RISC-V 64비트 리눅스 커널 v5.x 포팅

1. 발표자 소개
2. CPU 성능 평가 지표
3. RISC-V 커널 포팅
4. 활용 예제(시연)
5. 질문 & 응답

커널연구회(www.kernel.bz) / 정재준(rgbi3307@naver.com)

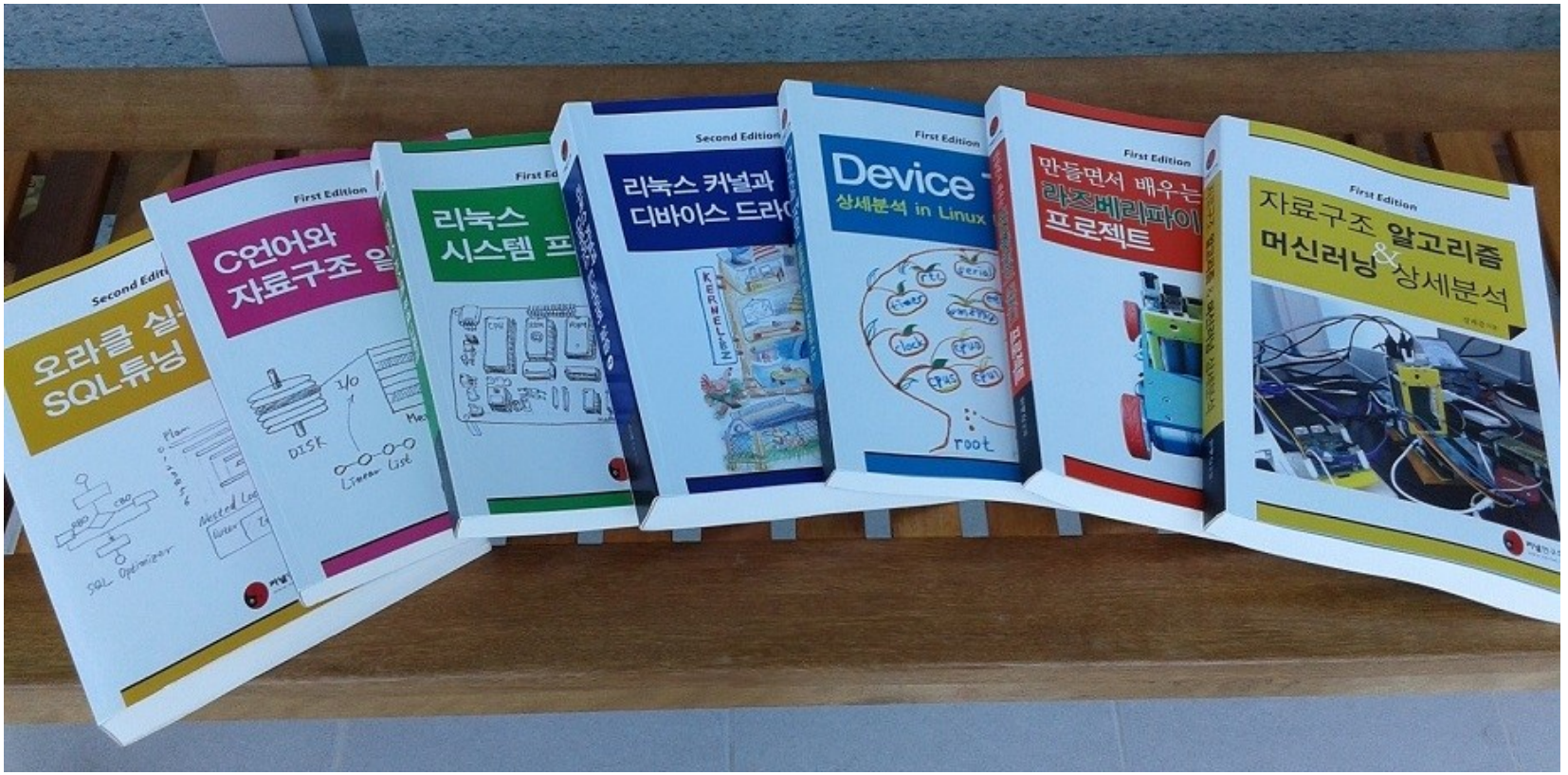
1.1 발표자 소개

정재준 (rgbi3307@naver.com) / 커널연구회(www.kernel.bz)
부천대학교 지능로봇과 겸임교수

저자는 학창시절 마이크로프로세서 제어 기술을 배웠으며 리눅스 커널을 연구하고 있다. 19년 이상 쌓아온 실무 경험을 바탕으로 "C언어와 자료구조 알고리즘", "리눅스 시스템 프로그래밍", "리눅스 커널과 디바이스드라이버 실습2", "자료구조 알고리즘 & 머신러닝 상세분석"등의 책을 집필하고, "맞춤형 문장 자동 번역 시스템 및 이를 위한 데이터베이스 구축방법 (The System for the customized automatic sentence translation and database construction method)" 라는 내용으로 프로그래밍을 하여 특허청에 특허등록 하였다. 최근에는 서울시 버스와 지하철 교통카드 요금결재 단말기에 들어가는 리눅스 커널과 디바이스 드라이버 개발 프로젝트를 성공적으로 수행했고 여러 가지 임베디드 제품을 개발했다. 저자는 스탠포드대학교의 John L. Hennessy 교수의 저서 "Computer Organization and Design" 책을 읽고 깊은 감명을 받았으며, 컴퓨터구조와 자료구조 알고리즘 효율성 연구를 통한 기술서적 집필에 노력하고 있다. 저자는 **커널연구회(http://www.kernel.bz/)** 웹사이트를 운영하며 연구개발, 교육, 관련기술 공유 등을 위해 노력하고 있다.

1.2 발표자 소개(집필 서적)

강사가 집필한 서적들



시중서점(알라딘, 교보문고, 예스24, 인터파크, 영풍문고)등에서
"커널연구회"로 검색하여 구매가능.

2.1 CPU 성능 평가 지표

CPI (Clock cycles per Instruction)

$$\text{CPI} = \frac{\text{CPU clock cycles}}{\text{Instruction count}}$$

예제 참조:
https://en.wikipedia.org/wiki/Cycles_per_instruction

MIPS (Million Instructions Per Second)

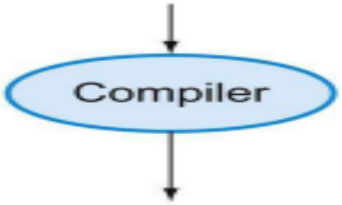
$$\text{MIPS} = \frac{\text{Instruction count}}{\text{Execution time} \times 10^6}$$

$$\text{MIPS} = \frac{\text{Instruction count}}{\frac{\text{Instruction count} \times \text{CPI}}{\text{Clock rate}} \times 10^6} = \frac{\text{Clock rate}}{\text{CPI} \times 10^6}$$

2.2 RISC-V 컴파일러

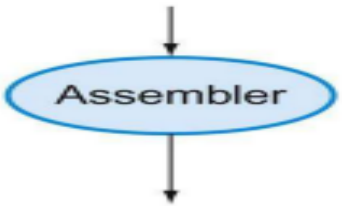
High-level language program (in C)

```
swap(size_t v[], size_t k)
{
    size_t temp;
    temp = v[k];
    v[k] = v[k+1];
    v[k+1] = temp;
}
```



Assembly language program (for RISC-V)

```
swap:
        slli  x6, x11, 3
        add   x6, x10, x6
        ld    x5, 0(x6)
        ld    x7, 8(x6)
        sd    x7, 0(x6)
        sd    x5, 8(x6)
        jalr  x0, 0(x1)
```



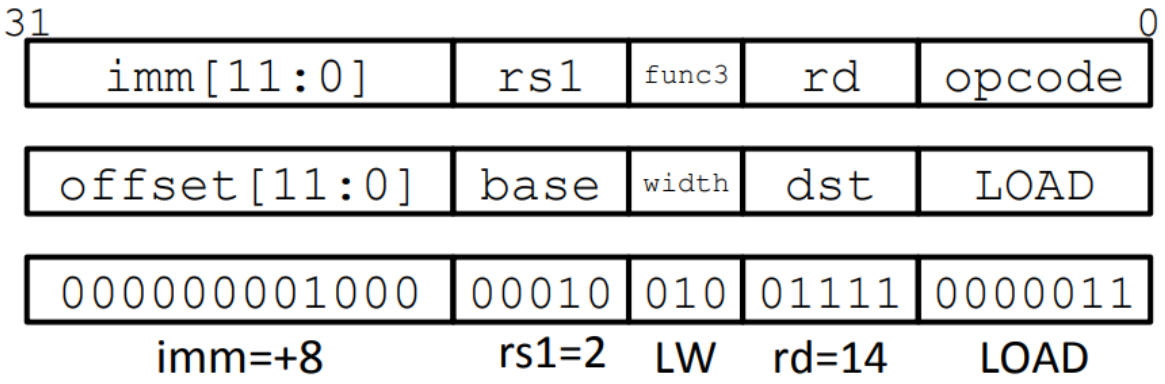
Binary machine language program (for RISC-V)

```
00000000001101011001001100010011
00000000011001010000001100110011
00000000000000011001100101000011
00000000100000011001100111000011
00000000011100110011000000100011
00000000010100110011010000100011
00000000000000001000000001100111
```

2.3 RISC-V 명령셋(ISA)

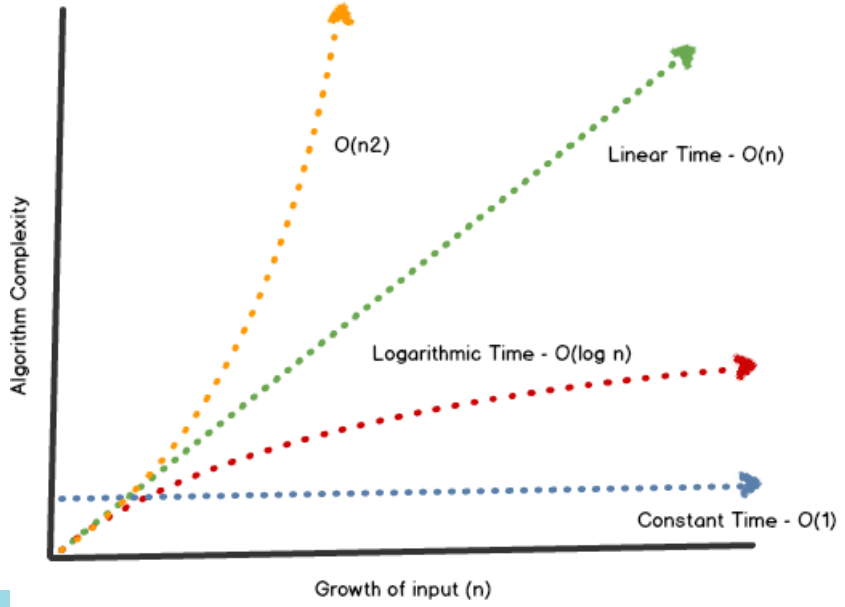
Figure 1.1: RISC-V instruction length encoding.

- **lw x14, 8(x2)**

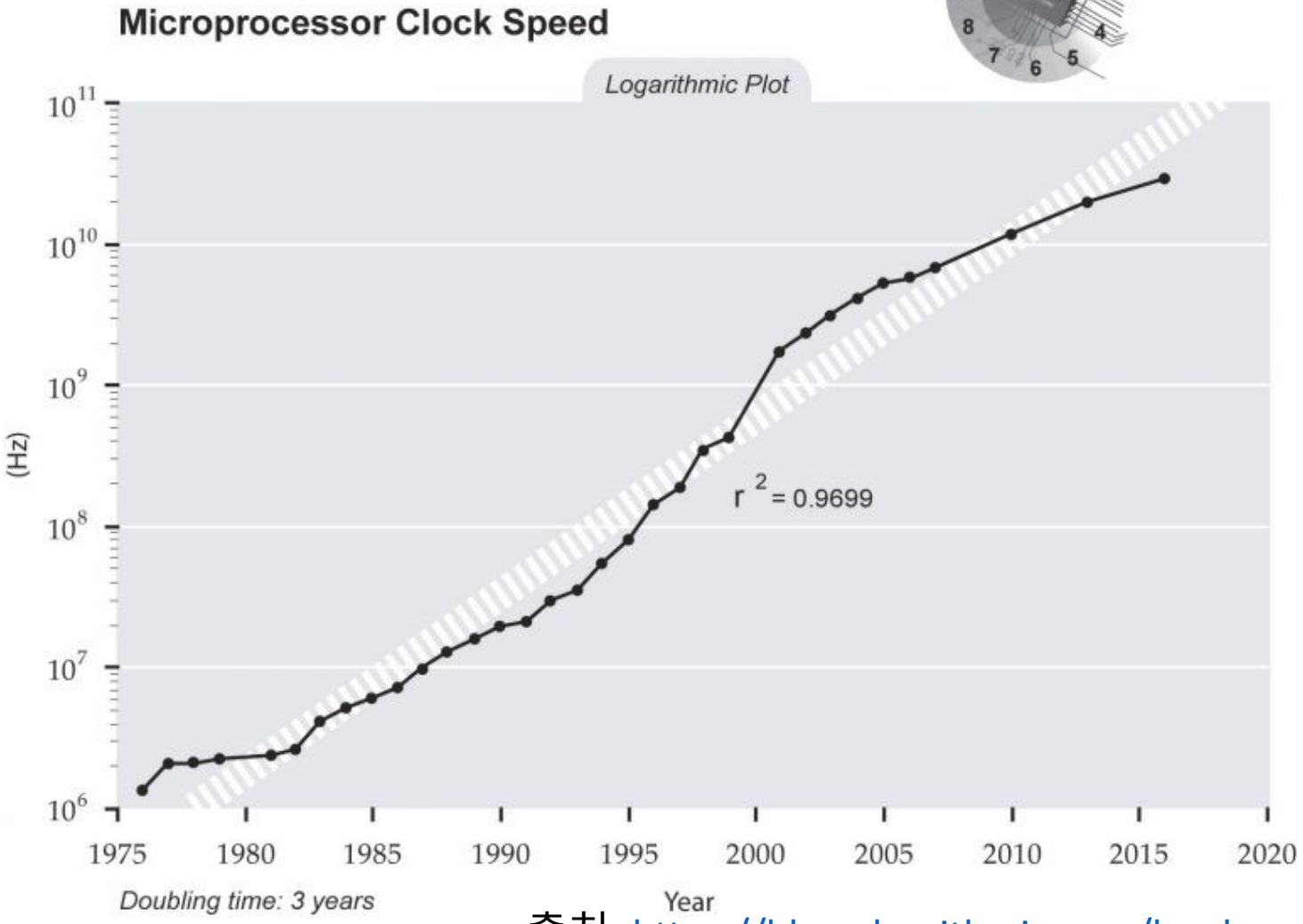


2.4 데이터 처리 용량과 알고리즘

Name	Abbr	Factor	SI size
Kilo	K/k	2^{10} = 1,024 (K)	10^{3} = 1,000 (k)
Mega	M	2^{20} = 1,048,576	10^{6} = 1,000,000
Giga	G	2^{30} = 1,073,741,824	10^{9} = 1,000,000,000
Tera	T	2^{40} = 1,099,511,627,776	10^{12} = 1,000,000,000,000
Peta	P	2^{50} = 1,125,899,906,842,624	10^{15} = 1,000,000,000,000,000
Exa	E	2^{60} = 1,152,921,504,606,846,976	10^{18} = 1,000,000,000,000,000,000
Zetta	Z	2^{70} = 1,180,591,620,717,411,303,424	10^{21} = 1,000,000,000,000,000,000,000
Yotta	Y	2^{80} = 1,208,925,819,614,629,174,706,176	10^{24} = 1,000,000,000,000,000,000,000,000



2.5 CPU 처리 속도(Clock)

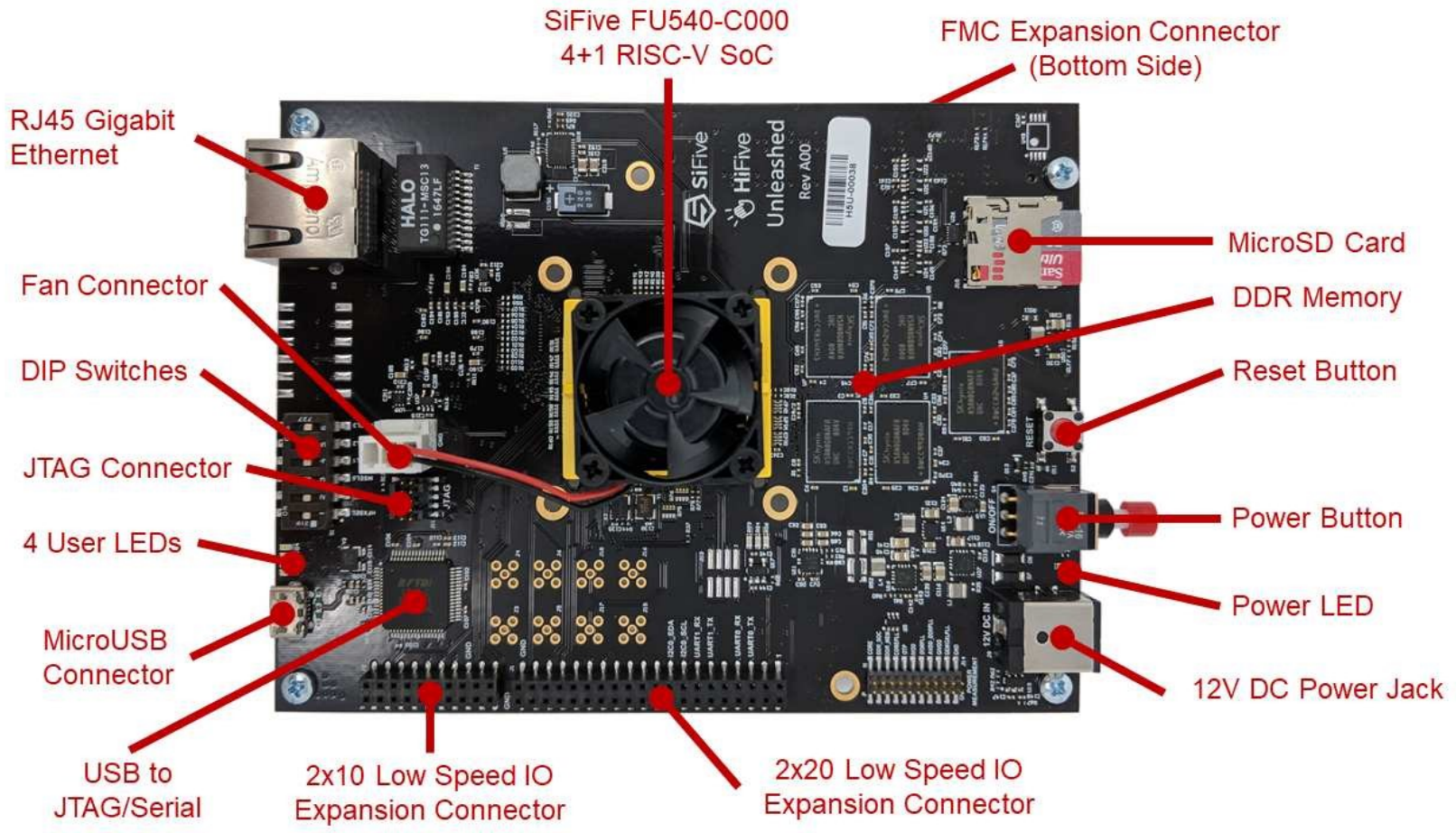


출처: https://blog.algorithmia.com/hardware-for-machine-learning/

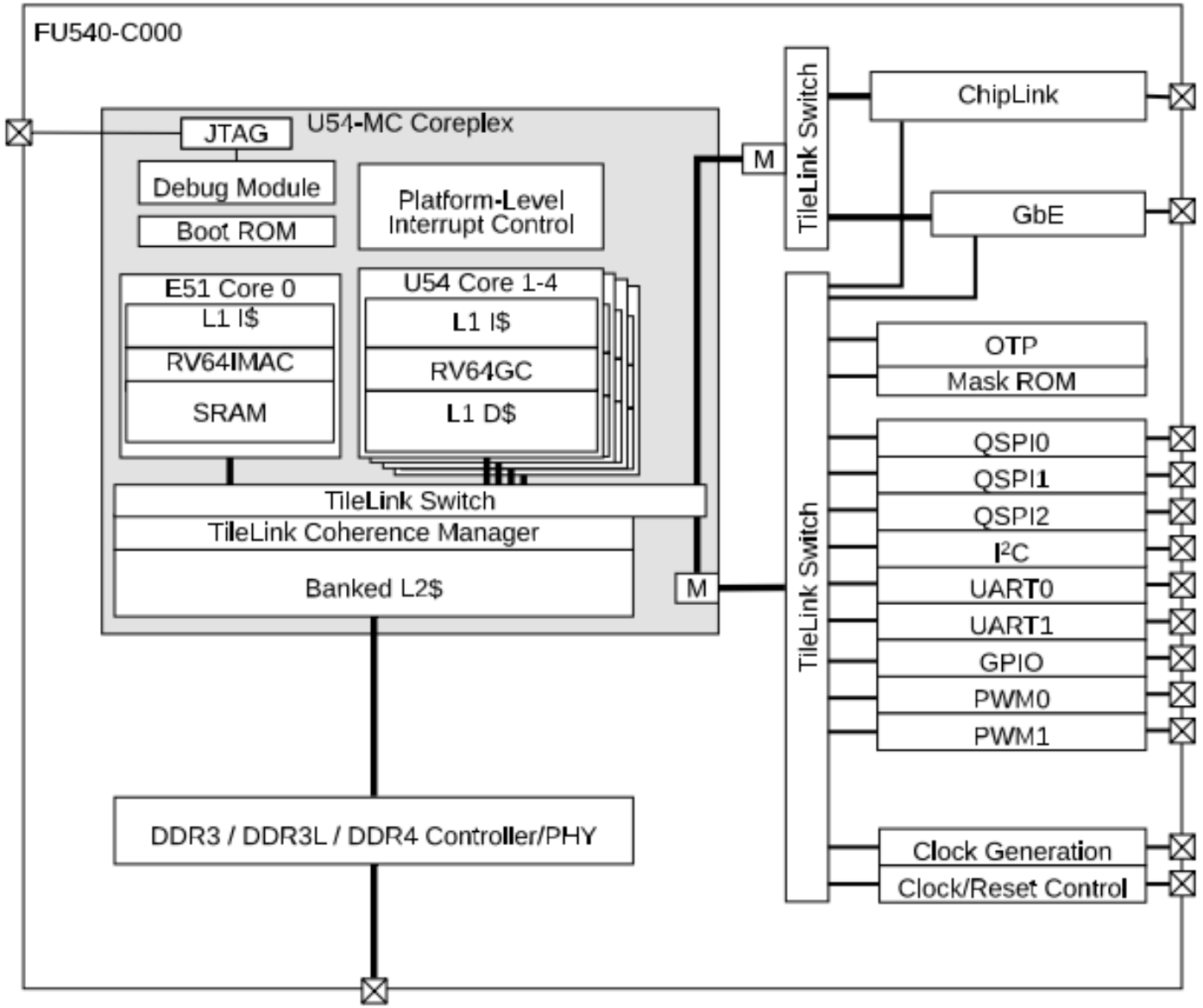
3.1 RISC-V 커널 포팅 (타겟 보드)

https://www.sifive.com/boards/hifive-unleashed

kernel v4.15 → v5.1



3.2 RISC-V 커널 포팅 (타겟 보드 사양)



3.3 RISC-V 커널 포팅

https://github.com/riscv

riscv 컴파일러 다운로드/버전확인

$ git clone --recursive https://github.com/riscv/riscv-gnu-toolchain
$ riscv64-unknown-linux-gnu-gcc -v
COLLECT_GCC=./bin/riscv64-unknown-linux-gnu-gcc
gcc version 8.3.0 (GCC)

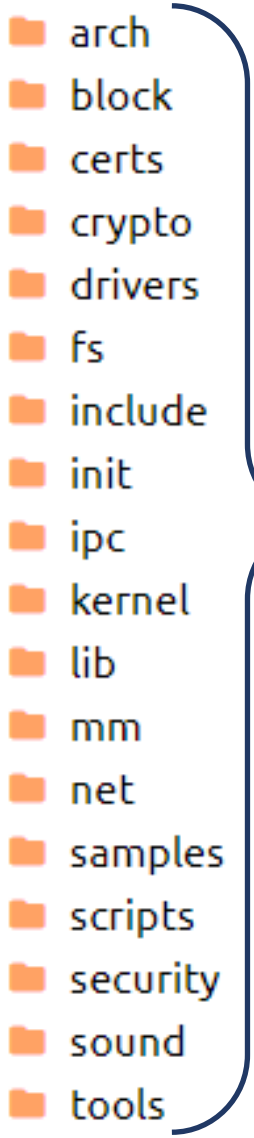
커널 소스 다운로드/버전확인/빌드

$ git clone git://git.kernel.org/pub/scm/linux/kernel/git/stable/linux-stable.git (origin)

$ git log -p
commit e93c9c99a629c61837d5a7fc2120cd2b6c70dbdd (HEAD -> CV5.1, tag: v5.1, origin/master, origin/linux-5.1.y, origin/HEAD, cv5.1)
Author: Linus Torvalds <torvalds@linux-foundation.org>
Date: Sun May 5 17:42:58 2019 -0700
Linux 5.1

$ make ARCH=riscv CROSS_COMPILE=riscv64-unknown-linux-gnu- defconfig

3.4 RISC-V 커널 포팅 (v4.15 → v5.1)



git diff v4.15 v5.1 –commpact-summary

```
arch/riscv/kernel/setup.c                          |  220 +-
arch/riscv/kernel/signal.c                         |   79 +-
arch/riscv/kernel/smp.c                            |
arch/riscv/kernel/smpboot.c                        |
arch/riscv/kernel/stacktrace.c                     |
arch/riscv/kernel/sys_riscv.c                      |
arch/riscv/kernel/time.c                           |
arch/riscv/kernel/traps.c                          |   19 +-
arch/riscv/kernel/vdso.c                           |    1 -
arch/riscv/kernel/vdso/Makefile                    |    6 +-
arch/riscv/kernel/vmlinux.lds.S                    |    2 +-
arch/riscv/lib/Makefile                            |    2 +
arch/riscv/lib/tishift.S (new)                     |
arch/riscv/lib/uaccess.S                           |
arch/riscv/lib/udivdi3.S                           |
arch/riscv/mm/Makefile                             |
arch/riscv/mm/dma.c (gone)                         |
arch/riscv/mm/fault.c                              |   14 +-
arch/riscv/mm/init.c                               |  219 +-
arch/riscv/mm/ioremap.c                            |    2 +-
arch/riscv/net/Makefile (new)                      |    1 +
arch/riscv/net/bpf_jit_comp.c (new)                | 1602 +
build_riscv.sh (new +x)                            |
drivers/clocksource/riscv_timer.c (gone)           |
drivers/clocksource/timer-riscv.c (new)            |
drivers/tty/serial/earlycon-riscv-sbi.c (new)      |
include/linux/irqchip/irq-riscv-intc.h (gone)      |
include/linux/timer_riscv.h (gone)                 |
.../riscv/include/uapi/asm/bitsperlong.h (new)     |   25 +
tools/arch/riscv/include/uapi/asm/unistd.h (new)   |   42 +
```

소스 편집(추가,수정)
***.c**

빌드설정
Kconfig

빌드적용
Makefile

3.5 RISC-V 커널 포팅 (v4.15 → v5.1)

FU540 CPU 아키텍쳐 포팅
arch/riscv/kernel/setup.c

FU540 클럭 포팅
drivers/clk/sifive/*

FU540 timer 포팅
drivers/clocksource/timer_riscv.c

FU540 irqchip 포팅
drivers/irqchip/irq-sifive-plic.c

FU540 콘솔 포팅
drivers/tty/hvc/hvc_riscv_sbi.c

FU540 GPIO 포팅
drivers/gpio/gpio-sifive.c

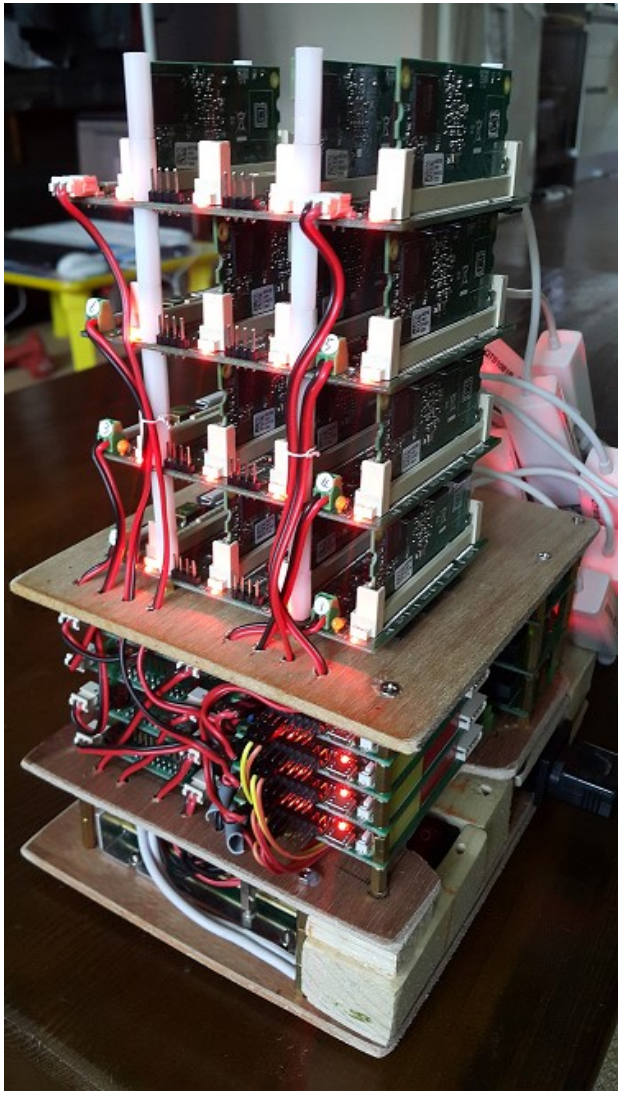
FU540 PWM 포팅
drivers/pwm/pwm-sifive.c

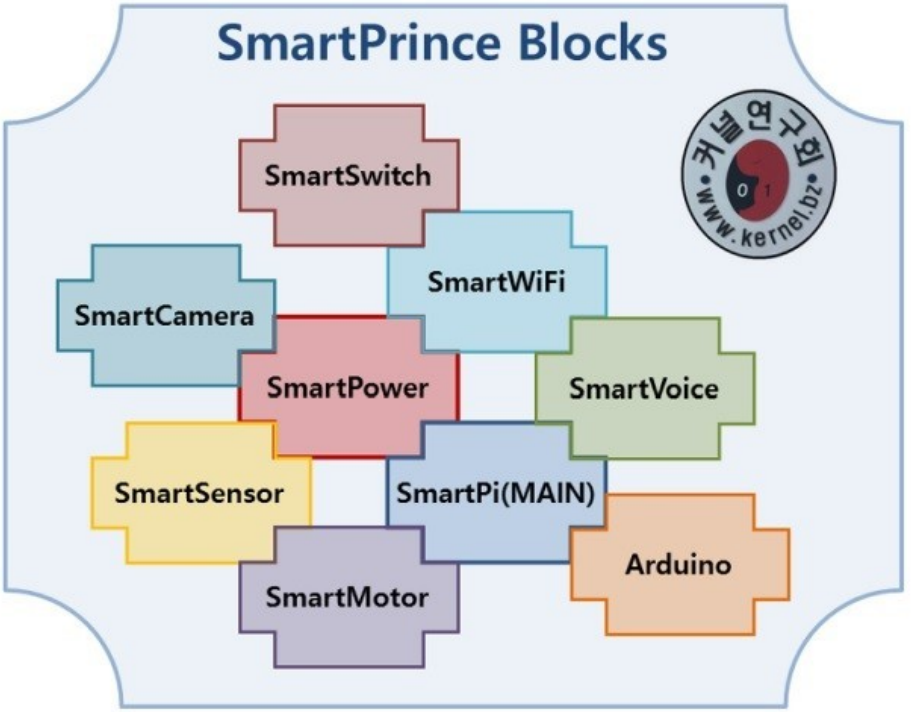
FU540 SPI 포팅
drivers/spi/spi-sifive.c

FU540 MMC/SD/SDIO 포팅
drivers/nvmem/sifive-u500-otp.c

4. RISCV와 병렬처리 클러스터 활용사례(시연)

커널연구회 SmartPrince 병렬처리 클러스터(SmartTower)





- 소개: https://www.kernel.bz/SmartPrince
- 판매: https://www.kernel.bz/product/SmartPrince

5. 질문 & 응답

감사합니다.

강사: 커널연구회(www.kernel.bz) 정재준(rgbi3307@naver.com)

HP: 010-2520-3307