IHEP

An implementation of module management controller for MicroTCA data processing system

Cong He, Jie Zhang, Xiaoshan Jiang (IHEP)

The 2nd MTCA/ATCA Workshop for Research and Industry

24/08/2021

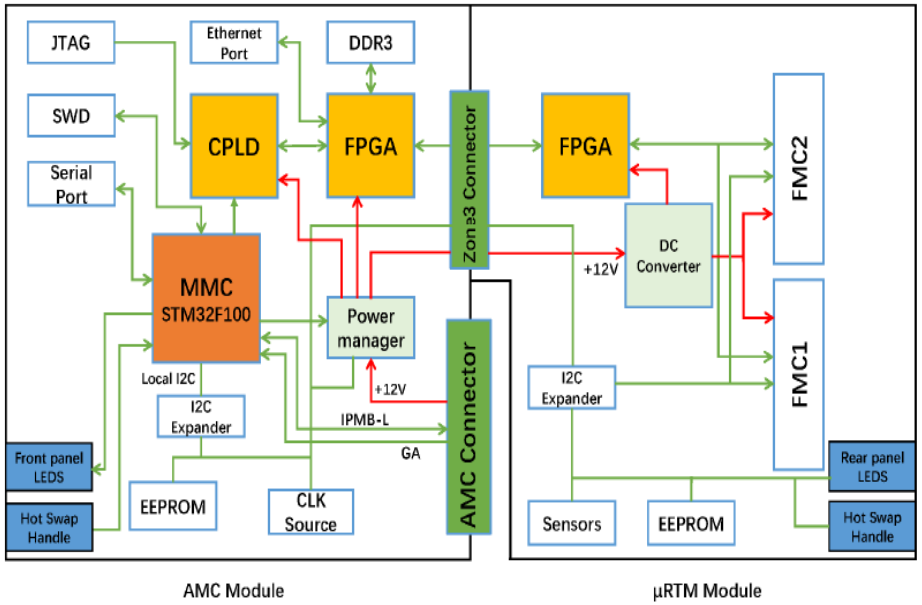
Contents

- Basic hard-firm structure & functions
- FreeRTOS & real-time performance test
- Application example

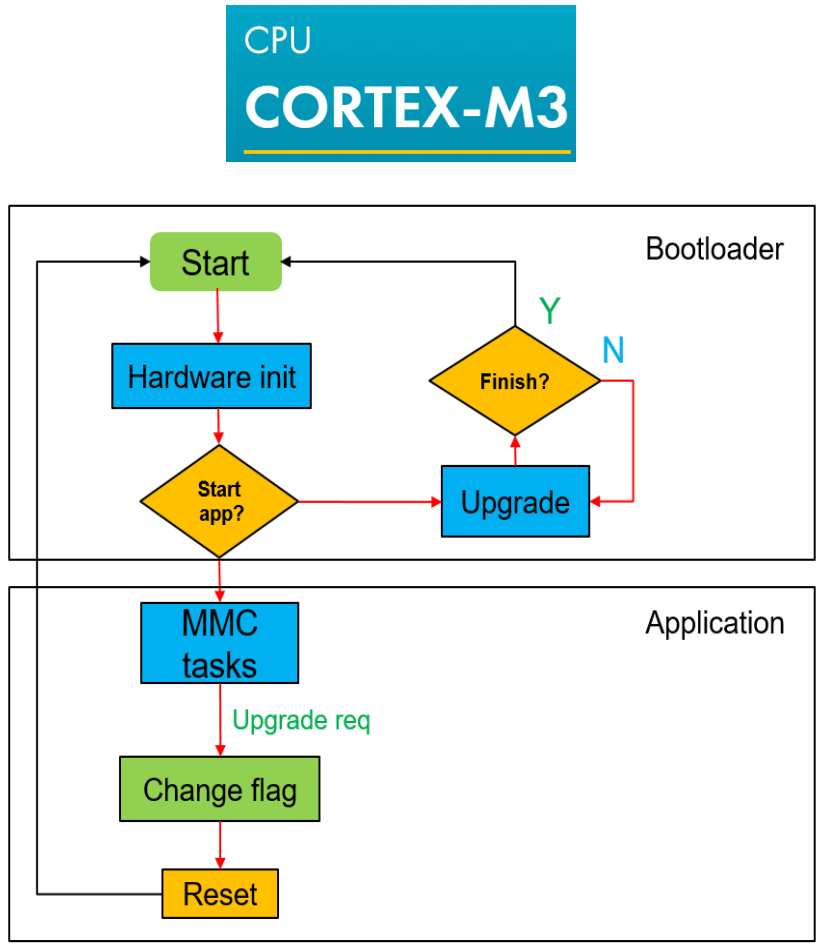
Basic hardware-firmware structure

Microcontroller tasks

- Hot-swap
- Programing FPGA and clock source
- RTM management
- Power management & monitoring
- Serial command support
- Remote firmware upgrade



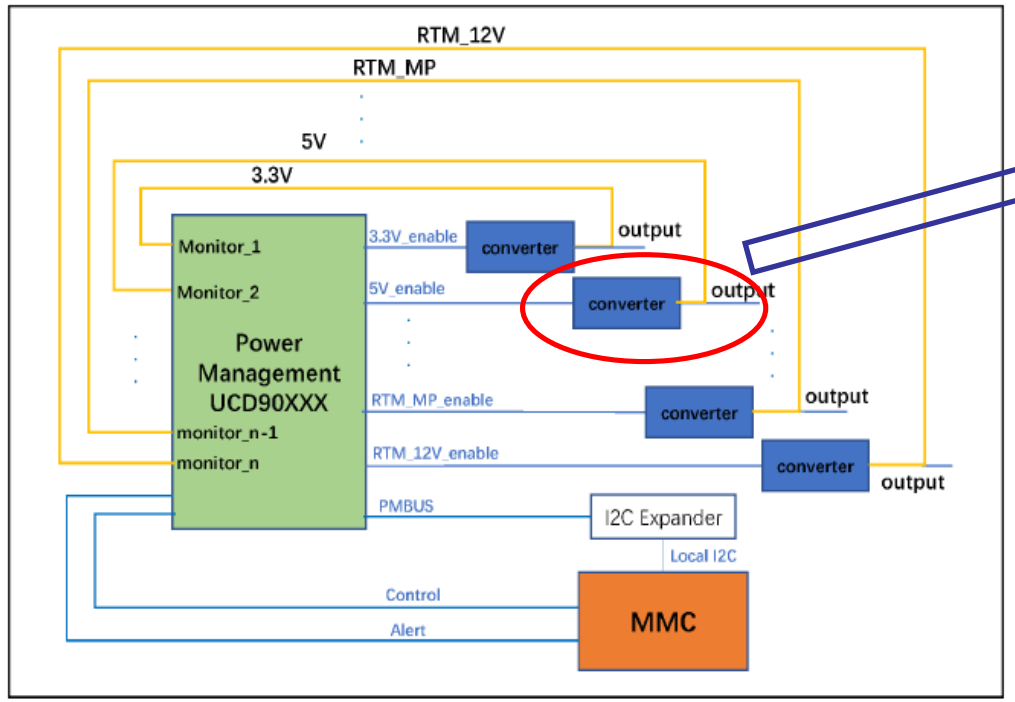
Hardware block diagram



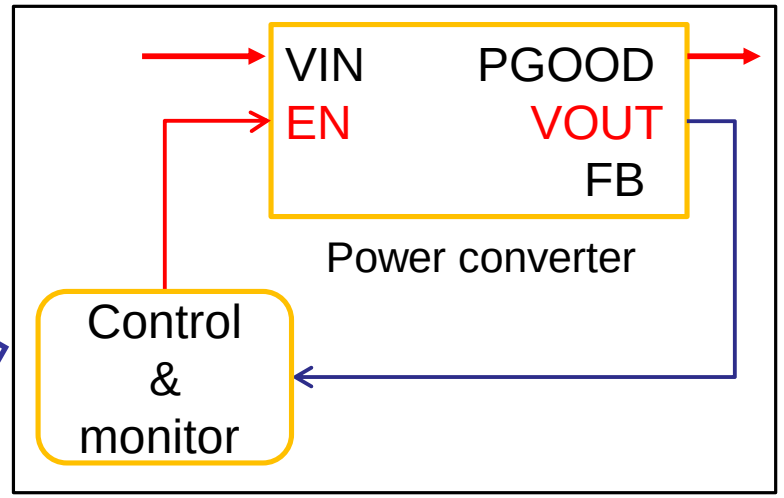
MCU firmware start-up flow

Reference：CERN MMC
https://espace.cern.ch/ph-dep-ESE-BE-uTCAEvaluationProject/MMC_project/default.aspx

Functions -Power management & Monitoring



Voltage Rails managements & monitoring



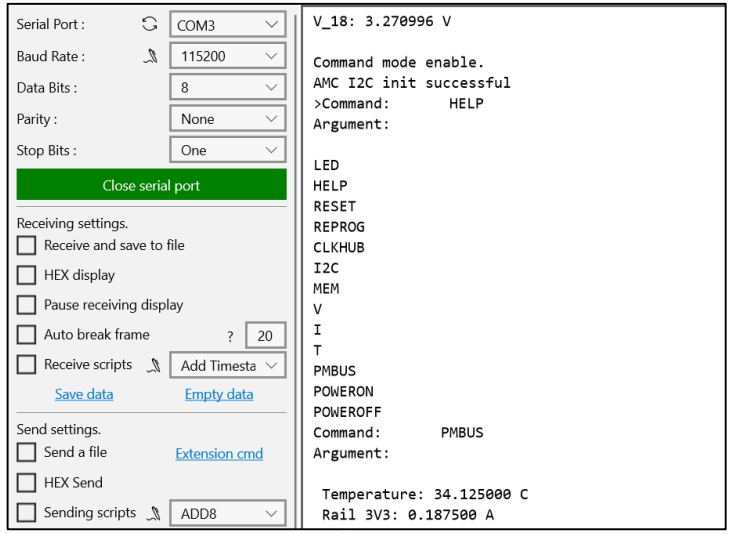
Enable the voltage rail, sample the VOUT and compare with threshold value.
Vout in reasonable range, PGOOD set.

Monitor all main voltage rails by hardware mechanism

Faster response to voltage abnormality

Provide alert signal to controller

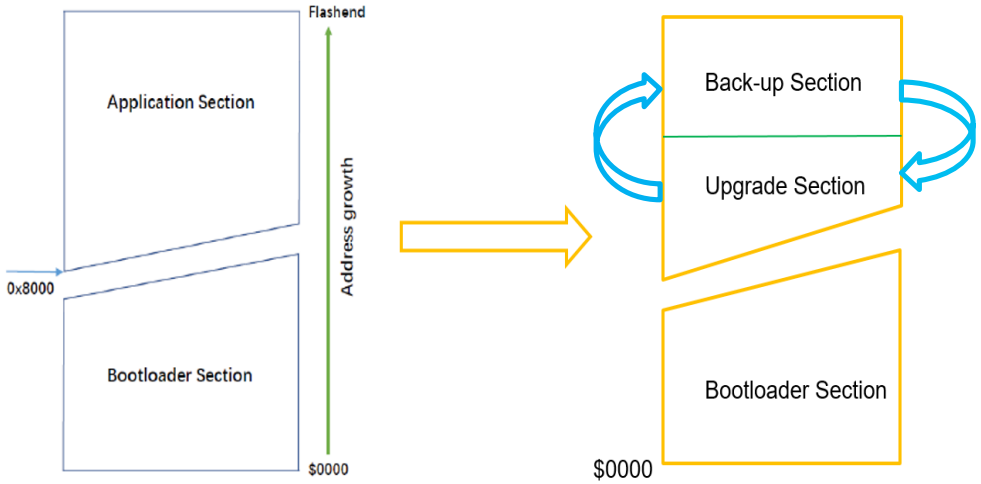
Functions- Serial command support & Remote firmware upgrade



Type in specified command for debug

Current supported commands:

- Read or revise system clock configuration
- Activate or deactivate payload voltage of RTM
- Reset the MCU or FPGA
- Program the FPGA
- Turn on or off the DC-DC converter
- Read or revise content of EEPROM
- Print voltages, currents and temperatures info



- Back-up section is divided to enhance the security of remote upgrade
- Position of back-up section and upgrade section is variable

RTOS?

Why RTOS is needed?

- Higher Operation efficiency than polling mode
- Quicker reaction to vital event
- Easier to extend functions

Why select freeRTOS?

RTOS: µCOS, RT-Thread, VxWorks, ThreadX

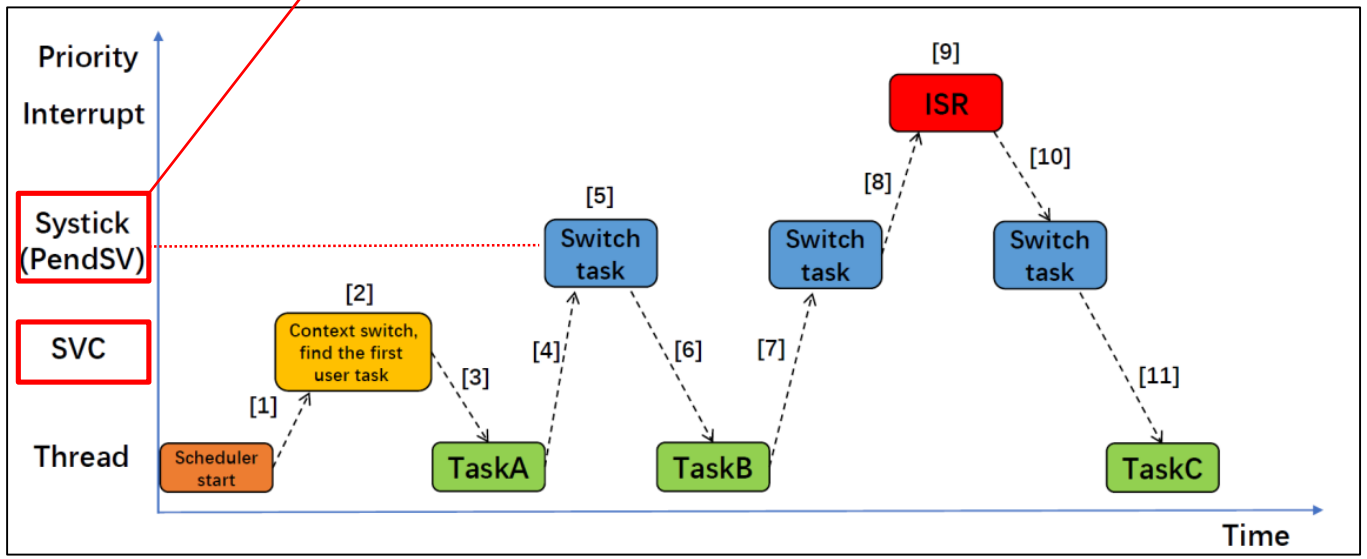
- Free & open-source
- Portable & cuttable
- Learning cost (time)
- Lightweight, less resource needed

Polling vs FreeRTOS

```
int main(void)
{
    while (1)
    {
        func1();
        func2();
        func3();
        ...
        funcn();
    }
}
```

Example code of polling

- PendSV exception:the lowest priority for task switch
- Triggered by systick or portYIELD

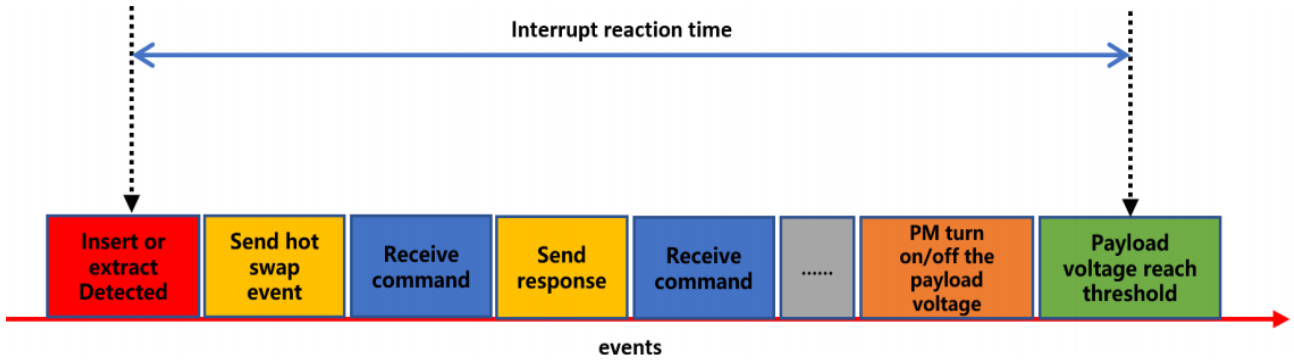


Basic principle of task switch in FreeRTOS

- In polling, code runs in a given order
- No priority

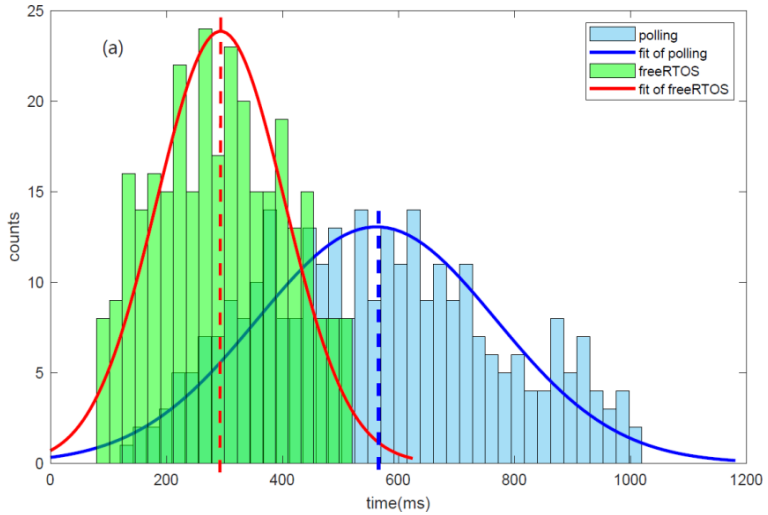
- Execution order of task according to priority
- The highest priority ready task executes firstly
- Higher level exception can interrupt the task switch

Real-time performance test



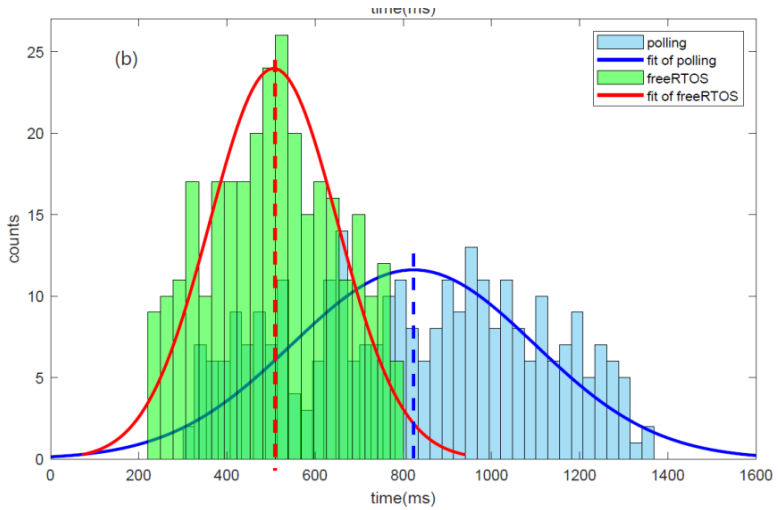
Interrupt reaction time starts counting from the detection of Insert/extract to payload voltage reach threshold

- Insertion : 580 →280 ms, a reduction of 52%
- Pulling out : 800 → 500 ms, a reduction of 37%



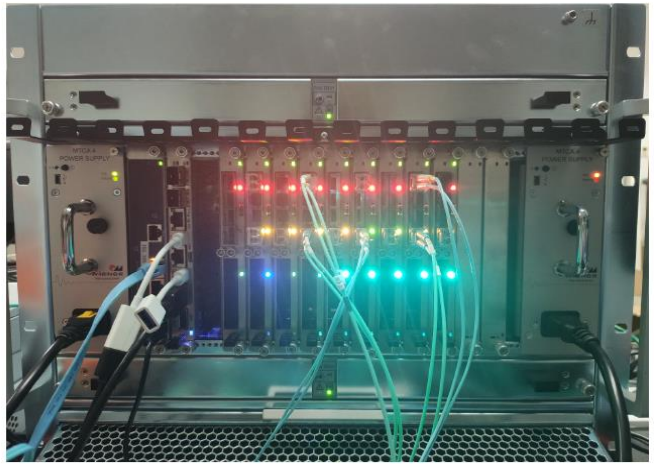
FreeRTOS
Polling

interrupt reaction time for insertion



interrupt reaction time for pulling out

Application example

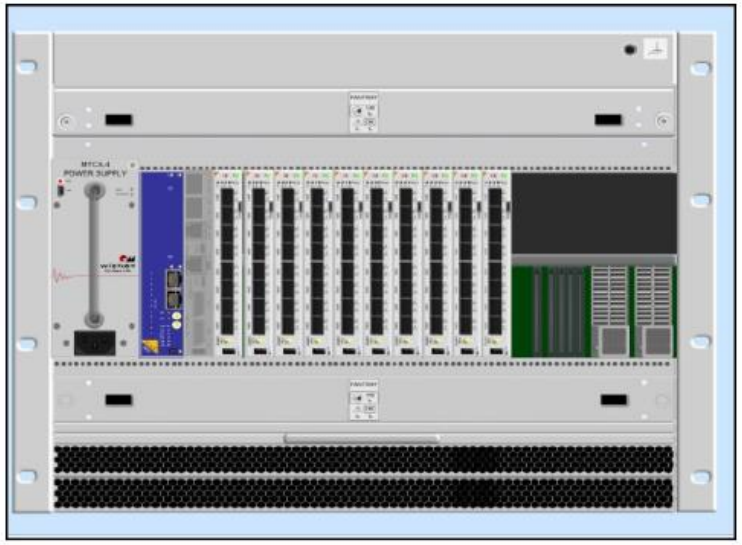


A practical application

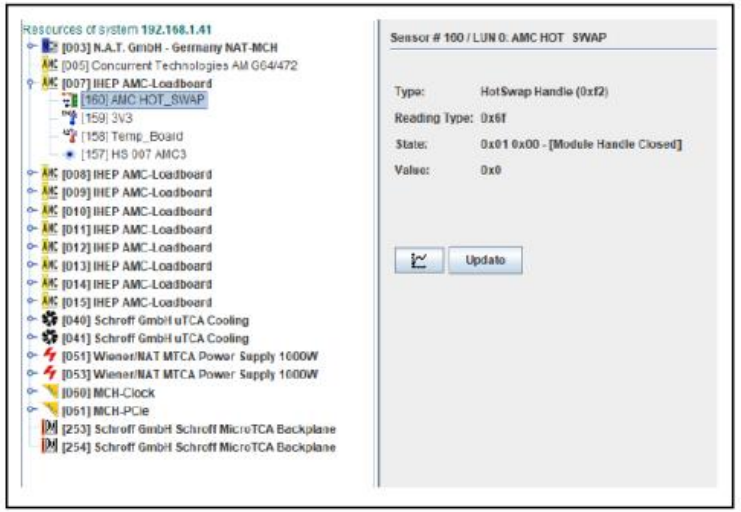
Device name	Manufacturer	Product Model
Chassis	N.A.T.	NATIVE-R9
Power Module	WIENER	MTCA 4.0 1000W Power Supply
MCH Module	N.A.T.	NAT-MCH-PHYS
AMC module	IHEP	UFC-V2
RTM Module	IHEP	RTM-M2

MicroTCA crate setup

The design has been applied in high energy photon source (HEPS) bandwidth upgrade project.



(a)



(b)

View by natview (NAT company)

Summary

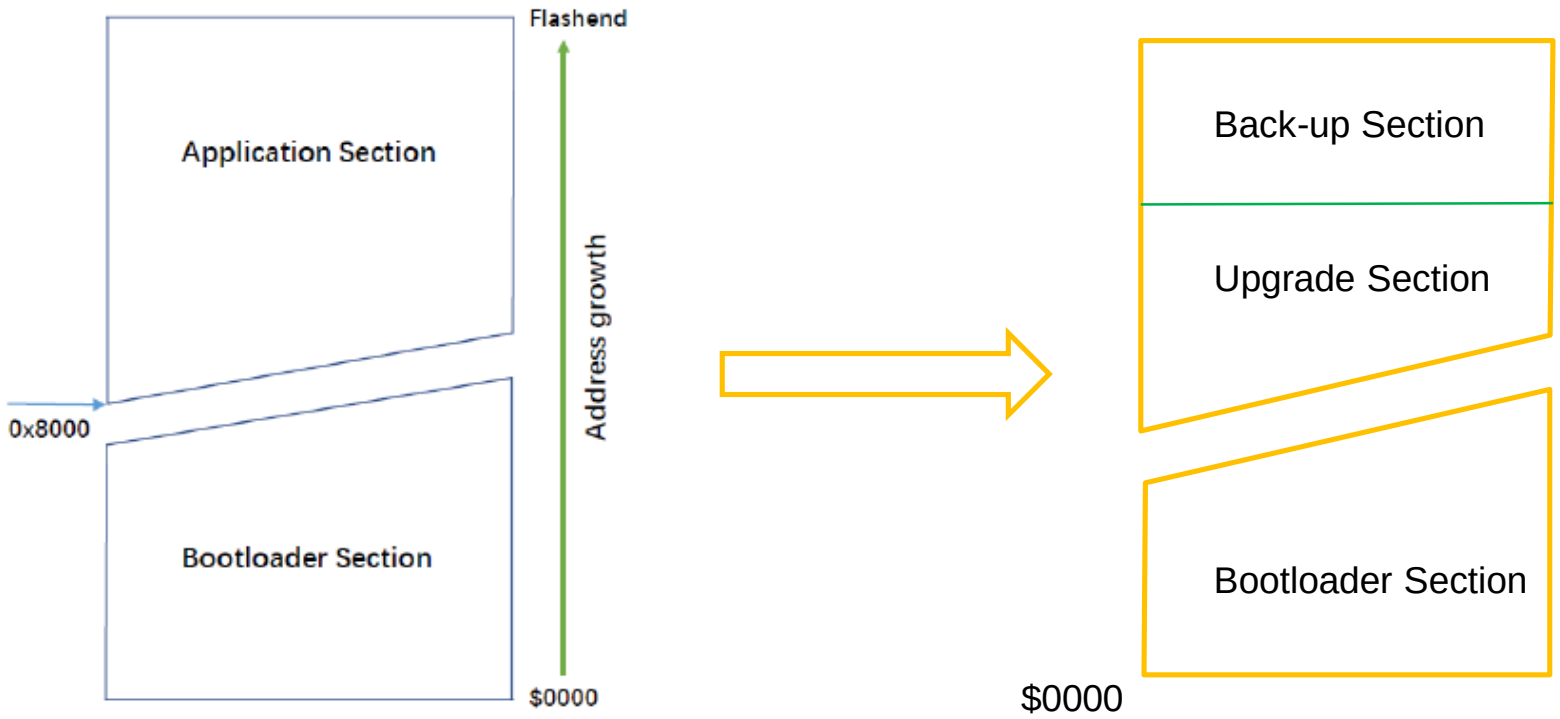
- In limited microcontroller resources, the solution RTOS-MMC V1.0 implements all necessary functions of MMC with FreeRTOS.
- Some improvements in hardware and firmware design.
- By the use of FreeRTOS, the real-time performance is improved. The design can provide a better guarantee for the rapid reaction to important events.

RTM & AMC

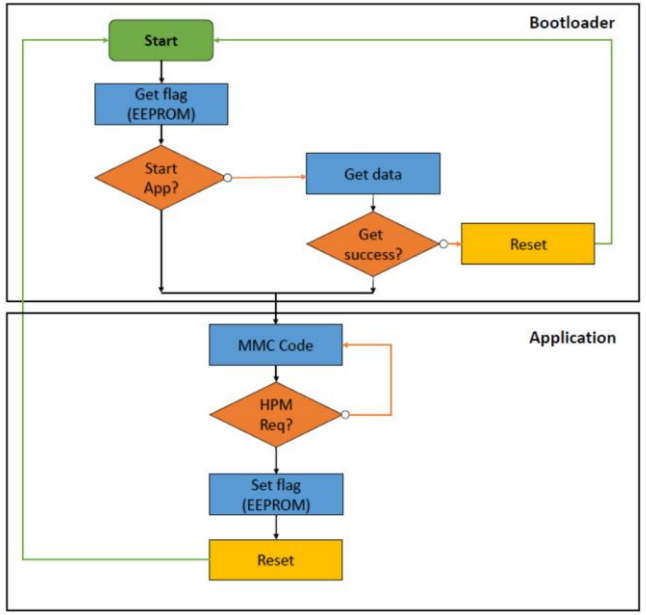
THANK YOU FOR YOUR ATTENTIONS!

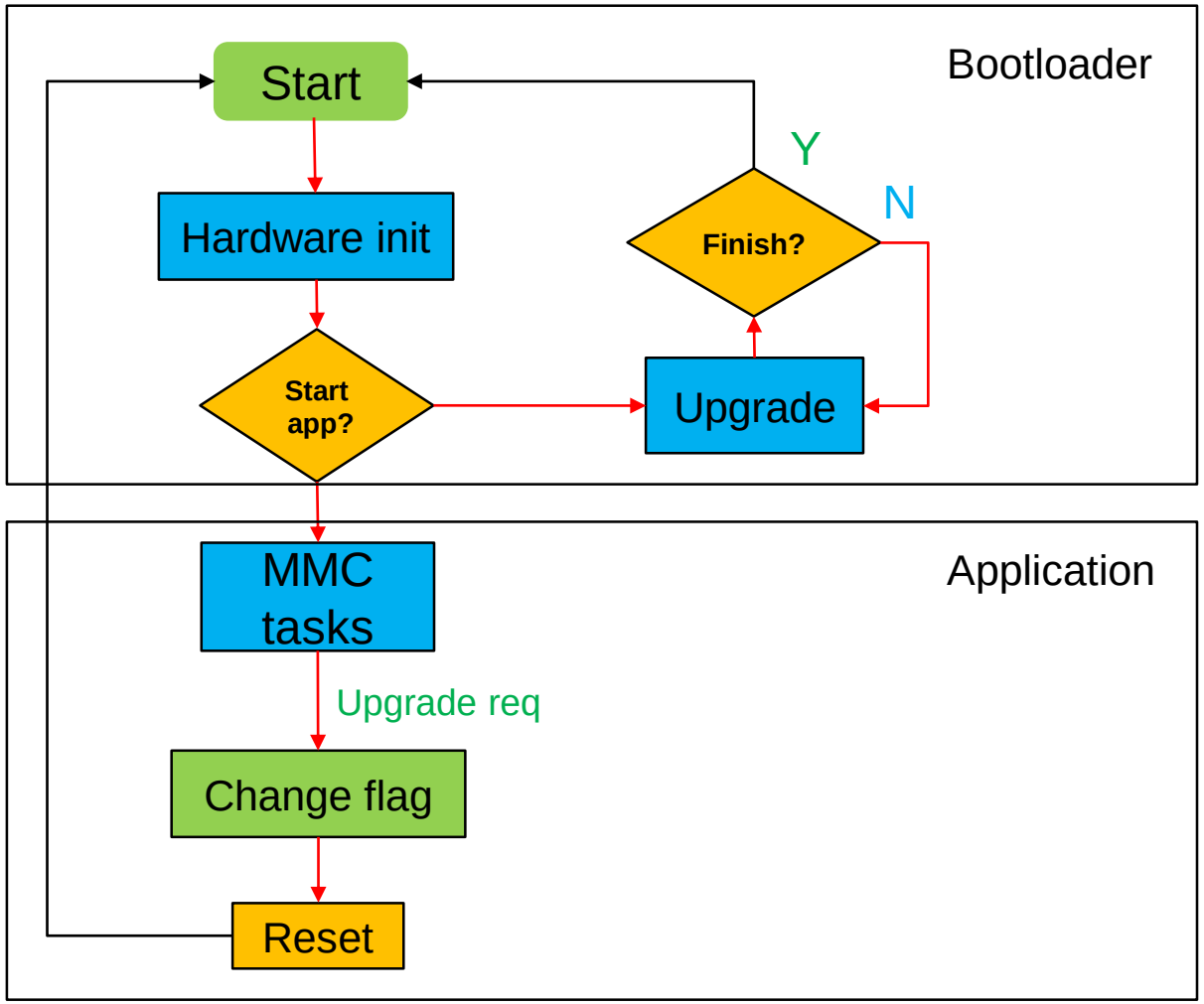
Functions

- Remote firmware upgrade



- RTM extension


Bootloader
Start
Get flag (EEPROM)
Start App?
Get data
Get success?
Reset
Application
MMC Code
HPM Req?
Set flag (EEPROM)
Reset


Bootloader
Start
Hardware init
Start app?
Finish?
Y
N
Upgrade
Application
MMC tasks
Upgrade req
Change flag
Reset