Enhanced Project Management for Embedded C/C++ Programming using Software Components

Evgueni Driouk
Principal Software Engineer
MCU Development Tools
Outline

- Introduction
  - Challenges of embedded software development
  - What is CMISIS and CMSIS-Packs

- CMSIS-Pack Eclipse Plug-in
  - Demo
  - Architecture: plug-ins and their dependencies, major types, data flow
  - RTE (Run Time Environment) Model: component filtering and resolving dependencies
  - CDT integration: creating and updating project, managing toolchain settings
  - Accessing RTE configuration information: device properties, selected components, files

- Conclusions
Embedded software development challenges

- **Growing complexity of embedded devices**
  - availability of device and board information
  - startup code
  - device drivers
  - flash programming algorithms
  - debug awareness

- **Software complexity and flexibility**
  - possibility to reuse middleware and application software components

- **Application portability across different devices from different vendors**
  - unification in software interfaces to processors and peripherals
What is CMSIS?

- The ARM Cortex Microcontroller Software Interface Standard, a vendor-independent standard for silicon partners, tool vendors and end users
- Establishes a software foundation with a set of specifications, libraries, and interfaces
- Enables consistent software layers and device support across a wide range of development tools and microcontrollers.
CMSIS-Pack: delivery mechanism for SW components

- CMSIS-Pack specifies a way to deliver software components and device information in a structured manner
- Designed to be versatile and usable for a wide range of use cases

The pack deliverables include:
- Source code, header files, software libraries
- Documentation, source code templates and examples
- Device parameters along with startup code and flash programming algorithms
## CMISIS-Pack: description file content

| Components       | • Uniquely identified by its taxonomy: class, group, sub-group, and variant  
|                  | • List of files that make up a software component |
| Conditions       | • Describes dependencies on device, processor, tools, or other components  
|                  | • Components are only available if their conditions resolve to fulfilled |
| Devices          | • Hierarchical list of devices supported by the Pack (family/subFamily/device)  
|                  | • Properties: processor, memory, debug info, books, features |
| Boards           | • Defines development boards  
|                  | • Information is used in tools but also on web pages |
| Examples         | • Project examples acting as usage references of software components  
|                  | • References to boards identify the targeted hardware |
Faster device support with CMSIS-Pack

Traditional Device Support

Device Vendor (SiP)
- Create CMSIS Files

Tool Vendor A
- Tool chain integration
- Verification

Tool Vendor C
- Tool chain integration
- Verification

Tool Vendor B

Customers
- Use Published Devices

Verification failed

Device Support using CMSIS-Pack

Device Vendor (SiP)
- Create CMSIS Files
- Create Software Pack
- In-house Verification
- Flexible Publishing

Development Tools
- Automatic Integration
- Customers can use right away
- Optional Distribution to Early Customers only

Early verification during chip design phase → better overall quality

One Pack for multiple tool chains
Flexible distribution to customers → faster development start
CMSIS-Pack Eclipse Plug-in

- Reference implementation of CMSIS-Pack support in Eclipse environment.
  - open source under Eclipse Public License 1.0
- Implements the fundamentals to access the Pack information and resources:
  - parses installed CMSIS-Packs
  - creates and manages Run-Time Environment configurations
- Provides CDT integration:
  - creates and dynamically manages C/C++ projects
  - updates toolchain settings
- Can be re-used by the ARM eco-system in tools such as:
  - development environments, configuration utilities
Architecture: modules

com.arm.cmsis.pack
- Pack data model
  - PDSC Parser
  - Pack Manager
- Pack data reference info
- RTE model & controller
  - Device tree
  - Component model

com.arm.cmsis.pack.common
- Generic types
- Constants
- Utilities

com.arm.cmsis.pack.project
- RTE project management
- CDT interface: template, hooks
- UI: decorator, properties

com.arm.cmsis.pack.build.
- Toolchain adapters
  - toolchain adapters

com.arm.cmsis.pack.ui
- Common widgets and views
- Console
- RTE configuration editor
Major data types

- Pack data model read from PDSC (Pack Description) files:
  - **ICpItem** – base interface for CMSIS-Pack elements, implements most of functionality
    - basics functions: `getTag()`, `getText()`, `getAttribute()`, `getParent()`, `getChildren()`
    - advanced functions: `getName()`, `getId()`, `getVendor()`, `getVersion()`, `getUrl()`, `getPack()`
    - derived interfaces: `ICpPack`, `ICpComponent`, `ICpFile`, `ICpDeviceItem`

- Reference information stored in `rteconfig` file:
  - **ICpItemInfo** – CMSIS-Pack element references, extends **ICpItem**
    - derived interfaces: `ICpPackInfo`, `ICpComponentInfo`, `ICpFileInfo`, `ICpDeviceInfo`

- RTE Model constructed from Pack data and reference info:
  - **IRteComponentItem** – component tree
  - **IRteDeviceItem** – device tree
  - **IRtePackItem** – pack tree
RTE (Run-Time-Environment) Model

- Filters components for selected device and toolchain
  - evaluates conditions using `ICpConditionContext` as strategy/visitor
- Aggregates components from different packs into single tree
  - according to taxonomy: class/group/sub-group
- Manages component selection via RTE controller
  - saves configuration as collection of `ICpComponentInfo` items
- Evaluates and resolves component dependencies
  - evaluates conditions using `IRteDependencySolver` as strategy/visitor
- Resolves previously saved component references
## CDT integration: Toolchain adapter

- Responsible for setting toolchain options according to selected device and components:
  - include paths, libraries, preprocessor defines
    - can be done generically, depending on `IOption.getValueType()`
  - CPU type, FPU type, endian
    - requires option base ID, for instance "com.arm.tool.c.compiler.option.targetcpu"
    - might require consistent update of several options
  - linker script / scatter file
    - requires option base ID
    - initial script could be generated from device information (optional)

- Any toolchain needs an adapter, even for the same compiler:
  - Use `com.arm.cmsis.pack.build.settings.ToolChainAdapter` extension point
  - Implement `IRteToolChainAdapter` or extend `RteToolChainAdapter`

- Toolchains supported by CMSIS-Pack Eclipse plug-in:
  - ARM Compiler 5 (ARM DS-5 built-in)
CDT integration: updating project

- Project update job is scheduled when:
  - project gets loaded or configuration file `${ProjName}.rteconfig` changes

- Component files are added to RTE/Cclass folders:
  - referenced as a link to pack location
   `${cmsis_pack_root}/ARM/CMSIS/4.3.0/CMSIS/RTOS/RTX/LIB/ARM/RTX_CM4.lib`
  - copied to project, _n suffix is used for multiple component instances
   `${workspace_loc}/${ProjName}/RTE/File System/FS_Config_MC_1.h`
  - unused files are removed

- `RTE_components.h` file is generated

- Linker script is generated on create and device change

- Toolchain adapter is called to update build settings
CDT integration: hooking into CDT startup

- Task: load C/C++ project, load RTE configuration, update project, update index
  - load Data Provider, then register Project Description Listener and Indexer Setup Participant
  - tricky: neither Index Manager nor Description Manager are available at Data Provider load point

- Solution: register listeners when indexer job gets scheduled
Accessing RTE configuration information

- IProject
  - IRteProjectManager
    - IRteProject
      - IRteConfiguration
        - IBuildSettings
  - Effective Properties:
    - processor
    - memory
    - features
    - flash algorithms
    - debug info
    - books
  - ICpConfigurationInfo
    - ICpDeviceInfo
      - ICpComponentInfo
        - ICpComponent
          - ICpFile
Conclusions

- CMSIS establishes a software foundation that enables consistent device support and software components reuse across an wide range of development tools and microcontrollers.

- CMSIS-Pack is a delivery mechanism for software components and device specifications.

- CMSIS-Pack plug-in is an open-source reference implementation of the CMSIS-Pack support for Eclipse environment.

- The plug-in enhances project management with CMSIS software components to accelerate Embedded Software Development.
Thank You!

- CMSIS-Pack Eclipse Plug-in:
  - https://github.com/ARM-software/cmsis-pack-eclipse

- Packs repository:

- CMSIS online specification and tutorials
  - http://www.keil.com/cmsis

- GNU ARM Eclipse Toolchain and Pack Manager
  - http://gnuarmeclipse.github.io/