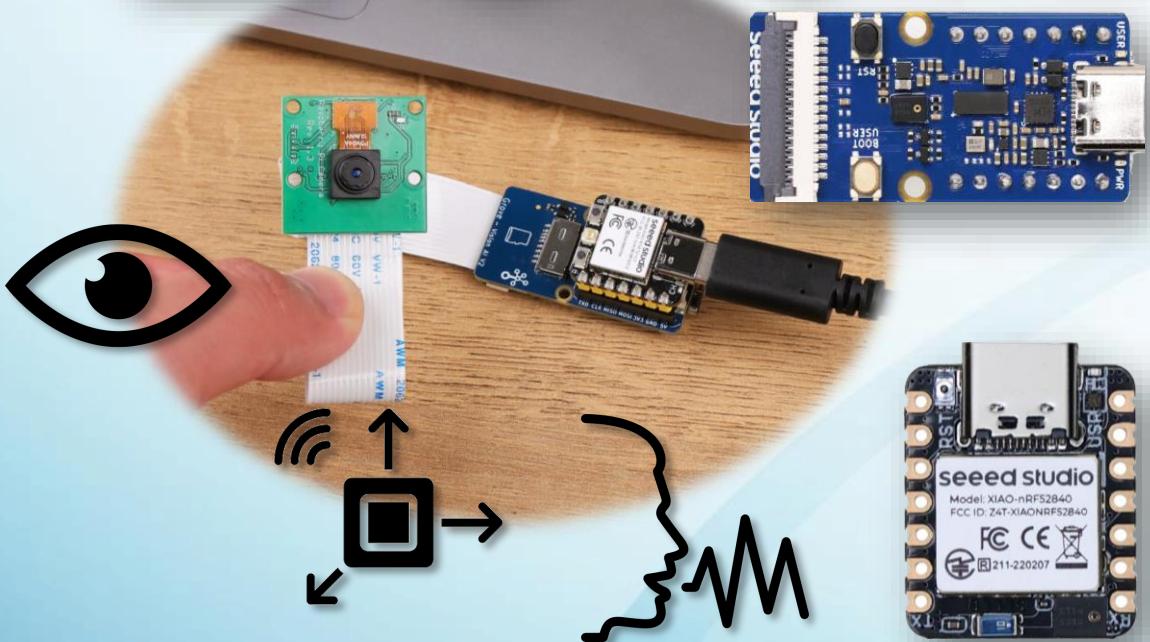


OmniXRI TinyML 小學堂 2025



Himax



歡迎加入
邊緣人俱樂部



沒有最邊



只有更邊

Cortex-M
Processor

Ethos-U
MicroNPU



【第 5 講】

通用微控制器軟體介面標準 (CMSIS)



歐尼克斯實境互動工作室 (OmniXRI Studio)
許哲豪 (Jack Hsu)

簡報大綱



- 5.1. CMSIS 基本介紹
- 5.2. CMSIS-Core & Driver
- 5.3. CMSIS-DSP
- 5.4. CMSIS-NN

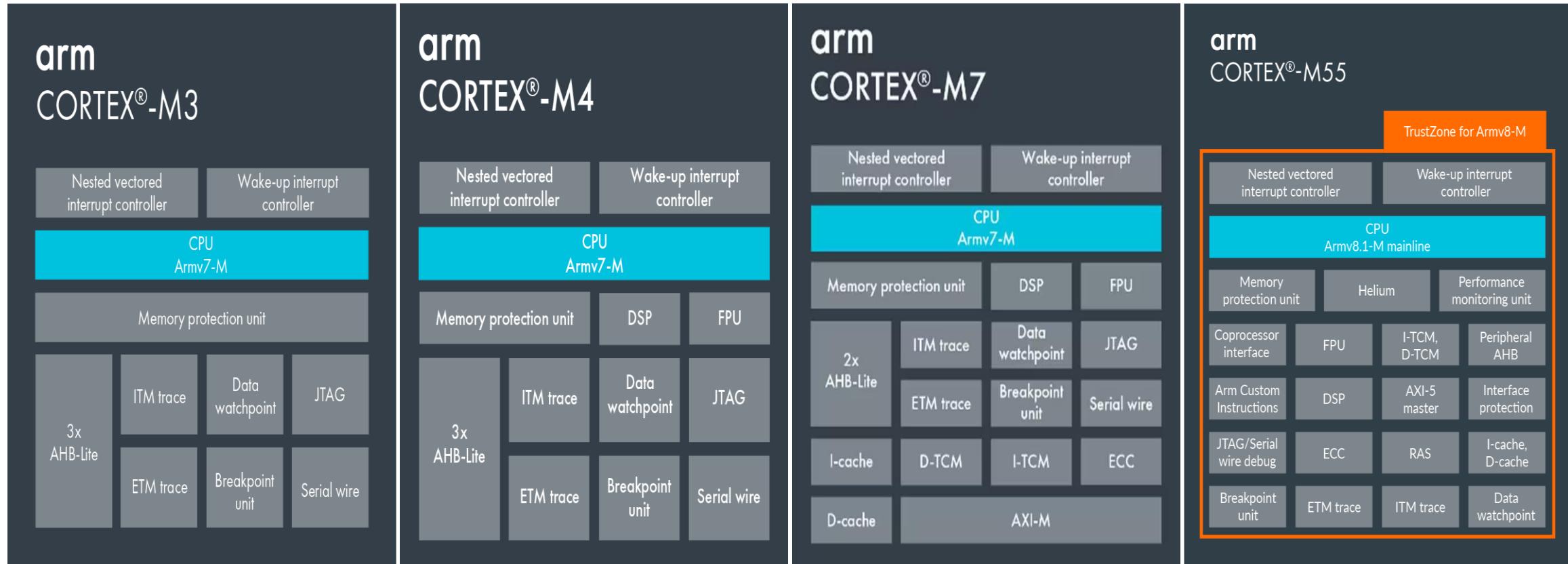
本課程完全免費，請勿移作商業用途！
歡迎留言、訂閱、點讚、轉發，讓更多需要的朋友也能一起學習。

完整課程大綱：<https://omnixri.blogspot.com/2025/03/omnixri-tinyml-2025-0.html>
課程直播清單：<https://www.youtube.com/@omnixri1784streams>



5.1. CMSIS 基本介紹

arm Cortex-M 架構差異



**MCU 的多樣性
造成開發的問題**

矽智財：內核不同、週邊不同、記憶體位址不同、安全性不同 ...
實體晶片：封裝不同、接腳數不同、電氣性能不同 ...
開發工具：IDE不同、程式語言不同、編譯器不同 ...

何謂 arm CMSIS

arm Cortex 微控制器（單晶片）軟體介面標準 (Cortex Microcontroller Software Interface Standard, CMSIS)

- 供應商無關之硬體抽象層，提供統一的外部設備（Peripheral, 簡稱外設）軟體介面。
- 符合 ANSI C (C99) 及 C++ (C++03) 語法，支援多種編譯器。
- 提升軟體重用性及移植性、加快學習時間、降低開發成本。



CMSIS-5

 CMSIS Version 5.9.0
Common Microcontroller Software Interface Standard

(2022/5/2 發行)

General	Core(A)	Core(M)	Driver	DSP	NN	RTOS v1	RTOS v2	Pack	Build	SVD	DAP	Zone
Main Page	Usage and Description											

Introduction

The CMSIS is a set of tools, APIs, frameworks, and work flows that help to simplify software re-use, reduce the learning curve for microcontroller developers, speed-up project build and debug, and thus reduce the time to market for new applications.

CMSIS started as a vendor-independent hardware abstraction layer Arm® Cortex®-M based processors and was later extended to support entry-level Arm Cortex-A based processors. To simplify access, CMSIS defines generic tool interfaces and enables consistent device support by providing simple software interfaces to the processor and the peripherals.

CMSIS is defined in close cooperation with various silicon and software vendors and provides a common approach to interface to peripherals, real-time operating systems, and middleware components. It is intended to enable the combination of software components from multiple vendors.

CMSIS is open-source and collaboratively developed on [GitHub](#).

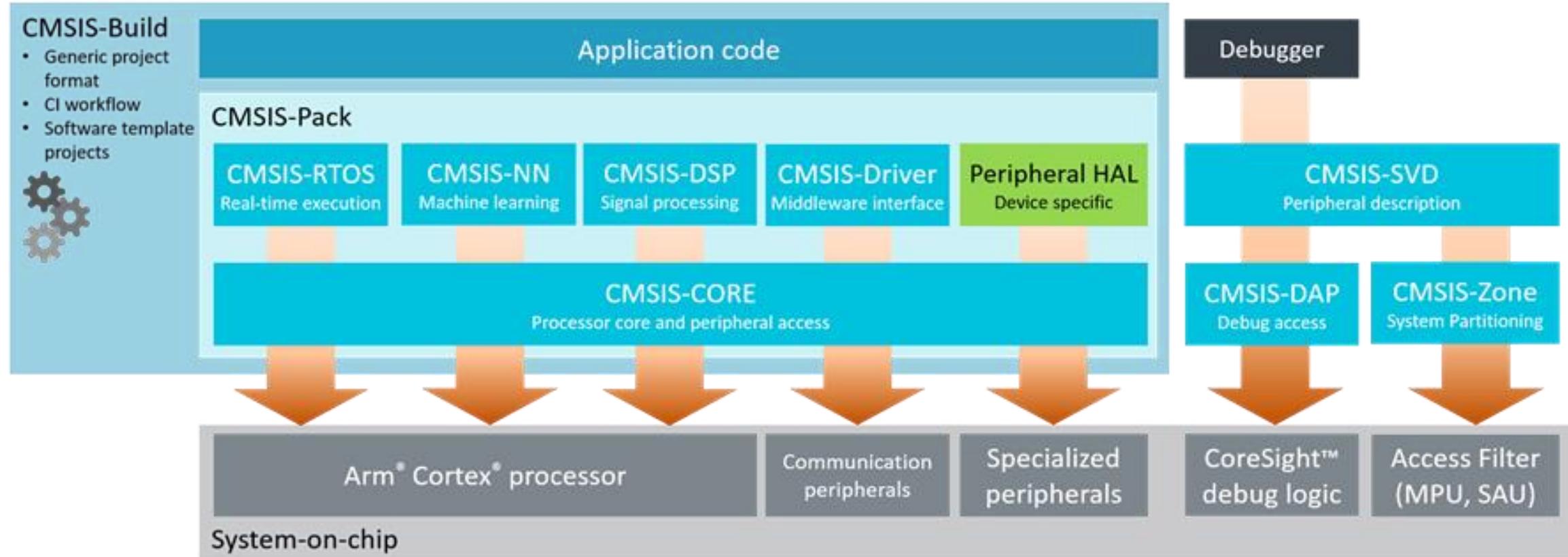
CMSIS Components

CMSIS-...	Target Processors	Description
Core(M)	All Cortex-M, SecurCore	Standardized API for the Cortex-M processor core and peripherals. Includes intrinsic functions for Cortex-M4/M7/M33/M35P SIMD instructions.
Core(A)	Cortex-A5/A7/A9	Standardized API and basic run-time system for the Cortex-A5/A7/A9 processor core and peripherals.
Driver	All Cortex	Generic peripheral driver interfaces for middleware. Connects microcontroller peripherals with middleware that implements for example communication stacks, file systems, or graphic user interfaces.
DSP	All Cortex-M	DSP library collection with over 60 functions for various data types: fixed-point (fractional q7, q15, q31) and single precision floating-point (32-bit). Implementations optimized for the SIMD instruction set are available for Cortex-M4/M7/M33/M35P.

Core(A)
Core(M)
Driver
DSP
NN
RTOS v1
RTOS v2
Pack
Build
SVD
DAP
Zone

https://arm-software.github.io/CMSIS_5/latest/General/html/index.html

CMSIS-5 架構圖



資料來源：https://arm-software.github.io/CMSIS_5/latest/General/html/index.html

CMSIS-6



CMSIS Version 6.1.0

Common Microcontroller Software Interface Standard

V6.0.0 2023/12/8 發行

Overview Core Driver RTOS2 DSP NN View Compiler Toolbox Stream DAP Zone

CMSIS Base Software Components

- Provide software abstractions for basic level functionalities of a device.
- Maintained in the same GitHub repository and delivered as one [CMSIS Software Pack](#) with the name `Arm::CMSIS`.

CMSIS-Core Standardized access to Arm Cortex processor cores Guide GitHub Pack	CMSIS-Driver Generic peripheral driver interfaces for middleware Guide GitHub Pack	CMSIS-RTOS2 Common API for real-time operating systems Guide GitHub Pack
---	---	---

CMSIS Extended Software Components

- Implement specific functionalities optimized for execution on Arm processors.
- Maintained in separate GitHub repositories and delivered in standalone CMSIS-Packs.

CMSIS-DSP Optimized compute functions	CMSIS-NN Efficient and performant neural	CMSIS-View Event Recorder and Component
---	--	---

Core
Driver
RTOS2
DSP
NN
View
Compiler
Toolbox
Stream
DAP
Zone

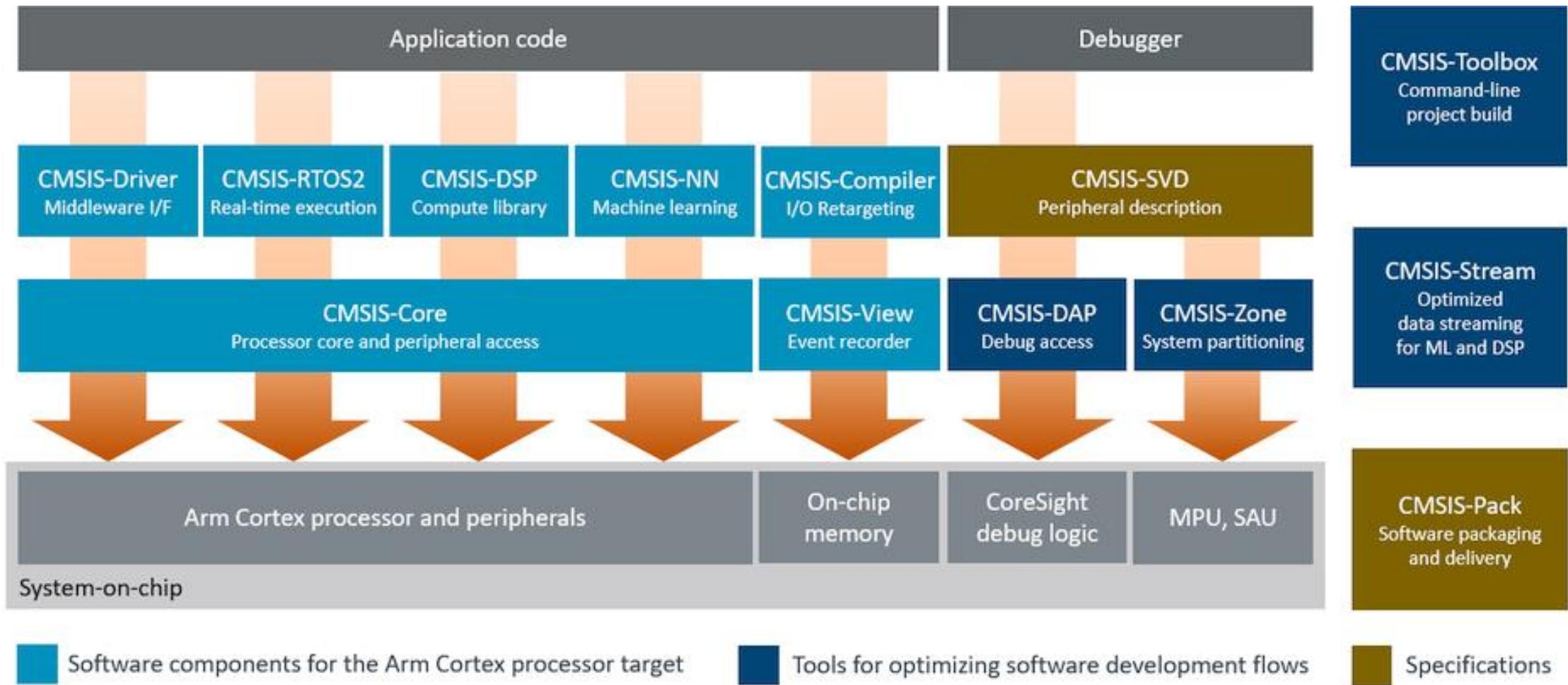
https://arm-software.github.io/CMSIS_6/latest/General/index.html

CMSIS-6 主要元件功能

元件名稱	功能說明	元件名稱	功能說明
Core	對Cortex內核的標準化處理	Toolbox	一套用於套件的命令行工具
Driver	中間層通用外設驅動介面	Stream	用於優化DSP/ML區塊數據流工具
RTOS2	即時作業系統通用API	DAP	連接到除錯埠除錯之韌體
DSP	嵌入式系統最佳化計算函式	Zone	定義用於描述系統並分區的方法
NN	高效及高性能神經網路內核	Pack	軟體元件和設備支援交付機制
View	事件記錄器及元件觀察器技術	SVD	用於除錯設備的外圍設備描述
Compiler	重定向標準C運行時庫的I/O函數		

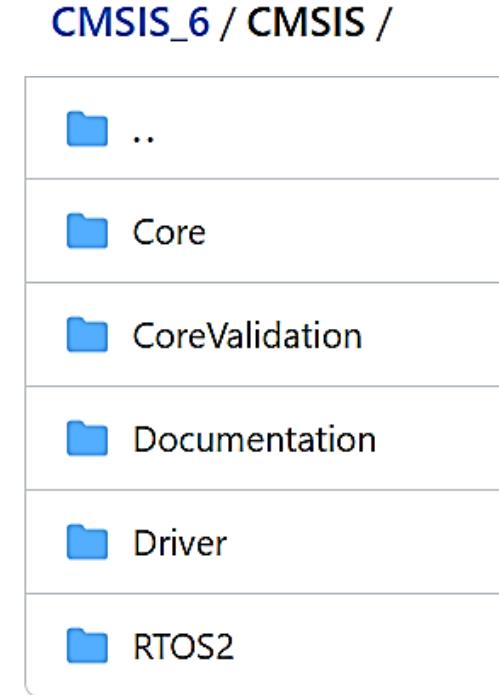
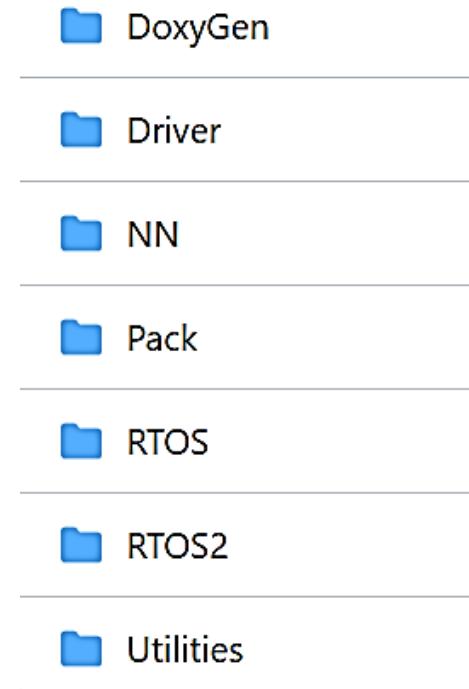
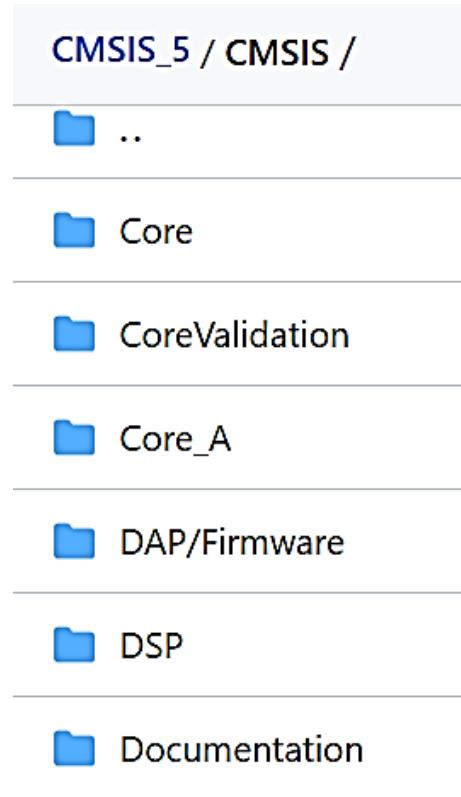
資料來源：https://arm-software.github.io/CMSIS_6/latest/General/index.html

CMSIS-6 架構圖



資料來源：https://arm-software.github.io/CMSIS_6/latest/General/index.html

CMSIS 5 / 6 Github 比較



https://github.com/ARM-software/CMSIS_5/tree/develop/CMSIS

https://github.com/ARM-software/CMSIS_6/tree/main/CMSIS

CMSIS 5 / 6 主要差異

項目	CMSIS-5	CMSIS-6
支持硬體	Cortex-M0,M0+,M3,M4,M7	新增Cortex-M23,M33,M55,M85
安全特性	不支持 TrustZone	原生支持 TrustZone
CMSIS-Pack改進	基礎功能	改進和MDK及其它IDE的整合
DSP & ML	提供基礎DSP, NN功能	擴展NN庫，滿足更複雜模型
工具鏈和兼容性	支援 GCC, IAR, Keil	改善現代工具鏈兼容性便於遷移
配置文件和標準化	基礎設備頭文件及啟動碼	改善設備配置文件更靈活應用

註：大多數應用使用 CMSIS-5 就能滿足，若要應用於 MCU AI 則建議改用 CMSIS-6



5.2. CMSIS-Core & Driver

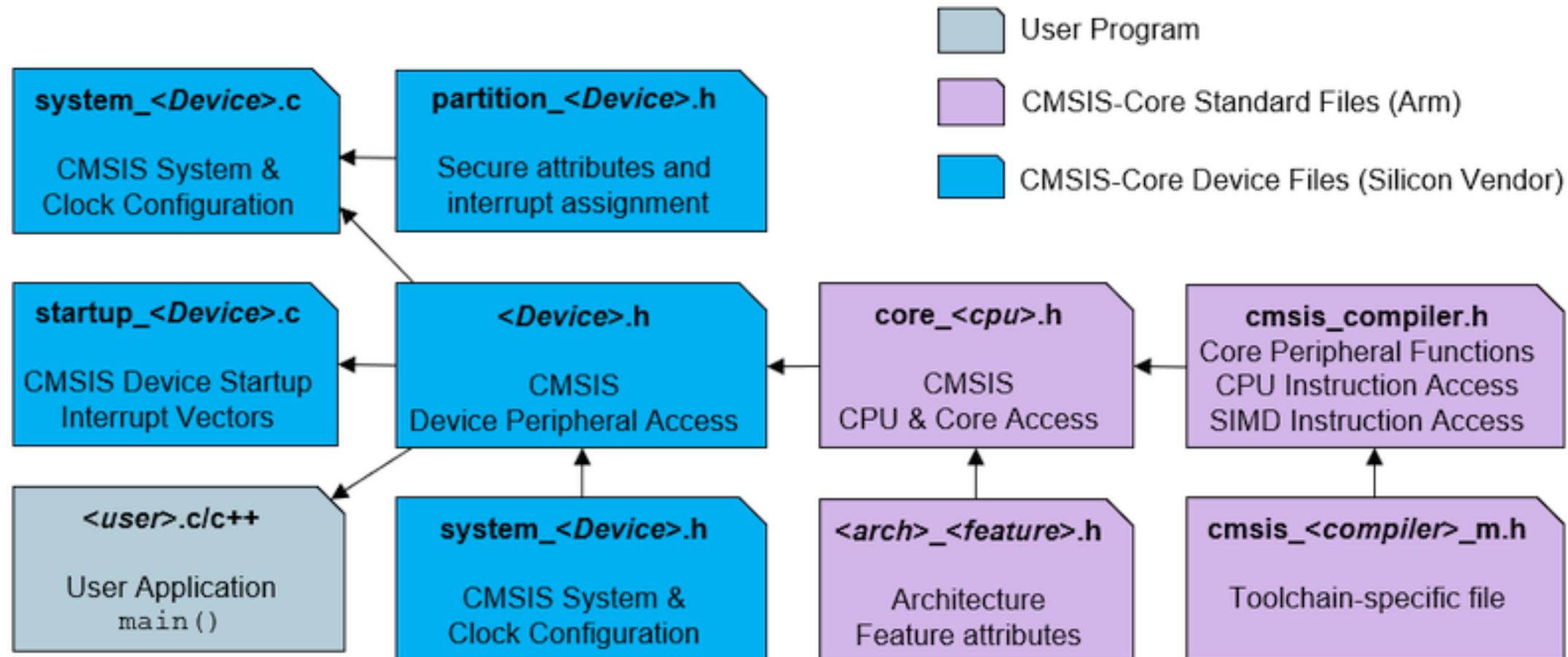
CMSIS-6 Core

CMSIS-Core (Cortex-M) 元件實現了 Arm Cortex-M 設備的基本運行時系統，並允許使用者訪問處理器內核(Core)和週邊裝置(Peripherals)。

Directory	Content
📁 CMSIS	CMSIS Base software components folder
↳ 📁 Documentation/html/Core	A local copy of this CMSIS-Core (M) documentation
↳ 📁 Core	CMSIS-Core files
↳ 📁 Include	CMSIS-Core Processor Files.
↳ 📁 m-profile	Header files specific for Arm M-Profile. See CMSIS-Core Compiler Files and CMSIS-Core Architecture Feature Files .
↳ 📁 Template	Device Template Files

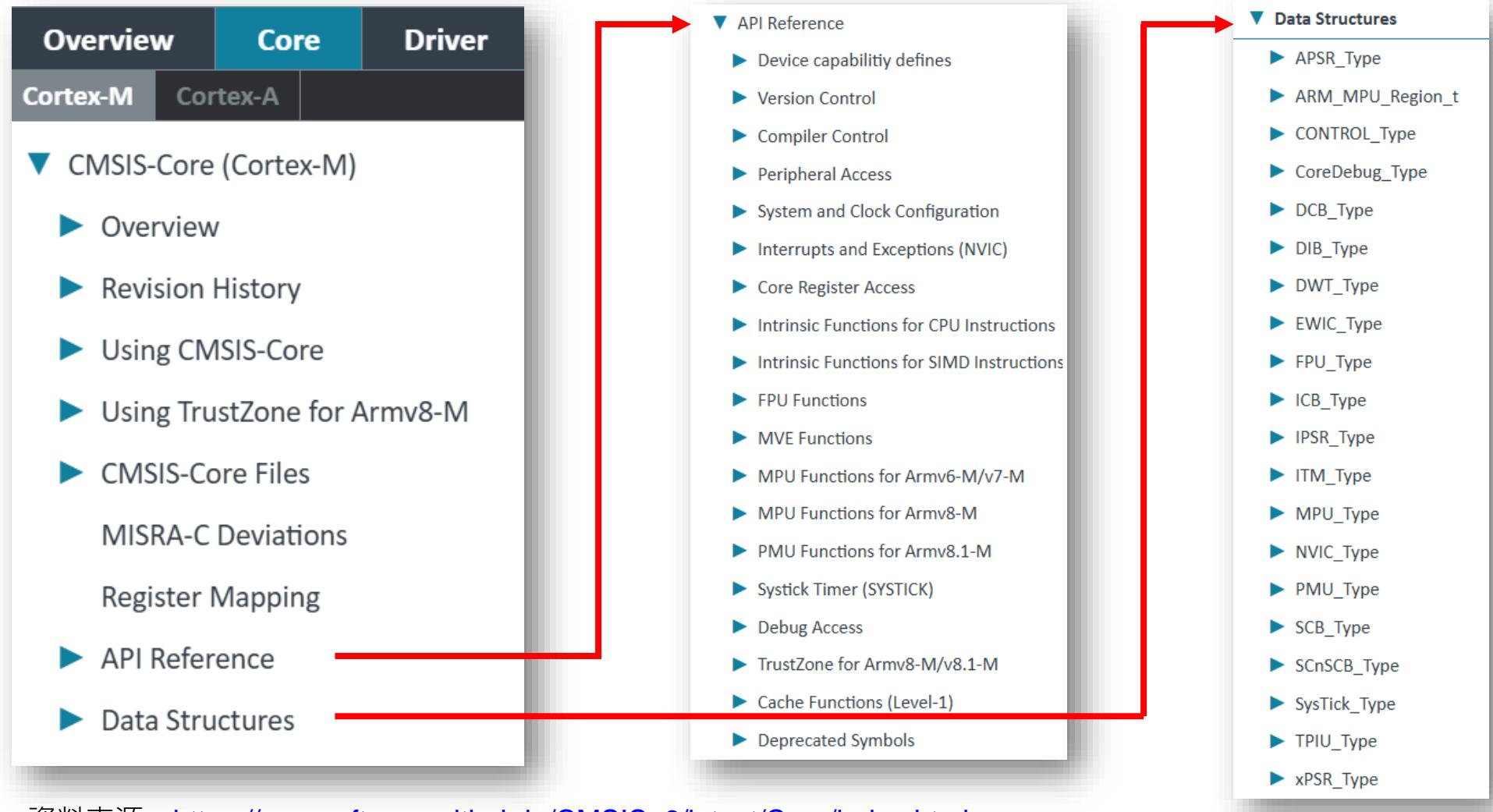
資料來源：https://arm-software.github.io/CMSIS_6/latest/Core/index.html

CMSIS-6 Core 相關檔案



資料來源：https://arm-software.github.io/CMSIS_6/latest/Core/index.html

CMSIS-6 Core 說明文件



The screenshot shows the CMSIS-6 Core API Reference page. On the left, there's a navigation bar with tabs for Overview, Core (selected), and Driver, and sub-tabs for Cortex-M and Cortex-A. Below this is a list of sections: CMSIS-Core (Cortex-M) with sub-sections like Overview, Revision History, Using CMSIS-Core, etc.; MISRA-C Deviations; Register Mapping; API Reference (which is the current page); and Data Structures. Two red arrows point from the 'API Reference' section in the sidebar to the 'API Reference' and 'Data Structures' sections in the main content area.

- ▼ API Reference
 - ▶ Device capability defines
 - ▶ Version Control
 - ▶ Compiler Control
 - ▶ Peripheral Access
 - ▶ System and Clock Configuration
 - ▶ Interrupts and Exceptions (NVIC)
 - ▶ Core Register Access
 - ▶ Intrinsic Functions for CPU Instructions
 - ▶ Intrinsic Functions for SIMD Instructions
 - ▶ FPU Functions
 - ▶ MVE Functions
 - ▶ MPU Functions for Armv6-M/v7-M
 - ▶ MPU Functions for Armv8-M
 - ▶ PMU Functions for Armv8.1-M
 - ▶ Systick Timer (SYSTICK)
 - ▶ Debug Access
 - ▶ TrustZone for Armv8-M/v8.1-M
 - ▶ Cache Functions (Level-1)
 - ▶ Deprecated Symbols
- ▼ Data Structures
 - ▶ APSR_Type
 - ▶ ARM_MPU_Region_t
 - ▶ CONTROL_Type
 - ▶ CoreDebug_Type
 - ▶ DCB_Type
 - ▶ DIB_Type
 - ▶ DWT_Type
 - ▶ EWIC_Type
 - ▶ FPU_Type
 - ▶ ICB_Type
 - ▶ IPSR_Type
 - ▶ ITM_Type
 - ▶ MPU_Type
 - ▶ NVIC_Type
 - ▶ PMU_Type
 - ▶ SCB_Type
 - ▶ SCnSCB_Type
 - ▶ SysTick_Type
 - ▶ TPIU_Type
 - ▶ xPSR_Type

資料來源：https://arm-software.github.io/CMSIS_6/latest/Core/index.html

CMSIS-6 Core 範例程式

```
#include <stm32f10x.h>

uint32_t volatile msTicks;

void SysTick_Handler (void) {
    msTicks++;
}

void WaitForTick (void) {
    uint32_t curTicks;

    curTicks = msTicks;
    while (msTicks == curTicks) {
        __WFE ();
    }
}

void TIM1_UP_IRQHandler (void) {
    ;
}

void timer1_init(int frequency) {
    NVIC_SetPriority (TIM1_UP_IRQn, 1);
    NVIC_EnableIRQ (TIM1_UP_IRQn);
}
```

```
void Device_Initialization (void)          // Configure & Initialize MCU
{
    if (SysTick_Config (SystemCoreClock / 1000)) { // SysTick 1mSec
        : // Handle Error
    }
    timer1_init ();                         // setup device-specific timer
}

// The processor clock is initialized by CMSIS startup + system file
void main (void) {                      // user application starts here
    Device_Initialization ();           // Configure & Initialize MCU
    while (1) {                        // Endless Loop (the Super-Loop)
        __disable_irq ();              // Disable all interrupts
        Get_InputValues ();           // Read Values
        __enable_irq ();              // Enable all interrupts
        Calculation_Response ();     // Calculate Results
        Output_Response ();          // Output Results
        WaitForTick ();               // Synchronize to SysTick Timer
    }
}
```

基於 STM32F10x MCU，使用 CMSIS 範例。

資料來源：https://arm-software.github.io/CMSIS_6/latest/Core/index.html

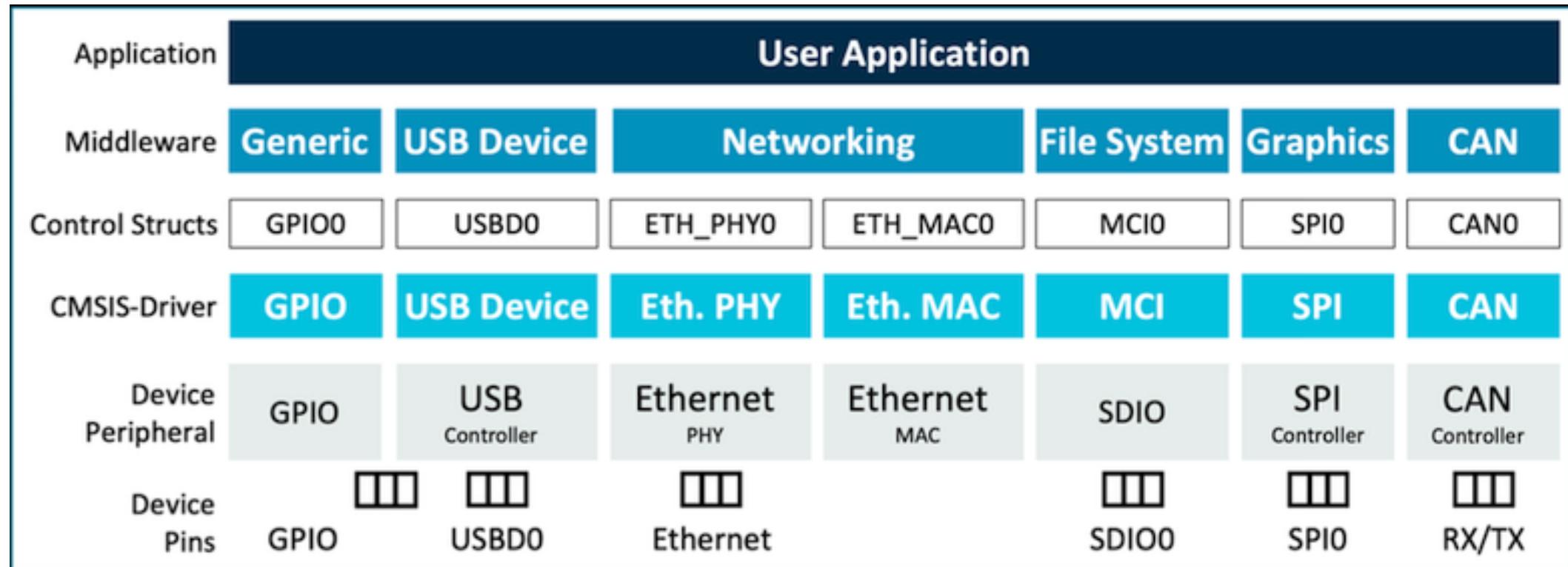
CMSIS-6 Driver

使用者應用程式和中間件元件可以使用 CMSIS-Driver API 控制此類週邊設備，從而在各種生態系統中實現更好的代碼重用和更簡單的集成。

Directory	Content
📁 CMSIS	CMSIS Base software components folder
↳ 📁 Documentation/html/Driver	A local copy of this CMSIS-Driver documentation
↳ 📁 Driver	Directory with CMSIS-Driver component, see CMSIS-Driver Files
↳ 📁 DriverTemplates	Driver Template files (<i>Driver_interface.c</i>)
↳ 📁 Include	API header files (<i>Driver_interface.h</i> , <i>Driver_Common.h</i>)
↳ 📁 VIO	Implementation of virtual Input/Output interface (VIO)

資料來源：https://arm-software.github.io/CMSIS_6/latest/Driver/index.html

CMSIS-6 Driver 架構圖



包括 CAN, Ethernet, I2C, MCI, NAND, Flash, SAI, API, Storage, USART, USB, GPIO, VIO, WiFi 等驅動定義

資料來源：https://arm-software.github.io/CMSIS_6/latest/Driver/index.html

CMSIS-6 Driver 工作流程

中間層軟體

確認 API 版本

獲取驅動特徵

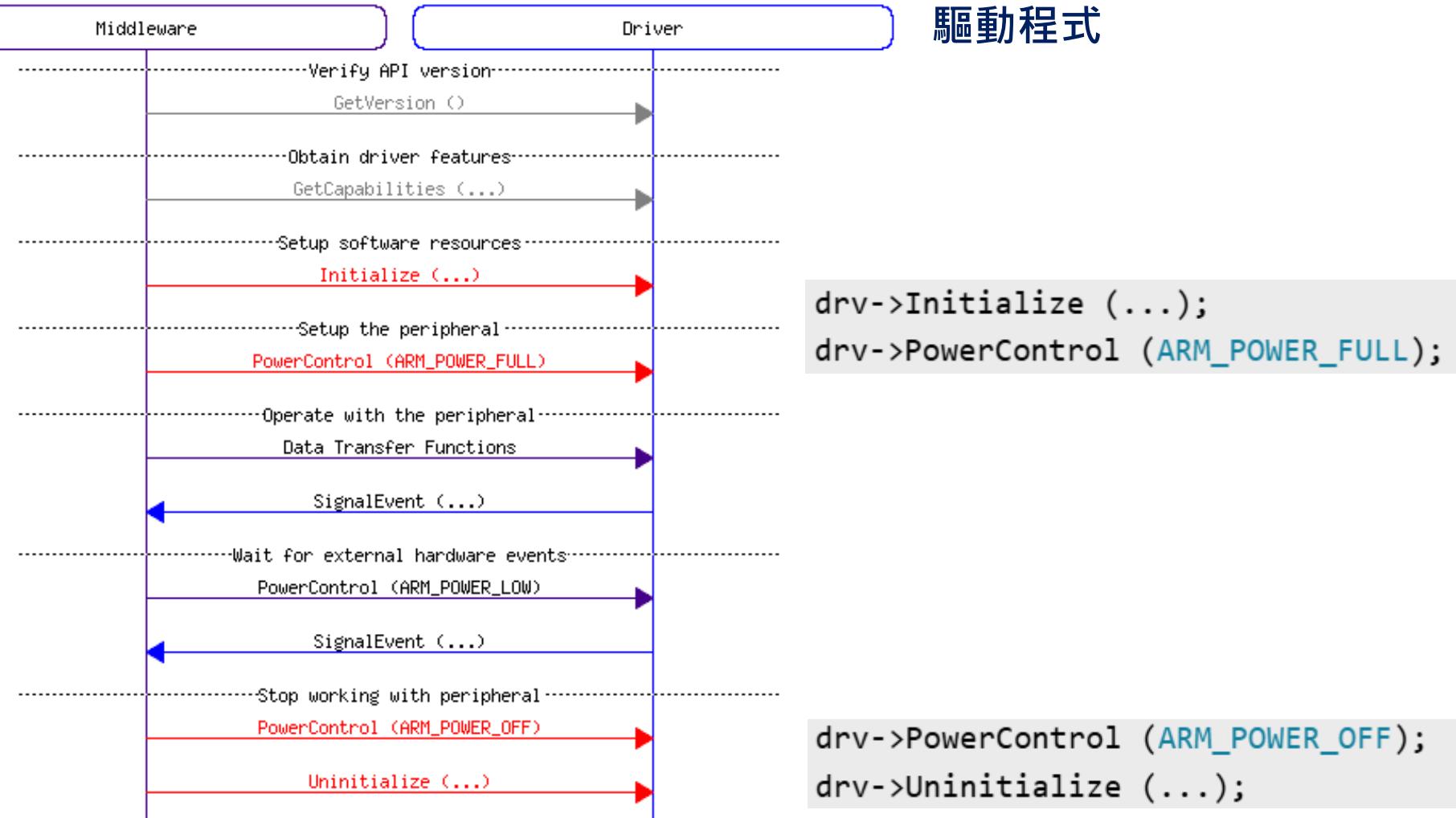
設定軟體資源

設定週邊裝置

操作週邊裝置

等待硬體事件

停止週邊裝置



資料來源：https://arm-software.github.io/CMSIS_6/latest/Driver/index.html

CMSIS-6 Driver 相關檔案

Header File	Template File	API Reference		
Driver_Common.h	Not applicable	Common Driver Definitions		
Driver_CAN.h	Driver_CAN.c	CAN Interface		
Driver_ETH.h	-	Ethernet Interface		
Driver_ETH_MAC.h	Driver_ETH_MAC.c	Ethernet MAC Interface		
Driver_ETH_PHY.h	Driver_ETH_PHY.c	Ethernet PHY Interface		
Driver_Flash.h	Driver_Flash.c	Flash Interface		
Driver_GPIO.h	Driver_GPIO.c	GPIO Interface		
Driver_I2C.h	Driver_I2C.c	I2C Interface		
Driver_MCI.h	Driver_MCI.c	MCI Interface		
			Driver_NAND.h	Driver_NAND.c
			Driver_SAI.h	Driver_SAI.c
			Driver_SPI.h	Driver_SPI.c
			Driver_Storage.h	Driver_Storage.c
			Driver_USART.h	Driver_USART.c
			Driver_USB.h	-
			Driver_USBD.h	Driver_USBD.c
			Driver_USBH.h	Driver_USBH.c
			Driver_WiFi.h	Driver_WiFi.c

資料來源：https://arm-software.github.io/CMSIS_6/latest/Driver/index.html

CMSIS-6 Driver – GPIO 為例

[CMSIS_6 / CMSIS / Driver / Include / Driver_GPIO.h](#)

Code	Blame	154 lines (127 loc) • 5.49 KB
37 /*		
38 typedef uint32_t ARM_GPIO_Pin_t;		
39		
40 /**		
41 \brief GPIO Direction		
42 */		
43 typedef enum {		
44 ARM_GPIO_INPUT, ///< Input (default)		
45 ARM_GPIO_OUTPUT, ///< Output		
46 } ARM_GPIO_DIRECTION;		
47		
48 /**		
49 \brief GPIO Output Mode		
50 */		
51 typedef enum {		
52 ARM_GPIO_PUSH_PULL, ///< Push-pull (default)		
53 ARM_GPIO_OPEN_DRAIN ///< Open-drain		
54 } ARM_GPIO_OUTPUT_MODE;		
55		
56 /**		
57 \brief GPIO Pull Resistor		
58 */		
59 typedef enum {		
60 ARM_GPIO_PULL_NONE, ///< None (default)		
61 ARM_GPIO_PULL_UP, ///< Pull-up		
62 ARM_GPIO_PULL_DOWN ///< Pull-down		
63 } ARM_GPIO_PULL_RESISTOR;		

GPIO方向

GPIO輸出模式

GPIO上拉電阻

[CMSIS_6 / CMSIS / Driver / DriverTemplates / Driver_GPIO.c](#)

Code	Blame	154 lines (130 loc) • 3.37 KB
19 #include "Driver_GPIO.h"		
20		
21 // Pin mapping		
22 #define GPIO_MAX_PINS 64U		
23 #define PIN_IS_AVAILABLE(n) ((n) < GPIO_MAX_PINS)		
24		
25		
26 // Setup GPIO Interface		
27 static int32_t GPIO_Setup (ARM_GPIO_Pin_t pin, ARM_GPIO_SignalEvent_t cb_event) {		
28 int32_t result = ARM_DRIVER_OK;		
29		
30 if (PIN_IS_AVAILABLE(pin)) {		
31 } else {		
32 result = ARM_GPIO_ERROR_PIN;		
33 }		
34		
35 return result;		
36 }		
37		
38 // Set GPIO Direction		
39 static int32_t GPIO_SetDirection (ARM_GPIO_Pin_t pin, ARM_GPIO_DIRECTION direction) {		
40 int32_t result = ARM_DRIVER_OK;		
41		
42 if (PIN_IS_AVAILABLE(pin)) {		
43 switch (direction) {		
44 case ARM_GPIO_INPUT:		
45 break;		

設定GPIO界面

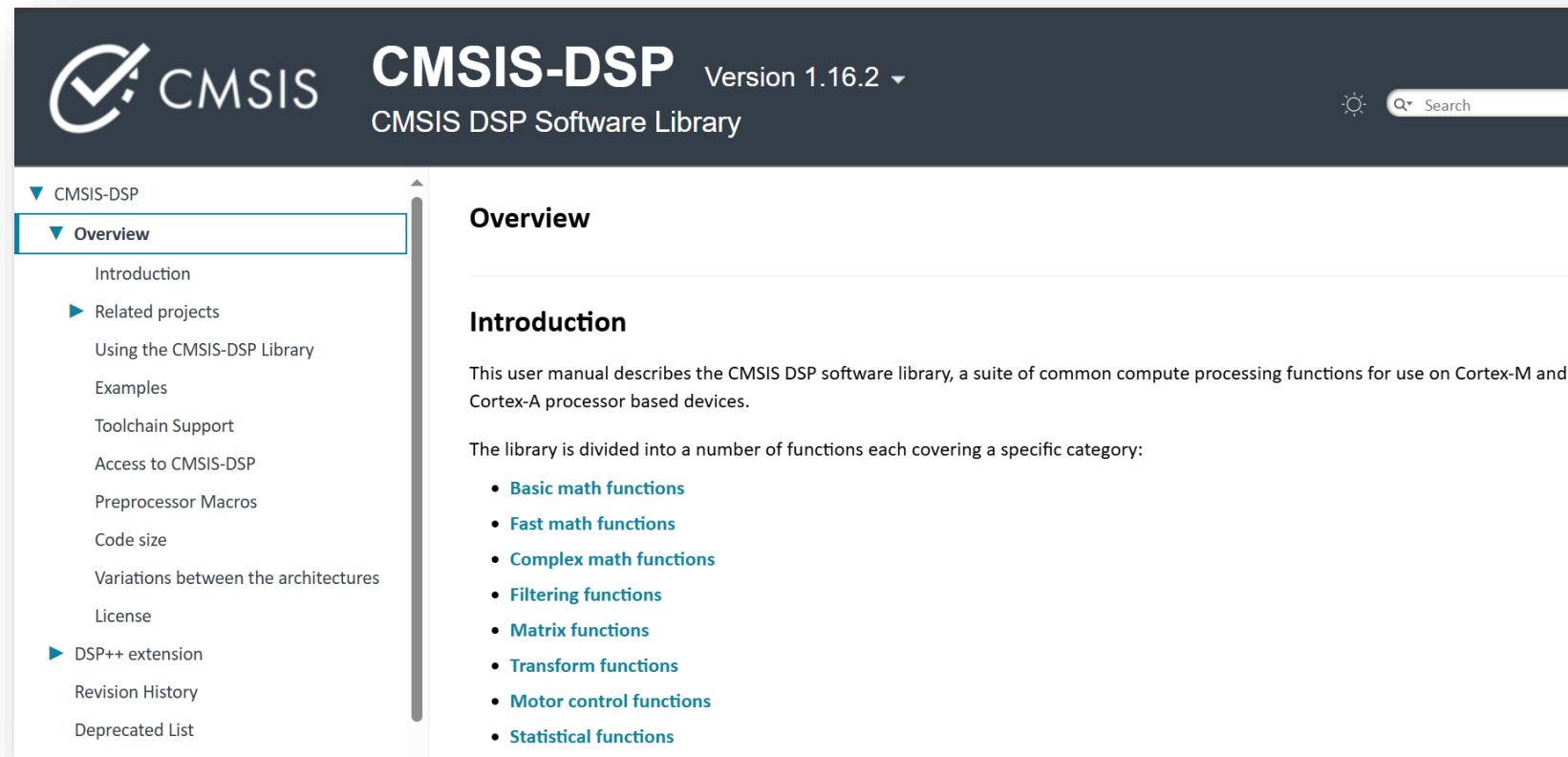
設定GPIO方向



5.3. CMSIS-DSP

CMSIS-6 DSP

CMSIS-DSP 是一個開源軟體庫（在Github已獨立），可實現針對 Arm Cortex-M 和 Cortex-A 處理器進行了優化的常見計算處理功能。



資料來源：<https://arm-software.github.io/CMSIS-DSP/latest/>

- 基本數學函數
- 快速數學函數
- 複數數學函數
- 濾波器函數
- 矩陣函數
- 變換函數
- 馬達控制函數
- 統計函數
- 支援函數
- 插值函數
- 支持向量機函數
- 貝葉斯分類器函數
- 距離函數
- 四元數函數
- 視窗函數

CMSIS-6 DSP – 通用格式

typedef int8_t q7_t 8-bit fraction	tvnedef int16x8x2_t q63x2_t vector types	q15x8x2_t 16-bit fractional	status64x2_t 64-bit status 12	f32x4x4_t 32-bit floating	edef int32x2x2_t q31x4x3_t 32-bit fraction	q31x2x2_t 32-bit fraction	q7x8x2_t 8-bit fractional
typedef int16_t q15_t 16-bit fraction	q31x4_t 32-bit fractional 12	q15x8x4_t 16-bit fractional	status32x4_t 32-bit status 12	q31x2_t 32-bit fraction	q15x8x3_t 16-bit fraction	q31x2x3_t 32-bit fraction	q7x8x3_t 8-bit fractional
typedef int32_t q31_t 32-bit fraction	q15x8_t 16-bit fractional 12	q7x16x2_t 8-bit fractional 1	status16x8_t 16-bit status 12	q15x4_t 16-bit fraction	q7x16x3_t 8-bit fraction	q31x2x4_t 32-bit fraction	q7x8x4_t 8-bit fractional
typedef int64_t q63_t 64-bit fraction	q7x16_t 8-bit fractional 128	q7x16x4_t 8-bit fractional 1	status8x16_t 8-bit status 128	q7x8_t 8-bit fraction	f32x2x2_t 32-bit floating	q15x4x2_t 16-bit fraction	status32x2_t 32-bit status 64
typedef float float32_t 32-bit floating	q31x4x2_t 32-bit fractional 12	q23_t 32-bit fractional	f32x4_t 32-bit floating-p	f32x2_t 32-bit float 6	f32x2x3_t 32-bit floating	q15x4x3_t 16-bit fraction	status16x4_t 16-bit status 64
typedef int32x4x4_t q31x4x4_t 32-bit fractional 12	q23x4_t 32-bit fractional	f32x4x2_t 32-bit floating-p	f32x4x3_t 32-bit floating	f32x2x4_t 32-bit floating	q15x4x4_t 16-bit fraction	status8x8_t 8-bit status 64-	

資料來源：<https://arm-software.github.io/CMSIS-DSP/latest/>

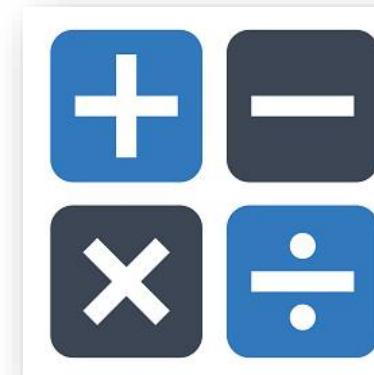
CMSIS-6 DSP – 數學函數

基本數學函式

Vector Absolute Value
 Vector Addition
 Vector bitwise AND
 Elementwise clipping
 Vector Dot Product
 Vector Multiplication
 Vector Negate
 Vector bitwise NOT
 Vector Offset
 Vector bitwise inclusive OR
 Vector Scale
 Vector Shift
 Vector Subtraction
 Vector bitwise exclusive OR

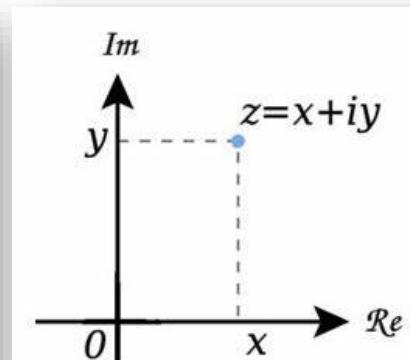
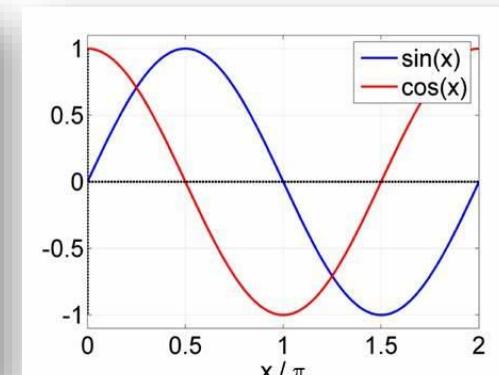
快速數學函式

ArcTan2
 Cosine
 Fixed point division
 Sine
 Vector Exponential
 Vector Log
 Square Root



複數數學函式

Complex Conjugate
 Complex Dot Product
 Complex Magnitude
 Complex Magnitude Squared
 Complex-by-Complex Multiplication
 Complex-by-Real Multiplication



資料來源：https://arm-software.github.io/CMSIS_6/latest/DSP/index.html

CMSIS-6 DSP – 濾波器函數

High Precision Q31 Biquad Cascade Filter

Biquad Cascade IIR Filters Using Direct Form I Structure

Biquad Cascade IIR Filters Using a Direct Form II Transposed Structure

Convolution

Partial Convolution

Correlation

Finite Impulse Response (FIR) Decimator

Finite Impulse Response (FIR) Filters

Finite Impulse Response (FIR) Lattice Filters

Finite Impulse Response (FIR) Sparse Filters

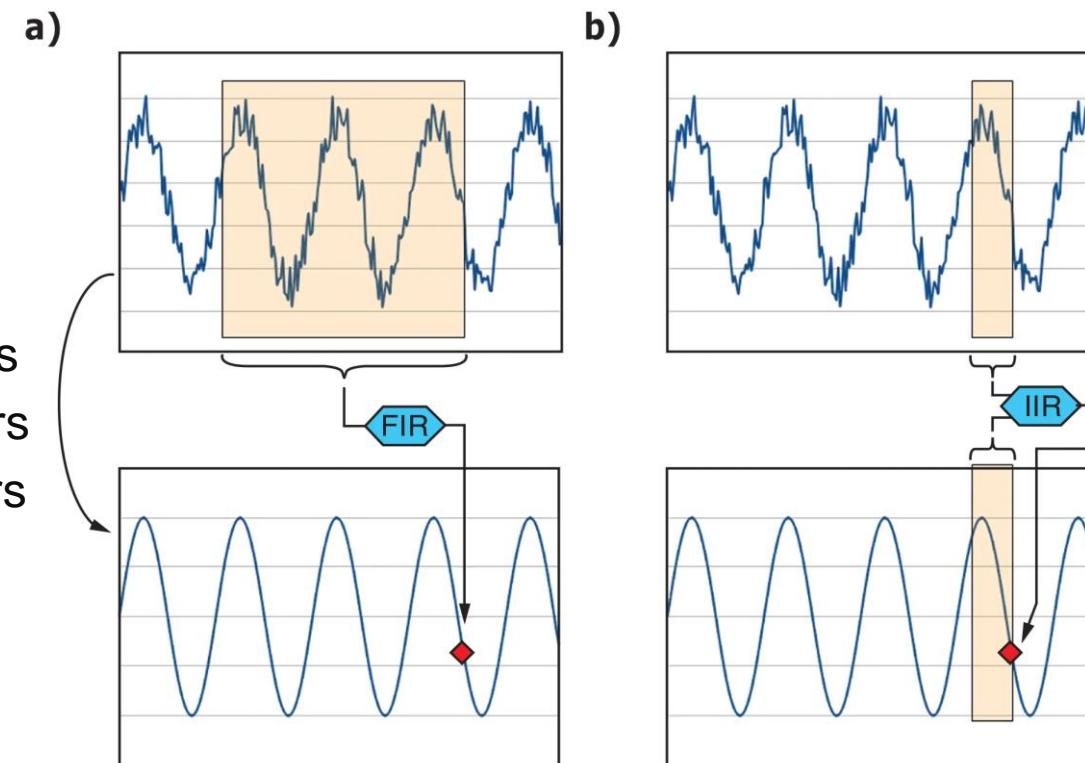
Infinite Impulse Response (IIR) Lattice Filters

Levinson Durbin Algorithm

Least Mean Square (LMS) Filters

Normalized LMS Filters

Finite Impulse Response (FIR) Interpolator



資料來源：https://arm-software.github.io/CMSIS_6/latest/DSP/index.html

CMSIS-6 DSP – 矩陣函式

Householder transform of a vector

Matrix Addition

Cholesky and LDLT decompositions

Complex Matrix Multiplication

Complex Matrix Transpose

Matrix Initialization

Matrix Inverse

Matrix Multiplication

QR decomposition of a Matrix

Matrix Scale

Matrix Subtraction

Matrix Transpose

Matrix Vector Multiplication

$$\begin{bmatrix} \mathbf{a}_{11} & \mathbf{a}_{12} & \mathbf{a}_{13} \\ \mathbf{a}_{21} & \mathbf{a}_{22} & \mathbf{a}_{23} \end{bmatrix} X \begin{bmatrix} \mathbf{b}_{11} & \mathbf{b}_{12} \\ \mathbf{b}_{21} & \mathbf{b}_{22} \\ \mathbf{b}_{31} & \mathbf{b}_{32} \end{bmatrix} = \begin{bmatrix} \mathbf{c}_{11} & \mathbf{c}_{12} \\ \mathbf{c}_{21} & \mathbf{c}_{22} \end{bmatrix}$$

↑ ↓

$$\mathbf{c}_{11} = \mathbf{a}_{11} * \mathbf{b}_{11} + \mathbf{a}_{12} * \mathbf{b}_{21} + \mathbf{a}_{13} * \mathbf{b}_{31}$$

$$\mathbf{c}_{12} = \mathbf{a}_{11} * \mathbf{b}_{12} + \mathbf{a}_{12} * \mathbf{b}_{22} + \mathbf{a}_{13} * \mathbf{b}_{32}$$

$$\mathbf{c}_{21} = \mathbf{a}_{21} * \mathbf{b}_{11} + \mathbf{a}_{22} * \mathbf{b}_{21} + \mathbf{a}_{23} * \mathbf{b}_{31}$$

$$\mathbf{c}_{22} = \mathbf{a}_{21} * \mathbf{b}_{12} + \mathbf{a}_{22} * \mathbf{b}_{22} + \mathbf{a}_{23} * \mathbf{b}_{32}$$

$$\mathbf{A} = \begin{vmatrix} 2 & 8 & 9 \\ 1 & 9 & 7 \\ 5 & 6 & 3 \\ 0 & 4 & 2 \end{vmatrix}$$

$$\mathbf{A}^T = \begin{vmatrix} 2 & 1 & 5 & 0 \\ 8 & 9 & 6 & 4 \\ 9 & 7 & 3 & 2 \end{vmatrix}$$

資料來源：https://arm-software.github.io/CMSIS_6/latest/DSP/index.html

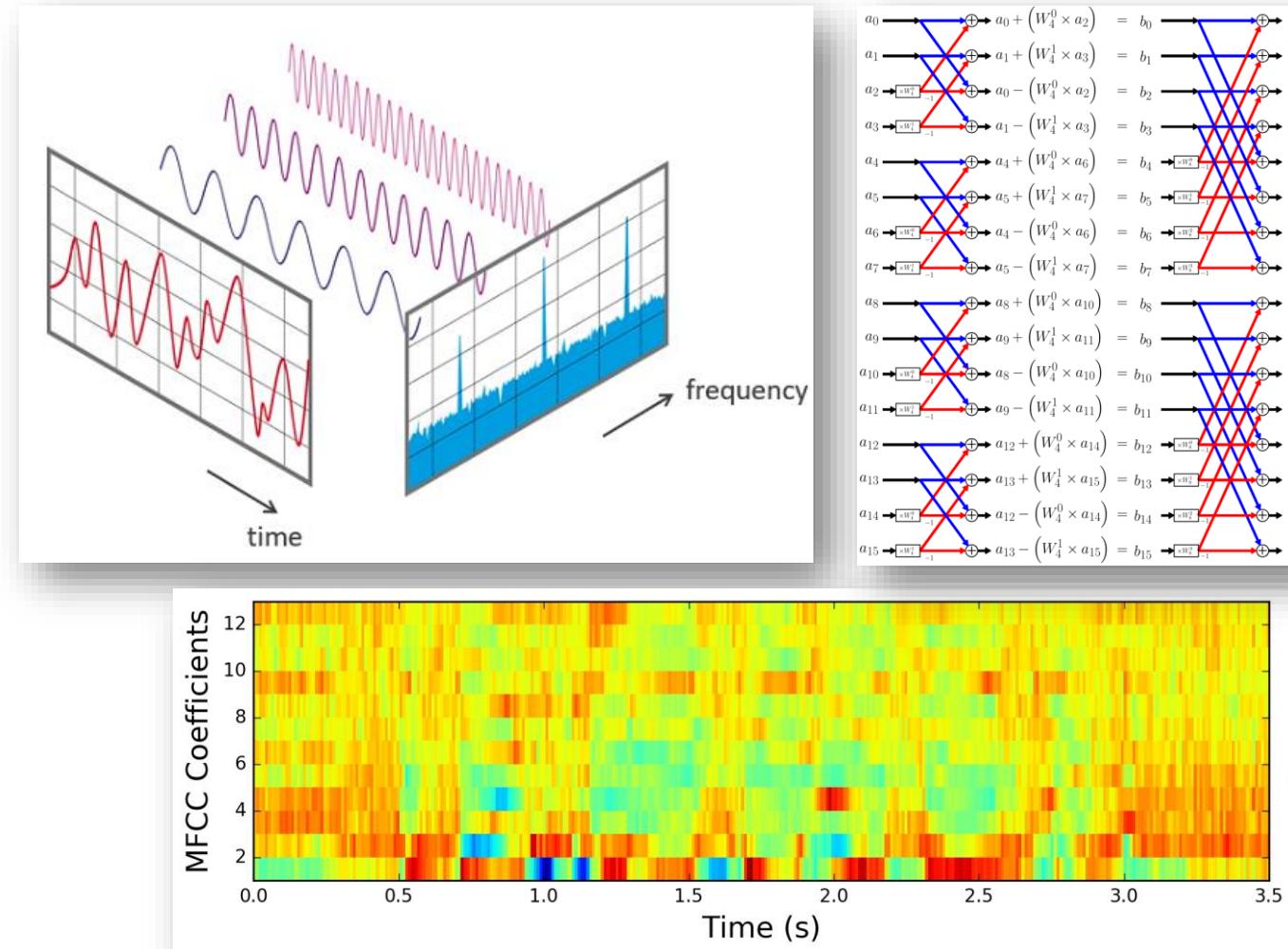
CMSIS-6 DSP – 變換函式

Complex FFT Functions

DCT Type IV Functions

MFCC

Real FFT Functions



資料來源：https://arm-software.github.io/CMSIS_6/latest/DSP/index.html

CMSIS 5 / 6 DSP 比較

主要函數差異不大，CMSIS-6 新增部份函數。

Functions	CMSIS v5 (DSP v1.10.0)			v6 (DSP v1.15.0)
Basic math	Vector Absolute Value Vector Addition Vector bitwise AND Elementwise clipping Vector Dot Product	Vector Multiplication Vector Negate Vector bitwise NOT Vector Offset Vector Scale	Vector Shift Vector Subtraction Vector bitwise inclusive OR Vector bitwise exclusive OR	
Bayes classifier	gaussian_naive_bayes_predict			
Complex math	Complex Conjugate Complex Dot Product Complex Magnitude	Complex Magnitude Squared Complex-by-Complex Multiplication Complex-by-Real Multiplication		
Controller	PID Motor Control Vector Park Transform Vector Inverse Park transform	Vector Clarke Transform Vector Inverse Clarke Transform Sine Cosine		
Distance	Float Distances	Boolean Distances		
Fast math	ArcTan2 Cosine	Fixed point division Sine	Vector Log Square Root	[+] Vector Exponential
Filtering	Biquad Cascade IIR Filters Using Direct Form I Structure Biquad Cascade IIR Filters Using a Direct Form II Transposed Structure Convolution Partial Convolution Correlation Levinson Durbin Algorithm Least Mean Square (LMS) Filters Normalized LMS Filters	High Precision Q31 Biquad Cascade Filter Finite Impulse Response (FIR) Decimator Finite Impulse Response (FIR) Filters Finite Impulse Response (FIR) Lattice Filters Finite Impulse Response (FIR) Sparse Filters Infinite Impulse Response (IIR) Lattice Filters Finite Impulse Response (FIR) Interpolator		

[+] New [-] Remove		Interpolation			Matrix			Quaternion			Statistical			Support			SVM			Transform			Window		
		Bilinear Interpolation	Linear Interpolation	Cubic Spline Interpolation																					
		Matrix Addition	Matrix Scale	Cholesky and LDLT decompositions																					
		Matrix Initialization	Matrix Subtraction	Complex Matrix Multiplication																					
		Matrix Inverse	Matrix Transpose	Complex Matrix Transpose																					
		Matrix Multiplication		Matrix Vector Multiplication																					
		Quaternion conversions	Quaternion Inverse	Quaternion normalization																					
		Quaternion Conjugate	Quaternion Norm	Quaternion Product																					
		Absolute Maximum	Mean	Mean Square Error																					
		Absolute Minimum	Minimum	Kullback-Leibler divergence																					
		Entropy	Power	Root mean square (RMS)																					
		LogSumExp	Variance	Standard deviation																					
		Maximum																							
		Typecasting	Weighted Sum	Convert 16-bit fixed point value																					
		Barycenter	Vector sorting algorithms	Convert 32-bit fixed point value																					
		Vector Copy	Convert 16-bit floating point value	Convert 8-bit fixed point value																					
		Vector Fill	Convert 32-bit floating point value																						
		Linear SVM	RBF SVM																						
		Polynomial SVM	Sigmoid SVM																						
		Complex FFT Functions	MFCC																						
		DCT Type IV Functions	Real FFT Functions																						
				none																					

OmniXRI 整理製作, 2024/02/16

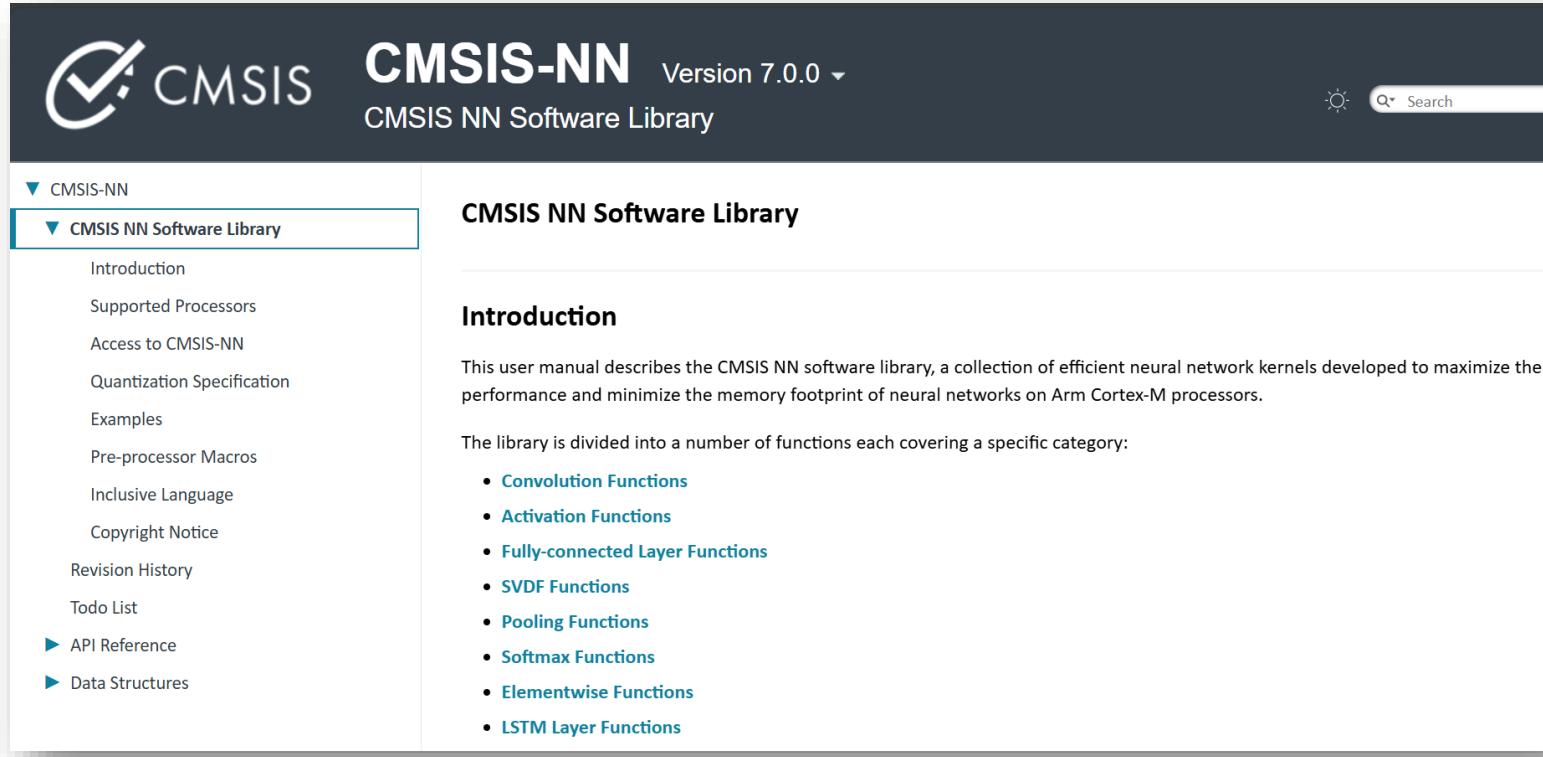
資料來源：<https://omnixri.blogspot.com/2024/02/tinyml-arm-cmsis-6-dsp-nn.html>



5.4. CMSIS-NN

CMSIS-6 NN

CMSIS-NN 是一個開源軟體庫（在Github已獨立），它提供了一組高效的神經網路（NN）內核，這些內核旨在最大限度地提高在 Arm Cortex-M 處理器上運行的神經網路的性能並最大限度地減少記憶體佔用。

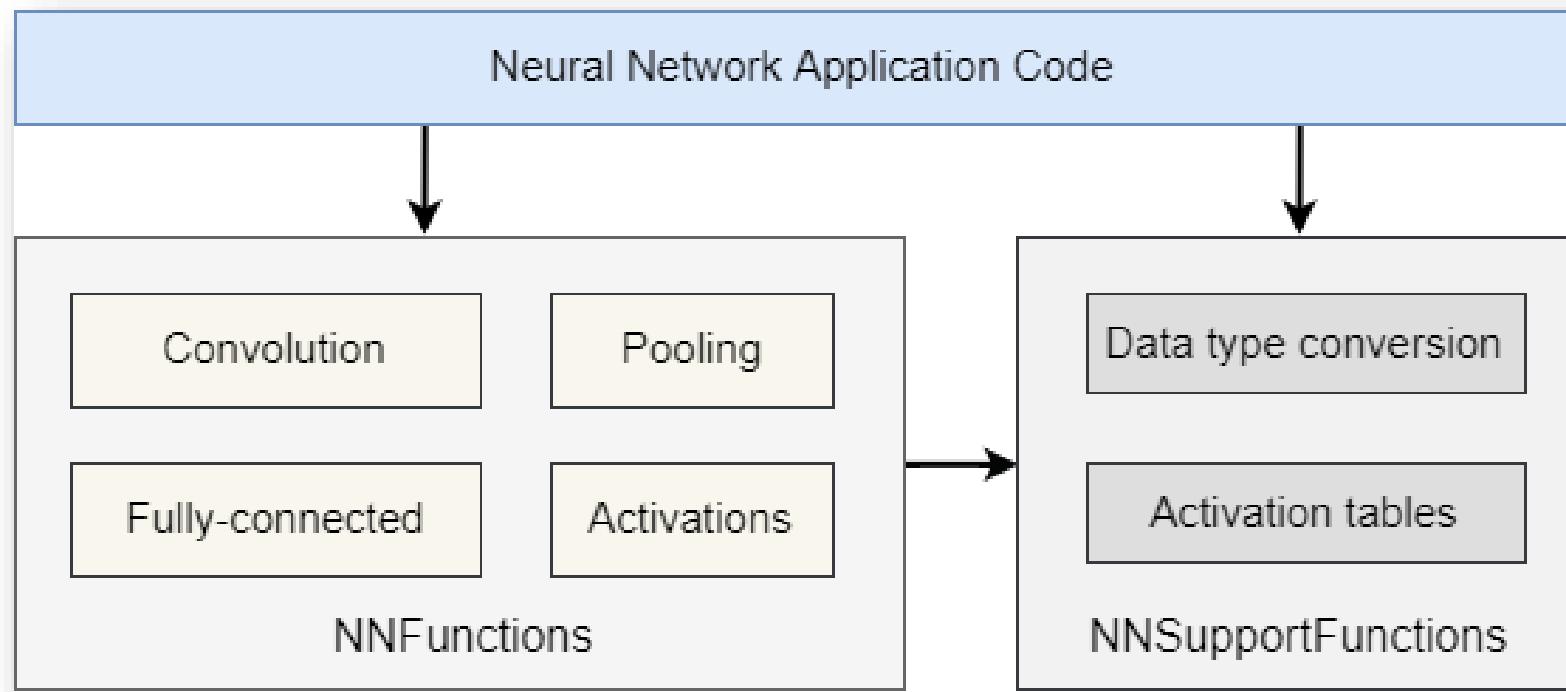


The screenshot shows the CMSIS-NN Software Library website. The header includes the CMSIS logo, the title "CMSIS-NN Version 7.0.0", and a search bar. The left sidebar has a navigation tree with "CMSIS-NN" expanded, showing "CMSIS NN Software Library" which is selected and highlighted with a blue border. Other items in the tree include "Introduction", "Supported Processors", "Access to CMSIS-NN", "Quantization Specification", "Examples", "Pre-processor Macros", "Inclusive Language", "Copyright Notice", "Revision History", "Todo List", "API Reference", and "Data Structures". The main content area is titled "CMSIS NN Software Library" and "Introduction". It describes the library as a collection of efficient neural network kernels developed to maximize performance and minimize memory footprint on Arm Cortex-M processors. It also lists categories of functions: Convolution Functions, Activation Functions, Fully-connected Layer Functions, SVDF Functions, Pooling Functions, Softmax Functions, Elementwise Functions, and LSTM Layer Functions.

資料來源：<https://arm-software.github.io/CMSIS-NN/latest/>

- 卷積函數
- 激勵函數
- 全連結層函數
- SVDF函數
- 池化函數
- Softmax函數
- 元素級別函式
- LSTM函數

CMSIS-6 NN 架構圖



CMSIS-NN 可支援

- 沒有SIMD指令的處理器，如 Cortex-M0/0+
- 帶有DSP擴展指令的處理器，如 Cortex-M4/M7
- 帶有MVE指令的處理器，如 Cortex-M55/M85
- 不直接支援 MicroNPU Ehtos-U55/U65/U85
- 不使用 CMSIS-NN 也可以使用傳統指令集執行相關計算，只是速度較慢。

資料來源：<https://arm-software.github.io/CMSIS-NN/latest/>

CMSIS-6 NN – 卷積函式

arm_convolve_XXX

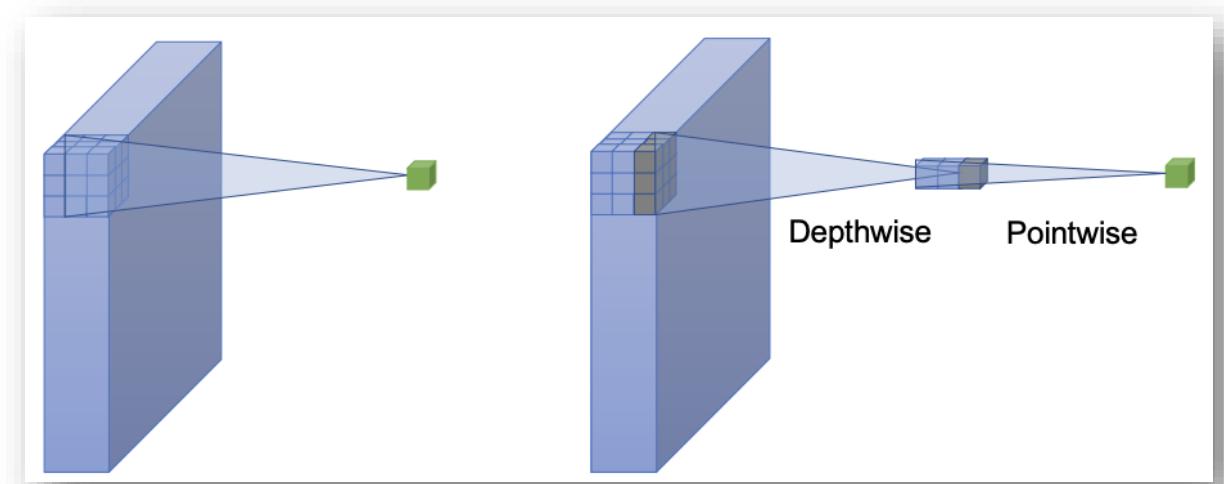
- **1_x_n_s4, 1_x_n_s8, 1x1_s4, 1x1_s4_fast, 1x1_s8, 1x1_s8_fast, even_s4, s16, s4, s8, wrapper_s16, wrapper_s4, wrapper_s8**

arm_depthwise_conv_XXX

- **3x3_s8, fast_s16, s16, s4, s4_opt, s8, s8_opt, wrapper_s16, wrapper_s4, wrapper_s8**

arm_transpose_conv_XXX

- **s8, wrapper_s8**



資料來源：<https://arm-software.github.io/CMSIS-NN/latest/>

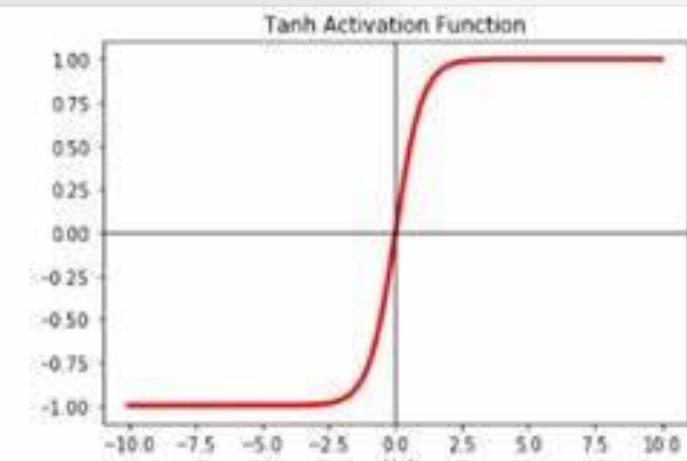
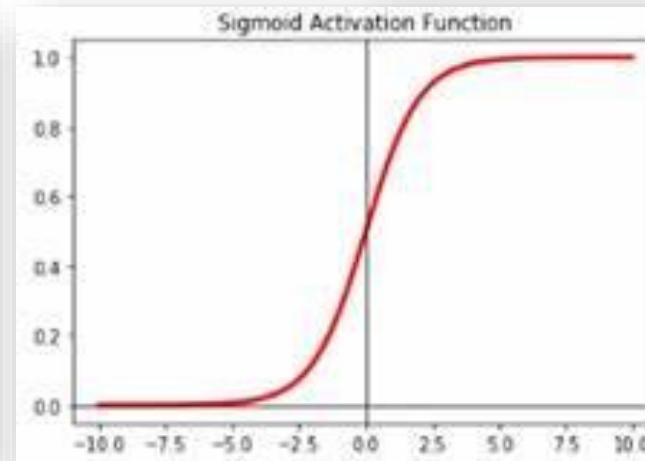
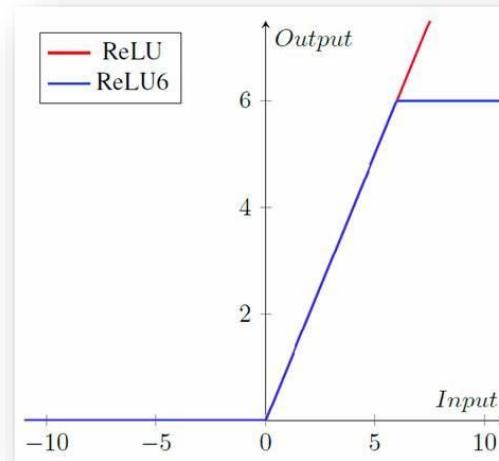
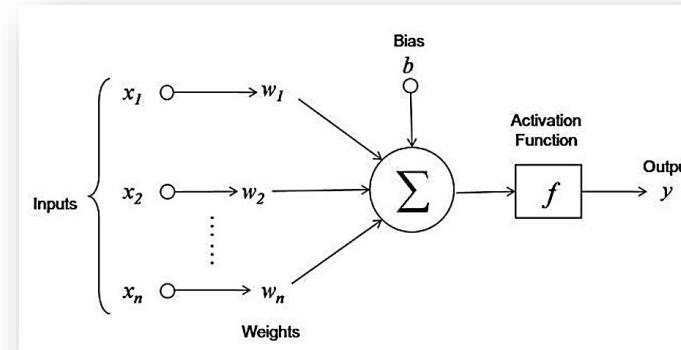
CMSIS-6 NN – 激勵函式

arm_nn_activation_s16 (ARM_SIGMOID, ARM_TANH)

arm_relu6_s8

arm_relu_q15

arm_relu_q7



資料來源：<https://arm-software.github.io/CMSIS-NN/latest/>

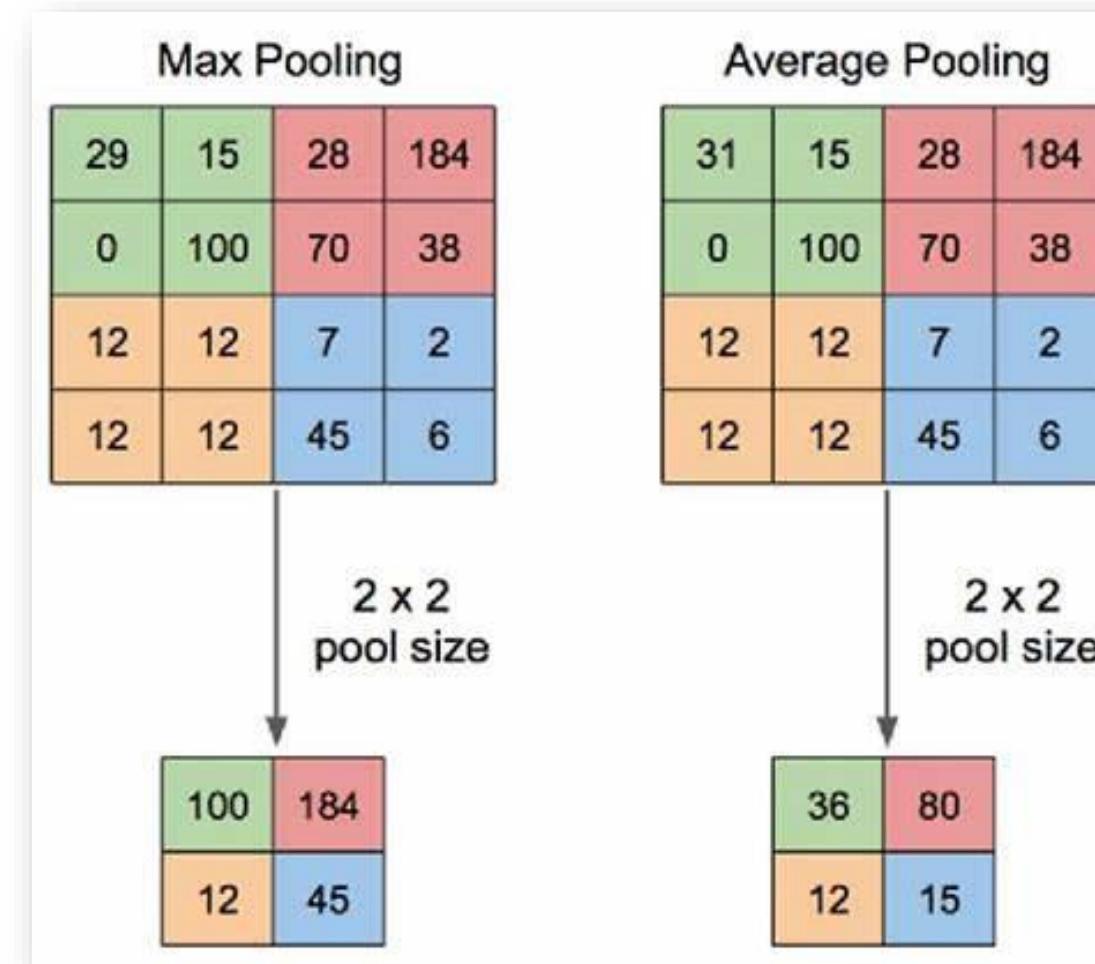
CMSIS-6 NN – 池化函式

arm_avgpool_s16

arm_avgpool_s8

arm_max_pool_s16

arm_max_pool_s8



資料來源：<https://arm-software.github.io/CMSIS-NN/latest/>

CMSIS-6 NN – 全連結層函式

arm_batch_matmul_s16

arm_batch_matmul_s8

arm_fully_connected_per_channel_s8

arm_fully_connected_s16

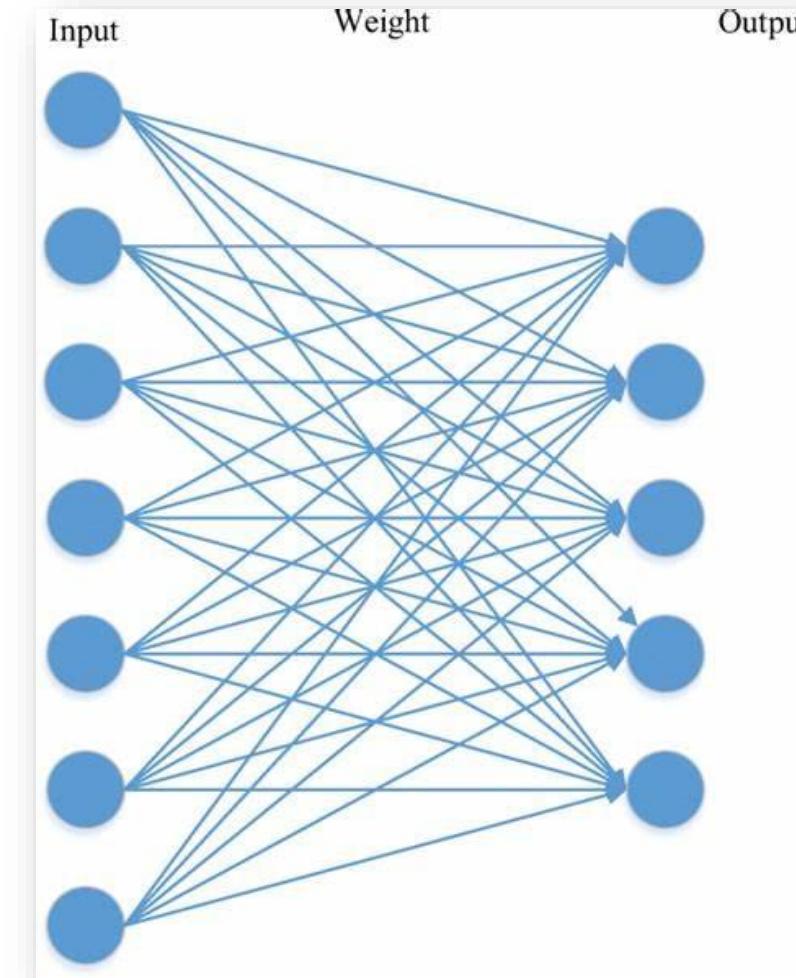
arm_fully_connected_s4

arm_fully_connected_s8

arm_fully_connected_wrapper_s8

arm_vector_sum_s8

arm_vector_sum_s8_s64



資料來源：<https://arm-software.github.io/CMSIS-NN/latest/>

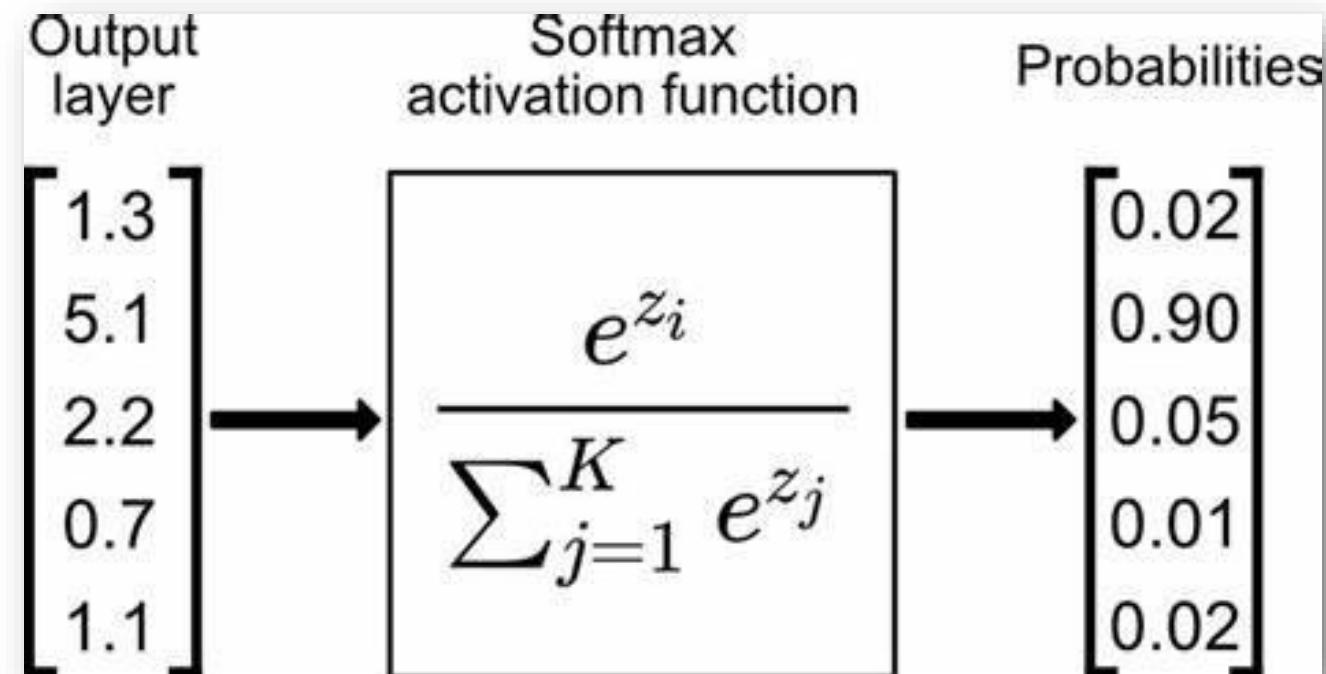
CMSIS-6 NN – Softmax函式

arm_softmax_s16

arm_softmax_s8

arm_softmax_s8_s16

arm_softmax_u8



資料來源：<https://arm-software.github.io/CMSIS-NN/latest/>

CMSIS 5 / 6 NN 比較

	CMSIS v5 (NN v3.1.0)		v6 (NN 5.0.0)
Functions			[+] New [-] Remove [*] private
Activation	arm_nn_activations_direct_q15 arm_nn_activations_direct_q7		arm_relu6_s8 [+] ..._activation_s16
Basic math	arm_elementwise_add_s16 arm_elementwise_add_s8		arm_elementwise_mul_s16 arm_elementwise_mul_s8
Concatenation	arm_concatenation_s8_w arm_concatenation_s8_x		arm_concatenation_s8_y arm_concatenation_s8_z
Fully-connected	arm_fully_connected_mat_q7_vec_q15 arm_fully_connected_q15 arm_fully_connected_q7 arm_fully_connected_s16 arm_fully_connected_s8		arm_fully_connected_mat_q7_vec_q15_opt arm_fully_connected_q15_opt arm_fully_connected_q7_opt arm_fully_connected_s16_get_buffer_size arm_fully_connected_s8_get_buffer_size
LSTM Layer	None		[+] arm_lstm_unidirectional_s16_s8
Pooling	arm_avgpool_s16 arm_avgpool_s8 arm_max_pool_s16 arm_maxpool_q7_HWC		arm_avgpool_s16_get_buffer_size arm_avgpool_s8_get_buffer_size arm_max_pool_s8 arm_avepool_q7_HWC
Reshape	arm_reshape_s8		
Softmax	arm_nn_softmax_common_s8 arm_softmax_q15 arm_softmax_q7 arm_softmax_s16		arm_softmax_s8 arm_softmax_s8_s16 arm_softmax_u8 [*] arm_nn_softmax_common_s8
SVDF	arm_svdf_s8		arm_svdf_state_s16_s8
Conversion	arm_q7_to_q15_no_shift arm_q7_to_q15_reordered_no_shift		arm_q7_to_q15_reordered_with_offset arm_q7_to_q15_with_offset
Basic Math	arm_nn_accumulate_q7_to_q15 arm_nn_add_q7 arm_nn_depthwise_conv_nt_t_padded_s8 arm_nn_depthwise_conv_nt_t_s8 arm_nn_mat_mult_core_1x_s8 arm_nn_mat_mult_core_4x_s8		arm_nn_mat_mult_nt_t_s8 arm_nn_mult_q15 arm_nn_mult_q7 arm_nn_vec_mat_mult_t_s16 arm_nn_vec_mat_mult_t_s8 [*] arm_elementwise_mul_s16_s8

OmniXRI 整理製作, 2024/02/16

差異頗大，許多函式被刪除或整併。

資料來源：<https://omnixri.blogspot.com/2024/02/tinyml-arm-cmsis-6-dsp-nn.html>

	CMSIS v5 (NN v3.1.0)	v6 (NN 5.0.0)
Functions		[+] New [-] Remove [*] private
Convolution	arm_convolve_1_x_n_s8 arm_convolve_1_x_n_s8_get_buffer_size arm_convolve_1x1_HWC_q7_fast_nonsquare arm_convolve_1x1_s8_fast arm_convolve_1x1_s8_fast_get_buffer_size arm_convolve_fast_s16 arm_convolve_fast_s16_get_buffer_size arm_convolve_HWC_q15_basic arm_convolve_HWC_q15_fast arm_convolve_HWC_q15_fast_nonsquare arm_convolve_HWC_q7_basic arm_convolve_HWC_q7_basic_nonsquare arm_convolve_HWC_q7_fast arm_convolve_HWC_q7_fast_nonsquare arm_convolve_HWC_q7_RGB arm_convolve_s16 arm_convolve_s16_get_buffer_size arm_convolve_s8 arm_convolve_s8_get_buffer_size arm_convolve_s8_get_buffer_size arm_convolve_wrapper_s16 arm_convolve_wrapper_s16_get_buffer_size arm_convolve_wrapper_s8 arm_convolve_wrapper_s8_get_buffer_size arm_depthwise_conv_3x3_s8 arm_depthwise_conv_fast_s16 arm_depthwise_conv_s16 [*] arm_depthwise_conv_s4 [*] arm_depthwise_conv_s4_opt arm_depthwise_conv_s8 arm_depthwise_conv_s8_opt [*] arm_depthwise_conv_wrapper_s16 [*] arm_depthwise_conv_wrapper_s4 arm_depthwise_conv_wrapper_s8 [*] arm_transpose_conv_s8	arm_convolve_1_x_n_s8 [+] arm_convolve_1x1_s4 [+] arm_convolve_1x1_s4_fast [+] arm_convolve_1x1_s8 arm_convolve_1x1_s8_fast arm_convolve_fast_s16 arm_convolve_s16 [+] arm_convolve_s4 arm_convolve_s8 arm_convolve_wrapper_s16 [+] arm_convolve_wrapper_s4 arm_convolve_wrapper_s8 arm_depthwise_conv_3x3_s8 [+] arm_depthwise_conv_fast_s16 arm_depthwise_conv_s16 [*] arm_depthwise_conv_s4 [*] arm_depthwise_conv_s4_opt arm_depthwise_conv_s8 arm_depthwise_conv_s8_opt [*] arm_depthwise_conv_wrapper_s16 [*] arm_depthwise_conv_wrapper_s4 arm_depthwise_conv_wrapper_s8 [*] arm_transpose_conv_s8
	V5 Basic Math move to V6 Convolution Private [*] arm_nn_depthwise_conv_nt_t_padded_s8 [*] arm_nn_depthwise_conv_nt_t_s8 [*] arm_nn_mat_mult_nt_t_s8 [*] arm_nn_mat_mult_core_1x_s8 [*] arm_nn_mat_mult_core_4x_s8	V5 Basic Math move to V6 Convolution Private [*] arm_nn_depthwise_conv_nt_t_padded_s8 [*] arm_nn_depthwise_conv_nt_t_s8 [*] arm_nn_mat_mult_nt_t_s8 [*] arm_nn_mat_mult_core_1x_s8 [*] arm_nn_mat_mult_core_4x_s8

OmniXRI 整理製作, 2024/02/16

參考文獻

- 許哲豪，臺灣科技大學資訊工程系「人工智慧與邊緣運算實務」（2021~2023）
<https://omnixri.blogspot.com/p/ntust-edge-ai.html>
- 許哲豪，OmniXRI's Edge AI & TinyML 小學堂 Youtube 直播課程總結
<https://omnixri.blogspot.com/2024/06/omnixris-edge-ai-tinyml-youtube.html>
- 許哲豪，歐尼克斯實境互動工作室系列發文—TinyML(MCU AI)系列
<https://hackmd.io/1PK1URhIQ7GutcWgpgsWbg#TinyMLMCU-AI%E7%B3%BB%E5%88%97>
- 許哲豪，TinyML 核心函式庫 Arm CMSIS 6 DSP & NN 更新比較
<https://omnixri.blogspot.com/2024/02/tinyml-arm-cmsis-6-dsp-nn.html>

延伸閱讀

- arm, Common Microcontroller Software Interface Standard (CMSIS)

<https://www.arm.com/technologies/cmsis>

- arm, CMSIS-5 Document (Version 5.9.0)

https://arm-software.github.io/CMSIS_5/latest/General/html/index.html

- arm, CMSIS-6 Document

https://arm-software.github.io/CMSIS_6/latest/General/index.html

- arm, CMSIS-DSP Document

<https://arm-software.github.io/CMSIS-DSP/latest/>

- arm, CMSIS-NN Document

<https://arm-software.github.io/CMSIS-NN/latest/>

沒有最邊



只有更邊



歡迎加入
邊緣人俱樂部



YOUTUBE 直播 : <https://www.youtube.com/@omnixri1784streams>



歐尼克斯實境互動工作室
(OmniXRI Studio)

許哲豪 (Jack Hsu)

[Facebook : Jack Omnixri](#)

[FB社團 : Edge AI Taiwan 邊緣智能交流區](#)

[電子郵件 : omnixri@gmail.com](#)

[部落格 : https://omnixri.blogspot.tw](#)

[開 源 : https://github.com/OmniXRI](#)