Open-CMSIS-Pack

Technical Project Meeting 2022-09-20

This meeting is recorded !

Agenda

- Welcome
- Change Boards
- For Review
- IoT Workshop Examples (Arm)
- Wrap Up

Review Change Boards

- Pack Specification Change Board
  - #144 accepted by Decision Committee -> PR#147 merged
  - #112 “hidden” components -> PR#153/PR#154

- Solution Specification Change Board

Feedback on Generator Concept

https://github.com/open-cmsis-pack/devtools/blob/main/tools/projmgr/docs/Manual/Generator%20(Proposal).md

- Is the overall concept useful?
- What type of backward compatibility (with MDK, CMSIS-Pack-Eclipse) do we need?
  - We should not break current implementation, but the new implementation might be `csolution` only (at least in the beginning).
  - Can STM32CubeMX support be made compatible by providing an additional *.FTL file in the DFP?
  - Decision would result in deprecating *.GPDSC format in the long run.
- Is a generator registered, i.e. by a PATH variable?
- Can we remove some options in the <generator> element?
  - When the generator is registered by a PATH variable it can be call from command line.
  - Deprecate <web url …>; generators may use service interfaces
  - Deprecate <eclipse plugin…>; as no generator works this way and `csolution` cannot support it.
- Is the proposed $G the path to the cbuild.yml file, do we need a different file? #433
- Should we deprecate the `csolution` run command?
- Is the simplified "instances" (see #76) sufficient given the fact that we can work via `interfaces:`?
- Duplicate `component` definition in `genlayer.yml`? #157

For Review

- Feedback on PR #154 aka "hidden" proposal from NXP #112 (also PR #153)
- PDSC: Conclusion on define in compile element #351 -> preference is to reject it & document better

- Component select/deselect behaviour and side effects (#466)
- Support for build-type and target-type in cproject #450 (<= #355)
- Best way to handle components selection/removal in a project #466
- Handling of Multiple Component Instances #76
- Please help us on tickets market with question

New in documentation, not yet implemented in `csolution` utility

- csolution --version option #128
- csolution list config option #142
- csolution updated help text #330 (deprecate command line option –p)
- csolution tool should issue error when adding multiple component variants #156
- Already implemented: csolution tool should use only latest packs, pack filter with wildcards #412
- csolution tool you mark config@version files R/O #313

For Review (cont'd)

- Yet another doubt about the concept of component ID #157
- Project customized config file replacement of SW pack component #296
  - csolution/cproject Format extension required ? (Holt)
- Csolution: add json file as list result #352
  - Machine readable list of generators
- Who owns implementation of C/C++ linker flag support (#224)
  - PR492 work in progress (Yves Linaro/NXP)

arm

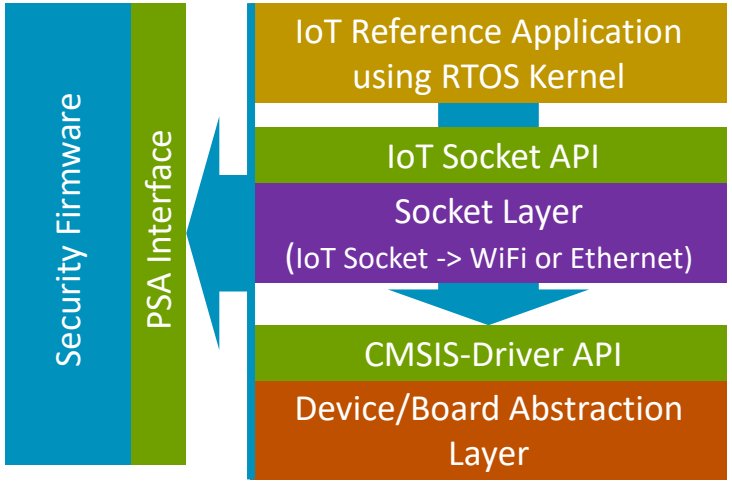
IoT Workshop Examples

Examples for to stimulate partner contributions

- Open-CMSIS-Pack for PoC demos
- Arm will use it for KSC and AVH demo workflows

IoT Workshop Example - Structure

Reference Application Framework: map many applications to many boards



Objectives

Re-use existing packs from AWS and NXP

- provide feedback when required

Define the overall structure of CMSIS-Packs, i.e. for

- TF-M
- mbedTLS
- AVH Corstone-300 BSP
- Board BSP (exemplified on STM32U5, NXP)

SW Building Blocks

- Should come from multiple vendors. Requirement for standardized interface between the components (Open-CMSIS-CDI)
- Reference Application: should be tested with a CI system against a standardized CDI framework
- Should run (within reason) on many different existing v8M and v7M devices (TrustZone optional)
- Should include OTA services with standardize interfaces
- Future variants of the Framework should also support other application types (DSP, ML, Graphics)

Designed for `csolution` tool

- Examples use Open-CMSIS-Pack and the csolution workflow with layers
- Layer type names should be descriptive, i.e. board, socket, security
- Examples are used to fine-tune the csolution workflow (see next slide)
- Interface requirements between cproject/clayer files should be described
- Final design is that layers are provided by software packs

Other Requirements:

Defined Startup/Call Sequence (see https://github.com/MDK-Packs/CB_Lab4Layer/tree/master/layer)

- Example: https://github.com/MDK-Packs/CB_Lab4Layer/blob/master/layer/Board/MIMXRT1064-EVK/main.c

Interface: node in cproject.yml / clayer.yml files

Objective: help to identify the best structure for csolution yml files

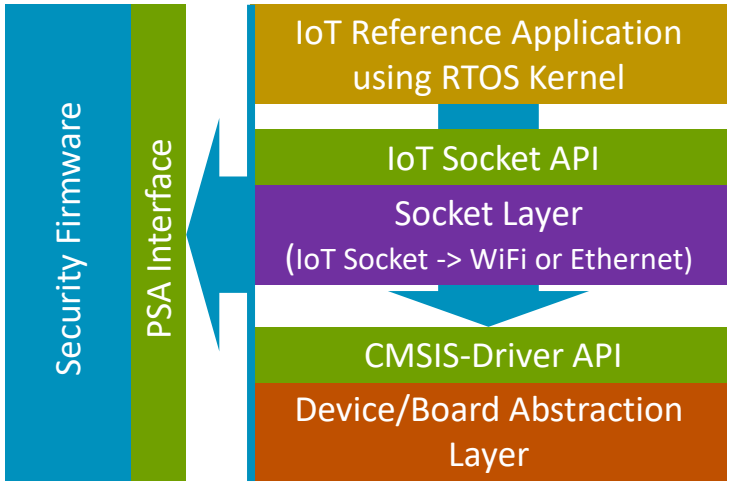
cproject.yml

```yaml
interfaces:
  provides:
  - RTOS2
  consumes:
  - CMSIS Driver USART Print
  - IoT Socket


requires-layers:             # project requires layer templates
  - type: Socket
  - type: Security
  - type: Board              # tool: check for a board layer

# csolution tool identifies compatible layers and lists it
# user enters then:
layers:
   - layer: security-tfm.clayer.yml
   - layer: socket-wifi.clayer.yml
   - layer: board-stm32u5-wifi.clayer.yml
```

clayer.yml

```yaml
layer:
  type: Board
  variant: IoT-WiFi-Ethernet
  description: Board setup with WiFi and Ethernet interface
# potentially compatibility information (not sure if required)
  for-device: device-name
  for-board: board-name
# i.e. future layer variants: ML-Sensor-WiFi, IoT-Azure-WiFi

  interfaces:   # interface descriptions
    consumes:
    - RTOS2:
    provides:
    - CMSIS Driver Ethernet: 0       # driver number
    - CMSIS Driver USART Print: 2    # driver number
    - CMSIS Driver WiFi
    - Heap : 65536                   # heap size
```

IoT Workshop Example – variants



Objectives

Discover:

- yml file structure for csolution
- Propose pack structure to partners
- Validate tool workflows with AVH and KSC
- Explore importer from csolution -> MDK
- Initiate Open-CMSIS-CDI work
- Test VS Code tooling based on Keil Studio Pack
- Explore Corillium with STM32U5 simulation

IoT Reference Application

- MQTT Demo
- OTA Demo (only when Security TF-M is used)

Socket - lets start with:

- WiFi connecting to CMSIS-Driver WiFi on Board
- VSocket connecting to AVH
- TCP/IP with FreeRTOS components connecting to Ethernet

Security Firmware

- TF-M (preferable with SFC mode only) using MCUBoot – only for v8M Secure mode
- mbedTLS (TF-M storage is a mock-up with fixed/hard-coded credentials) – running on v8M NS or v7M

Device/Board Abstraction

- STM32U5
- NXP iMX 1000 series
- AVH

Possible configurations (initially)

- AVH with (a) VSocket, TF-M, (b) VSocket, mbedTLS
- NXP with Ethernet, mbedTLS
- STM with (a) WiFi, TF-M, (b) WiFi, mbedTLS

IoT Workshop Example – enable new labs + DevSummit demos

Like AWS Workshops

mbedTLS based workshops

Lab A1: Develop MQTT demo on KSC and with GitHub actions (show development workflow with AVH) [uses AVH with VSocket, mbedTLS]

Lab A2: Deploy MQTT demo on STM32U5 [uses STM32U5 with WiFi, mbedTLS]

Lab A3: Extend MQTT demo with “mock sensor data interface” (using KSC / GitHub actions setup) [uses AVH with VSocket, mbedTLS]

Lab A3-A: same demo as A3 but using VS Code tool setup

Lab A4: Deploy MQTT demo on STM32U5 and connect to physical sensors [uses STM32U5 with WiFi, mbedTLS]

Lab A4-A: same demo as A4 but using VS Code tool setup

TF-M based workshops

Lab B1: Develop MQTT, OTA demo using AVH with csolution command-line and AMI interactive [uses AVH with Vsocket, TF-M]

Lab B2: Deploy MQTT, OTA demo using STM32U5 with csolution command-line (ideally VS Code tool setup) to deploy to STM32U5

Lab B3/B4: Deploy Lab A3/Lab A4 to AVH or STM32U5 (using Mock Sensor data or physical sensor data)

Question: should we host this example on github.com/open-cmsis-pack?

Wrap Up

- Other review “asks” before finally closing issues:
  - Anything still open for “Discussion Done” items.
  - Can you provide answers for “Question” items.

- Next Open-CMSIS-Pack meeting: 27th Sep. 2022 @ 16:00 CET (15:00 UK)

Thank you