LG 부트캠프 10기

프로젝트 결과보고서

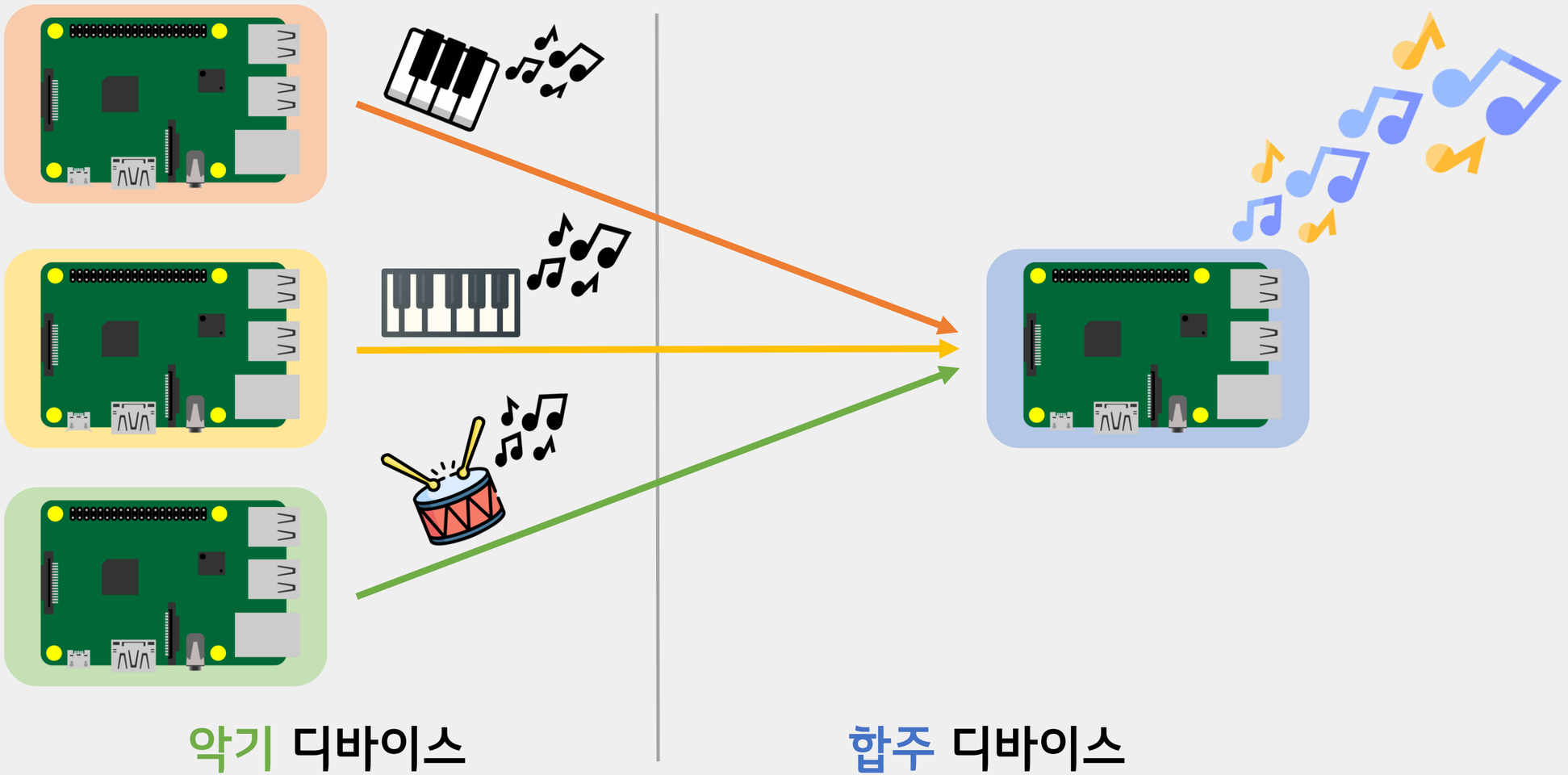
Sound Maker

Linux System 반 6팀 브레멘 합창단

이윤성, 공하영, 안설령, 문명석

1. 프로젝트 개요

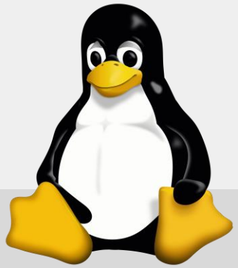
- 여러 대의 악기 디바이스에서 연주 → 하나의 합주 디바이스에서 음 믹싱 후 재생



2. 시연 영상

시연 영상 재생

3-1. 핵심 기술 - Linux Kernel & Device Driver



Linux
v6.12.35

최신 LTS

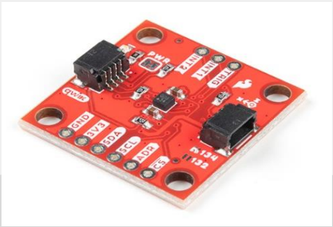
BuildRoot
Making Embedded Linux Easy

RFS

합주 디바이스
실행 환경

Button

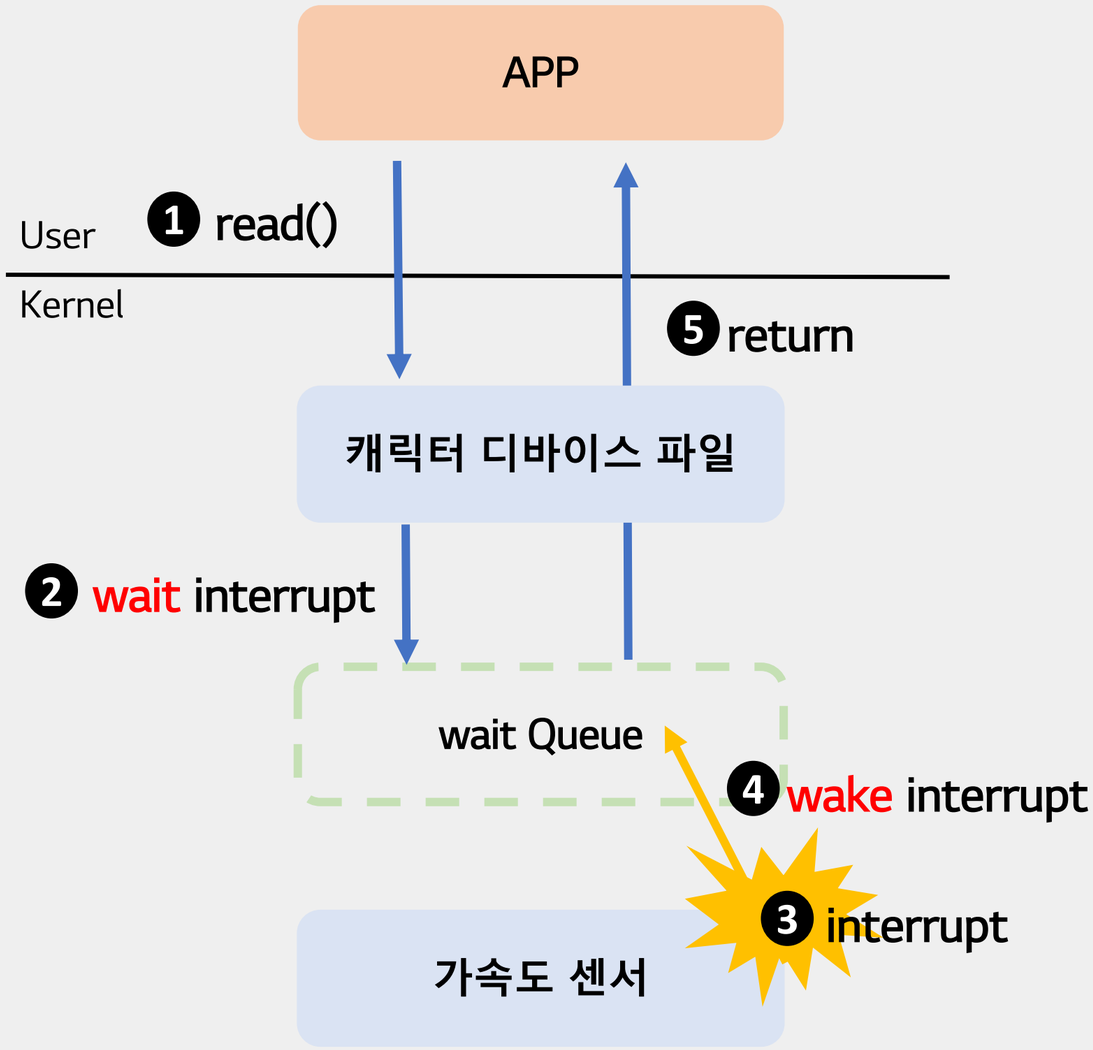
Device Driver



가속도

Device Driver

3-1. 핵심 기술 - Linux Kernel & Device Driver

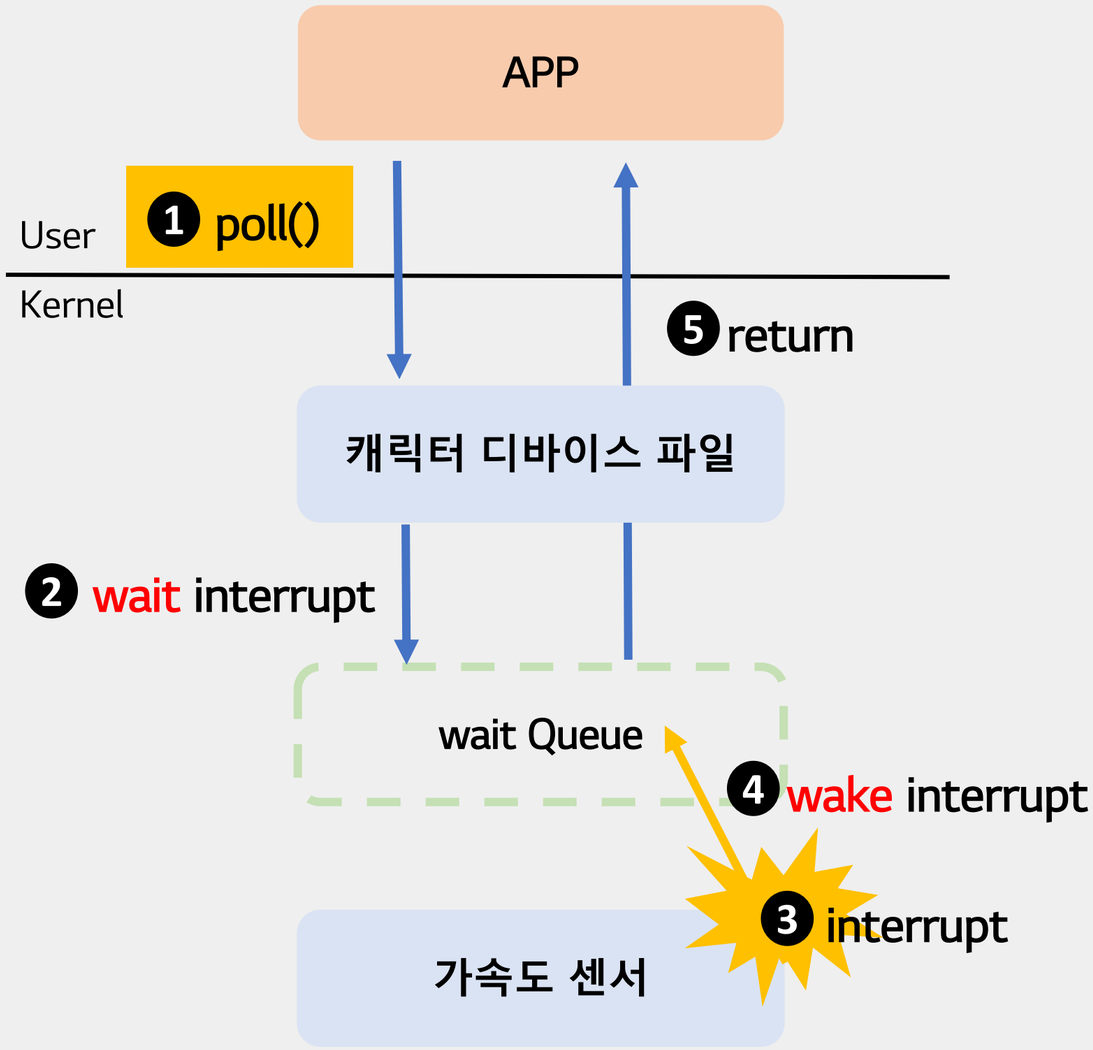


가속도 센서 Interrupt 제어

- APP read Thread 생성 → read()
- read() → Interrupt 발생 전까지 Wait
- 가속도 센서의 충격 감지 기능 활용
- 임계치 이상의 충격이 감지 → Interrupt
- read() Wait 탈출 → return

- 가속도 센서 Interrupt 제어 구현 (커널 소스 코드 수정)
- 캐릭터 디바이스 구현

3-1. 핵심 기술 - Linux Kernel & Device Driver



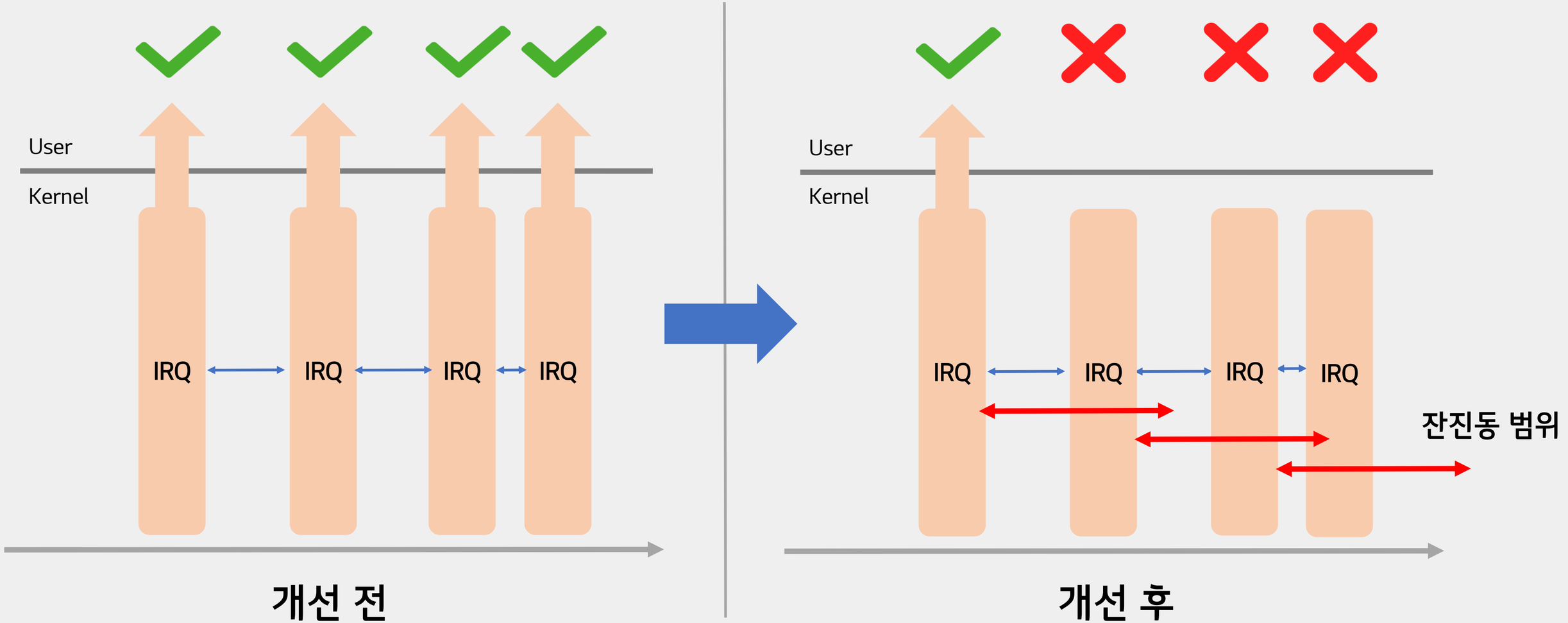
Button Interrupt 제어

- QT QSocketNotifier 사용 → poll()
- read() → Interrupt 발생 전까지 Wait
- 가속도 센서의 충격 감지 기능 활용
- 임계치 이상의 충격이 감지 → Interrupt
- read() Wait 탈출 → return

- Button Interrupt 제어 구현
- 캐릭터 디바이스 구현

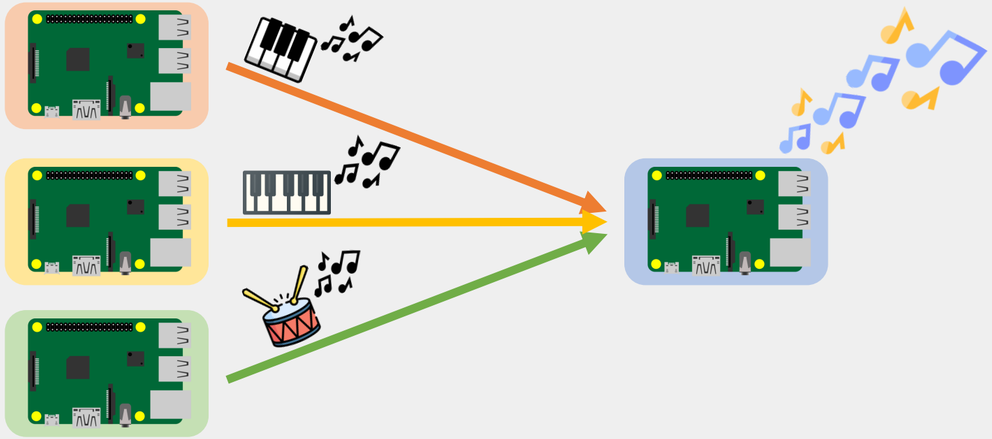
3-1. 핵심 기술 - Linux Kernel & Device Driver

- **가속도 센서 충격 동작 잔진동 제어**
  - Jiffies 사용 → 임계치 이내의 IRQ는 잔진동으로 판단

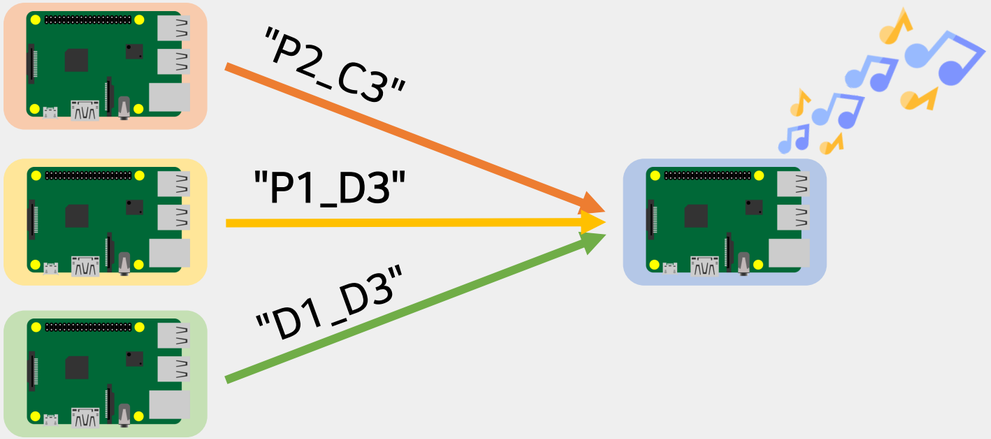


3-2. 핵심 기술 - Audio Latency 개선

- UDP Socket 활용 및 API 정의를 통한 고성능, 저데이터 통신 구현

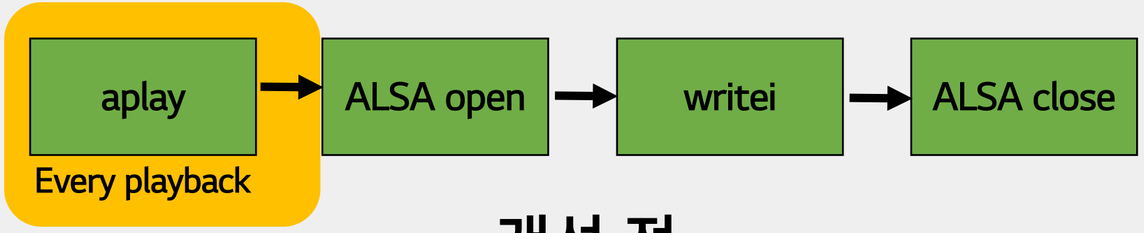


개선 전

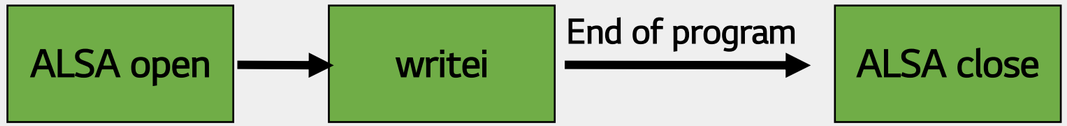


개선 후

- APLAY가 아닌 ALSA lib을 직접 활용해, 음원 실시간 재생시 HW open/close 단계 제거
  - 프로그램 시작 시 ALSA 초기화를 통해 HW open 및 설정



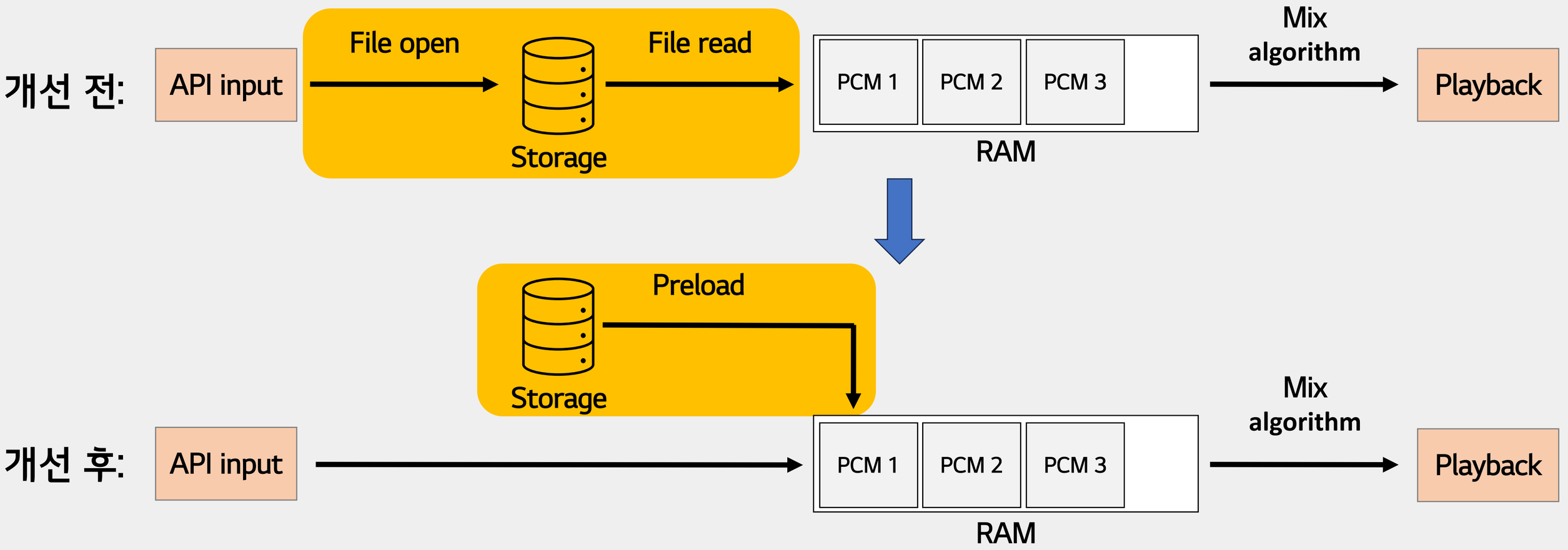
개선 전



개선 후

3-1. 핵심 기술 - Audio Latency 개선

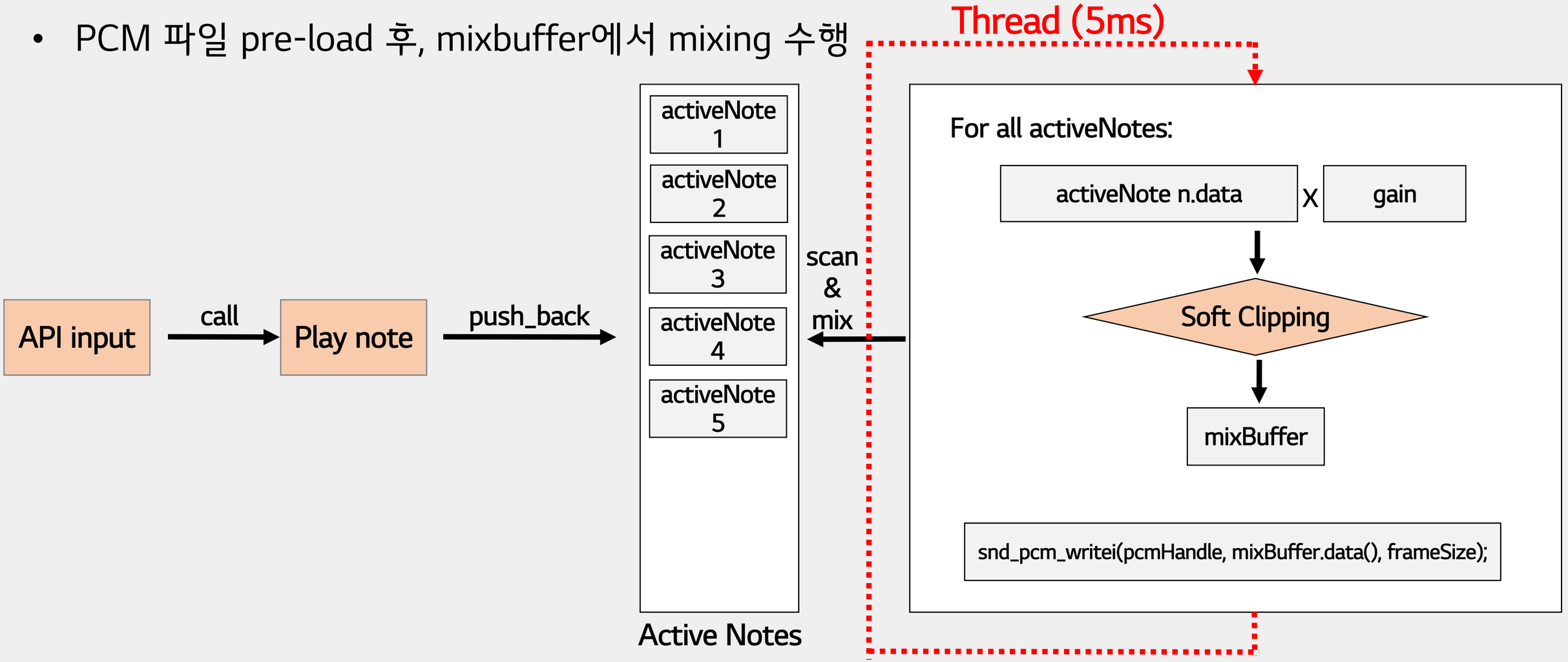
- **음원 파일 RAM Preload를 통해 File I/O Latency 최적화**
  - 프로그램 시작 시 Audio Data Pre-load 구현



** PCM(Pulse Code Modulation) : 신호를 디지던 샘플 값으도 양자화*
*** PCM file : header 없이 순수한 PCM data 저장, 재생 시 sample rate, bit depth, channel 정보 필요*

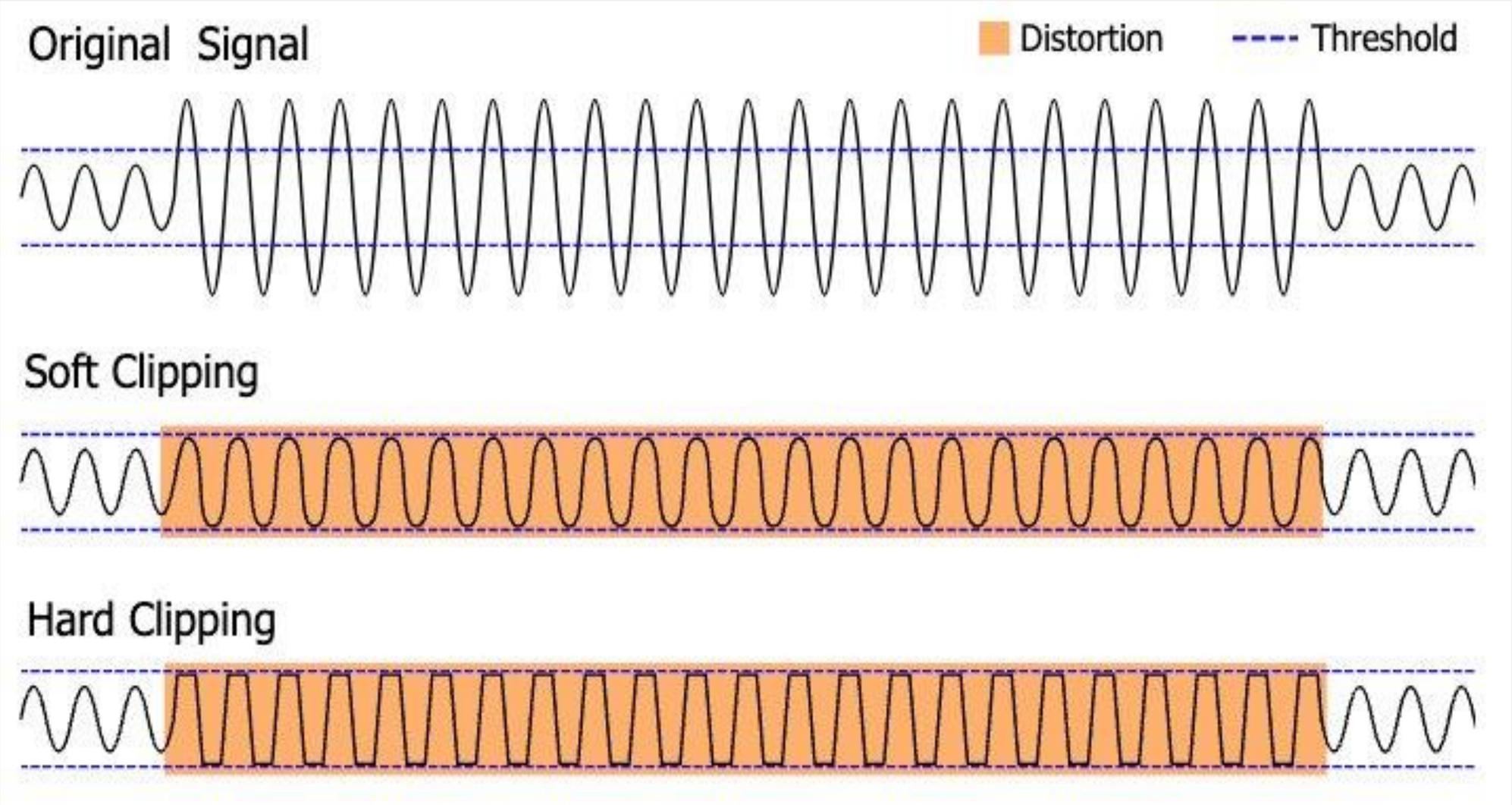
3-1. 핵심 기술 - Audio Mixing

- **SW Mixer 구현**
  - PCM 파일 pre-load 후, mixbuffer에서 mixing 수행

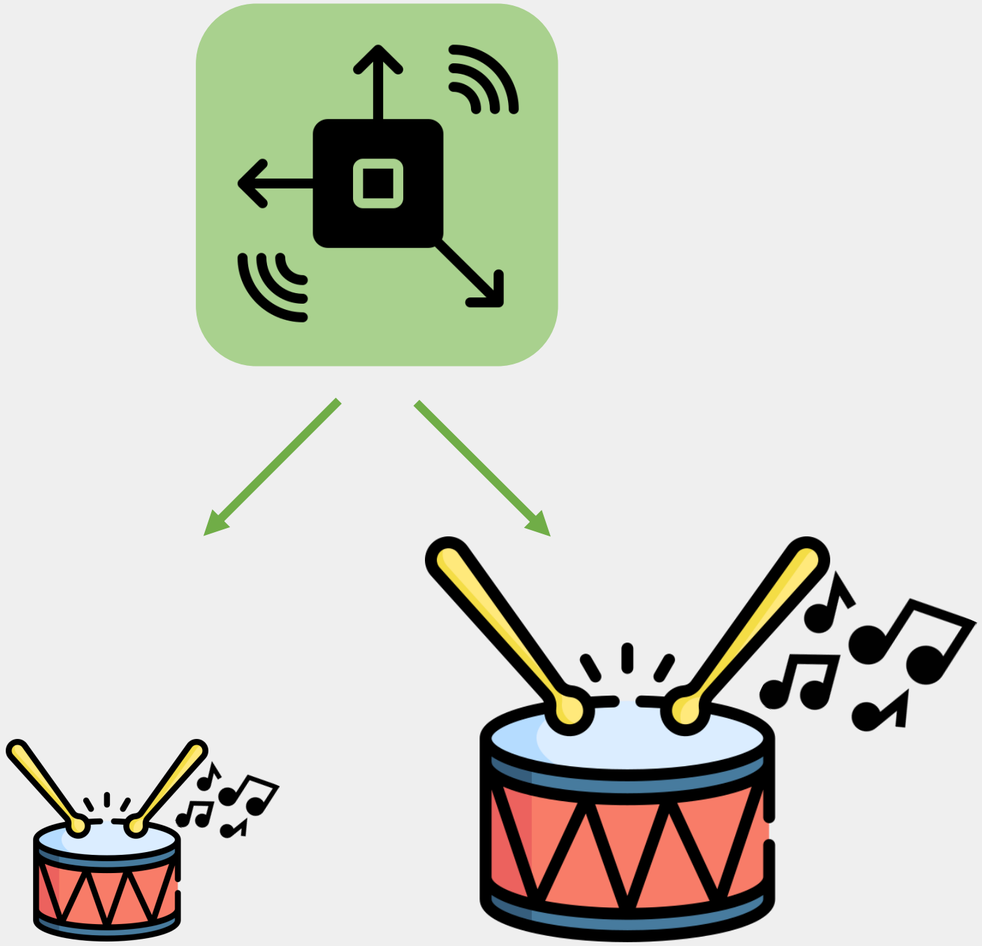


** PCM(Pulse Code Modulation) : 신호를 디지털 샘플 값으로 양자화*
*** PCM file : header 없이 순수한 PCM data 저장, 재생 시 sample rate, bit depth, channel 정보 필요*

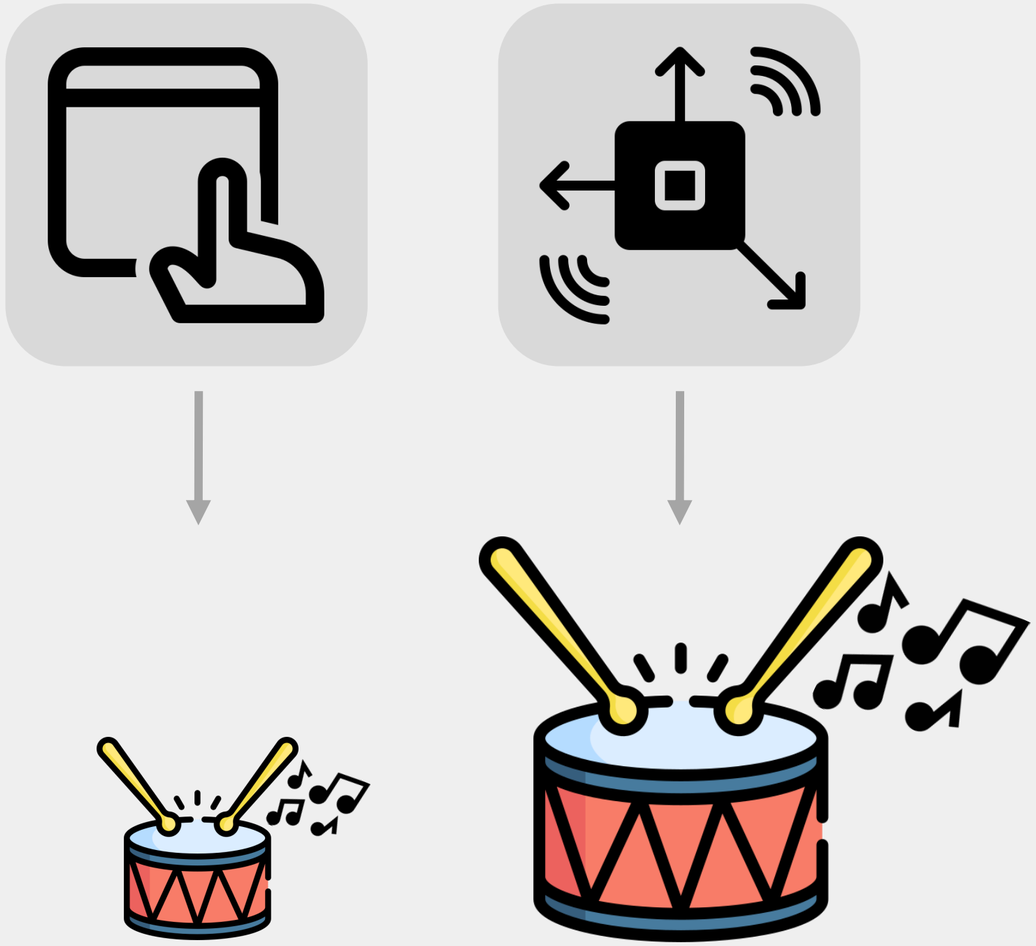
3-2. 핵심 기술 - Audio Mixing



4-1. 향후 연구 과제 - 타악기 입력 강도 감지



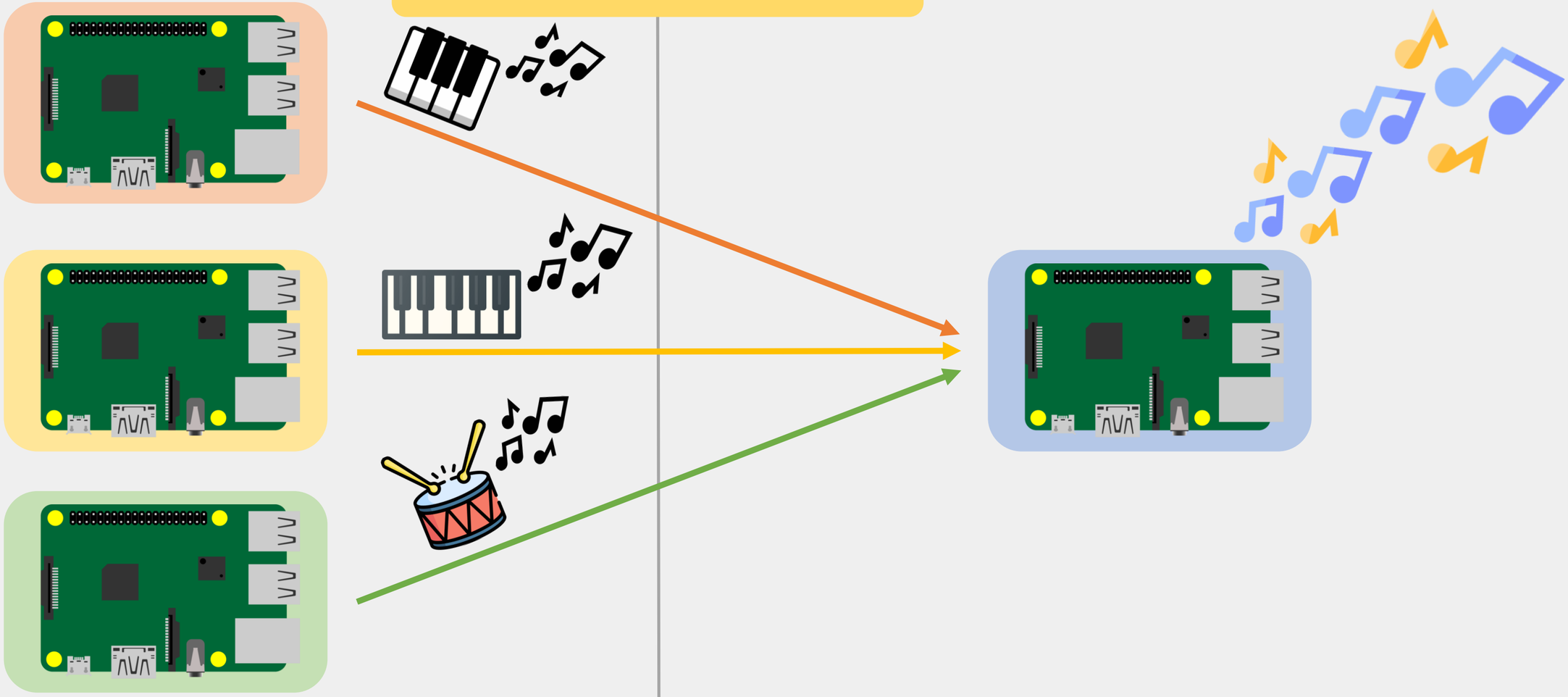
1. 고성능 가속도 센서



2. 기존 터치스크린 + 가속도 센서

4-2. 향후 연구 과제 - 원격 연주

구현 1단계 (완료)

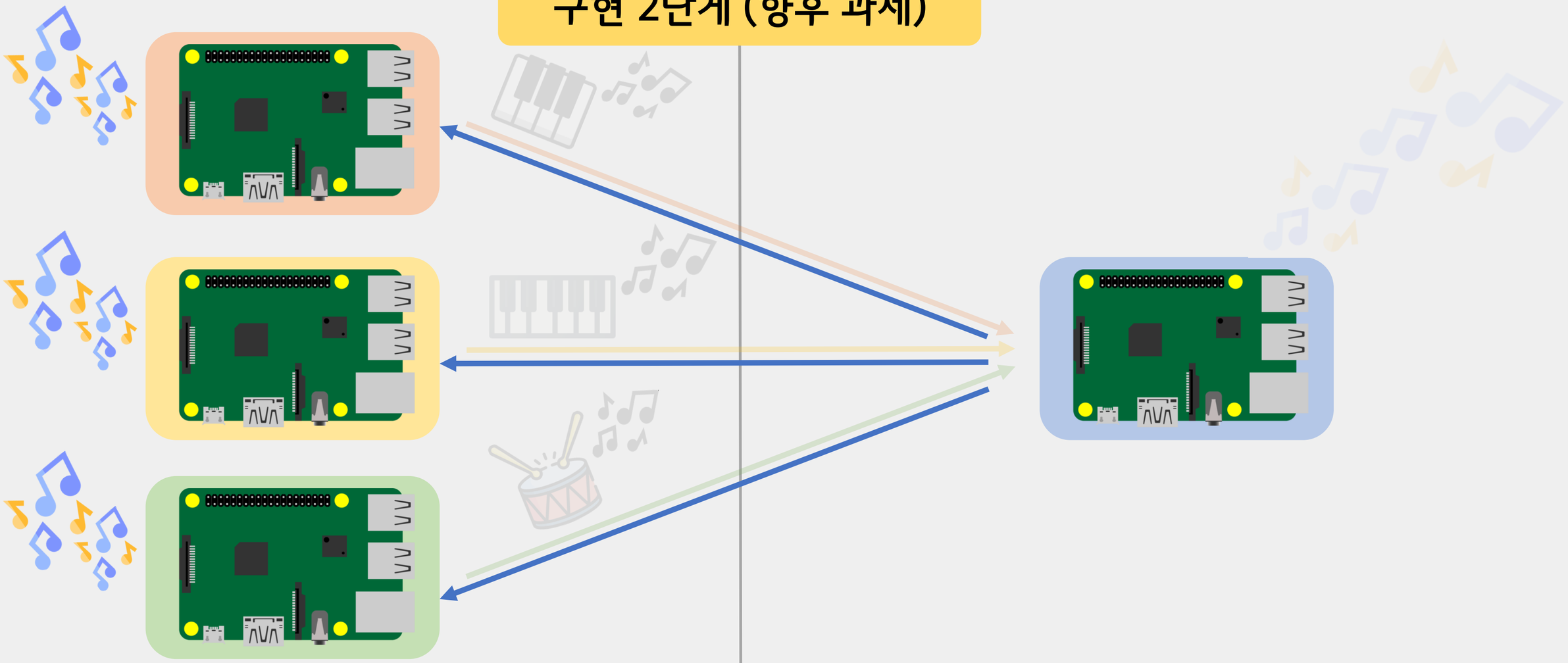


악기 디바이스

합주 디바이스

4-2. 향후 연구 과제 - 원격 연주

구현 2단계 (향후 과제)



악기 디바이스

합주 디바이스

5. 프로젝트 수행 후기

이윤성 연구원

소프트웨어공학과 Linux System programming 강좌에서 배웠던 내용들을 짧은 프로젝트에 적용하면서 실무 활용 가능성을 높일 수 있었던 것 같습니다. 강사님을 통해 많은 내용을 배웠고, 팀원들과 함께 치열하게 프로젝트에 임할 수 있어서 감사했습니다.

문명석 연구원

Embedded Linux Programming 주제에 대해 폭넓게 학습할 수 있었던 좋은 기회였습니다. 프로젝트를 진행하며, 팀 리더십 및 협업의 중요성을 크게 느꼈습니다.

안설령 연구원

커널 포팅부터 디바이스 드라이버 제어까지 다양한 것을 학습하고 직접 구현해볼 수 있는 좋은 기회였던 것 같습니다. 팀프로젝트를 통해 소통과 협업의 중요성을 크게 느낀 것 같습니다.

공하영 연구원

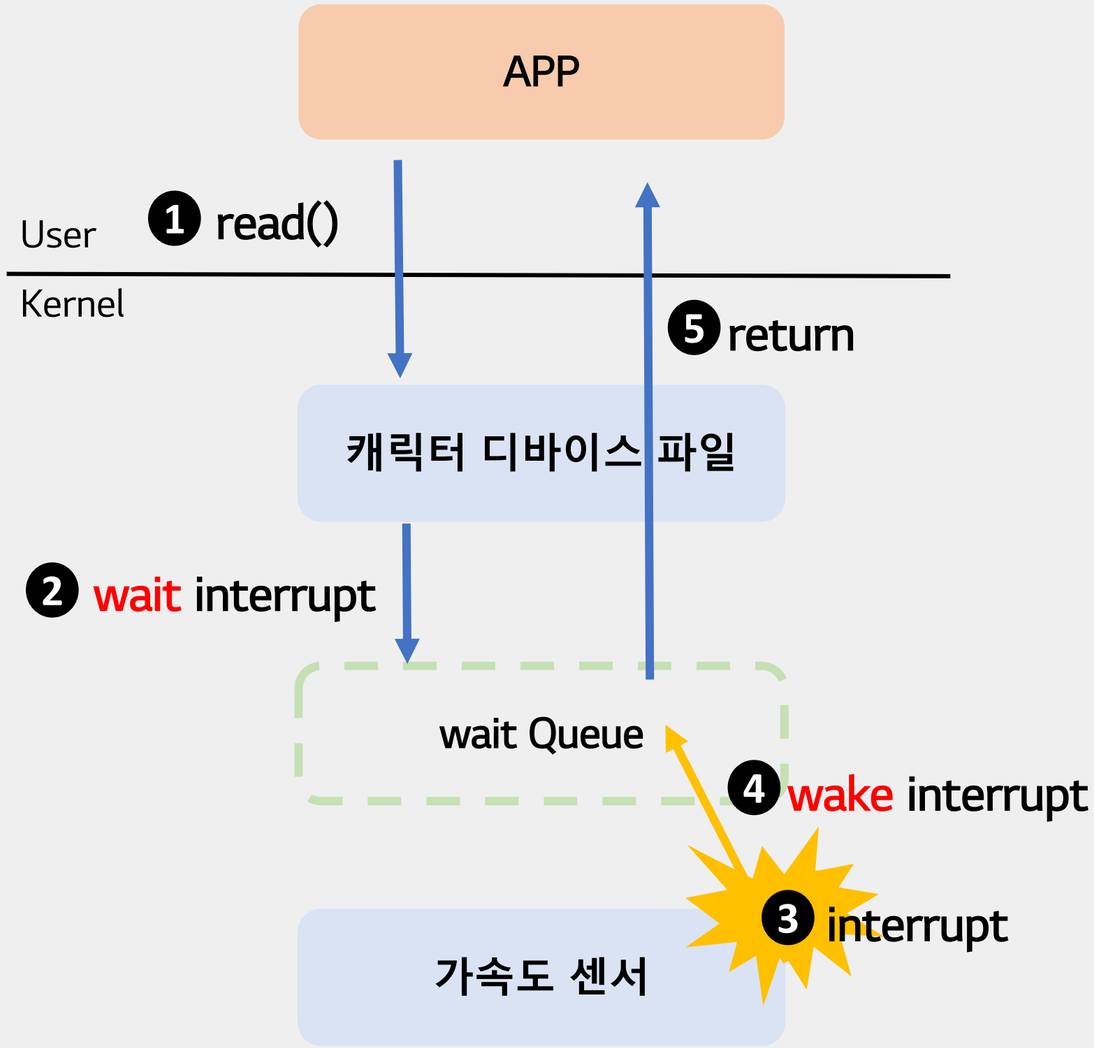
ALSA를 활용하여 오디오 데이터를 처리하고 직접 제어하는 방법을 이해할 수 있는 기회가 되었습니다. 2주간의 압축된 일정 속에서도 많은 것을 배우고 팀원들과 함께 성장할 수 있던 시간이었습니다.

감사합니다!

Q&A

6-1. 부록

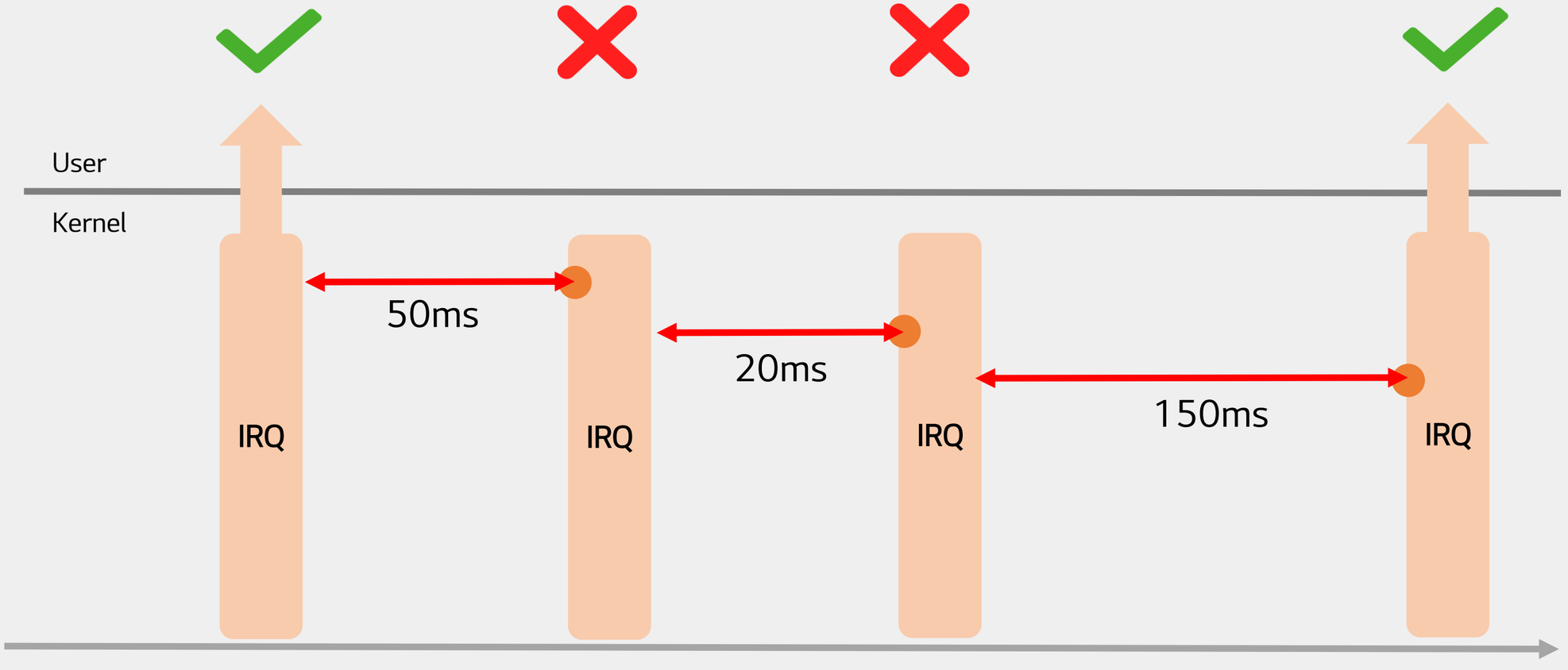
- 디바이스 드라이버 Interrupt 제어 방식



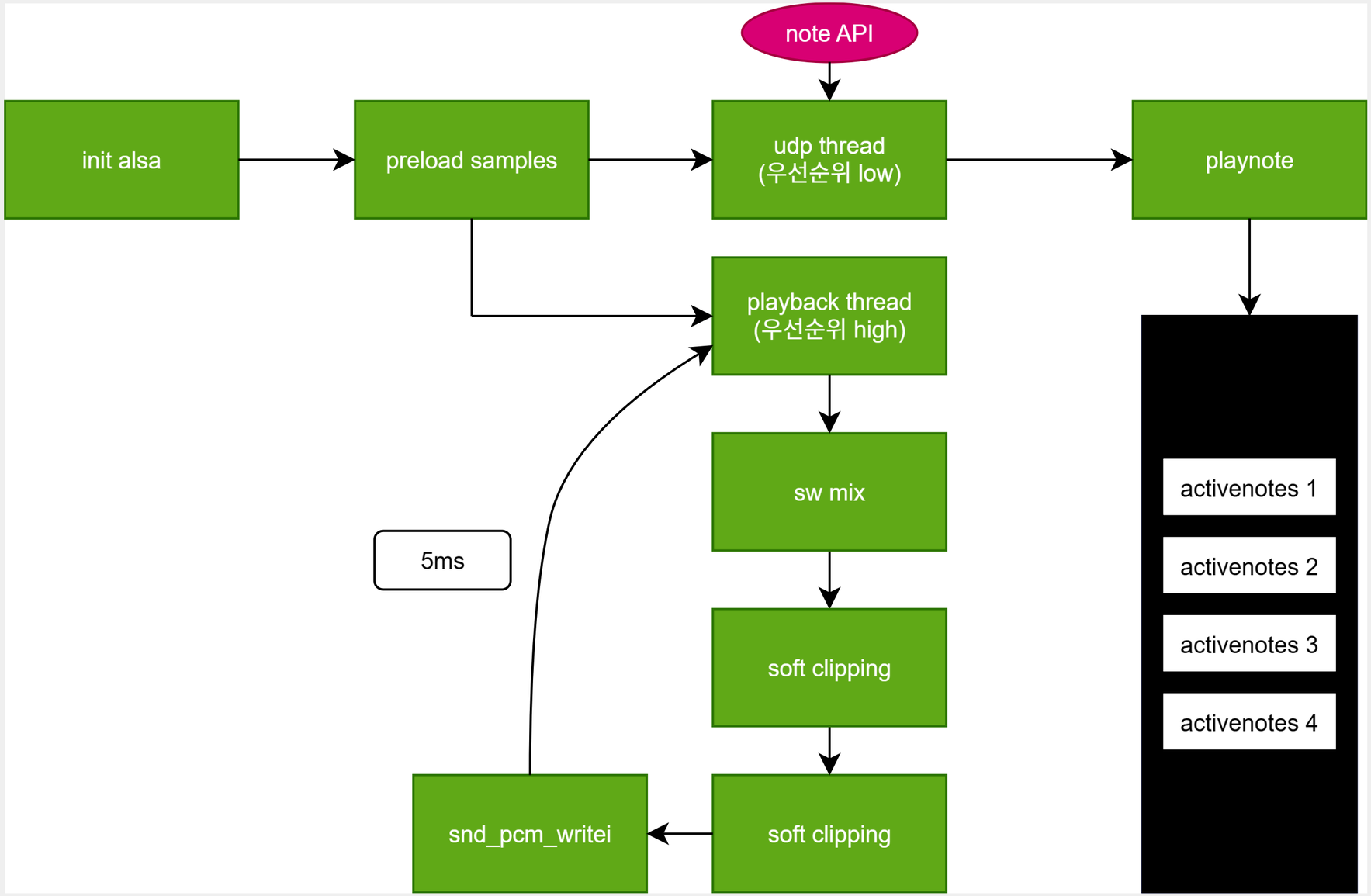

```
DECLARE_WAIT_QUEUE_HEAD(int_wq);

1 read()
static ssize_t dev_read( ) {

    2 wait interrupt
    wait_event_interruptible(int_wq, condition);   → 대기 상태

}                                                      ↓
                                                      진행

3 interrupt
static irqreturn_t lis302dl_interrupt() {

    4 wake interrupt
    wake_up_interruptible(&int_wq);
}
```

6-2. 부록

- **가속도 센서 충격 동작 잔진동 제어**
  - 잔진동 임계치 : 100ms
  - 이전 IRQ와의 시간 간격 계산 (임계치 이상인 경우 IRQ로 인식)
  - Jiffies 사용하여 시간 계산 간소화



6-3. 부록


note API
init alsa
preload samples
udp thread
(우선순위 low)
playnote
playback thread
(우선순위 high)
sw mix
5ms
soft clipping
snd_pcm_writei
soft clipping
activenotes 1
activenotes 2
activenotes 3
activenotes 4

6-4. 부록

ALSA util

aplay

Command Line 재생

실행(재생)시마다 ALSA PCM device open

File을 read하여 ALSA로 writei

재생 종료 후 ALSA close

Process로 실행

제한된 설정 옵션

VS

ALSA lib

pcm.h

Digital audio stream HW open 최초 1회 실행
+ HW 옵션 설정(rate, periodsize, buffersize, ...)

음원 재생 시에 writes

File preload to memory

File I/O latency 제거 (file I/O 대기 X)

다중 음원 재생 시 성능 저하 감소

File handler 관리 불필요