An open-source, ZYNQ-based IPMC solution

13th xTCA Interest Group Meeting

Marcelo Vicente (University of Wisconsin-Madison) on behalf of the ATCA APx Consortium
The ZYNQ-IPMC project

- Develop a highly versatile IPMC platform for ATCA applications based on Xilinx ZYNQ SoC:
  - Exploit fast ARM Cortex-A9 processor and high FPGA parallelism for time-critical decisions
  - Support for a high number of peripherals (power controllers, fast ADCs, DACs, temperature sensors, RTMs, JTAG, etc.)
  - Previous experience with ZYNQ designs
  - **Open-source** and **NDA-free** (both hardware and software)
ZYNQ-IPMC Mezzanine

- 244-pin LP miniDIMM form factor (82mm x 30mm, 1mm thickness)
- Pinout similar to other IPMCs using the 244 DIMM form factor
- GPIOs can be configure to standard or custom interfaces: I2C, SPI, UART, MMC, XVC, etc.
  - Facilitates routing and layout of carrier PCB due to the extra flexibility
Hardware Overview

ZYNQ-7020

Processing System (PS)

Dual Cortex-A9 @ 666 MHz

Firmware Acceleration

Programming Logic (PL)

256MB DDR3-1066

32MB Flash

256Kb EEPROM

1Gb Ethernet PHY

BASE-T Transformer

8x 16-bit 1kS/ch/s ADC

DIMM connector

16 ADC Channels

108 GPIOs

Hardware Address

IPMB-A (I2C)

IPMB-B (I2C)

RGMII

QSPI

SPI

DDR
• High-level realtime framework:
  • **FreeRTOS**: Per-service threads allow clean separation of code and prioritization of critical tasks
  • **C++**: Object-Oriented structure allows clean encapsulation of different modules and subcomponents and easy customization for any number of instances
  • **lwIP**: Provides easy to use TCP/IP for various network services. (console, upgrades, XVC, monitoring, etc.)
  • **Services**: Service based framework allows easy coordination & management of many varied tasks
Software Framework (2)

- Easy to add new features or services by writing and instantiating a task thread.
- Publisher/Subscriber notifications & message delivery for tasks
- Granular logging configures per-service verbosity at runtime
- Watchdog protection configurable at a per-task level for specific needs
- PL-supported tasks for realtime power, sensor, hardware management, XVC & more
Software Framework (3)

- Easy reconfiguration for similar application by rearranging existing component with OO & inheritance

```java
PS_SPI *ps_spi0 = new PS_SPI(XPAR_PS7_SPI_0_DEVICE_ID, XPAR_PS7_SPI_0_INTR);
eeprom_data = new SPI_EEPROM(*ps_spi0, 0, 0x8000, 64);
eeprom_mac = new SPI_EEPROM(*ps_spi0, 1, 0x100, 16);
persistent_storage = new PersistentStorage(eeprom_data, LOG["persistent_storage"], SWDT);
persistent_storage->register_console_commands(console_command_parser, "eprom.");
```

- Base services (terminal, IPMB communication, storage, etc.) included as primary components of the framework
Firmware and software expansion go hand-in-hand:

- Using Xilinx IP packager in Vivado for AXI integration
- IPs include firmware blocks and low-level drivers

1. Add firmware IP or custom VHDL in block design

- Example of custom IP: Management Zone Control
- Aggregates and manages related power trees
- IP is configurable through GUI, driver will automatically adapt
- Same procedure for Xilinx IPs and other custom IPs
2. Use provided C++ drivers in framework or use expandable templates

- Example C++ driver for Management Zone Control

```cpp
/**
 * A single management zone.
 */
class MGMT_Zone {
public:
    const u16 DeviceId; ///< The DeviceId of the controller hosting this zone.
    const u32 MZNo;    ///< The MZ number within the MZ Controller
    MGMT_Zone(u16 DeviceId, u32 MZNo);
    virtual ~MGMT_Zone();

    virtual void set_power_state(PowerAction action);
    virtual bool get_power_state(bool *in_transition=NULL);

};
/**/
```

- Drivers are initialized on dedicated function in framework

```cpp
void driver_init(bool use_pl) {
    /**/
    // Create required instances for management zones
    for (int i = 0; i < XPAR_MGMT_ZONE_CTRL_0_MZ_CNT; ++i)
        mgmt_zones[i] = new MGMT_Zone(XPAR_MGMT_ZONE_CTRL_0_DEVICE_ID, i);
    /**/
}
```
Test platform (1)

• Controller Development Board (CDB)
  • Development platform for several key technologies that will be used on other blades
  • Test platform for IPMCs and MMCs with Shelf Manager connectivity via the backplane

• Several IPMC peripherals available:
  • IPMB-A and IPMB-B
  • I2C, SPI, MDIO
  • Ethernet Switch Module
    • See backup
  • etc..
Opportunities to collaborate

• Framework under GPL license:
  • Full rights to customize, reuse or modify, so long as software changes are shared
  • No license associated with hardware design

• Initial public GitHub release expectancy:
  • Hardware design in Altium Design: < Q4 2018
  • Software framework & IPMC package: < Q4 2018

Interested peers are welcome to get in touch with us:
marcelo.vicente@cern.ch

We will reach out when the project is released

Thank you!
Backup
• Pooling of efforts in ATCA Processor hardware, firmware and software development
  • Present efforts include CMS Phase-2 Calorimeter, Correlator and Muon Triggers; ECAL Barrel, CSC and GEM readouts.

• Multiple ATCA processors and mezzanine board types

• Modular design philosophy, emphasis on platform solutions with flexibility and expandability

• Reusable circuit, firmware and software elements
Ethernet Switch Module (ESM)

- Compact 6-port 1Gb ethernet switch module (40mm x 35mm)
  - 4x BASE-T and 2x SGMII interfaces available
- Unmanaged with low to no maintenance required
- Monitoring via UART and status LEDs provided by serial-out interface
- Circuit fully validated on CDB
- Available by Q3 2018