OpenIPMC
A portable FOSS software for IPMCs

André Cascadan, Bruno Casu, Luigi Calligaris (SPRACE)
Lucas Arruda Ramalho (UNEMAT), Luis Ardila, Oliver Sander (KIT)
In this presentation

1. Motivation: ZynqMP as central control module
2. Pigeon Point IPMC and OpenIPMC
3. Reference development platform: Ultra96
4. Tests on Trenz + Serenity setup @ KIT
5. Tests on ESP32 setup
Motivation: ZynqMP as central control module

- **ATCA boards for LHC experiments need**
  - An IPMC for board management & monitoring
  - A Linux system for higher-level functions
  - The ZynqMP is a proposed integrated solution

- **Zynq Ultrascale+ MPSoC**
  - Two ARM processor domains (APU and real-time)
  - Xilinx FPGA programmable logic (good 4 sys integration)
  - Plethora of peripherals (PCIe, Eth, I2C, UART, USB, …)

- **Power domain partitioning**
  - LPD: IPMC (standalone/RTOS) on ARM-R5, TCM
  - FPD: Slow Control (Linux) on ARM-A53 quad cores
  - FPGA for services with HW support

- **Two firmware solutions proposed for IPMC**
  - Commercial closed-source by Pigeon Point
  - Free and open-source (OpenIPMC)
Zynq reference design for ZynqMP development

Trenz + Serenity setup @ KIT

- Based on Serenity (Imperial College)

Hardware

- Trenz Elektronik TE0803 module
  - Zynq US+ ZU4EG SoC
- Trenz Adapter board
  - Interface TE0803 to COM Express slot
  - Additional IPMC features
  - (I2C buffers, Eth Phy, EEPROM, SDCard…)
  - Interface to DIMM adapter
- DDR3 Mini-DIMM Adapter
  - Fits into CERN IPMC-compatible slot
  - Access to IPMC backplane signals

Serenity
IPMC software solution: Pigeon Point IPMC

- Pigeon Point ATCA IPMC software
  - Prototype version based on VPX version for ZynqMP (BMR-ZNQ-VPX)
  - Extensions by Pigeon Point and KIT for ATCA compliance

- Status
  - Integration and functional tests at KIT
  - Functional tests in Tracker Integration Facility (TIF) at CERN
  - Compliance tests at CERN test stand successful
    (some failed because of disabled functionality)
Pigeon Point IPMC - Boot log & IPMI FRU info

Board Information

Verbose mode turned on
Board # 2

Physical Slot # 2
96: Entity: (0x08, 0x6b) Maximum FRU device ID: 0x6b

P2PNIC Version: 2.3

Hot Swap State: M: Active, Previous: M: Activation In Progress, Last State Change Cause: Normal State Change (0x0)
Device ID: 0x12, Revision: 0, Firmware: 1.00 (Ver 1.0.0), P2PNIC Version: 2.3
Manufacturer ID: 0x4000, Product ID: cda, Auxiliary Rev: 00000000
Device ID String: "BMR-2NDP-ATCA"

Global Initialization: On, Power State Notification: On, Device Capabilities: 0x2d
Controller provides Device DMAs
Supported features: 0x2b
"Sensor Device" "SEL Device" "FRU Inventory Device" "IPMI Event Generator"

Links:
96: Base Interface (0x6b), Channel: 1
96: Base Interface (0x6b), Channel: 2
96: Fabric Interface (0xb3), Channel: 1
96: Fabric Interface (0xb3), Channel: 2
96: Update Channel Interface (0xb2), Channel: 1
96: Disabled Ports: 1

96: FRU # 0
Entity: (0x08, 0x6b)
Hot Swap State: M: Active, Previous: M: Activation In Progress, Last State Change Cause: Normal State Change (0x0)
Device ID String: "BMR-2NDP-ATCA"

Site Type: 0x00, Site Number: 02
Current Power Level: 0x0d, Maximum Power Level: 0x8d, Current Power Allocation: 17.7 Watts

96: FRU # 1
Entity: (0x12, 0x6b)
Hot Swap State: M: Active, Previous: M: Activation In Progress, Last State Change Cause: Normal State Change (0x0)
Device Type: "FRU Inventory device behind management controller" (0x0a), Modifier 0x0
Device ID String: "BMR-2NDP-Chassis"
Current Power Level: 0x0d, Maximum Power Level: 0x8d, Current Power Allocation: 6.8 Watts

< : > BMR-2ND Firmware (v1.8.8)
< : > nVPE (c) Copyright 2004-2019.
< : > Perform Power-on Self Testing.
< : > POST is OK
< : > Reset type: hard, reset cause: power failure
< : > Device type: Zymp Ultrascale+
< : > Operating mode: normal
< : > Hardware address: 0x3B
< : > Setting RTC with the default date
< : > E-keying link #0 is deactivated
< : > E-keying link #1 is deactivated
< : > E-keying link #2 is deactivated
< : > FRU 0 state: M:0->M:1, cause = 0
< : > E-keying link #0 is deactivated
< : > E-keying link #1 is deactivated
< : > E-keying link #2 is deactivated
< : > FRU 1 state: M:0->M:1, cause = 0
< : > Buffer control activated
< : > FRU 0 state: M:3->M:2, cause = 2
< : > FRU 0 state: M:3->M:2, cause = 1
< : > E-keying link #0 is activated
< : > E-keying link #2 is activated
< : > E-keying link #1 is activated
< : > FRU 0 state: M:3->M:4, cause = 0
Calling hal_on_mainfru_power_on(), try release reset
In function: hal_on_mainfru_power_on()
hal_power_present() = 1
hal_power_powered() = 1
FPO powered and present
< : > Create backup of running image
< : > Backup Image size 008340080 CRC: 83DEF3AC
< : > FRU 1 state: M:3->M:2, cause = 2
< : > FRU 1 state: M:2->M:3, cause = 1
< : > FRU 1 state: M:3->M:4, cause = 0
IPMC software solution: OpenIPMC

- Free open source PICMG-compliant IPMC software
  - Implements the IPMI functions required by PICMG ATCA
    - Instantiate board sensors, declare them to ShM, read-out and publish data
    - Power negotiation and hot-swap (M-states, handle, etc.)
    - Focus on simplicity: optional functions can be added to the project by the user
  - Developed at SPRACE to run on the ZynqMP Management Module (KIT)
    - Platform-independent choices allowed to quickly extend to other architectures

- FreeRTOS operating system
  - Can run independent processes in parallel (w/ prioritization)
  - Flexible software development, thanks to process decoupling
  - Supported by many SoC manufacturers (TI, NXP, Xilinx, Microsemi…)
Hardware decoupling

- For each uController architecture, OpenIPMC needs a Hardware Abstraction Layer (HAL)
- HAL functions are passed to OpenIPMC to interface it with the low-level drivers
  - This enables hardware management (I2C multi-master mode, GPIOs readings, etc)

---

**Diagram:**

```
+-------------------+         +-------------------+
|                   |         |                   |
|  MCU              |         |  User can         |
|                   |         |  implement its    |
|                   |         |  own interface to |
|                   |         |  manage specific  |
|                   |         |  peripherals in   |
|                   |         |  the platform     |
+-------------------+         +-------------------+
| FreeRTOS runtime  |         |                   |
|                   |         | This layer allows  |
|                   |         |  the OpenIPMC to  |
|                   |         |  be attached to   |
|                   |         |  the platform,   |
|                   |         |  and is (ideally) |
|                   |         |  the only part    |
|                   |         |  that needs to be |
|                   |         |  adapted         |
|                   |         |                   |
| OpenIPMC          |         | The ipmc_iios     |
|                   |         |  interface can    |
|                   |         |  use the i2c      |
|                   |         |  functions in the |
|                   |         |  driver to run  |
|                   |         |  the IPMB         |
+-------------------+         +-------------------+
|                   |         | Board-Specific    |
|                   |         | Controls         |
|                   |         | Drivers           |
|                   |         | SoC Hardware      |
```

---

This layer allows the OpenIPMC to be attached to the platform, and is (ideally) the only part that needs to be adapted.

The ipmc_iios interface can use the i2c functions in the driver to run the IPMB.
Reference development platform: Ultra96 board

- We used this board at SPRACE for development
  - [https://www.96boards.org/product/ultra96/](https://www.96boards.org/product/ultra96/)

- Zynq Ultrascale+ ZU3EG
  - Same SoC family as ZynqMP Mgmt. Module
  - 4 Cortex A-53 + 2 Cortex R-5 + PL and peripherals
  - Plenty of tutorials & Vivado support
  - Excellent price (249$)
    - More boards for devs working on the project

- Why did we use HAL from the start?
  - IO config may vary considerably between boards
  - The HAL makes porting much easier
Reference development platform: full setup

- Ultra96 Dev board
- Pulsar-2b ATCA board
  - Exposes the IPMB-A and -B buses to DIMM slot
  - We are currently not reading P2B sensors
- Mini-DIMM adapter
  - Connects Pulsar-2b DIMM slot to Ultra96
  - Translates 1.8 V (Ultra96) ↔ 3.3V (ATCA)
  - Design and manufacture by Luis Ardila (KIT)
- Comtel CO6 ATCA chassis
  - Full-mesh, 6 slots horizontal
  - 2 PigeonPoint ShelfManagers (redundant)
Development Platform: Ultra96

Digilent Analog Discovery USB o’scope as I2C logic analyzer

Monitoring IPMB-A with TeK o’scope
OpenIPMC tests on Trenz + Serenity setup @ KIT

- OpenIPMC code was successfully ported and tested in **Trenz+Serenity** setup at KIT

- All changes relays into the `ipmc_ios.c` file: the physical layer of OpenIPMC

- Hot-Swap operation were successfully performed on Serenity board

- Since no real sensor are currently being read in this hardware
OpenIPMC tests on Trenz + Serenity setup @ KIT

Activation Status

```bash
# clia fru -v 96
Pigeon Point Shelf Manager Command Line Interpreter
96: FRU # 0
  Entity: (0xb0, 0x1)
  Hw Swap State: M4 (Active), Previous: M3 (Activation In Process), Last State Change Cause: Normal State Change (0x0)
  Device ID String: 'Trenz-Serenity'
  Site Type: 0x00, Site Number: 02
  Current Power Level: 0x02, Maximum Power Level: 0x02, Current Power Allocation: 100.0 Watts
```

FRU Information  (testing data from example code)

```bash
# clia fruinfo 96 0
Pigeon Point Shelf Manager Command Line Interpreter
96: FRU # 0, FRU Info
Common Header: Format Version = 1
Board Info Area:
  Version = 1
  Language Code = 25
  Mfg Date/Time = Oct 1 00:00:00 2019 (12490560 minutes since 1990)
  Board Manufacturer = SPRACE - KIT
  Board Product Name = OpenIPMC @ Trenz-Serenity
  Board Serial Number = 180189981-18998
  Board Part Number = AA00Y99
  FRU Programmer File ID = 01
```

Sensor Reading  (testing data from example code)

```bash
# clia sensordata 96 3
Pigeon Point Shelf Manager Command Line Interpreter
96: LUN: 0, Sensor # 3 ("FPGA TEMP")
  Type: Threshold (0x01), "Temperature" (0x01)
  Belongs to entity (0x00, 0x60)
  Status: 0xc0
  All event messages enabled from this sensor
  Sensor scanning enabled
  Initial update completed
  Raw data: 50 (0x32)
  Processed data: 50.00000000 degrees C
  Current State Mask: 0x00
```
OpenIPMC portability exercise: ESP32

- ESP32 microcontroller (see backup slides)
  - Very different from a Zynq US+
- Questions answered by this test
  - Architecture independency?
    - Trivial, thanks to C and FreeRTOS
  - Ease of integration on a different SoC?
    - OpenIPMC needs I2C peripheral
    - Many SoCs have 2 or more
  - Effort needed to port OpenIPMC?
    - Mainly IO/HAL interface bindings
    - Fixes needed in ESP32 IDF (see backup)
    - Porting took just 3 person-weeks :-)
- Overall the exercise was a success
- Repo: [gitlab.com/openipmc/ipmc-esp32](https://gitlab.com/openipmc/ipmc-esp32)
OpenIPMC tests on ESP32

- IPMBus communication works
- ShM happily accepts the FRU
- Activation and deactivation are triggered using an ‘improvised’ Handle Switch
- The activation time is significantly longer when compared with the operation using the Ultra96 board
Summary

- **ZynqMP Management Module requires customized IPMC software**
  - Pigeon Point provides a commercial IPMC solution
  - OpenIPMC is a FOSS solution for the same application

- **OpenIPMC project**
  - Portable to any architecture with FreeRTOS support (w/ I2C periph.)
  - Fully customizable, simple, easy to debug
  - Templates for sensor declaration in the examples
  - GitLab repository [here](#) and Doxygen documentation [here](#)

- **Successfully tested on 3 different hardwares**
  - Ultra96 (Zynq UltraScale+, ARM Cortex-R5)
  - Serenity board with Trenz TE0803 mezzanine (Zynq UltraScale+, ARM Cortex-R5)
  - ESP32 (Tensilica Xtensa LX6)
  - Can be easily ported to other archs (SmartFusion? MSP430? STM32? LPC1700? ...)
Questions?
Backup Slides
OpenIPMC Internal Concept

**ipmc_ios.c**

- **IPMB-A**
- **IPMB-B**

Buffers the messages
- Collects income IPMI messages (Requests and Responses)
- Manage the transmission over the IPMB channels (arbitration)

**ipmb_0.c**

- **IPMB-A**
- **IPMB-B**

Physical link layer
- Multi-master I²C
- Use 2 PS I²C controllers

**ipmi_request_manager.c**

- **INCOMING MESSAGES**
  - Responses
  - Requests
  - task

- **OUTGOING MESSAGES**
  - Requests
  - Responses
  - task

- **LOCAL REQUESTS**
  - post_ipmi_request()
  - task

- **REMOTE REQUESTS**
  - solve_ipmi_request()
  - task

“Application layer”
- Hot Swap operation
- Power Negotiation
- Sensor Records
- Sensor Readings
- Other IPMI functions
  - user-implemented

Abstracts away the IPMI transport layer
- Manages sequence #, destination and checksums
- Associates responses to requests
- Retries and timeouts
- Accepts multiple internal requests (from different tasks)
- Call the specific functions to solve external requests
OpenIPMC on ESP32 (Espressif Systems, CN)

- Quite powerful & flexible uC
  - 240 MHz Xtensa LX6 dual core
  - FPU, Big INTs & Crypto
  - WiFi, BT, SPI, I2C, UART…
  - FreeRTOS support

- Cheap Linux-supported boards
  - CP2102 USBtoUART converter
  - Boards sell for 5$

- 3.3 V device (same as IPMB)

- Development software
  - Arduino IDE, PlatformIO or esp-idf

- Very different arch w.r.t a Zynq US+
  - Good exercise on portability/
Fixes to ESP32 Integrated Development Framework

- I2C multi-master with variable size I2C msgs is required for IPMB bus communication
  - End of message is signalled by a stop bit
- The official esp-idf I2C driver was not supporting variable size msgs correctly
  - The driver expected a message size to be specified in advance
- We modified the driver* and now the slave read function correctly returns if receiving a stop bit

*The github esp-idf repository was forked. A merge request is still in progress
ZynqMP-IPMC ATCA Test Board

- modified stencil + glue allowed all TPS82130 to operate as expected
- one PCB fully assembled and operational used to bring up vivado project for future OpenIPMC & PP-IPMC software migration
- two more PCBs assembled, but one not working, needs rework on problematic regulators

https://gitlab.cern.ch/p2-xware/hardware/kicad-pcbs/atca/atca_zynqmp
Pigeon Point - Performed Tests

- **IPMC**
  - Power-up
  - Com. to shelf manager
  - State change to M4
  - Sensor readings
  - Activation/Deactivation
  - Handle Switch
  - Cold Reset
  - Trigger Linux boot

- **IPMC compliance test**
  - 137 tests executed (most automatic)
  - failing tests were expected to fail because of missing functionality

<table>
<thead>
<tr>
<th>Pass</th>
<th>Fail</th>
<th>Error</th>
<th>Abort</th>
<th>Skip</th>
<th>Inconclusive</th>
</tr>
</thead>
<tbody>
<tr>
<td>79</td>
<td>34</td>
<td>0</td>
<td>0</td>
<td>24</td>
<td>0</td>
</tr>
</tbody>
</table>

- **Linux**
  - Start U-boot
  - Network stack in U-boot
  - Load Linux kernel via tftp
  - Start Linux (login shell)
  - Rootfs
    - Initramfs
    - NFS
    - Sdcard
  - Network connection
  - SSH to ZynqMP
  - SSH from ZynqMP
  - CENTOS rootfs
  - Yum install