Using TrustZone for ARMv8-M on ARM Cortex-M23 and ARM Cortex-M33

ARM

Christopher Seidl
Technical Marketing Manager

Thomas Ensergueix
Director, Product Marketing

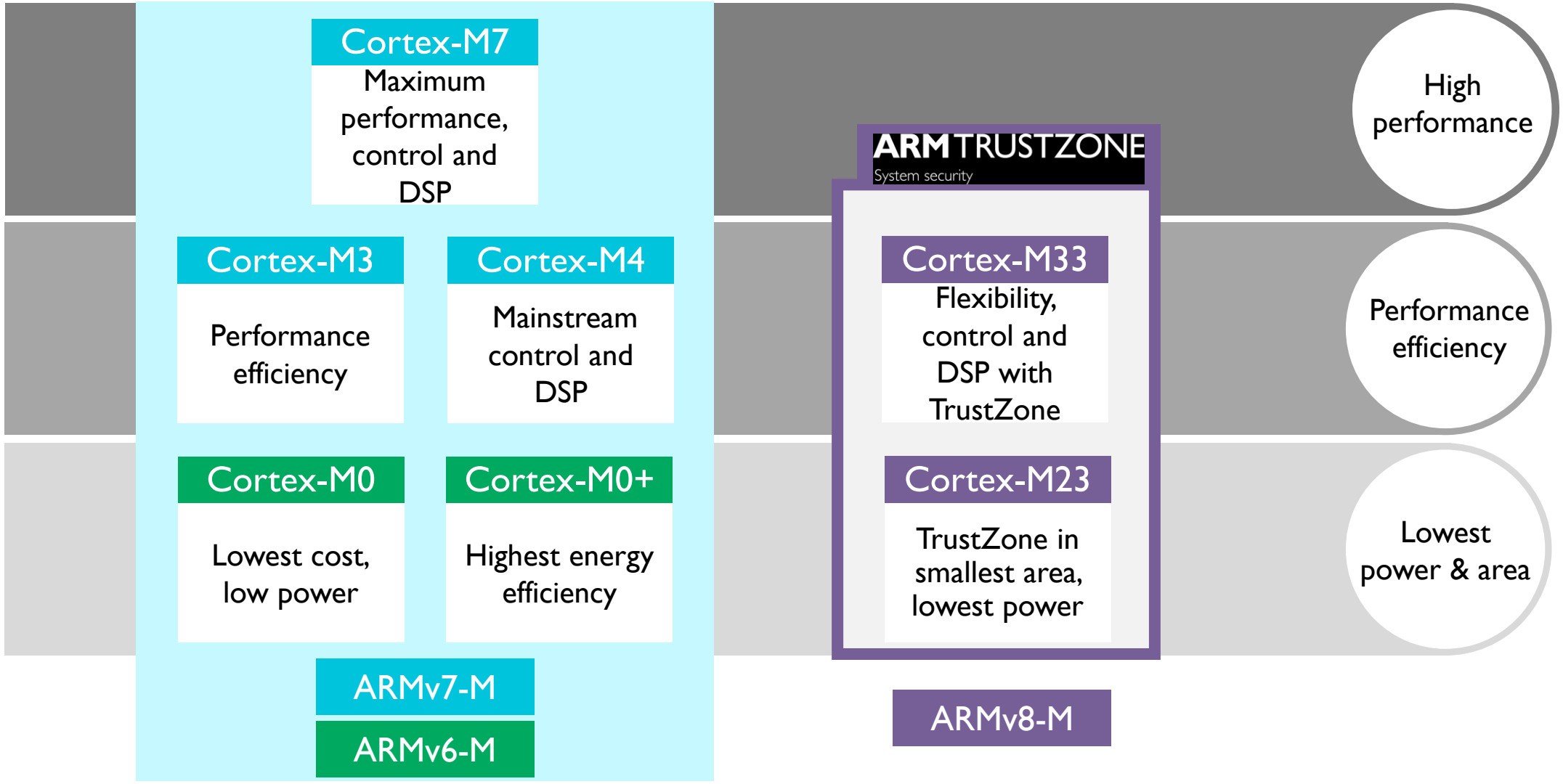
Webinar
11/17/2016

Agenda

- Introduction of ARM® Cortex® -M23 and ARM Cortex-M33
- TrustZone® for ARMv8-M: Security foundation in hardware
- Software development tools and software components
- Demo

ARM Cortex-M23 and ARM Cortex-M33

Bringing TrustZone to the Cortex-M family



ARM

TrustZone for ARMv8-M

Separation and access control Isolate trusted software and resources

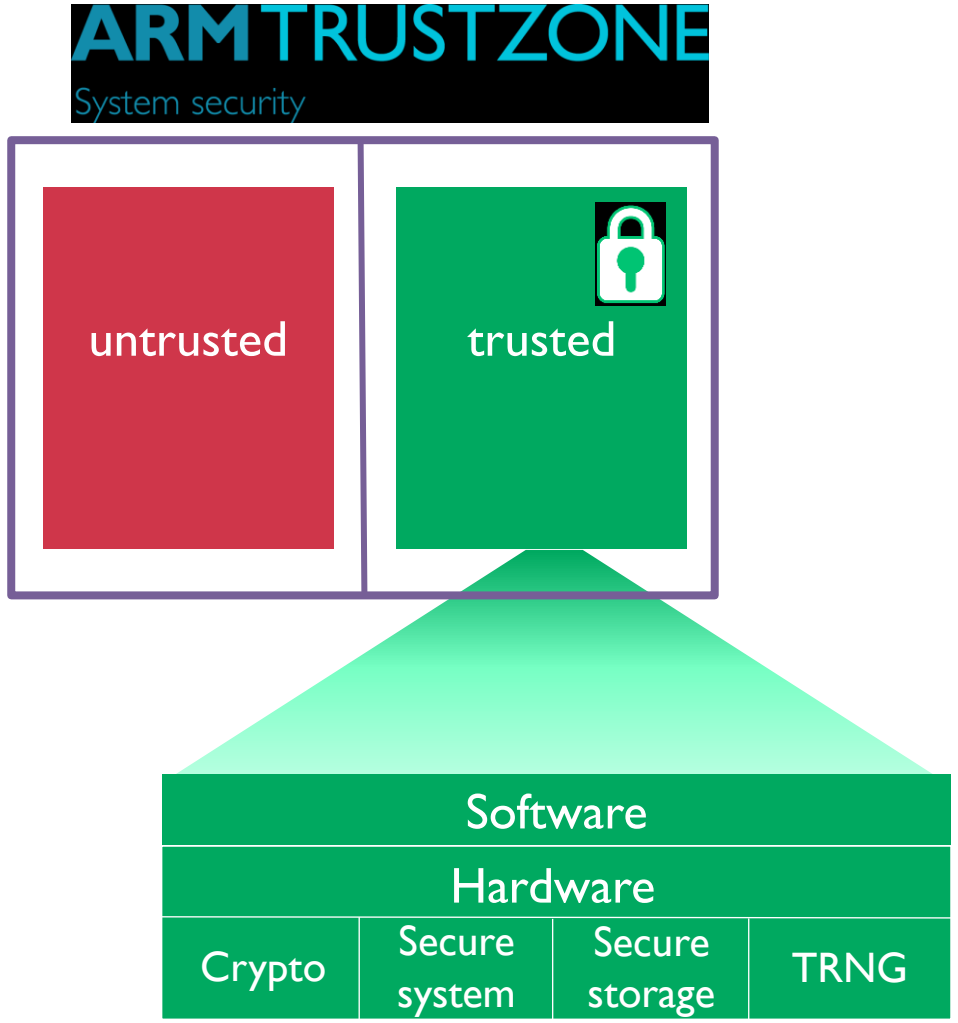
- Reduce attack surface of key components

Trusted software

- Provision of security services
- Small, well-reviewed code

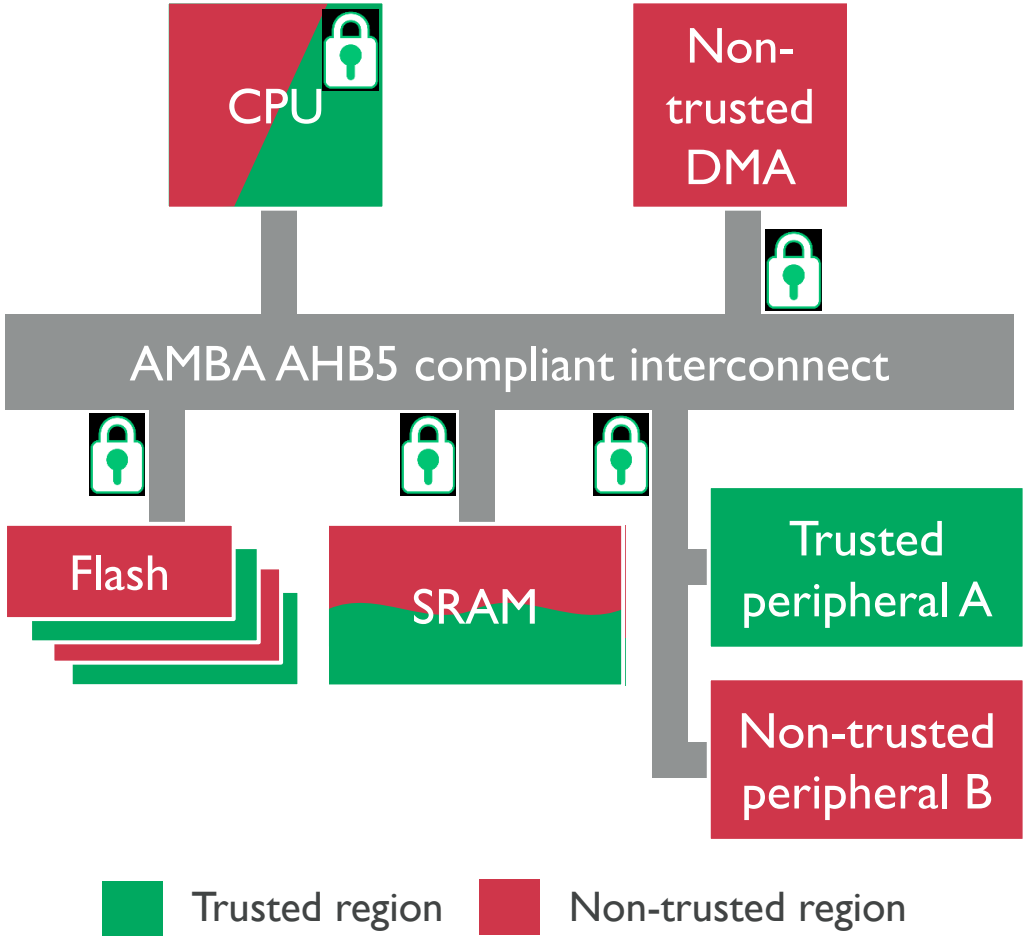
Trusted hardware

- Hardware assist for cryptography
- Secure-access validation built into SoC



Bringing security protection to the system

- Secure the system, secure the processor
  - Hardware separation and isolation
  - Protect memories, peripherals, legacy IP
- AMBA AHB5 bus protocol
  - Signals security through the interconnect
  - Complementary to ARMv8-M
- Optimized for embedded systems
  - Fewer wires saves area and power
  - Hardware protection simplifies software

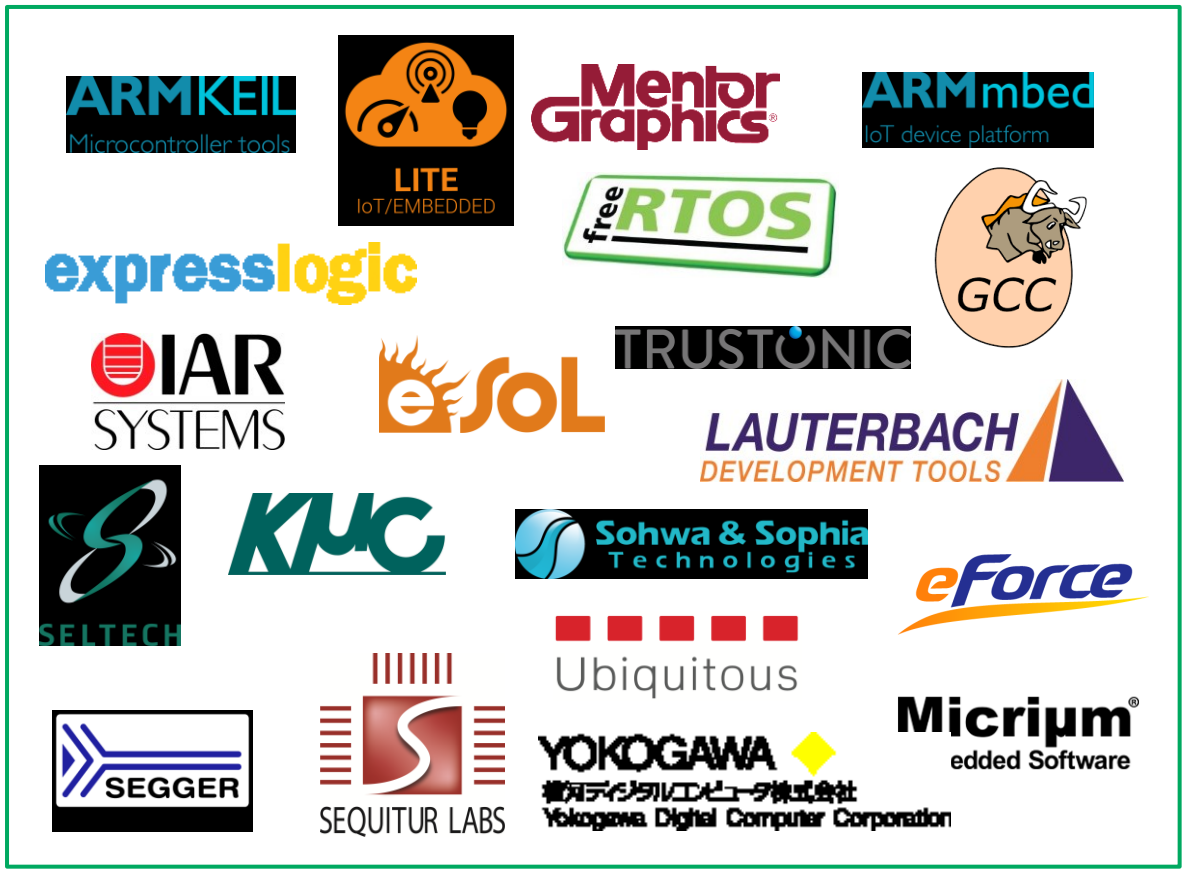


ARM

Ever-expanding world's #1 embedded ecosystem

Public silicon lead partners

Public ecosystem lead partners

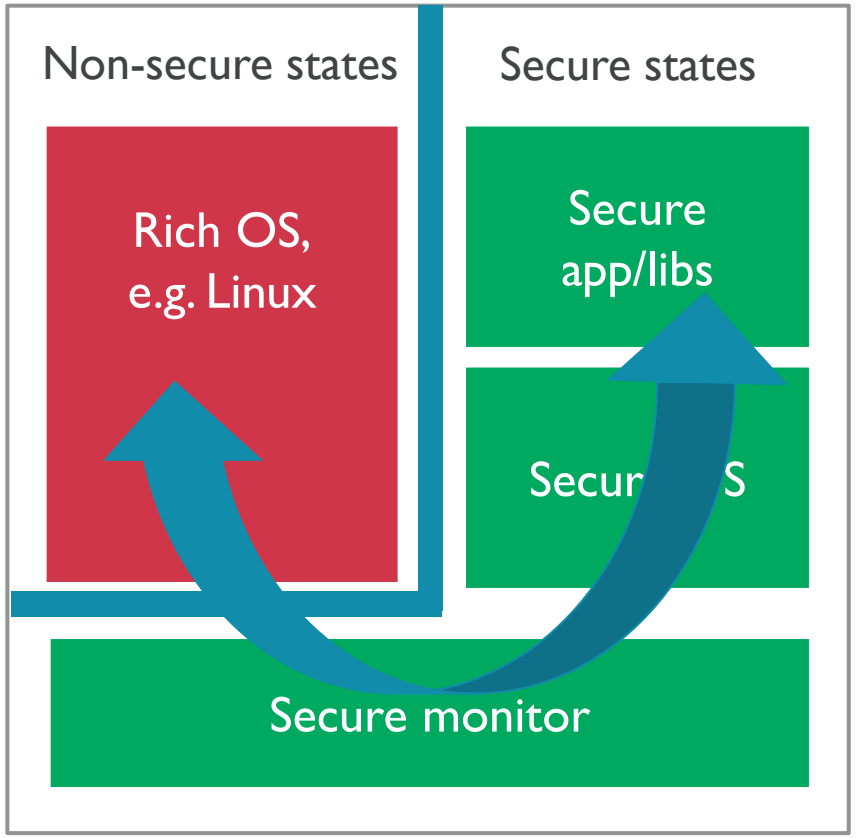


ARM

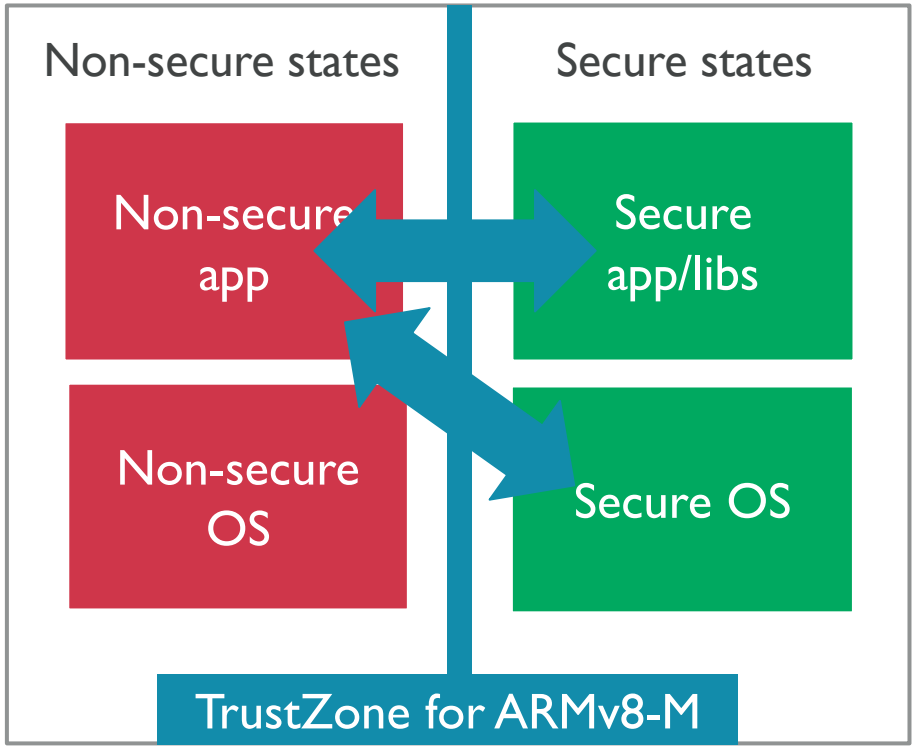
TrustZone for ARMv8-M

Security foundation in hardware

TrustZone for ARMv8-A



TrustZone for ARMv8-M



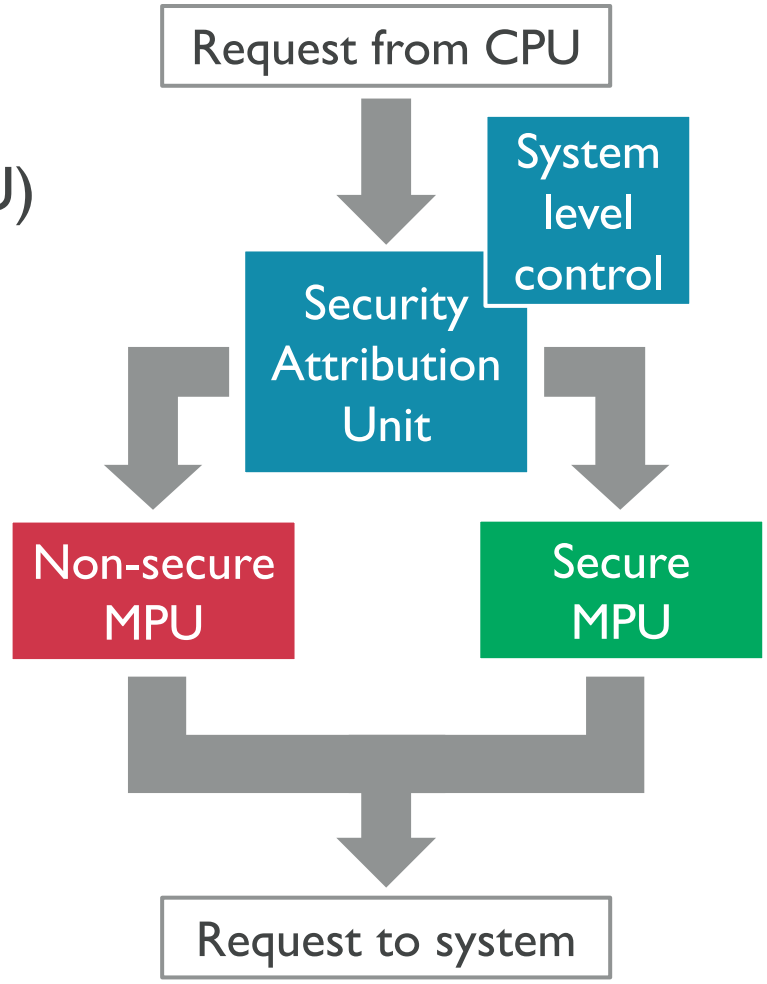
Secure transitions handled by the processor
to maintain embedded class latency

ARM

Security defined by memory map

All transactions from core and debugger are checked

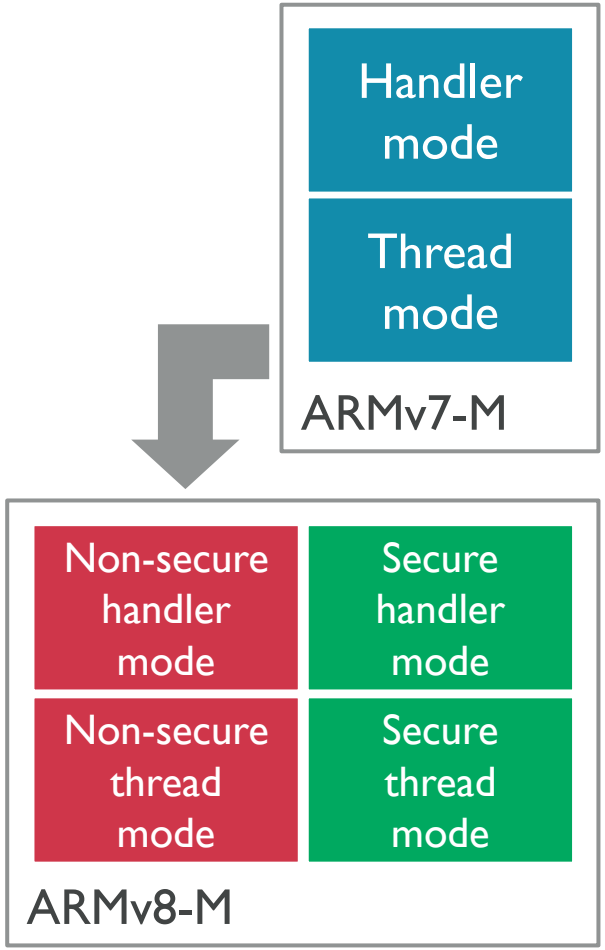
- All addresses are either secure or non-secure
- Policing managed by Security Attribution Unit (SAU)
  - Internal SAU similar to MPU
  - Supports use of external system-level definition
  - For example, based on flash blocks or per peripheral
- Banked MPU configuration
  - Independent memory protection per security state
- Load/stores acquire non-secure attribute based on address
  - Non-secure access to secure address → memory fault



ARMv8-M additional states

Existing handler and thread modes are mirrored

- Secure and non-secure code runs on a single CPU
  - For efficient embedded implementation
- Secure state for trusted code
  - New secure stack pointers for robust operation
  - Addition of stack-limit checking
- Dedicated resources for isolation between domains
  - Separate memory protection units for secure and non-secure
  - Private SysTick timer for each state
- Secure side can configure target domain of interrupts

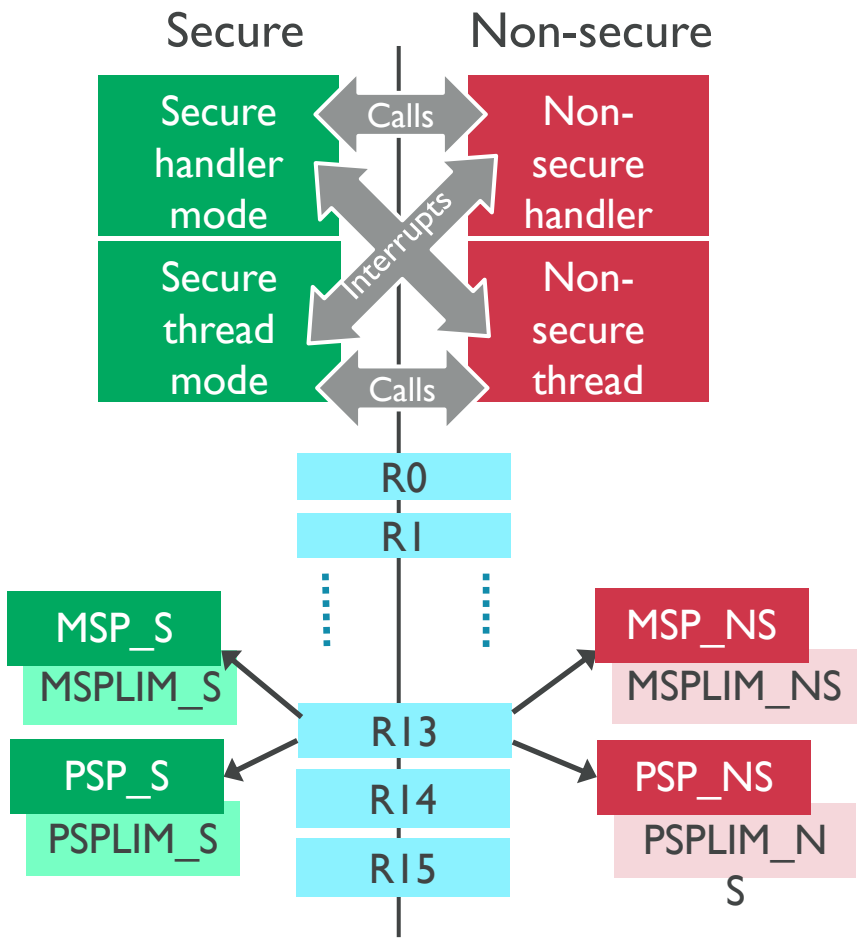


ARM

High performance cross-domain calls

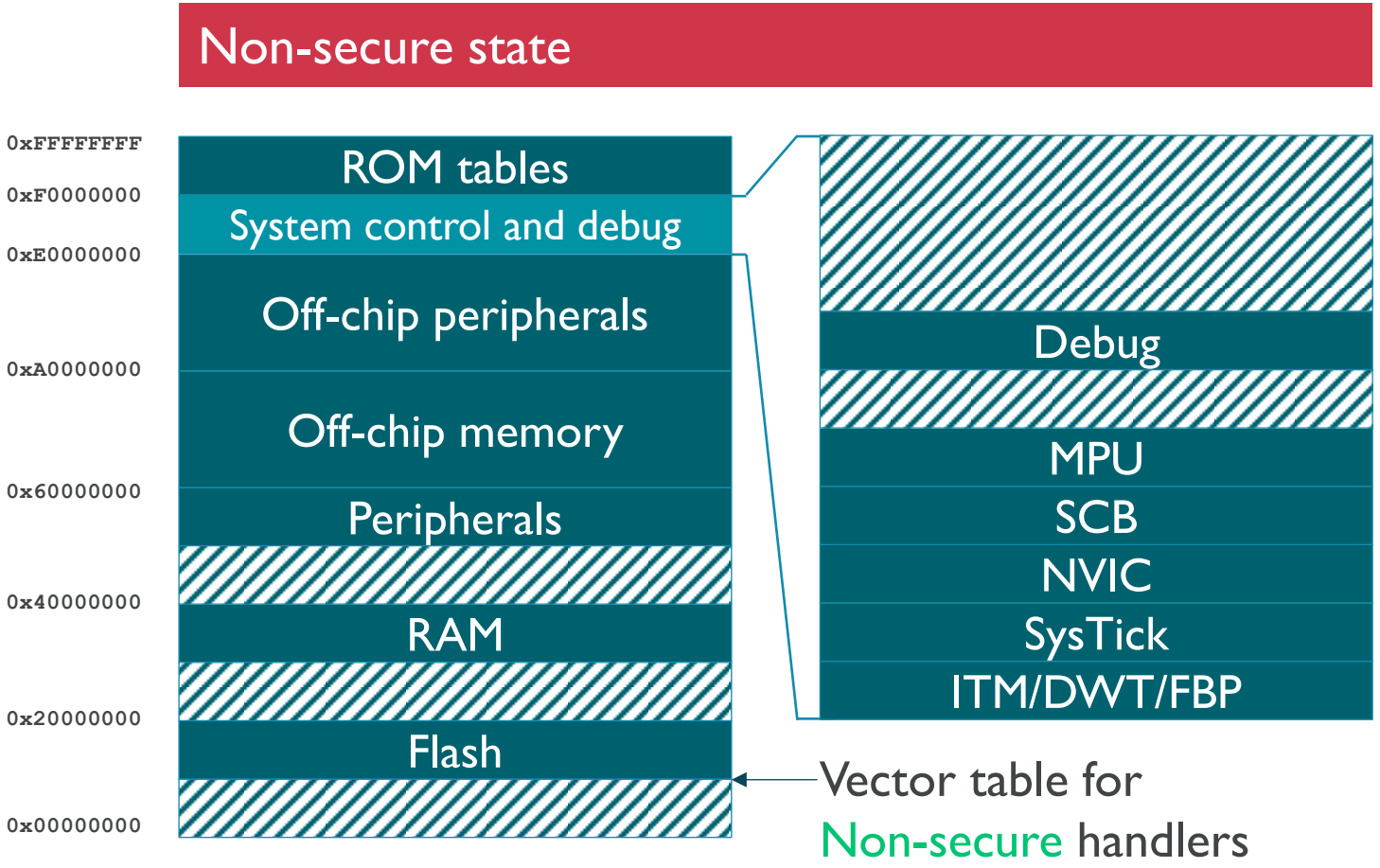
Efficient implementation focused on microcontroller

- Security inferred from instruction address
  - Secure memory considered to hold secure code
- Direct function calls across boundary
  - High performance and high security
  - Multiple entry points
  - No need to go via 'monitor' for transitions
- Uses Secure Gateway (SG) instruction
  - Only permitted in special secure memory with non-secure callable (NSC) attribute



ARM

ARMv8-M programmer's model: Memory map



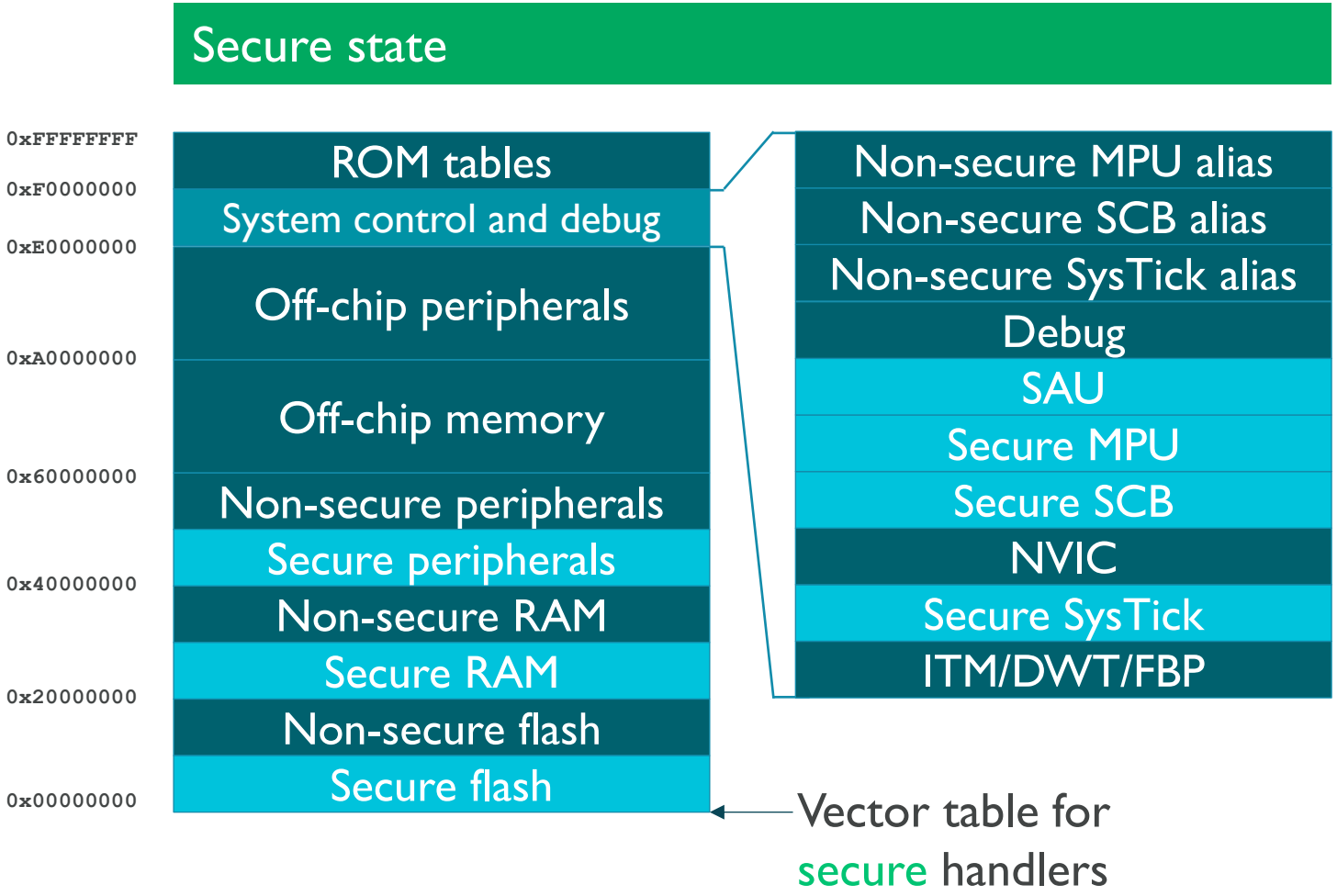
Non-secure memory view is identical with Cortex-M

Branches to fixed memory locations access secure firmware

Secure memory is invisible

ARM

ARMv8-M programmer's model: Memory map



Secure memory view shows additional Flash, RAM, and peripherals

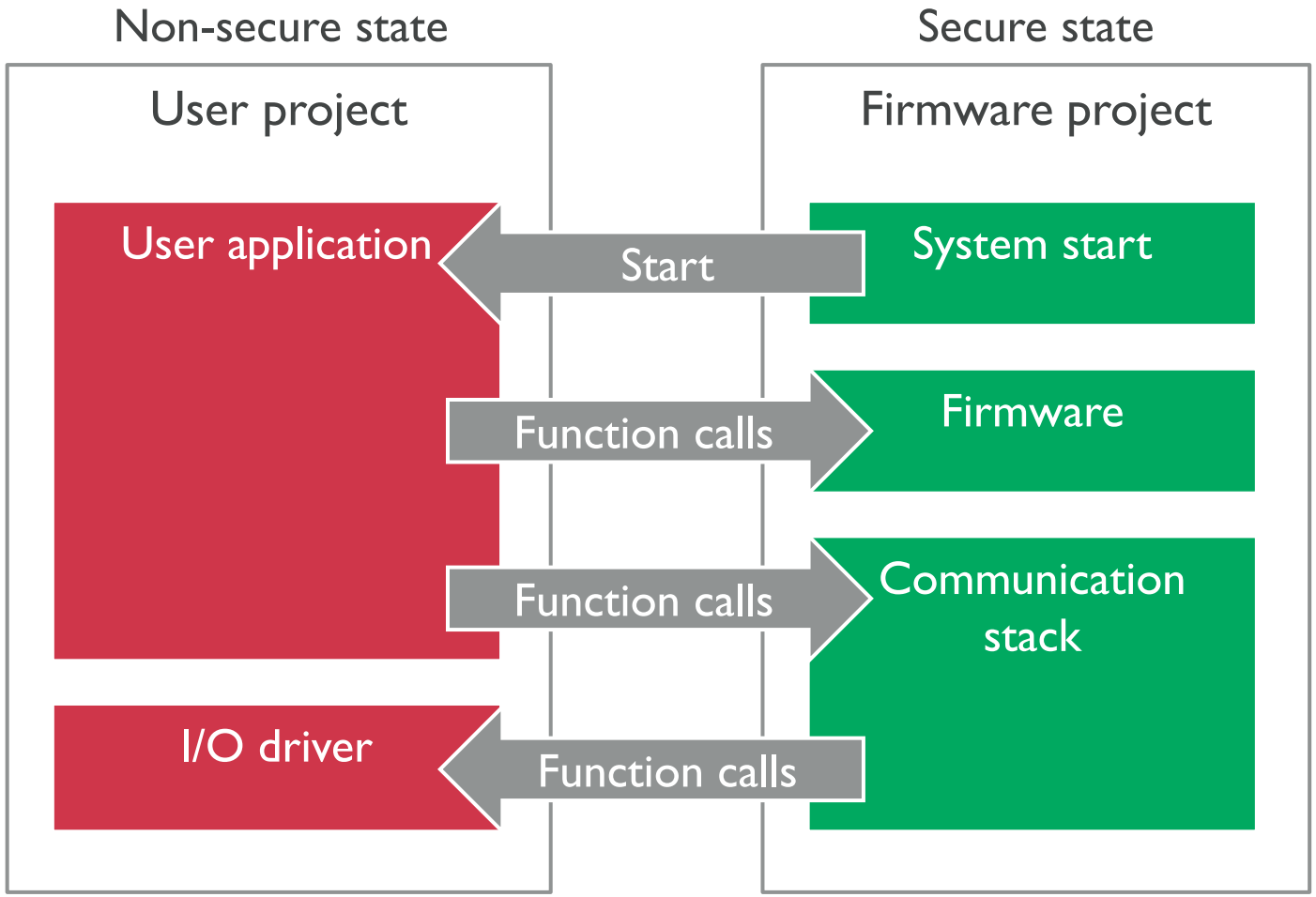
Access to all regions is possible in secure state

Regions can be configured in secure state using the SAU

ARM

A simplified use case

Composing a system with secure and non-secure projects



- Non-secure projects cannot access secure resources
- Secure project can access everything
- Secure and non-secure projects may implement independent time scheduling

ARM

Software development tools and software components

Accelerate software creation for ARMv8-M devices with TrustZone

Tools and components for software development

Keil® MDK
IDE & debugger

ARM
Compiler 6

CMSIS v5

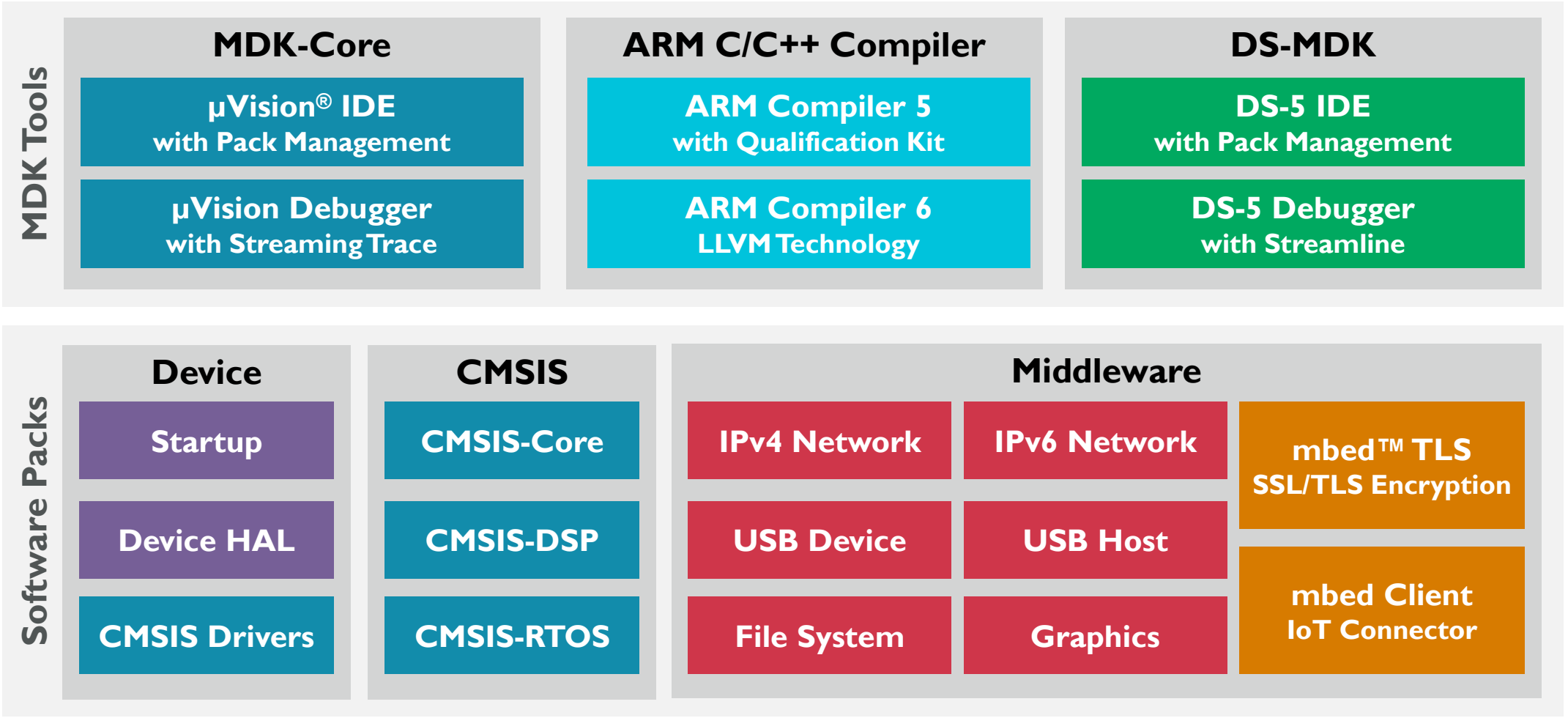
Fast Models

ULINK™
debug adapters

MPS2
Cortex-M
Prototyping System

Keil MDK Microcontroller Development Kit

Most comprehensive development solution supporting over 3600 devices



ARM

CMSIS: Pathway to the ARM ecosystem

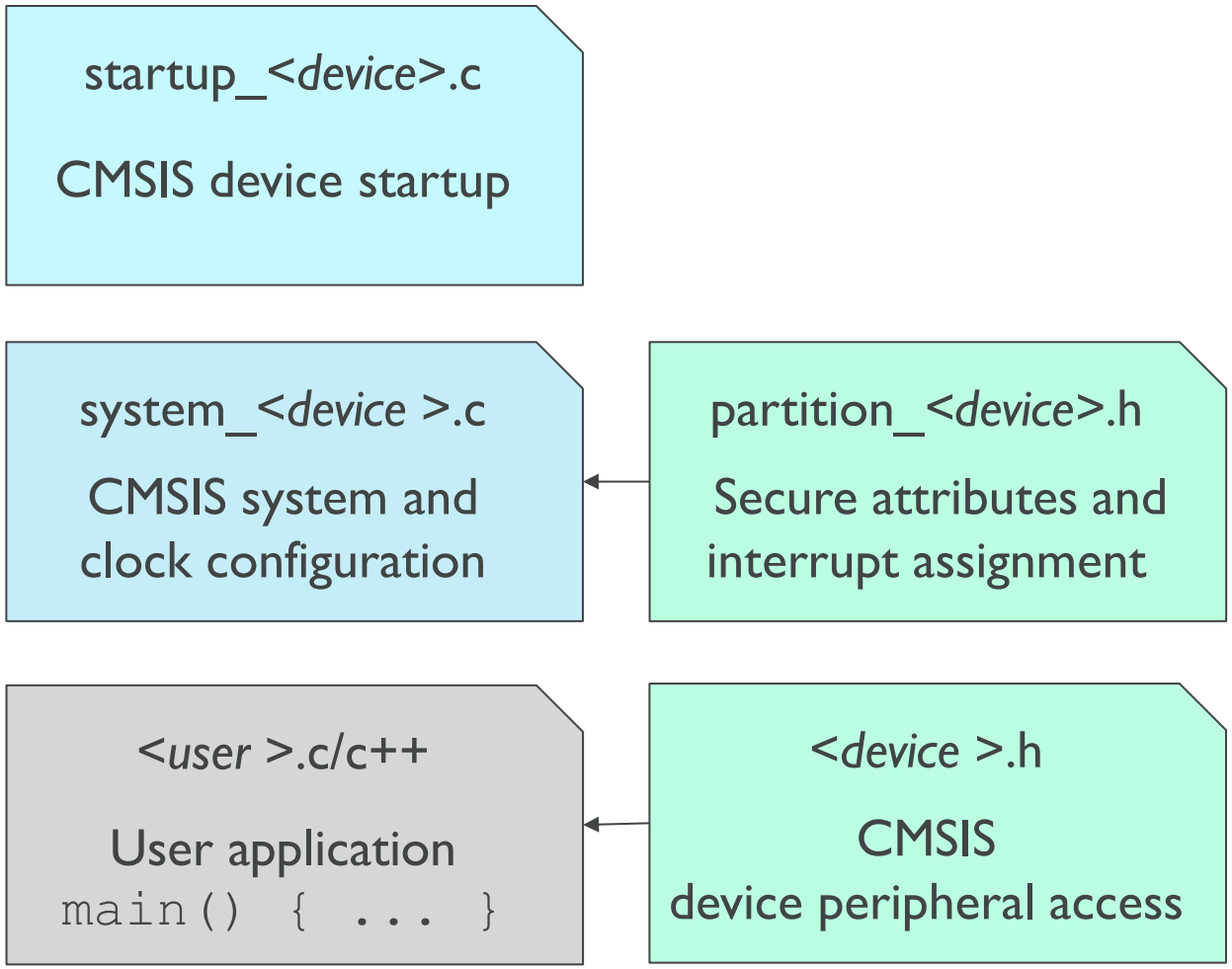
- Vendor-independent hardware abstraction layer for Cortex-M series
  - Open source software framework with processor HAL, DSP library, and RTOS kernel
- Consistent, generic, and standardized software building blocks
  - Optimized API that software creation, code portability, and middleware interfaces
- Infrastructure to accelerate time to market for device deployment
  - Software Packs to distribute device support, board support, and software building blocks

3668
devices supported

1.2M+
source files on GitHub

3M+
downloads in past six months

CMSIS-CORE for secure mode projects

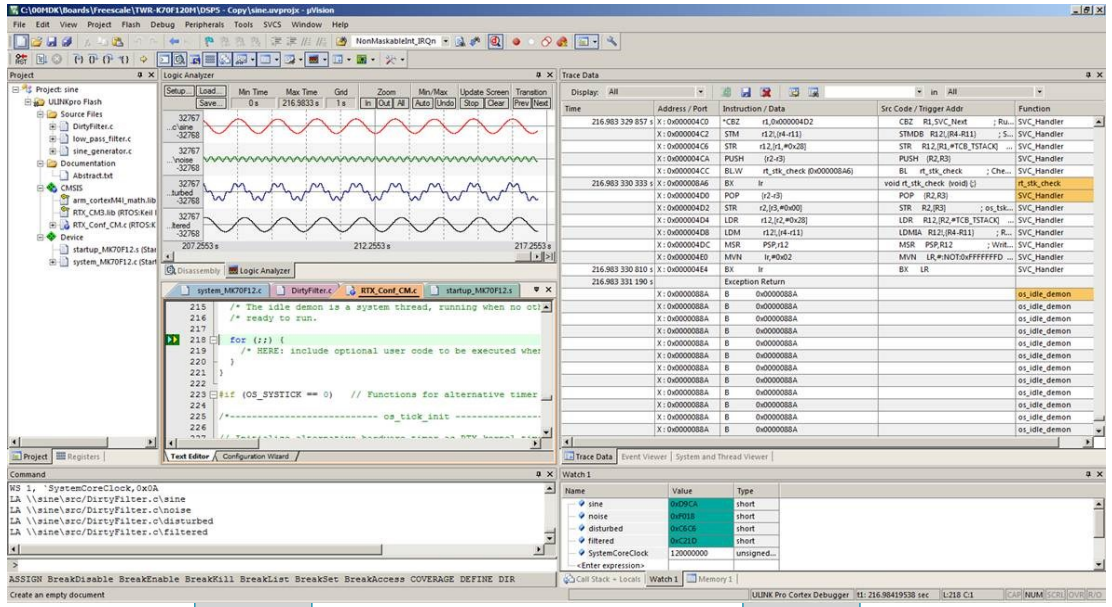


Files relating to CMSIS-CORE including device specific files

partitions.h provides initial setup for SAU and configures non-secure mode memory areas and interrupts

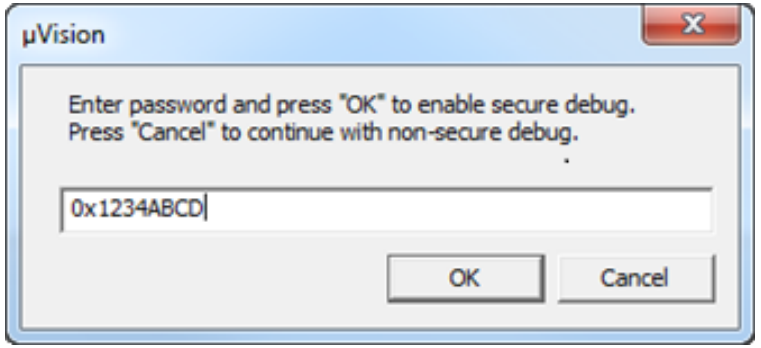
ARM

Debugging of software projects



ARM Fast Models
Virtual Prototyping

- MDK offers debugging with:
  - Simulation environment for software development prior silicon availability
  - MPS2 target connection for testing with microcontroller prototypes
- Secure & Non Secure Debug Access



Enter password for Secure Debug Access

ARM

Demo

More information

- Application note 291: Using TrustZone on ARMv8-M (keil.com/appnotes/docs/apnt_291.asp)

- Upcoming webinars (keil.com/events):
  - Dec 1st: Dynamic software analysis with MDK event recorder
  - Dec 8th: What's new in CMSIS-RTOS2 and Keil RTX5

Summary

- ARMv8-M provides the architecture for the next generation of secure connected embedded devices
- Software and tools make it easy for developers to use secure mode
- CMSIS provides software building blocks for faster time to market of embedded applications that require security

ARM