Open-CMSIS-Pack

Technical Project Meeting 2023-09-05

This meeting is recorded !

Agenda

- Welcome back
- Project Boards
- Component Taxonomy
- Generator Workflow (Revised)
- Issues to Review
- Wrap Up

Boards:

- Open-CMSIS-Pack Specification Change Board
    - Adding `image` as child element of `part` #246 (PR #250) - merged

- CMSIS-Toolbox 2.1 Project Board
    - Released on Friday 2023-09-01
    - https://github.com/Open-CMSIS-Pack/cmsis-toolbox/releases/tag/2.1.0
    - https://artifacts.keil.arm.com/cmsis-toolbox/2.1.0/ - vcpkg / signed binaries

- CMSIS-Toolbox 2.2 Project Board
    - See progress and issues in scope for version 2.2.0
    - Please review and provide feedback in case you see topics missing
    - Add issues or comment on existing issues that you think should be added to 2.2.0

Taxonomy Specification

- Tool based taxonomy definition (NXP David)
  - Protege as tool for taxonomy modeling (discussion #251)
- Analysis of current public pdsc files Cclass:
  - When are vendor or product names useful?
    - EmSA, Qualcomm, Clarinox, NXP Component, Sin_TouchKey, SharkSSL, wolfSSL, FreeRTOS, etc.
    - edgefast_wifi, edgefast_wifi_nxp
    - Convention: Cclass="<vendor> Drivers"
  - Can we unify similar names?
    - Driver, Drivers, Native Driver, Device Driver, MCU Driver HAL, HAL
    - Graphics, Graphics Display
    - IoT Client, IoT Service, IoT Utility, AWS IoT
    - BSP, Board Support, Board
  - Can we avoid superfluous characters making the componentID hard to edit manually?
    - ___Group___, ___Subgroup___, ___Variant___, ___Peripheral___

Taxonomy (cont'd)

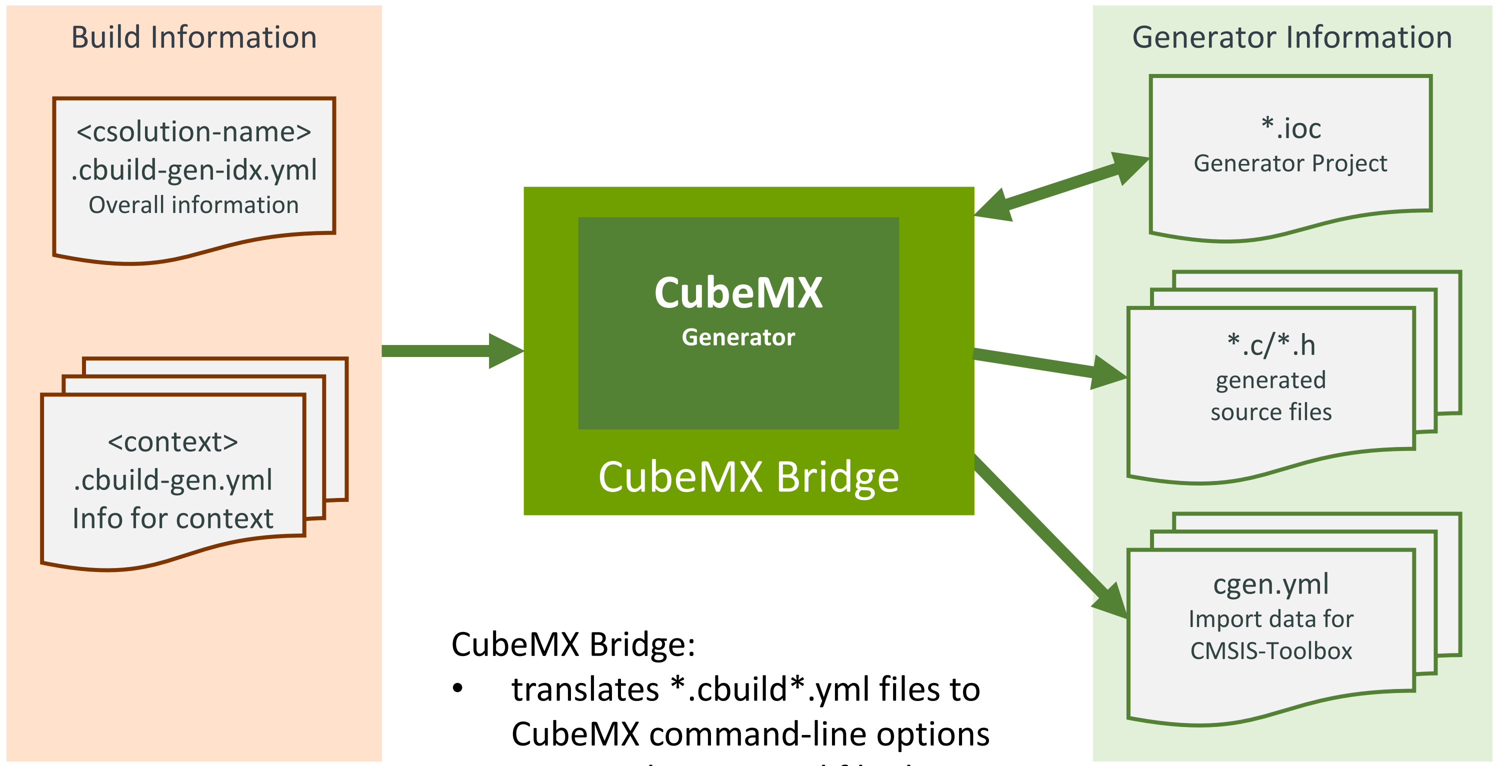
- Some Cclass names to review and specify further e.g.:
  - Project
  - Library
  - Simulation

- What role should Cbundle play in the context of taxonomy?

- Proposal: Create a dedicated repository for the taxonomy definition pack (Feedback: #252)
  - CMSIS.Taxonomy.pdsc
  - Documentation of scope and purpose
  - Pull Request and Review Process for extending
  - packchk validates components against the CMSIS.Taxonomy.pdsc
  - packchk flags definition of <taxonomy> in packs

Generator Workflow (Revised Proposal)

- Implement support for STM32CubeMX and MCUxpresso: Simplified Generator Proposal



CubeMX Bridge:

- translates *.cbuild*.yml files to CubeMX command-line options
- Creates the cgen.yml file that is used by CMSIS-Toolbox to integrate generated files.

Actions:

- Provide Feedback on Proposal #1112
- What features are required in cgen.yml?
- Closing the gaps for layers
- Would this proposal also work for MCUxpresso?

Issues to Review

- `cpackget add` incorrect error message in case the pack cannot be found #206
- [packchk] validate `<url>` starts with `https://` #1109
- *.cbuild.yml - add pack ID + path entries for `device` and `board` #1111
- [csolution] Add `warnings:` option `all` in yml input #974
  - Clarify: `on` versus `all`

Feedback:

- CMakeLists Proposal (Daniel) - leave comments and feedback in #1044
  - Mixing toolchains, mixing toolchain versions in one build of a context set
  - Separate binary or adding to csolution
  - Programming language C++ vs. GO

Wrap Up

Is anyone preparing/working on a topic to present and discuss in the coming weeks?

- Please contact Joachim.Krech@arm.com ahead of the meeting

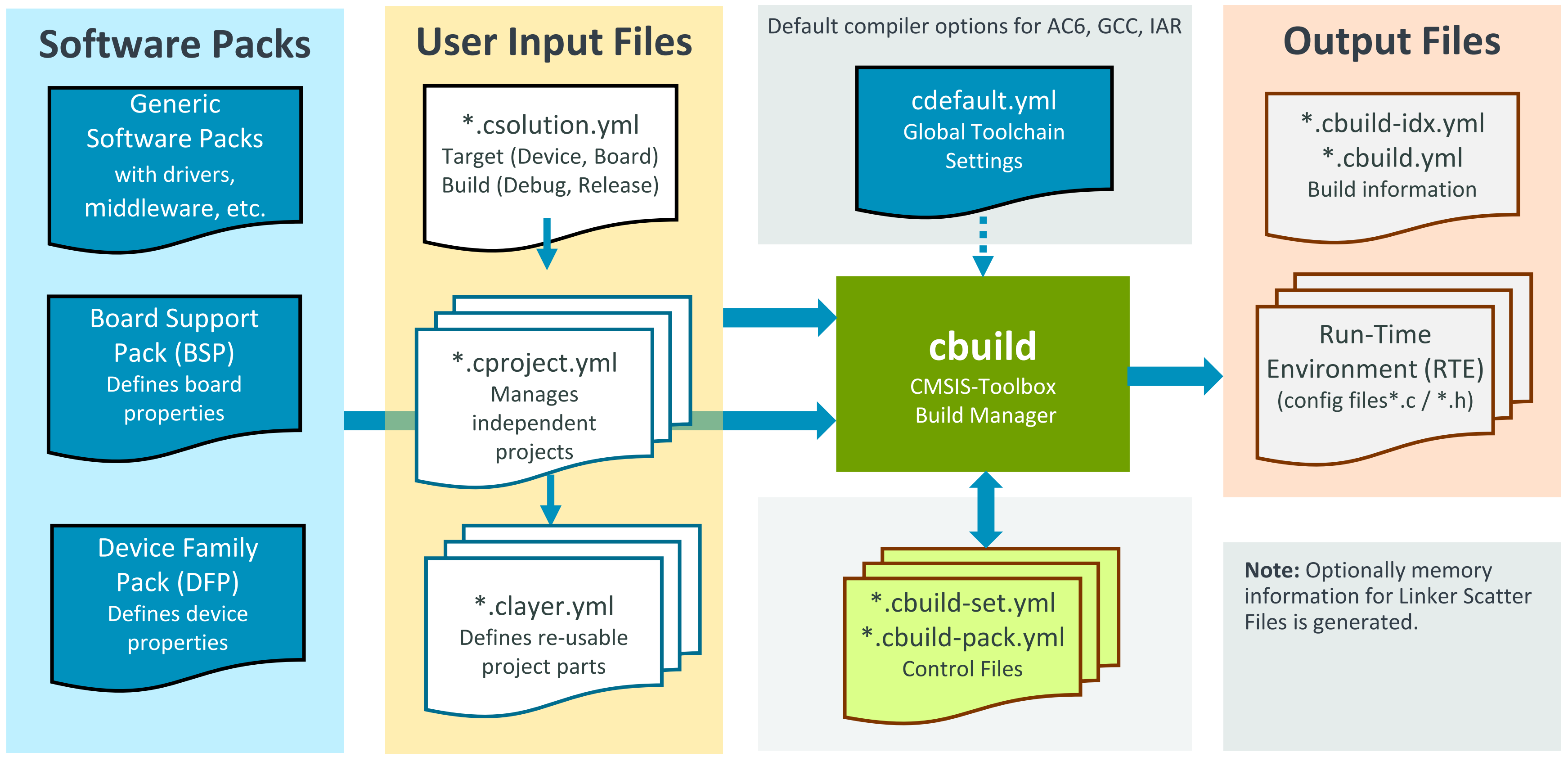
Next Open-CMSIS-Pack meeting: 12th Sep 2023 @ 16:00 CET (15:00 UK)

Thank you

Additional “cbuild” files

- ***<csolution-name>.cbuild-idx.yml*** - always contains the list of **all** project contexts with their corresponding <context>.cbuild.yml files references (as for: `cbuild convert <csolution-name>.csolution.yml`)
- cbuild/csolution tool: ***<csolution-name>.cbuild-set.yml*** - stores context-set specified at command line
  - See next slide
- ***<csolution-name>.cbuild-pack.yml*** - stores the pack versions used by the last conversions
  - See next slide
- ***<context>.cbuild-gen.yml*** - dedicated file written by csolution prior to calling the generator and passed as file reference to generator via $G command line argument. This file follows the cbuild.yml schema but contains absolute paths. It also specifies the packID and generatorID of the current generator.
  This is a temporary file, as it will be out of date once the generator completed. Generate into **intdir** of the context - due to absolute paths this file is location independent.
  ```
  csolution run <csolution-name>.csolution.yml -c <context>  -g <generatorID>
  ```
- Consider: Add a `clean` command removing *.cbuild*.yml files for a csolution:
  - `csolution clean <csolution-name>.csolution.yml`

cbuild Build Manager: File Overview



Software Packs
Generic Software Packs with drivers, middleware, etc.
Board Support Pack (BSP) Defines board properties
Device Family Pack (DFP) Defines device properties
User Input Files
*.csolution.yml Target (Device, Board) Build (Debug, Release)
*.cproject.yml Manages independent projects
*.clayer.yml Defines re-usable project parts
Default compiler options for AC6, GCC, IAR
cdefault.yml Global Toolchain Settings
cbuild CMSIS-Toolbox Build Manager
*.cbuild-set.yml *.cbuild-pack.yml Control Files
Output Files
*.cbuild-idx.yml *.cbuild.yml Build information
Run-Time Environment (RTE) (config files*.c / *.h)

Note: Optionally memory information for Linker Scatter Files is generated.

<csolution-name>.cbuild-pack.yml

- when file exist, it defines the scope of packs along with pack versions for the *.csolution.yml.
- when file does not exist, it is generated with currently processed packs.
- For a reproduceable build, only this file is required. Removing the need to store *.cbuild.yml files in repos

<csolution-name>.cbuild-set.yml

- stores the setting of the –context options.
- When no –context is given, the settings from this file are used.
- When no –context and no file exists, all target-types and the first build-type is generated.

- <csolution-name>.cbuild-set.yml - stores context-set that is currently processed
- <csolution-name>.cbuild-pack.yml - stores pack along with versions that are used