arm

CMSIS Partner Meeting
Embedded World 2019

Welcome to
10 years of CMSIS

Reinhard Keil
Sr. Director Embedded Tools

Joachim Krech
Director of Engineering CMSIS/Tools

Shebu Varghese Kuriakose
Senior Software Technology Manager

Agenda

Welcome to 10 years of CMSIS

- Armv8.1-M enhancements that improve DSP and ML performance
- Secure Debug: Reference implementation with CMSIS-DAP
- CMSIS-Zone system partitioning and TrustZone setup
- PSA (Platform Security Architecture) and Trusted Firmware-M
- How CMSIS and TF-M software packs simplify IoT
- CMSIS roadmap & discussion

IMPORTANT: This presentation will be available here:

https://github.com/arm-software/cmsis_5 **- CMSIS_EW2019.pdf**

CMSIS has 10 years of history – how it began!

Making the News: CMSIS Pres

Industry puts weight Cmsis software standard

Reinhard Keil: "Our goal is to reduce complexity."

Jean Anne Booth: "It is the software that takes the time."

Jim Nicholas: "There is a greater good."

troller software interface standard), and acts as a vendor-independent hardware abstraction layer for the Cortex-M series.

"Embedded developers re-use code heavily," said Reinhard Keil, Arm's director for MCU tools. "But purchased code and code from other sources is not often integrated into the project. That is because there is no standard, so we came up with a standard that solves this."

Cmsis should let silicon vendors and middleware providers create software that can be easily integrated. It should also reduce the learning curve for new microcontroller developers.

Creating software is seen as one of the major costs in the embedded industry. Standardising the software interfaces across all Cortex silicon vendor products has the potential to reduce this cost significantly, especially when creating projects for new devices or migrating

for safety requirements."

Fabless semiconductor
ny Luminary Micr
involved in developing C
"It is the software th
the time," said Luminar
marketing officer Jea
Booth. "We will have fu
support on our Stellari
controllers early next ye

ST Microelectronics,
has standardised on Co
its 32bit microcontroll
also given its backing to

"There is a greater goo
Jim Nicholas, general
of STM's microcontroll
sion. "It serves all our in
we collaborate so our cu
have flexibility. We cann
differences with our con
to undermine our cus
routes to market."

NXP is sampling
LPCAxx family of Cort
ucts and is planning
availability early nex
which is why it ha

THE ARCHITECTURE FOR THE DIGITAL WOR

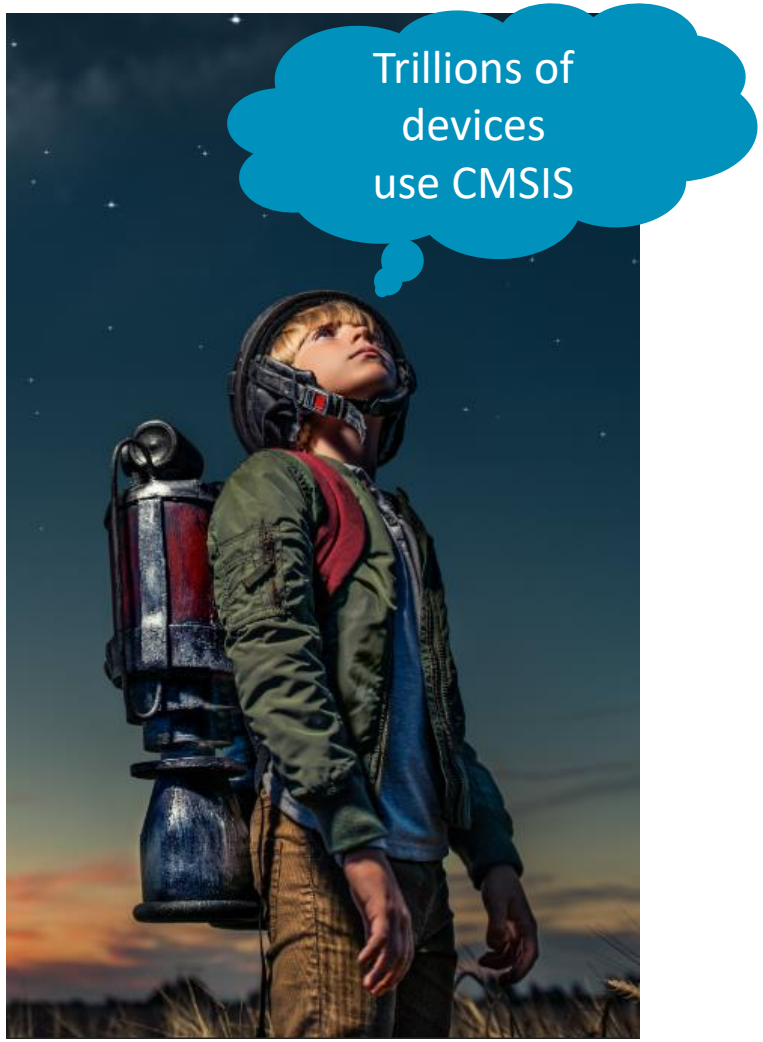
CMSIS – Lead Partners

- Silicon Partners
  - Atmel
  - Luminary
  - NXP
  - STMicroelectronics
- Software Partners
  - IAR Systems
  - KEIL, An ARM Company
  - Micrium
  - SEGGER
- Open Source Community (GCC)

THE ARCHITECTURE FOR THE DIGITAL WORLD®

CMSIS 10 years later – where are we today?

CMSIS is the pathway to the Arm microcontroller eco-system of tools and software

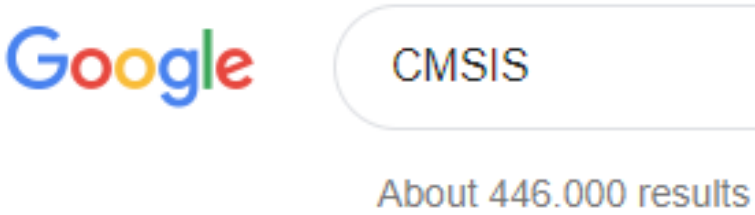


5,500+ MCUs / ASSPs
supported with CMSIS

Used in many projects
> 8,000,000 source files
public on GitHub

CMSIS installations
260.000 downloads
of CMSIS-Pack 5.4.0

- Support for **all Cortex-M, Cortex-A5, A7, A9**
- Open source – development public on GitHub:
  https://github.com/ARM-software/CMSIS_5
  with good contributions – thank you!



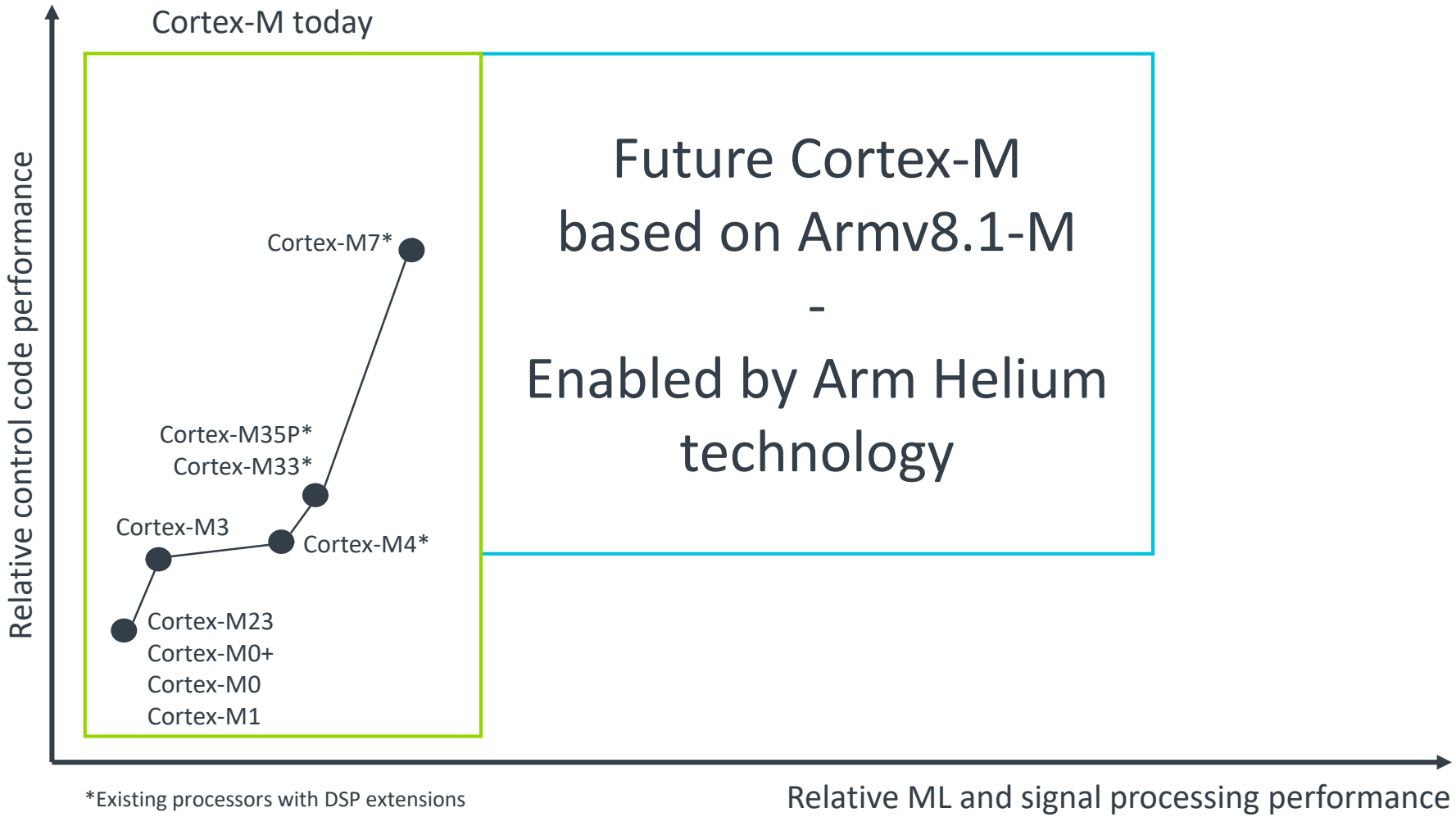
- **32 silicon vendors** that provide public CMSIS device family packs
- CMSIS pack support in **various IDE/toolchains**:
  - Arm DS
  - Arm Keil MDK
  - IAR EWARM
  - github.com/ARM-software/cmsis-pack-eclipse
    (which enables several vendor specific tools)

arm

Armv8.1-M enhancements that improve DSP and ML performance

CMSIS enables consistent software for all Cortex-M (& A5/A7/A9)

+45 billion
Cortex-M
based chips
shipped**



**Based on Arm data

*Existing processors with DSP extensions

Graph not to scale

Evolving the architecture for more capable, secure devices

Armv8.1-M

Helium
M-Profile Vector Extension

- Support for half-precision floating-point, loops and branches
- Additional debug features for signal processing
- Enhanced error reporting using Reliability, Availability and Serviceability (RAS) extension

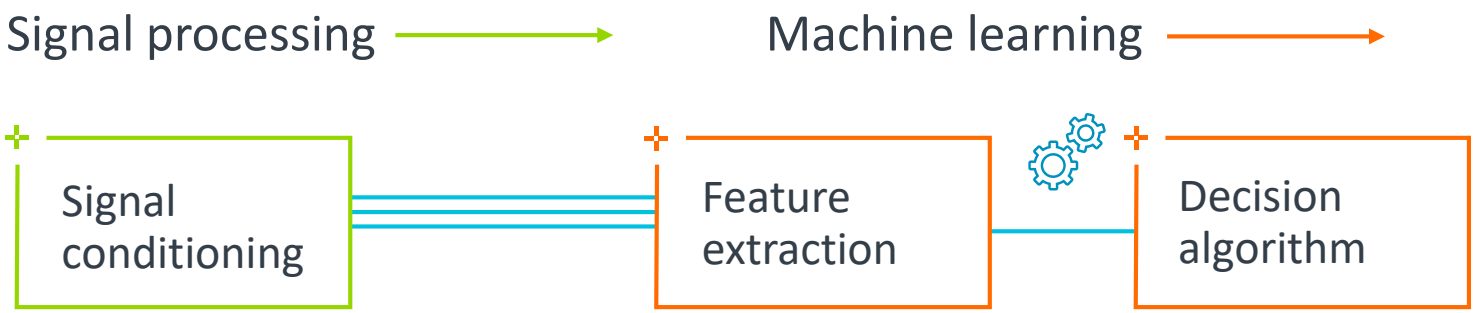
Armv8.0-M mainline
Built-in security with TrustZone for Armv8-M and PSA principles

developer.arm.com/technologies/helium

arm

Transforming the capabilities of the smallest devices

Helium is boosting signal processing and ML performance for millions of developers

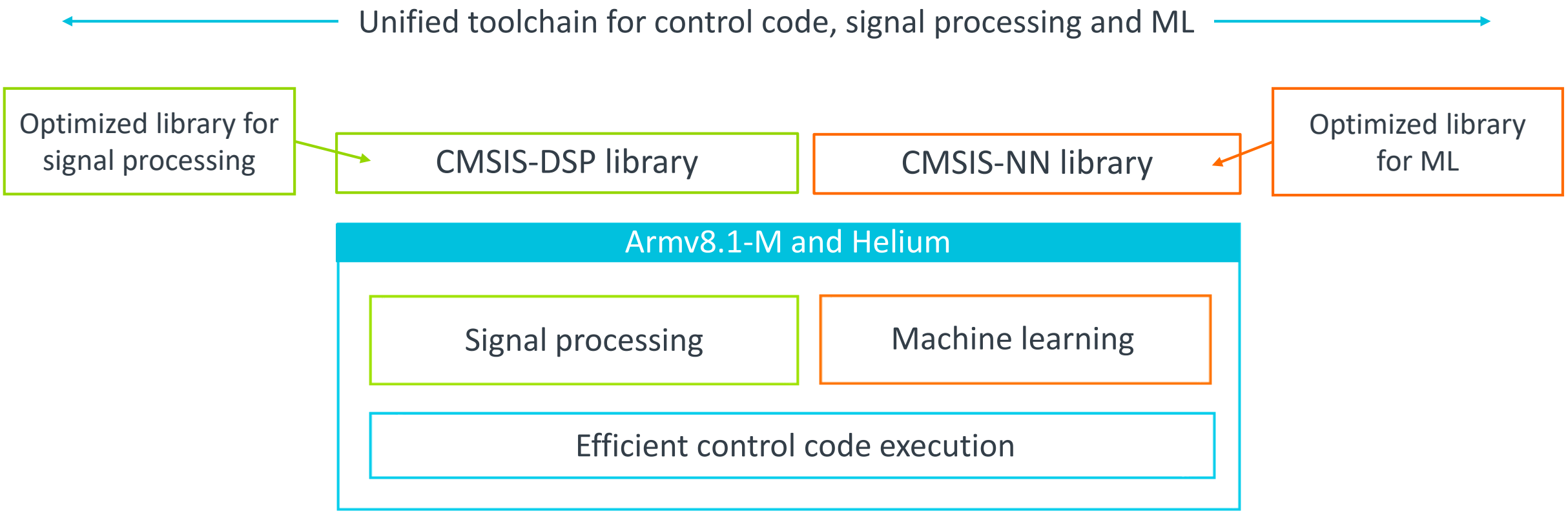


Up to 5x higher signal processing performance*
(CFFT in int32)

Up to 15x higher ML performance*
(matrix multiplication in int8)

*Compared to existing Armv8-M implementation

Simplified software development based on a unified programmer's view



Armv8.1-M Code Example: Vector Addition

Code Snippet

```
void matrix_add_const(ee_u32 N, MATDAT *A, MATDAT
val)
{
   ee_u32 i,j;
   for (i=0; i<N; i++)
   {
      for (j=0; j<N; j++)
      {
         A[i*N+j] += val;
      }
   }
}
```

Assembly Snippet with Helium

```
wlstp.16 lr, r5, exit
```

Low Overhead Loop Instruction variant

Note: This variant is to enable loop tail predication

```
.LBB2_5 :
vldrh.16 q0, [r6]        @ load 8 half words
vadd.i16 q0, q0, r0      @ add val to 8 half words
vstrh.16 q0, [r6], #16   @ store 8 half words
letp lr, .LBB2_5
```

Loop-end with Tail predication

Note: LR stores no. of vector elements processed in this case.

```
exit:
```

Notes: Above branching performance results in significant cycles savings, thereby giving higher performance, smaller code footprint and better power efficiency

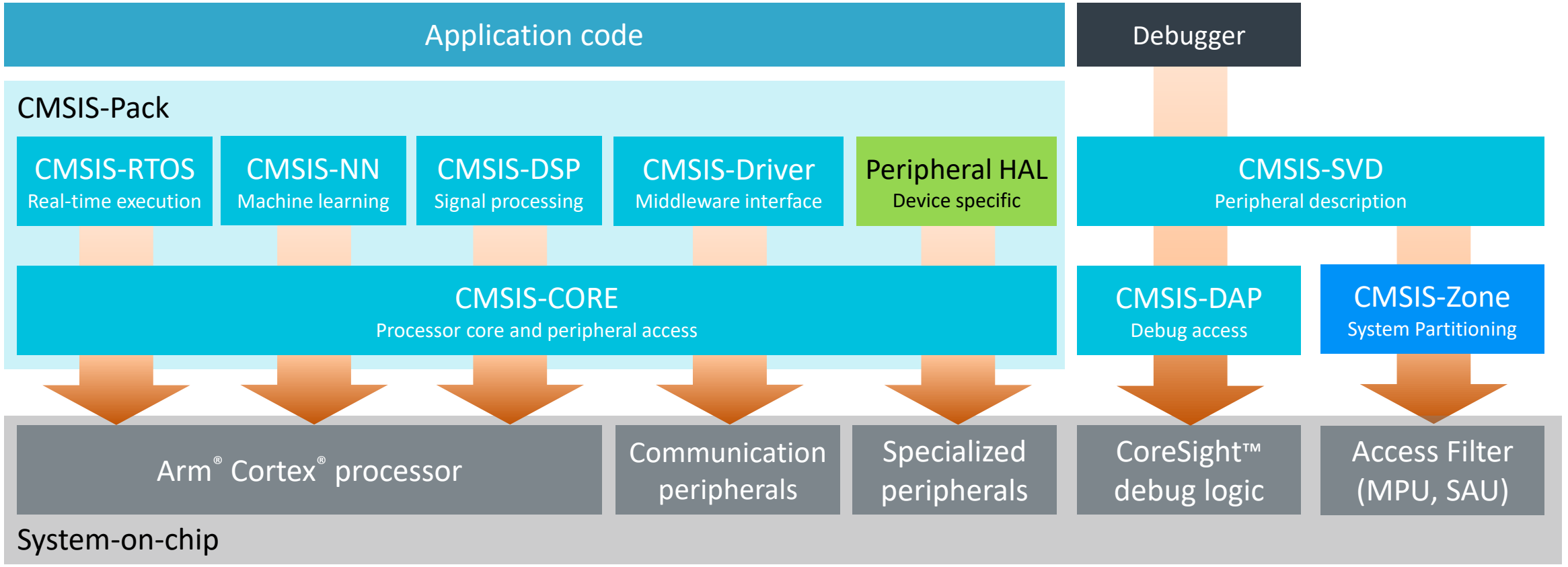
arm

arm

CMSIS Components

CMSIS 5

Consistent software framework for Arm Cortex-M and Cortex-A5/A7/A9 based systems



Complete documentation: http://arm-software.github.io/CMSIS_5/General/html/index.html

CMSIS-Pack – delivery mechanism

Delivery

- Software components, examples, code templates, documentation, device and board support files

Versioning

- Semantic versioning for lifecycle management using an embedded industry standard for reliable production

Dependency

- Specify dependencies upon other packs, software components, toolchains, and APIs

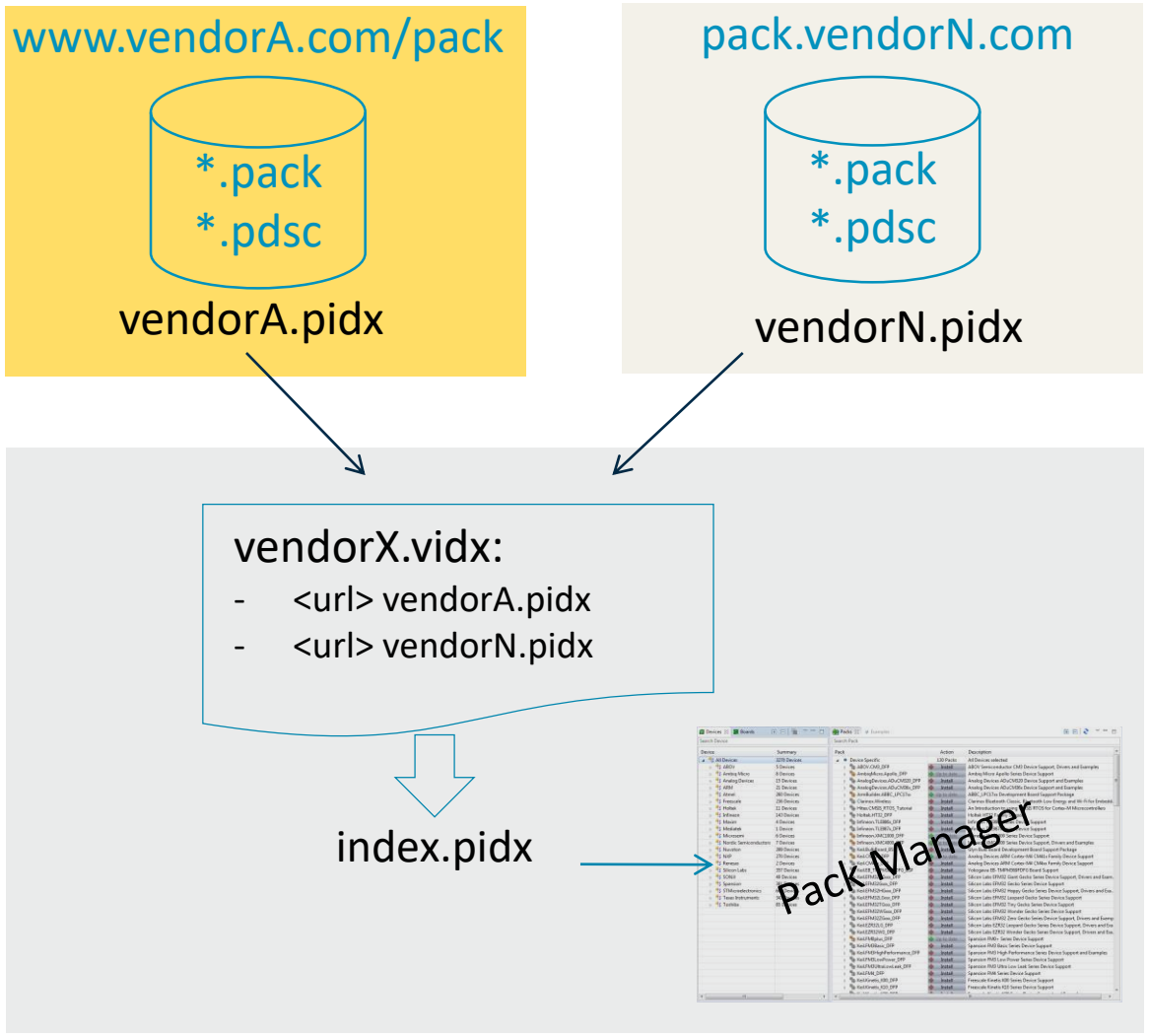
Retargeting

- Automatically replace files based on hardware selection or toolchain requirements

arm

CMSIS-Pack Index Files: Standardization for Download Portals

Multiple download portals (all are equal), with references to other portals



Software Pack vendors publish:

- *Vendor.pidx*: Index file that lists all available packs from “VendorN”

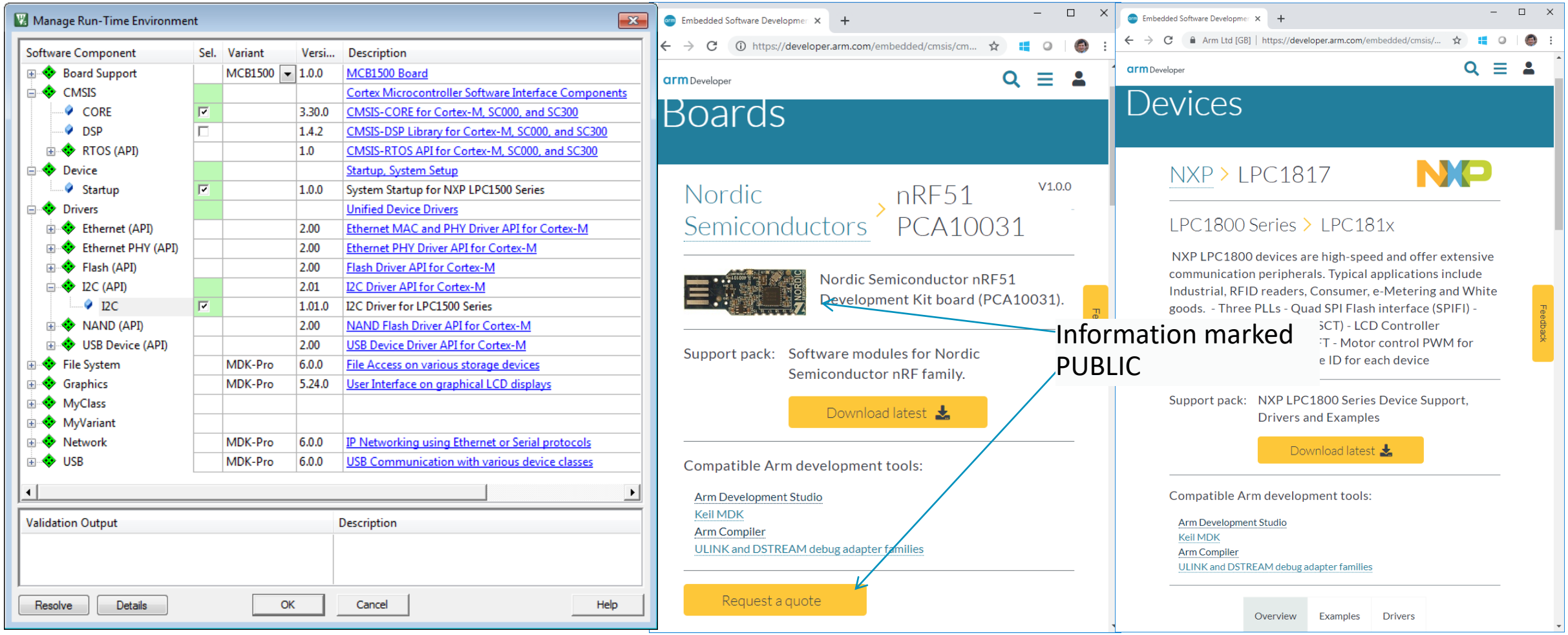
Tool vendors use this index file to list all available packs.

Benefits

- Software Pack vendors can update and add packs. This gets automatically distributed to tools and web.
- Software partners can create packs for software evaluation to promote their products.

CMSIS-Pack is designed for tools and web portals

Information you publish in packs is shown in tools and on web pages



CMSIS-Pack – what's new?

Work with repositories

- *<repository>* element specifies version control location, i.e. **git** or **GitHub**
- IDE's support "local repositories" workflow

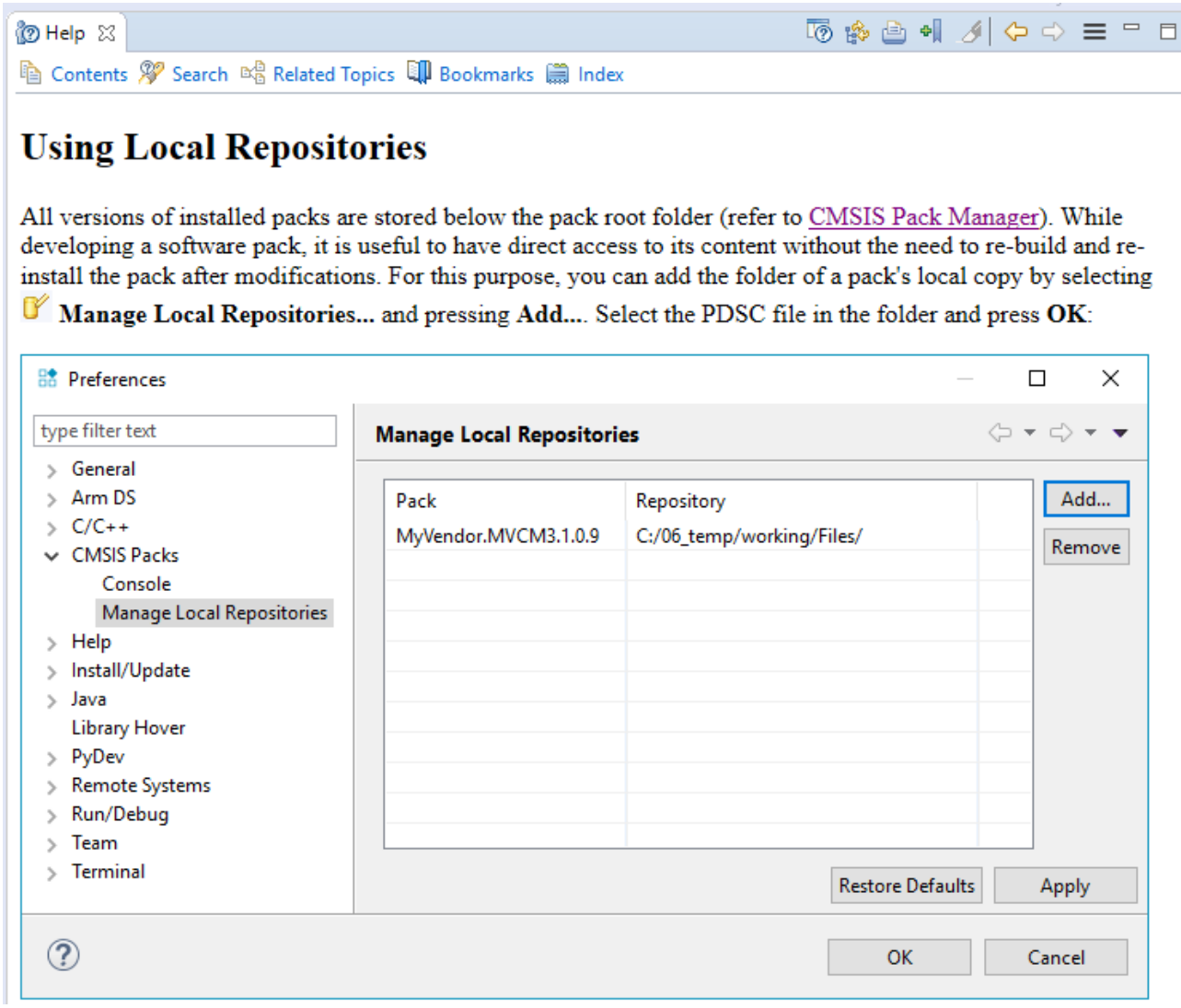
Eclipse CMSIS-Pack v2.4.0 released

- https://github.com/arm-software/cmsis-pack-eclipse
- Many enhancements, for example: Headless build:

```
eclipsec.exe -nosplash -application
com.arm.cmsis.pack.project.CmsisHeadlessBuilder -help
```

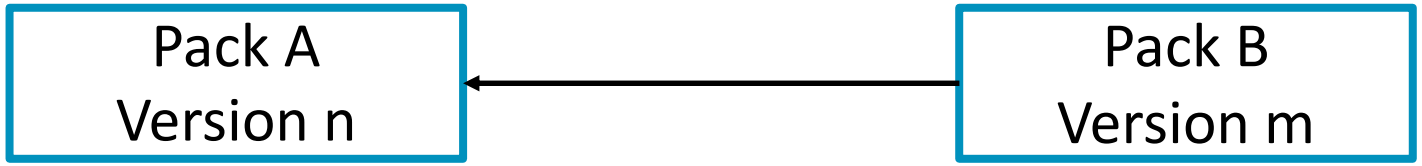
Flash programming via DAP interface

- *<flashinfo>* describes sequence-based flash download – required for devices that cannot execute algorithm from RAM.

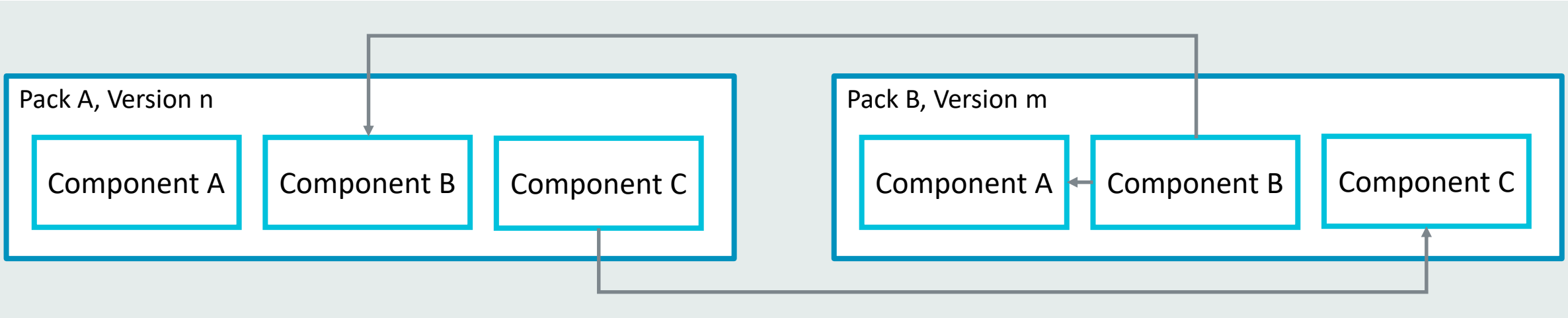


Relationships of packs and software components

- **Packs** can require other packs to be installed:

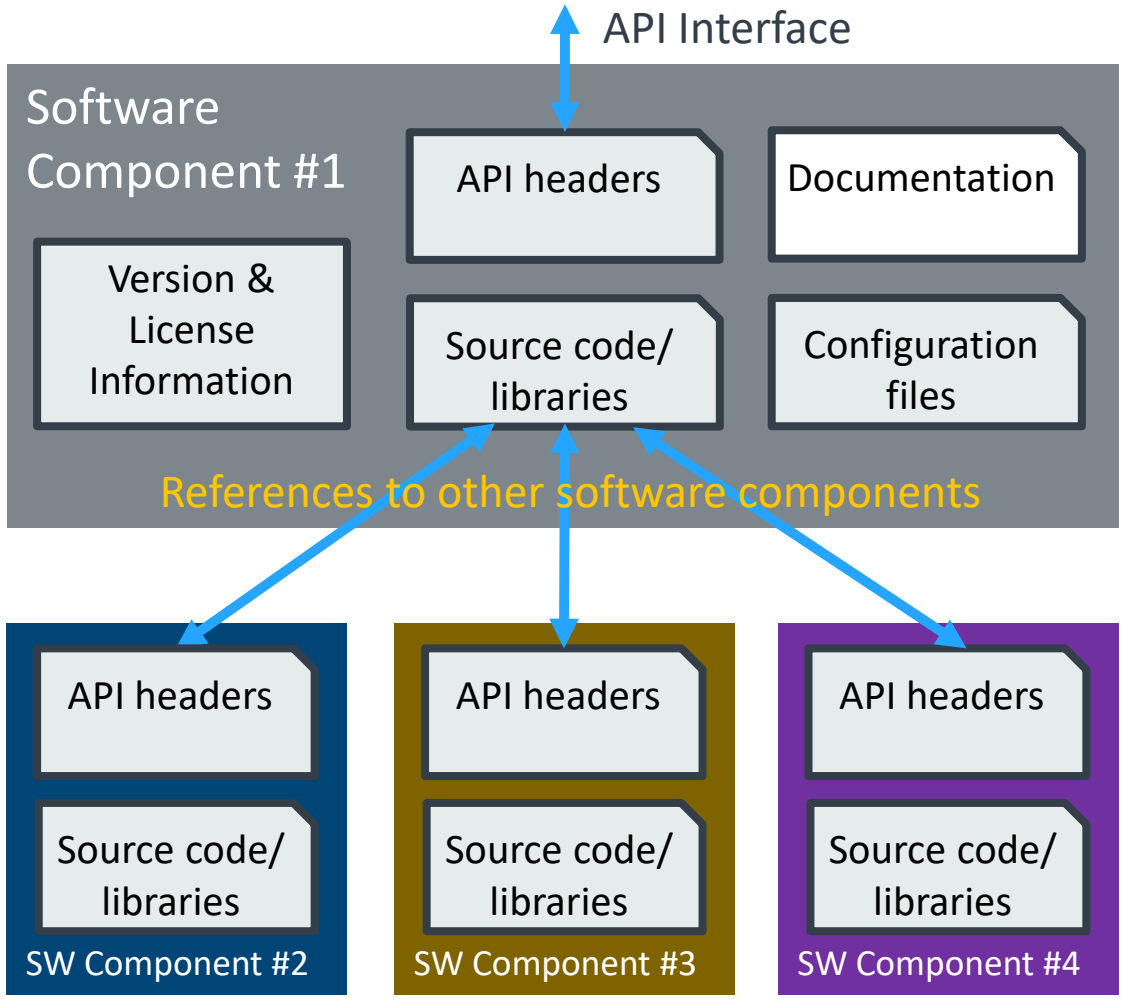


- **Components** can have dependencies on other components; either from the same or from other packs:



arm

CMSIS-Pack: What is a software component



Software components should have:

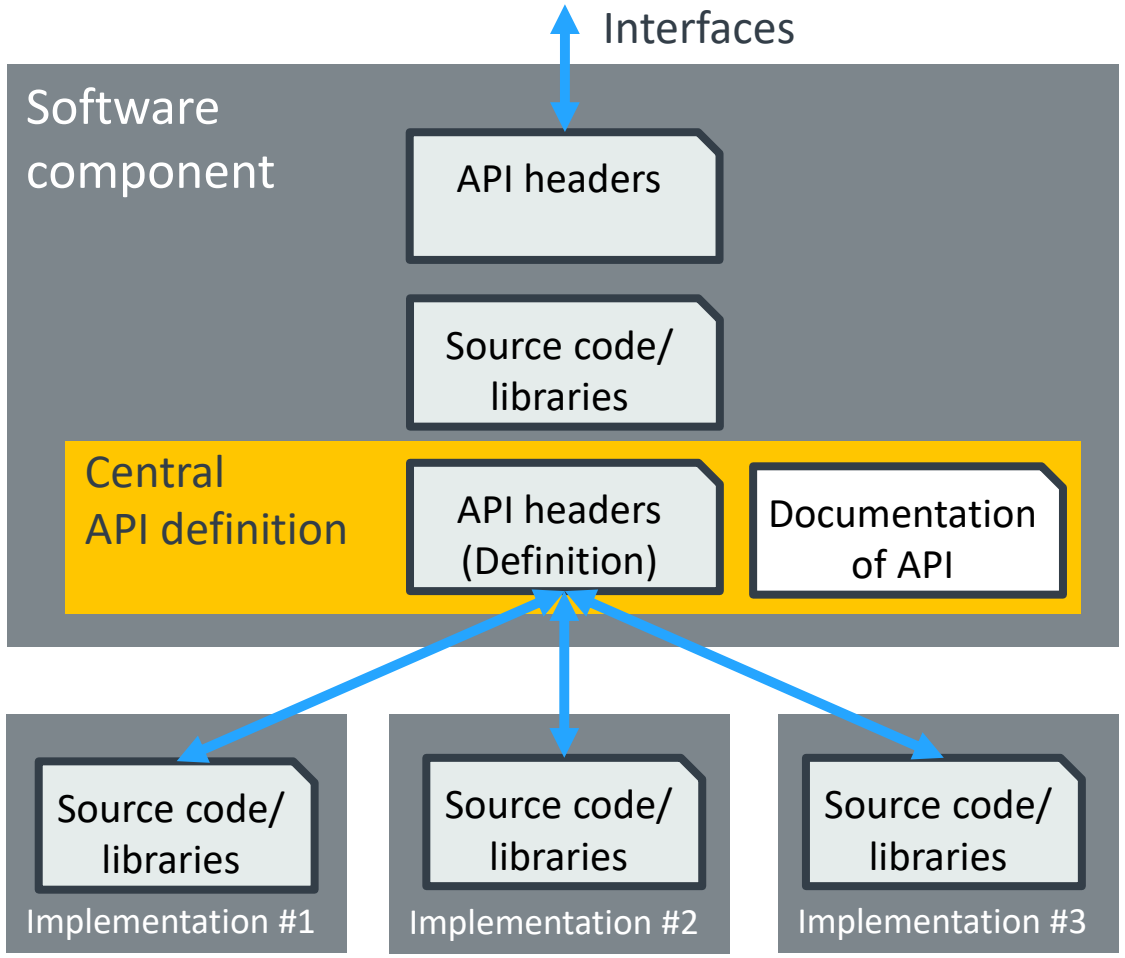
- Version and history information
- License information
- API interface definition
- Documentation
- Source files
- Configuration files (optional)
- Requirements to other components (optional)

CMSIS-Pack defines an XML format that frames all this information and can be used by project management utilities from various tools.

arm

CMSIS-Pack: Central API Interface definition

Ensuring consistent interfaces across standard components



A common problem: API headers evolve over time.

A central API definition shares header file and documentation of an API interface across multiple other software components to ensure consistency.

The API interface is distributed separate or as part of the software component that defines this interface. The API header file is therefore consistent.

An example is the CMSIS-Driver pack that contains various Ethernet and Flash drivers – all compatible with the CMSIS-Driver APIs that are published in the CMSIS Pack.

arm

CMSIS-Driver Version 2.6.0

Peripheral Interface for Middleware and Application Code

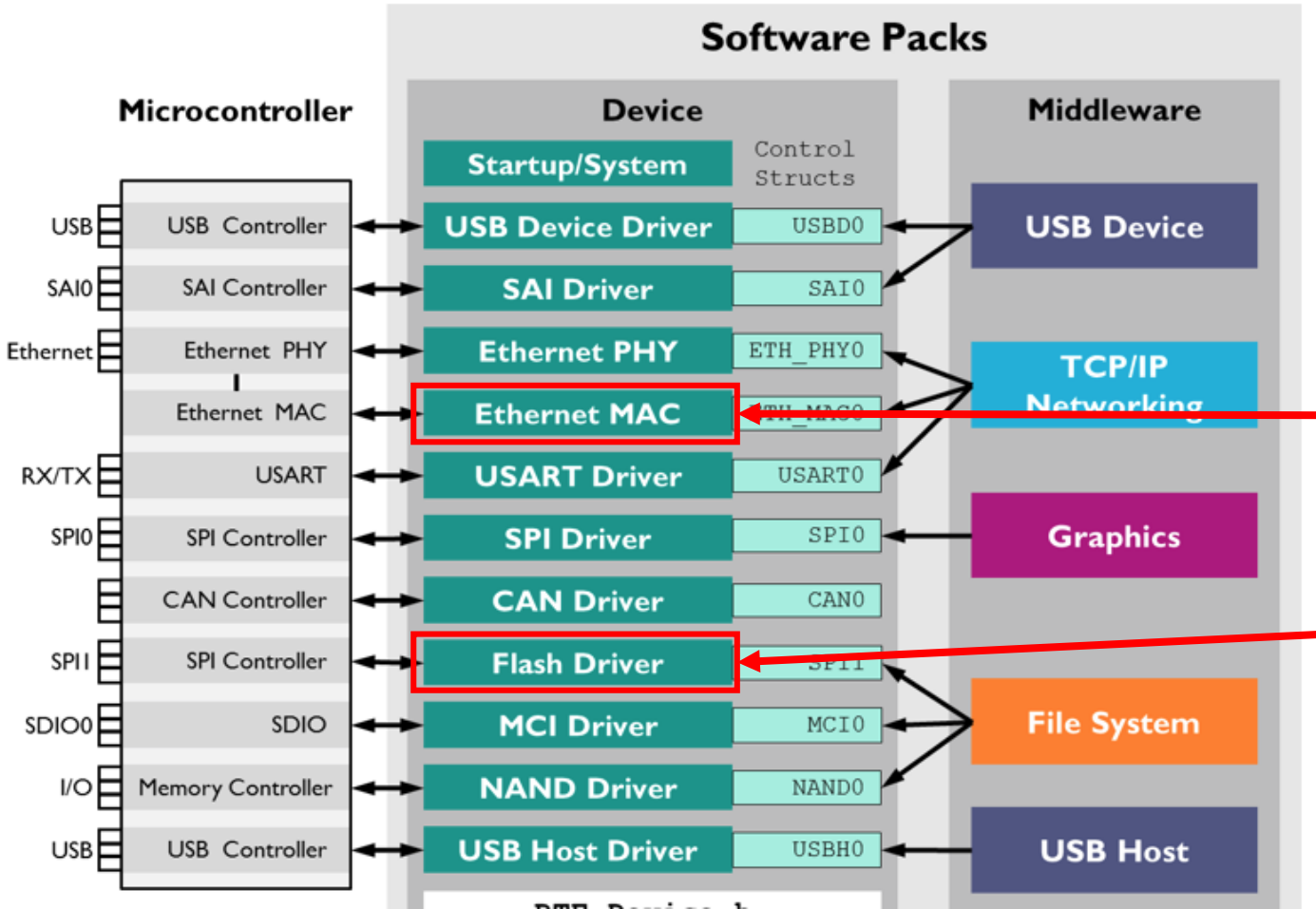
General | CMSIS-Core(A) | CMSIS-Core(M) | Driver | DSP | NN | RTOS v1 | RTOS v2 | Pack | SVD | DAP | Zone

Main Page | Usage and Description | Reference | Search

- CMSIS-Driver
  - Overview
  - Revision History of CMSIS-Driver
  - Theory of Operation
  - Reference Implementation
  - Driver Validation
  - Reference
  - Data Structures
  - Data Structure Index
  - Data Fields

Overview

The CMSIS-Driver specification is a software API that describes peripheral driver interfaces for middleware stacks and user applications. The CMSIS-Driver API is designed to be generic and independent of a specific RTOS making it reusable across a wide range of supported microcontroller devices. The CMSIS-Driver API covers a wide range of use cases for the supported peripheral types, but can not take every potential use-case into account. Over time, it is indented to extend the CMSIS-Driver API with further groups to cover new use-cases.



[CMSIS-Driver Templates]

GitHub projects:
CMSIS-Driver Implementations for [NPX LPC Series]

[CMSIS-Driver Validation]

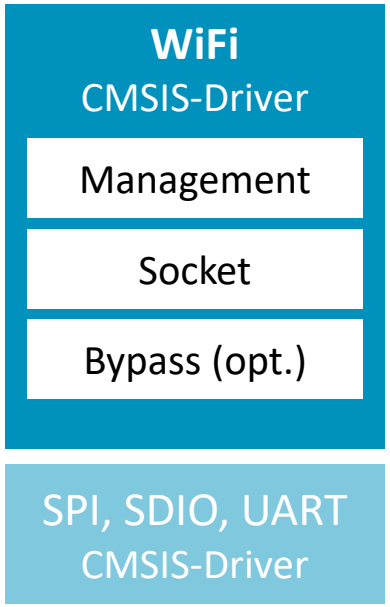
CMSIS-Driver Pack
Generic Drivers that are device independent are now available as separate software pack.

[github.com/ARM-software/CMSIS-Driver]

arm

CMSIS-Driver: WiFi Interface (beta available)

Standard interface to WiFi chipset

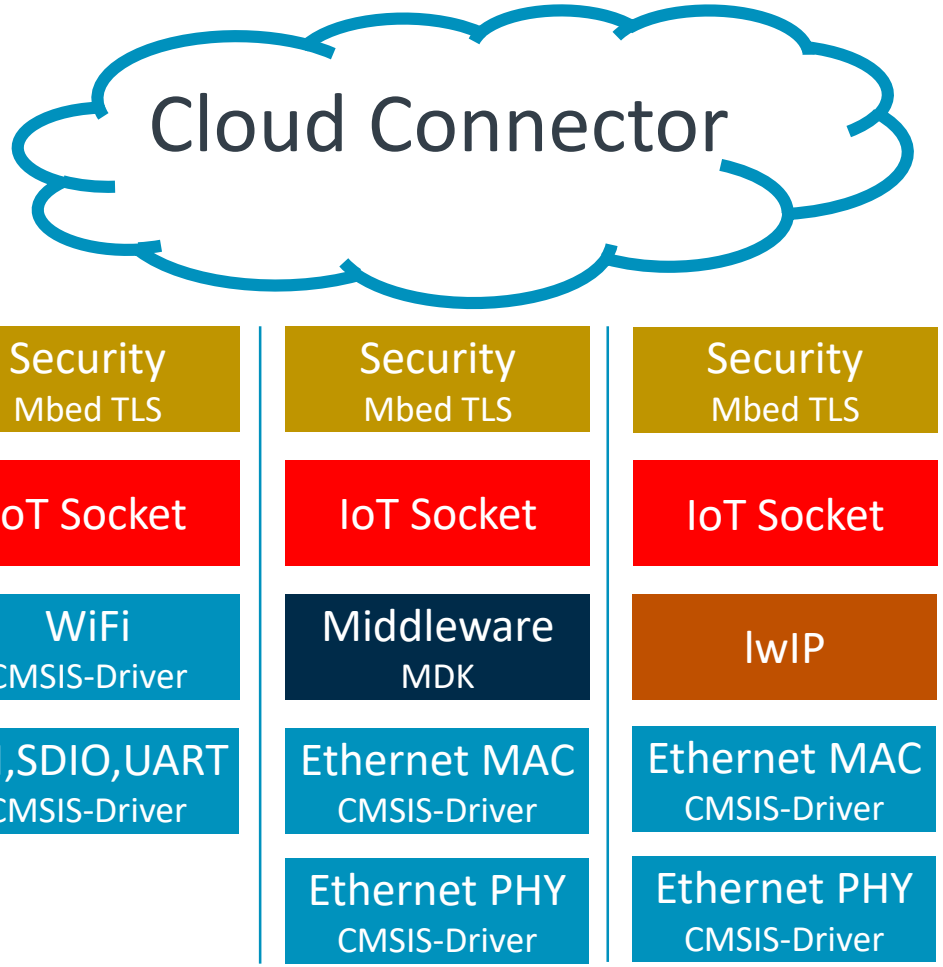


- WiFi CMSIS-Driver that provides access to:
  - Management interface: connection to an access point (AP)
  - Socket interface: IP stack running on WiFi module handles data communication
  - Bypass interface: IP stack is running on MCU; Ethernet frames transferred by WiFi module
- WiFi CMSIS-Driver is typically implemented as:
  - Wrapper for SDK for WiFi module
  - May use CMSIS-Driver compliant implementation for SPI, SDIO or UART connection
- Beta implementation available for:
  - ISM43362 – **Inventek WiFi ISM43362** driver using CMSIS-Driver SPI interface
  - QCA400x – **Qualcomm WiFi QCA400x** driver using CMSIS-Driver SPI interface
  - WiP – release April 2019: **Redpine Signals RS14100** on chip WiFi
  - WiP – release April 2019: **Espressif ESP8266 WiFi Arduino shield**
- CMSIS-Driver template will be available to add custom WiFi chipsets

arm

WiFi Driver and IoT Socket component combined

Generic communication foundation for Cloud connectors on Cortex-M



Azure

IBM

Google Cloud Platform

aws

Security is provided by Mbed TLS

IoT Socket can interface with:

- WiFi CMSIS-Driver to connect to various wireless chipsets
- MDK-Middleware network stack
- LwIP (optional, WiP – contributions welcome)

Implementations available via www.keil.com/IoT (update for WiFi planned in April 2019)

arm

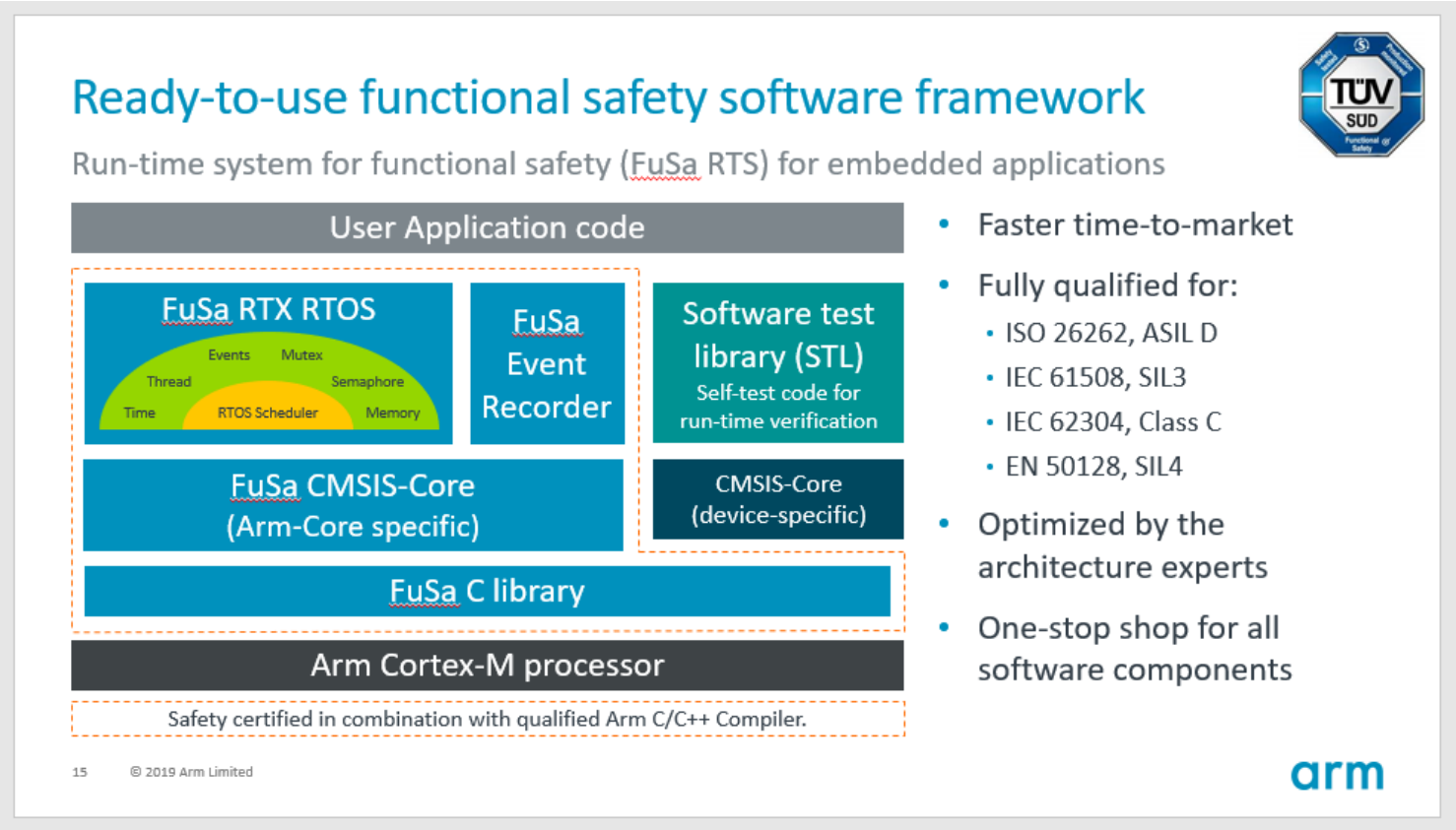
Other CMSIS improvements

CMSIS-RTOS2 implemented by:

- FreeRTOS
- RTX5
- Zephyr

FreeRTOS and RTX5 available for:

- Arm Compiler 5 / 6
- GCC Compiler
- IAR Compiler

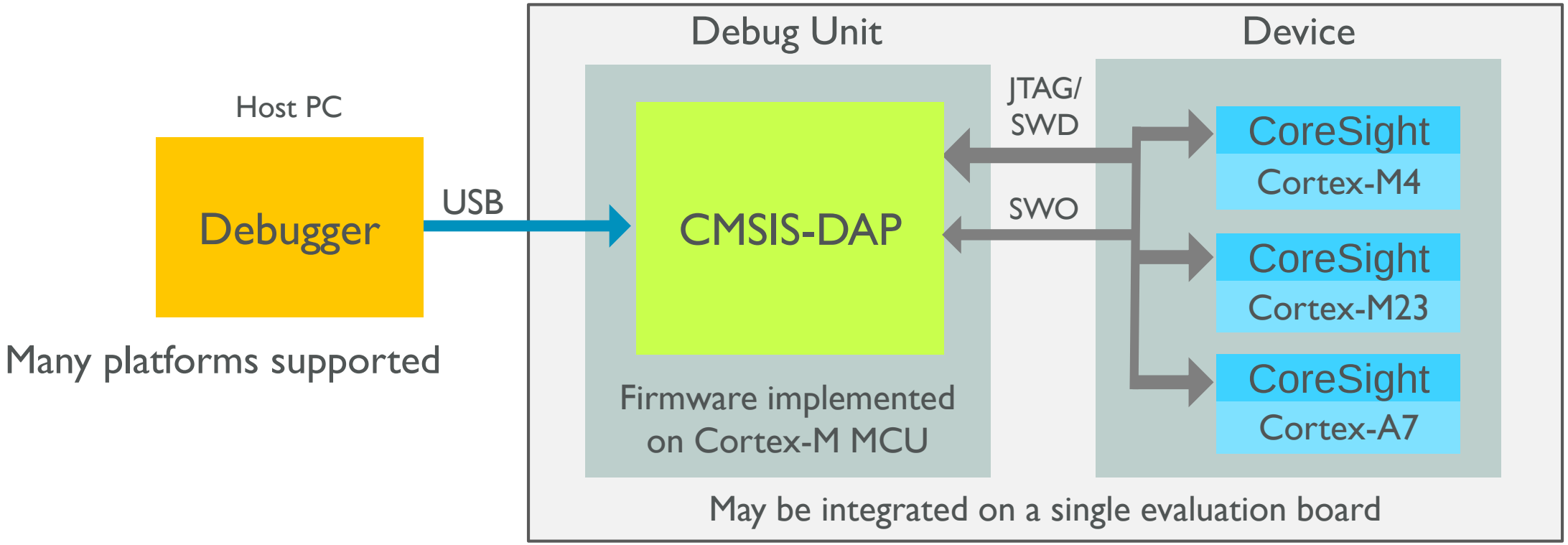


arm

Secure Debug

Reference implementation with CMSIS-DAP

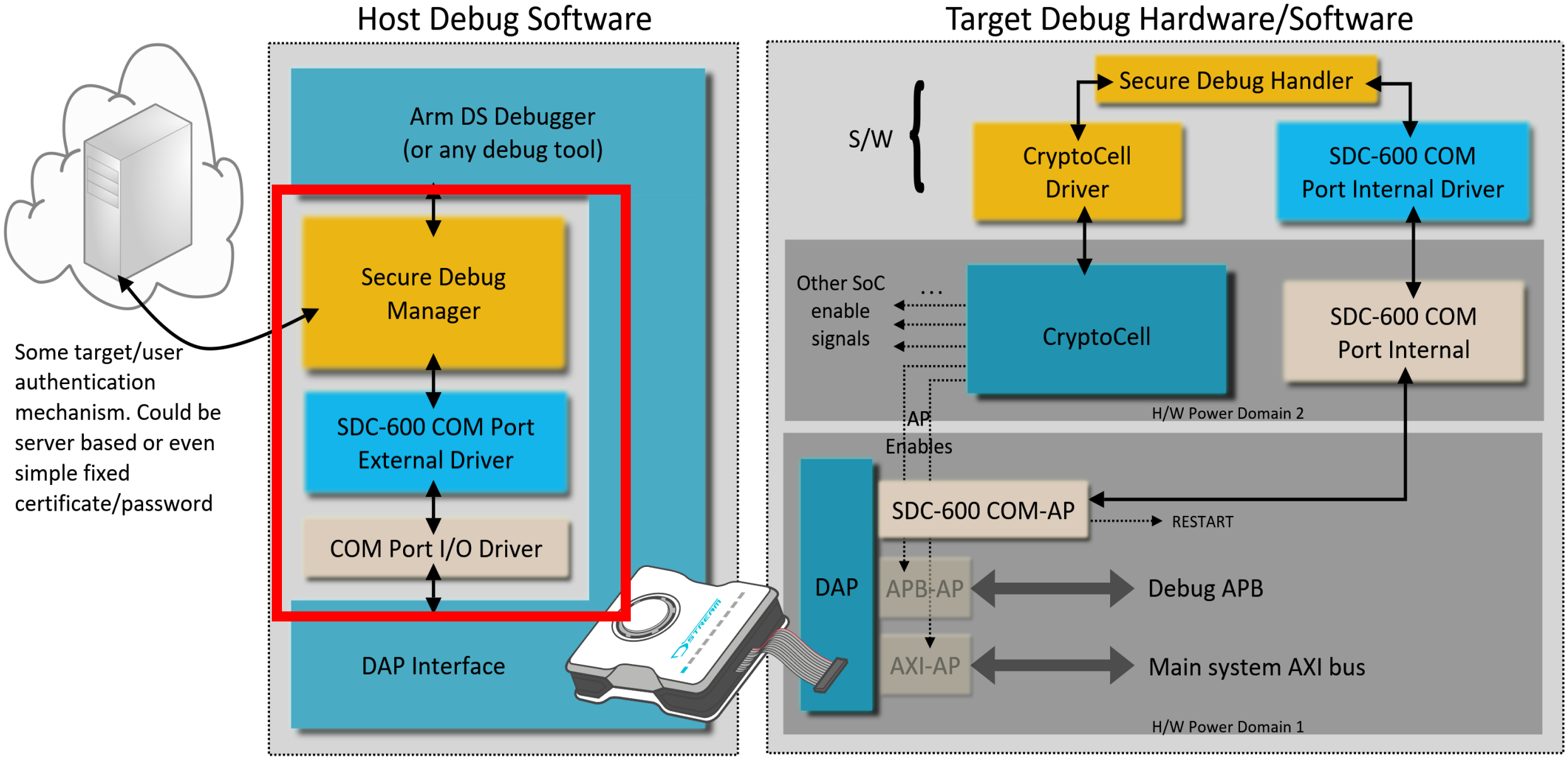
CMSIS-DAP: v1.2.0 (USB HID) + v2.0.0 (WinUSB)

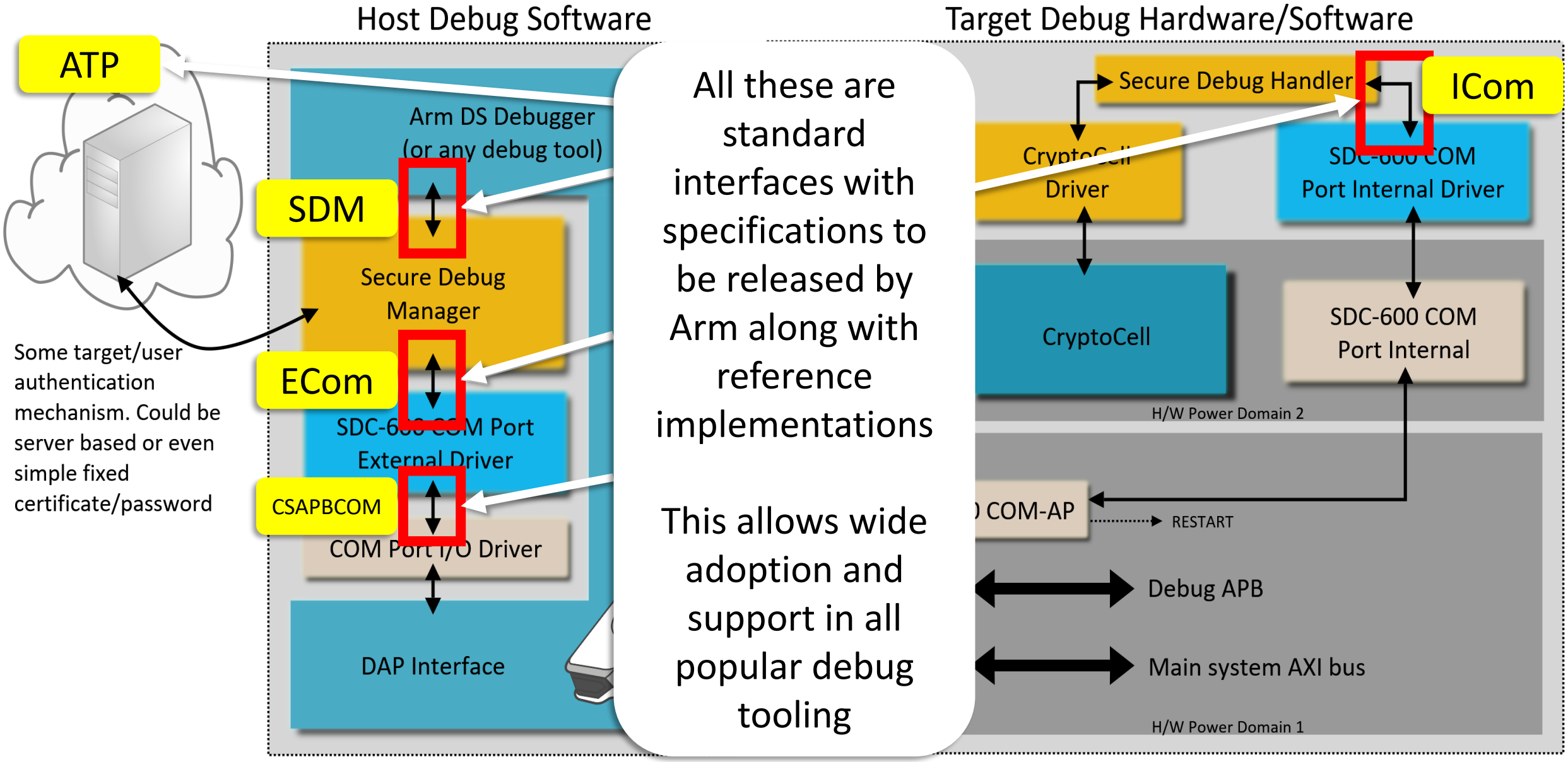


Many platforms supported

- **CMSIS v1.2.0**: continues to support USB HID as interface
- **CMSIS v2.0.0**: introduces USB WIN support with >5x better performance
  - SWO streaming via separate pipe allows significant better trace bandwidth
  - Windows 10 does not require USB driver installation

Now working towards secure debug, learn more in this video: http://bit.ly/2TRO6J8





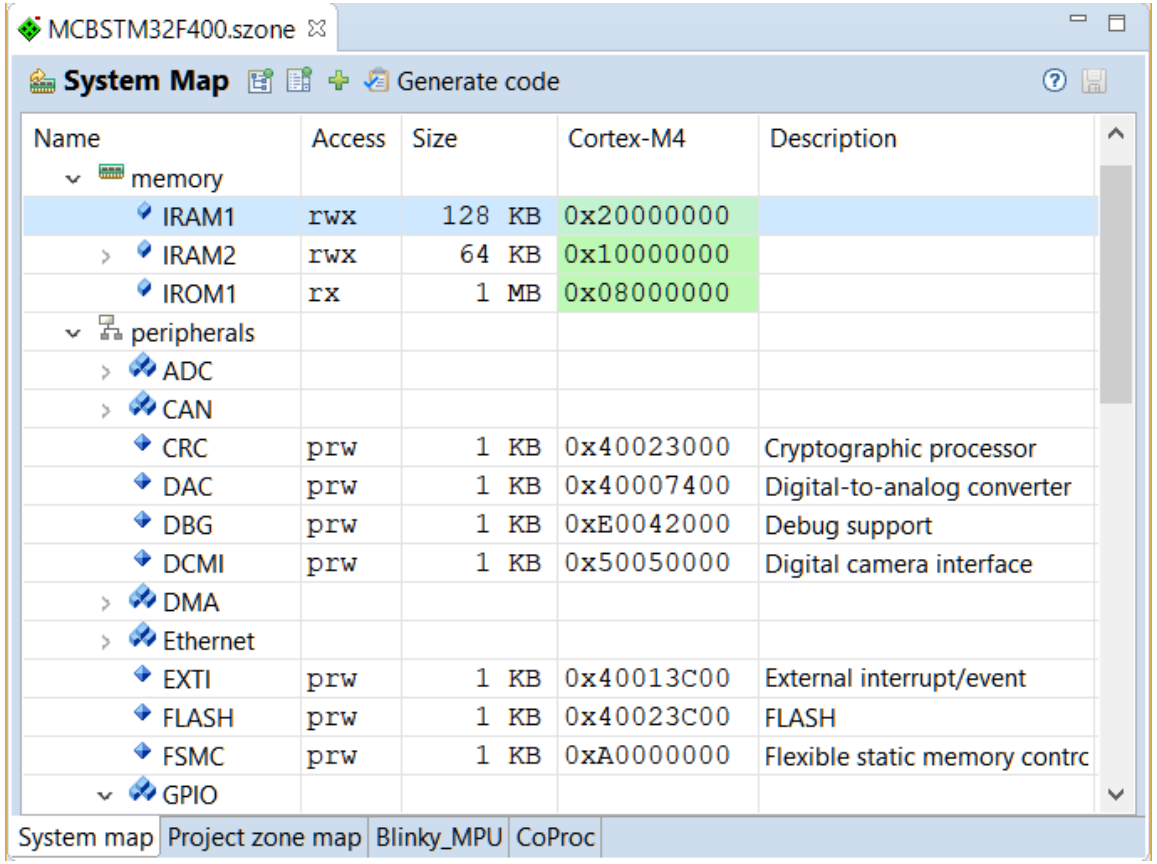
arm

CMSIS-Zone system partitioning and TrustZone setup

CMSIS-Zone – resource management for SoC systems

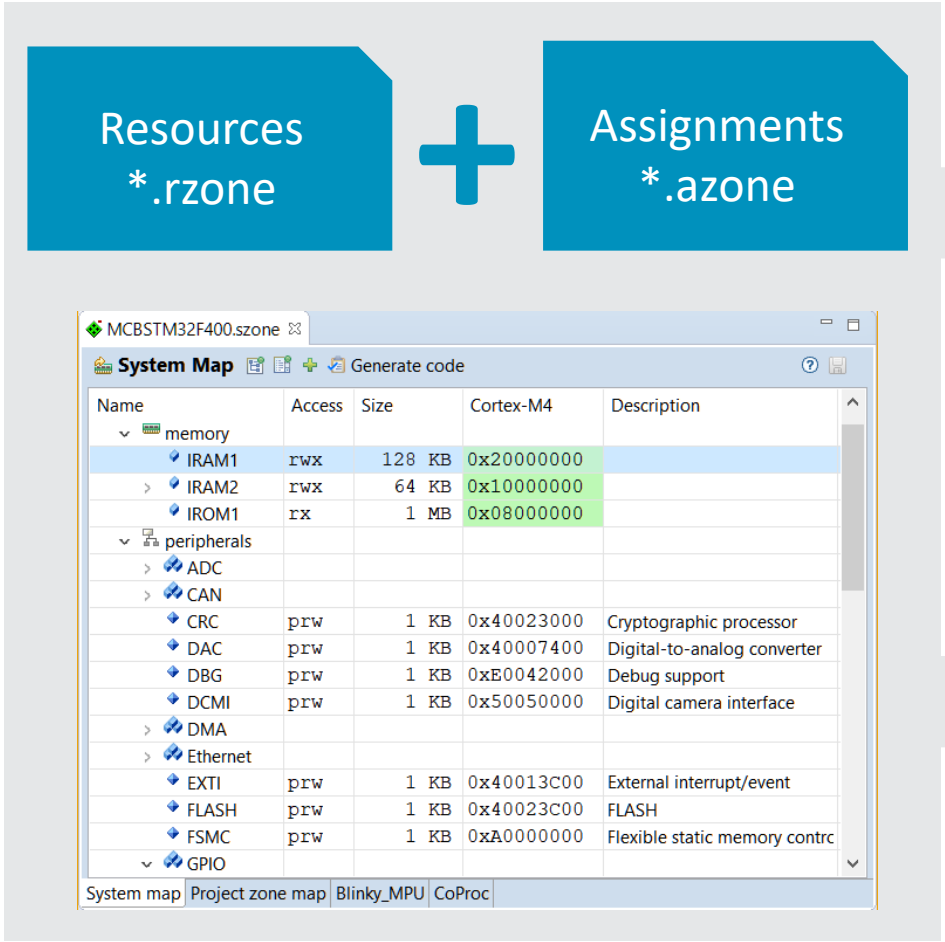
Supports partitioning of multi-processor systems; TrustZone and MPU configuration

System Resources	List available system resources in a SoC system
System Partitioning	Select resources for sub-partitions i.e. independent software projects
TrustZone & MPU Setup	Partition memory & peripherals for safe process execution
Build	Generate hardware configuration and tool setup



CMSIS-Zone – Development workflow

Configuration and build management for system resources



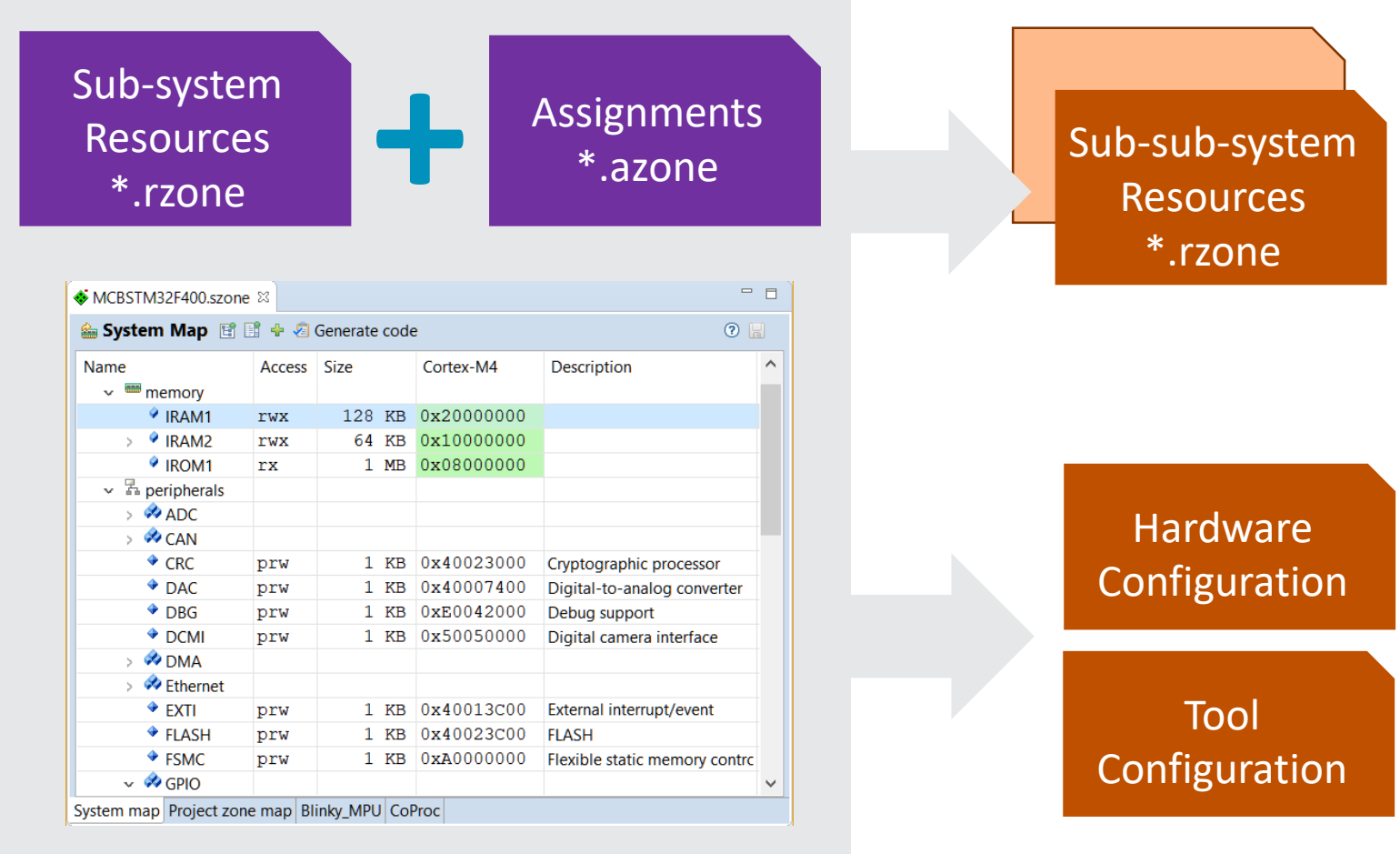
Resource *.rzone file lists all systems resources. Import from CMSIS-Pack/SVD possible

Assignment *.azone file contains partitioning information and is managed by CMSIS-Zone tool

CMSIS-Zone tool can generate sub-system resource files

CMSIS-Zone – Development Workflow

Multi-step approach shows only relevant sub-system



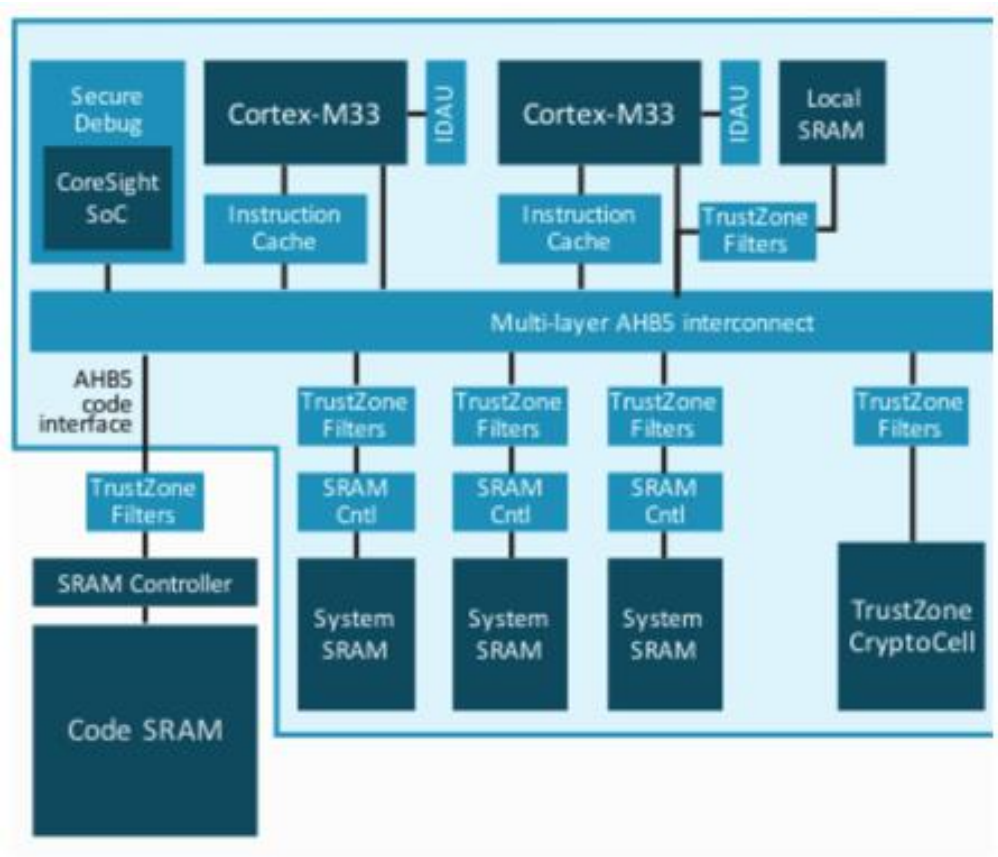
It is possible to break down complexity of a system in multiple steps.

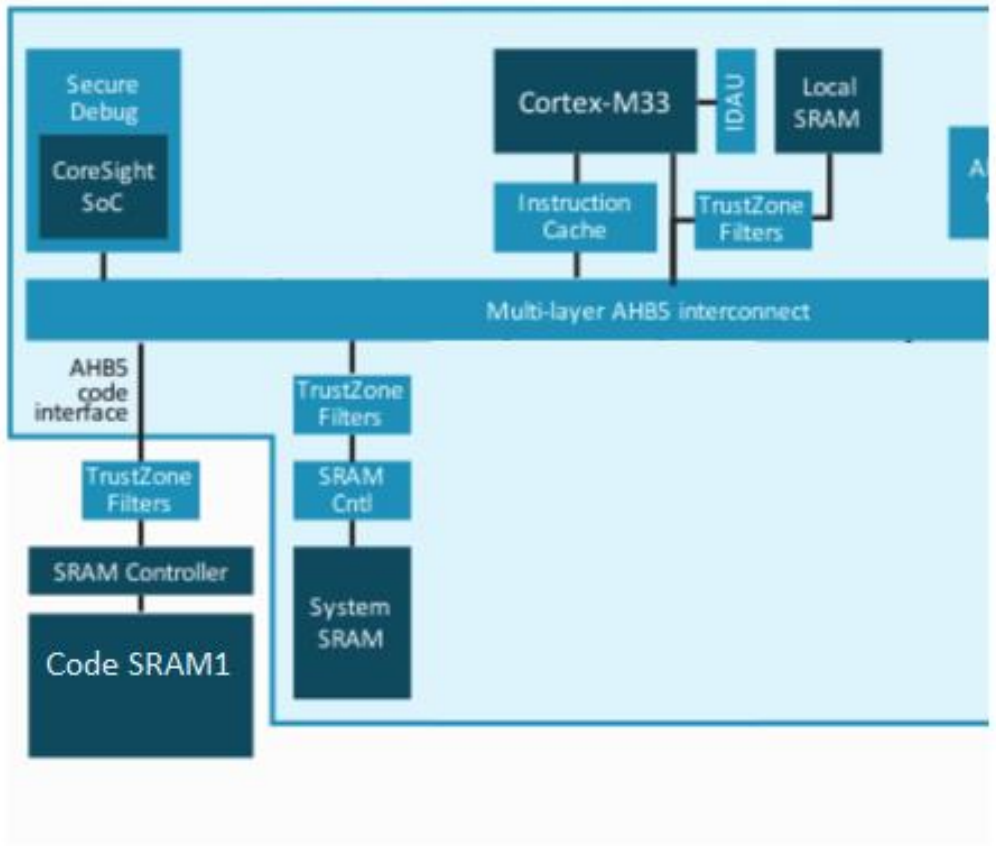
Sub-systems expose only the part of the system that is relevant for the user.

A sub-system user has no visibility to other parts of the system (access protection).

CMSIS-Zone – configuration steps - example

- Step 1: split the multi-processor system into single processor sub-systems.
- Step 2: create the partitions for secure and non-secure execution.
- Step 3: configure MPU protected execution zones.



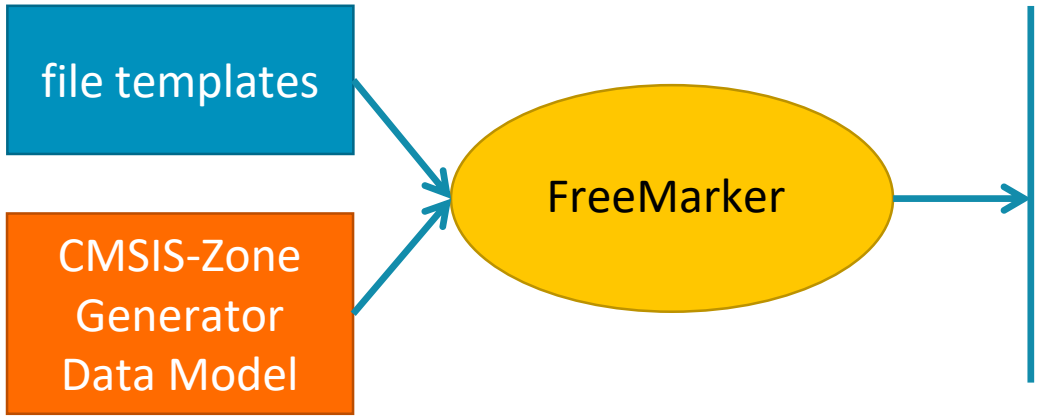


CMSIS-Zone – data export for projects

FreeMarker template engine allows to export CMSIS-Zone data to arbitrary formats

Build

Flexible data export for project build supports many different use cases:
i.e. device configuration, MPU setup, linker scripts, etc.



Project files for:

- Memory assignments
- Linker configuration
- SAU, MPU configuration
- Peripheral assignments

MCBSTM32F400.szone | Blinky_MPU.pzone | scatter.sc

```
LR_flash 0x08000000 0x00080000 { ; lo
  ER_flash 0x08000000 0x00080000 { ;
    *.o (RESET, +First)
    *(InRoot$$Sections)
    .ANY (+RO)
    .ANY (+XO)
  }
  RW_priv 0x20000000 0x00008000 { ; P
    .ANY (+RW +ZI)
    .ANY (.data.priv*)
    .ANY (.bss.priv*)
  }
  RW_shared 0x20008000 0x00008000 { ;
    .ANY (.data.shared*)
    .ANY (.bss.shared*)
  }
  RW_processA 0x20010000 0x00000200 {
    .ANY (.data.processA*)
    .ANY (.bss.processA*)
  }
  RW_processB 0x20010400 0x00000200 {
    .ANY (.data.processB*)
    .ANY (.bss.processB*)
```

arm

Platform Security Architecture, Trusted Firmware-M and CMSIS alignment

Shebu V. Kuriakose

arm

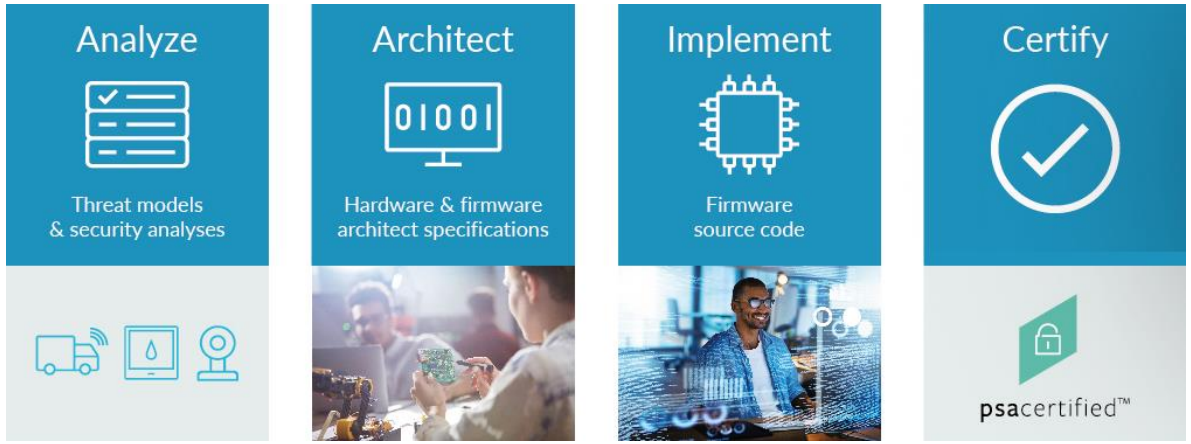
Trusted Firmware-M (TF-M)

To Secure Trillion Connected Devices

Shebu V. Kuriakose

Platform Security Architecture

A complete security offering independently tested

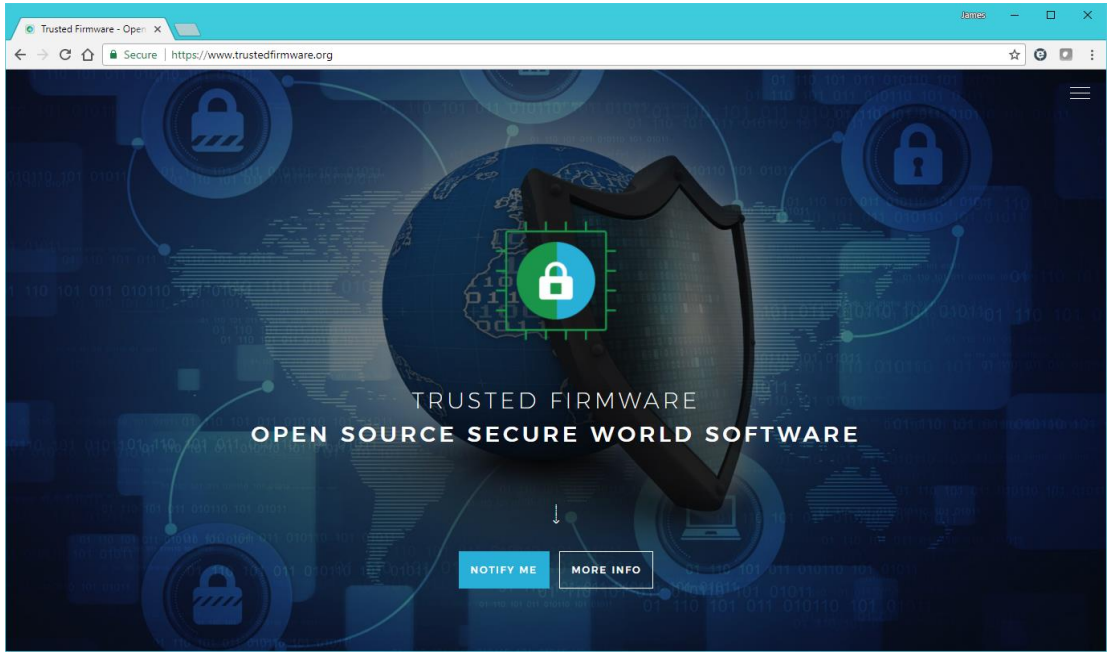


Find out more at **2pm on 28th February** with Rob Coombs: Building Trust: Evaluating Platform Security Architecture (PSA)

Trustedfirmware.org

Open Source, open governance

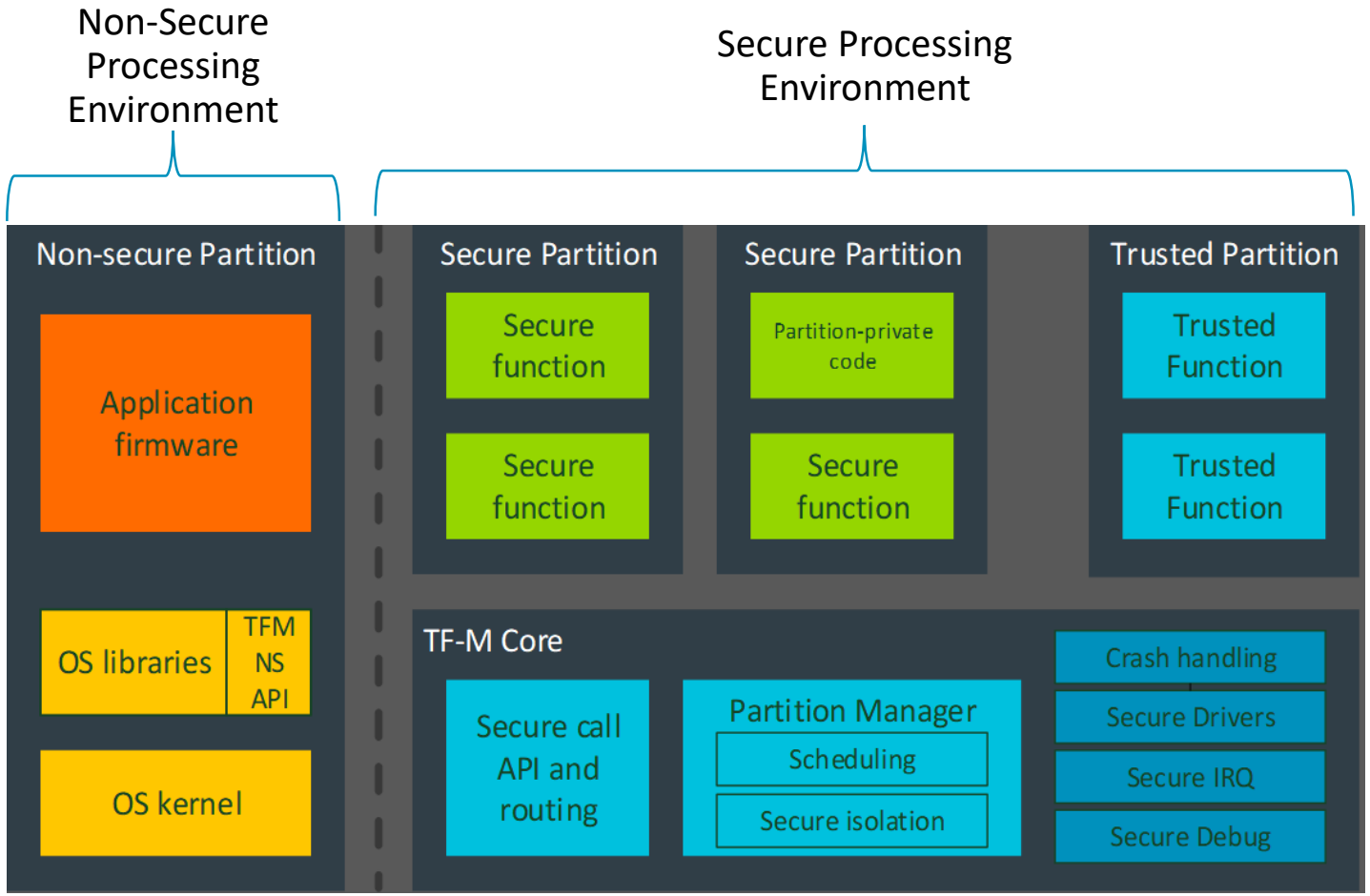
Openly published



arm

Trusted Firmware-M Framework

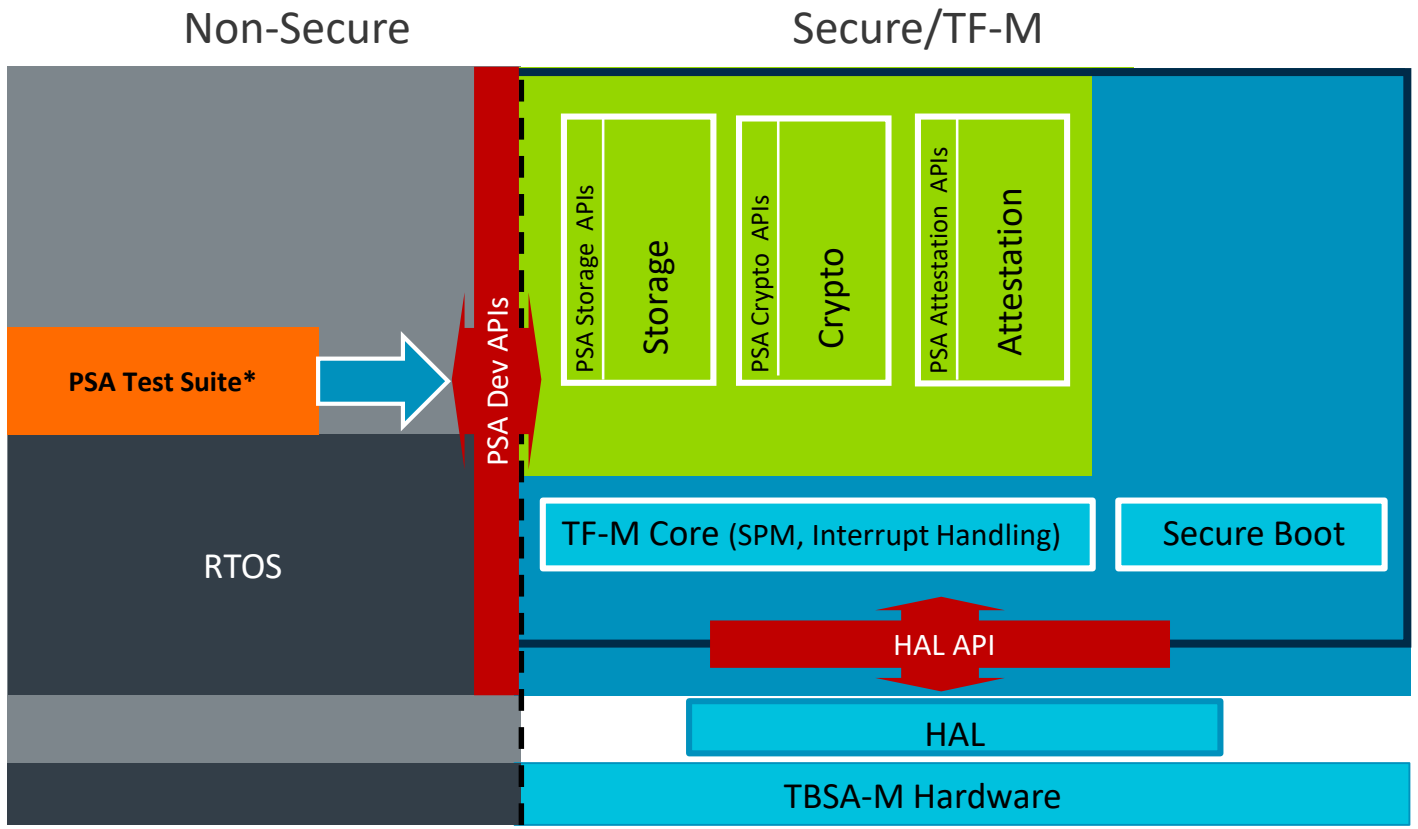
Secure Boot, Isolation, Secure Partition



TF-M v1.0-Beta @ Embedded World'19

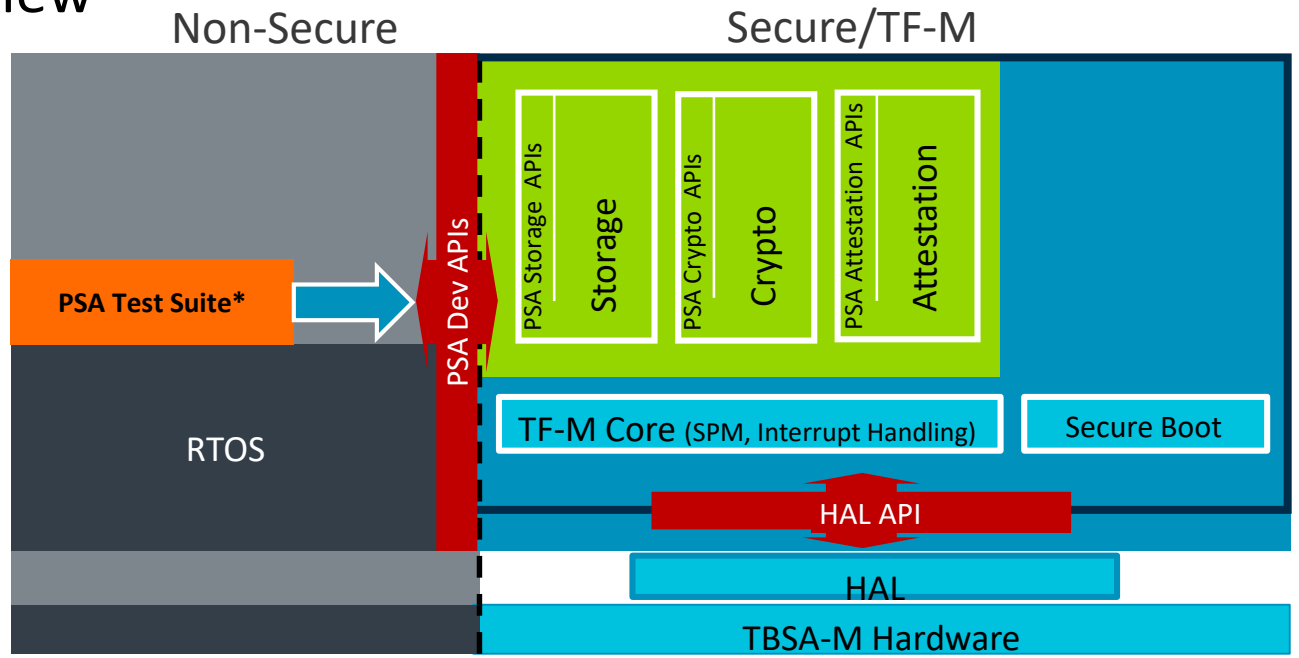
- Secure Boot based on mcuboot
- Level 1 Isolation
- Secure Storage
- Crypto Functions
- Attestation following EAT (Entity Attestation Token) Spec
- PSA Functional API Certification

  PSA L1 Security Certification



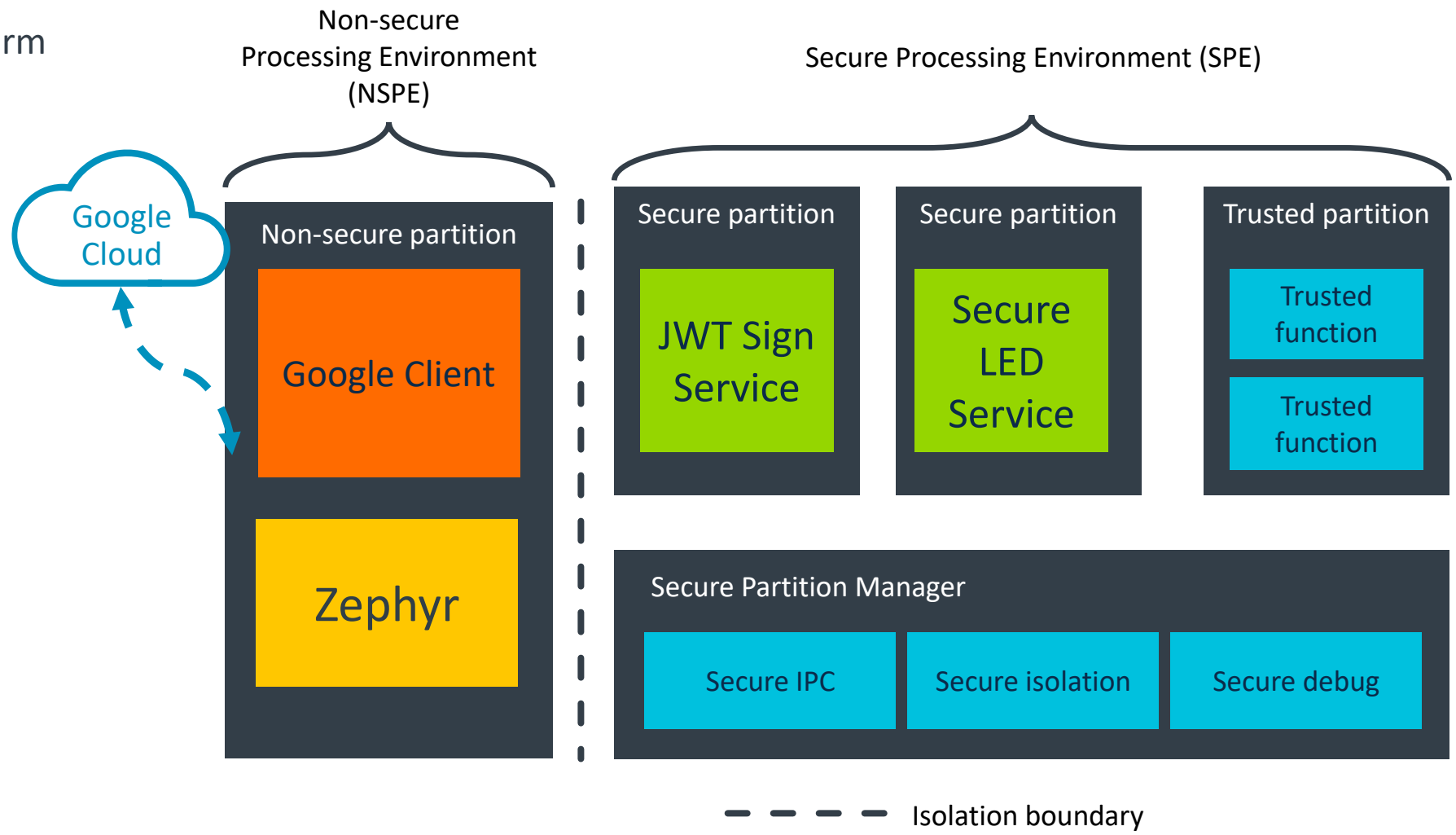
Trusted Firmware-M: Enabling a Secure Ecosystem

- Sample Integration with RTOS
- CMSIS Packs
- Standardized PSA APIs
- Standard HAL Interface
- Public Mailing List: Open Design and Review
- Multiple Toolchains
- Certifiable Friendly



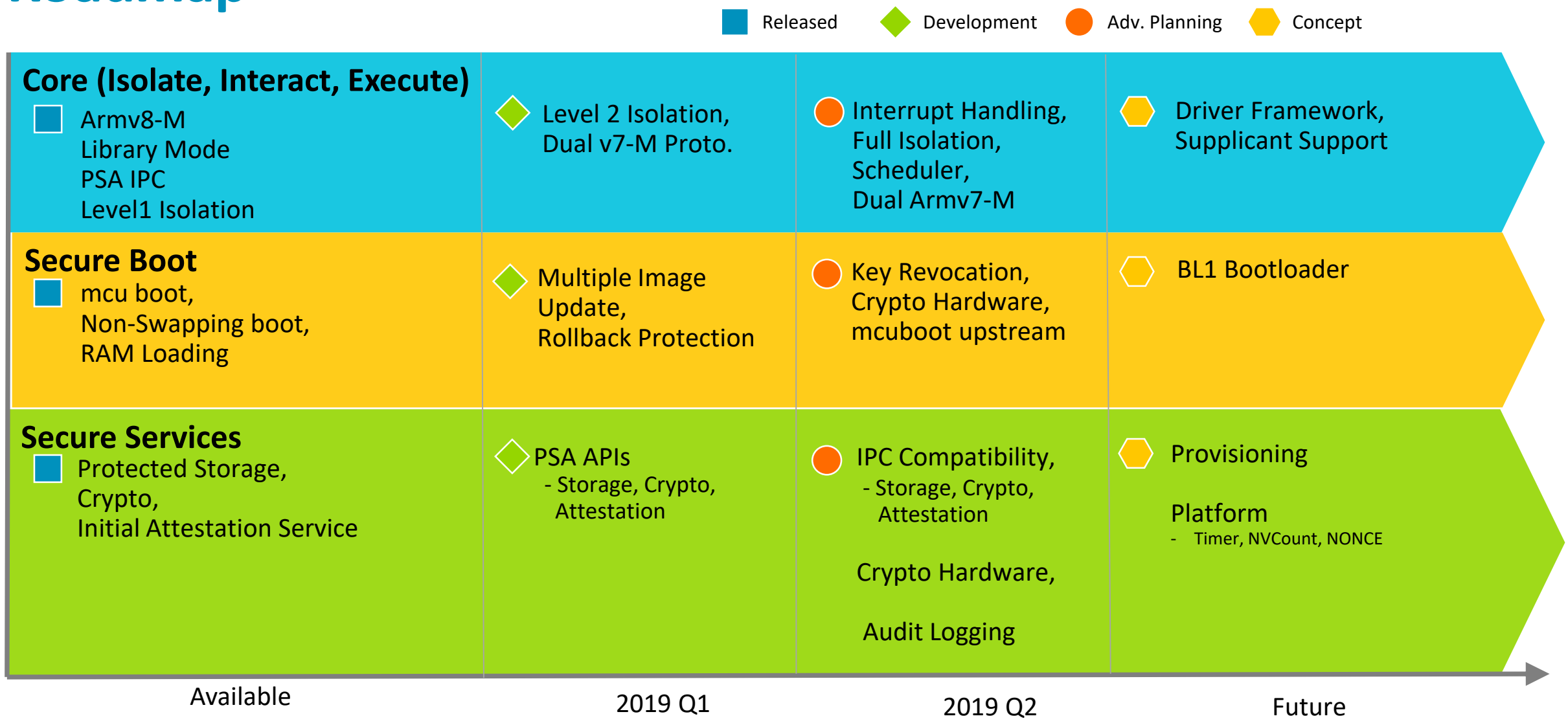
Proof Of Concept: Google Cloud/Zephyr/TF-M

- Musca-A Board
  - Arm IoT Reference Platform
  - Cortex-M33 core
- SPE - TF-M
- NSPE - Google Client on Zephyr RTOS
- Google Client uses JWT Sign Service and Secure LED Service



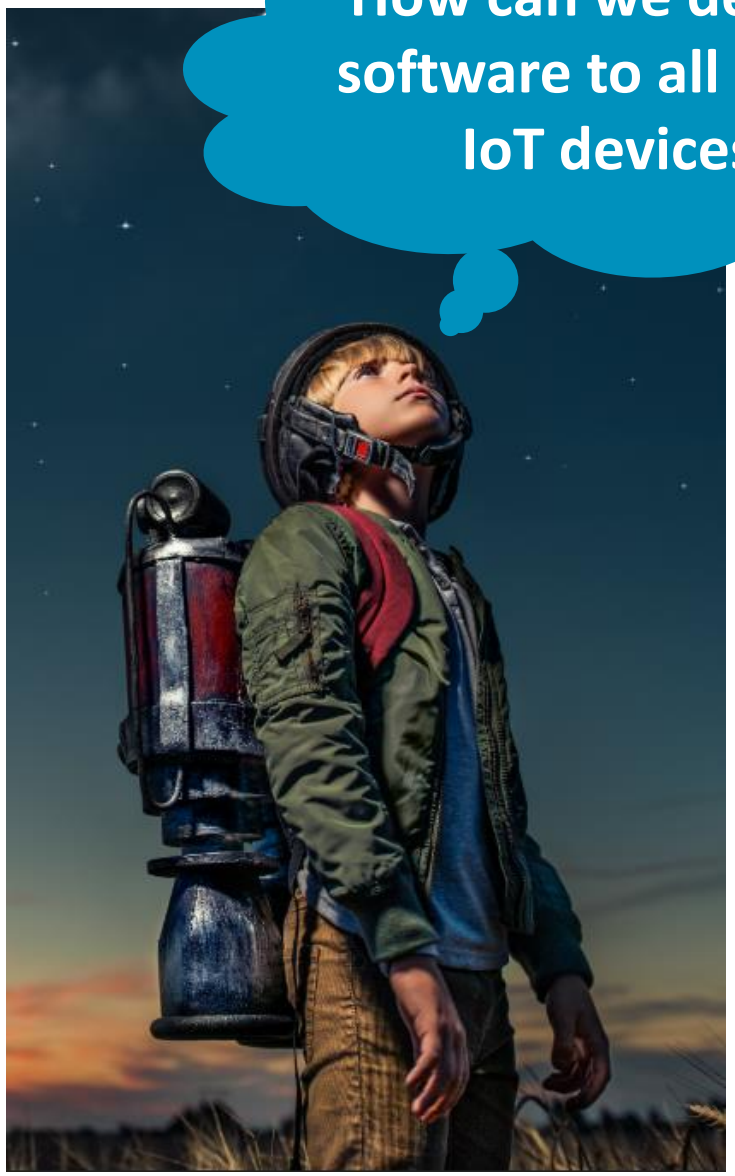

PSA Launched
Q3’2017
TF-M v0.1
Q1’2018
PSA Specs, TF.org
Q4’2018
TF-M v1.0-Beta PSA Func, L1 Certified
Today
TF-M v1.0 L2 Certified
2019

Roadmap

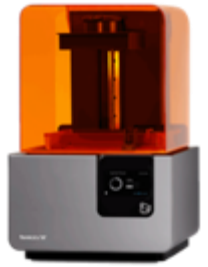


arm

How CMSIS and TF-M software packs simplify IoT end-node development



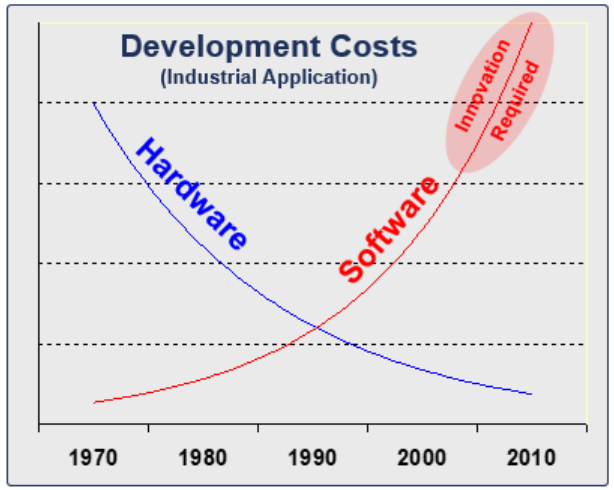
How can we deploy software to all these IoT devices

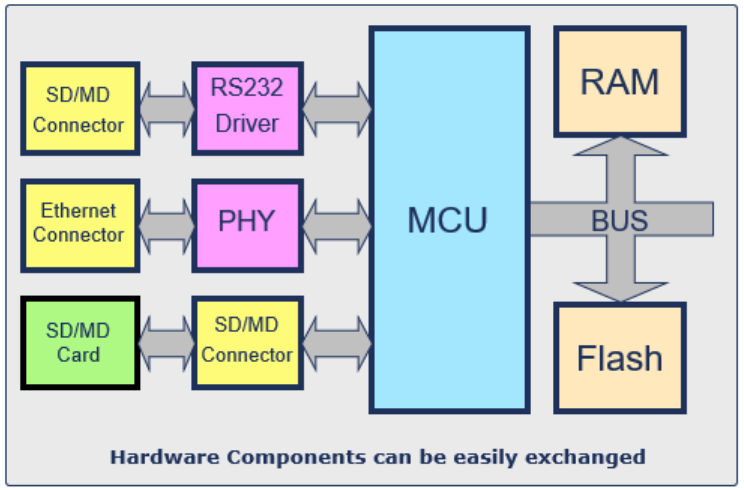


arm

CMSIS vision 10 years ago: SW components easy to exchange

Software Complexity – The Challenge





- Well-known issues the drive software costs
  - Increasing product requirements that are implemented by software
  - Hardware problems tend to become compensated by software
- Software components are incompatible and cannot be re-used.

Software Standards are key for the future!

THE ARCHITECTURE FOR THE DIGITAL WORLD® Confidential 5 ARM

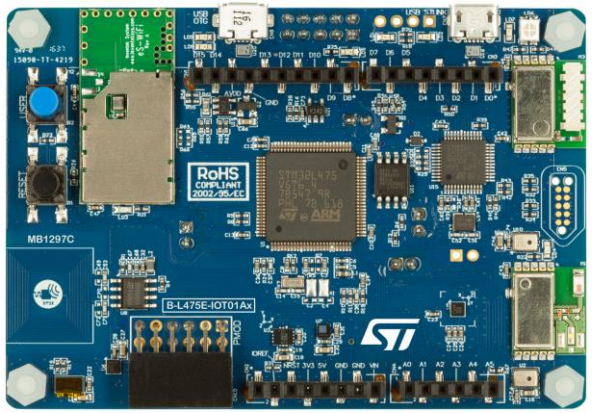
Where are we today?

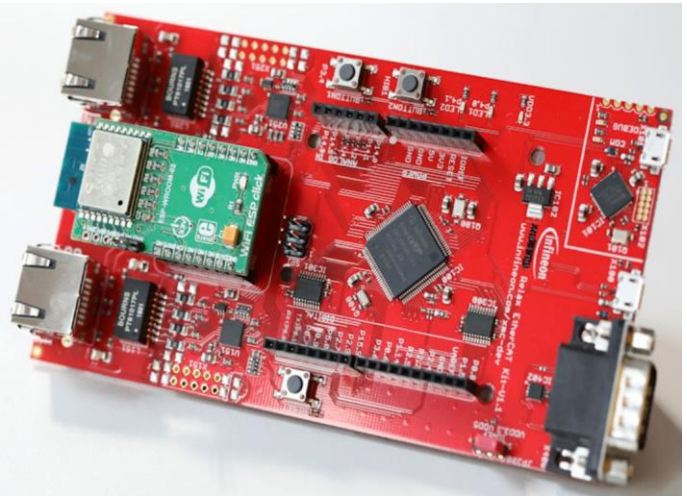
- CMSIS-Core, RTOS, DSP, NN, Driver define common SW for variou processors/devices and peripherals
- TF-M adds security for IoT
- All relevant silicon vendors deliver device family packs in CMSIS format
- CMSIS-Pack defines the frame work for software components
- CMSIS-Pack management is implemented by several mainstream toolchains: IAR, Arm Development Studio, Keil MDK, and SiP toolchains

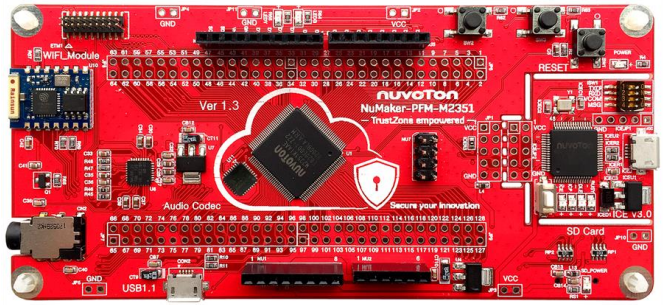
So, lets apply it….

arm

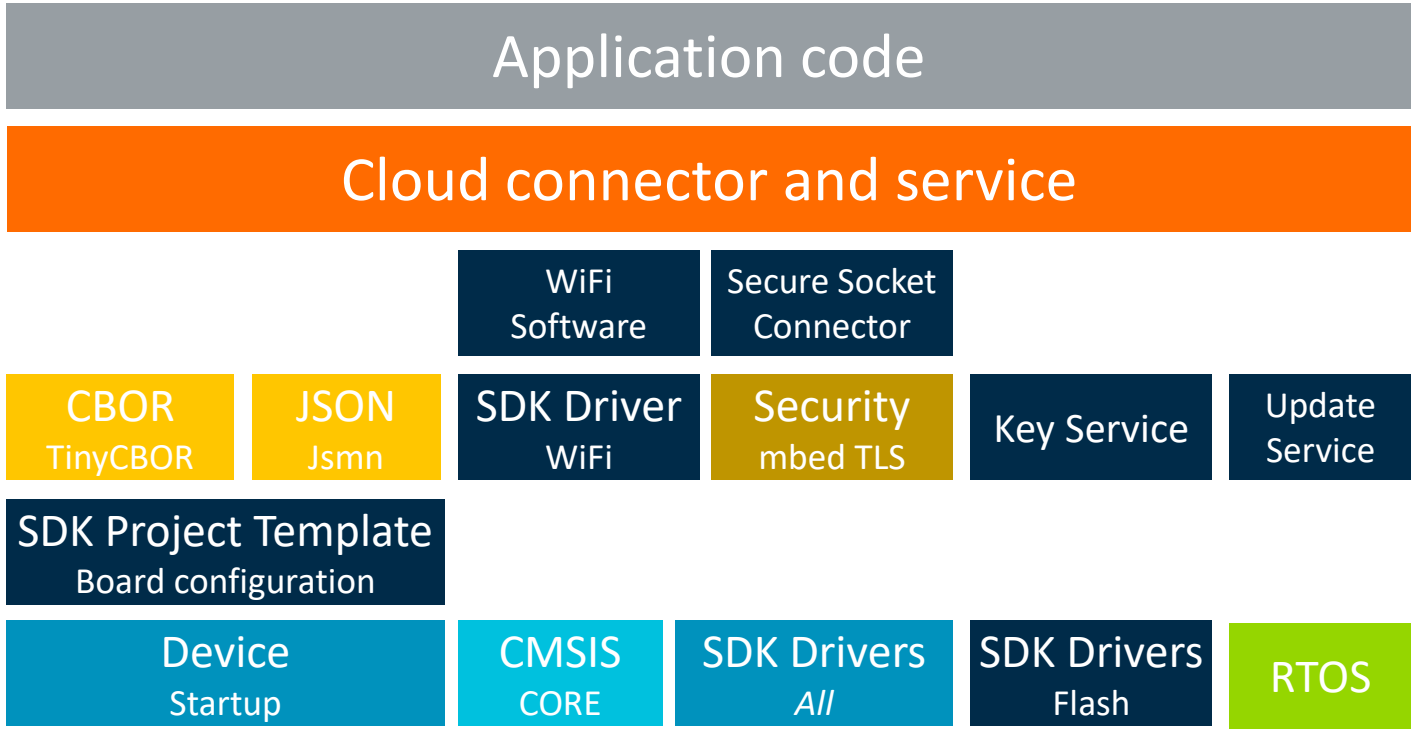
Cloud connector software stack and IoT kits







Cloud connector software stack – framed in packs



Benefit: easy to add new components. SiP can differentiate!

Proof of Concept

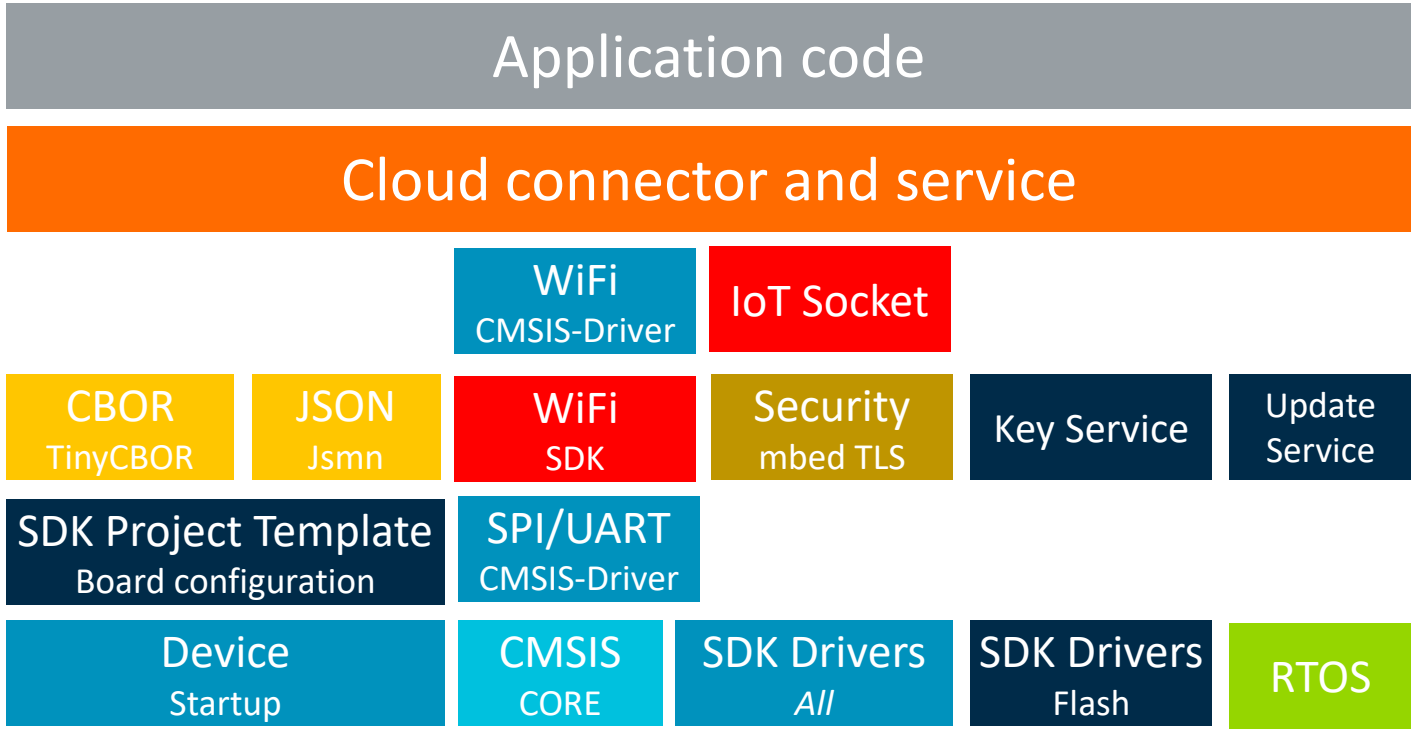
Cloud connector stack re-worked to use software packs.

PoC separates common software components: now it is simpler to re-use and update.

Seven components are hardware dependant and need rework to adopt a new target hardware.

CMSIS pack | RTOS pack | Cloud pack | BSP | DFP | 3rd party pack | Mbed TLS pack

Adding WiFi Driver and IoT Socket packs

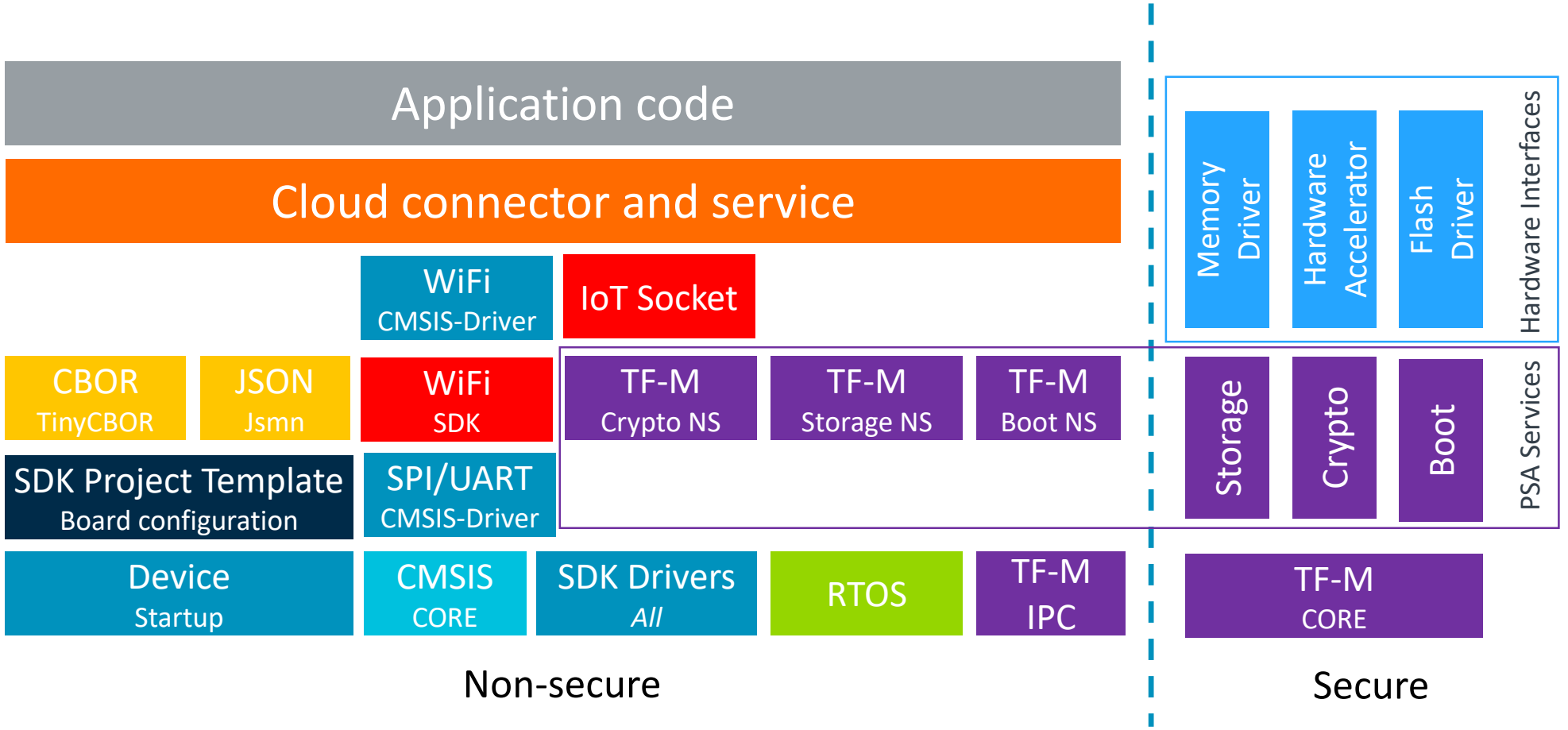


Utilizing generic WiFi, Networking, and IoT Socket packs can further reduce efforts of hardware adoption.

WiFi chipsets frequently interface via SPI. Utilizing the CMSIS-Driver interface makes it generic.

CMSIS pack | RTOS pack | Cloud pack | BSP | DFP | 3rd party pack | Mbed TLS pack | Com pack

Cloud software stack on Armv8-M with TrustZone



CMSIS + TF-M can drastically simplify software configuration to custom hardware

Alignment with CMSIS and TF-M can simplify security, crypto, and firmware update

CMSIS pack | RTOS pack | Cloud pack | BSP | DFP | 3rd party pack | Mbed TLS pack | Com pack

arm

CMSIS timeline	Description	How you can contribute
April 2019	**CMSIS v5.5.0** release with: - Core(M): Armv8.1-M support - Driver: WiFi API - DSP: resolved reported issues	Review live repository on https://github.com/arm-software/cmsis_5 Use '**Issues**' to report problems or raise requests
April/May 2019	WiFi driver implementations and IoT Connector release	WiFi chip set vendor: add your own driver Tool vendor: adopt projects to your toolchain
April 2019 April – June July 2019	**CMSIS-Zone** beta version: - examples: for v8M devices - heterogenous system setup - RTX5 with MPU protection **CMSIS-Zone** final release	Feedback on current CMSIS-Zone specification Review example projects, adapt examples to your toolchain or RTOS Open source project with examples on https://github.com/arm-software/CMSIS-Zone
June 2019	TF-M with generic HAL adopted to several Cortex-M23/33	tbd
Outlook	CMSIS-DAP: Secure Debug reference implementation CMSIS-SVD: extensions for v8-M devices to provide data to CMSIS-ZONE CMSIS-Pack: Generic project file format	

arm