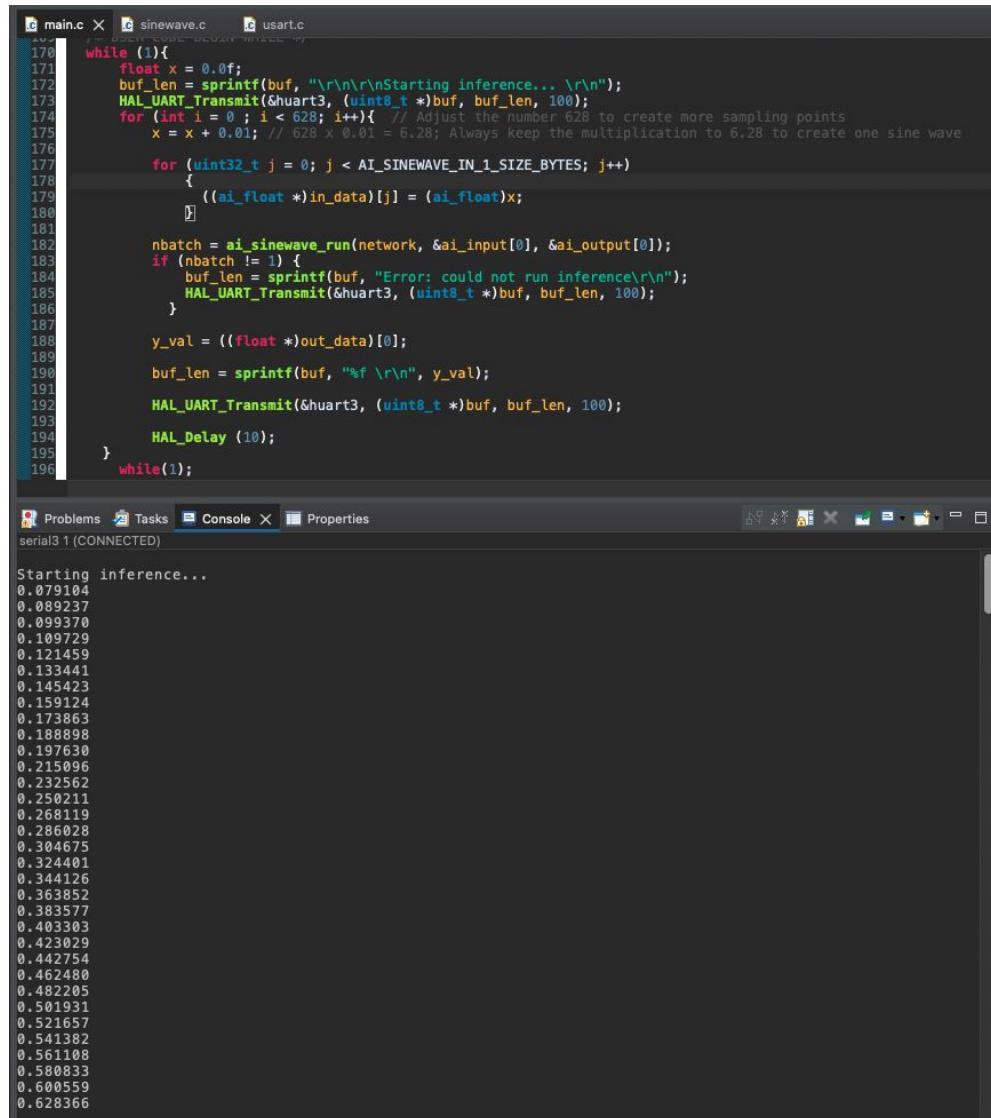
P2

1	2
3	4
5	6
7	8
9	10
11	12
13	14
15	16
17	18
19	20
21	22
23	24
25	26
27	28
29	30
31	32
33	34
35	36
37	38
39	40
41	42
43	44
45	46
47	48
49	50

Case Study 1 - tflite_SineWave



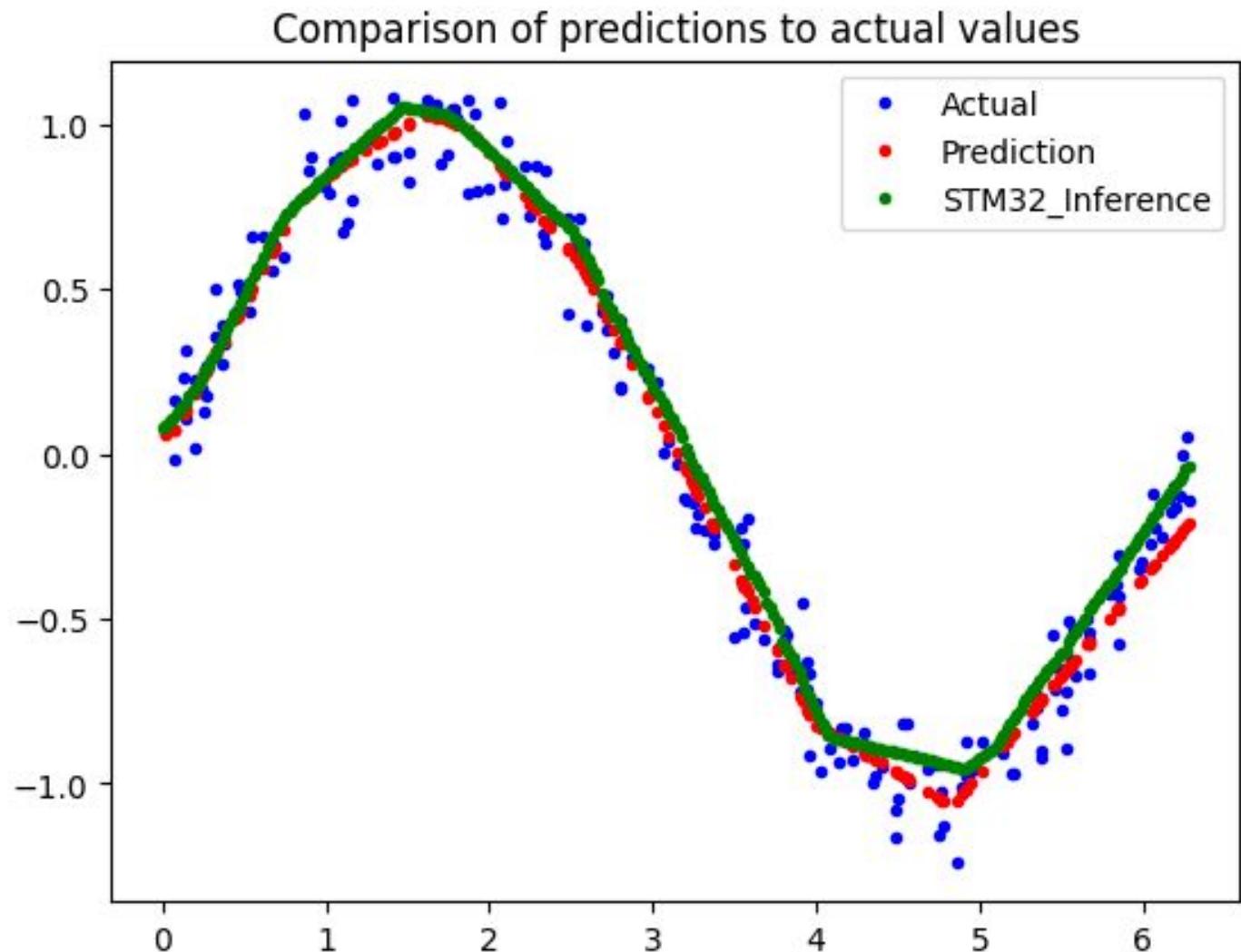
Case Study 1 - tflite_SineWave



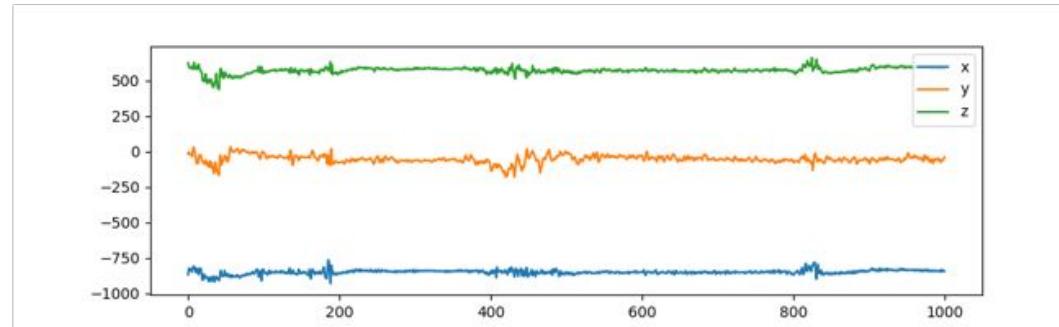
The screenshot shows a code editor with three tabs: main.c, sinewave.c, and usart.c. The main.c file contains the following code:

```
170 while(1){
171     float x = 0.0f;
172     buf_len = sprintf(buf, "\r\n\r\nStarting inference...\r\n");
173     HAL_UART_Transmit(&huart3, (uint8_t *)buf, buf_len, 100);
174     for (int i = 0 ; i < 628; i++){ // Adjust the number 628 to create more sampling points
175         x = x + 0.01; // 628 * 0.01 = 6.28; Always keep the multiplication to 6.28 to create one sine wave
176
177         for (uint32_t j = 0; j < AI_SINEWAVE_IN_1_SIZE_BYTES; j++)
178         {
179             ((ai_float *)in_data)[j] = (ai_float)x;
180         }
181
182         nbatch = ai_sinewave_run(network, &ai_input[0], &ai_output[0]);
183         if (nbatch != 1) {
184             buf_len = sprintf(buf, "Error: could not run inference\r\n");
185             HAL_UART_Transmit(&huart3, (uint8_t *)buf, buf_len, 100);
186         }
187
188         y_val = ((float *)out_data)[0];
189
190         buf_len = sprintf(buf, "%f \r\n", y_val);
191
192         HAL_UART_Transmit(&huart3, (uint8_t *)buf, buf_len, 100);
193
194         HAL_Delay (10);
195     }
196 }
```

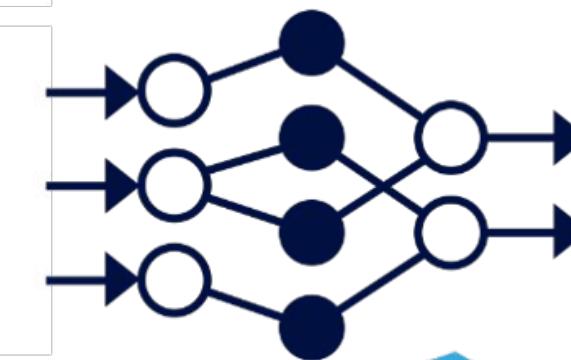
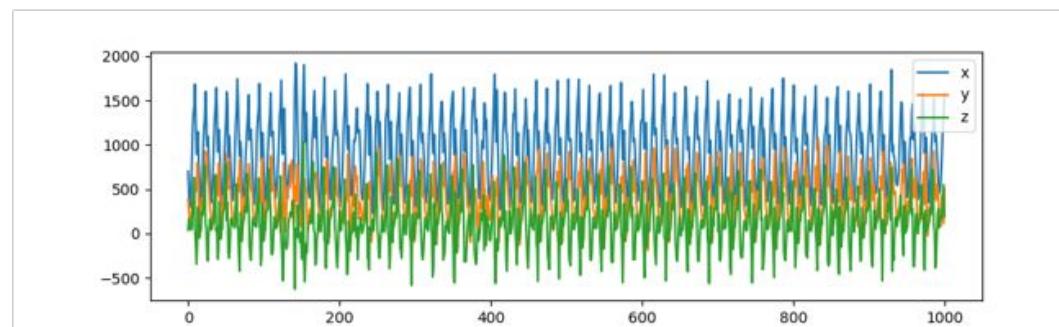
The console window below shows the output of the program, starting with "Starting inference..." followed by a series of floating-point numbers representing the sine wave values.



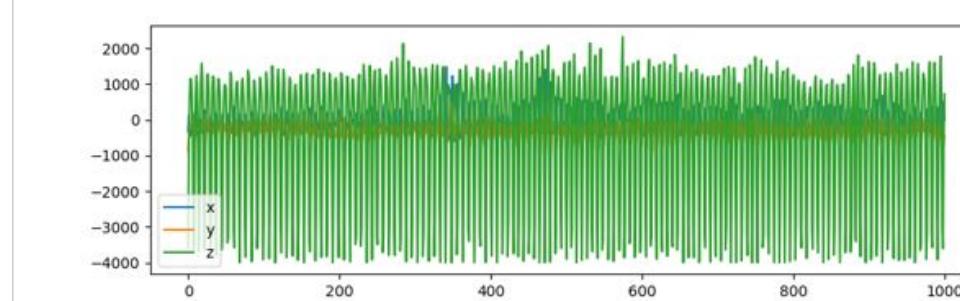
Case Study 2 - Activity Recognition



[GitHub - teomaras76/AI_on_STM32_Gesture_recognition: Gesture recognition Deep Learning project for STM32 \(based on Keras model\)](#)

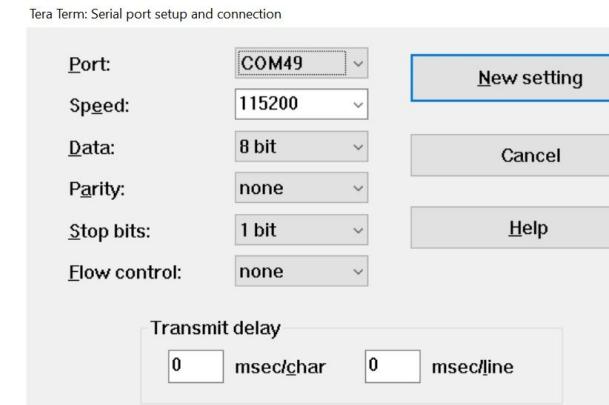
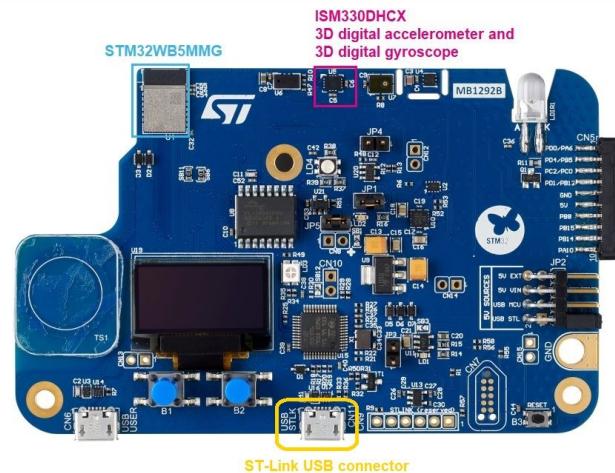
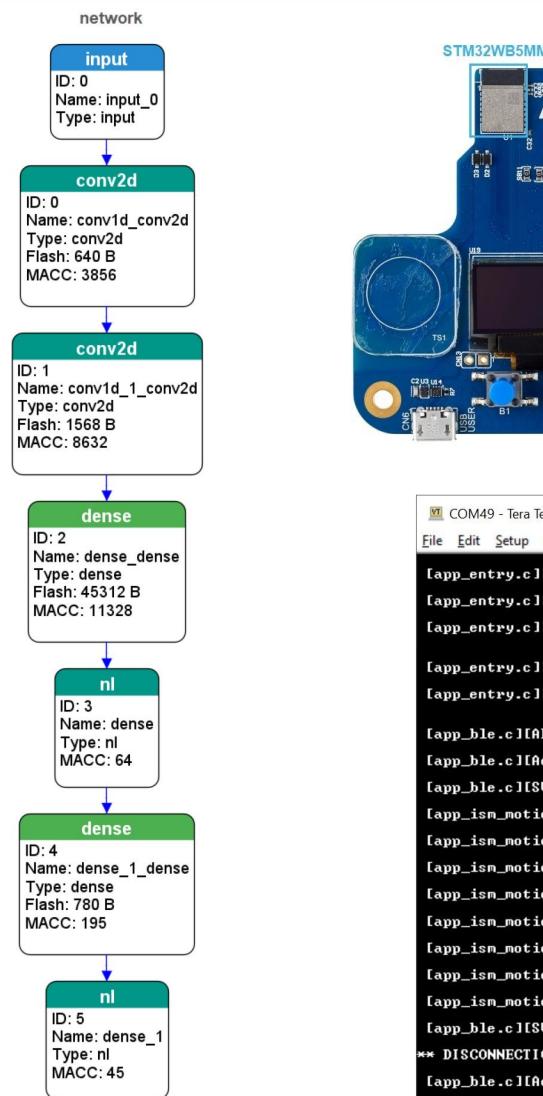


Stationary
Walking
Running



[GitHub - stm32-hotspot/STM32WB-BLE-AI-MotionSense: Example demonstrates how to create a motion sensing application to recognize human activities using machine learning on an STM32WB microcontroller](#)

Case Study 2 - Activity Recognition



COM49 - Tera Term VT

```

File Edit Setup Control Window Help

[app_entry.c][APPE_SysUserEvtRx][368] Wireless Firmware version 1.13.3
[app_entry.c][APPE_SysUserEvtRx][369] Wireless Firmware build 2
[app_entry.c][APPE_SysUserEvtRx][370] PUS version 1.2.0

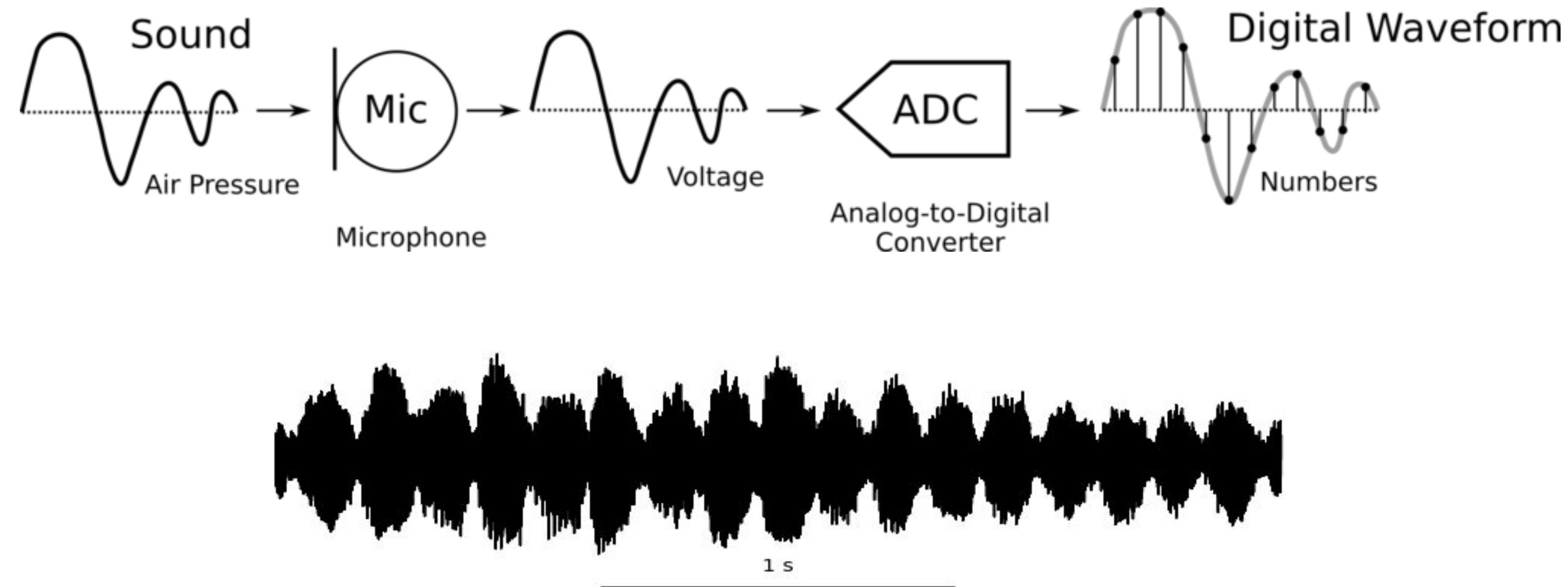
[app_entry.c][APPE_SysEvtReadyProcessing][458] SHCI_SUB_EUT_CODE_READY - WIRELESS_FW_RUNNING
[app_entry.c][APPE_SysEvtReadyProcessing][482] DBGMCU_GetRevisionID= 2001

[app_ble.c][APP_BLE_Init][348] ISM330DHX init complete
[app_ble.c][Adv_Request][825] Successfully Start Fast Advertising
[app_ble.c][SUCCCTL_App_Notification][452] HCI_LE_CONNECTION_COMPLETE_SUBLT_CODE for connection handle 0x801
[app_ism_motion.c][motion_run_ai_task][247] Motion: 1 <BLE data = 2> - walking / stationary = 0.000 walking = 0.996 running = 0.003
[app_ism_motion.c][motion_run_ai_task][247] Motion: 0 <BLE data = 1> - stationary / stationary = 0.996 walking = 0.004 running = 0.000
[app_ism_motion.c][motion_run_ai_task][247] Motion: 1 <BLE data = 2> - walking / stationary = 0.001 walking = 0.994 running = 0.005
[app_ism_motion.c][motion_run_ai_task][247] Motion: 1 <BLE data = 2> - walking / stationary = 0.000 walking = 0.781 running = 0.219
[app_ism_motion.c][motion_run_ai_task][247] Motion: 2 <BLE data = 4> - running / stationary = 0.000 walking = 0.005 running = 0.995
[app_ism_motion.c][motion_run_ai_task][247] Motion: 2 <BLE data = 4> - running / stationary = 0.000 walking = 0.000 running = 1.000
[app_ism_motion.c][motion_run_ai_task][247] Motion: 0 <BLE data = 1> - stationary / stationary = 0.795 walking = 0.201 running = 0.004
[app_ism_motion.c][motion_run_ai_task][247] Motion: 0 <BLE data = 1> - stationary / stationary = 0.996 walking = 0.003 running = 0.000
[app_ble.c][SUCCCTL_App_Notification][407]
** DISCONNECTION EVENT WITH CLIENT
[app_ble.c][Adv_Request][825] Successfully Start Fast Advertising
  
```

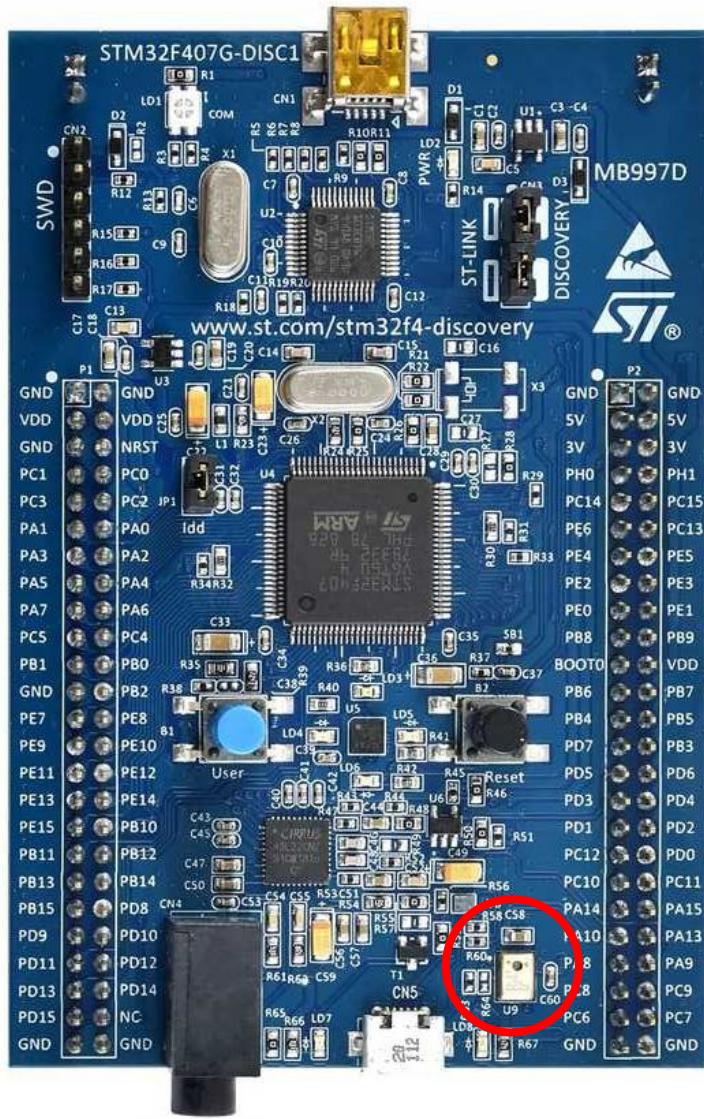
Case Study 2 - Activity Recognition



Case Study 3 - Audio classifier



Case Study 3 - Audio classifier



Microphone

The board is equipped with a **MP45DT02** MEMS microphone, which produces a stream of 1-bit **Digital Pulse Modulation** (PDM) samples. The PDM samples are received using DMA and then converted to 16-bit signed **Pulse Code Modulation** (PCM) samples through the usage of a software low pass filter.

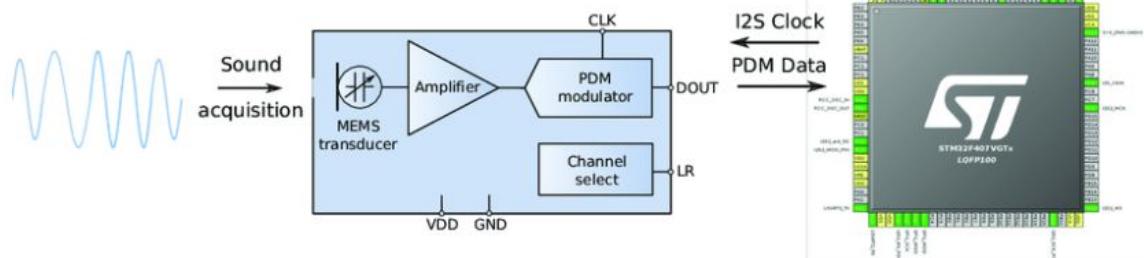
The PCM data is sampled at a frequency of 32 kHz, which is obtained by setting the *PLL12SN*, *PLL12SR* and *I2SDIV* registers to appropriate values (see chapters 7.3.23 and 28.4.4 of the datasheet for further details):

$$f_{(\text{VCO clock})} = f_{(\text{PLL12S clock input})} * \frac{\text{PLL12SN}}{\text{PLLM}} = 8 \text{ MHz} \frac{213}{8} = 213 \text{ MHz}$$

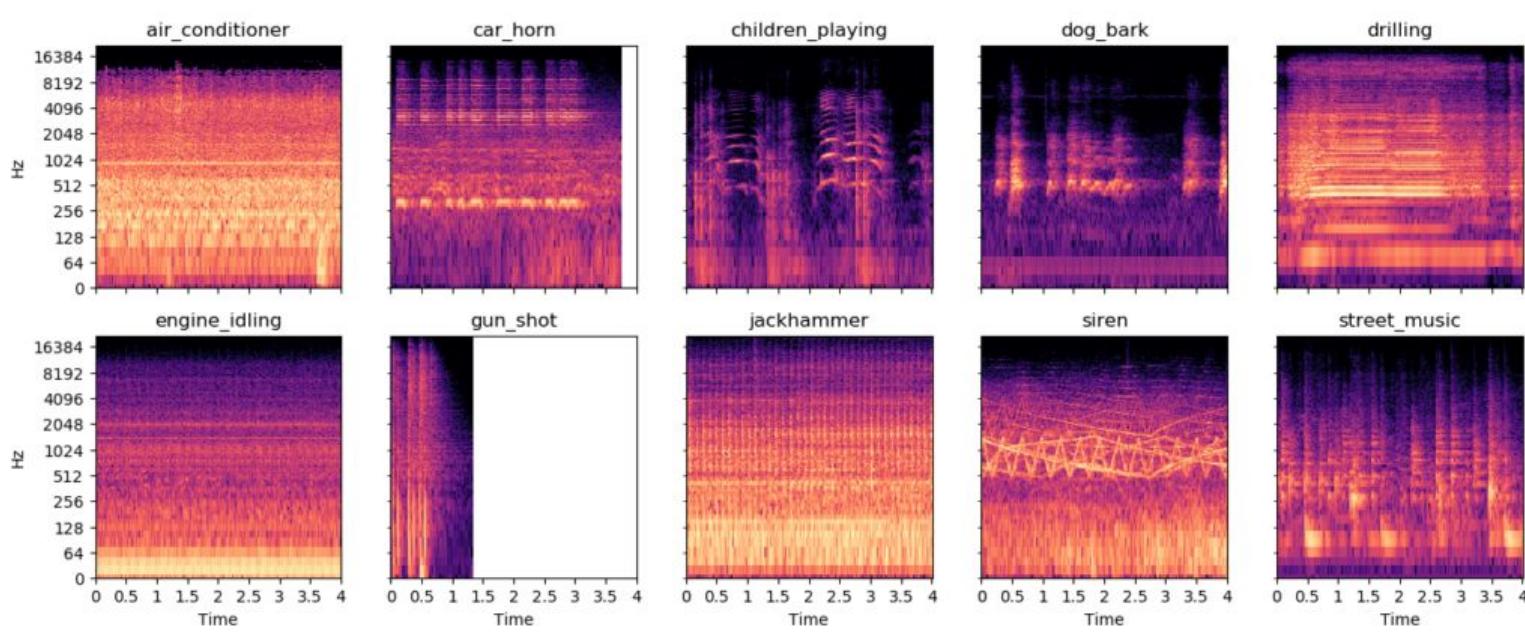
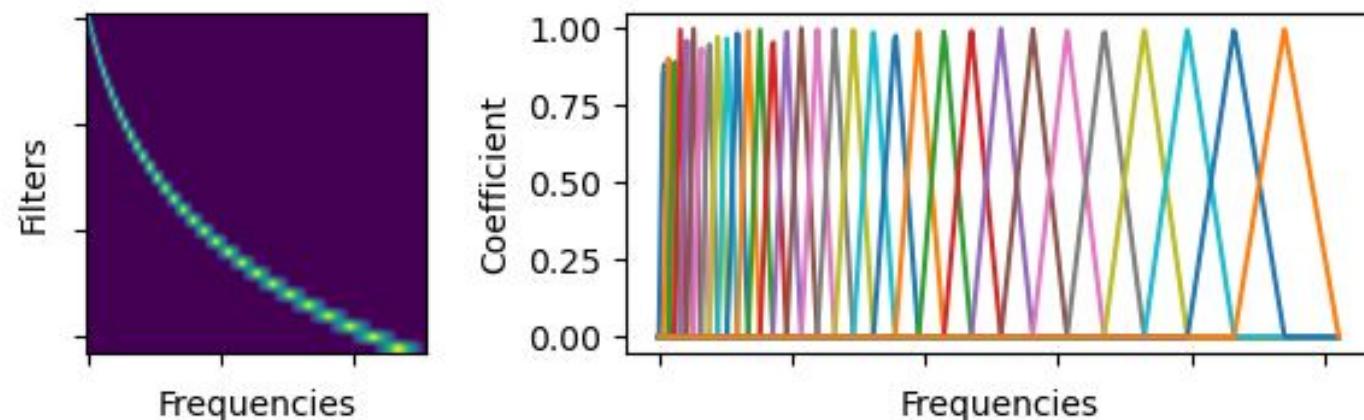
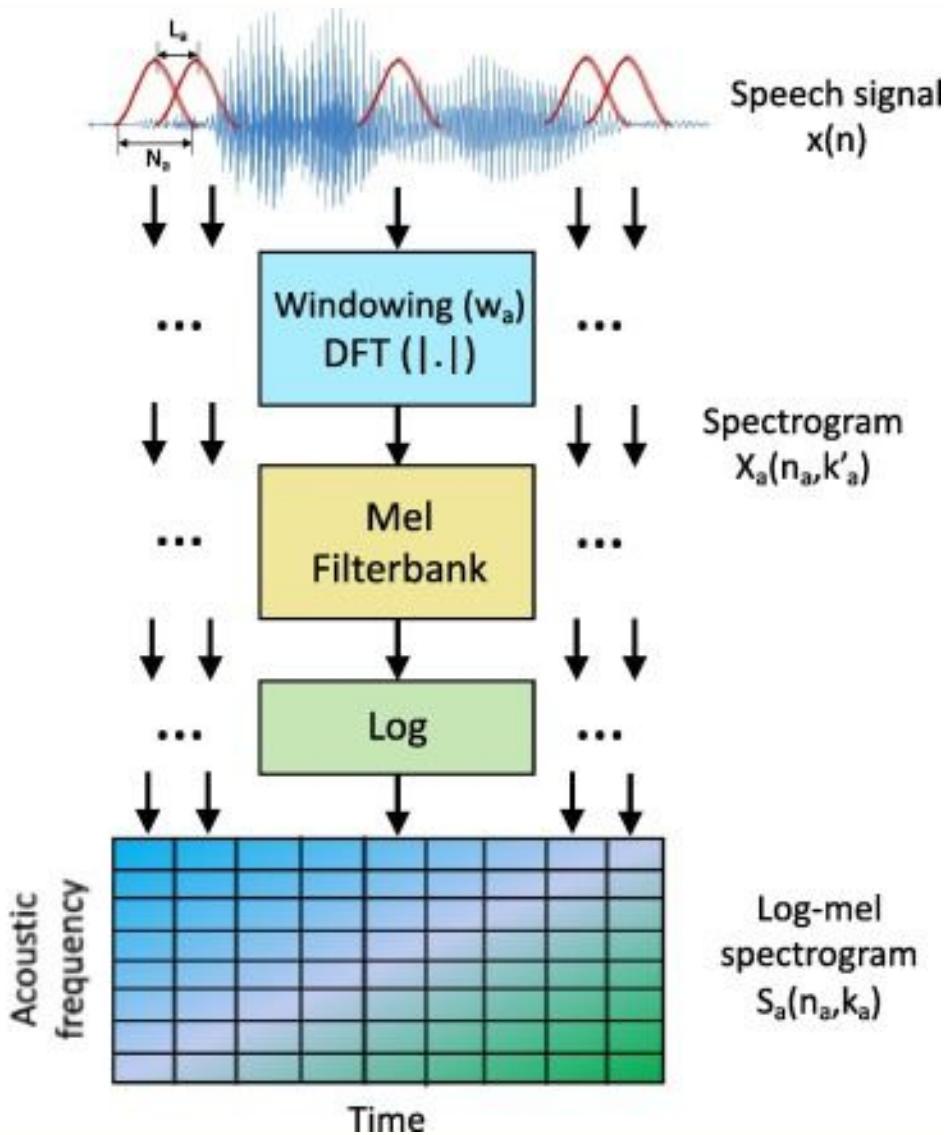
$$f_{(\text{PLL I2S clock output})} = \frac{f_{(\text{VCO clock})}}{\text{PLL12SR}} = \frac{213 \text{ MHz}}{2} = 106,5 \text{ MHz}$$

$$f_{(\text{PDM})} = \frac{f_{(\text{PLL I2S clock output})}}{2 * \text{I2SDIV} * 8} = \frac{106,5 \text{ MHz}}{2 * 13 * 8} = 512 \text{ kHz}$$

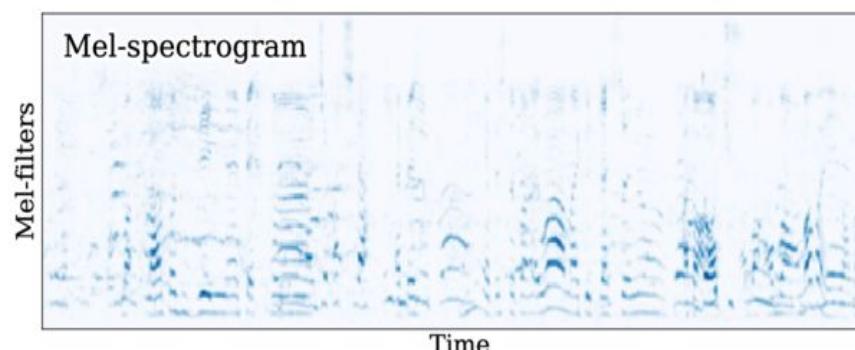
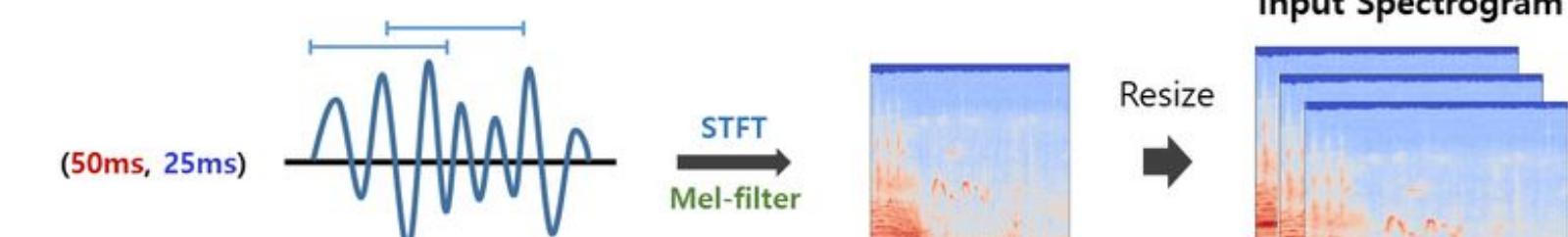
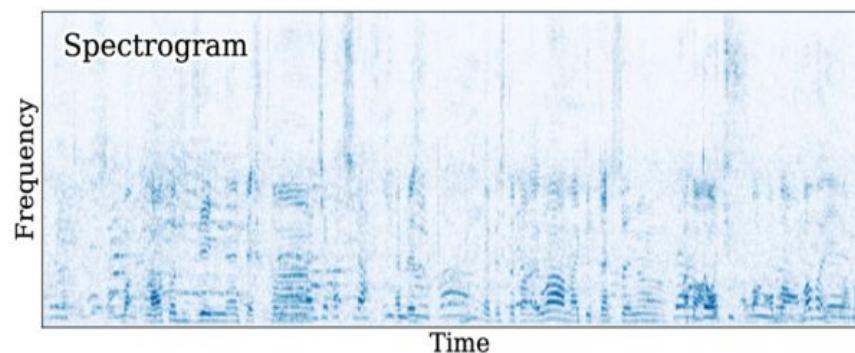
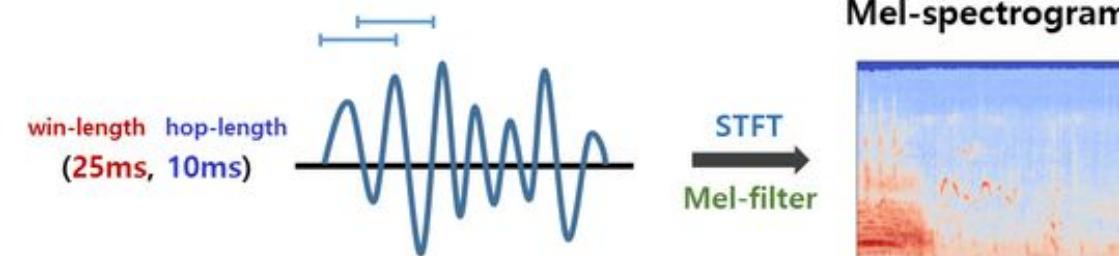
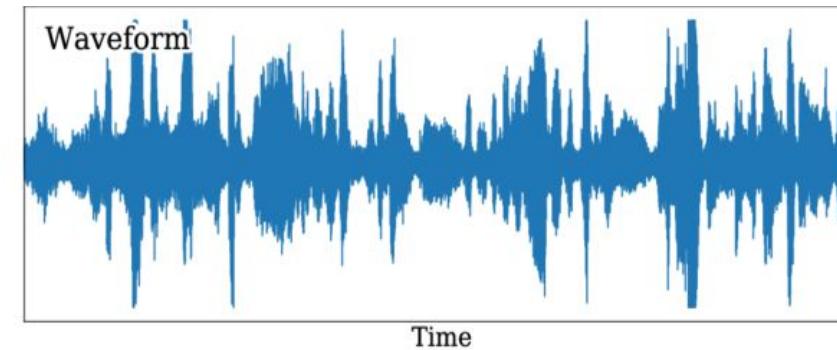
$$f_{(\text{PCM})} = \frac{f_{(\text{PDM})}}{16} = \frac{512 \text{ kHz}}{16} = 32 \text{ kHz}$$



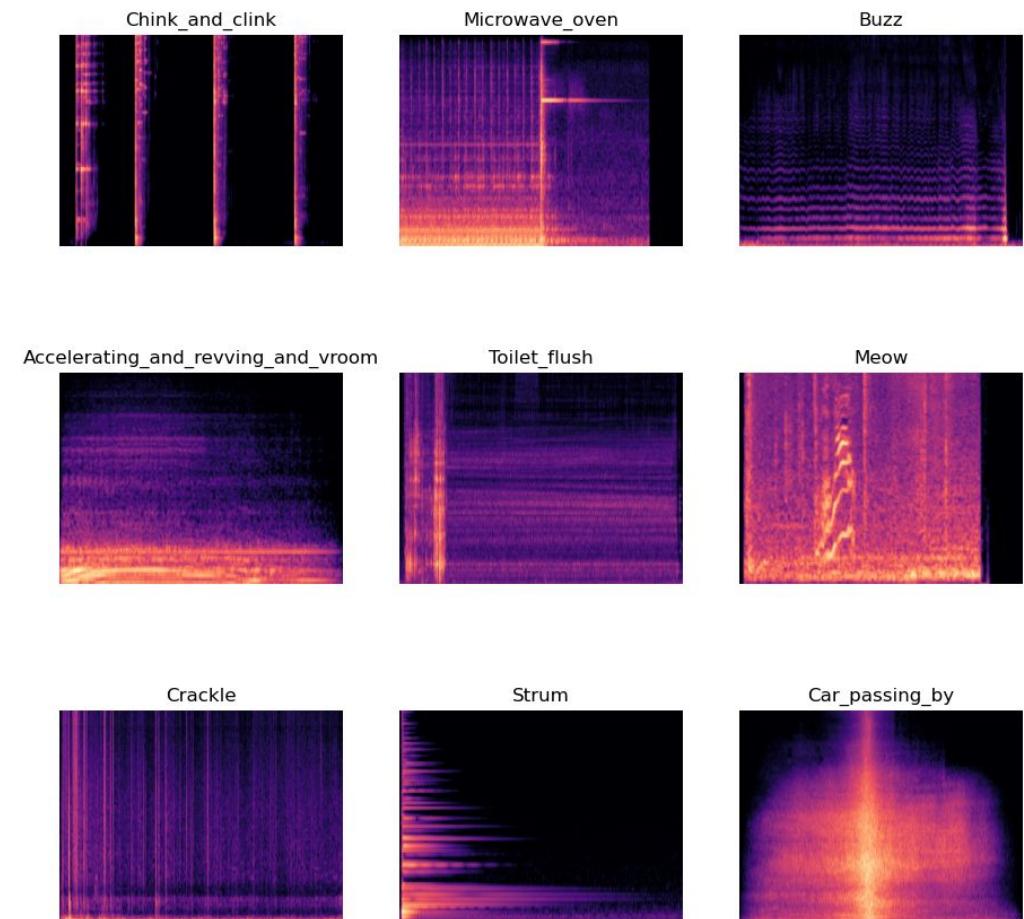
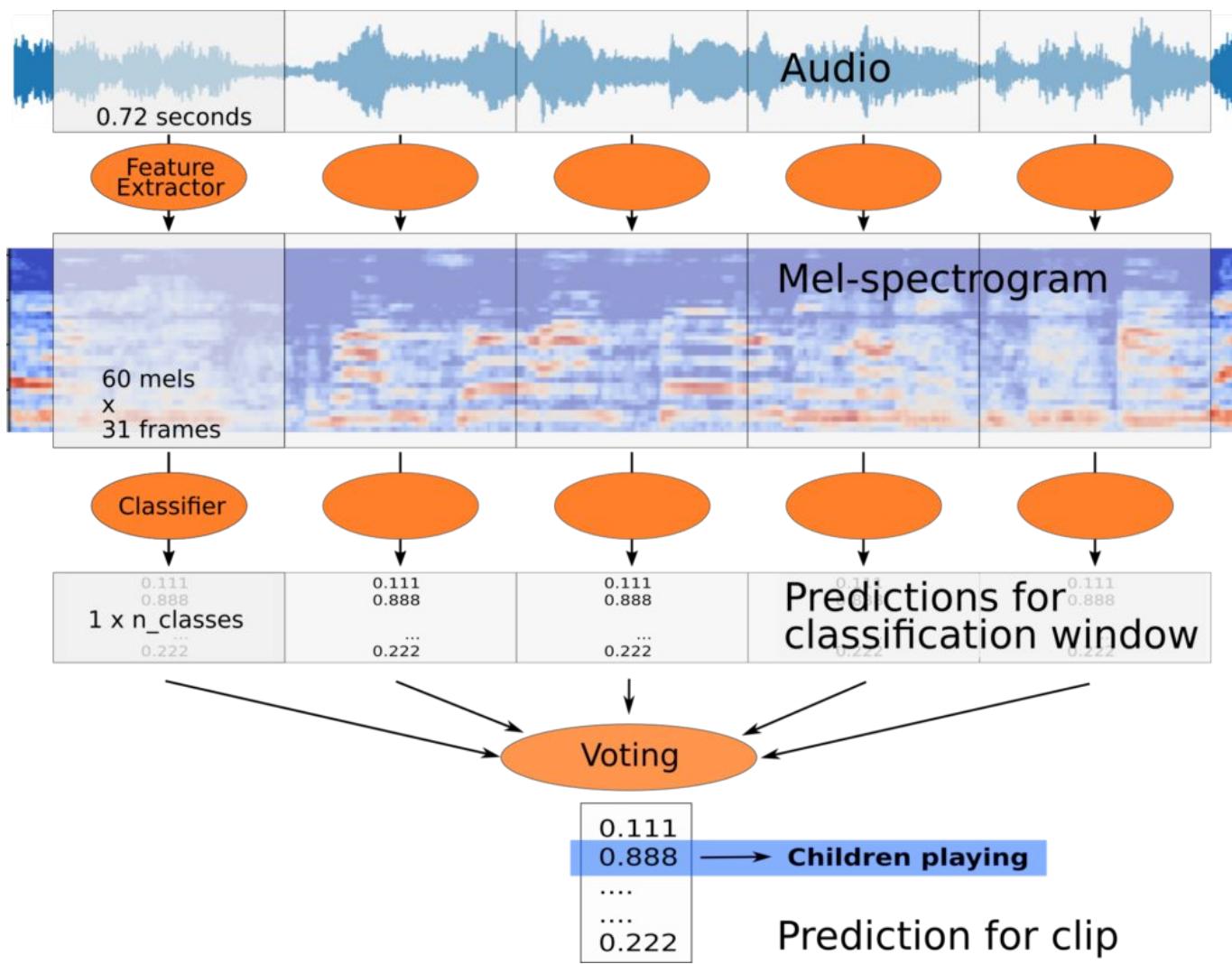
Case Study 3 - Audio classifier



Case Study 3 - Audio classifier

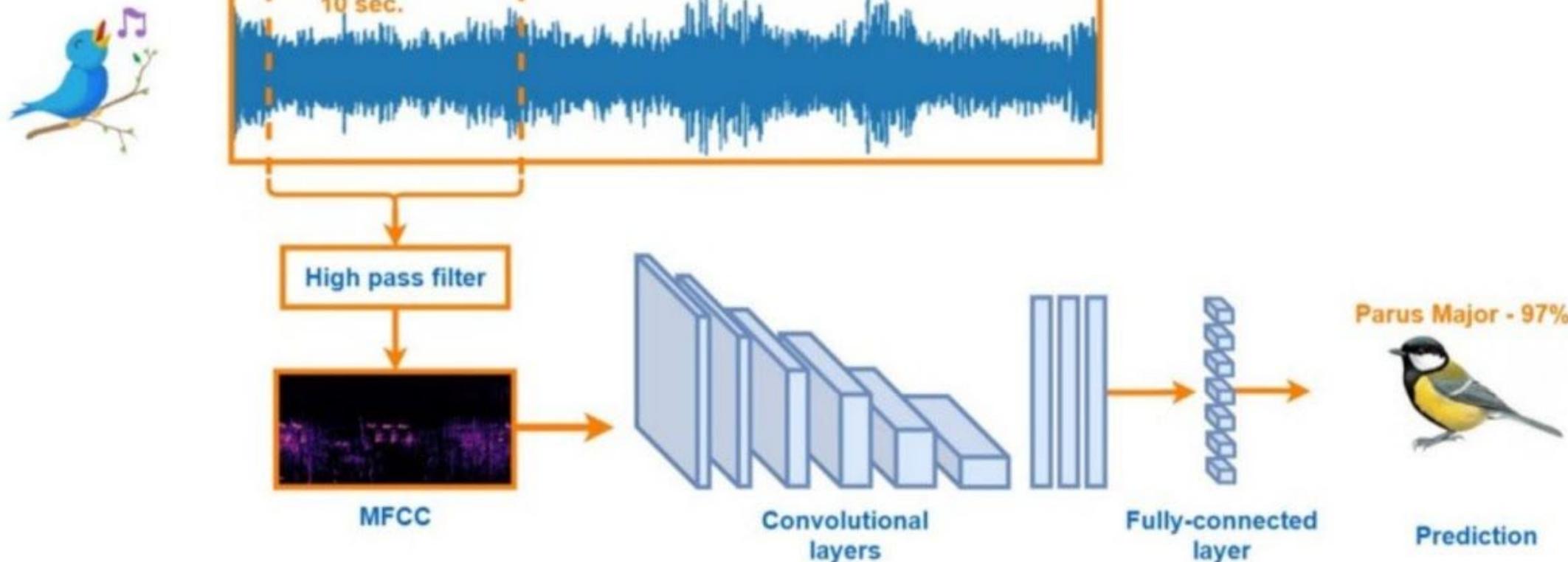


Case Study 3 - Audio classifier



<https://www.jonnor.com/tag/audio-classification/>

Case Study 3 - Audio classifier



<https://www.jonnor.com/tag/audio-classification/>

Case Study 3 - Audio classifier

Case Study 3 - Audio classifier

STM32CubeMX Untitled*: STM32F407VGTx STM32F407G-DISC1

Please wait...

Analyzing Network
100%

Analyzing model
arch -arm64 /Users/kithminrw/STM32Cube/Repository/Packs/STMicroelectronics/X-CUBE-AI/9.1.0/Utilities/macarm/stedgeai analyze sound_classifier -m /Users/kithminrw/STM32Cube/embedded-sound-classifier-master/neural-network/model.h5 --compression inputs --allocate-outputs --workspace /var/folders/8g/_xjnmg_17fz5fm2s4yf8p7dc000gn/T/mxAI_workspace1264159166547875183875
output /Users/kithminrw/.stm32cubemx/sound_classifier_output
ST Edge AI Core v1.0.0-19899

Creating c (debug) info json file /var/folders/8g/_xjnmg_17fz5fm2s4yf8p7dc000gn/T/mxAI_workspace12641591665478751838750972.json

Exec/report summary (analyze)

```

model file      : /Users/kithminrw/STM32Cube/embedded-sound-classifier-master/neural-network/model.h5
type           : keras
c_name          : sound_classifier
compression     : none
options         : allocate-inputs, allocate-outputs
optimization    : balanced
target/series   : stm32f4
workspace dir  : /var/folders/8g/_xjnmg_17fz5fm2s4yf8p7dc000gn/T/mxAI_workspace126415916654787518387509727113100480
output dir     : /Users/kithminrw/.stm32cubemx/sound_classifier_output
model_fmt       : float
model_name      : model
model_hash      : 0x9ec28770e730f3b5a7b2bfa42b57bcf5
params #       : 5,163 items (20.17 KiB)

input 1/1       : 'input_0', f32(1x512), 2.00 KBytes, activations
output 1/1      : 'dense_1', f32(1x3), 12 Bytes, activations
macc           : 5,218
weights (ro)    : 20,652 B (20.17 KiB) (1 segment)
activations (rw) : 2,088 B (2.04 KiB) (1 segment) *
ram (total)     : 2,088 B (2.04 KiB) = 2,088 + 0 + 0

(*) 'input'/'output' buffers can be used from the activations buffer

Computing AI RT data/code size (target=stm32f4)...

Model name - model

```

m_id	layer (original)	oshape	param/size	macc	connected to
0	input_0 (None) dense_dense (Dense) dense (Dense)	[b:1,c:512] [b:1,c:10] [b:1,c:10]	5,130/20,520	5,130 10	input_0 dense_dense

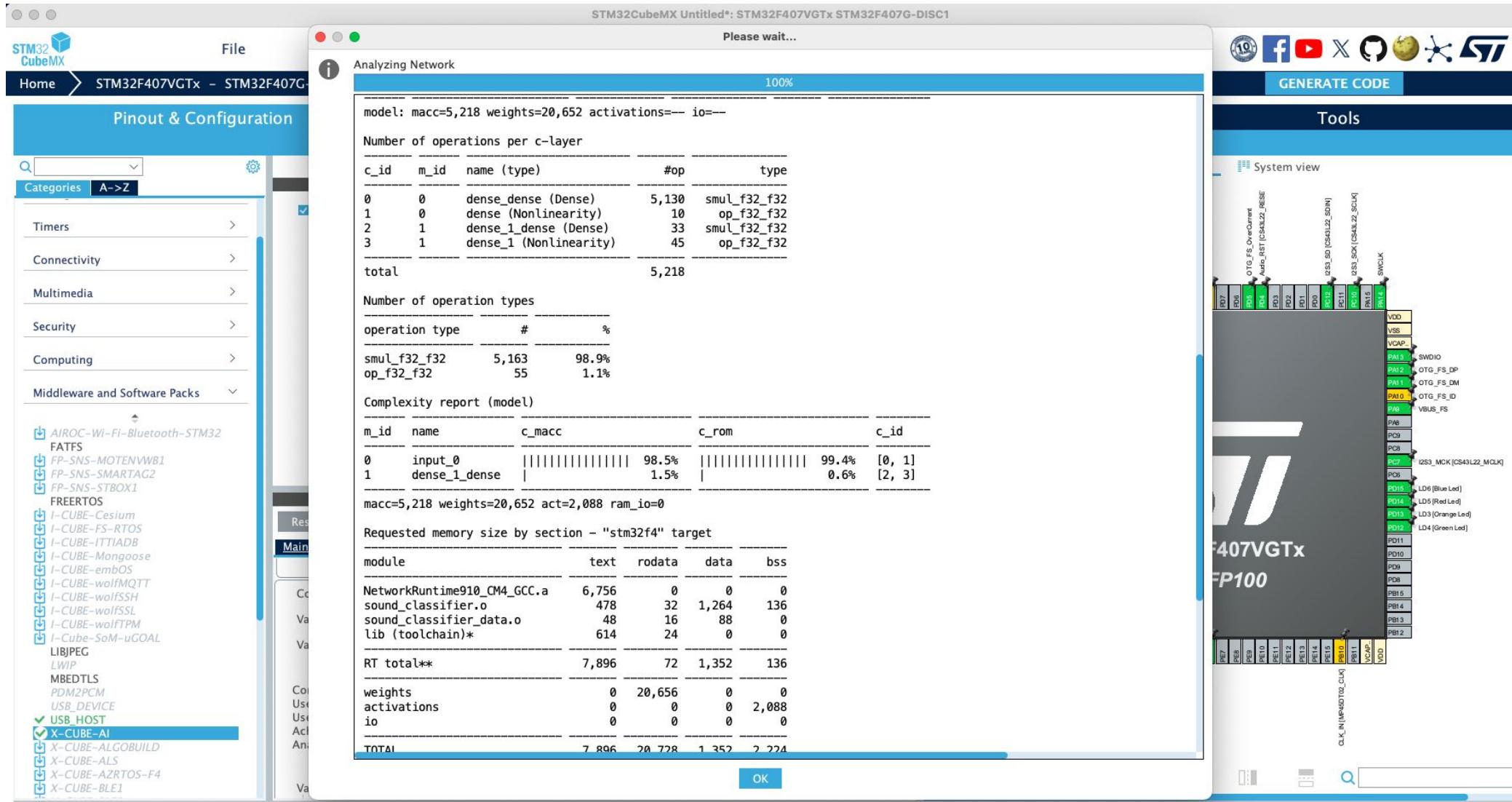
OK

System view

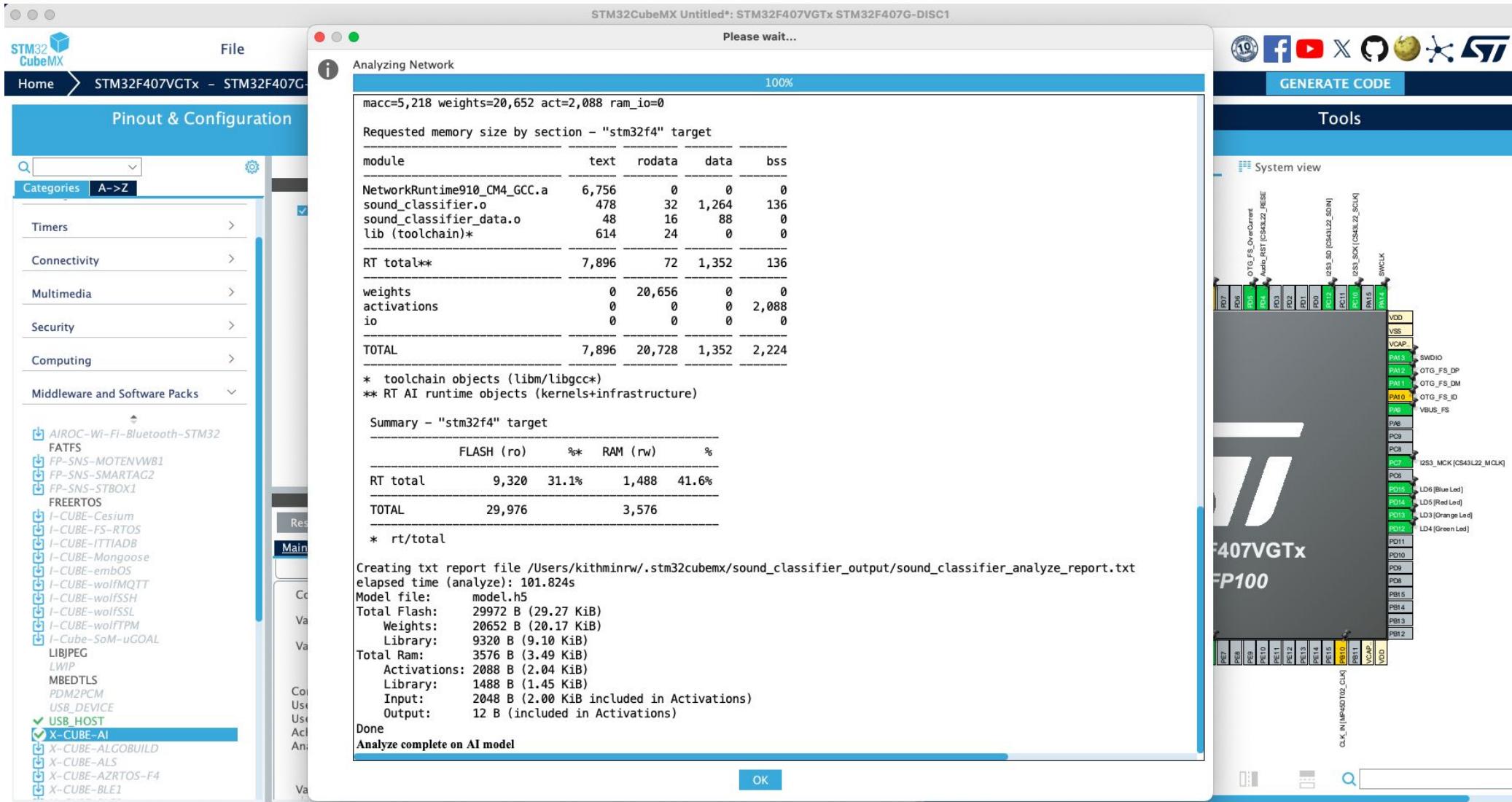
The pinout diagram shows the physical layout of the STM32F407VGTx chip, with pins labeled by name and color-coded for function. Key pins include:

- PD7: OTG_FS_OverCurrent
- PD6: OTG_FS_RST [CS43L22_RESET]
- PD5: VSS
- PD4: VCAP
- PD3: SWDIO
- PD2: OTG_FS_DM
- PD1: OTG_FS_ID
- PD0: VBUS_FS
- PC8: PA8
- PC7: I2S3_MCK [CS43L22_MCLK]
- PC6: PA6
- PC5: PD16: LD6 [Blue Led]
- PC4: PD15: LD5 [Red Led]
- PC3: PD14: LD3 [Orange Led]
- PC2: PD13: LD4 [Green Led]
- PC1: PD11: PA10
- PC0: PD9: PA9
- PA8: PB8
- PA7: PB7: CLK_IN [MP460T12_CLK]
- PA6: PB6: PE6
- PA5: PB5: PE5
- PA4: PB4: PE4
- PA3: PB3: PE3
- PA2: PB2: PE2
- PA1: PB1: PE1
- PA0: PB0: PE0
- PA15: PB15: PE15
- PA14: PB14: PE14
- PA13: PB13: PE13
- PA12: PB12: PE12
- PA11: PB11: PE11
- PA10: PB10: PE10
- PA9: PB9: PE9
- PA8: PB8: PE8
- PA7: PB7: PE7
- PA6: PB6: PE6
- PA5: PB5: PE5
- PA4: PB4: PE4
- PA3: PB3: PE3
- PA2: PB2: PE2
- PA1: PB1: PE1
- PA0: PB0: PE0

Case Study 3 - Audio classifier

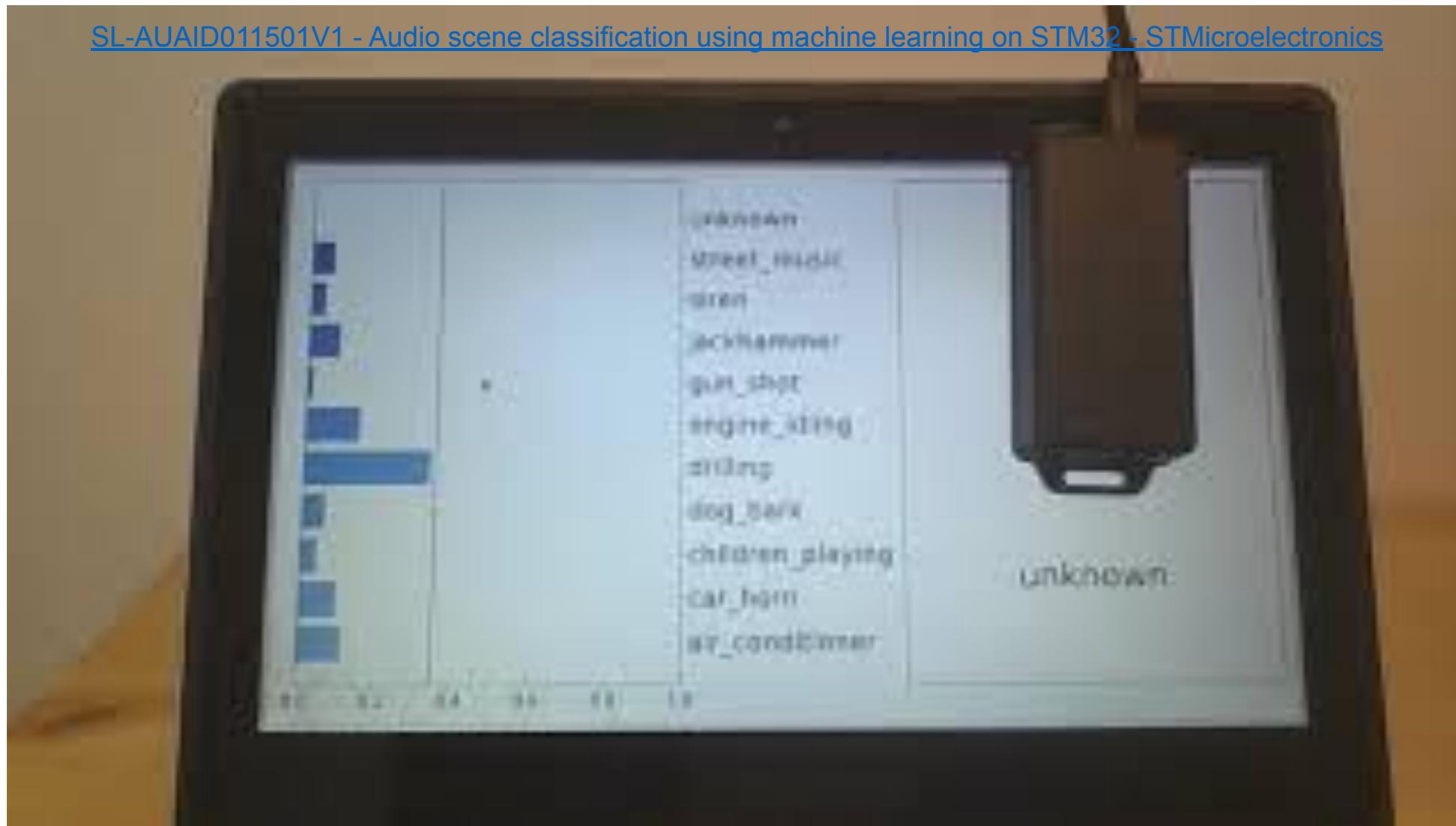


Case Study 3 - Audio classifier



Case Study 3 - Audio classifier

[SL-AUAID011501V1 - Audio scene classification using machine learning on STM32 - STMicroelectronics](#)



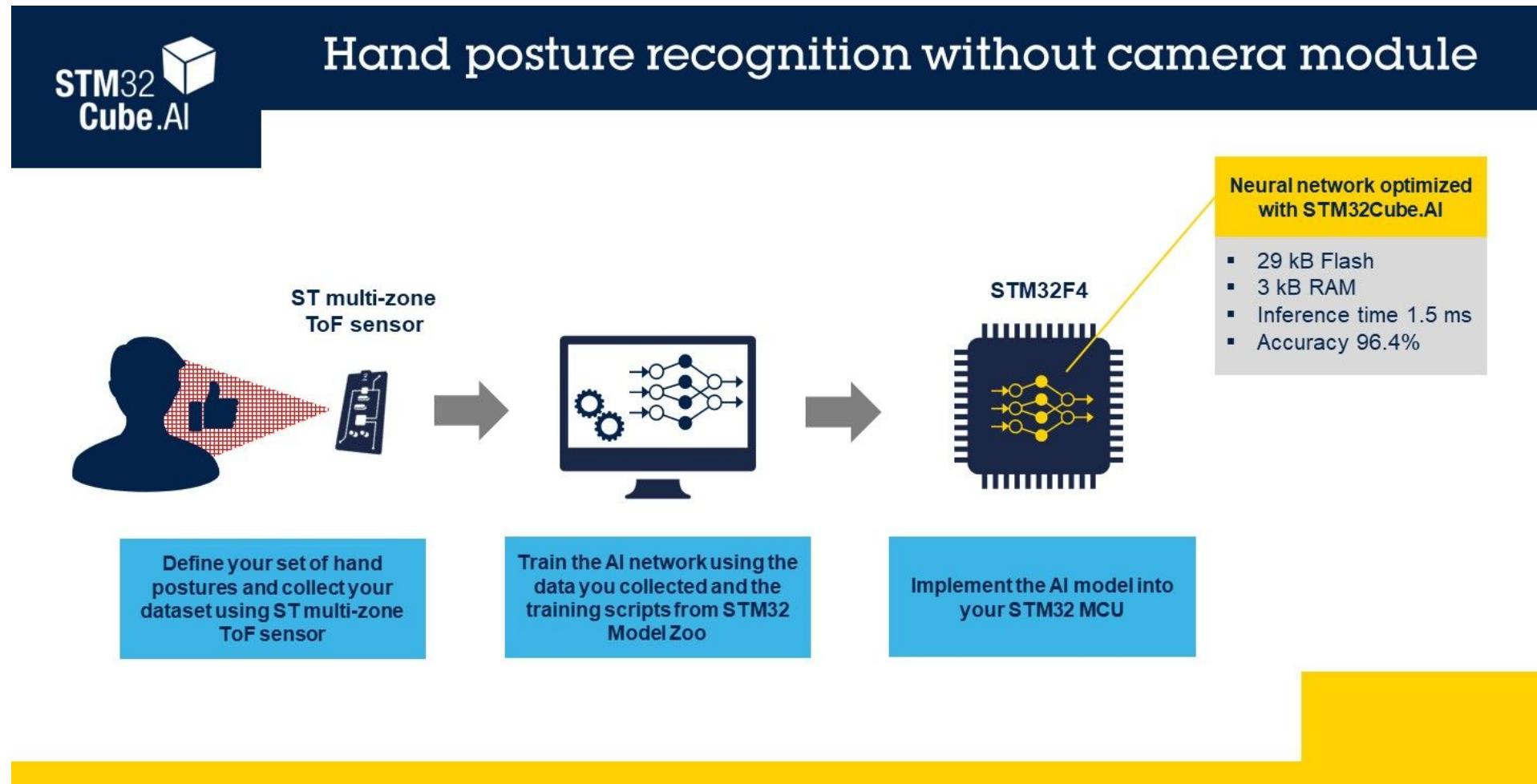
Case Study 4 - Person detection

Person presence
detection
Neural Network
on low power
STM32 microcontrollers



More Case Studies

[Edge AI case studies - STMicroelectronics](#)



[MOOC - STM32Cube.AI workshop - YouTube](#) and [Workshop AI](#)

Assignment 4

WOKWI SAVE SHARE Docs User Profile

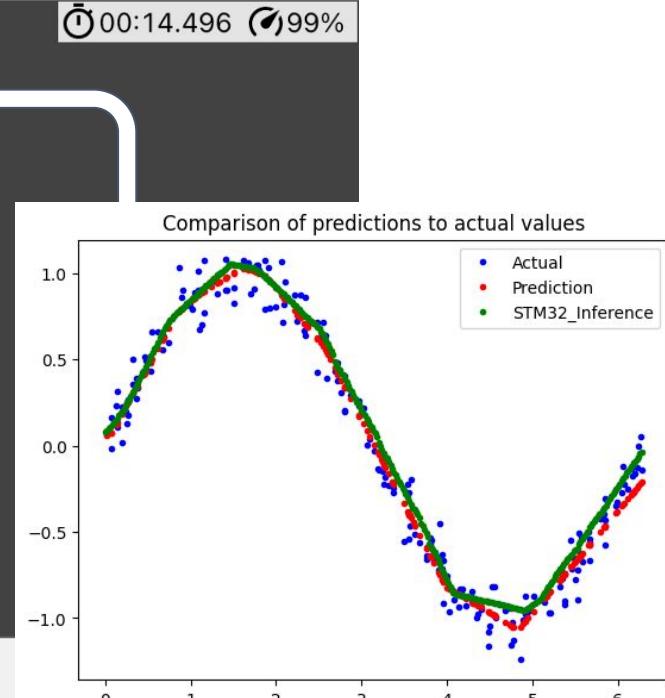
main.c diagram.json Library Manager

```
1 // STM32 Nucleo-L031K6 HAL Blink + printf() example
2 // Simulation: https://wokwi.com/projects/367244067477216257
3
4 #include <stdio.h>
5 #include <stdint.h>
6 #include <string.h>
7 #include <stm32l0xx_hal.h>
8
9 // ST Nucleo Green user LED (PB3)
10 #define LED_PORT          GPIOB
11 #define LED_PIN           GPIO_PIN_3
12 #define LED_PORT_CLK_ENABLE __HAL_RCC_GPIOB_CLK_ENABLE
13 #define VCP_TX_Pin GPIO_PIN_2
14 #define VCP_RX_Pin GPIO_PIN_15
15
16 UART_HandleTypeDef huart2;
17
18 void SystemClock_Config(void);
19 static void MX_USART2_UART_Init(void);
20
21 void osSystickHandler(void)
22 {
23     // 1 Hz blinking:
24     if ((HAL_GetTick() % 500) == 0)
25     {
26         HAL_GPIO_TogglePin(LED_PORT, LED_PIN);
27     }
28 }
29
30 void initGPIO()
31 {
32     GPIO_InitTypeDef GPIO_Config;
```

Simulation 00:14.496 99%

ST Nucleo L031K6 Board Diagram showing a green LED connected to PB3 via a resistor. The board also has pins for VCP TX and RX.

Comparison of predictions to actual values



The graph plots three series against time (X-axis, 0 to 6) against value (Y-axis, -1.0 to 1.0). The Actual data (blue dots) shows a noisy sinusoidal wave. The Prediction (red dots) and STM32_Inference (green line) show smooth sinusoidal fits to the data.

Output Terminal:

```
Hello, World!
Hello Students! Tracing X = 11
Hello, World!
Hello Students! Tracing X = 12
Hello, World!
Hello Students! Tracing X = 13
```

Thank You!