Open-CMSIS-Pack

Technical Project Meeting 2022-01-25

This meeting is recorded !

Open-CMSIS Pack

Linaro
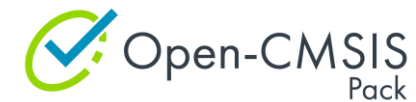
Agenda

- Welcome
- IDE Development Flow (outline)
- Project Manager Feedback
- Targeted Reference Example
- Wrap Up

Linaro

IDE Development Flow with Open-CMSIS-Pack tools

1. Install the required packs using cpackget to work with devices/boards/software components

2. IDE dialogs create *.yml files (when a project/solution is started)
3. IDE dialogs support selecting devices, boards, components and management of files/groups.
4. IDE drop-down selections allow to choose the target-type and build-type that a user wants to work with.

This will get first prototyped with Keil Studio Cloud – the open-source version will be derived from it.

- **In first VS Code deployment, *.yml file editing is supported by schema support in the editor.**

5. A change to *.yml files should trigger **projmgr** that generates the output described including the *.cprj files used to build the project. The error message synch back to the related file.

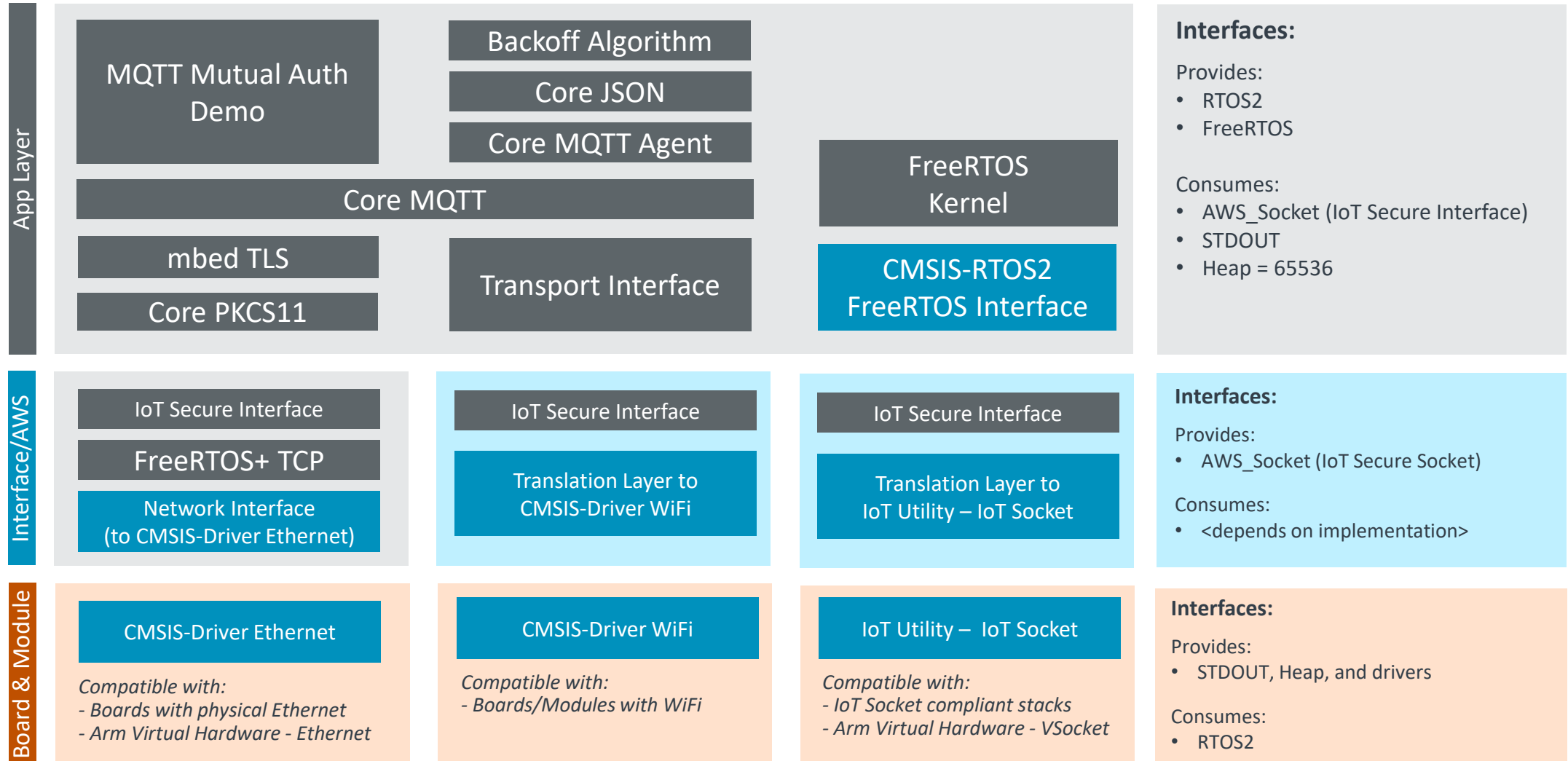6. On "Build" the *.cprj files are used by CBuild to generate CMake files and run the build process.

Suggest: we should get this working with VS Code to get a better feeling of the overall workflow.

CMSIS Project Manager – Proposal Review Feedback

Document Overview.md :

- https://github.com/Open-CMSIS-Pack/devtools/pull/77 - got a lot of feedback
- Prioritization of next development focus – suggest to handle each in separate issues.

    1. PLM of config files (proposal is included in specification)
        - Specify a preferred 3-way merge utility?
    2. RTE directory structure finalize
    3. Local copy of components (ST requirement)
    4. Handling of packs (CMSIS-Pack-Root, Repositories, Local packs, etc.)
        - Finalize definition cdefault.yml
    5. CMSIS-Zone integration (resource management)
    6. Execution Groups (or perhaps better execution phases)
    7. Generator support / extensions
    8. Layer interface definitions
    9. Board conditions
    10. Layers distributed in packs
    11. Vendor specific additions (ST requirement) devtools #83
    12. Multiple devices defined by a board
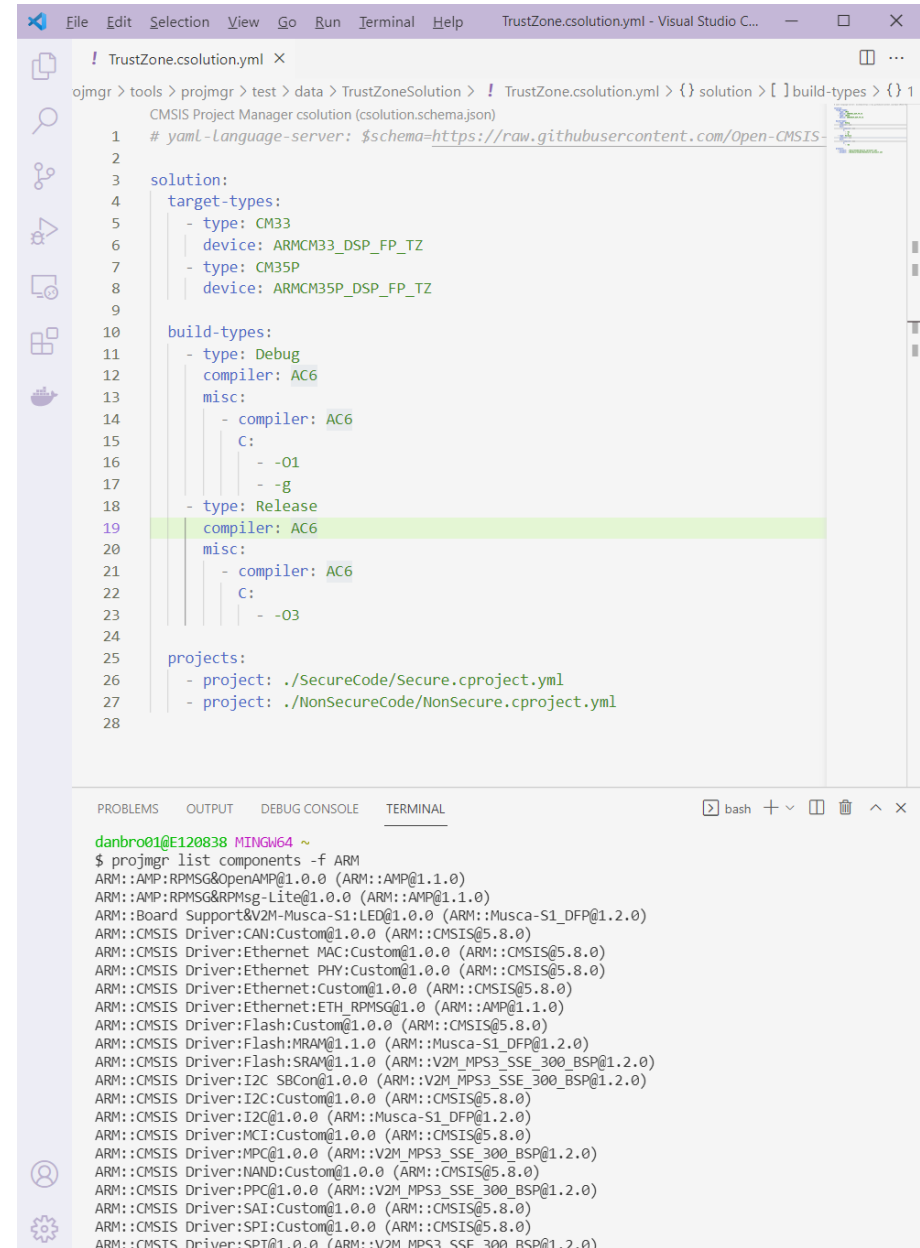
Structure of AWS Software Stack – example MQTT Auth Demo

App Layer

MQTT Mutual Auth Demo

Backoff Algorithm

Core JSON

Core MQTT Agent

Core MQTT

FreeRTOS Kernel

mbed TLS

Core PKCS11

Transport Interface

CMSIS-RTOS2 FreeRTOS Interface

Interfaces:

Provides:
- RTOS2
- FreeRTOS

Consumes:
- AWS_Socket (IoT Secure Interface)
- STDOUT
- Heap = 65536

Interface/AWS

IoT Secure Interface

FreeRTOS+ TCP

Network Interface (to CMSIS-Driver Ethernet)

IoT Secure Interface

Translation Layer to CMSIS-Driver WiFi

IoT Secure Interface

Translation Layer to IoT Utility – IoT Socket

Interfaces:

Provides:
- AWS_Socket (IoT Secure Socket)

Consumes:
- <depends on implementation>

Board & Module

CMSIS-Driver Ethernet

Compatible with:
- Boards with physical Ethernet
- Arm Virtual Hardware - Ethernet

CMSIS-Driver WiFi

Compatible with:
- Boards/Modules with WiFi

IoT Utility – IoT Socket

Compatible with:
- IoT Socket compliant stacks
- Arm Virtual Hardware - VSocket

Interfaces:

Provides:
- STDOUT, Heap, and drivers

Consumes:
- RTOS2

This stack should be aligned with PSA to provide OTA, Key Storage, …

© 2022 Arm

Complete layers here: https://github.com/RobertRostohar/CB_Lab4Layer/tree/aws

arm

Plan for next week

- Provide a development snapshot:

open-cmsis-pack-tools_0.9.0.zip

- cpackget
- projmgr + schemas
- cbuildgen

Wrap Up

- Actions all:
  - Provide review feedback
  - Suggest topics for next week's agenda

- Review of currently open issues in respository:
  - Issues: Still relevant? Open Actions? Next Steps? Refinements?

- Next Open-CMSIS-Pack meeting: 1st Feb. 2022 @ 16:00 CET (15:00 UK)

Thank you