Modernization of MMC software development at DESY using open-source tools.

Patrick Huesmann, Michael Fenner 2020-12-03

MicroTCA Workshop 2020, Hamburg

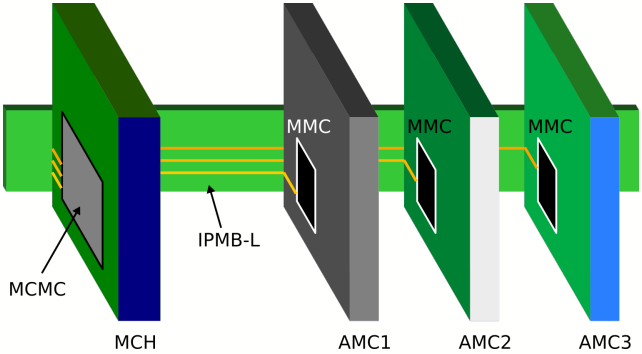
MicroTCA TECHNOLOGY LAB

HELMHOLTZ RESEARCH FOR GRAND CHALLENGES

DESY.

MMC tasks & scope (recap)

One of most important features of MicroTCA is out-of-band management interface.



MicroTCA Carrier Management Controller (MCMC) (part of MicroTCA Carrier Hub - MCH) connects to Module Management Controller (MMC) on Advanced Mezzanine Card (AMC)) over IPMB-L

from: MMC Stamp and its applications, J. Marjanovic, MTCA Workshop China 2019

MMC tasks & scope (recap)

- Hot swap & power management
- Status information
- Sensors, thresholds & alerts
  - temperature
  - voltages
  - currents
  - pin levels ...

MMC tasks & scope (recap)

- Hot swap & power management
- Status information
- Sensors, thresholds & alerts
  - temperature
  - voltages
  - currents
  - pin levels . . .

```
$ ipmitool -H MCH100191.tech.lab -P "" -B
AMC Hot Swap      | 0x00              | ok
049162C0F354      | 0x00              | ok
STAMP Temp        | 28 degrees C      | ok
AMC MP 3V3        | 3.41 Volts        | ok
AMC PP 12V        | 12.68 Volts       | ok
ADC0              | 0.03 Volts        | ok
ADC1              | 0.08 Volts        | ok
ADC2              | 0.03 Volts        | ok
I_RTM MP 3V3      | 0.01 Amps         | ok
I_RTM PP 12V      | 0.01 Amps         | ok
CPLD Done         | 0xff              | ok
RTM MP 3V3 PG     | 0x00              | ok
RTM PP 12V PG     | 0x00              | ok
RTM Fault         | 0x00              | ok
PGood_A           | 0xff              | ok
PGood_B           | 0xff              | ok
FPGA1 Init        | 0x00              | ok
FPGA1 Done        | 0x00              | ok
FPGA2 Init        | 0x00              | ok
FPGA2 Done        | 0x00              | ok
Inlet Temp        | 34 degrees C      | ok
Outlet Temp       | 31.50 degrees C   | ok
DC/DC1 1V2        | 34.50 degrees C   | ok
DC/DC2 Core       | 34 degrees C      | ok
```

MMC tasks & scope (recap)

- Hot swap & power management
- Status information
- Sensors, thresholds & alerts
  - temperature
  - voltages
  - currents
  - pin levels ...
- Application-specific payload management
  - JTAG multiplexing
  - Firmware update (HPM)
  - Custom IPMI commands ...

```
$ ipmitool -H MCH100191.tech.lab -P "" -B
AMC Hot Swap      | 0x00              | ok
049162C0F354      | 0x00              | ok
STAMP Temp        | 28 degrees C      | ok
AMC MP 3V3        | 3.41 Volts        | ok
AMC PP 12V        | 12.68 Volts       | ok
ADC0              | 0.03 Volts        | ok
ADC1              | 0.08 Volts        | ok
ADC2              | 0.03 Volts        | ok
I_RTM MP 3V3      | 0.01 Amps         | ok
I_RTM PP 12V      | 0.01 Amps         | ok
CPLD Done         | 0xff              | ok
RTM MP 3V3 PG     | 0x00              | ok
RTM PP 12V PG     | 0x00              | ok
RTM Fault         | 0x00              | ok
PGood_A           | 0xff              | ok
PGood_B           | 0xff              | ok
FPGA1 Init        | 0x00              | ok
FPGA1 Done        | 0x00              | ok
FPGA2 Init        | 0x00              | ok
FPGA2 Done        | 0x00              | ok
Inlet Temp        | 34 degrees C      | ok
Outlet Temp       | 31.50 degrees C   | ok
DC/DC1 1V2        | 34.50 degrees C   | ok
DC/DC2 Core       | 34 degrees C      | ok
```

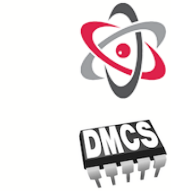
see also: http://www.rehlich.com/MicroTCA_IPMI_management

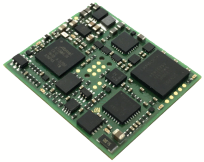
DESY MMC codebase history

- AVR-based legacy MMC, dating back to 2014
  - Original code based on a version from DESY MCS department
  - DESY continued development and ported code to ATxmega
  - Many contributions, e.g. from NCBJ, DMCS@TUL
  - MMC components scattered across AMC board
  - MMC-related HW not standardized (differs between board implementations)

DESY MMC codebase history

- AVR-based legacy MMC, dating back to 2014
  - Original code based on a version from DESY MCS department
  - DESY continued development and ported code to ATxmega
  - Many contributions, e.g. from NCBJ, DMCS@TUL
  - MMC components scattered across AMC board
  - MMC-related HW not standardized (differs between board implementations)
- Introduction of MMC Stamp based on ARM Cortex-M4 in 2018
  - Unified hardware platform (SoM, same HW across all AMC boards)
  - All management-related components on a single high-density board
  - ARM Cortex-M4: much more powerful than AVR, better suited for growing functionality
- Unified MMC: Hardware (processor) abstraction
  - Common codebase for legacy (AVR) and modern (ARM Stamp) MMC
  - Easy backporting of bugfixes and features to legacy MMC



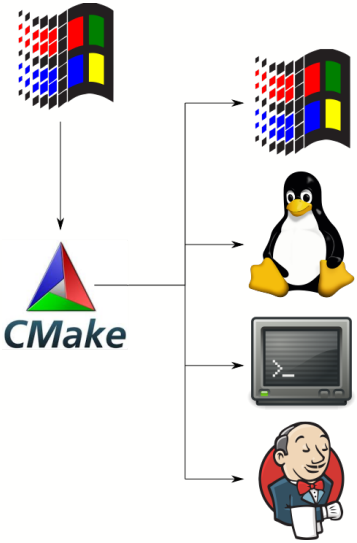


From Atmelstudio to CMake

- Previous solution: Atmelstudio IDE (Windows-only)

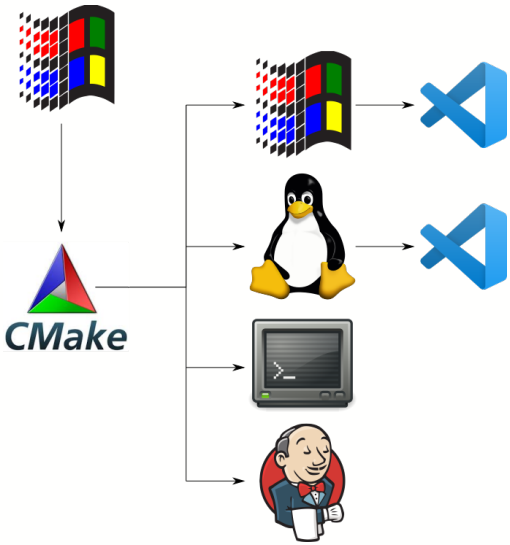
From Atmelstudio to CMake

- Previous solution: Atmelstudio IDE (Windows-only)
- New solution: CMake build system (cross-platform)
  - Supports any IDE w/ CMake integration
  - Supports build on command line w/o IDE
  - Supports CI on build server



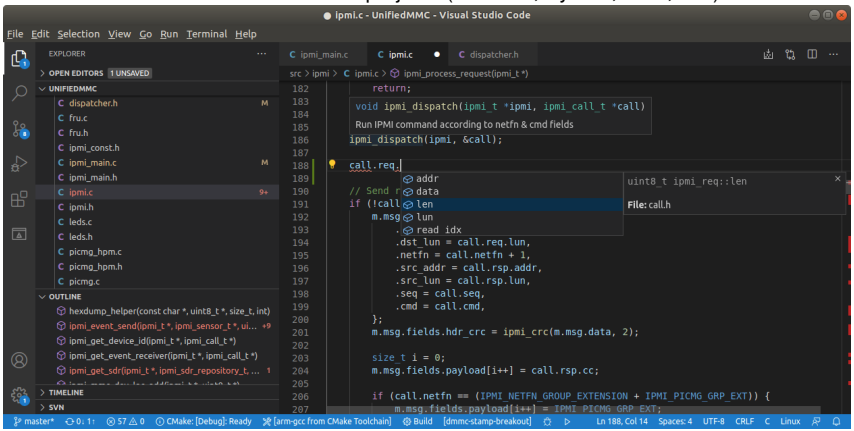
From Atmelstudio to CMake

- Previous solution: Atmelstudio IDE (Windows-only)
- New solution: CMake build system (cross-platform)
  - Supports any IDE w/ CMake integration
  - Supports build on command line w/o IDE
  - Supports CI on build server
- VS Code selected as default IDE
  - Cross-platform
  - Good CMake integration
  - Good ARM debugger integration
  - Huge user base & dev community



VS Code integration

- Code navigation, auto completion, gdb integration, ...
- Uses all CMake settings out of the box
- Just sits on top of CMake, doesn't get in the way
- Can also be used for all other kinds of projects (C++/x86, Python, Rust, etc.)



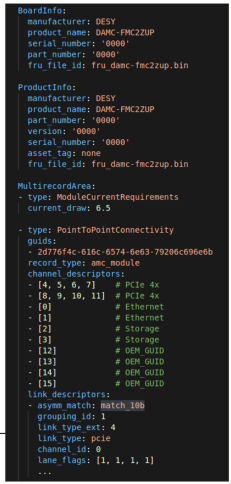
Build system pipeline

Source code & FRU config file (text only)

Bootloader source

Application source

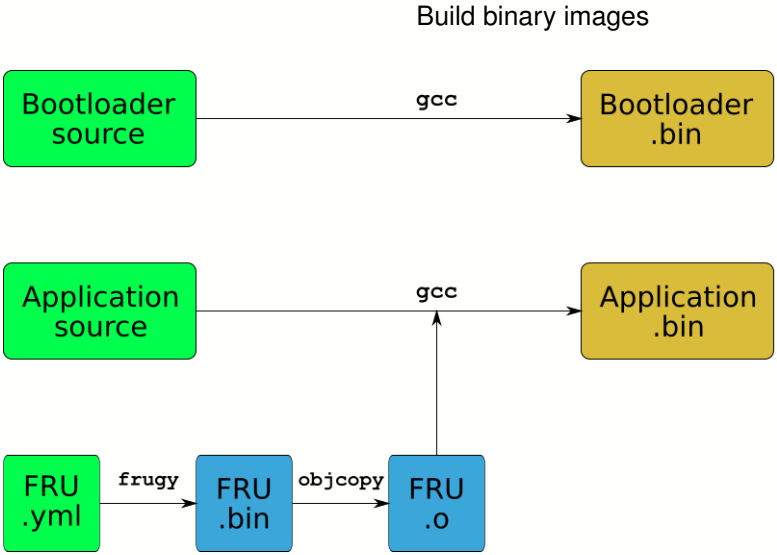
FRU .yml



```
BoardInfo:
  manufacturer: DESY
  product_name: DAMC-FMC2ZUP
  serial_number: '0000'
  part_number: '0000'
  fru_file_id: fru_damc-fmc2zup.bin

ProductInfo:
  manufacturer: DESY
  product_name: DAMC-FMC2ZUP
  part_number: '0000'
  version: '0000'
  serial_number: '0000'
  asset_tag: none
  fru_file_id: fru_damc-fmc2zup.bin

MultirecordArea:
- type: ModuleCurrentRequirements
  current_draw: 6.5

- type: PointToPointConnectivity
  guids:
  - 2d776f4c-616c-6574-6e63-79206c696e6b
  record_type: amc_module
  channel_descriptors:
  - [4, 5, 6, 7]    # PCIe 4x
  - [8, 9, 10, 11]  # PCIe 4x
  - [0]             # Ethernet
  - [1]             # Ethernet
  - [2]             # Storage
  - [3]             # Storage
  - [12]            # OEM_GUID
  - [13]            # OEM_GUID
  - [14]            # OEM_GUID
  - [15]            # OEM_GUID
  link_descriptors:
  - asymm_match: match_10b
    grouping_id: 1
    link_type_ext: 4
    link_type: pcie
    channel_id: 0
    lane_flags: [1, 1, 1, 1]
    ...
```

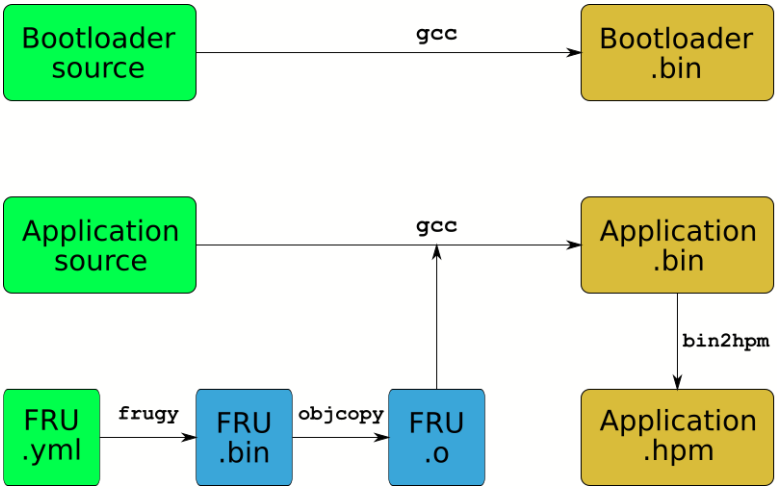
see also: https://github.com/MicroTCA-Tech-Lab/frugy

Build system pipeline



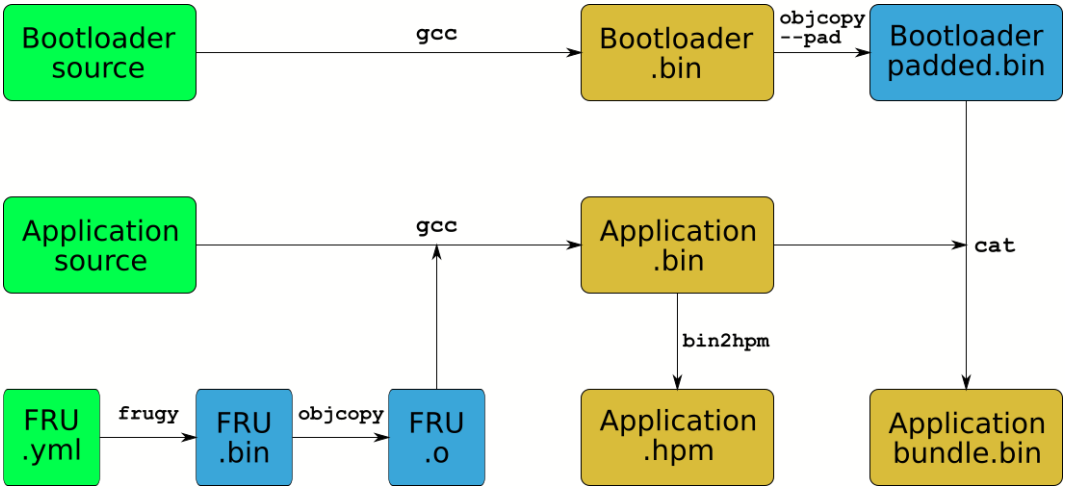
Build system pipeline

Build HPM update file from application image



Build system pipeline

Build bundle file containing bootloader & application



Continuous Integration

- CI setup based on Jenkins and Docker
- No special setup necessary on build server
- Build environment fully specified in Dockerfile
- All information needed to build the firmware is contained in source repository itself (Dockerfile and Jenkinsfile).

```
node {
    stage('Git') {
        step([$class: 'WsCleanup'])
        checkout scm
    }

    stage('Build') {
        sh 'docker/build_docker.sh'
    }

    stage('Archive') {
        archiveArtifacts artifacts: 'artifacts/**/*', fingerprint: true
    }
}
```

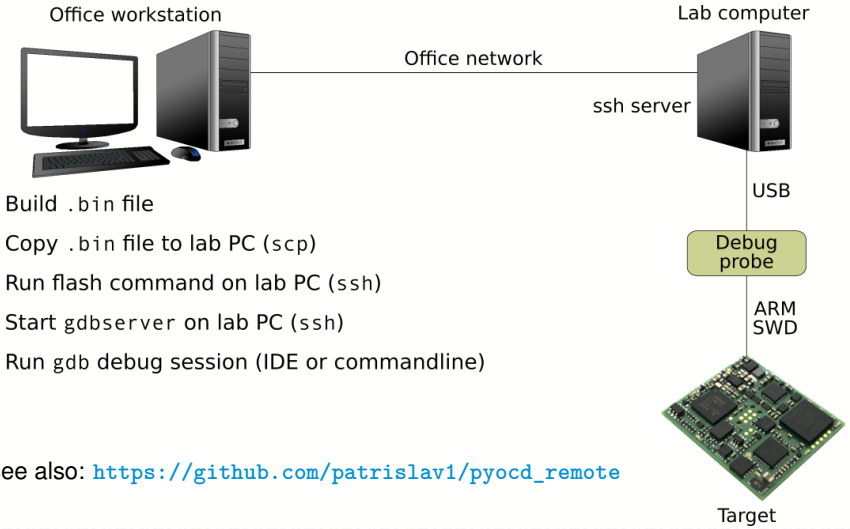
Continuous Integration

Automated IPMI integration test based on `pyipmi` (work in progress)

```
huesmann@mskpcx29856: ~/src/UnifiedMMC
File  Edit  View  Search  Terminal  Help
huesmann@mskpcx29856 ~/src/UnifiedMMC  master ● ./tests/test_ipmi.py 0x74 -m 192.168.1.252
Requesting Device ID ...
Device ID:
  device_id       : 0 ..................................... OK
  revision        : 0 ..................................... OK
  available       : False ................................. OK
  fw_revision     : 0.1 ................................... OK
  ipmi_version    : 1.5 ................................... OK
  manufacturer_id : 1343 .................................. OK
  product_id      : 49374 ................................. OK

Requesting FRU #0...
  received 500 bytes

FRU #0 board info:
  manufacturer    : DESY .................................. OK
  product_name    : DAMC-MMC-EvalKit ...................... OK
  serial_number   : 0000 ................................ FAILED
  part_number     : 0000 ................................ FAILED
  fru_file_id     : none .................................. OK
  custom_mfg_info : [] .................................... OK

FRU #0 board info FAILED

IPMI Test Suite FAILED.
✗ huesmann@mskpcx29856 ~/src/UnifiedMMC  master ●
```

Remote workflow

Remote debugging or ”How I Learned to Stop Worrying about Loud MTCA Crates”



Build `.bin` file

Copy `.bin` file to lab PC (`scp`)

Run flash command on lab PC (`ssh`)

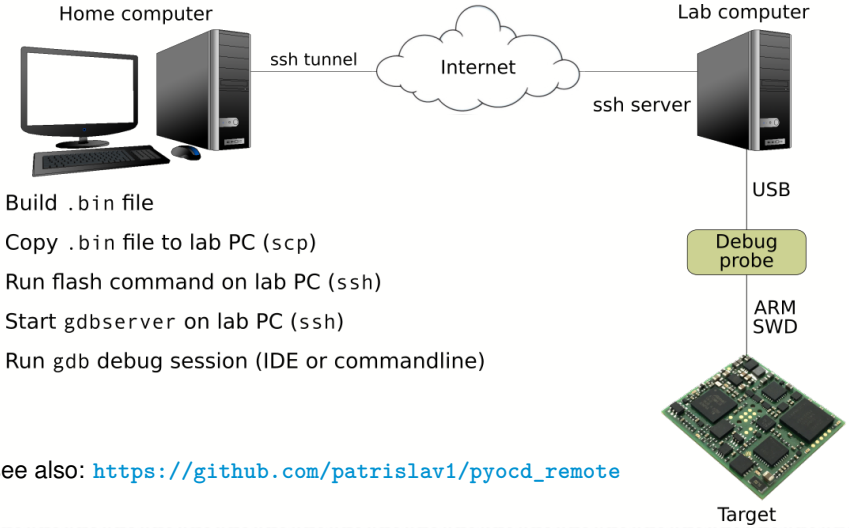
Start `gdbserver` on lab PC (`ssh`)

Run `gdb` debug session (IDE or commandline)

see also: https://github.com/patrislav1/pyocd_remote

Remote workflow

Remote debugging, ”Corona Lockdown Edition”



Build `.bin` file

Copy `.bin` file to lab PC (`scp`)

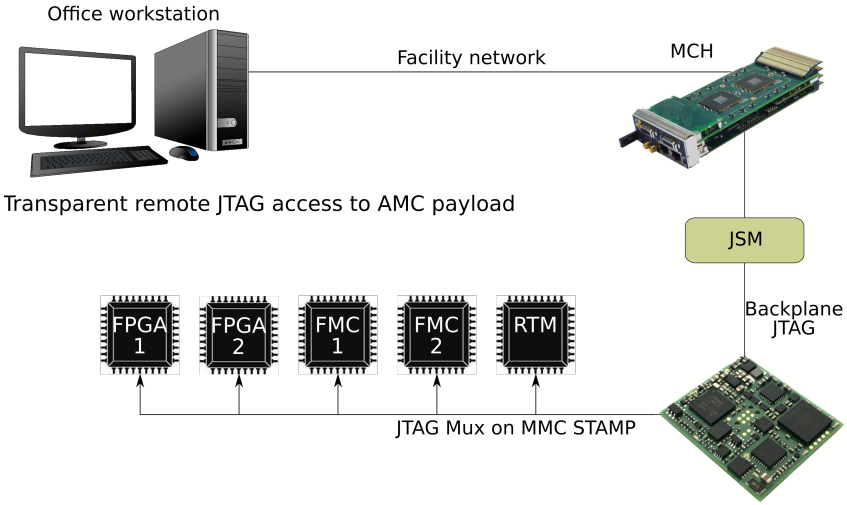
Run flash command on lab PC (`ssh`)

Start `gdbserver` on lab PC (`ssh`)

Run `gdb` debug session (IDE or commandline)

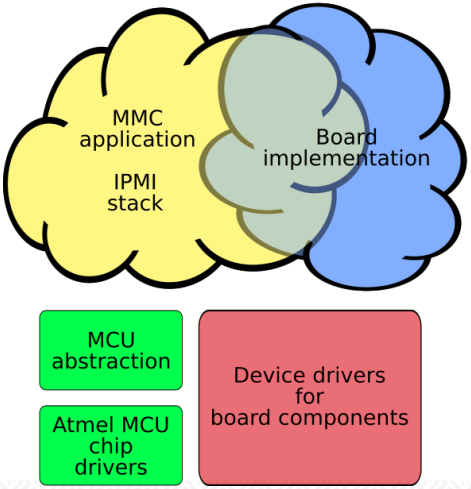
see also: https://github.com/patrislav1/pyocd_remote

Remote FPGA access via JTAG multiplexer



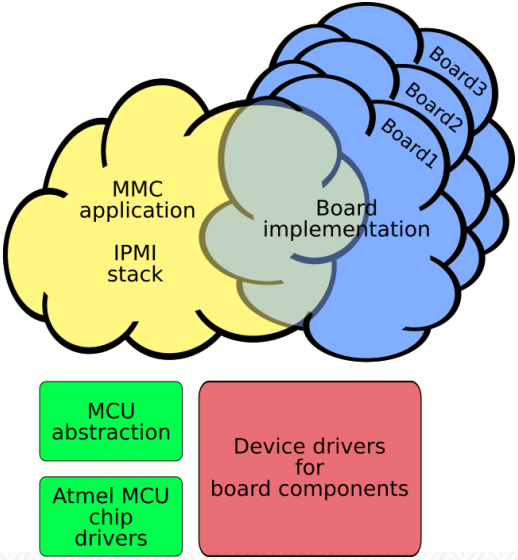
Code base structure - legacy

MMC legacy codebase

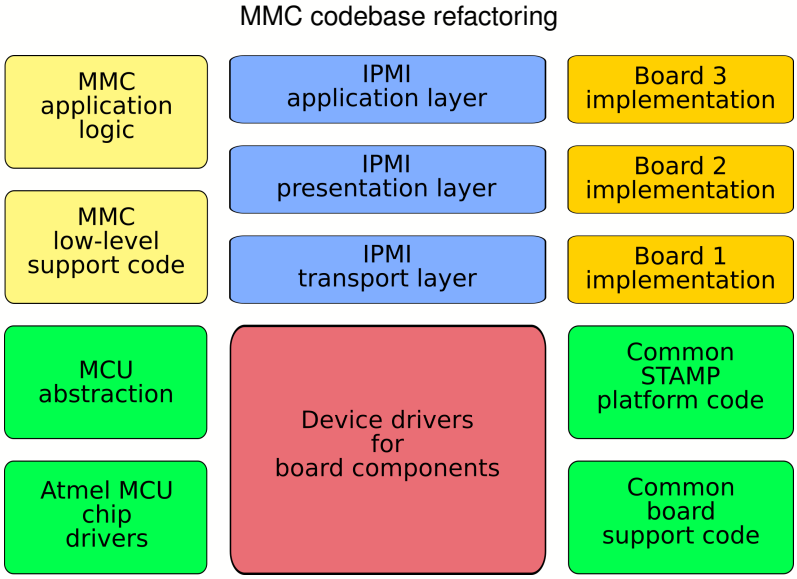


Code base structure - legacy

MMC legacy codebase supporting multiple boards

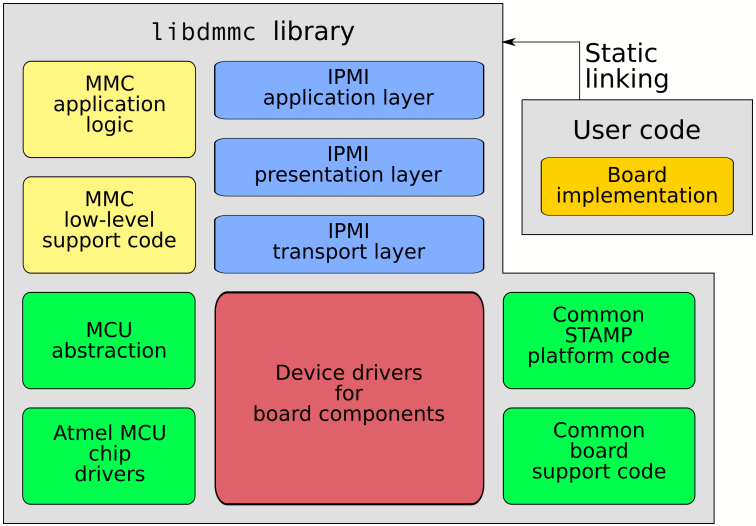


Code base structure - refactoring



Code base structure - refactoring

Separation into library and user application



Interfaces for board-specific customization

Power management

- Define `brd_pp_on()`, `brd_pp_off()`, `brd_pp_good()`

Interfaces for board-specific customization

Power management

- Define `brd_pp_on()`, `brd_pp_off()`, `brd_pp_good()`

IPMI sensors

- Straightforward interface to define arbitrary IPMI sensors
- Some ADCs, power monitors and temperature sensors already predefined

Interfaces for board-specific customization

Power management

- Define `brd_pp_on()`, `brd_pp_off()`, `brd_pp_good()`

IPMI sensors

- Straightforward interface to define arbitrary IPMI sensors
- Some ADCs, power monitors and temperature sensors already predefined

Custom IPMI commands

- Define table of IPMI command codes and callbacks
- Register with IPMI stack

Interfaces for board-specific customization

Power management
- Define `brd_pp_on()`, `brd_pp_off()`, `brd_pp_good()`

IPMI sensors
- Straightforward interface to define arbitrary IPMI sensors
- Some ADCs, power monitors and temperature sensors already predefined

Custom IPMI commands
- Define table of IPMI command codes and callbacks
- Register with IPMI stack

Command line interface
- Define table of CLI commands and callbacks
- Register with CLI module

Interfaces for board-specific customization

Power management
- Define `brd_pp_on()`, `brd_pp_off()`, `brd_pp_good()`

IPMI sensors
- Straightforward interface to define arbitrary IPMI sensors
- Some ADCs, power monitors and temperature sensors already predefined

Custom IPMI commands
- Define table of IPMI command codes and callbacks
- Register with IPMI stack

Command line interface
- Define table of CLI commands and callbacks
- Register with CLI module

FRU
- Define YAML configuration of FRU record
  see also: `https://github.com/MicroTCA-Tech-Lab/frugy`

Interfaces for board-specific customization

Power management
- Define `brd_pp_on()`, `brd_pp_off()`, `brd_pp_good()`

IPMI sensors
- Straightforward interface to define arbitrary IPMI sensors
- Some ADCs, power monitors and temperature sensors already predefined

Custom IPMI commands
- Define table of IPMI command codes and callbacks
- Register with IPMI stack

Command line interface
- Define table of CLI commands and callbacks
- Register with CLI module

FRU
- Define YAML configuration of FRU record
  see also: https://github.com/MicroTCA-Tech-Lab/frugy

HPM upgrade
- Define table of HPM targets and callbacks
- Register with HPM module

Conclusion

- Open source tools today offer an amount of features and convenience similar to vendor-supplied tools.
- With additional benefits of:
  - No vendor lock-in.
  - More portable, scriptable, extensible.
- SW engineering on modern embedded MCUs is similar to ”regular” SW engineering. Don’t underestimate it.
- Systems that are supposed to be generic and adaptable tend to get more complex and require more abstraction.

Thank you

https://techlab.desy.de

Deutsches Elektronen-Synchrotron DESY
A Research Centre of the Helmholtz Association
Notkestr. 85, 22607 Hamburg, Germany