

CMSIS Partner Meeting

Embedded World 2017

ARM

Reinhard Keil
Senior Director MCU Tools

Nuremberg – Embedded World 2017
14. March 2017



 SoftBank **ARM**

ARM is now part of Softbank

- No change to ARM's business model
 - No change to ARM's organisation
 - No change to DSG
-
- Expect more investment in both ARM traditional and new businesses
 - Expect more focus on opportunities in Asia

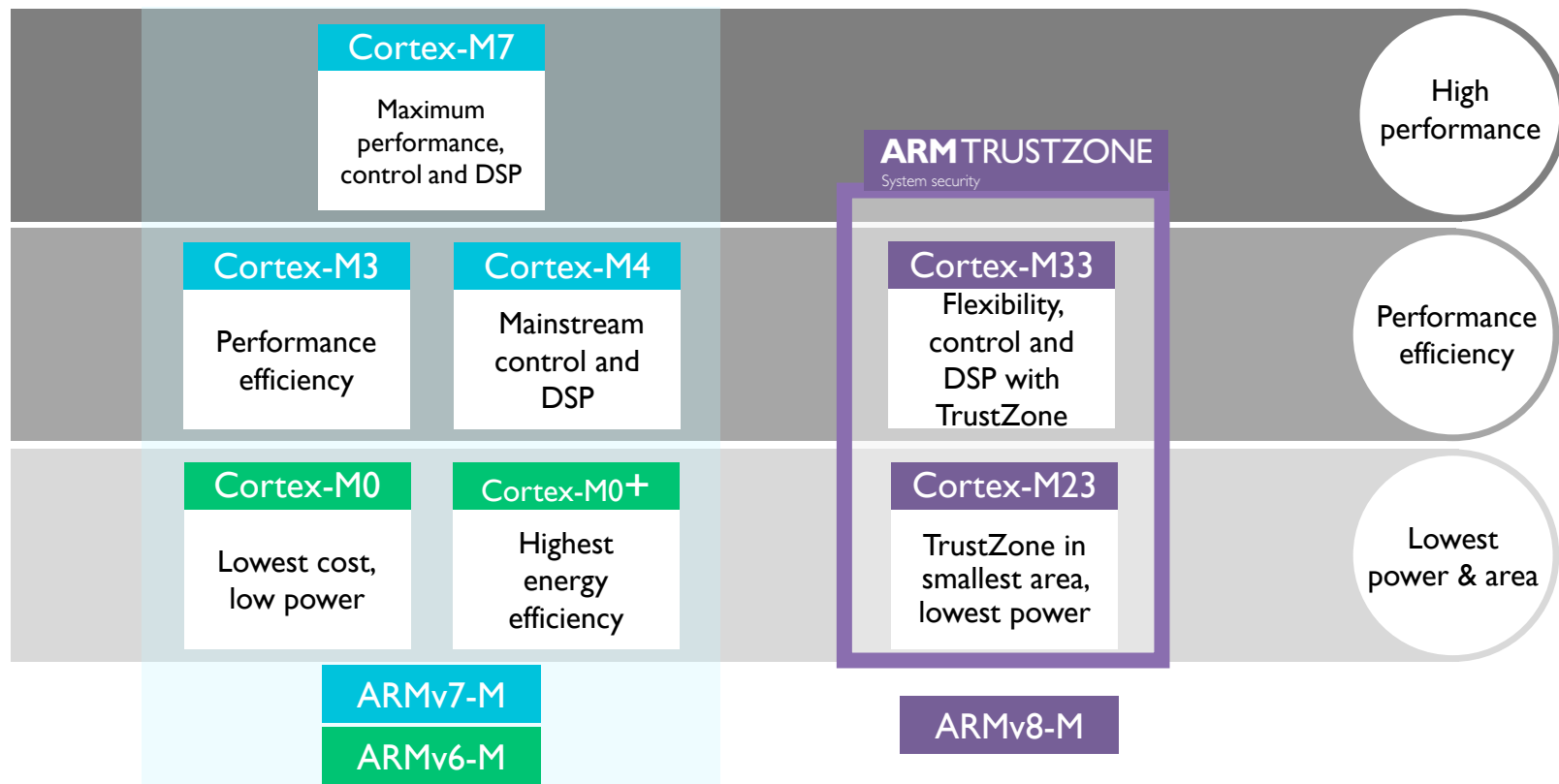


Masayoshi Son, CEO of Softbank

Agenda

- Welcome & CMSIS Overview, Status, Plans
- CMSIS-Pack / Driver enhancements
- CMSIS-RTOS2: real-time operating system - Status
- CMSIS-Zone: management and partitioning of complex systems
- Summary and discussion

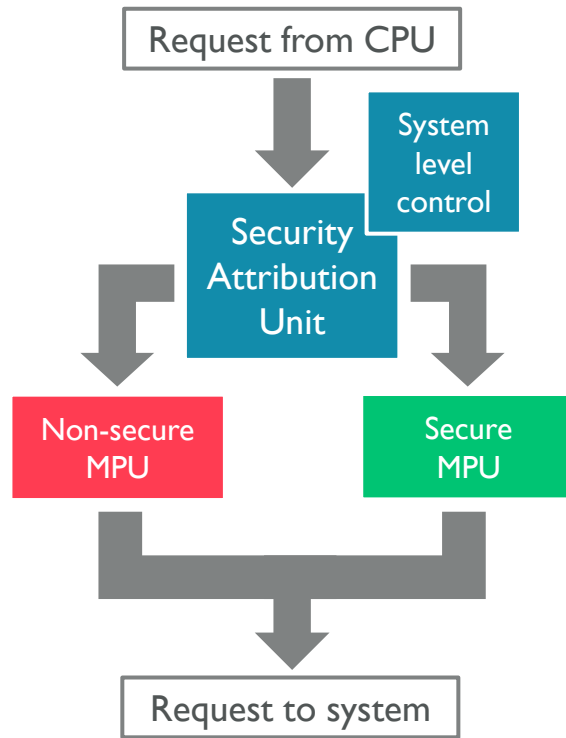
Bringing TrustZone to the Cortex-M family



Security defined by memory map

All transactions from core and debugger are checked

- All addresses are either secure or non-secure
- Policing managed by Secure 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 (NS) attribute based on address
 - Non-secure access to secure address → memory fault



Cortex Microcontroller Software Interface Standard (CMSIS)

Vendor-independent Standard for hardware manufacturers and tool vendors

Software Layers for Cortex-A / M processor based devices

- CMSIS-Core-M API for Cortex-M processor and core peripherals
- **CMSIS-Core-A** API for Cortex-A single-core processors
- CMSIS-DSP DSP Math Library with more than 60 functions
- CMSIS-RTOS API for RTOS integration
- CMSIS-Driver API for peripheral driver interfaces

CMSIS Version 5.1.0 will address hybrid devices based on Cortex-A / M

Infra-Structure for Cortex-A / R / M processor based devices

- CMSIS-SVD XML system view description for peripheral debugging
- CMSIS-DAP Firmware Debug Units to access the Debug Access Port
- CMSIS-Pack XML description for software components, device parameters, board support
- **CMSIS-Zone** Work-In-Progress: management for complex systems

www.arm.com/cmsis

Cortex-M Series

- > Cortex-M7 Processor
- > Cortex-M4 Processor
- > Cortex-M3 Processor
- > Cortex-M1 Processor
- > Cortex-M0+ Processor
- > Cortex-M0 Processor
- > CMSDK
- > **CMSIS**

CMSIS5 Status – What we said last year and what we delivered

Released in October 2016 → updated 5.0.1 in Januar 2017

No new components → Focus on Improvements & Further Industry Adoption!

- License change to **Apache 2.0** to enable contributions from 3rd parties
- Public development using GitHub: https://github.com/ARM-software/CMSIS_5
- Add support for ARMv8-M Architecture (Cortex-M23 and Cortex-M23)
- Improvements for Cortex-A / M hybrid devices (focus on Cortex-M interaction) – in progress

CMSIS-RTOS2 API and RTX reference implementation with several enhancements:

- Dynamic object creation, Flag events, C and **C++** API, additional thread and timer functions
- Secure and Non-Secure support, multi-processor support

CMSIS-Pack

- Additions for **generic example, project templates**, multiple download portals
- Adoption of IAR Flash Loader technology – in preparation

CMSIS enhancements in the works

- CMSIS-Core-A: Cortex-A processor support
- CMSIS-RTOS: RTX5 for Cortex-A
- CMSIS-DAP: extended trace support

CMSIS 5.1.0 release planned for May 2017

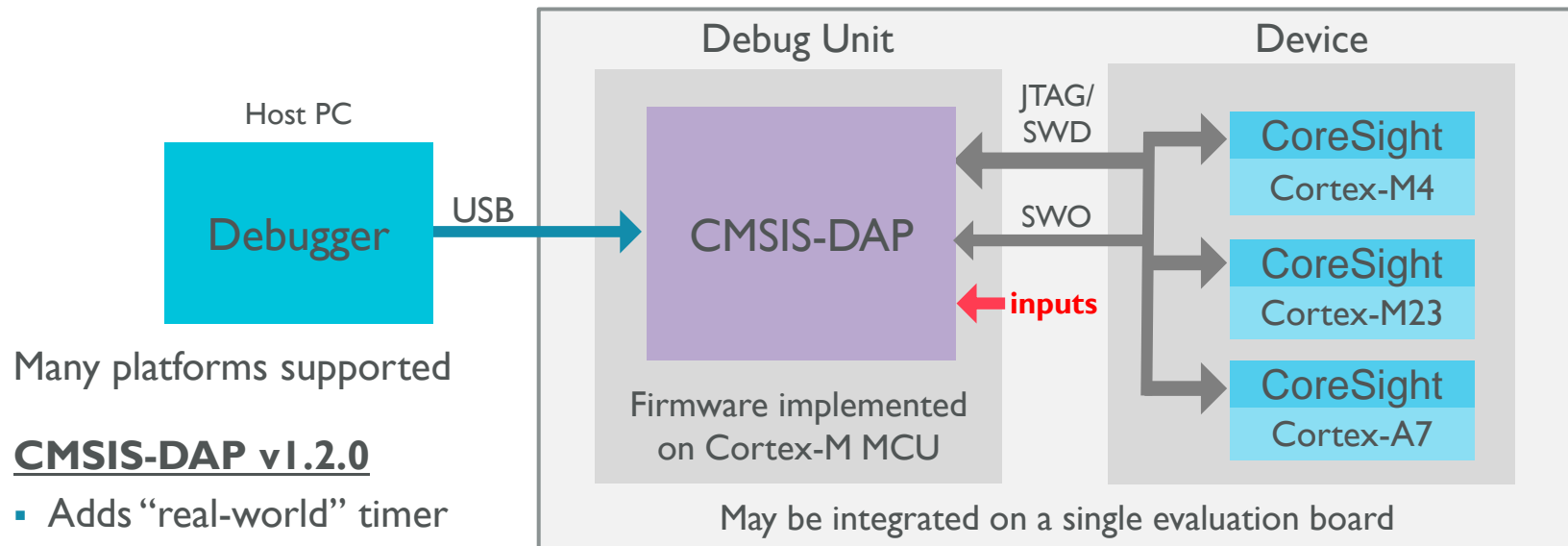
CMSIS-DAP 1.2.0 – Preview & request for feedback

Introduces trace recording for custom Performance Counters, for example:

- Power measurement (U, I) from external A/D converters
- Capture performance parameters from a memory system (wait states)
- Record data transfer parameters of an RF interface

Adds timer to the Debug Unit for synchronization with external world

CMSIS-DAP: Overview + Enhancements (v1.2.0)

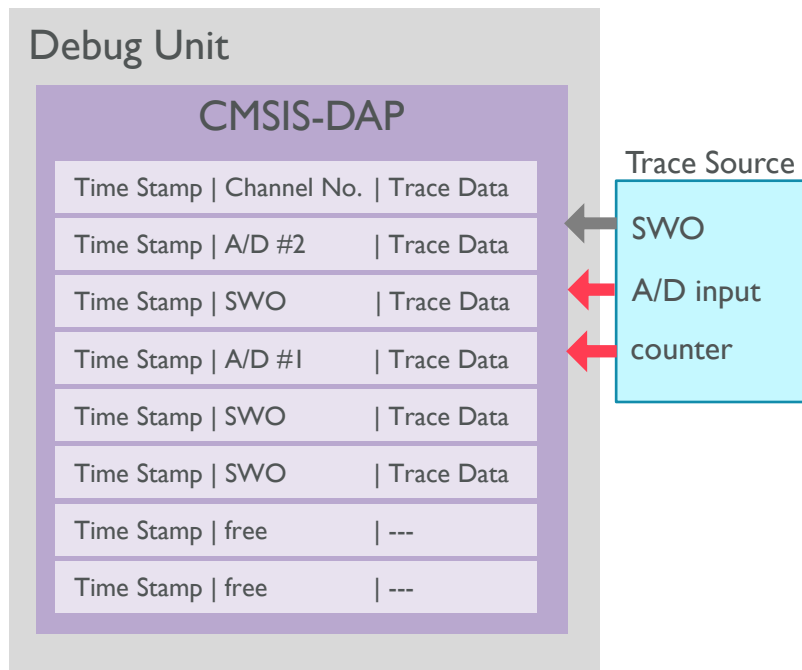


Many platforms supported

CMSIS-DAP v1.2.0

- Adds “real-world” timer to the Debug Unit
- Introduces trace recording for custom Performance Counters (**inputs**), for example:
 - Power measurement (U, I) from A/D converters
 - Performance parameters from a external system (i.e. wait states)
 - Transfer parameters of an RF interface

CMSIS-DAP v1.2.0 trace data management



- Trace sources are recorded in blocks
 - Block size 512 bytes ... 2 Kbytes
 - Includes Time Stamp and Channel No.
- Trace communication optionally with data difference compression
- SWO trace is handled in same way
 - Improves SWO trace performance

CMSIS plans for 2017

- CMSIS-Pack: generic template / project format
- CMSIS-Zone: management of complex system
- Multicore support for M+M and A+M systems

CMSIS-Pack & CMSIS-Driver

Status



Joachim Krech
Director of Engineering, MCU Tools

CMSIS - Partner Meeting, Embedded World
14. March 2017

▼ CMSIS-Pack

Revision History of CMSIS-Pack

- ▶ Create Software Packs
- ▶ Pack with Software Components
- ▶ Pack with Device Support
- ▶ Pack with Board Support
- ▶ Pack Example **Tutorials**
- ▶ Utilities for Creating Packs
- ▶ Publish a Pack
- ▶ Pack Description (*.PDSC) Format
- ▶ Configuration Wizard Annotations
- ▶ Flash Programming Algorithms

CMSIS-Pack Documentation

CMSIS-Pack describes a delivery mechanism for software components, device parameters, and evaluation board support. The XML-based package description (PDSC) file describes the content of a **Software Pack** (file collection) that includes:

- Source code, header files, and software libraries
- Documentation and source code templates
- Device parameters along with startup code and programming algorithms
- Example projects

Software Pack Use Cases

Variant	Device Family Pack	CMSIS Pack	Middleware Pack	Board Support Pack	In-house Software Pack
Source	Silicon Vendor, Tool Vendor	ARM	Silicon Vendor, Tool Vendor, 3rd Party	Board Vendor	Tool User

CMSIS-Pack: enhancements (since 2016 & in discussion)

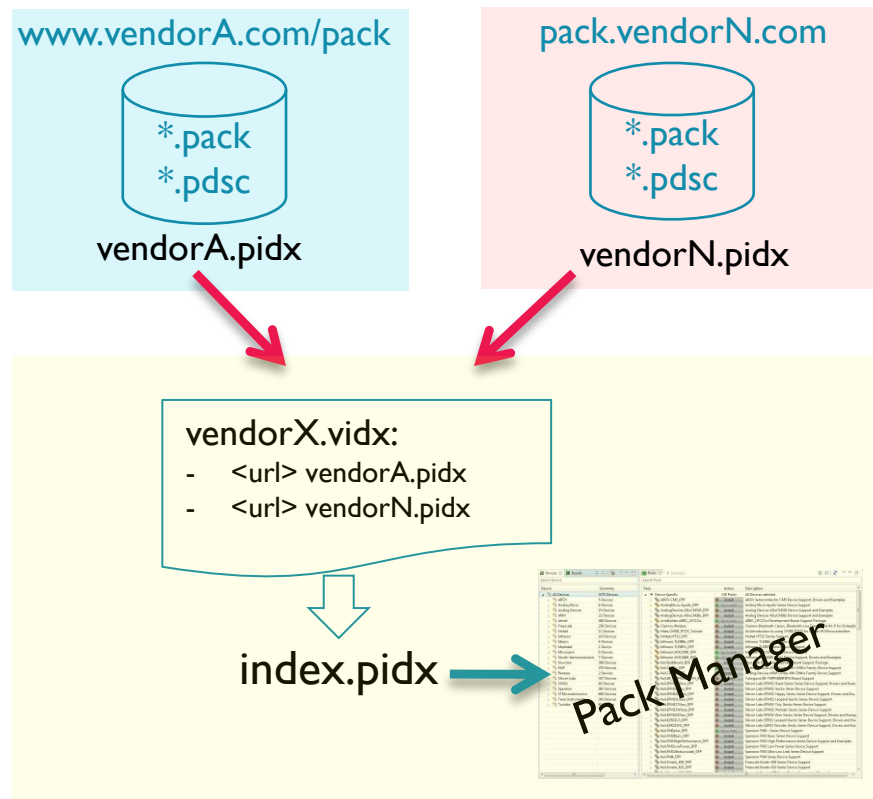
- `<requirements>` element allows to specify
 - Packs that should be used in combination with this pack
 - Compiler versions required for this pack
 - Programming languages that are required to compile the software
- CMSIS-Pack Index Files: allow to implement independent download portals.
- Generators (*.gpdsc) consistently implemented in MDK and Eclipse Pack management

In discussion:

- RTE Inventory that reflects current component selection back to a generator
- Import folder structure that contains multiple Software Packs
 - Simplifies transition from IDEs that do not support pack concept

CMSIS-Pack Index Files: Standardization for Download Portals

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



Software Pack vendors publish:

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

Tool vendors combine this into:

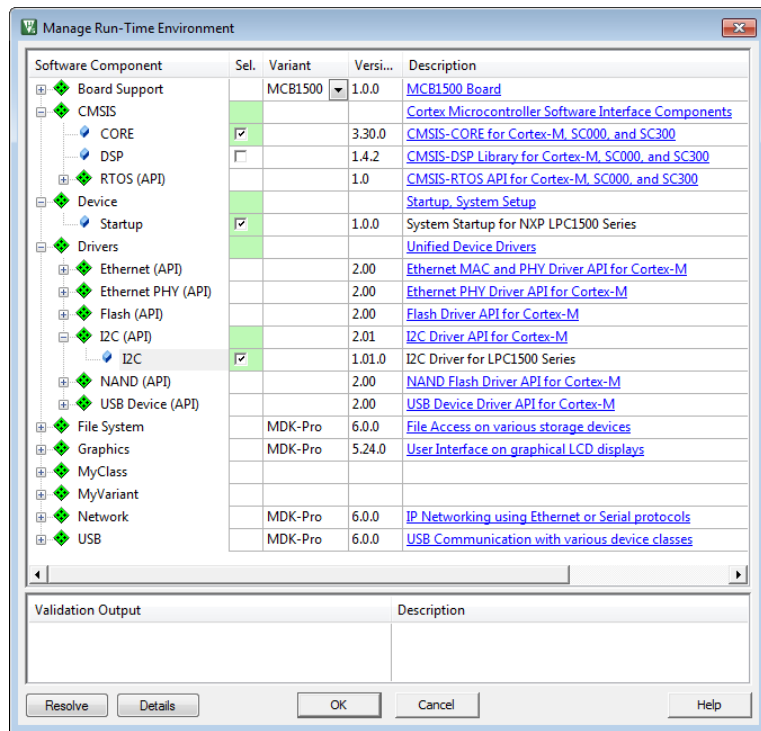
- **VendorX.vidx:** Index file that collects software pack vendors
- **Index.pidx:** Index file that lists all available packs

Benefits

- Software Pack vendors can update and add packs. This gets automatically distributed to tools and web.

CMSIS-Pack is Designed for Tools and Web Portals

Information in Packs is Shown in Tools and on Web Pages

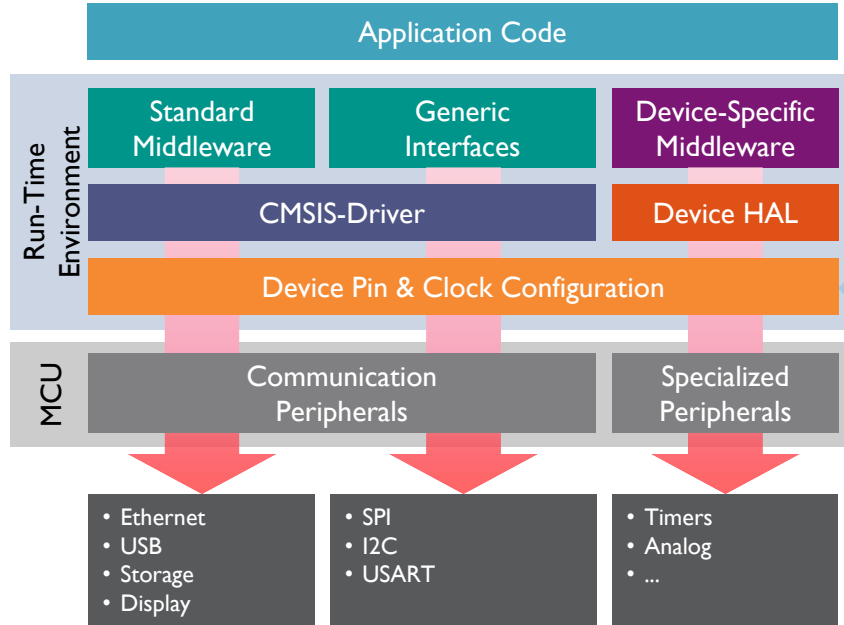


Companies that develop and publish Software Packs

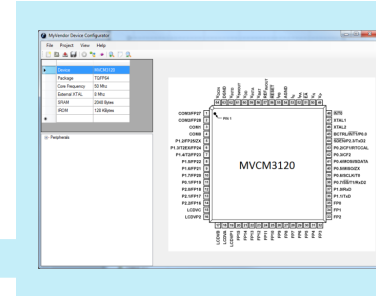
Information from www.keil.com/pack/keil.vidx

- Abov
- AmbiqMicro
- Analog Devices
- Cypress
- GigaDevice
- Holtek
- Infineon
- Nordic Semiconductor
- Nuvoton
- NXP
- MediaTek
- Microchip (Atmel)
- Microsemi
- Mindmotion
- SONiX
- Texas Instruments
- Zilog
- ARM
- Clarinox
- Hitex
- Huawei
- Micrium
- Oryx-Embedded
- RealTimeLogic

CMSIS-Driver and CMSIS-Pack - focus on refinement



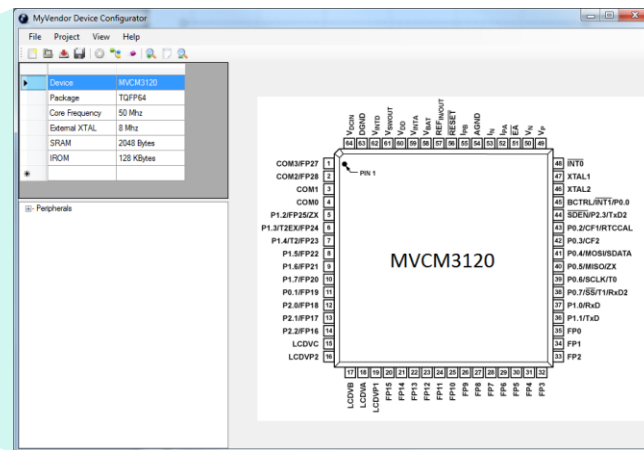
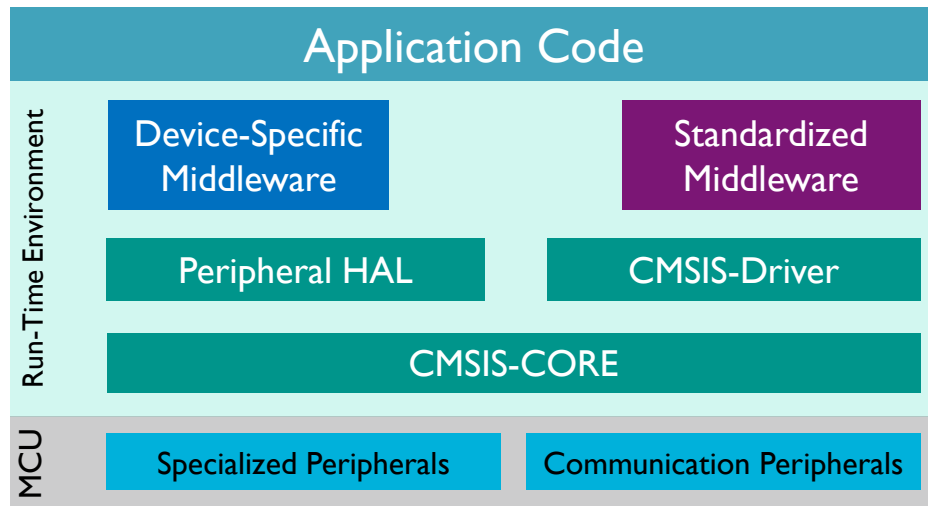
- GPDSC - Simplify Configuration



Generated Software Pack
*.GPDSC

- IDE Independent Project Examples & Templates
 - *.cpdsc file with <create> element
- Eclipse integration
- Multiple Pack Download Portals

Generator Interface to Simplify Device Configuration



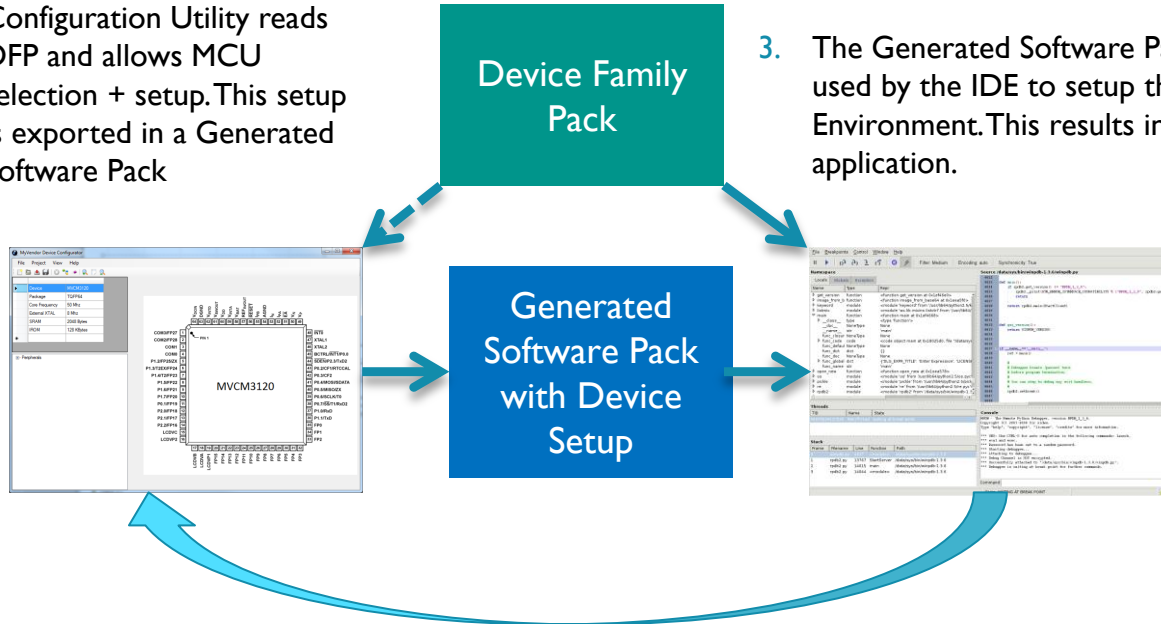
A Device Configuration Utility is simplifying:

- Microcontrollers need typically configuration for:
 - Clock System, I/O pins for peripherals, Interrupt & DMA usage
- A Generated Software Pack (*.gpdsc) is designed to interface with utilities
 - Exchange configuration files and generated source code

Workflow

- I. Device Family Pack (DFP) contains information for IDE and the Configuration Utility (element <environment>). This approach ensures consistency of the generated Pack with the DFP

2. Configuration Utility reads DFP and allows MCU selection + setup. This setup is exported in a Generated Software Pack

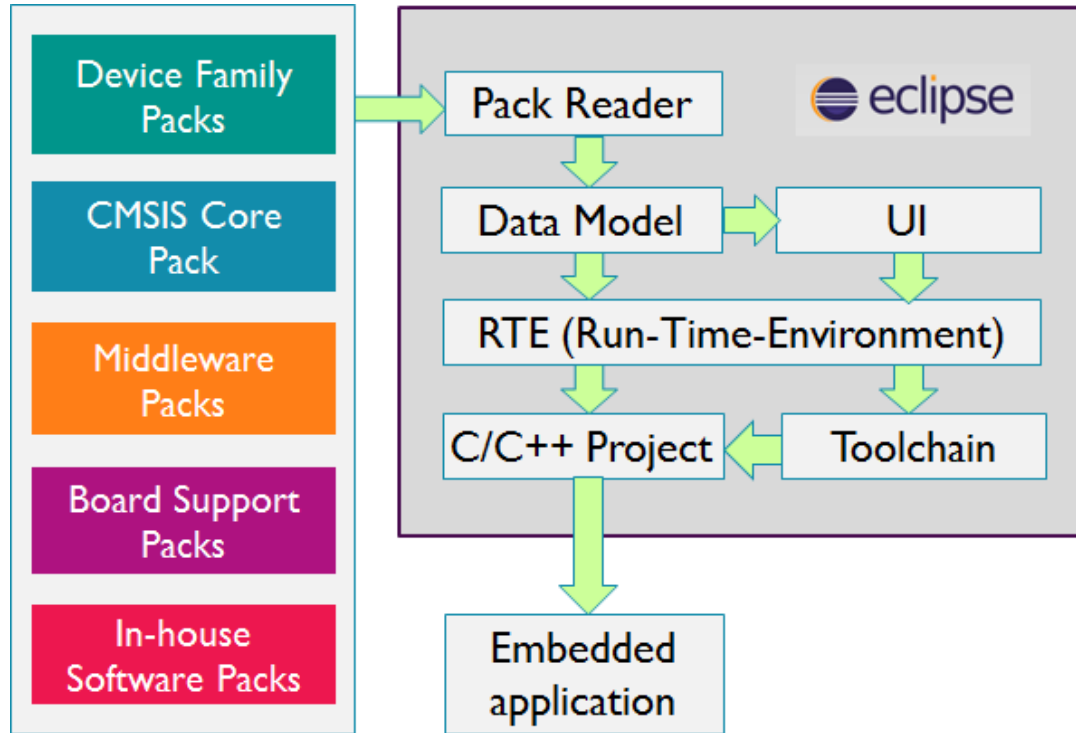


3. The Generated Software Pack along with the DFP is used by the IDE to setup the project and the Run-Time Environment. This results in the basic configuration for the user application.

- 4. User can add more software components in the IDE

5. The device setup may be altered using the Configuration Utility. The changes are imported to the IDE and reflected in the project

CMSIS-Pack for Eclipse – Overview



Plug-in maintained by ARM

Basis for many IDEs

- ARM DS-MDK, DS-5
- Atollic Studio
- IAR EW-ARM

GPDSC Support allows connection to code generators

Demo

The screenshot displays the Eclipse IDE interface for configuring an STM32 project. The main window is titled "C/C++ - STM32/STM32.rteconfig - Eclipse Platform". The left sidebar shows the "Project Explorer" with a tree view of the project files, including "Includes", "RTE", "CMSIS", "Device", and "main.c". The central pane shows the "Components*" table, which lists various software components and their configurations.

Software Components	Sel.	Variant	Vendor	Version	Description
STM32F407IEHx	<input checked="" type="checkbox"/>		STMicroelectr		
Board Support	<input checked="" type="checkbox"/>	MCBSTM32F400	Keil	2.0.0	
CMSIS	<input checked="" type="checkbox"/>		ARM	4.1.0	
CORE	<input checked="" type="checkbox"/>		ARM	14.5	
DSP	<input checked="" type="checkbox"/>		ARM	1.0	
RTOS (API)	<input checked="" type="checkbox"/>		ARM	4.78.0	
Keil RTX	<input checked="" type="checkbox"/>				
CMSIS Driver	<input checked="" type="checkbox"/>				
Compiler	<input checked="" type="checkbox"/>				
Device	<input checked="" type="checkbox"/>				
Startup	<input type="checkbox"/>		Keil	2.3.1	System

Below the table, the "Validation Output" section shows a warning: "ARM::CMSIS.RTOS.Keil RTX require Cclass='Device', Cgroup='Startup'".

The "Select Device" dialog is open, showing the following information:

- Device: LPC4357:Cortex-M4
- Vendor: NXP
- Pack: Keil.LPC4300_DFP.2.5.0
- URL: <http://www.keil.com/dd2/nxp/lpc4357>
- CPU: ARM Cortex-M4
- Clock: 204 MHz
- Memory: 96 kB RAM, 1 MB ROM
- FPU: single precision
- Endian: Little-endian

The dialog also displays a tree view of the device's components, including "LPC432x", "LPC433x", "LPC435x", "LPC4350", "LPC4353", "LPC4357", "LPC4357:Cortex-M0", "LPC4357:Cortex-M4", "LPC437x", "Spansion", and "STMicroelectronics".

CMSIS-Pack: what we promised – slides from 2013

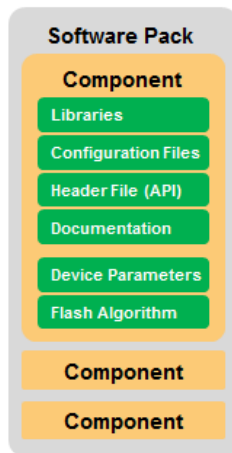
Questions when using external source code

- Usage of files (source code, header, library) is unclear
 - Will my Compiler and Tool Environment work?
 - Where is the documentation to the software component?
 - What other requirements (i.e. libraries or RTOS) does this software have?
 - Will the source code run on my target hardware?
 - Does the source code need adaptations or configuration?
- Project Maintenance after files are integrated
 - Where did I get the source code from; who to contact for support?
 - What is the version of that source code?
 - What do I need to change when I update the files?
- What are the License conditions of the code
 - Can I use this files in my project?

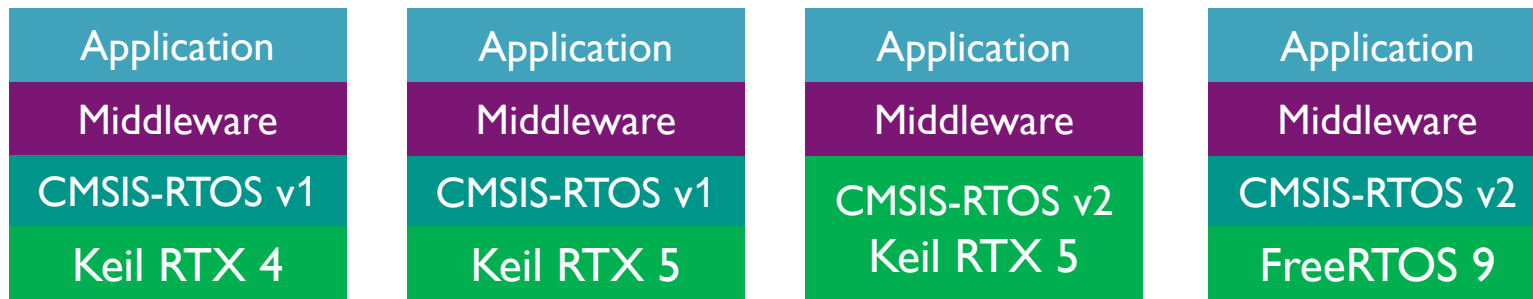
Packing Software is a Solution



- All software components are delivered in a CMSIS-Pack that is easy to install (like a library)
- A package description file (PDSC) contains
 - Supplier information
 - Download URL
 - License
 - Release version
 - Usage of source code and libraries files for:
 - Specific processors
 - Specific microcontroller families and devices
 - Tool Chains
 - Other components that are required or related



Usage example: maintain an RTOS application



Scenario:

Maintain a project developed a year ago

Tasks for the developer:

- Upgrade from RTX4 to RTX5
- Update RTX5
- Change RTOS kernel to FreeRTOS

Benefits:

- Components are easy to identify
- Documentation is accessible
- Incompatible configuration files are spotted
- No problem to identify processor layers

CMSIS-RTOS2

Status

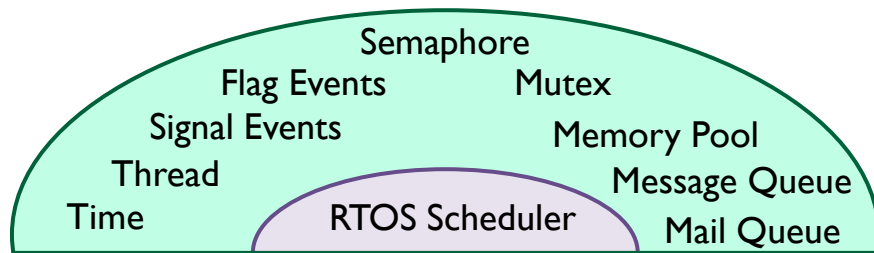


Christopher Seidl
Technical Marketing Manager

CMSIS Partner Meeting, Embedded World
14. March 2017

CMSIS-RTOS v2 API – Enhancements

Common API for Real-Time Operating System, compatible to Version 1



Pre-emptive thread scheduling with priorities

CMSIS-RTOS v2 addresses:

- Enhanced C API (feedback from RTOS v1)
- ARMv8-M Security domain support

Coming during 2017:

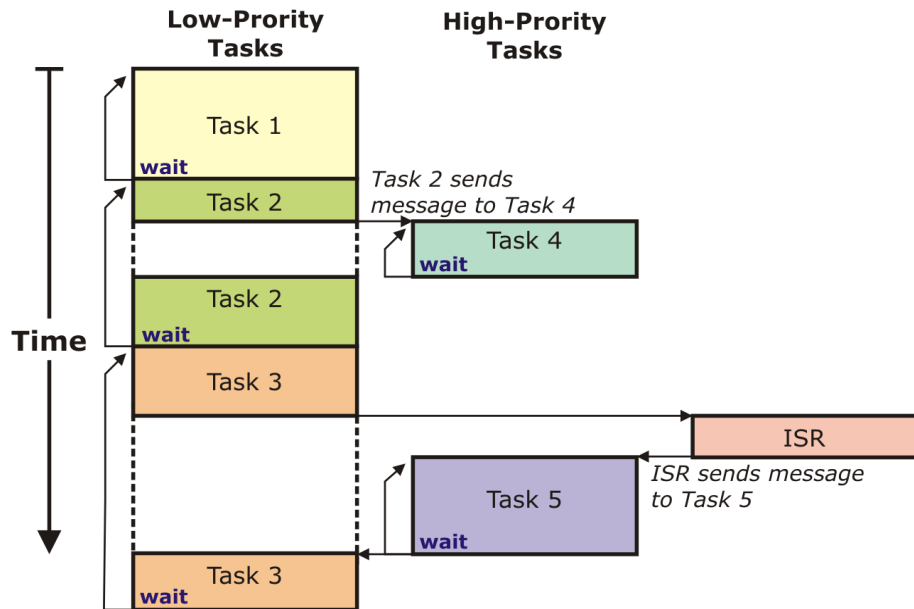
- C++14 Integration
- Multi-processor message passing support

Version 2 Enhancements / additions:

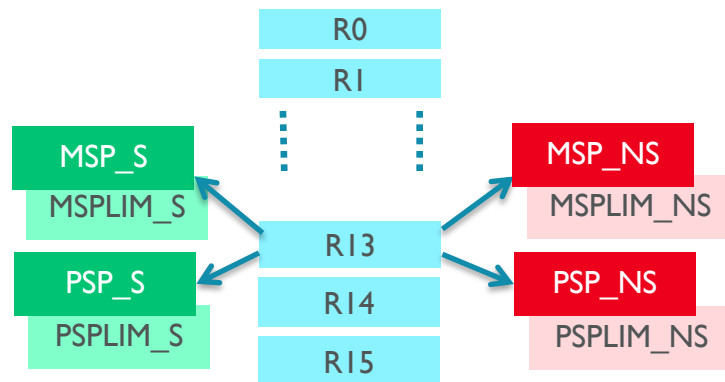
- More thread priorities
- Dynamic Object creation
 - Initializing osXxxxDef definitions
 - Multiple instances Mutex & Semaphore
- External reference to object definitions
- osKernelTime, osKernelStop
- osThreadSuspend, osThreadResume
- osPoolDelete
- osMessageCount, ...Reset, ...Delete
- osMailCount, osMailReset, osMailDelete
- osFlagXxx global event flags

RTOS Scheduling on v8-M

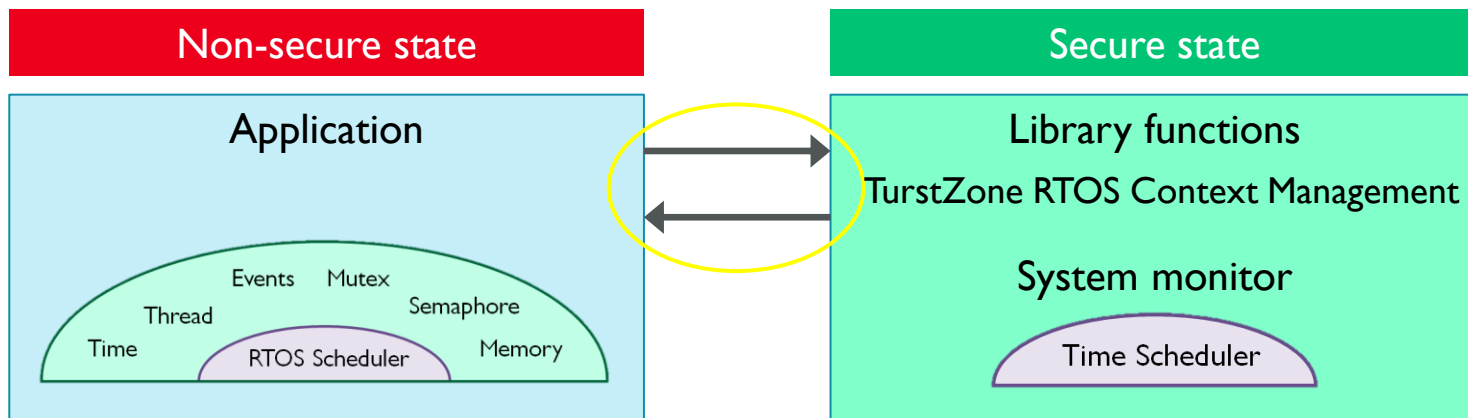
- Thread or task context requires to change stack



- ARMv8-M provides additional stack pointers for Secure State



API for RTOS interface to secure state: CMSIS



- RTOS running in non-secure state: RTOS functionality available to non-secure and secure software
- Full-featured RTOS for non-secure application
 - Supports function calls to secure state
 - Callback events from secure state
- CMSIS-CORE provides TrustZone extensions:
 - RTOS Context Management for secure state
 - Example projects that show system recovery
- Secure state provide/s data and firmware protection

TZ_context.h: RTOS Thread Context Management

Initialize secure context memory system

```
int32_t TZ_InitContextSystem S (void);           returns 0: success, != 0 error code
```

Allocate Memory for Secure Process Stack Management (called on osThreadCreate)

```
int32_t TZ AllocModuleContext S (module id);           returns >=0 context id, < 0: no memory
```

Free Memory for Secure Process Stack Management (called on osThreadTerminate)

```
int32 t TZ FreeModuleContex S (context id);           returns 0: success, != 0 error code
```

Load Secure Context (set **PSP** [and **PSP_LIM_S**]) (called on thread context switch)

```
int32_t TZ_LoadContext S (context id);           returns 0: success, != 0 error code
```

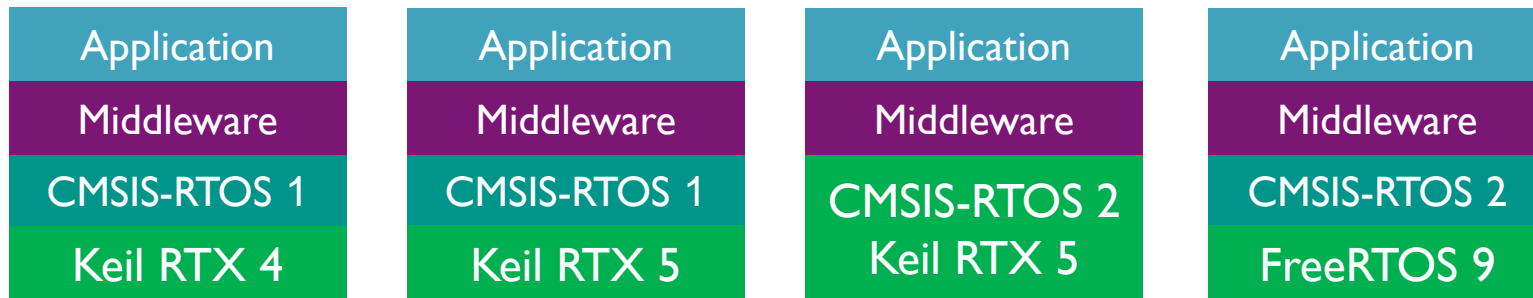
Store Secure Context (save current PSP) (called on thread context switch)

```
int32_t TZ_StoreContext S (uint32_t context id);          returns 0: success, != 0 error code
```

http://arm-software.github.io/CMSIS_5/Core/html/using_TrustZone_pg.html#RTOS_TrustZone

CMSIS-RTOS: RTX & FreeRTOS implementations

Choices for Developers – with compatible API that eases migration

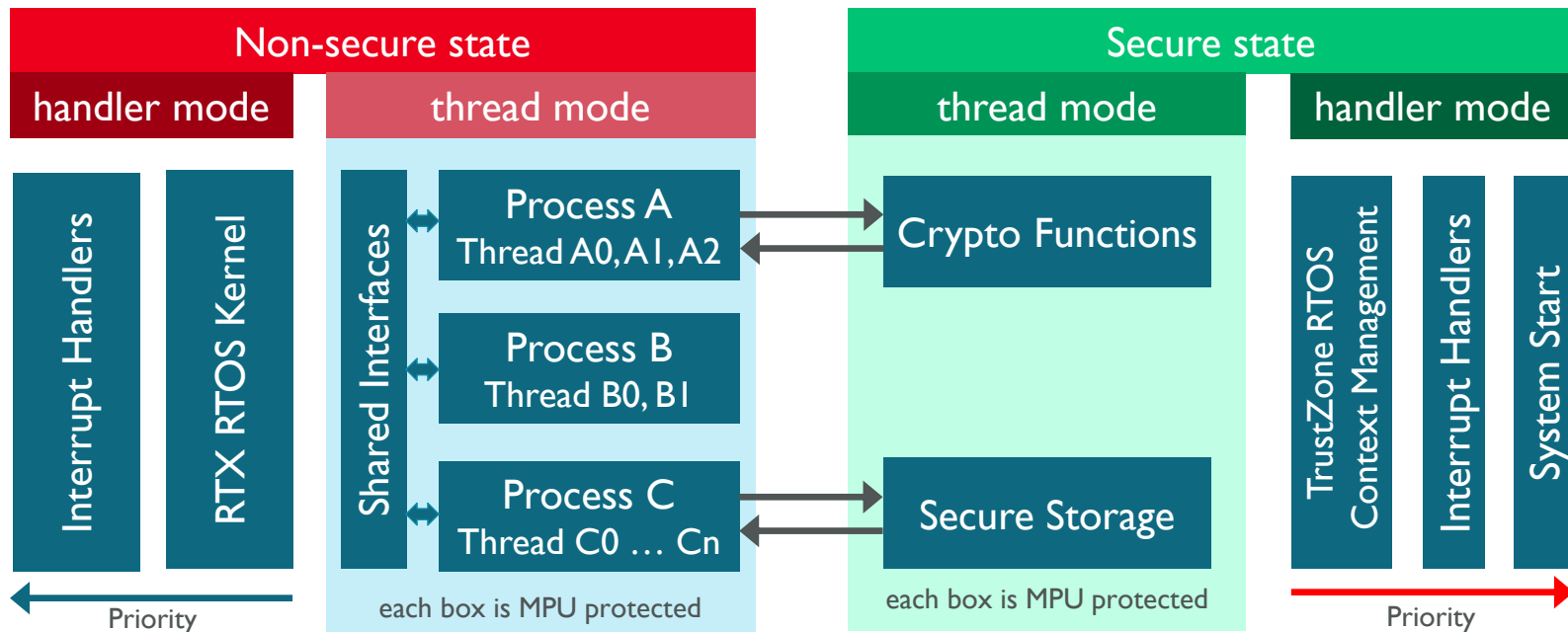


Benefits:

- Allows to choose between different RTOS versions / variants
- Allows to use native RTOS API or CMSIS-RTOS API depending on the application requirements
- Using CMSIS-Pack it is easy to upgrade or change the RTOS kernel

CMSIS-RTOS2 – MPU Extension for RTX

Add Memory Protection Unit to extend security to process/thread execution



- Interrupt Handlers are time deterministic: RTX RTOS Kernel never blocks interrupts
 - Secure state interrupts may have **escalated** priority levels

CMSIS-Zone

Proposal for Managing complex Embedded Systems



Jonatan Antoni
CMSIS Technical Lead

CMSIS - Partner Meeting, Embedded World
14. February 2017

Embedded/IoT Challenges: Performance/Security

Addressed by multi-core processors with security executions

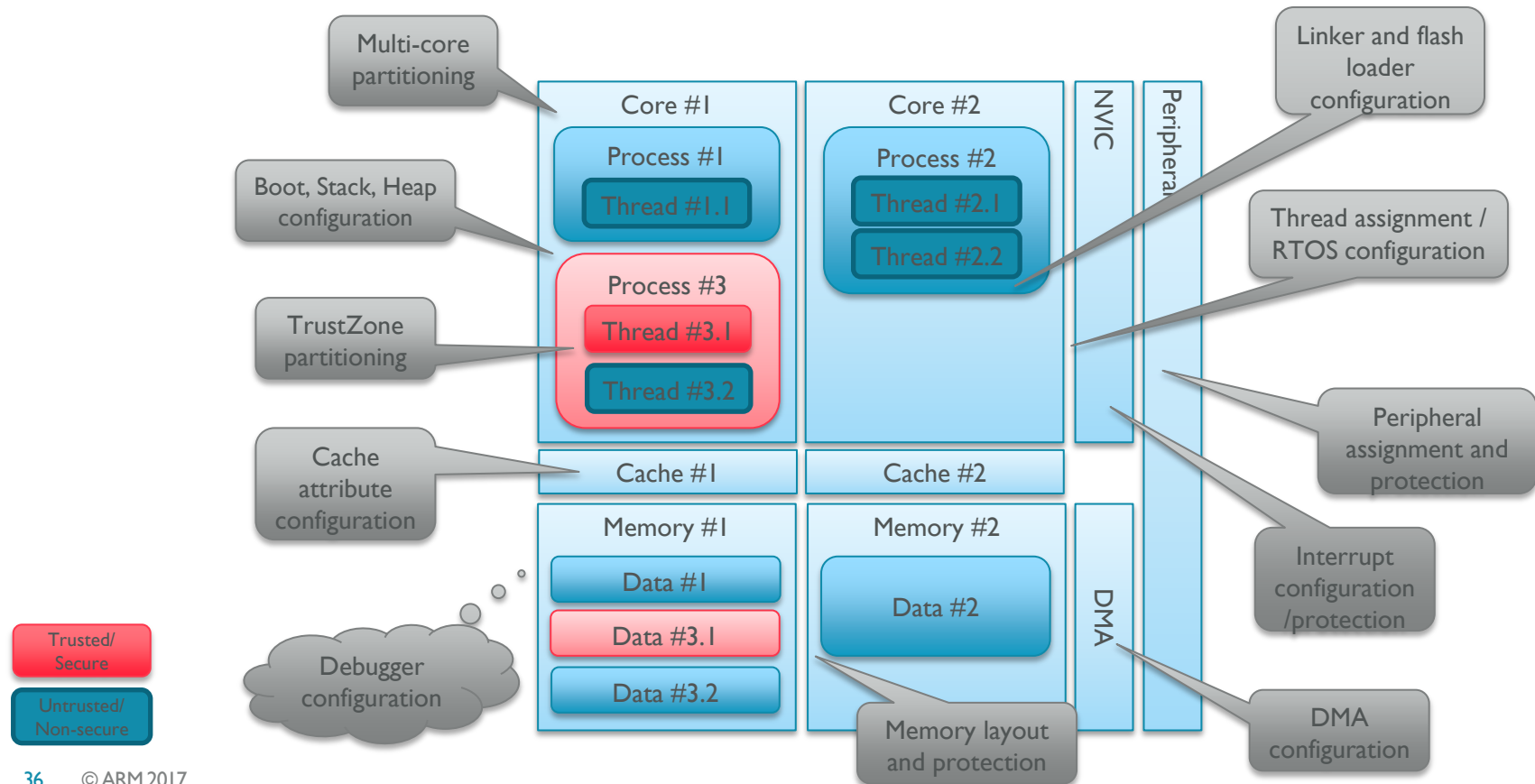
Developers have to face

- Multi-core processor systems
- Shared memory and peripherals
- Multi-master (DMA) and caches
- Security Extensions, aka TrustZone, and additional protection IP
- (Refined power management capabilities)

This results in

- complex project setup for software development.
- consistent system configuration which is difficult to maintain.

Embedded/IoT Challenges: Performance/Security



CMSIS-Zone: Goals and Objectives

Developers need utilities to handle the complexity.

- According to the available SoC components...
 - Cortex-M0 up to Cortex-M33, possibly bare metal Cortex-A types
 - Caches, MPU (MMU), ARMv8-M Security Extensions
 - Multi core SMP/AMP/HMP¹ systems, shared memory, RTOS
- ... all those need to be configured consistently.
 - Reduce the repetition of system settings.
 - Assure well configured SoC components, especially where features interfere.
 - Prevent security threats due to misconfiguration.
- This leads to the need for a configuration tooling.
 - System level configuration
 - Checkers and generators
 - Easy accessible and usable

¹ Symmetric/asymmetric/heterogeneous multi-processing

CMSIS-Zone: Scope

Standardized methods for system level configurations

- Define a data format that allows
 - Specification of memory zones for complex embedded systems
 - Transformation into project setup and device setup/configuration
- Prototype implementation for Memory Zone management UI
 - Extended version of the CMSIS-Pack Eclipse Plug-ins
- Specify standardized methods for
 - Compiler, Linker controls that define memory zones
 - CMSIS functions to control MPU, SAU and similar memory configuration hardware

Request for Feedback

- Please get involved -

Get Involved with us on CMSIS

https://github.com/ARM-software/CMSIS_5

- Reflects our current development status [Branch: **develop**]
- Gives access to current specifications (work in progress)
- Allows for feedback via **Issues**

Review our current and coming enhancements:

- CMSIS-Pack – Project Description (*.CPDSC) Format
- CMSIS-Pack – CMSIS-Pack Index Files
- CMSIS-Pack – Debug Sequences and System Definition Files (coming soon) for multi-core
- CMSIS-Core-A – Cortex-A base software framework
- CMSIS-DAP – Enhancements for Performance Counters (Energy Measurement)
- CMSIS-Zone – Stay tuned

ARM

World's No. 1 Embedded Ecosystem

Thank you.....now it's time for drinks and side discussions...

The trademarks featured in this presentation are registered and/or unregistered trademarks of ARM Limited (or its subsidiaries) in the EU and/or elsewhere. All rights reserved. All other marks featured may be trademarks of their respective owners.

Copyright © 2016 ARM Limited