MicroTCA Module Management with DMMC-STAMP

Overview & recent developments

Patrick Huesmann (DESY)
2023-12-07

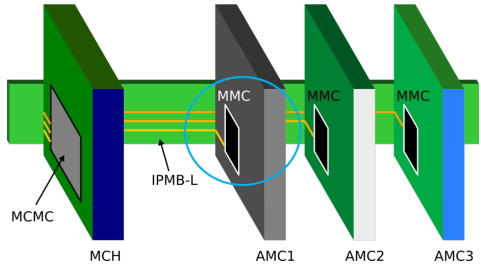
12th MicroTCA Workshop for Industry and Research, Hamburg

DMMC-STAMP overview

Overview MicroTCA MMC

- Management in MicroTCA
  - On MCH side:
    MicroTCA Carrier Management Controller (MCMC)
  - On AMC side:
    Module Management Controller (MMC)

- Responsibilities of MMC
  - IPMI protocol handling
  - FRU information (AMC, RTM, FMCs)
  - AMC payload management (FPGAs/SoCs)
  - RTM control (hot-plug)
  - Monitoring (temperatures, voltages, currents)
    Event handling (thresholds, alerts)



Overview DMMC-STAMP

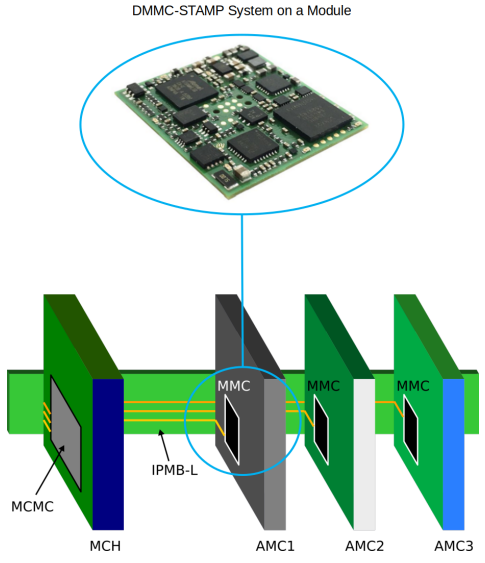
- DESY's drop-in solution for AMC MMC
- Used at DESY for internally developed AMCs
  - in production:

    DAMC-FMC2ZUP
    DAMC-FMC1Z7IO
    DAMC-DS812ZUP
    DAMC-MOTCTRL

  - in development:

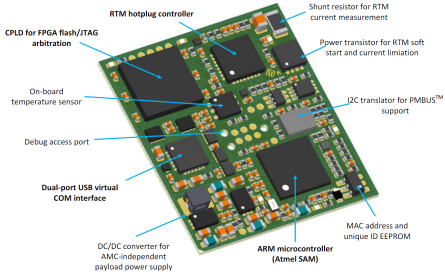
    DAMC-UNIZUP
    DAMC-DS5014DR
    DAMC-X3TIMER

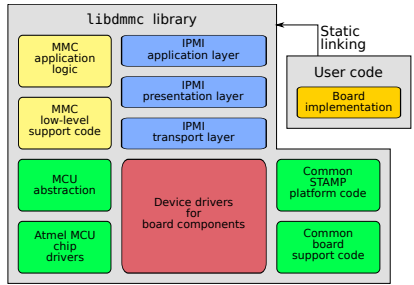
- Used by several customers in research and industry
- Interoperability tested with various MCHs (NAT, Vadatech)



DMMC-STAMP ecosystem

- System on Module (SoM)
  - 25.5 x 29.5 x 2.3 mm
  - Pre-programmed firmware
  - Evaluation board available (BoB)
- Software Development Kit (SDK)
  - MMC firmware customization
  - DESY MMC Software Library (libdmmc)
  - Example implementations (BoB, DAMC-FMC2ZUP)
- Open Source Tools and Templates
  - AMC and RTM Altium Designer Templates
  - `mmcterm`: serial over IPMB
  - `bin2hpm`: create HPM files for IPMI upgrade
  - `frugy`: read and write FRUs
  - `cpld-img-tools`: bitstream conversion for Lattice CPLDs





Recent developments & special features

FMC sensor integration

- Some FMCs dissipate lots of heat, requiring extra cooling
- To monitor their temperature and control cooling on demand, local temperature sensors are needed
  $\rightarrow$ FMC-local temperature sensors need to be integrated into IPMI monitoring
- But we don't want to change the MMC for every new FMC module

FMC sensor integration

- Some FMCs dissipate lots of heat, requiring extra cooling
- To monitor their temperature and control cooling on demand, local temperature sensors are needed
  → FMC-local temperature sensors need to be integrated into IPMI monitoring
- But we don't want to change the MMC for every new FMC module

- Solution: ANSI/VITA 57.1
  I2C Device Definition FRU record
- FRU record encodes a "device name" and a I2C address
  - MMC implements a lookup table from device name to sensor driver
  - MMC parses device name and address, then reads the sensor and exposes its reading to IPMI

  → MMC only needs to know the particular sensor,
  not the particular FMC

Field	Byte Offset	Bit Location	Length	Description
Subtype	0	7:4	4 bits	1 for I2C device definition subtype
Version	0	3:0	4 bits	0 for current version
Device String	1..N/8		N bits	Device address / name strings, see below

Table 9. Subtype 1: I2C Device Definition (variable length and optional)

The device string portion of this MultiRecord subtype consists of 6-bit ASCII text as defined in the ISD. The string is divided into one or more I2C device records. Each device record consists of one or more address characters followed by one or more bytes of device name.

From FMC standard (ANSI/VITA 57.1)

Example: Atom Computing DAC FMC

```
DAMC-FMC2ZUP@0x7E MMC>fru  Show parsed FRUs on MMC CLI
FRU #0:
Product info: DESY/CAEN ELS DAMC-FMC2ZUP-11EG
              S/N 22Y23W0632 P/N DAMCFMC2ZUP1
              Version rev8
  Board info: DESY/CAEN ELS DAMC-FMC2ZUP-11EG
              S/N 22Y23W0632 P/N DAMCFMC2ZUP1
              Mfg.Date 2022-09-27 10:17:00
Module current requirements: 6.5A
Zone3 interface compat: Class D1.1

FRU #1:
Not present

FRU #2:  This is FMC1 on the FMC carrier
Product info: N/A
  Board info: Atom Computing, Inc. Opus DAC
              S/N 2 P/N Opus-DAC-revA
              Mfg.Date 2022-05-18 18:02:00
DC Load P1_12P0V: 12V (min 11.4, max 12.6) ~0mV, min 0mA / max 1000mA
DC Load P1_3P3V: 3.3V (min 3.12, max 3.46) ~0mV, min 0mA / max 100mA
DC Load P1_VADJ: 2.5V (min 1.8, max 3.3) ~0mV, min 0mA / max 100mA
DC Output P1_VREF_B_M2C: 0 -0+0V ~0mV, min 0mA / max 0mA
DC Output P1_VREF_A_M2C: 0 -0+0V ~0mV, min 0mA / max 0mA
DC Output P1_VIO_B_M2C: 0 -0+0V ~0mV, min 0mA / max 0mA
FMC size: single, clock dir: m2c, TCK max clock: 64
      P1: lpc, num signals: A 68, B 0, num Gbt trcv: 0
      P2: lpc, num signals: A 0, B 0, num Gbt trcv: 0
FMC I2C device definition:
  name: AT30TS75A, addrs: 9 (0x48)  FMC1 has an AT30TS75A sensor at
                                    I²C address 0x48

FRU #3:
Product info: N/A
  Board info: CAENELS FMC-4SFP+
              S/N 17006 P/N F
              Mfg.Date 2015-06-10 00:00:00
```

```
nat> show_sensorinfo 11  Show AMC sensors on MCH console
Sensor Information for FRU 11 / AMC7
=============================================================================
  #   SDRType  Sensor Entity Inst  Value     State    Name
-----------------------------------------------------------------------------
  -   MDevLoc         0xc1  0x67                    DAMC-FMC2ZUP-11E
  0   Full     0xf2   0xc1  0x67  0x01              AMC Hot Swap
  1   Compact  0x0b   0xc1  0x67  0x00              0x00  682719B5F5E3
  2   Full     Temp   0xc1  0x67  30.0 C     ok     STAMP Temp
  3   Full     Voltage 0xc1 0x67  3.360 V    ok     AMC MP 3V3
  4   Full     Voltage 0xc1 0x67  12.32 V    ok     AMC PP 12V
  5   Full     Current 0xc1 0x67  0.000 A    ok     I_RTM MP 3V3
  6   Full     Current 0xc1 0x67  0.00 A     ok     I_RTM PP 12V
  7   Compact  0x14   0xc1  0x67  0x01              0x00  CPLD Done
  8   Compact  0x14   0xc1  0x67  0x00              0x00  RTM MP 3V3 PG
  9   Compact  0x14   0xc1  0x67  0x00              0x00  RTM PP 12V PG
 10   Compact  0x14   0xc1  0x67  0x00              0x00  RTM Fault
 11   Compact  0x14   0xc1  0x67  0x01              0x00  PGood_A
 12   Compact  0x14   0xc1  0x67  0x01              0x00  PGood_B
 13   Compact  0x14   0xc1  0x67  0x01              0x00  FPGA1 Init
 14   Compact  0x14   0xc1  0x67  0x01              0x00  FPGA1 Done
 15   Compact  0x14   0xc1  0x67  0x01              0x00  FPGA2 Init
 16   Compact  0x14   0xc1  0x67  0x01              0x00  FPGA2 Done
 17   Full     Temp   0xc1  0x67  37.5 C     ok     Inlet Temp
 18   Full     Temp   0xc1  0x67  37.5 C     ok     Outlet Temp
 19   Full     Temp   0xc1  0x67  37.0 C     ok     LTM4630 Temp
 20   Full     Temp   0xc1  0x67  39.0 C     ok     LTM4650 Temp
 21   Full     Temp   0xc1  0x67  41.0 C     ok     LTM4633_F Temp

 30   Full     Voltage 0xc1 0x67  1.7856 V   ok     VCC_Vadj
 31   Full     Voltage 0xc1 0x67  1.1904 V   ok     VCC_1V2
 32   Compact  0x14   0xc1  0x67  0x01              0x00  Opus DAC PG_M2C
 33   Full     Temp   0xc1  0x67  35.0 C     ok     Opus DAC AT30TS7
 34   Compact  0x14   0xc1  0x67  0x01              0x00  FMC-4SFP+ PG_M2C
 35   Compact  0xf0   0xc1  0x67  0x10              HS 011 AMC7
-----------------------------------------------------------------------------
```

The FMC temperature is fully integrated into MTCA management

PMBUS multi-chip configuration

- On complex boards with lots of power rails, a lot of power management logic is required
- Example DAMC-MOTCTRL:
  - 2x LTC2979 (PMBUS manager for ext. DC/DCs)
  - 1x TPS40425 (PMBUS controller for ext. MOSFETs)
  - 2x TPS65400 (PMBUS PMU with int. MOSFETs)
- **3 different Windows applications** and **3 different programming adapters** needed, just to configure the PMBUS chips...
  - Analog Devices LTpowerPlay & DC1613A
  - Texas Instruments Fusion Digital Power Designer & USB2ANY
  - Texas Instruments PI-Commander & USB-TO-GPIO2

Programming adapters needed for power managers on the DAMC-MOTCTRL

Conversion script for PMBUS project files


```
$ ../libdmmc/tools/pmbus_conv.py \
    motctrl_power_manager_all_ok_r2.proj \
    tps40425_09h_0V8Strimmed_rc1.xml \
    tps65400_device1_69h.xml,0x69 \
    tps65400_device2_6Ch.xml,0x6c \
      -o pmbus_conf.bin
Parsing motctrl_power_manager_all_ok_r2.proj...
Converting a LTC29xx configuration file
Found LTC2979 at 0x5c
Writing page section 0x00
Writing page section 0x01
Writing page section 0x02
Writing page section 0x03
Writing page section 0x04
Writing page section 0x05
Writing page section 0x06
Writing page section 0x07
Found LTC2979 at 0x5e
Writing page section 0x00
Writing page section 0x01
Writing page section 0x02
Writing page section 0x03
Writing page section 0x04
Writing page section 0x05
Writing page section 0x06
Writing page section 0x07
Parsing tps40425_09h_0V8Strimmed_rc1.xml...
Converting a TPS40425 configuration file
TI file detected: Texas Instruments Fusion Digital Power Designer v7.7.1.0 [2021-10-01]
Found TPS40425 at 0x09
Writing page section 0xff
Writing page section 0x00
Writing page section 0x01
Parsing tps65400_device1_69h.xml...
Converting a TPS65400 configuration file for device at 0x69
Writing page section 0x03
Writing page section 0x02
Writing page section 0x01
Writing page section 0x00
Parsing tps65400_device2_6Ch.xml...
Converting a TPS65400 configuration file for device at 0x6c
Writing page section 0x03
Writing page section 0x02
Writing page section 0x01
Writing page section 0x00
```

```
$ hexdump -C pmbus_conf.bin
00000000  00 4e 05 5c 4c 54 43 32  39 37 39 00 00 00 00 00  |.N.\LTC2979.....|
00000010  03 97 5e 4c 54 43 32 39  37 39 00 00 00 03 97 03  |..^LTC2979......|
00000020  97 09 54 50 53 34 30 34  32 35 00 00 07 2e 01 1e  |..TPS40425.....|
00000030  69 54 50 53 36 35 34 30  30 00 00 08 4c 00 a2 6c  |iTPS65400...L.\l|
00000040  54 50 53 36 35 34 30 30  00 00 08 ee 00 a2 01 80  |TPS65400........|
00000050  61 5c 10 12 10 08 13 35  d2 80 13 36 d2 40 13 4f  |a\.....5...6.@.O|
00000060  e0 48 12 50 80 13 51 ea  30 13 52 80 00 13 53 e5  |.H.P..Q.0.R...S.|
00000070  80 12 54 80 13 55 d3 c0  12 56 80 13 57 d3 80 13  |..T..U...V..W...|
00000080  58 80 00 13 59 80 00 12  5a 00 01 b0 00 00 13 b2  |X...Y...Z.......|
00000090  00 00 13 b4 00 00 13 d1  1c 4b 13 d4 00 ff 12 d5  |.........K......|
000000a0  0f 12 d6 00 12 d7 0f 12  d8 00 12 d9 00 12 da 00  |................|
000000b0  13 db 13 e8 13 dc 13 e0  13 e1 e6 20 13 e2 80 00  |........... ....|
000000c0  13 e3 80 00 12 e4 ff 12  e6 5c 12 f7 00 11 00 12  |.........\......|
000000d0  01 06 12 02 16 13 21 73  33 13 24 79 9a 13 25 78  |......!s3.$y..%x|
000000e0  fe 13 26 6d 70 13 40 7e  b8 12 41 80 13 42 7b d7  |..&mp.@~..A..B{.|
000000f0  13 43 6a 8f 13 44 67 ae  12 45 47 13 5e 6e 97 13  |.Cj..Dg..EG.^n..|
00000100  5f 6c 4a 13 60 ba 00 13  61 db c0 13 62 e2 80 12  |_lJ.`...a...b...|
00000110  63 80 13 64 0a bc 13 b1  00 00 13 b3 00 00 13 d0  |c..d............|
```

Writes binary blob containing device list & register contents, to be linked into DMMC-STAMP firmware

PMBUS configuration from DMMC-STAMP

Now the MMC configures the whole zoo of PMBUS chips with one CLI command

- No more fiddling with 3 different HW / SW tools
- Saves time & effort during production / development
- Less error prone
- Build system can directly read project files of PMBUS config tools
- PMBUS project files become part of the versioned MMC source tree
- PMBUS config is in a known state across all DAMC-MOTCTRL boards and can update with new MMC version

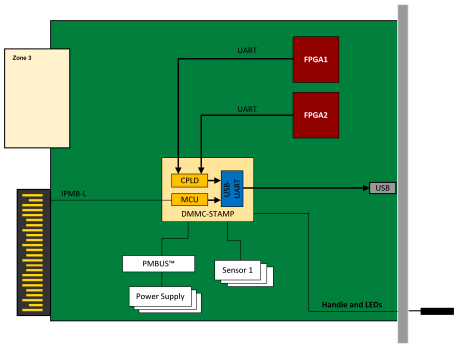
```
DAMC-MOTCTRL@0x7C MMC>pmc write
Waiting for PSM 0x5c shutdown...
Waiting for PSM 0x5e shutdown...
Write chip at 0x5c
LTC29xx device detected at 0x5c: 0x8061
Write globals
Write page 0
Write page 1
Write page 2
Write page 3
Write page 4
Write page 5
Write page 6
Write page 7
Write for #0 (LTC2979) successful
Store chip at 0x5c
LTC29xx device detected at 0x5c: 0x8061
Store for #0 (LTC2979) successful
Write chip at 0x5e
LTC29xx device detected at 0x5e: 0x8071
Write globals
Write page 0
Write page 1
Write page 2
Write page 3
Write page 4
Write page 5
Write page 6
Write page 7
Write for #1 (LTC2979) successful
Store chip at 0x5e
LTC29xx device detected at 0x5e: 0x8071
Store for #1 (LTC2979) successful
Write chip at 0x09
TPS40425 device detected at 0x09: 0x00c3
Write page 255
Write page 0
Write page 1
Write for #2 (TPS40425) successful
Store chip at 0x09
TPS40425 device detected at 0x09: 0x00c3
Store for #2 (TPS40425) successful
Write chip at 0x69
TPS65400 device detected at 0x69: 0x00f1
```

Remote console with Serial-over-IMPB

For serial console access we have USB-UART on the front panel.
But how to access the serial console remotely?

- Serial-over-IMPI would require MMC connected to Ethernet directly
- But in MicroTCA, Ethernet is part of payload, not management
  → DMMC-STAMP has no Ethernet connection, only IPMB(I2C)

- UART implemented on DMMC-STAMP CPLD
- Custom IPMI protocol for serial port forwarding (Serial-over-IPMB)
- Open source tool: `mmcterm`
- Support for 3 channels (MMC, FPGA1, FPGA2)



Remote console implementation on DMMC-STAMP

Remote console with Serial-over-IMPB (example)

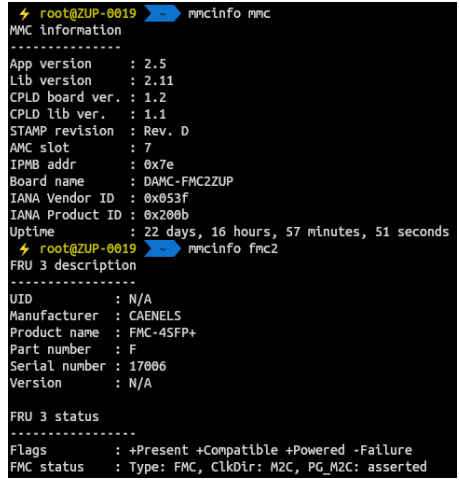
```
      MCH URL IPMB addr
$ mmcterm mskmchscav2 0x7c -l List channels
channel 0: MMC Console
channel 1: MPSoC Console
channel 2: Kintex-7 Console

$ mmcterm mskmchscav2 0x7c -c 0 Open channel 0 (MMC)
Press Ctrl-x to exit
DAMC-MOTCTRL@0x7C MMC>v
App. version      : V2.01
 Build host, date: msktechjenkins.desy.de, 2023-03-29T11:15:41Z
 Compiler version: 10.2.1 20201103 (release)
Library version   : V2.08
 Build host, date: msktechjenkins.desy.de, 2023-03-29T11:15:41Z
 Compiler version: 10.2.1 20201103 (release)
IPMI version      : 1.5
Vendor ID         : 0x053F
Product ID        : 0x0710
Board             : DAMC-MOTCTRL
STAMP revision    : Rev. A
STAMP UID         : 801F1234DD71
Copyright (C) 2022 Deutsches Elektronen-Synchrotron (DESY)
DAMC-MOTCTRL@0x7C MMC>%

$ mmcterm mskmchscav2 0x7c -c 1 Open channel 1 (MPSoC)
Press Ctrl-x to exit

root@mskdamcmotctrl:~# uname -a
Linux mskdamcmotctrl 5.4.0-xilinx-v2020.2 #1 SMP Thu Sep 21 13:28:03 UTC 2023 aarch64 aarch64 aarch64 GNU/Linux
root@mskdamcmotctrl:~# %
```

mmcterm on GitHub

mmc-mailbox

- Data interface between management (DMMC-STAMP) and payload (primary FPGA)
- Implemented as virtual I2C “EEPROM” for ease of access (U-Boot, etc.)

- Use cases
  - Ethernet MAC Address from DMMC-STAMP UID
    (No dedicated EEPROM necessary)
  - Orderly shutdown of Payload OS, when handle is pulled
  - Propagation of management data
    (RTM/FMC FRU, sensor values, slot number, ...) to payload side
  - AMC Ethernet discovery (`mmceth`)
    MMC retrieves ethernet address from payload and exposes it over IPMI
  - Application specific data transfer from/to MMC



mmcinfo command on payload Linux

Summary

- DMMC-STAMP is a drop-in solution for management of MicroTCA AMC boards
- FMC sensors are supported in compliance with ANSI/VITA 57.1
  $\rightarrow$ FMC temperatures are fully integrated into standard IPMI monitoring
  (i.e. can be monitored and trigger increase of cooling or AMC shutdown)
- In-system configuration of PMBUS components facilitates development & production
  Configuration of all PMBUS chips can be bundled with MMC code
- Remote console over Serial-over-IPMB provides useful remote debugging facilities
  for MMC itself and for payload FPGAs
- MMC mailbox allows data sharing between management (MMC) and payload (FPGA)

Thank you!

Acknowledgments: Carsten Duelsen, Michael Fenner

https://innovation.desy.de

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