XSLAB 엑세스랩(주)

/XSLAB | xslab.co.kr
voramall.com

한국 최초 ARM 서버 개발 기업 엑세스랩 주식회사

한국산 토종
OpenBMC 삽질기

2023-07-04 엑세스랩 유 명 환 yoo@xslab.co.kr

OpenInfra DAYS Korea

https://www.facebook.com/**funfunyoo**

유명환 (FunFun Yoo)

친구 2.8천명

- **한국 최초 ARM 서버 개발 기업, 엑세스랩 대표**
- yoo@xslab.co.kr
- (전) 한양대학교 소프트웨어학부 겸임교수
- (전) 국가정보자원관리원 클라우드 기술위원
- (전) 서울 구로구청 스마트 도시 기술정책 자문위원

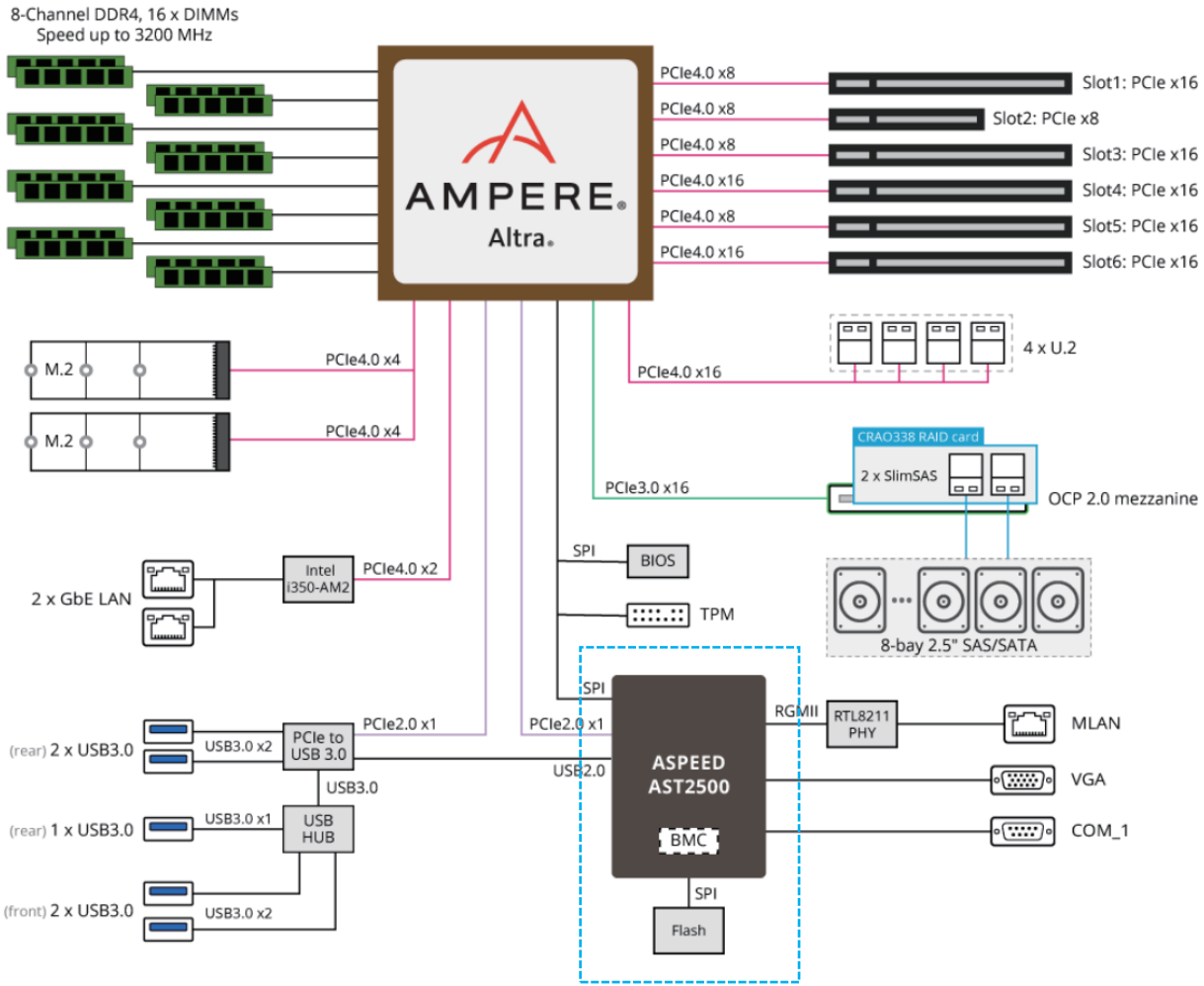
게시물 정보 친구 사진 동영상 체크인 더 보기 ▾ ...

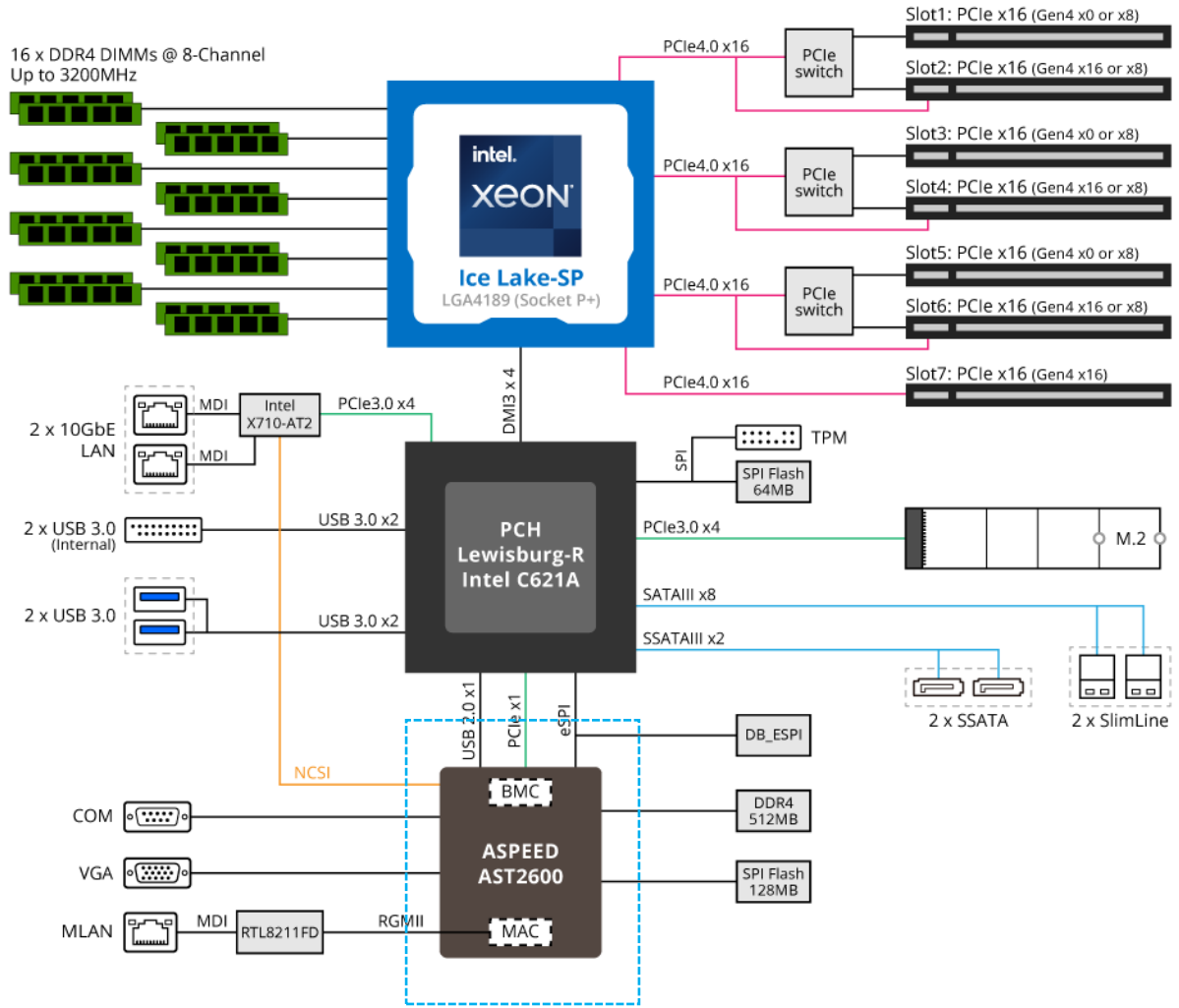
< 2 >

소개

무슨 생각을 하고 계신가요?

BMC 란? : 데이터센터 서버 마더 보드 구조

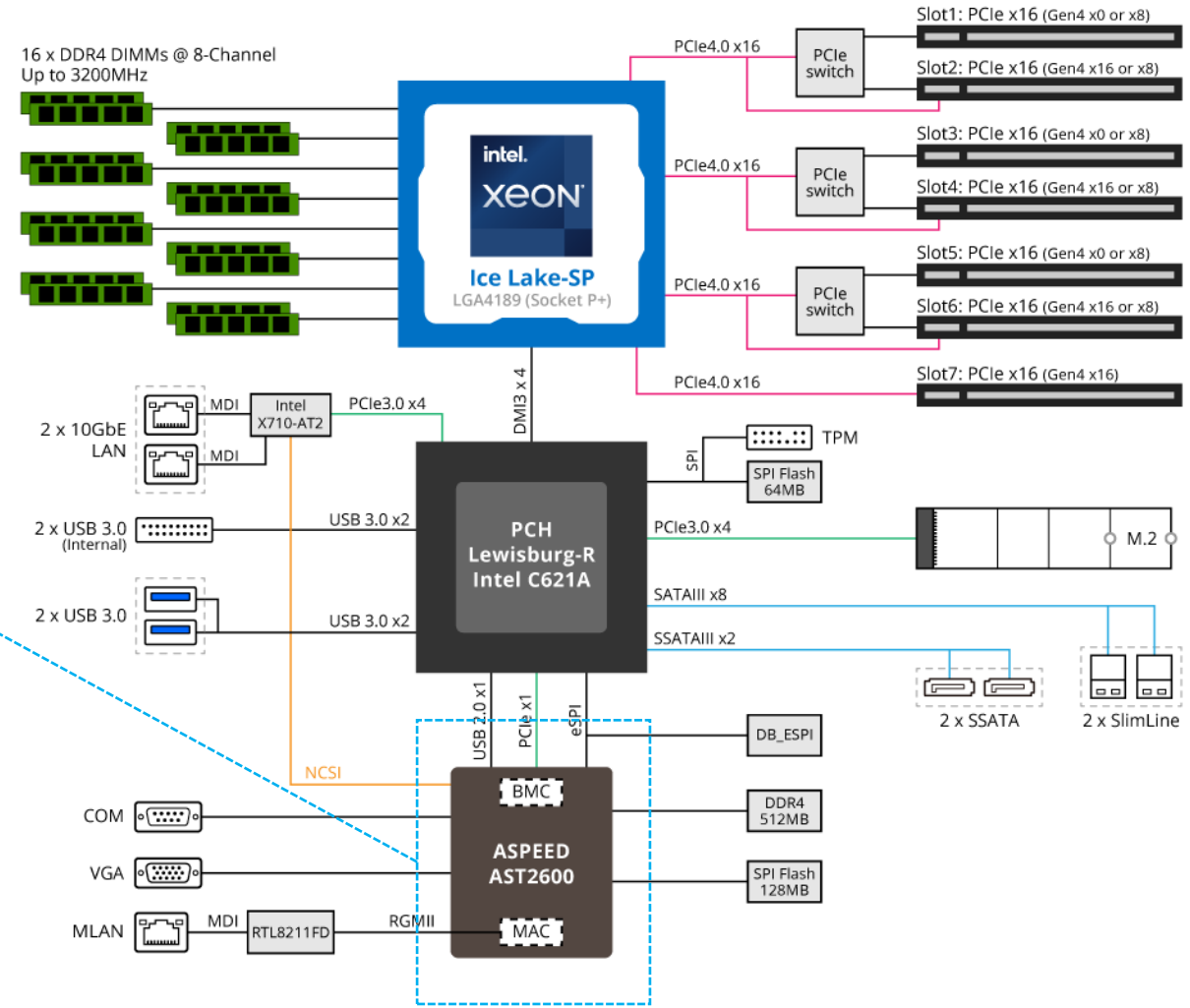




BMC 란? : BMC 전용 칩

BMC : Baseboard Management Controller

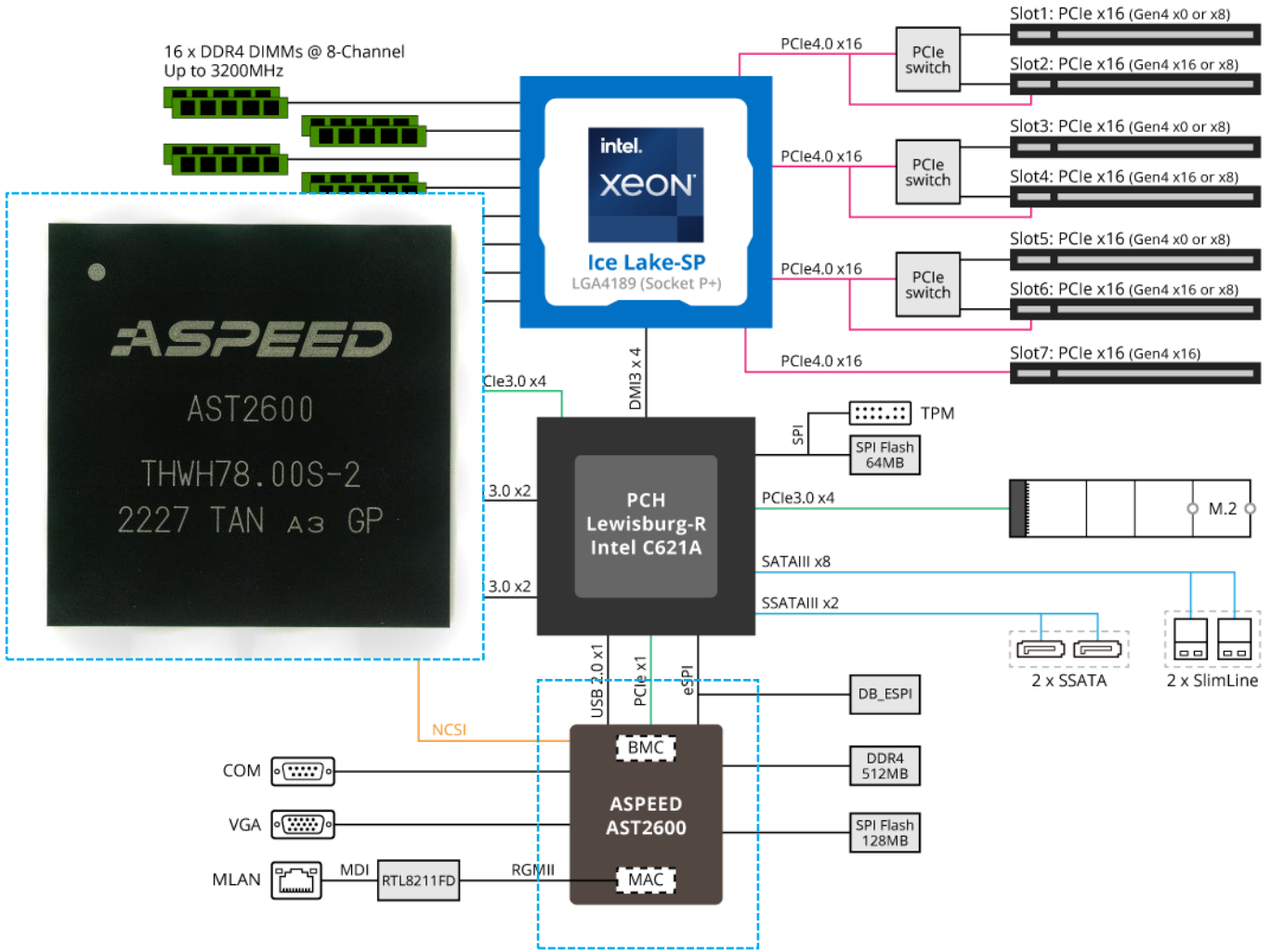
- 32-bit **ARM** Cortex-A7 1.2GHz Dual Core
- 32-bit **ARM** Cortex-M3 200MHz



BMC 란? : BMC 와 IPMI

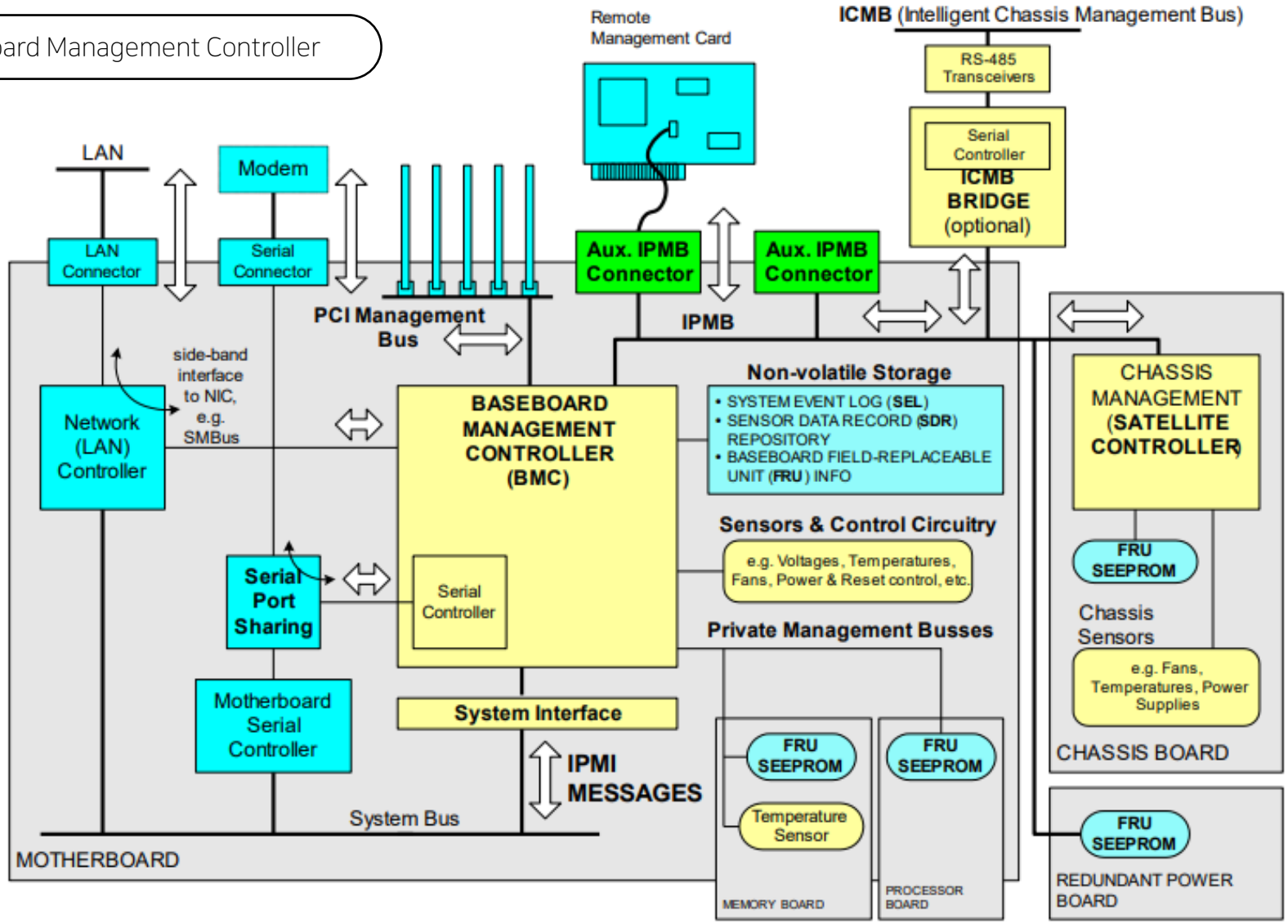
BMC : Baseboard Management Controller

- 서버는 시스템 서비스 관리, 모니터링, 유지 보수 및 제어를 위해 BMC 와 IPMI 를 사용
- **BMC** (Baseboard Management Controller)
  - 센서를 사용하여 시스템의 실제 상태를 모니터하는 특수화된 프로세서
  - 시스템에 장착된 서로 다른 종류의 센서들은 BMC를 통해 온도, 팬 속도, 전원 모드, OS 상태 등을 경보 방식으로 시스템 관리자에게 알려짐
  - System Event Log (SEL) : 시스템 이벤트 로그
  - Sensor Data Repository (SDR) : 센서 데이터 저장
  - Field Replaceable Unit (FRU) : 제품 시리얼 번호, 모델명, 부품 번호 등
  - Serial Over Lan (SOL) : 시리얼 포트의 리디렉션 기능, BIOS Setup
- **IPMI** (Intelligent Platform Management Interface)
  - 하드웨어를 원격으로 관리하는데 사용되는 프로토콜
  - IPMI 서버 : AMI MegaRAC, OpenBMC
  - IPMI 클라이언트 : IPMItool



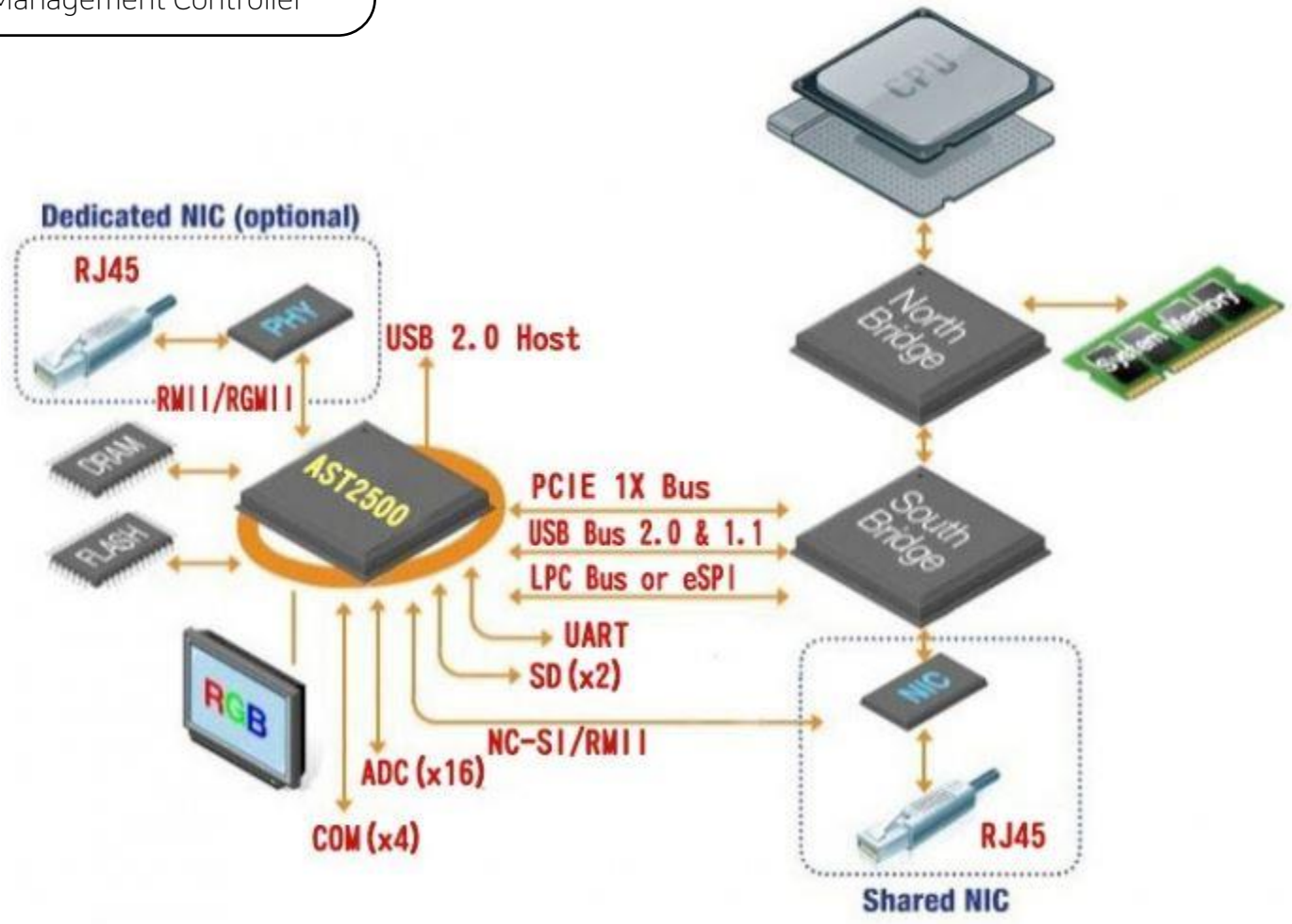
BMC 란? : Motherboard & BMC Block Diagram

BMC : Baseboard Management Controller

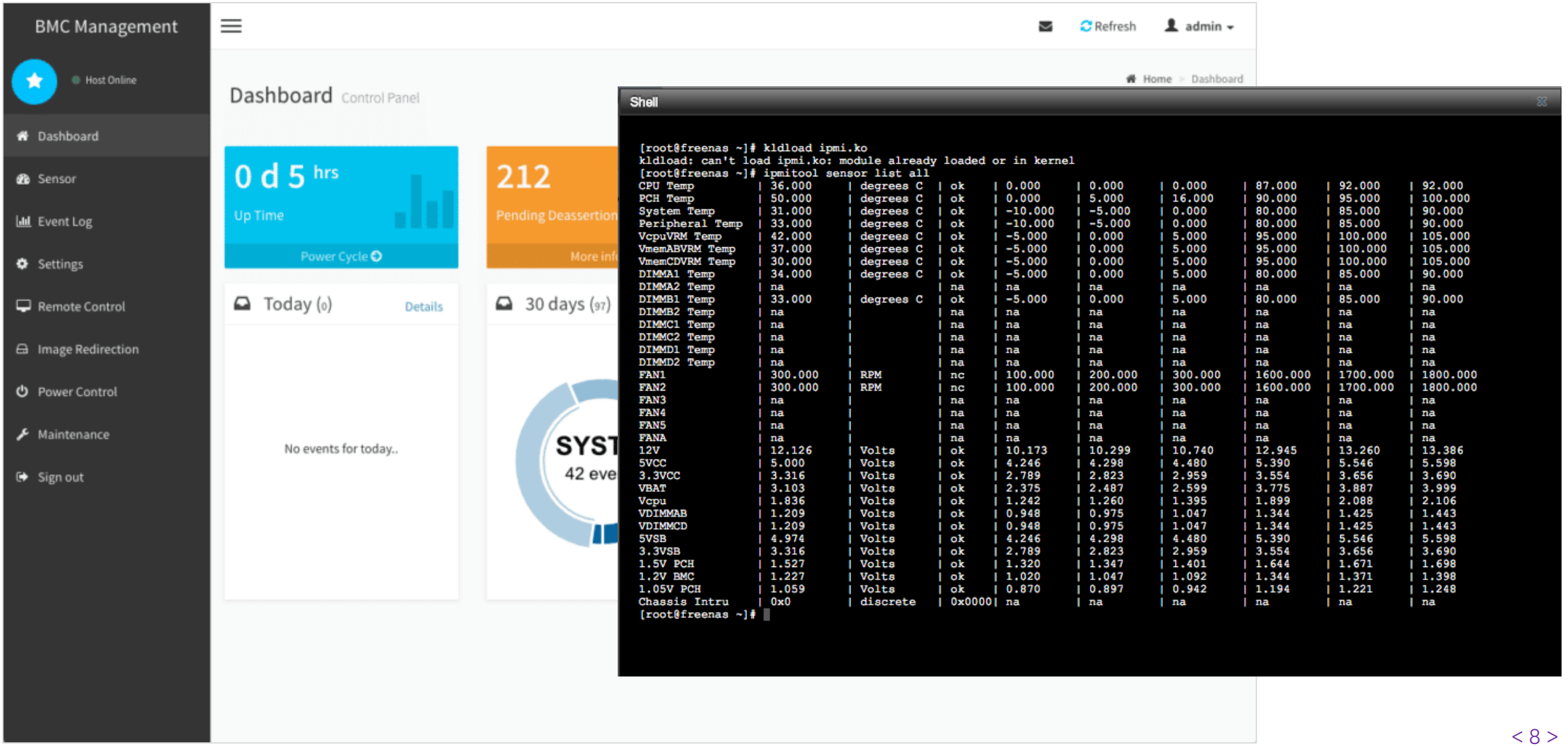


BMC 란? : Motherboard & BMC Block Diagram

BMC : Baseboard Management Controller



BMC 란? : 웹 or CLI 기반 BMC 소프트웨어



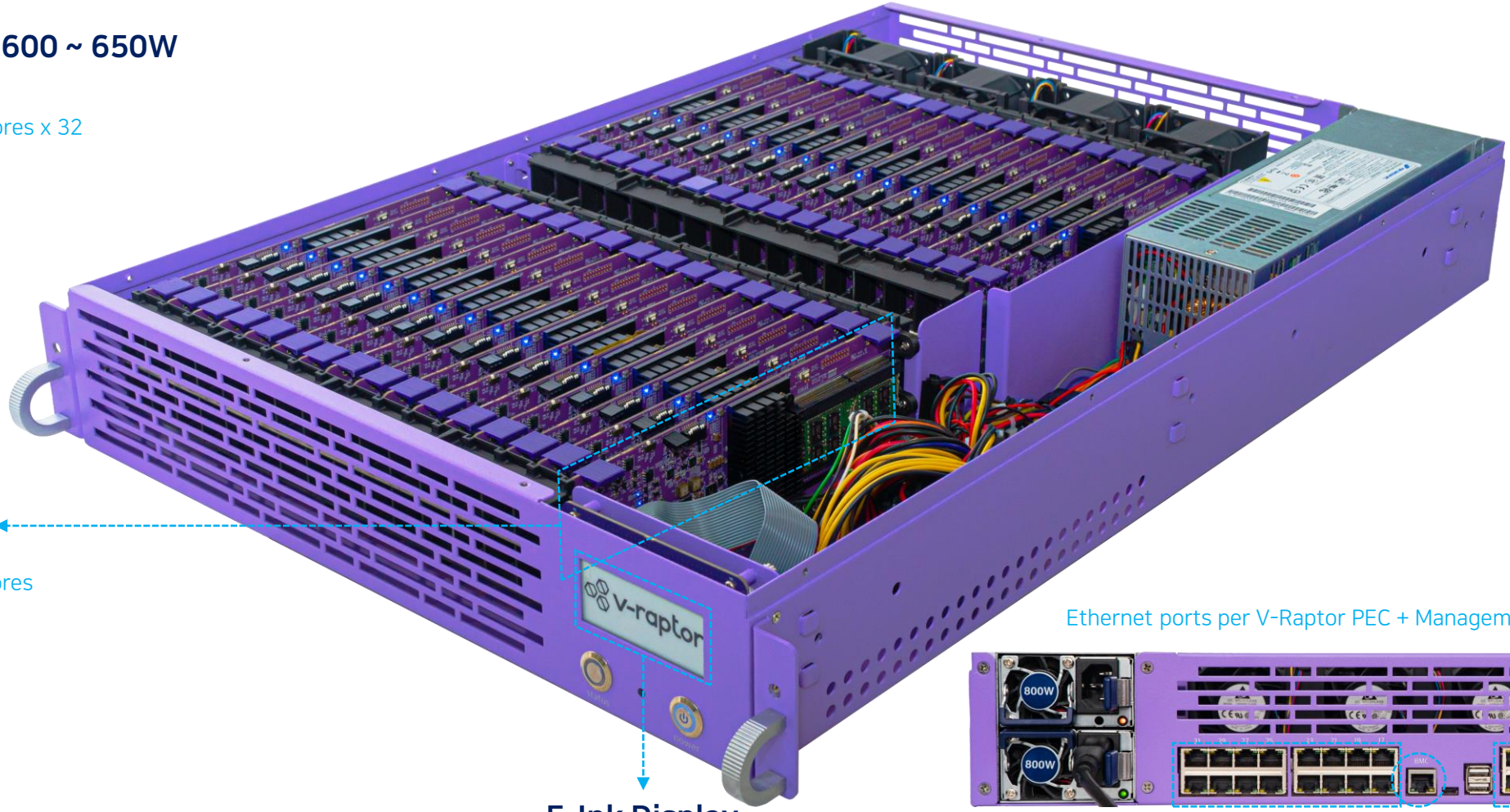
그럼 왜 BMC 를 개발하게 됐을까? : ARM 서버 제조사

Total 600 ~ 650W

64bit ARM 1GHz 24 cores x 32

DDR4 16GB x 32

SATA SSD 250GB x 32



V-Raptor PEC

64bit ARM 1GHz 24 cores

DDR4 16GB

SATA SSD 250GB

E-Ink Display

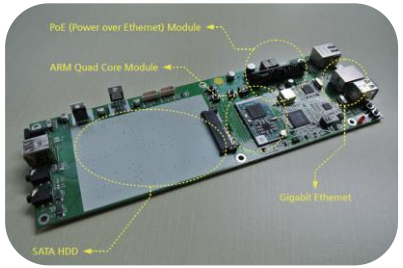
Ethernet ports per V-Raptor PEC + Management Ethernet port

그럼 왜 BMC 를 개발하게 됐을까? : ARM 서버 제조사

2011

ARM Cloud 컴퓨터 회사 구상

Calxeda 분석 및 기반 기술 습득



2012

ARM 서버 1차 시제품 개발

ARM 서버 관련 ETRI 과제 수행

2014

ARM 서버 2차 시제품 개발

네이버 DEVIEW ARM 서버 발표

2015 ~ 2016

네이버 구매조건부 과제 계약 체결

V-Raptor 1차, 2차 시제품 개발

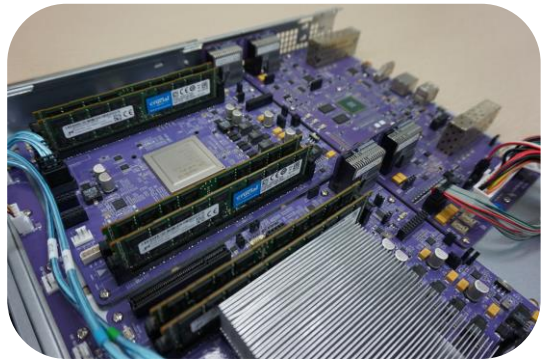
2017 ~ 2018

엑세스랩 창업

TIPS 창업팀 선정 (2018.06)

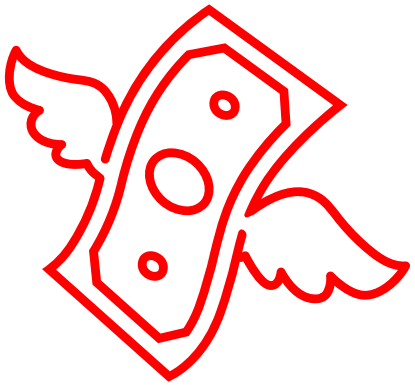
2020

V-Raptor SQ 공식 출시

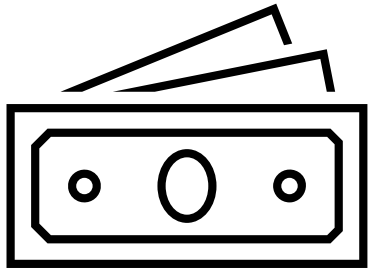


그럼 왜 BMC 를 개발하게 됐을까? : 높은 라이선스 비용

American Megatrends



ami®



OpenInfra DAYS Korea

그럼 왜 BMC 를 개발하게 됐을까? : Embedded Linux

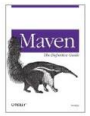
OS & Linux

Java Power Tool
저자 John Ferguson Smart
역자 강수영, 고영태, 백성민, 조창석

O'Reilly Media / 888 페이지 예상
2009년 12월 출간 예정

JavaScript: the Missing Manual
저자 David Sawyer McFarland
역자 김태곤

O'Reilly Media / 560 페이지 예상
2009년 12월 출간 예정

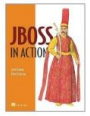


Maven: the Definitive Guide
저자 Sonatype Company
역자 소프트웨어 인 라이프

O'Reilly Media / 480 페이지 예상
2010년 1월 출간 예정

SCJP Sun Certified Programmer for Java 6 Exam 310-065
저자 David McFarland
역자 심재철

McGraw-Hill / 870 페이지 예상
2010년 1월 출간 예정



jBoss in Action
저자 Javid Jamae, Peter Johnson
역자 심재철

Manning Publication / 500 페이지 예상
2010년 출간 예정

Wicket in Action
저자 Martijn Dashorst, Eelco Hillenius
역자 황재선

Manning Publication / 450 페이지 예상
2010년 출간 예정

세상에서 가장 이해하기 쉬운
임베디드 리눅스 서적!

이 책은 임베디드 리눅스를 처음 접하는 개발자들이 가장 빠른 시일 내에 임베디드 리눅스 개발에 필요한 지식들을 습득하는데 목적을 두고 있다. 임베디드 리눅스 개발을 위해 필요한 마이크로프로세서(CPU)와 운영체제(OS)에 대한 분석부터 임베디드 리눅스 기반의 디바이스 드라이버 프로그래밍까지, 임베디드 리눅스 개발에 필요한 전반적인 내용들을 저자의 풍부한 개발 및 강의 경험을 토대로 최대한 쉬운 표현으로 작성되어 있어 누구나 쉽고 빠르게 관련 지식을 습득할 수 있다.
더불어 기존 임베디드 리눅스 서적과는 달리 Windows 에서도 임베디드 리눅스 개발이 가능하도록 Cygwin 기반의 개발 환경을 구축하는 방법부터, GUI 기반의 IDE 툴로 현재 각광을 받고 있는 이클립스(Eclipse) 툴을 기반으로 임베디드 리눅스 어플리케이션을 개발하고 이를 실제 타겟 보드와 연동하여 개발하는 내용까지 다루고 있어, 복잡한 명령어와 불편한 사용자 환경 때문에 임베디드 리눅스 개발이 어려웠던 분들께 획기적인 도움이 될 것이다.

이 책의 주요 내용

임베디드 시스템 개발 방법론
마이크로프로세서(CPU), 운영체제(OS), 개발 툴(Tool) 관점에서의 임베디드 시스템 개발 방법론 제시

국내 임베디드 마이크로프로세서(CPU) 및 운영체제(OS) 분석
마이크로프로세서(CPU) 내부 구조 및 동작 원리 분석
RTOS (Real-Time OS) 와 Non-RTOS 의 비교 분석을 통한 운영체제(OS) 원리 분석

임베디드 리눅스 개발 환경 구축 : Windows 및 Linux 기반
Cygwin 기반의 Windows 에서의 임베디드 리눅스 개발 환경 구축 방법 / Linux 기반의 임베디드 리눅스 개발 환경 구축 방법

Makefile 문법 및 개발자 전용 라이브러리 작성법
Makefile 구성 요소 및 문법 / 개발자 전용 Static Library, Shared Library 작성 방법

리눅스 커널 모듈 프로그래밍
Firmware, Kernel Module, Device Driver 간 차이점 분석 / Kernel Module 프로그래밍 문법
커널 심볼(Symbol) 공유 및 Startup Parameter 예제 수록

리눅스 디바이스 드라이버 프로그래밍
디바이스 드라이버 동작 원리 분석 / 리눅스 커널 2.4 버전과 2.6 버전 간 디바이스 드라이버 차이점 분석
디바이스 드라이버 문법 / 디바이스 드라이버와 사용자 어플리케이션 간 연동 방법 분석

이클립스(Eclipse) 기반의 임베디드 리눅스 프로그래밍
오픈 소스 기반의 이클립스(Eclipse)를 통한 임베디드 리눅스 프로그래밍
이클립스와 타겟 보드 간 연동 방법 분석

값 22,000원

ISBN 978-89-93827-15-6

志&嬋 지앤선

뻔뻔하게 fun+fun Embedded Linux 배우는 임베디드 리눅스
유명환 지음
志&嬋 지앤선

저자 소개

유명환 funfun.yoo@gmail.com

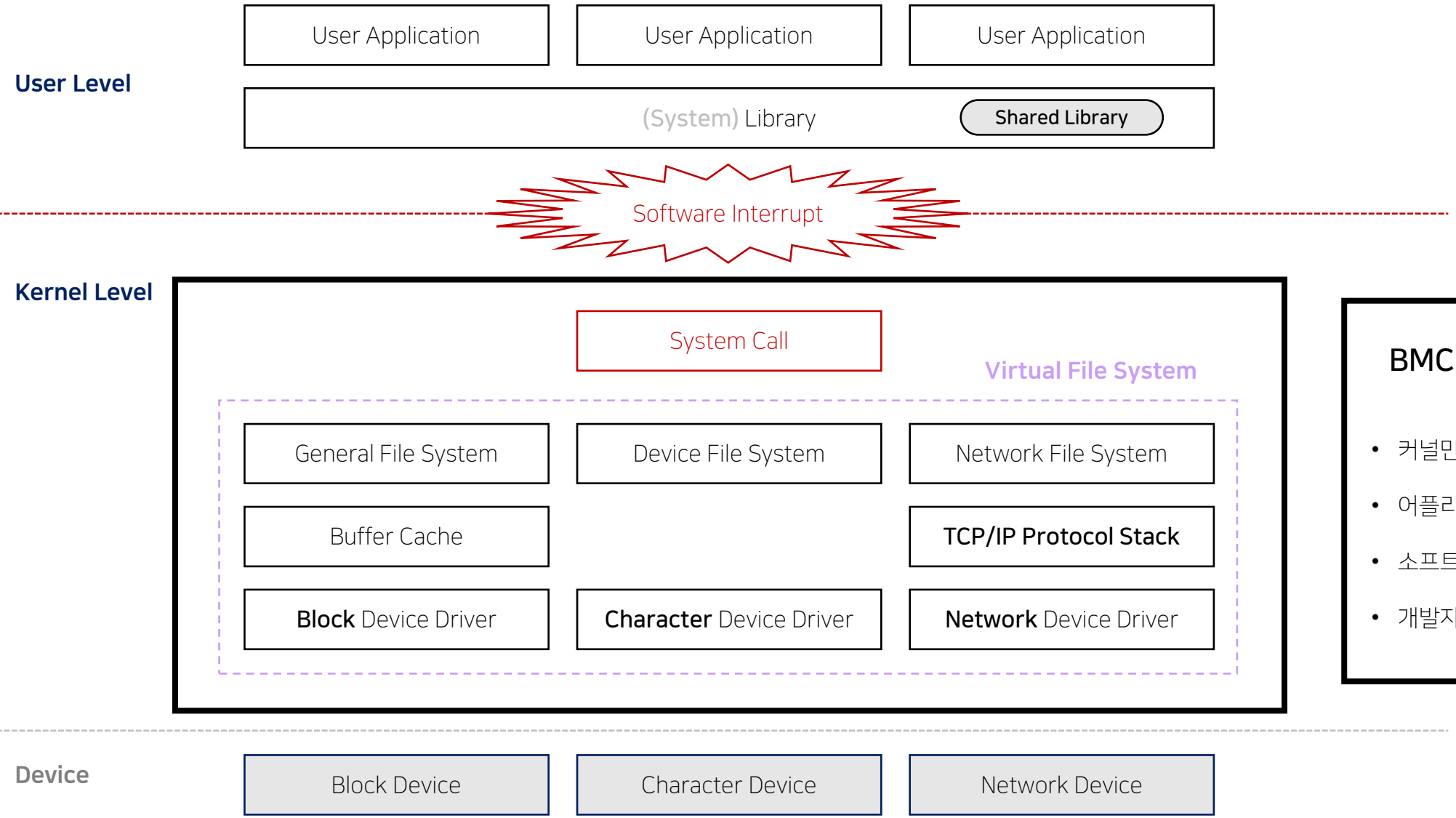
재미(FUN)있다는 이유 하나만으로 어플리케이션 개발자에서 임베디드 개발자로 전향하여, 중소기업 프로젝트부터 국가연구소 프로젝트까지 다년간 임베디드 프로젝트를 진행한 베테랑이다.

아무 것도 모르는 상태에서 임베디드 개발자로 거듭나기까지 겪었던 수많은 경험들 때문에 그 누구보다 처음 임베디드 기술을 접하는 개발자들의 마음을 헤아릴 줄 알기에 그들에게 도움이 되고자 2005년부터 현재까지 삼성전자, 삼성SDS, LG전자, LG CNS, 한국HP, 현대중공업, 한국전자통신연구소(ETRI), 국방과학연구소(ADD), 한국소프트웨어진흥원(KIPA), 한국정보산업연합회(FKII), KAIST EMDEC, 삼성SDS 멀티캠퍼스, 비트캠퍼스 등 다양한 기업 및 기관들에서 임베디드 관련 강의를 진행해오고 있으며, 2009년부터 "뻔뻔강사" 라는 별명으로 인터넷 카페 등에 임베디드 관련 강좌를 올리면서 수많은 개발자들의 호응을 받고 있다.

"FUN = Funny(재미있고) + Useful(유익하며) + New(새로운)" 지식 전파의 달인으로 거듭나기 위해 지었다는 뻔뻔강사라는 별명답게, 현재 인터넷 사이트 FUN-MARKET(www.fun-market.co.kr)을 통해 기존에 나왔던 교재들과는 사뭇 다른 교재와 강좌, 교육용 키트 등을 보급하고 있으며, 최근에는 로봇과 구글 안드로이드(Android)를 주제로 저술하고 있다.

jmson.tistory.com

삽질의 시작 : 먼저 개념을 바꾸자!



BMC = Embedded Linux

- 커널만 같다
- 어플리케이션 개발 및 포팅이 힘들다
- 소프트웨어 발전이 더디다
- 개발자 생태계가 좁다

삽질의 시작 : 먼저 개념을 바꾸자!

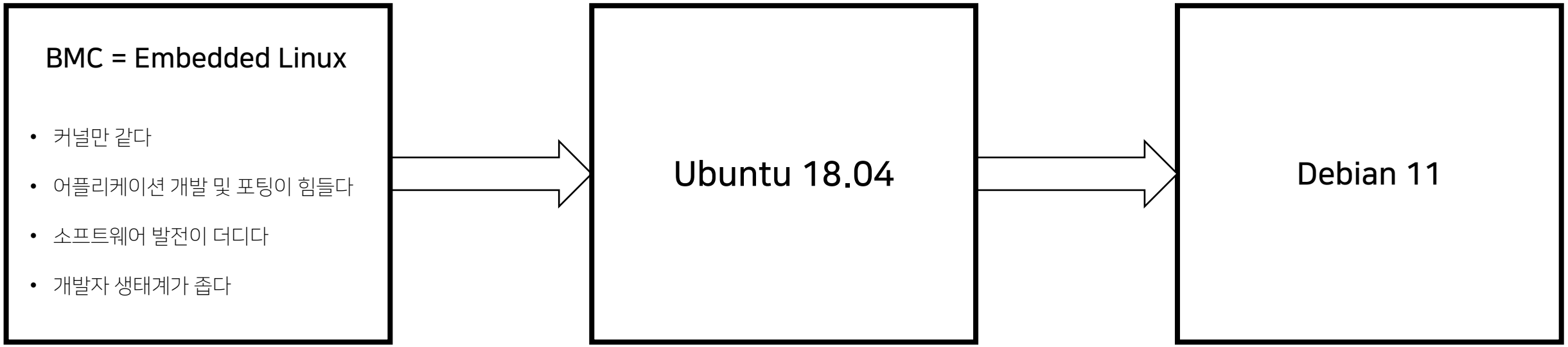
BMC = Embedded Linux

- 커널만 같다
- 어플리케이션 개발 및 포팅이 힘들다
- 소프트웨어 발전이 더디다
- 개발자 생태계가 좁다

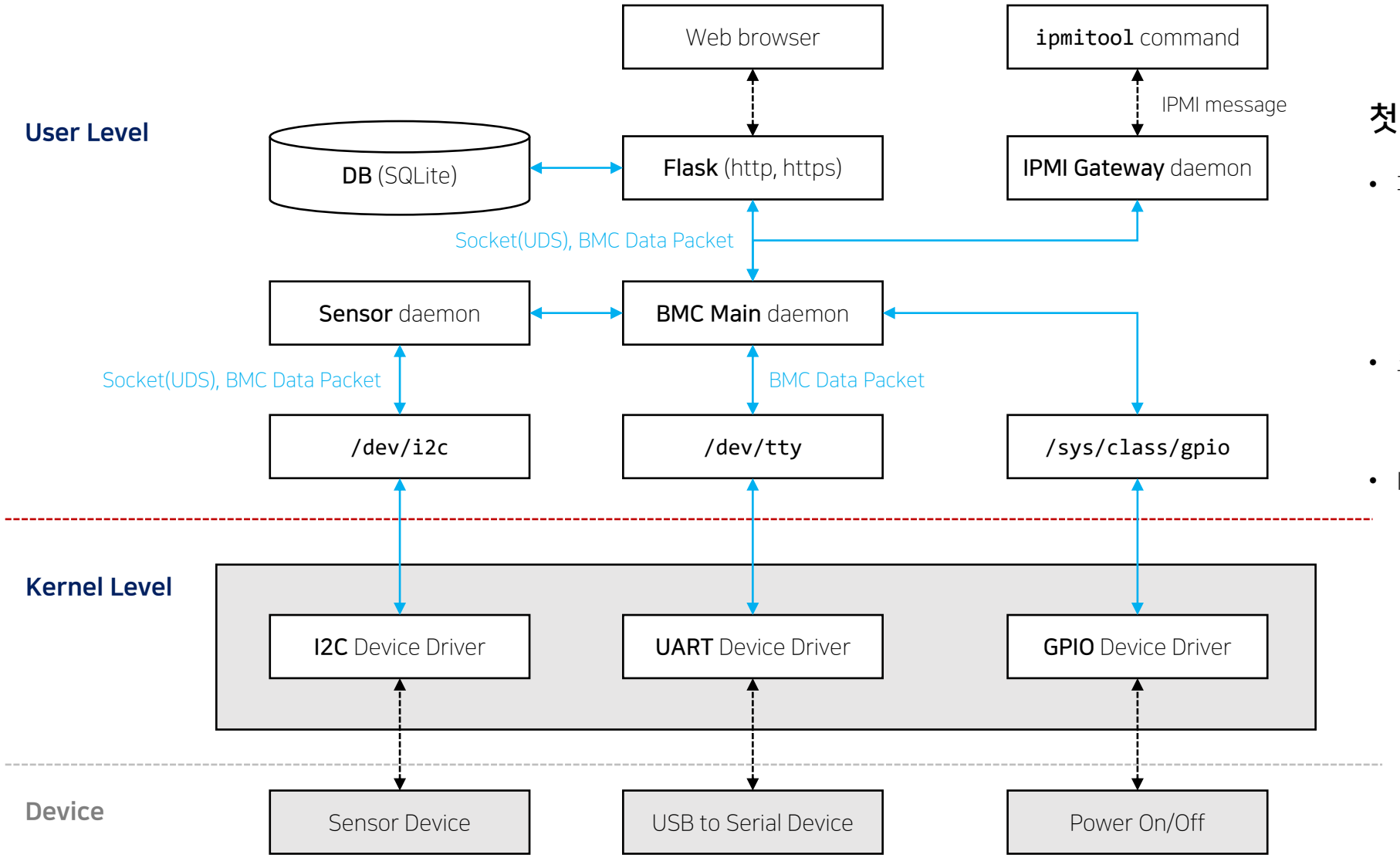
BMC 는 왜 임베디드 리눅스 인가?

- BMC 전용 ARM 칩의 한계 : 32bit ARM 칩
  - ✓ 라즈베리 파이(Raspberry Pi) 첫 ARM 칩 = 32bit ARM11 700MHz, LPDDR1 256MB
  - ✓ 데비안(Debian) : 무척 가벼운 범용 리눅스, 32비트/64비트 둘 다 지원
- 임베디드 리눅스는 안정적이다 라는 고정관념
- 개발 방법론만 Yocto 로 바뀔 뿐 바꿀 생각을 하지 않는다!

삽질의 시작 : 먼저 개념을 바꾸자!



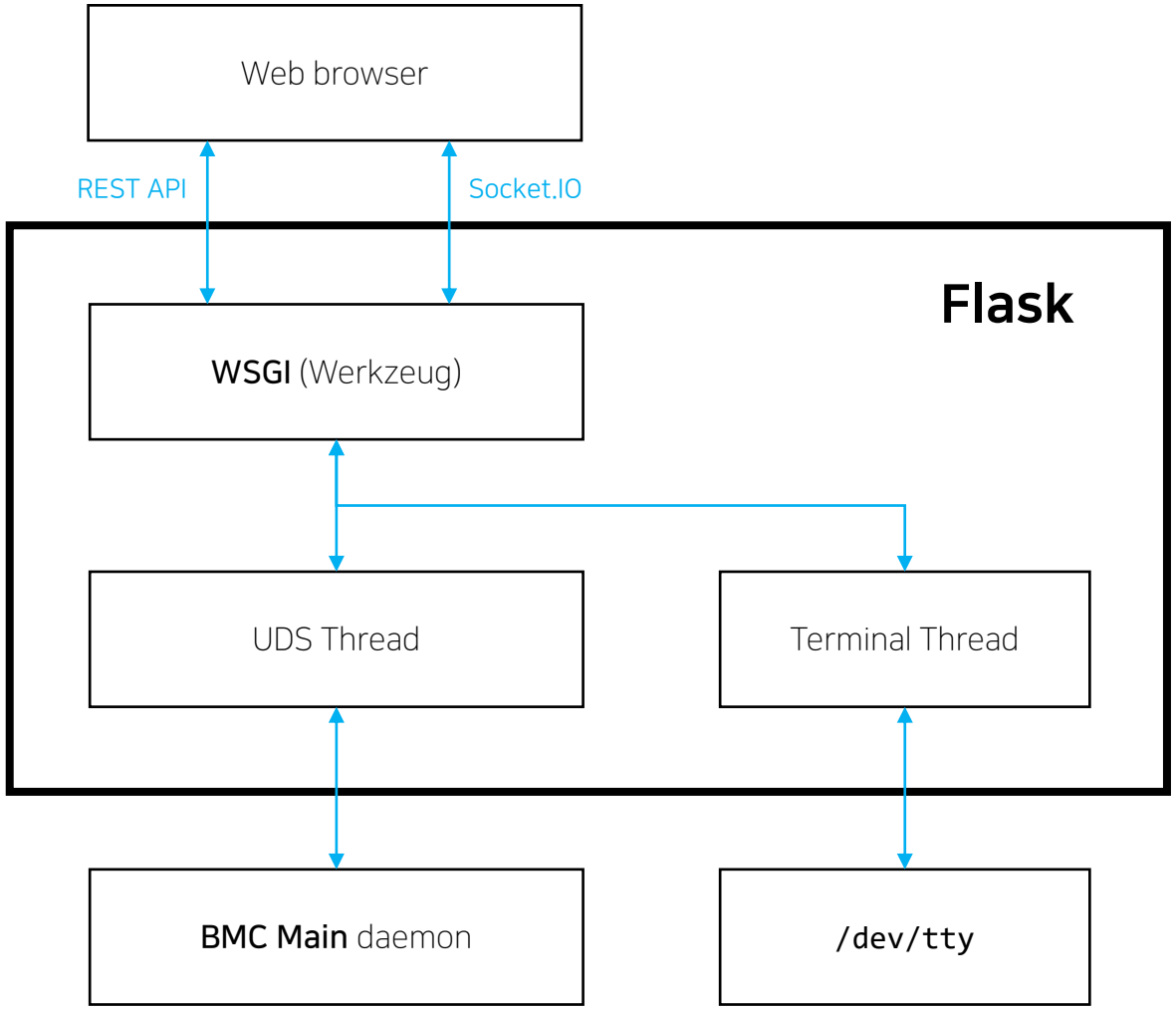
삽질의 시작 : 첫번째 삽질



첫번째 설계의 원칙

- 파이썬으로 개발
  - 백엔드에서 가장 많이 사용하는 언어
  - 백엔드 개발자 생태계를 고려
- 코드는 최대한 가볍게
  - BMC는 받쳐주는 역할
- IPMI 표준 메시지 지원(호환)
  - `ipmitool` 명령어도 지원되어야 함

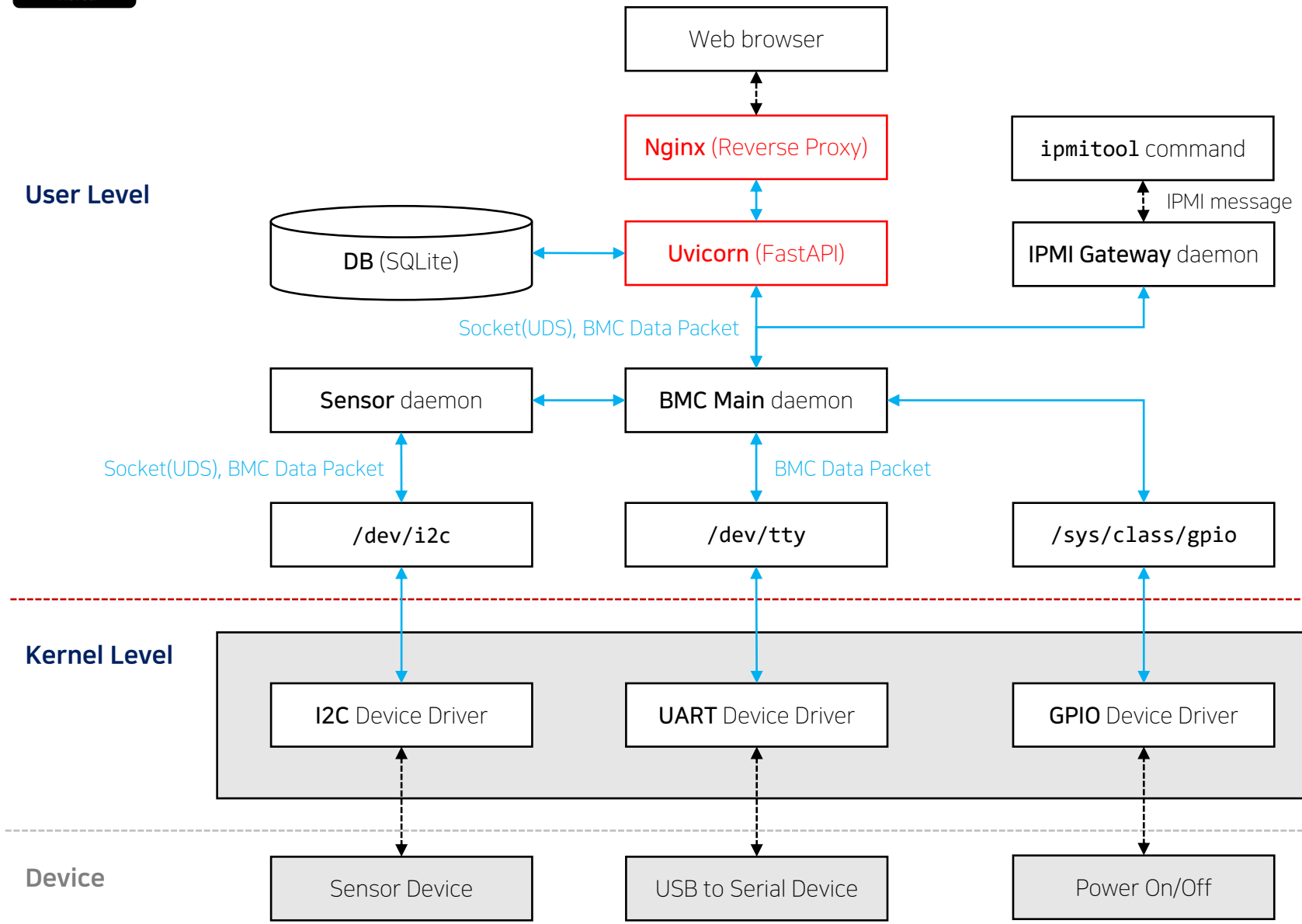
삽질의 시작 : 첫번째 삽질



첫번째 삽질의 문제점

- Flask 의 한계점
  - Flask는 Websocket 을 지원하지 않고 Socket.IO 만 지원
  - Socket.IO 는 표준 기술이 아니기에, 브라우저의 Javascript 라이브러리와 Flask 의 라이브러리 간 프로토콜 버전이 맞아야 동작 가능
  - Websocket 는 HTML5 표준 기술로 오래된 웹브라우저에서는 동작되지 않는 문제가 있는데 Socket.IO 는 해결이 가능
- Flask 의 WSGI (Werkzeug) 서버는 한번에 하나의 요청만 처리 가능
  - 쓰레딩 없이, 정적 파일을 매우 늦게 처리
- SSL 적용했을 때 하나의 프로세스로만 HTTP 와 HTTPS 두개의 프로토콜을 대응할 수 없어서 별도의 서브 프로세스를 실행시켜줘야 함

삽질의 진화 : 두번째 삽질



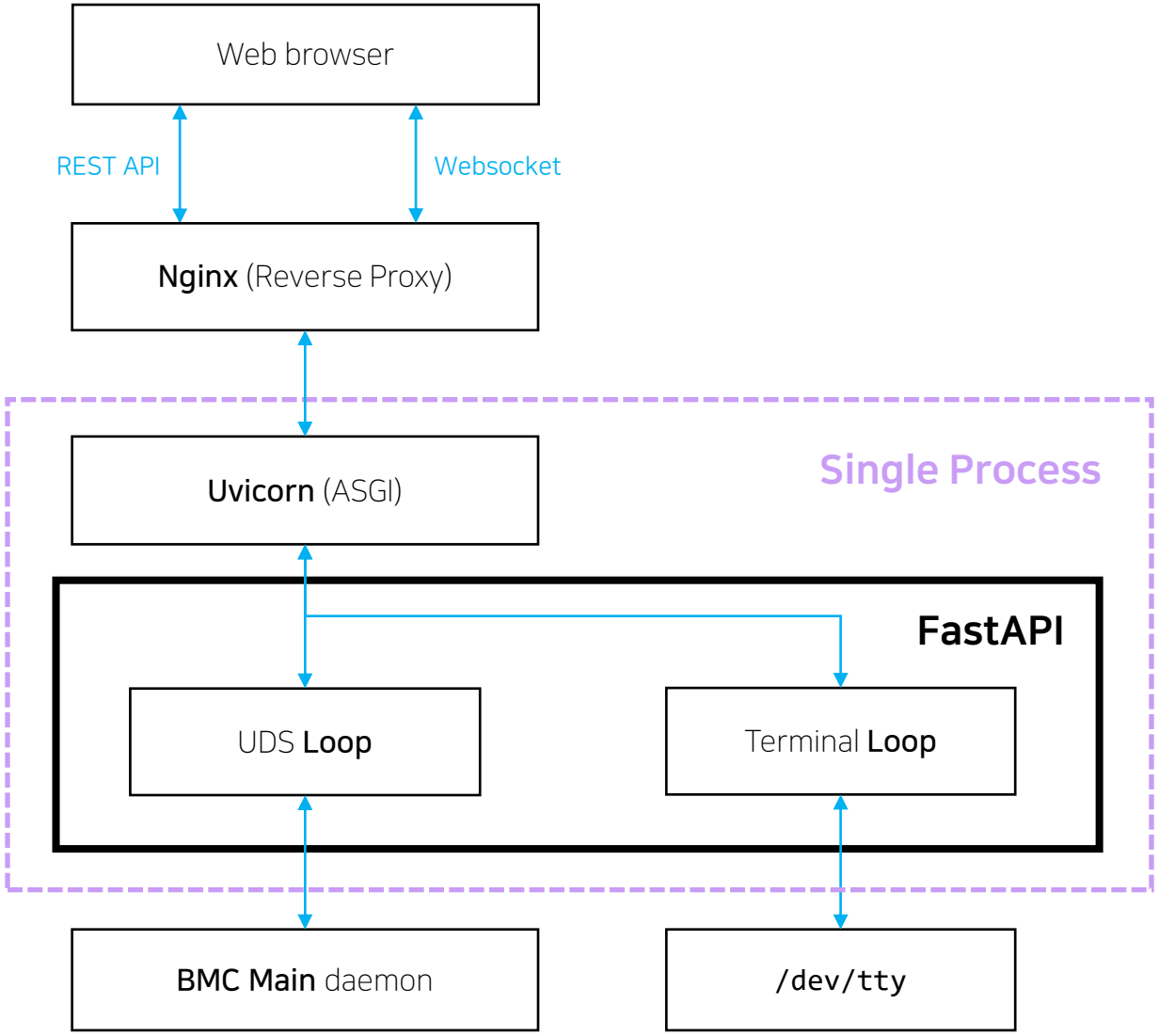
두번째 설계의 원칙

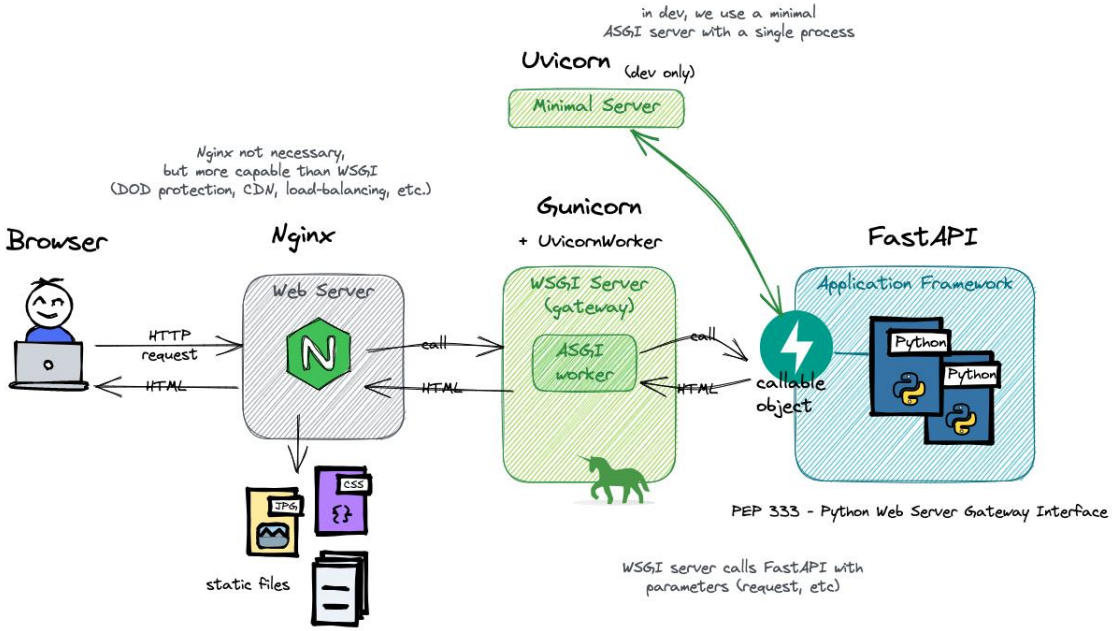
- 파이썬으로 개발하는 원칙은 유지
- FastAPI 방식으로 변경
  - Flask 대비 비동기로 I/O 및 성능에 유리
  - Socket.IO 대신 Websocket 사용
  - 파이썬 비동기 함수로 변경

    : `def` 함수 대신 `async def` 함수 사용

    : Thread 대신 Event Loop 사용
  - WSGI 서버 대신 ASGI 서버 사용
  - FastAPI 내부에 있는 Swagger, Pydantic 라이브러리를 사용하여 데이터 유효성 검증 및 유지 보수에 사용
- 웹 리버스 프록시 (Nginx) 서버 추가
  - 상대적으로 쉬운 SSL 적용 및 HTTP to HTTPS 리다이렉트
  - 웹 정적 파일 서비스 성능에 유리

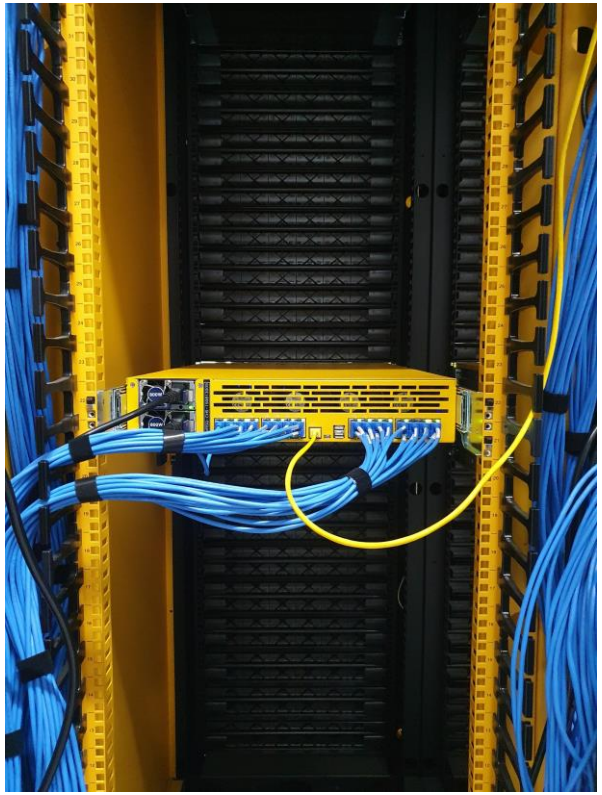
삽질의 진화 : 두번째 삽질





Rnk	Framework	Best performance (higher is better)	
1	fastapi	159,464	100.0% (2.3%)
2	fastapi	159,054	99.7% (2.3%)
3	express	139,760	87.6% (2.0%)
4	express	137,554	86.3% (2.0%)
5	express	111,445	69.9% (1.6%)
6	flask	83,340	52.3% (1.2%)
7	django	79,565	49.9% (1.1%)
8	flask	36,958	23.2% (0.5%)
9	flask	7,960	5.0% (0.1%)

삽질의 변화 : OpenBMC 포팅의 필요성 제기



삽질의 변화 : OpenBMC 포팅의 필요성 제기

[FB소식] 효율적인 데이터 센터의 구축: 오픈 컴퓨트 프로젝트 (Open Compute Project)

2011년 4월 10일 오전 2:20

Facebook은 이번에 산업계 공동으로 더 효율적이고 경제적인 데이터센터를 구축하기 위한 오픈 컴퓨트 프로젝트라는 런칭하며, 데이터센터 구축에 대한 모든 도면 및 기술을 공개했습니다. Facebook이 돌아갈 수 있도록 하기 위한 기술을 개선함으로서 더 많은 친구들과 더 새로운 소셜 경험을 Facebook에서 즐기실 수 있도록 하겠습니다.

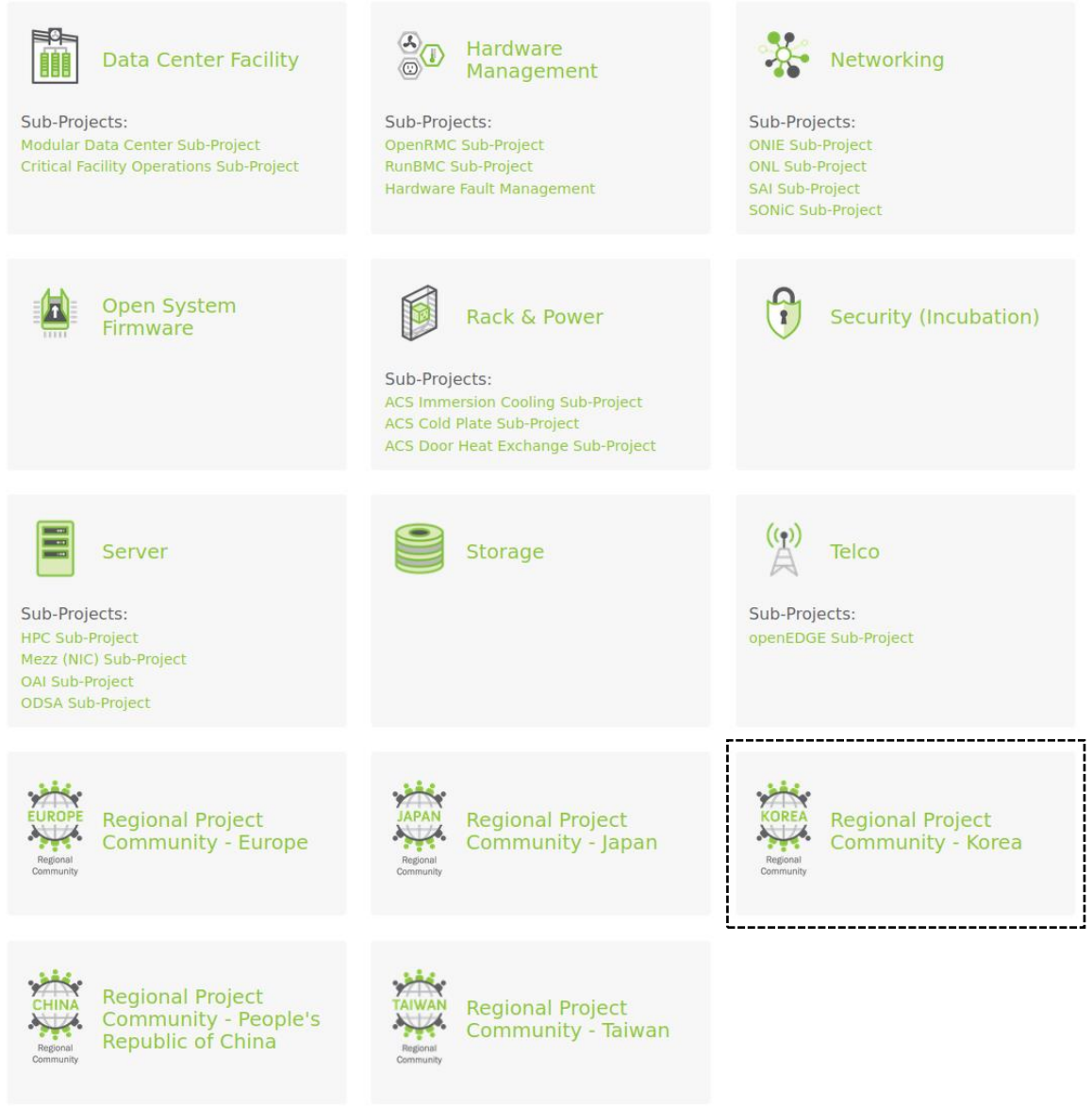
지난 2년간 Facebook의 몇명 엔지니어들은 "어떻게 Facebook의 작업 인프라를 가장 효율적이고 경제적인 방법으로 확장해나갈 수 있을 것인가"라는 큰 과제를 해결하기 위해 노력해왔습니다.

캘리포니아 팔로알토에 위치한 본사의 지하에 위치한 연구실에서 저희 팀은 저희의 첫번째 데이터센터를 기초부터 설계했으며, 몇 달 후 오리건주 프라인빌에 센터를 구축하기 시작했습니다. 세 명으로 시작한 이 프로젝트에서 서버, 전원 공급기, 서버 랙(랙: 서버를 놓는 선반, 캐비넷), 배터리 백업 시스템까지 모두를 직접 제작하게 되었습니다.

아무 것도 없는 상태에서 시작했기 때문에, 저희는 소프트웨어, 서버에서 데이터센터까지 시스템의 모든 부분을 완벽히 통제할 수 있었으며, 따라서 아래와 같은 것들이 가능했습니다.

- 에너지 손실을 줄이기 위한 480볼트 전력 유통 시스템의 사용
- 효율성에 도움이 되지 않는 모든 것을 서버에서 제거
- 서버가 있는 공간(Hot aisle)에서 발생하는 뜨거운 공기를 겨울 중 사무실 공간 및 데이터 센터로 들어오는 외풍을 따뜻하게 하기 위해 활용
- 중앙 통제 전원 공급에 대한 필요성을 제거

결과적으로 저희의 프라인빌 데이터센터는 Facebook의 현재 설비들과 같은 작업을 하는데 38% 더 적은 에너지를 소모하며, 24% 더 적은 비용이 들게 되었습니다.



OpenBMC 삽질 : OpenBMC 포팅 시작

OpenBMC

1 language

Article Talk | Read Edit View history Tools

From Wikipedia, the free encyclopedia

The **OpenBMC** project is a Linux Foundation collaborative open-source project whose goal is to produce an open source implementation of the baseboard management controllers (BMC) firmware stack.[1][2][3] OpenBMC is a Linux distribution for BMCs meant to work across heterogeneous systems that include enterprise, high-performance computing (HPC), telecommunications, and cloud-scale data centers.[3][4]

History [edit]

In 2014, four Facebook programmers at a Facebook hackathon event created a prototype open-source BMC firmware stack named OpenBMC.[5] In 2015, IBM collaborated with Rackspace on an open-source BMC firmware stack also named OpenBMC. These projects were similar in name and concept only.[6] In March 2018, OpenBMC became a Linux Foundation project and converged on the IBM stack. Founding organizations of the OpenBMC project are Microsoft, Intel, IBM, Google, and Facebook.[7][3] A technical steering committee was formed to guide the project with representation from the five founding companies. Brad Bishop from IBM was elected chair of the technical steering committee.[8] In April 2019, Arm Holdings joined as the 6th member of the OpenBMC technical steering committee.[9]

OpenBMC

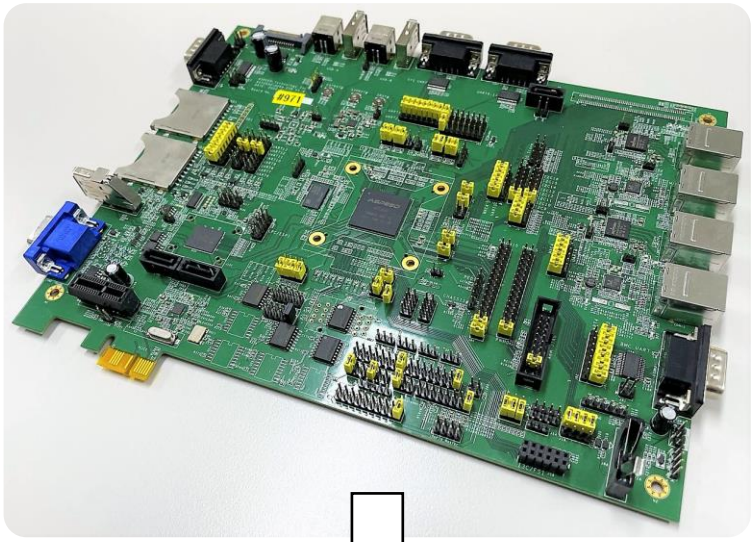
OpenBMC

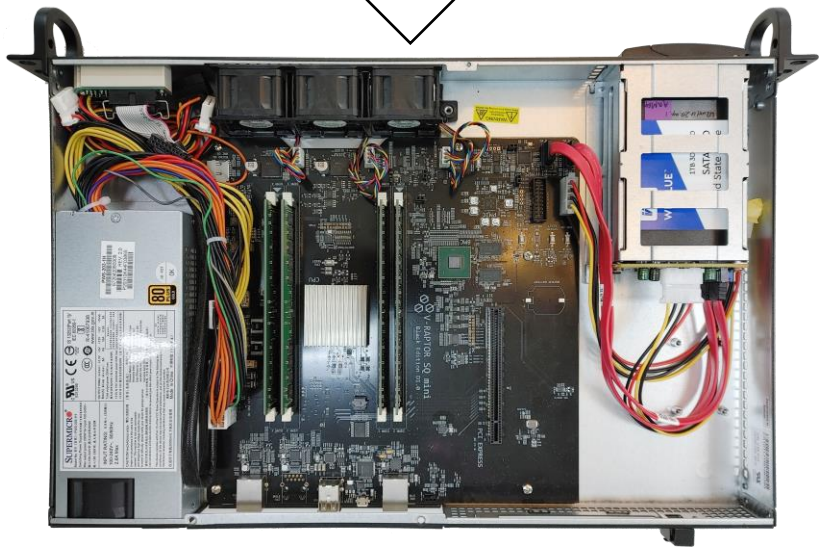
Developer(s)	OpenBMC community
Initial release	3 November 2015; 7 years ago
Stable release	2.12.0 / 21 December 2022; 5 months ago
Repository	github.com/openbmc/openbmc
Written in	C, C++
Available in	Mainly English
License	Apache License 2.0
Website	www.openbmc.org

OpenBMC

- 다양한 종류의 장비들을 원격 관리하기 위한 BMC 를 구현하기 위한 오픈 소스 프로젝트
- 주요 기술들
  - Yocto
  - OpenEmbedded
  - systemd
  - D-Bus

OpenBMC 삽질 : OpenBMC 포팅 시작

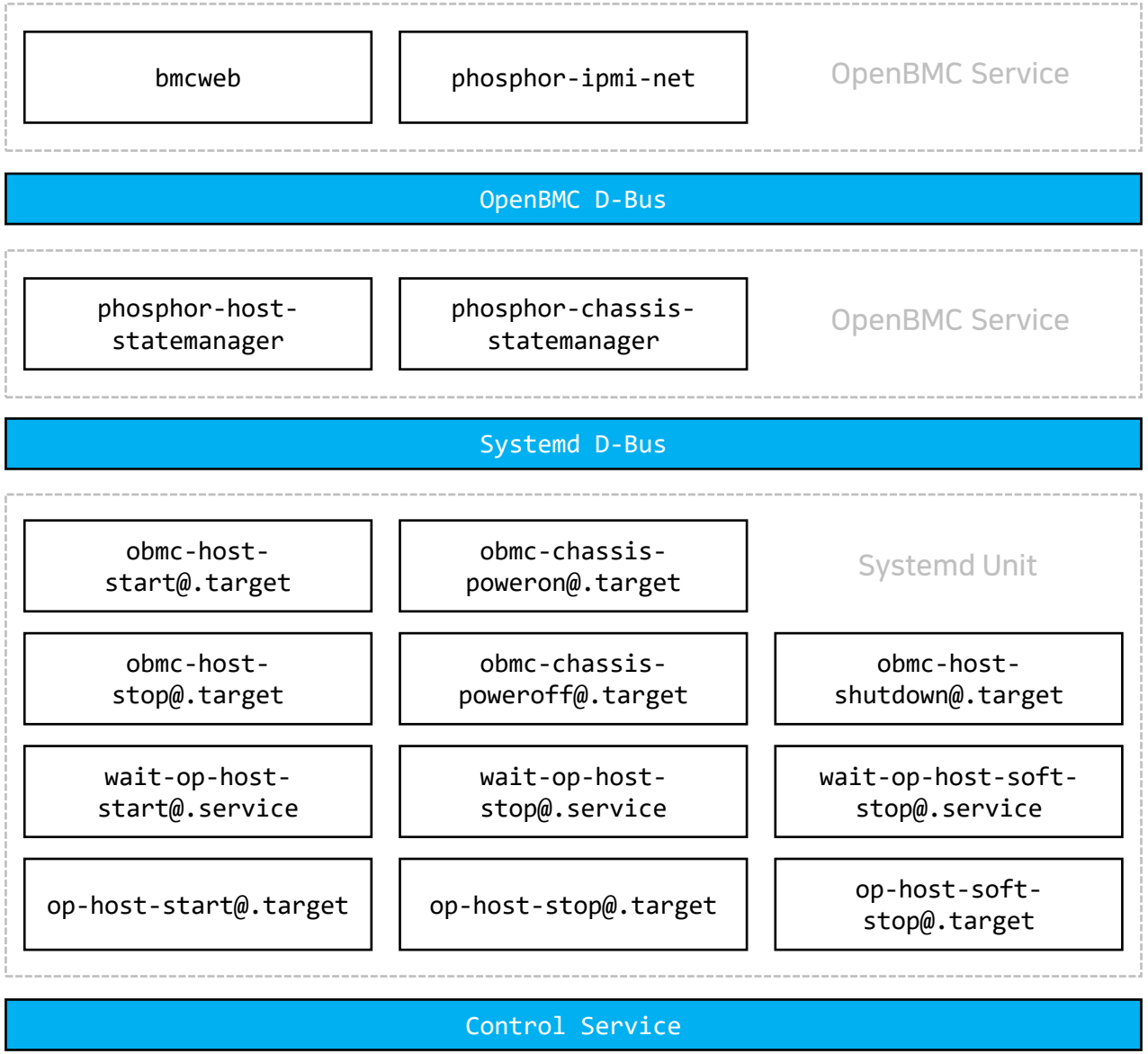




OpenBMC 포팅의 시작

- 부팅부터 확인 -> 실행되는 시스템을 분석하는 방식으로 진행
- AST2600 레퍼런스 보드에서 사용 가능한 부팅 이미지를 빌드한 후에 결과 이미지를 보드에 포팅 후 실행되는 과정을 분석 -> 최초엔 Yocto 사용
- 실행되는 서비스 분석한 후부터 우리 회사 ARM 서버의 BMC 에서 직접 빌드하여 포팅 작업 진행
    - 빌드 환경 (BMC) : NXP i.MX6 (32-bit ARM Cortex-A53 1GHz Quad core), Debian 11
- ARM 서버 BMC 에 포팅된 gcc 와 glibc 버전 업그레이드 진행
    - BMC 에 포팅된 Debian 11 기반 gcc 와 glibc 버전이 OpenBMC 에서 사용하는 버전 보다 낮아 OpenBMC 에 맞게 상위 버전으로 업그레이드하여 포팅 작업 진행

OpenBMC 삽질 : OpenBMC Architecture



- `phosphor-net-ipmid`
  - https://github.com/openbmc/phosphor-net-ipmid
  - `ipmitool` 명령어를 파싱하여 phosphor-host-ipmid 의 D-Bus 값을 변경
- `phosphor-host-ipmid`
  - https://github.com/openbmc/phosphor-host-ipmid
  - phosphor-net-ipmid 에서 파싱되어 전달된 명령어 처리를 위해 D-Bus 의 Property 를 설정
- `phosphor-state-manager`
  - https://github.com/openbmc/phosphor-state-manager
  - BMC, Chassis, Host CPU 를 관리하는 서비스
  - D-Bus 의 Property 가 변경되면 systemd service 를 active 하는 역할
- `phosphor-hwmon`
  - https://github.com/openbmc/phosphor-hwmon
  - I2C 인터페이스 기반 센서들의 값을 D-Bus에 등록/수정/삭제하는 서비스
- `entity-manager`
  - https://github.com/openbmc/entity-manager
  - FRU 값들을 D-Bus Property로 설정하는 서비스

OpenBMC 삽질 : 센서 처리 옵션 이슈 해결

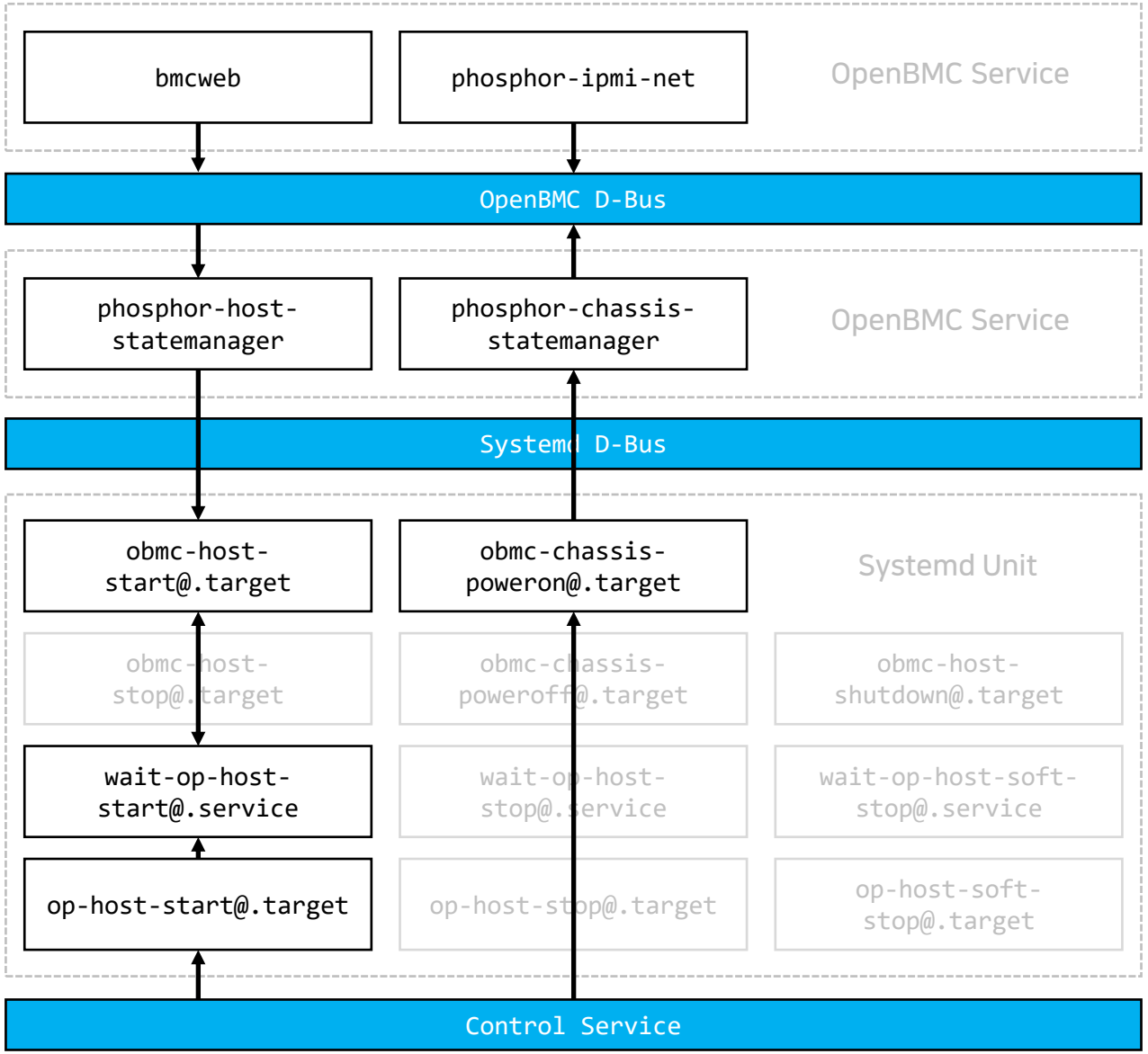
mainloop.cpp

```
...  ...  @@ -309,12 +309,12 @@ MainLoop::MainLoop(sdbusplus::bus::bus&& bus, const std::string& param,
309  309                       const std::string& instanceId,
310  310                       const hwmonio::HwmonIOInterface* ioIntf) :
311  311       _bus(std::move(bus)),
312       -     _manager(_bus, root), _pathParam(param), _hwmonRoot(), _instance(),
     312  +     _manager(_bus, "/"), _pathParam(param), _hwmonRoot(), _instance(),
313  313       _devPath(devPath), _prefix(prefix), _root(root), _state(),
314  314       _instanceId(instanceId), _ioAccess(ioIntf),
315  315       _event(sdeventplus::Event::get_default()),
316  316       _timer(_event, std::bind(&MainLoop::read, this))
317       - {
     317  + {
318  318       // Strip off any trailing slashes.
319  319       std::string p = path;
320  320       while (!p.empty() && p.back() == '/')
```

ipmitool 의 sensor 옵션 처리 이슈

- **ipmitool** 명령어의 sensor 옵션을 실행했을 때 아무런 값이 출력되지 않는 이슈
- **ipmitool** 명령어를 실행(처리)하는 **phosphor-host-ipmid** 에서 확인하는 D-Bus 경로(path)가 **“/”** 로 확인되어 수정
- 즉, **phosphor-hwmon** 소스 코드와 **phosphor-host-ipmid** 소스 코드 경로가 서로 맞지 않아 발생된 이슈

OpenBMC 삽질 : 전원 제어 이슈 해결

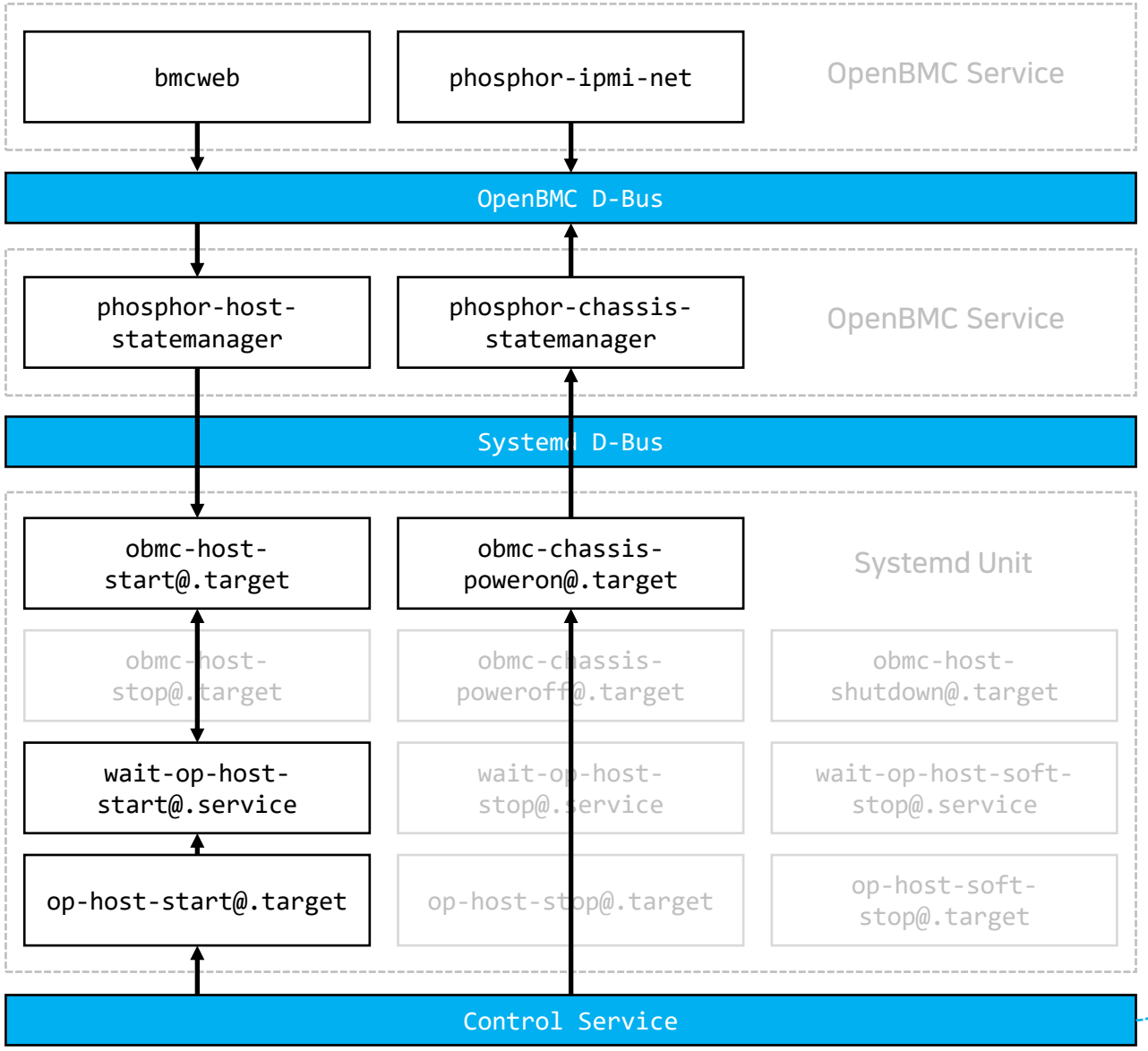


전원 제어 Power Control 이슈

- 기존 OpenBMC 의 서버 전원을 제어하는 계층 구조에서 Systemd 서비스들의 의존성 문제가 발생
- 전원 제어를 위한 Systemd 의존성을 새롭게 구현하는 작업을 진행

Power **On** 과정

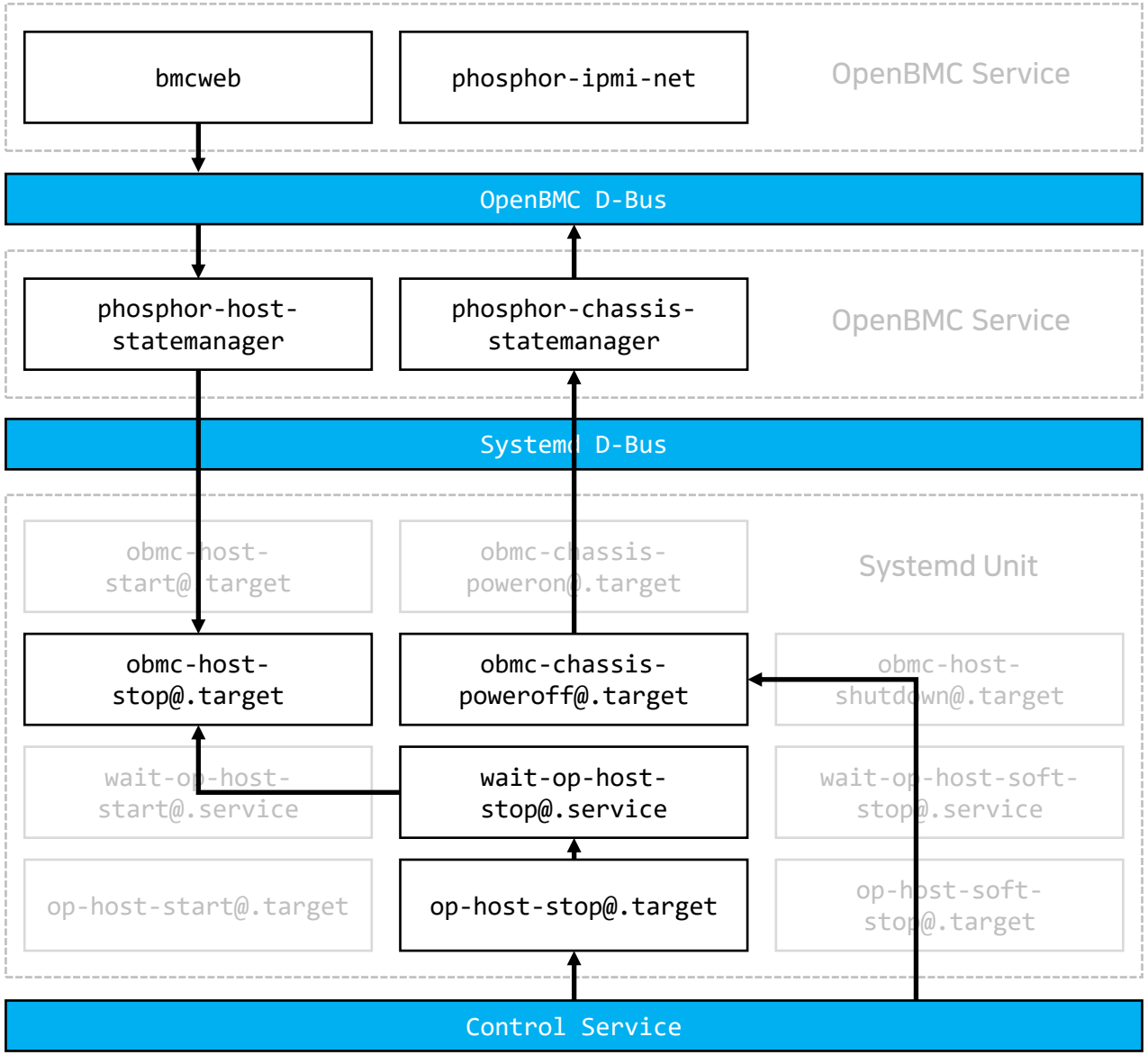
OpenBMC 삽질 : 전원 제어 이슈 해결



별도로 구현한 Control Service

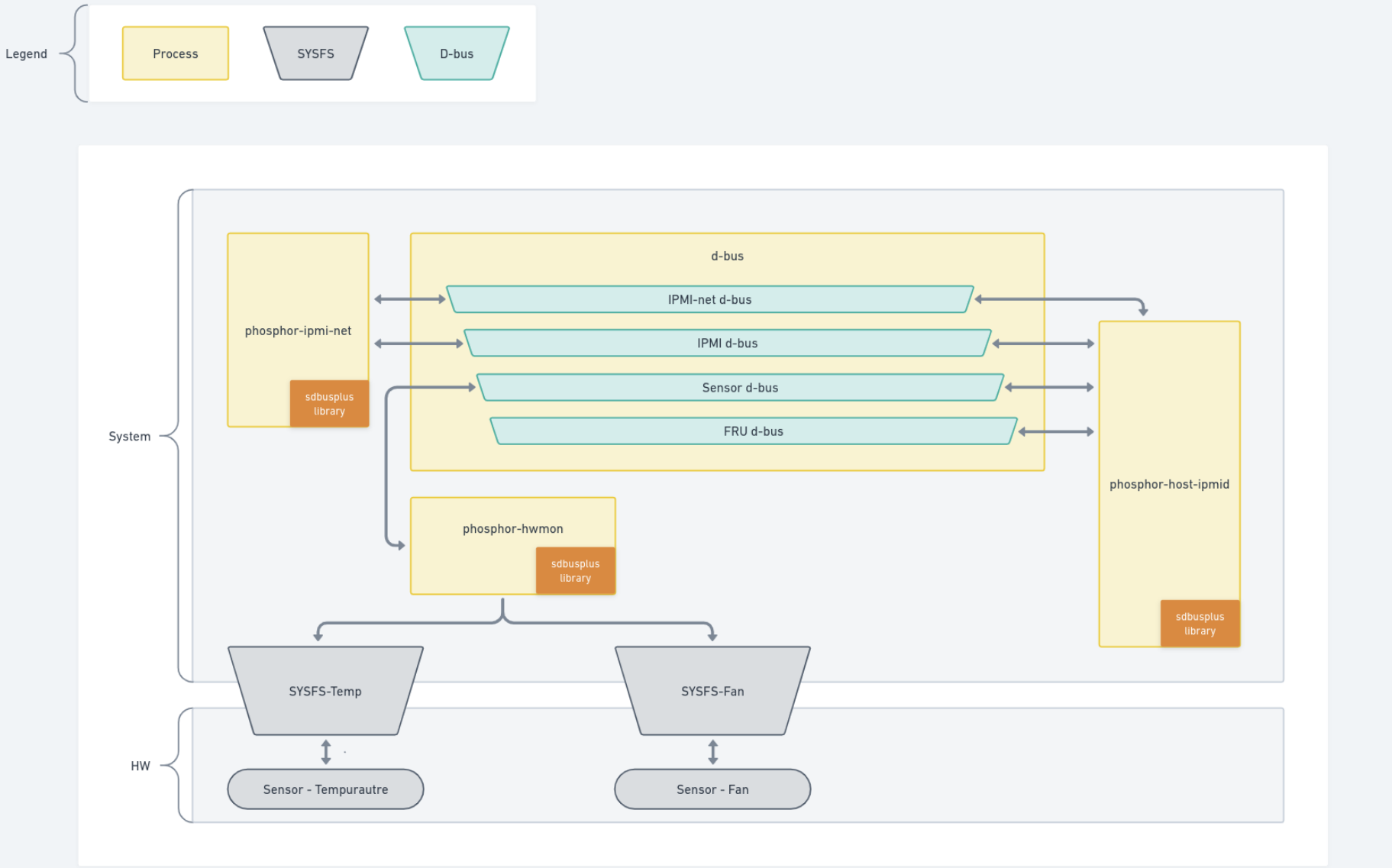
```
#!/bin/sh

if [ "$#" -ne 1 ]; then
        echo "$0 [system_unit_name]"
        exit 1
fi

timeout=50

while (( $timeout != 0 ))
do
        status=$(systemctl is-active $1)
        if [ "$status" = "active" ]; then
                echo "active!"
                exit 0
#       else
#               echo "wait for active: $1"
        fi
        sleep 0.2
        timeout=$(($timeout-1))
done

echo "Timeout"
exit 1
```

OpenBMC 삽질 : 전원 제어 이슈 해결

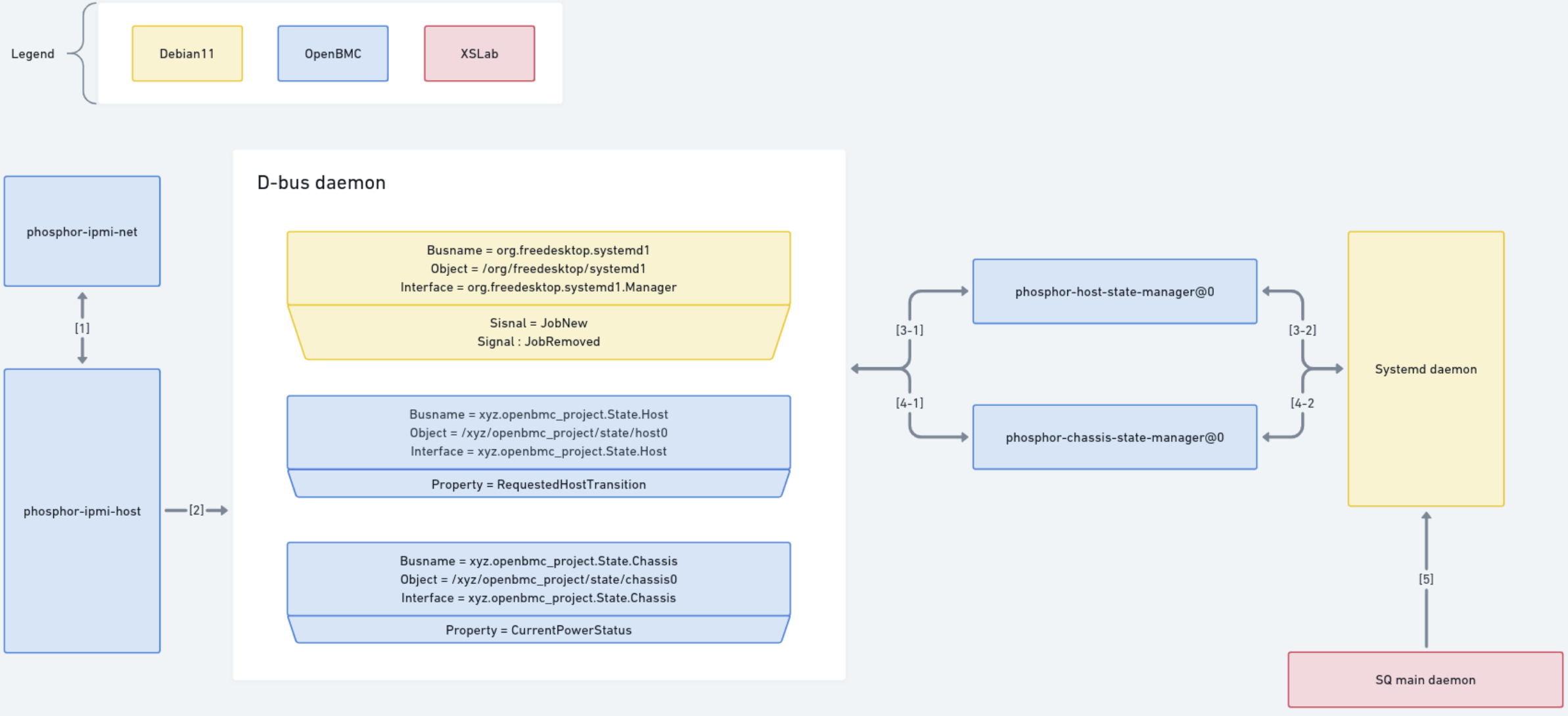


Power **Off** 과정

OpenBMC 삽질 : OpenBMC 포팅 완료



OpenBMC 삽질 : 자체 BMC 와 OpenBMC 통합



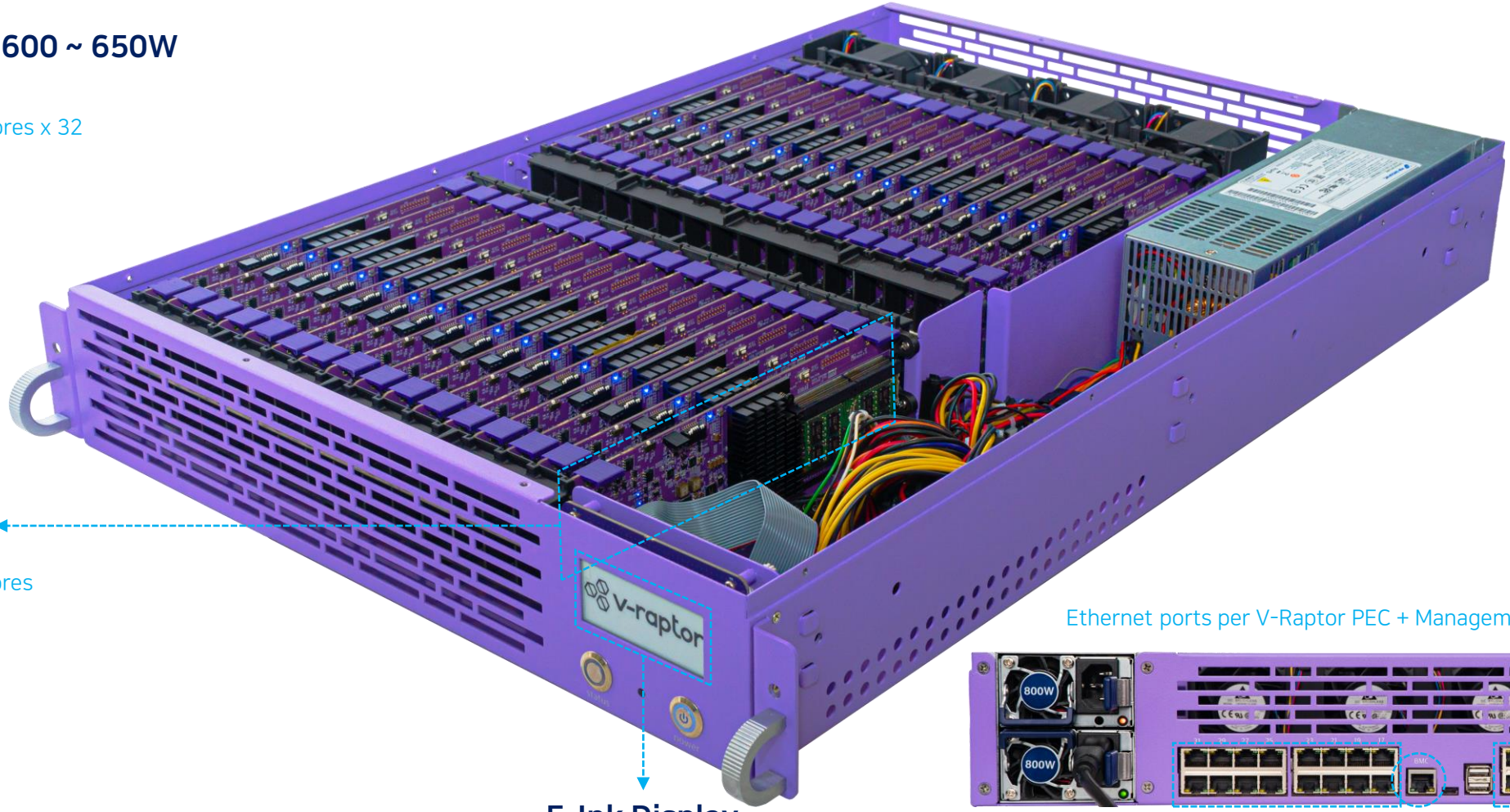
OpenBMC 삽질 : 다중 Host CPU 이슈 해결

Total 600 ~ 650W

64bit ARM 1GHz 24 cores x 32

DDR4 16GB x 32

SATA SSD 250GB x 32



V-Raptor PEC

64bit ARM 1GHz 24 cores

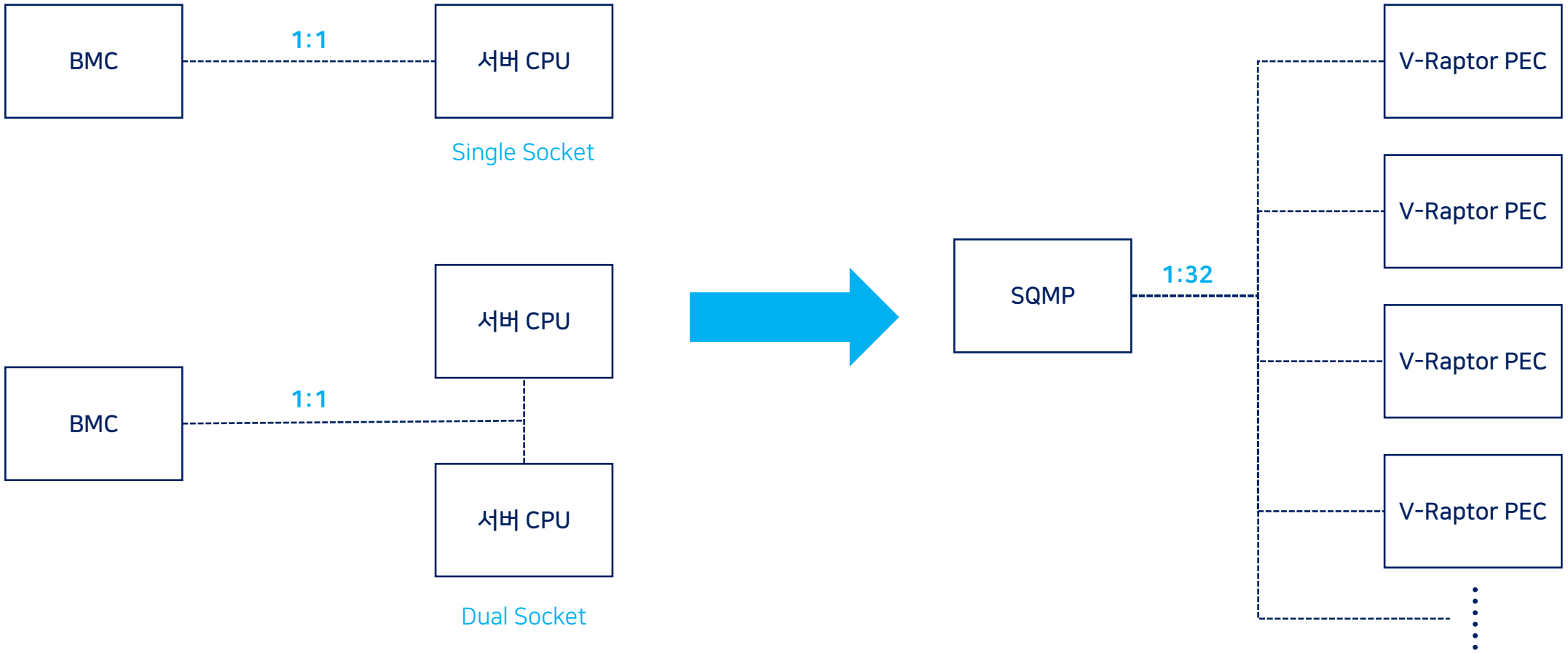
DDR4 16GB

SATA SSD 250GB

E-Ink Display

Ethernet ports per V-Raptor PEC + Management Ethernet port

OpenBMC 삽질 : 다중 Host CPU 이슈 해결



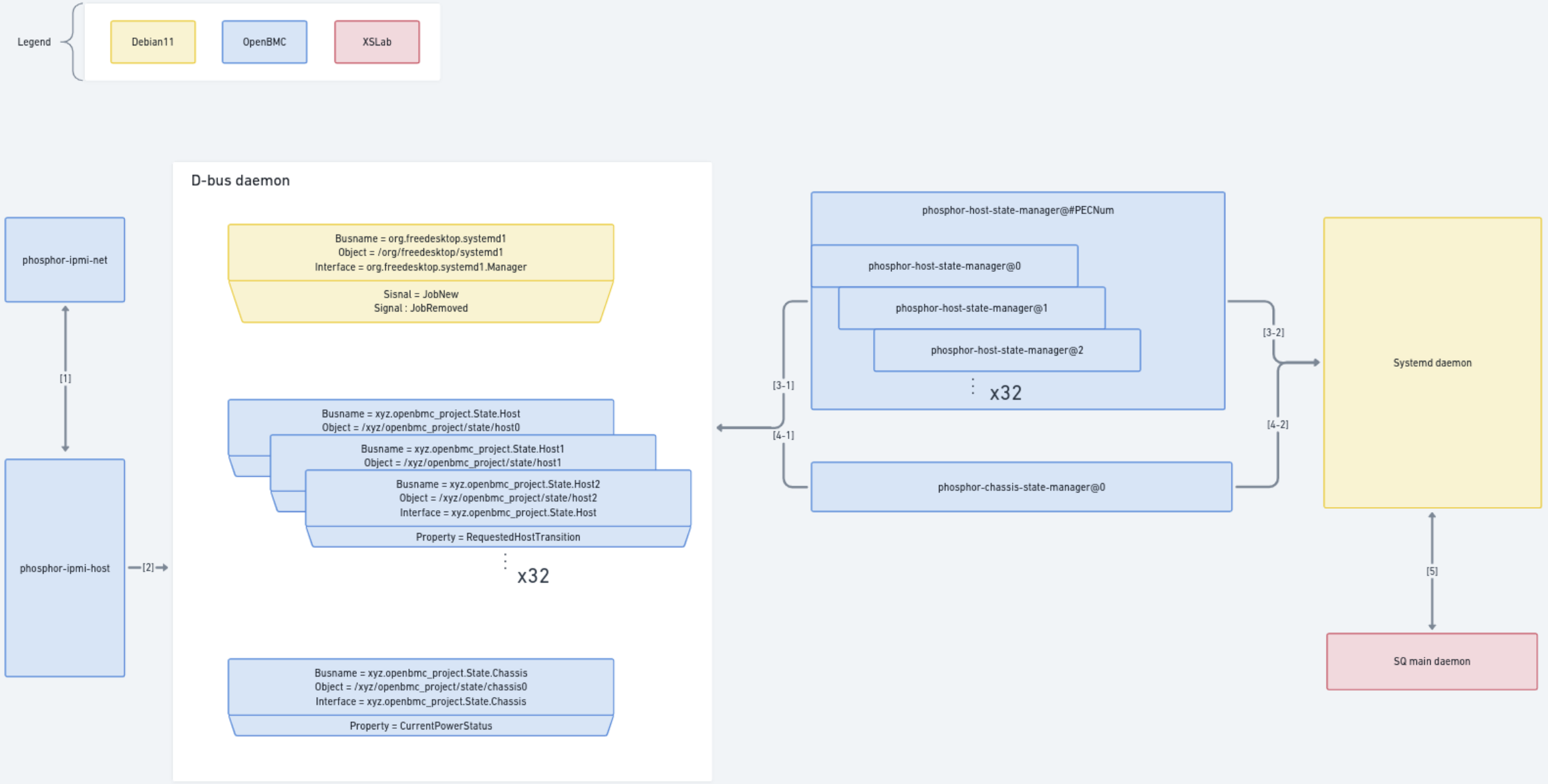
OpenBMC 삽질 : 다중 Host CPU 이슈 해결

```
int initiateHostStateTransition(ipmi::Context::ptr& ctx,
                                State::Host::Transition transition)
{
  // OpenBMC Host State Manager dbus framework
  constexpr auto hostStatePath = "/xyz/openbmc_project/state/host0";
  constexpr auto hostStateIntf = "xyz.openbmc_project.State.Host";

  // Convert to string equivalent of the passed in transition enum.
  auto request = State::convertForMessage(transition);

  std::string service;
  boost::system::error_code ec =
      ipmi::getService(ctx, hostStateIntf, hostStatePath, service);

  if (!ec)
  {
      ec = ipmi::setDbusProperty(ctx, service, hostStatePath, hostStateIntf,
                                 "RequestedHostTransition", request);
  }
  if (ec)
  {
      log<level::ERR>("Failed to initiate transition",
                      entry("EXCEPTION=%s, REQUEST=%s", ec.message().c_str(),
                            request.c_str()));
      return -1;
  }
  log<level::INFO>(
      "Transition request initiated successfully",
      entry("USERID=%d, REQUEST=%s", ctx->userId, request.c_str()));
  return 0;
}
```

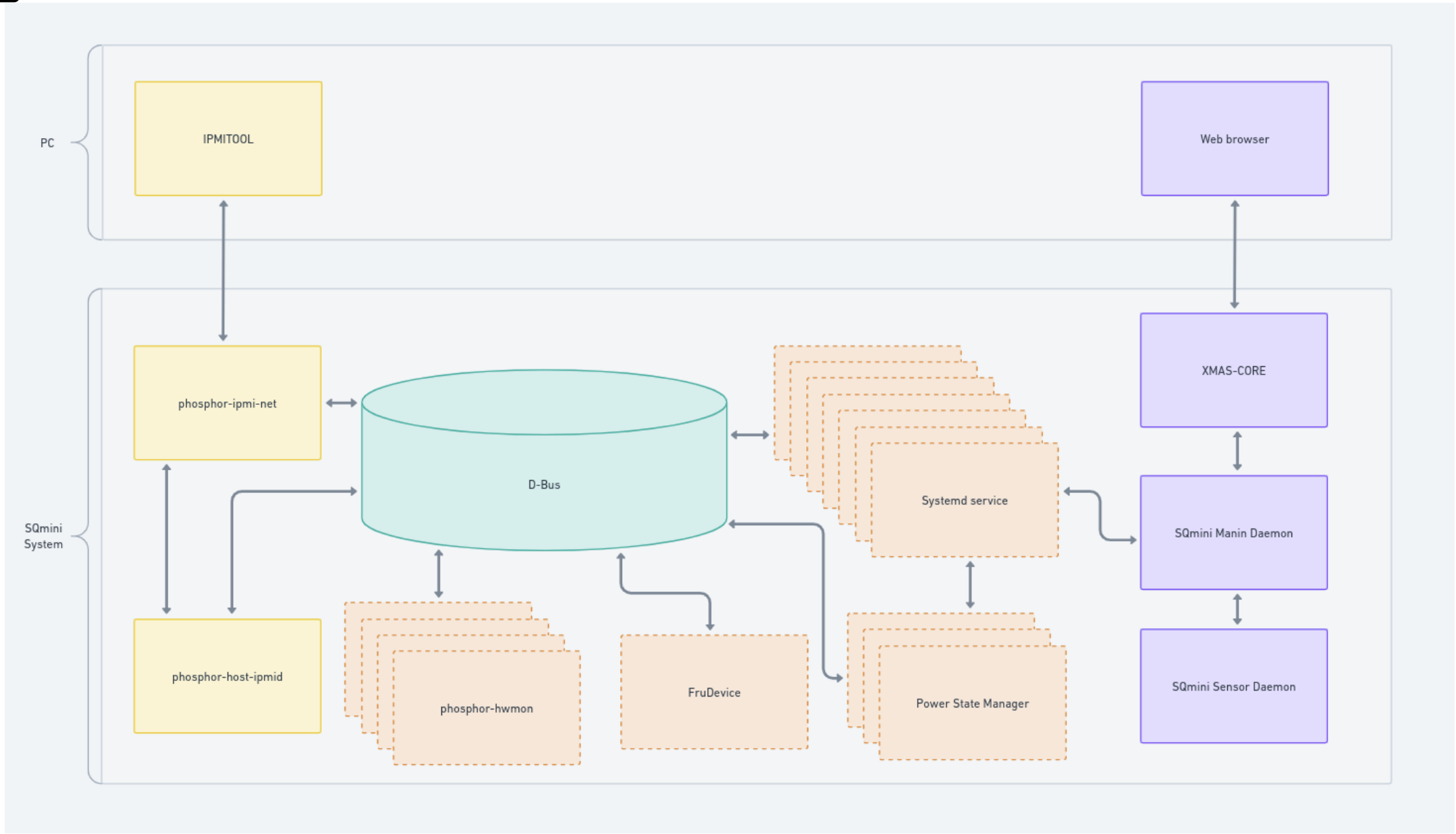
```
int XSinitiateHostStateTransition(const int nPecNum,
                                  ipmi::Context::ptr& ctx,
                                  State::Host::Transition transition)
{

  // OpenBMC Host State Manager dbus framework
  std::string hostStateIntf = "xyz.openbmc_project.State.Host";
  std::string strPecNum = std::to_string(nPecNum);

  std::string strTempHostStatePath("/xyz/openbmc_project/state/host");

  std::string hostStatePath = strTempHostStatePath + std::to_string(nPecNum);

  // Convert to string equivalent of the passed in transition enum.
  auto request = State::convertForMessage(transition);

  std::string service;
  boost::system::error_code ec =
      ipmi::getService(ctx, hostStateIntf, hostStatePath, service);

  if (!ec)
  {
      ec = ipmi::setDbusProperty(ctx, service, hostStatePath, hostStateIntf,
                                 "RequestedHostTransition", request);
  }
  if (ec)
  {
      log<level::ERR>("Failed to initiate transition",
                      entry("EXCEPTION=%s, REQUEST=%s", ec.message().c_str(),
                            request.c_str()));
      return -1;
  }
  log<level::INFO>(
      "Transition request initiated successfully",
      entry("USERID=%d, REQUEST=%s", ctx->userId, request.c_str()));
  return 0;
}
```

서버 Node 번호 처리 부분을 추가

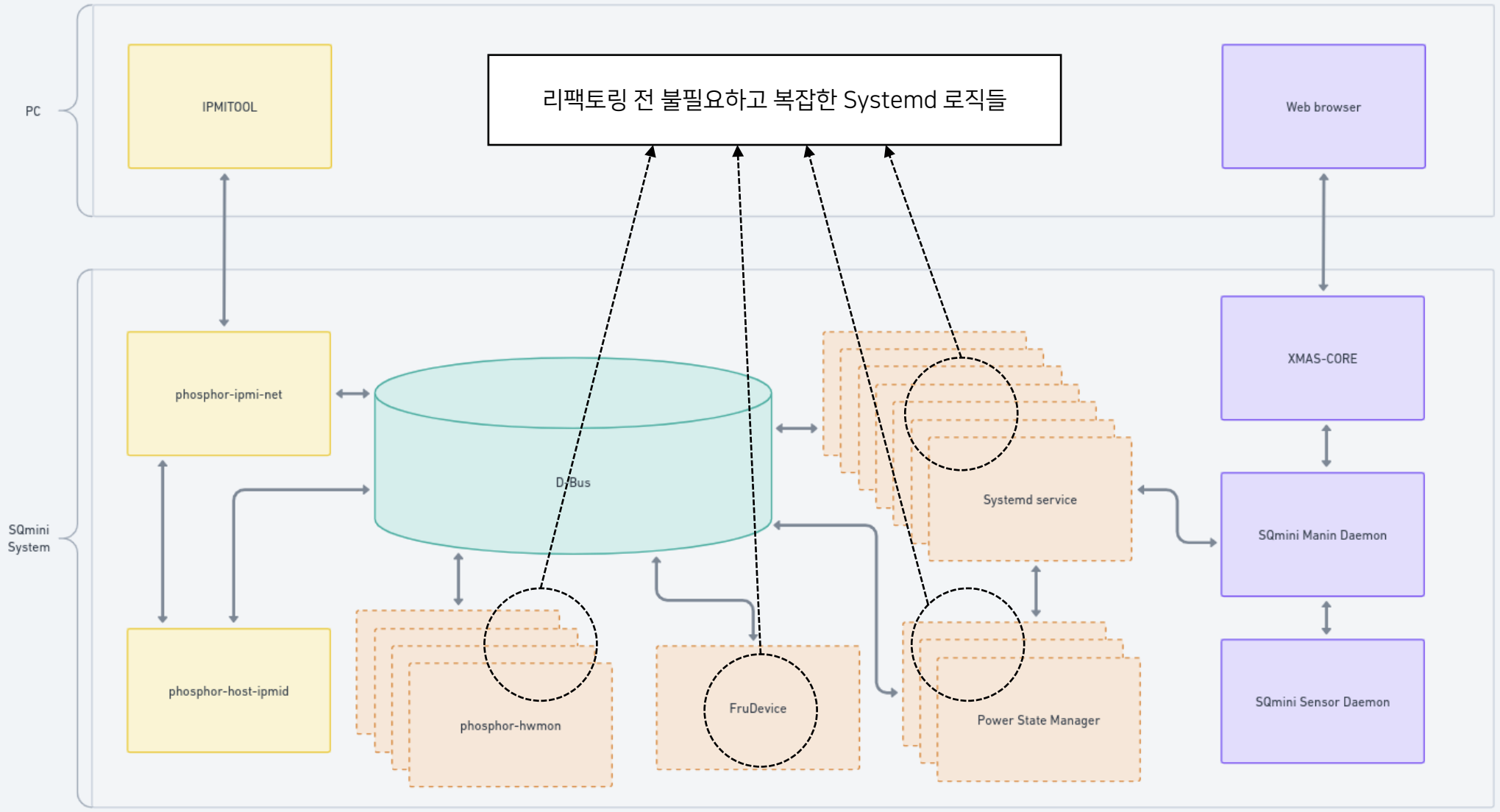
OpenBMC 삽질 : 다중 Host CPU 이슈 해결

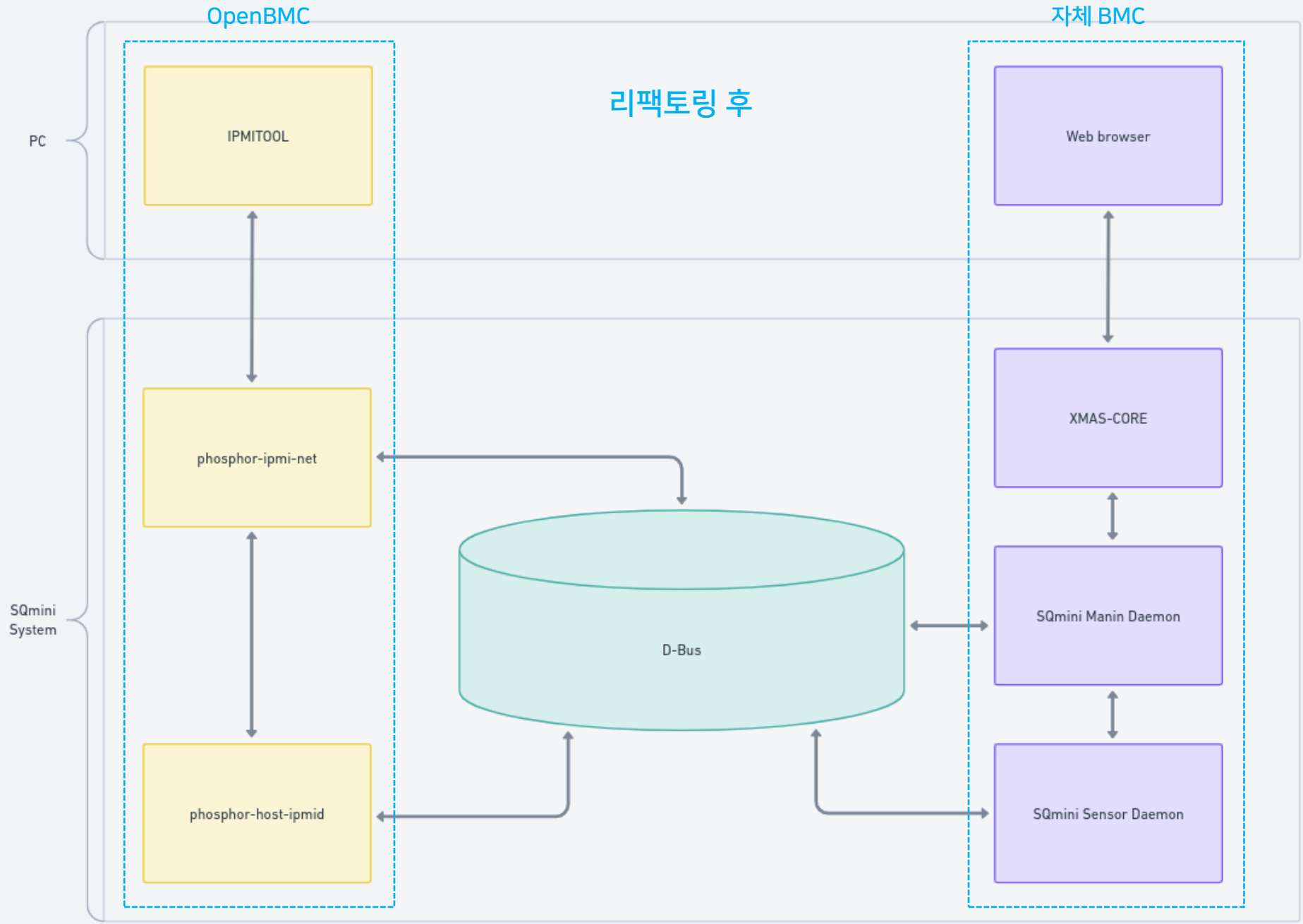


OpenBMC 삽질 : 최종 삽질 Refactoring

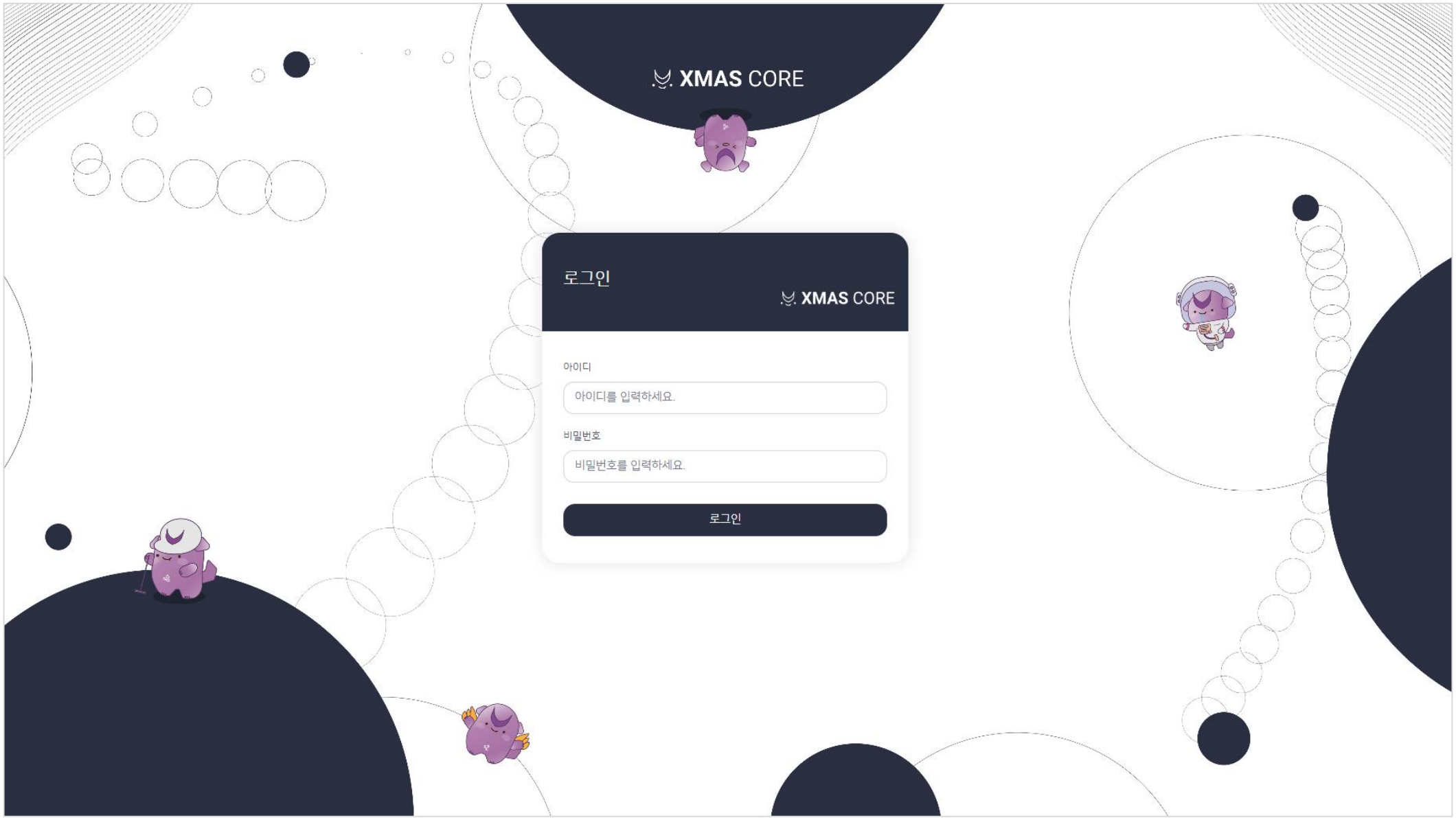


OpenBMC 삽질 : 최종 삽질 Refactoring




OpenBMC
자체 BMC
리팩토링 후
PC
IPMITOOL
Web browser
phosphor-ipmi-net
XMAS-CORE
SQmini
System
D-Bus
SQmini Manin Daemon
phosphor-host-ipmid
SQmini Sensor Daemon

삽질의 결과 : 한국산 토종 BMC 탄생



OpenInfra DAYS Korea

삽질의 결과 : 한국산 토종 BMC 탄생

사용자 안내서

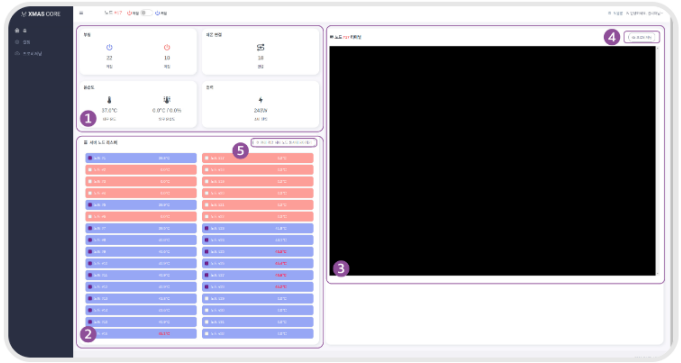
User Guide

메인 페이지 Main

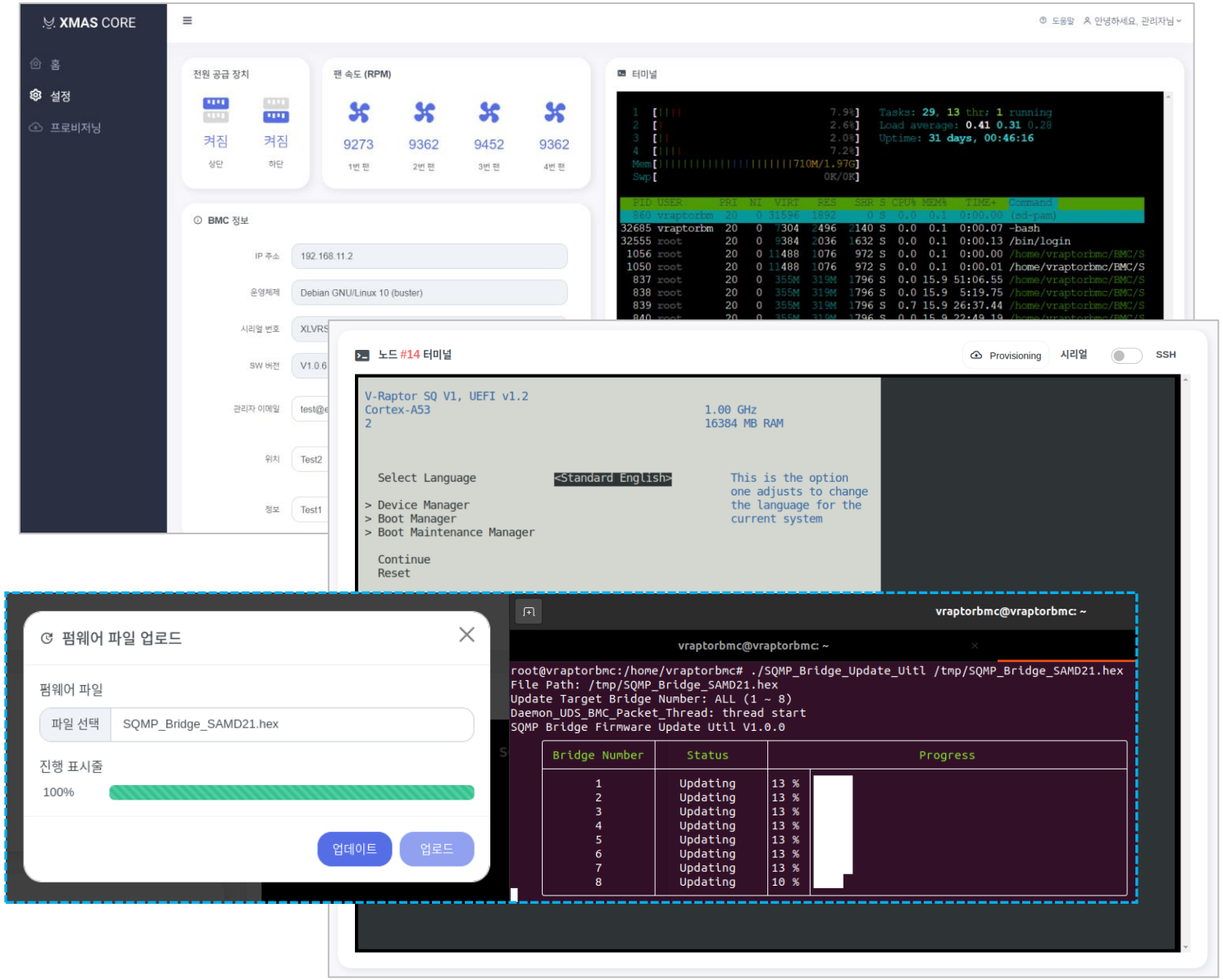
설정 페이지 Setting

프로비저닝 페이지 Provisioning

메인 Main



* 이미지를 클릭하시면 확대된 이미지를 보실 수 있다.



엑세스랩 주식회사

[주 소] 서울특별시 구로구 디지털로 33길 11 에이스테크노타워 8차 701호

[Contact] Tel. 02-6952-9974, E-mail. yoo@xslab.co.kr

701, ACE Techno Tower 8th

11, Digital-ro 33-gil, Guro-gu, Seoul, Republic of Korea.

It is close to Exit 3 of Guro Digital Complex Station on Seoul Subway Line 2.

XSLAB 엑세스랩(주)