# openMMC
## A modular open source firmware for MMCs

10th meeting of the xTCA interest group

**Henrique Aires Silva**

Brazillian Synchrotron Light Source (LNLS/Sirius)

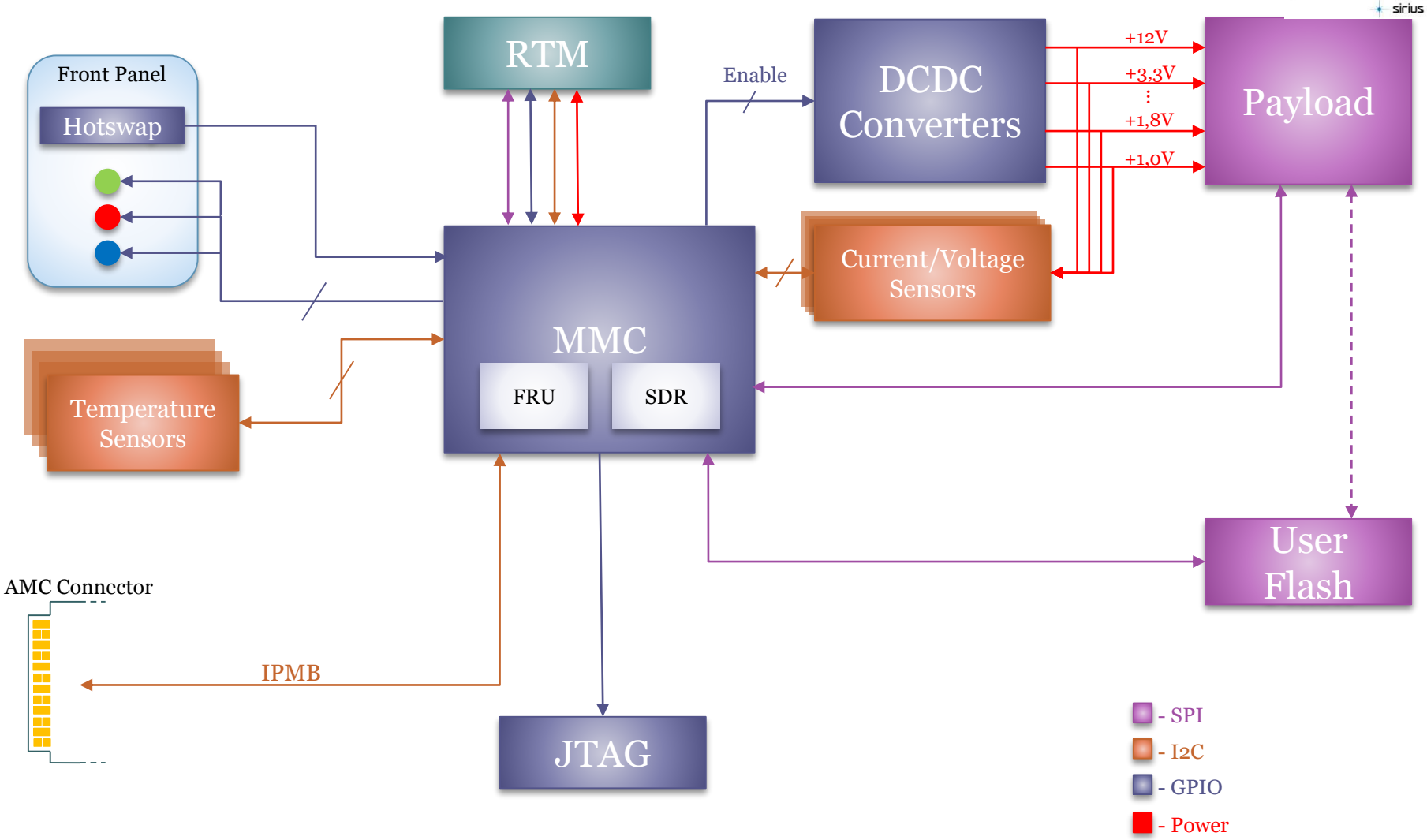
CNPEM

LNLS
sirius

# Outline

- The MMC in a uTCA system
- Existing Implementations
- Why a new design?
- openMMC Firmware
  - Features
  - Structure
- Porting Example
- Integration Tests
- Future Developments

# The MMC in a uTCA system



# Existing Implementations

## DESY MMC
- ATMega128

## CERN MMC
- ATMega128

## N.A.T. Europe
- ATMega128

## coreIPM
- LPC2xxx
- Cypress nvPSoC

## University of Winsconsin
- AT32UC3A1512

## Warsaw University of Technology
- LPC1764

## COSYLAB
- LPC2136

## JAMMCI (GSI)
- LPC1764
- MATPEX 1A
- ATMega128

# Why a new design?

- Difficult maintenance with spread forks
- Incompatibility between the MMC and MCHs from different vendors
- Hard to port the code to different boards/controllers
  - Microcontroller specific code mixed with application routines
  - IDE-dependent compilation/debug


# OpenMMC

# Features

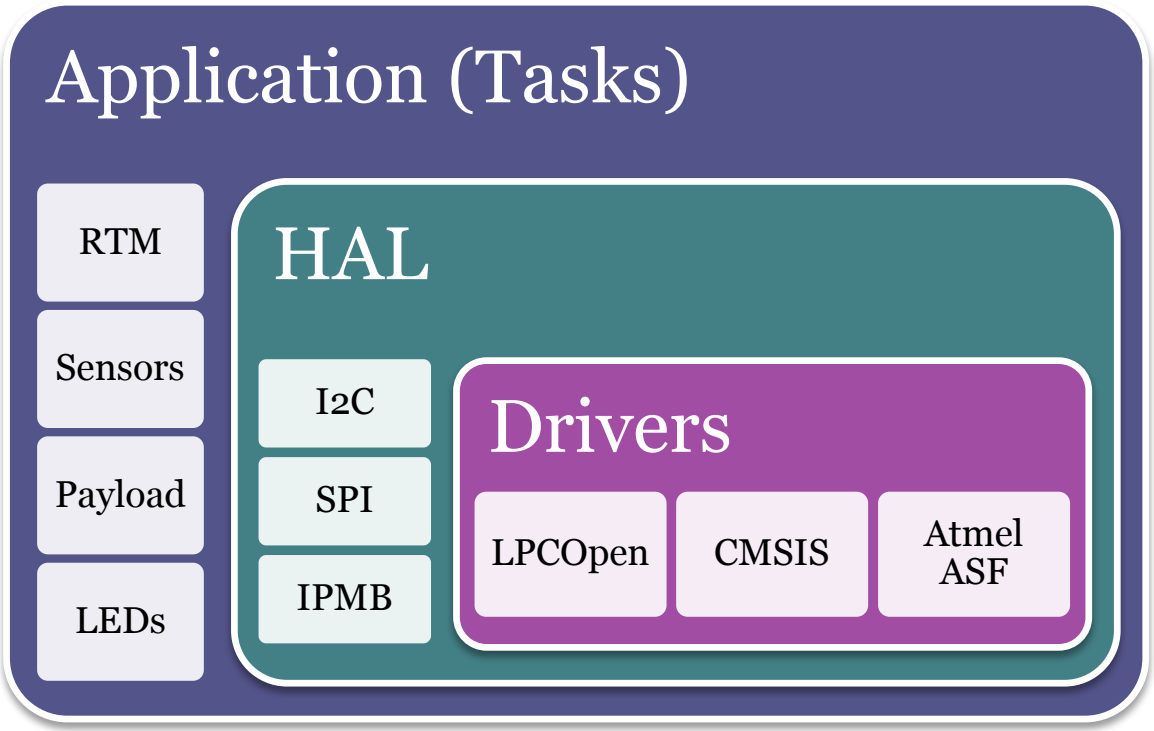
- Open Source – GPLv3 (All code hosted at GitHub)
- Modular structure
  - Chip specific code separated from application
  - Independent modules
- CMake build system
- All board ports in the main repository
- HPM.1
- Doxygen documentation (in progress)

*Initial collaboration between LNLS (Brazil) and GSI (Germany)

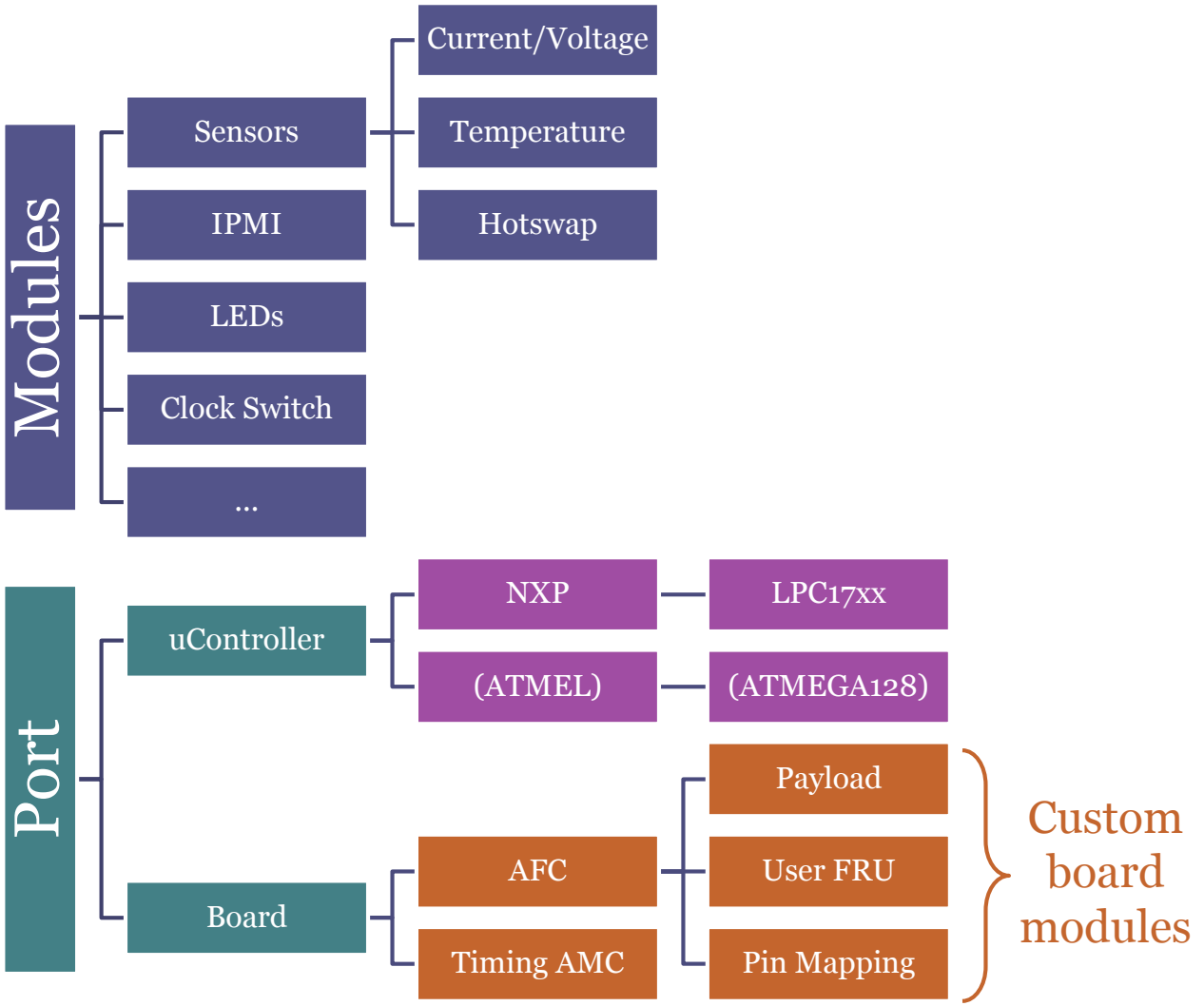
# FreeRTOS


- Multitasking
  - Preemptive scheduler
- Inter-task communication
  - Queues
  - Binary and counting semaphores
  - Recursive mutexes
  - Task notifications
- 6K to 12K ROM footprint
- Numerous ports ( > 100 )
  - All maintaned and supported by Real Time Engineers!

# Code Structure



# Code Structure



# Code Structure

## Application – Custom task

```
gpio_set_pin_state( GPIO_DAC_VADJ_RST_PORT, GPIO_DAC_VADJ_RST_PIN, LOW);
```

## Port layer – Interface between driver and application

```
#define gpio_set_pin_state( port, pin, state ) ioport_set_pin_level( pin, state )
```

## Drivers – Custom hardware control

```
static inline void ioport_set_pin_level(ioport_pin_t pin, bool level)
{
        arch_ioport_set_pin_level(pin, level);
}
```

**ATMEL ASF**

# Code Structure

## Application – Custom task

```
gpio_set_pin_state( GPIO_DAC_VADJ_RST_PORT, GPIO_DAC_VADJ_RST_PIN, LOW);
```

## Port layer – Interface between driver and application

```
#define gpio_set_pin_state( port, pin, state ) Chip_GPIO_SetPinState( LPC_GPIO, port,...
```

## Drivers – Custom hardware control

```
STATIC INLINE void Chip_GPIO_SetPinState(LPC_GPIO_T *pGPIO, uint8_t port, uint8_t pin, bool setting)
{
        if (setting) {  /* Set Port */
                pGPIO[port].SET |= 1UL << pin;
        }
        else {  /* Clear Port */
                pGPIO[port].CLR |= 1UL << pin;
        }
}
```

LPCOpen

# Porting Example

Different boards

AFC

```
#List all modules used by this board
set(AFC_MODULES
  "FRU"
  "PAYLOAD"
  "SDR"
  "WATCHDOG"
  "JTAG_SWITCH"
  "CLOCK_SWITCH"
  "FPGA_SPI"
  "DAC_AD84XX"
  "HOTSWAP_SENSOR"
  "TEMPERATURE_SENSOR"
  "VOLTAGE_SENSOR"
  "HPM"
  )
```

TimingAMC

```
#List all modules used by this board
set(TIMINGAMC_MODULES
  "FRU"
  "PAYLOAD"
  "SDR"
  "WATCHDOG"
  "PLL_CTRL"
  "WHITE_RABBIT"
  "HOTSWAP_SENSOR"
  "TEMPERATURE_SENSOR"
  "VOLTAGE_SENSOR"
  "HPM"
  )
```

# Porting Example

```
#define gpio_init()                          Chip_GPIO_Init( LPC_GPIO )
#define gpio_read_pin( port, pin )           Chip_GPIO_GetPinState( LPC_GPIO, port, pin )
#define gpio_read_port( port )               Chip_GPIO_GetPortValue( LPC_GPIO, port )
#define gpio_set_pin( port, pin )            Chip_GPIO_SetPinOutHigh( LPC_GPIO, port, pin )
#define gpio_set_port( port, mask )          Chip_GPIO_SetPortOutHigh( LPC_GPIO, port, mask )
#define gpio_clr_pin( port, pin )            Chip_GPIO_SetPinOutLow( LPC_GPIO, port, pin )
#define gpio_clr_port( port, mask )          Chip_GPIO_SetPortOutLow( LPC_GPIO, port, mask )
#define gpio_pin_toggle( port, pin )         Chip_GPIO_SetPinToggle( LPC_GPIO, port, pin )
#define gpio_set_pin_state( port, pin, state ) Chip_GPIO_SetPinState( LPC_GPIO, port, pin, state )
#define gpio_set_pin_dir( port, pin, dir )   Chip_GPIO_SetPinDIR( LPC_GPIO, port, pin, dir )
```

**Driver function aliasing**

```
set(TARGET_MODULES
  "FRU"
  "PAYLOAD"
  "SDR"
  "WATCHDOG"
  "JTAG_SWITCH"
  "CLOCK_SWITCH"
  "FPGA_SPI"
  "DAC_AD84XX"
  "HOTSWAP_SENSOR"
  "TEMPERATURE_SENSOR"
  "VOLTAGE_SENSOR"
  "HPM"
```

**Chip**

- Driver functions
- FreeRTOS port for the controller

**Module**

- Develop new managing / monitoring tasks

**Board**

- Define which modules will be used

**FreeRTOS port**

- ARM7_AT91FR40008
- ARM7_AT91SAM7S
- ARM7_LPC2000
- ARM7_LPC23xx
- ARM_CA9
- ARM_CM0
- ARM_CM3
- ARM_CM3_MPU

```
void vTaskPayload(void *pvParameters)
{
    payload_state state = PAYLOAD_NO_POWER;
    payload_state new_state = PAYLOAD_STATE_NO_CHANGE;

    uint8_t P12V_good = 0;
    uint8_t P1V0_good = 0;
    uint8_t FPGA_boot_DONE = 0;
    uint8_t QUIESCED_req = 0;

    uint8_t current_message;
```

# Integration tests

NAT-MCH-PHYS

⇨ Fully functional!

UTC002

⇨ Mostly functional
(Few bugs on FRU shutdown)

⇨ Q3/2016 (Planned)

*All tests with AFC boards

# Future developments

- ✓Bootloader upgrade
  - Integrate HPM module
- ✓Code documentation
  - 06/2016
- ✓Detailed wiki page on GitHub

# Try out!

- Already ported to ATMega and LPC17 families
- Easy to customize
- Large set of tools provided by FreeRTOS to ease development
- Expand the project to other boards and controllers

- Encourage collaboration
  - Open to pull-requests!

openMMC repository
https://github.com/lnls-dig/openmmc/


# Thank you!

**Henrique Aires Silva**

henrique.silva@lnls.br

openMMC repository
https://github.com/lnls-dig/openmmc/

Critics and suggestions are more than welcome!

Brazillian Synchrotron Light Source (LNLS/Sirius)

*10th meeting of the xTCA interest group*