

아두이노 기반 MIDI Controller 개발 : 공명 제어 시스템

Development of An Arduino-Based MIDI Controller
: Resonance Control System

본 보고서를 2018년도 코너스톤디자인(창의설계입문) 과제의
최종보고서로 제출합니다.

〈 요약 문 〉

설계과제 목표	<ul style="list-style-type: none"> ➤ 아두이노를 활용한 MIDI 공명 제어 기술 개발 <ul style="list-style-type: none"> ○ 아두이노 기반 기술 구현 <ul style="list-style-type: none"> ▪ 아두이노 회로도 설계 및 구현 ➤ 외형 제작 <ul style="list-style-type: none"> ○ 3D 모델링 및 제작 		
내용	<ul style="list-style-type: none"> ➤ 아두이노를 기반 MIDI Protocol 공명 제어 기술 개발 <ul style="list-style-type: none"> ○ 하드웨어적인 관점과 소프트웨어적인 관점 ○ UML, Flow Chart 작성 ○ 공명 제어 기술 회로도, SW 설계 및 개발 <ul style="list-style-type: none"> ▪ 버튼형식의 디자인 ▪ 센서 부착 ○ 공명 제어 시스템 테스트 ➤ MIDI 장치 외관 디자인 ➤ 테스트 및 오류 수정 		
기대효과	<ul style="list-style-type: none"> ➤ 아두이노를 활용한 MIDI Controller 개발 방향 제시 ➤ 아두이노와 3D 프린팅 기술로 제작된 제품은 원가를 절약할 수 있음 ➤ MIDI Controller의 소비자 진입장벽이 낮아짐 		
Keywords	아두이노	MIDI Controller	Seaboard
	공명 제어 시스템	공명통	진동 센서

목 차

I. 서론	1
1. 설계 배경	1
2. 기술적 배경 및 최종 목표	2
II. 설계 수행 내용	5
1. 제안한 공명 제어 시스템 개발 방법	5
2. 공명 제어 시스템 설계	6
2.1 하드웨어 설계	7
2.2 소프트웨어 설계	8
3. 공명 제어 시스템 시뮬레이션	9
3.1 하드웨어 구현	9
3.2 소프트웨어 구현	10
4. 시뮬레이션 테스트	11
5. 외관 디자인 설계	12
6. 설계 구성요소 및 제한요소	14
III. 결론	16
1. 과제 결과 요약	15
2. 개선 방안 및 향후 과제 방향	15
3. 설계 목표의 중요도 및 달성도	16
IV. 참고문헌	18
[부록]	19

I. 서론

1. 설계 배경

“ 저가의 MIDI Controller는 공명 제어 기능이 없을까? ”

고가의 장비가 가지는 기능을 호환성과 확장성이 좋은
아두이노 기반 저가 기기로 구현

MIDI Controller는 MIDI(Musical Instrument Digital Interface) 프로토콜을 이용한 하드웨어 또는 MIDI 데이터 생성 및 전송하는 소프트웨어를 의미한다[1]. 현재 MIDI Controller는 공명 구현 기술이 적용된 제품이 부족하고 이마저도 인터페이스에서 오는 문제점이 있다. 또한, 일반적으로 MIDI Controller를 제어하는데 움직임의 제약이 있어 자유도가 낮은 실정이다. 이를 해결하기 위해 ROLI사는 모듈 형태로 확장이 가능하고 5D Touch 기술이 적용된 Seaboard를 개발하였다. 5D Touch 기술은 Strike, Glide, Slide, Press, Lift 총 5가지의 기술을 말한다[2].

본 설계에서는 Seaboard를 모티브로 Open Source 기반의 마이크로 컨트롤러인 아두이노를 활용하여보다 쉽게 접근할 수 있도록 하였다. 아두이노를 활용한 공명 제어 시스템을 개발 방안을 제시하고, 이를 MIDI 장치에 적용해보기 위한 시뮬레이션을 하였다.

압전센서와 진동 센서를 이용한 시스템 시뮬레이션으로 공명 제어 시스템의 가능성을 보여준다.

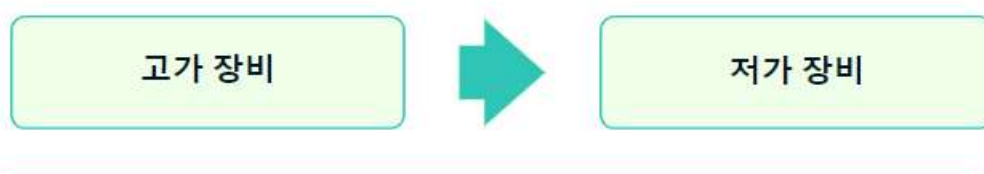


그림 1. 설계 배경

2. 기술적 배경 및 최종 목표

2.1. 기술적 배경

아두이노는 오픈소스 기반 마이크로 컨트롤러 보드로, C 기반의 프로그래밍 언어로 프로그래밍하여 다양한 분야에 활용할 수 있는 기기이다. 다른 마이크로 컨트롤러에 비해 저렴한 가격과 넓은 확장성, 비교적 접근성이 높은 IDE 등이 있고, 무엇보다 가장 큰 장점은 오픈소스 하드웨어라는 점이다[3].

아두이노 보드는 다양한 모델이 있는데 가장 대표적인 모델은 [그림 2]의 아두이노 우노로 14개의 디지털 포트와 6개의 아날로그 포트를 가지고 있다.

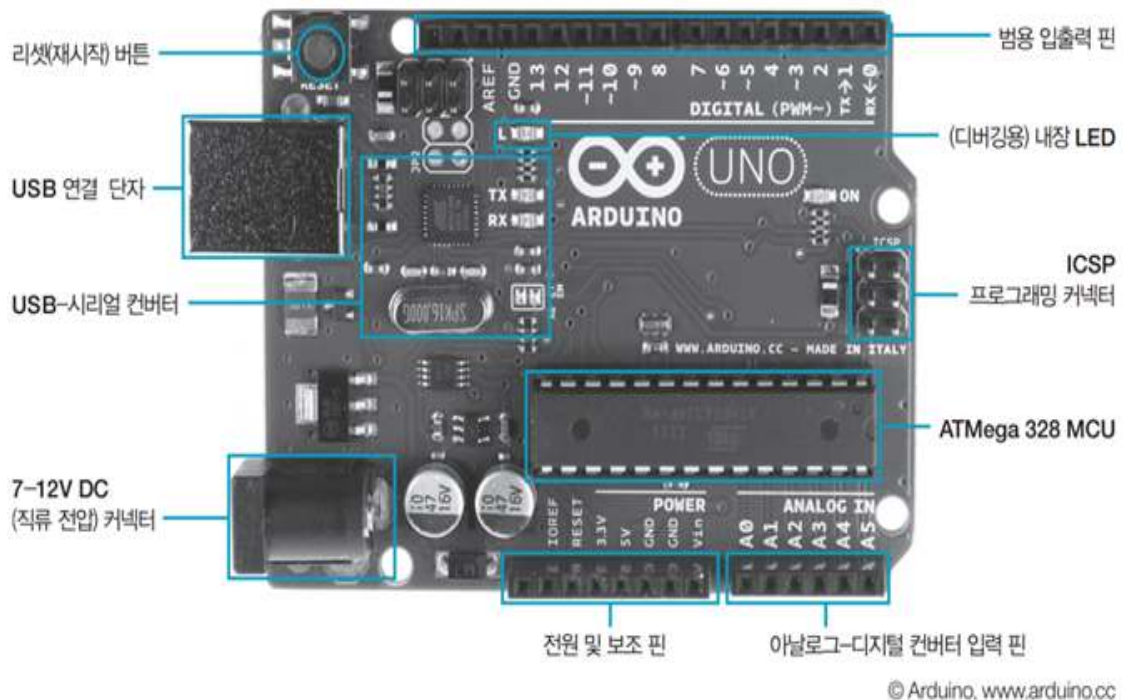


그림 2. 아두이노 구조

모든 아두이노 프로그램에서 필요한 함수가 2가지 있는데, 하나는 `setup()` 함수이고 다른 하나는 `loop()` 함수이다. `setup()` 함수는 프로그램 시작 시 단 한 번만 실행되어, 변수 설정, 핀 모드 등 초기설정에 관한 코드가 구현되어야 하며, `loop()` 함수는 `main` 함수라 보면 되는데 무한 반복하여 연산하는 함수이다.

2.2. MIDI Protocol

MIDI는 전자 악기 사이의 호환성과 정보 전달을 위해 각 신호를 규칙화한 것으로 1byte 단위로 구성되고, 최상위 비트에 따라서 상태 정보와 데이터 정보로 구분된다[4].

MIDI 상태 정보는 건반을 누르거나 눌렀다 떼 효과 등을 나타내는 채널 Message와 시작 및 종료 등 특정 채널이 아닌 MIDI System 전체를 control 하는 시스템 Message로 구분된다. MIDI 데이터 정보는 0부터 127까지 약 10옥타브로 구성된 Note Number와 Controller Number, Program Number, Pressure, Velocity 정보로 구성된다.

<i>MIDI commands</i>	
0x80	Note Off
0x90	Note On
0xA0	Aftertouch
0xB0	Continuous controller
0xC0	Patch change
0xD0	Channel Pressure
0xE0	Pitch bend
0xF0	(non-musical commands)

그림 3. 기본적인 MIDI 명령어

[그림 3]은 기본적인 MIDI 명령어이다[5]. C 기반 프로그래밍 언어에서 16진수는 숫자 앞에 '0x'를 붙여 표시한다. 예를 들어, Note Off 명령어인 0x80은 16진수 80인 것이고 10진수로 변환하면 128인 것이다.

MIDI 명령어에서 채널은 음색을 할당하는 단위가 된다. MIDI 채널 번호는 1에서 16까지이며, 이 값은 0에서 15(0~F)의 값으로 표시된다. 즉, 1개의 MIDI IN 포트를 가진 기기는 동시에 최대 16개의 음색을, 2개의 포트를 가진 기기는 최대 32개의 음색을 연주할 수 있다.

Note Off 명령어는 MIDI 전송을 중지시키는 명령어이며, 반대로 Note On 명령어는 MIDI 전송을 시작하라는 명령이다. 또한, Aftertouch 명령어는 건반을 누른 상태에서 건반을 누르는 압력에 따라 비브라토나 음색의 밝기 등을 조정하는 기능을 한다.

2.3. 설계 최종 목표

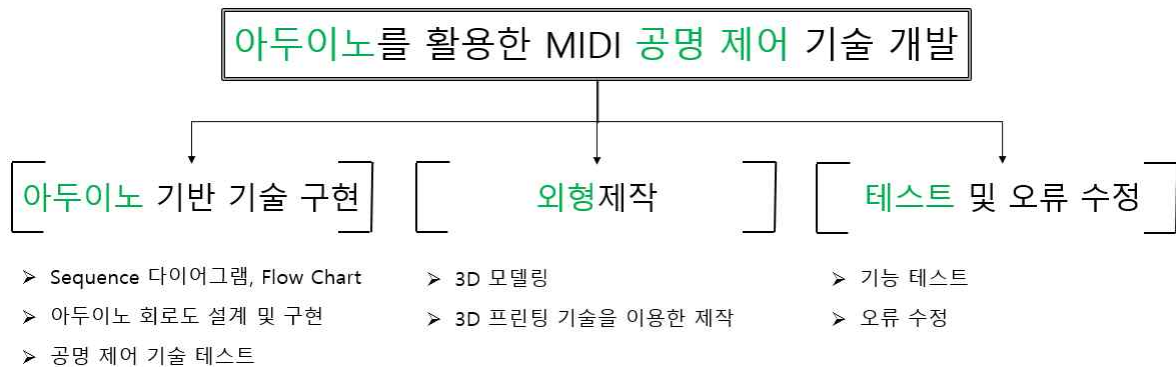


그림 4. 설계 목표

[그림 4]는 설계과정에서 목표를 설정한 것을 도식화한 것이다. 아두이노를 활용한 MIDI 공명 제어 기술 개발을 설계의 최종 목표로 설정하고 하위 목표로 아두이노 기반 기술 구현, 외형 제작, 테스트 및 오류 수정으로 설정하였다.

아두이노 기반 기술 구현을 위해 아두이노 회로도 설계를 비롯하여 Sequence Diagram, Flow Chart와 같은 UML과 알고리즘을 도식화하였고, 회로도에 기반한 시뮬레이션 테스트를 진행하였다.

외형 제작을 위하여 3D 프린팅 기술을 이용한 제작을 목표로 아두이노 회로도에 기반하여 3D 모델링을 진행하였다.

테스트 및 오류 수정 부분에서는 시뮬레이션 테스트 결과를 바탕으로 문제점을 도출하고, 해결 방안을 제시하는 것을 목표로 진행하였다.

본 설계의 최종 목표는 다음과 같다.

압전센서와 진동 센서를 이용한 공명 제어 시스템 개발

II. 설계 수행 내용

1. 제안한 공명 제어 시스템 개발 방법

본 설계에서 제안하는 관점에 따른 공명 제어 방법은 다음과 같다.

[표 1] 공명 제어 관점에 따른 방안

HW	➤ 공명통과 진동 센서를 활용하여 소리의 흐름 제어
SW	➤ MIDI 라이브러리 활용(Aftertouch)

모든 물체는 고유의 파동이 있다. 파동은 같은 주파수의 파동과 만날 때 공명을 일으킨다. 공명이란 고유 진동수와 같은 진동을 외부로부터 받았을 때, 큰 진폭으로 물리는 현상을 말한다. 악기 연주를 하며 멀리 떨어진 유리컵이 깨지는 경우는, 유리컵이 가진 고유 파동수에 해당하는 커다란 음을 악기가 내면서 공명을 일으켰기 때문이다. 생명체를 비롯해 모든 물체에는 각기 고유의 파동수가 있고 모든 파동을 같은 주파수의 파동과 만날 때 공명을 일으킨다[6].

특히, 현악기에서 공명 현상을 볼 수 있다. 현의 떨림에 따라 파동이 생성되는데 떨림이 반복되면서 같은 주파수의 파동과 만나는 횟수가 증가하게 된다. 이때, 공명이 발생하게 되며 특유의 현악기의 감성을 만들어낸다. 현악기 특유의 공명을 MIDI Controller로 구현하여 제어할 수 있다면 사용자에게 좋은 UX(User Experience)를 줄 수 있을 것이다.

공명을 구현하여 제어하는 방안은 하드웨어적인 방안과 소프트웨어적인 방안으로 크게 두 가지로 구분할 수 있다.

하드웨어적인 방안은 공명통이라는 소리의 흐름을 제어할 수 있는 하드웨어를 이용하여 진동으로 공명을 구현하는 방안이다. 소프트웨어적인 방안은 MIDI 라이브러리에 정의된 'aftertouch' 기능을 이용하여 구현하는 방안이다[7].

본 설계에서는 하드웨어적인 방안에 대해 진행할 것이다. 하드웨어 부분에서 구현 및 제어를 할 수 있고, 다른 기능까지 확장이 편리하다면 아날로그 감성적인 부분에서 경쟁력을 가질 수 있을 것으로 생각된다.

2. 공명 제어 시스템 설계

본 설계에서는 공명통을 이용한 공명 제어 실험을 설계하였다. 설계한 실험을 Adobe사에서 Online으로 제공하는 Tinker CAD Circuits 시뮬레이션을 활용하여 공명 제어 시스템을 설계 및 테스트를 진행하였다. 시뮬레이션 상에서는 이론상 구현 가능한 부분은 전부 구현되었기 때문에 테스트 결과를 확인하기 위해 아두이노 우노 R3를 이용하여 구현하였다. 시뮬레이션 환경에서는 부품과 작동에서 제한적인 부분이 있어 전체적인 테스트 환경을 간략화하여 진행하였다.

[표 2] 공명 제어 시스템 설계의 목표와 구현 사항

목표	➤ 공명 제어
구현	➤ 공명 제어 + MIDI 연주

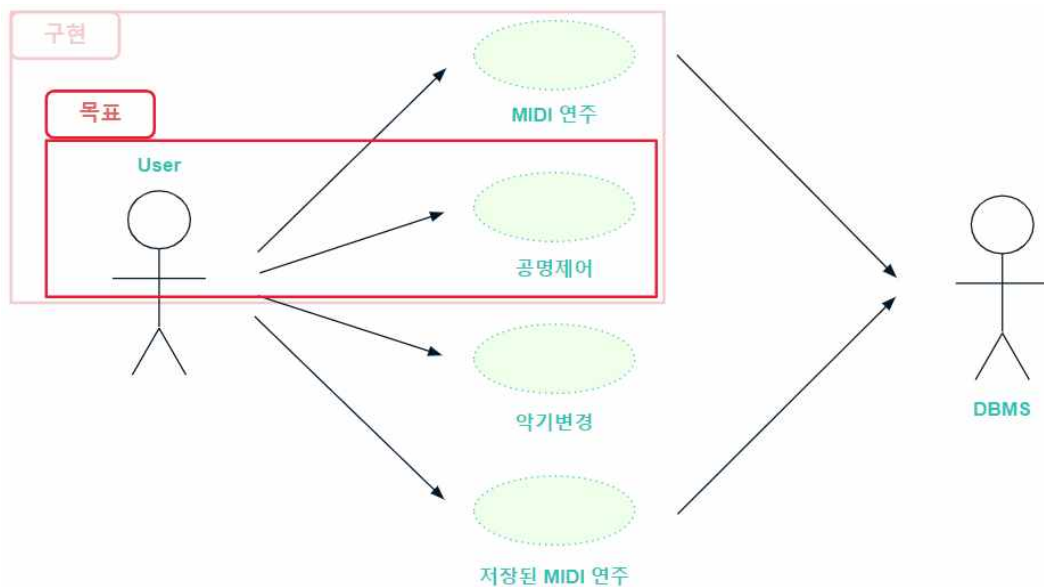


그림 5. 일반적인 MIDI Control System Use-Case Diagram

[그림 5]는 일반적인 MIDI Control System의 Use-Case Diagram이다[8]. MIDI 연주, 공명 제어, 악기변경, 저장된 MIDI 연주 총 4개의 Use Case로 구성되어 있고, User가 MIDI 연주, 저장된 MIDI 연주라는 행위를 하였을 시, 데이터베이스에 저장되는 시스템이다.

하지만 본 설계에서는 고가 장비의 기능 중 하나인 공명 제어를 호환성이 높은 아두이노를 이용하여 저가 장비에 탑재하는 것이 목표이다. MIDI 연주보다 공명 제어에 초점을 맞추어 설계하였다.

2.1. 하드웨어 설계

2.1.1. 회로도 설계

아두이노 R3에 호환 보드인 MP3 플레이어 쉴드를 적층하고, 압전센서를 이용하여 연주하는 압력의 값을 받고 진동 센서를 이용하여 진동을 주어 소리의 공명을 구현하는 것으로 설계를 하였다.

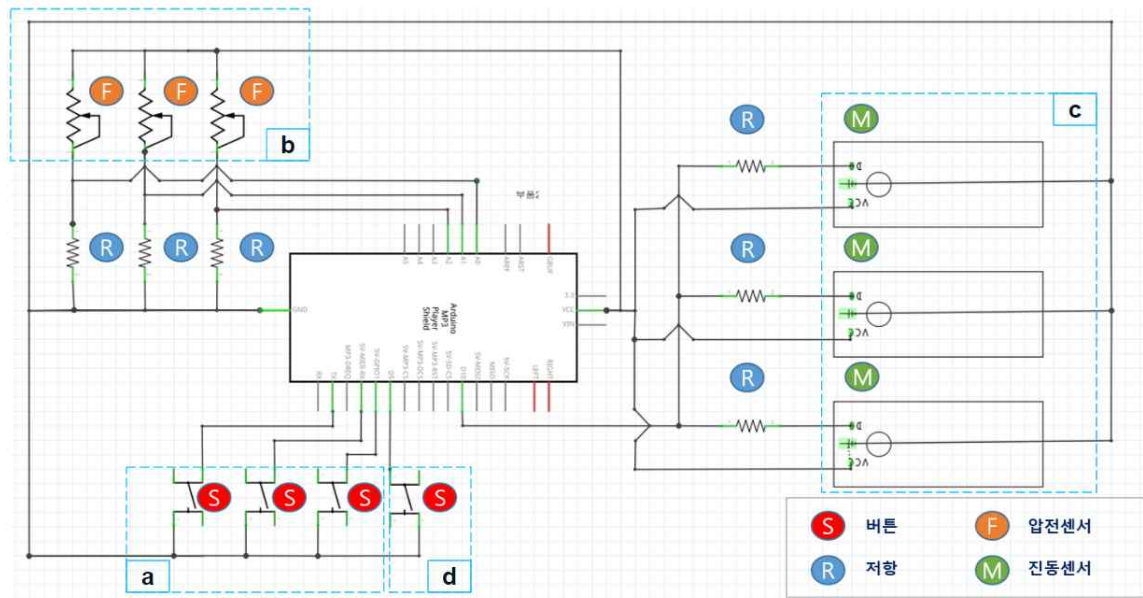


그림 6. 전체 회로도

[그림 6]은 실험 구상 시 전체 회로도이다. (a)는 키보드 버튼이며 음을 구현하는 역할을 하고, (d)는 공명 버튼이다. (b)는 압전센서이며 키보드 압력을 인식하여 보드에 입력된다. (c)는 진동센서로 보드에서 아날로그값이 PWM으로 변환된 값을 입력받아 진동을 출력한다. 진동센서가 공명통 안에서 진동을 출력 후 공명이 구현되는 방식이다.

저항은 버튼을 제외한 모든 부품에 연결된다. 일반적으로, Pull Up 저항(잔류 전류 방지)이 버튼과 연결되어 있어야 한다. 하지만 본 실험에서는 보드의 내장 저항을 이용하기 위하여 연결하지 않았다.

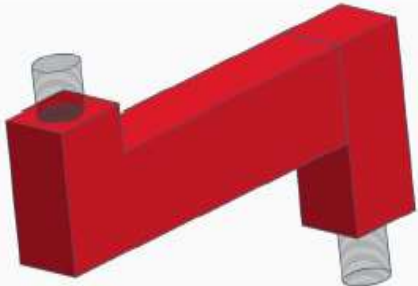
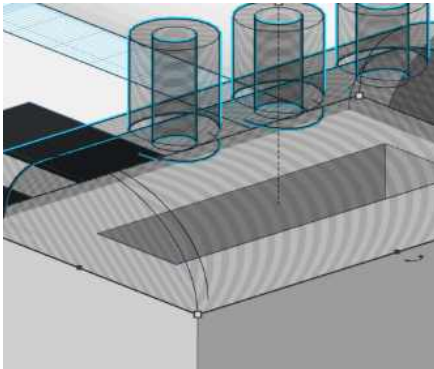
[그림 6]의 전체 회로도를 통해 다음과 같은 정보를 얻을 수 있다.

- D1, D3, D4 핀에 음 버튼이 연결됨
- D4 핀에 공명 버튼이 연결됨
- A0, A1, A2 핀에 압전센서가 연결됨
- D10 핀에 진동 센서가 연결됨

2.1.2. 공명통 설계

본 설계에서는 2단계로 공명이 이루어지도록 설계하였다. 1차 공명통을 통해 공명을 구현되도록 하였고, 2차 공명통을 통해 아날로그 감성의 음색 효과를 내도록 설계하였다. 파형이 같은 소리가 겹쳐질 시 보강 간섭, 즉 공명이 발생한다. 진동 센서를 통해 발생한 진동으로 스피커에서 나오는 원 파장의 변화를 형성하는데 3개의 진동 센서가 같은 세기로 진동하므로 보강 간섭을 형성하는 시스템이다.

[표 3] 공명통 요구사항과 설계 내용

CAD 설계	요구사항
	<ul style="list-style-type: none"> ➢ 외관에 진동 센서 설치 ➢ 외부는 진동 센서를 부착하도록 육면체로 구현 ➢ 내부는 전파가 용이하도록 원통형으로 구현 ➢ 스피커에서 나오는 소리가 공명통을 통해 진동하며 전파되어야 함
	<ul style="list-style-type: none"> ➢ 내부에 1차 공명통 설치 ➢ 원통형으로 공명통 구현

[표 3]은 공명통 요구사항과 설계 내용을 요약하여 정리한 것이다. 외관과 내부를 다른 형태로 설계하였다. 1차 공명통 외관에 진동 센서 3개를 부착할 예정이며, 원활한 부착을 위하여 외관은 육면체로 설계하였다. 내부는 파장의 전파가 용이하고, 보강 간섭이 이루어지도록 반지름이 일정한 원통형으로 설계하였다. 한쪽 구멍에는 스피커를 연결할 것이며, 다른 한쪽 구멍에는 2차 공명통으로 파장이 전달되도록 하였다. 스피커에서 나오는 소리가 공명통을 통해 진동하여 전파되도록 스피커와 1차 공명통의 간격을 최소화하는 방안으로 설계하였다. 2차 공명통은 3개의 파이프 오르간 형태를 주어 공명이 형성된 소리의 전파를 용이하도록 설계하였다.

2.1. 소프트웨어 설계

공명 제어 기능을 구현하는 UML Diagram은 [그림 12]에 명시된 바와 같으며 시스템 구동 후(공명 버튼 누른 후) 공명 제어 시스템에 대해 명시되어 있다. 아날로그 입출력을 표현하기 위해 NOTE 표기법을 이용하였으며, 입출력 제약이나 부가적인 정보를 기술하기 위해 OCL(Object Constraint Language)을 이용하여 추가사항을 명시하였다[9].

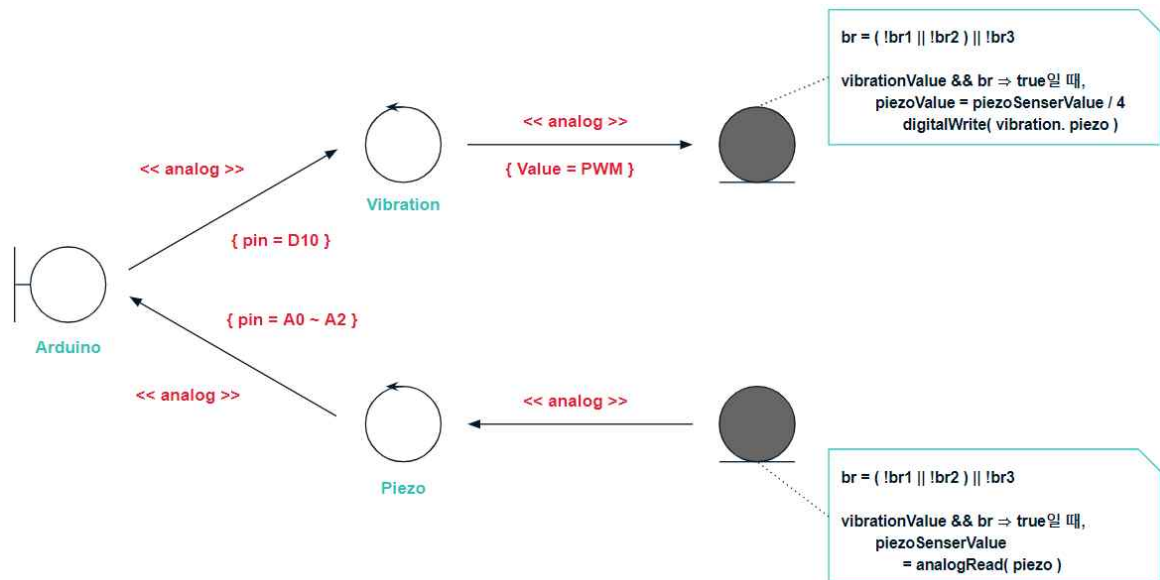


그림 8. 공명 제어 시스템에 대한 확장된 UML Diagram

[그림 8] Diagram을 통해 다음과 같은 정보를 얻을 수 있다.

- 보드에 압전센서가 입력장치로 연결되고, 진동 센서가 출력장치로 연결됨
- 압전센서는 A0 ~ A2에 연결되어 있고, 아날로그를 입력(analogRead)함
- 진동센서는 D10에 연결되어 있고, 아날로그를 PWM으로 출력(digitalWrite)함
- pinMode(piezo, INPUT), pinMode(10, OUTPUT), analogRead(piezo), digitalWrite(vibration, piezo) 등 주요 함수와 그 파라미터 정보를 제공함

UML Diagram을 기초로 서비스를 구현하기 위한 loop()함수 알고리즘은 [그림 9]에 명시된 바와 같다. 어떤 버튼을 눌렀는지에 대한 질의를 통해서 동작이 변한다.

공명 버튼을 눌렀을 때, ON/OFF 값이 저장되는 vibrationValue 값이 참 혹은 거짓에 따라 vibrationValue 값이 변한다. 참일 때는 공명 버튼이 ON 상태이고, 거짓일 경우에는 OFF 상태이므로 참일 때는 false로 변경해주고, 거짓일 때는 true로 변경해준다.

음 버튼을 눌렀을 때 마찬가지로 vibrationValue 값이 참인지 거짓인지 질의를 한다.

vibrationValue 값이 거짓일 경우 별다른 과정 없이 음이 출력되지만, vibrationValue 값이 참일 경우 공명을 구현하기 위한 과정이 추가된다.

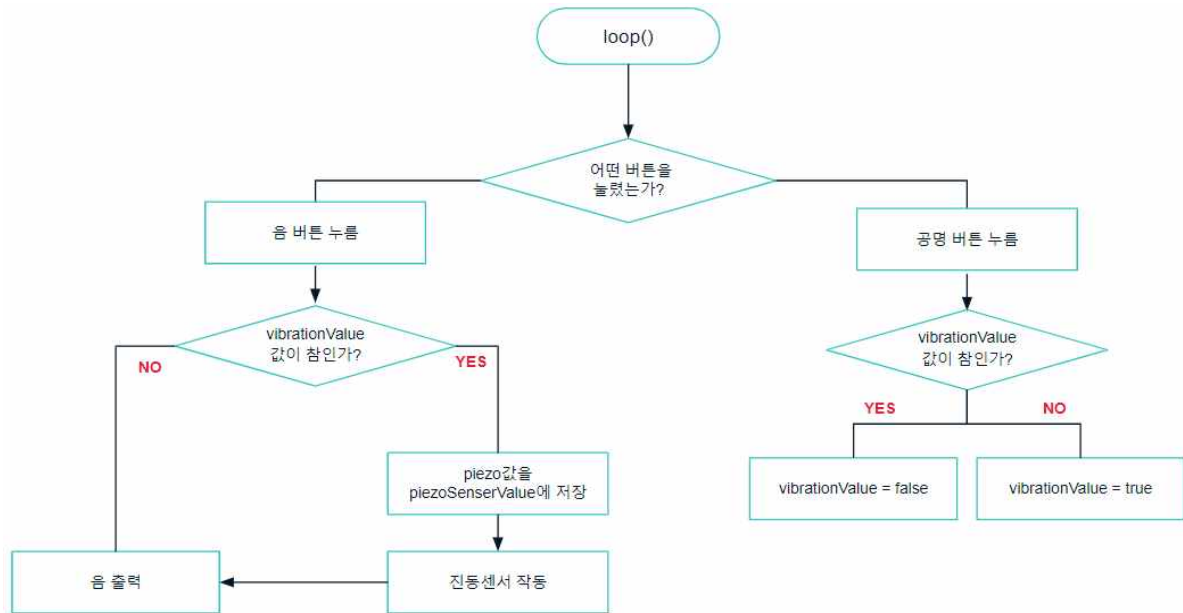


그림 9. 공명 제어 시스템이 탑재된 MIDI Controller 알고리즘

3. 공명 제어 시스템 시뮬레이션

3.1. 하드웨어 테스트 장치 구현

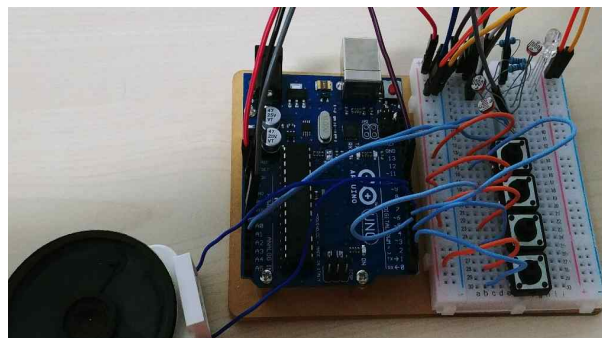


그림 10. 아두이노 테스트 장치 구현

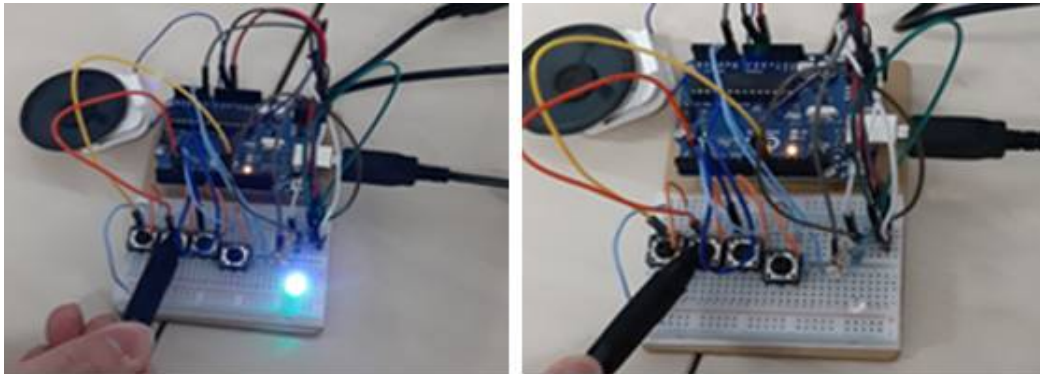
[그림 10]은 시뮬레이션에 사용한 장치이다. 일부 부품을 대체하여 구현하였다. 압전센서를 대체하여 아날로그값을 입력시킬 수 있는 포토 레지스터를 이용하였고, 진동 센서를 대체하여 PWM 값을 출력할 수 있는 RGB LED 모듈을 이용하였다. R, G, B 값을 하나의 모듈이 나누어 출력 가능하므로 한 모듈만을 사용하였다.

장치 구현 전에 한 핀에 여러 부품이 연결되어도 같은 값이 입출력되는 것을 시뮬레이션에서 확인하였다. 실제 환경에서 아날로그값이 오차가 5에 수렴하여 오차 범위 내이므로 RGB LED 모듈에서 R, G, B 입력 핀은 D10에 연결하여 구현하였다. 또한, 스피커 출

력을 위해 스피커 모듈을 D7에 연결하였다.

버튼 중 3개는 MIDI 생성 및 출력을 위한 버튼이며, 1개는 공명 제어 버튼이다.

3.2. 시뮬레이션 결과



(a)

(b)

그림 11. 공명 버튼 On(a) / Off(b)에 따른 시뮬레이션 테스트 결과

[그림 11]은 공명 버튼 ON/OFF에 따른 결과이다. (a)일 때는 LED 불이 들어오고, (b)일 경우에는 LED 불이 들어오지 않는 것을 볼 수 있다. 본 실험의 설계와 같이 구현을 하였다면 음 버튼 하나 누를 때 R, G, B 중 하나씩 값이 들어가지만, 테스트에서는 압전센서를 대체하여 포토 레지스터를 사용하였으므로 빛을 완전히 차단하지 못하여 [그림 12]와 같이 RGB 값이 입출력되는 것을 볼 수 있다. 시리얼 통신의 결과를 보면 Row 센서값이 PWM으로 변환되는 과정에서 오류가 발견되지 않고 출력이 정상적으로 되는 것을 볼 수 있다. 또한, 공명 버튼이 ON일 때는 1이 출력되고 OFF일 때는 0이 출력되어 정상적으로 출력되는 것을 볼 수 있다.

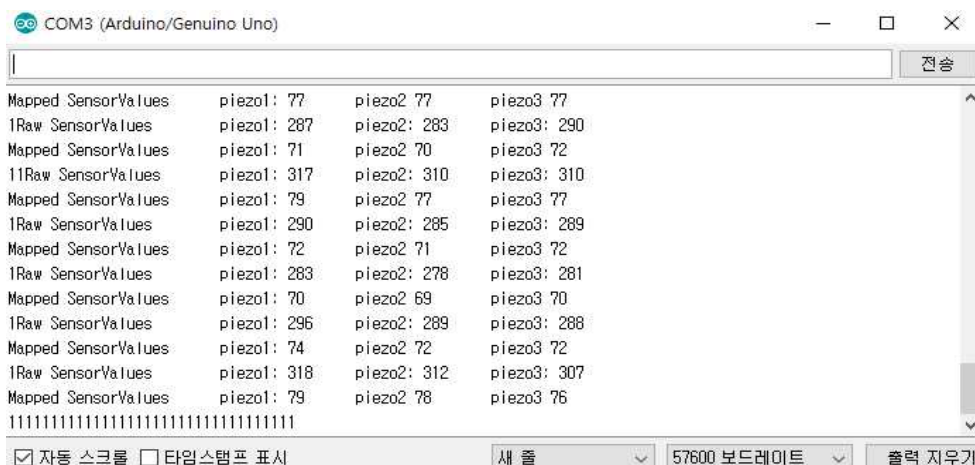


그림 12. 공명 버튼 ON일 때의 포토 레지스터값의 변화 결과 화면

4. 공명 제어 시스템 구현

4.1. 하드웨어 구현

4.1.1. 아두이노 장치 구현

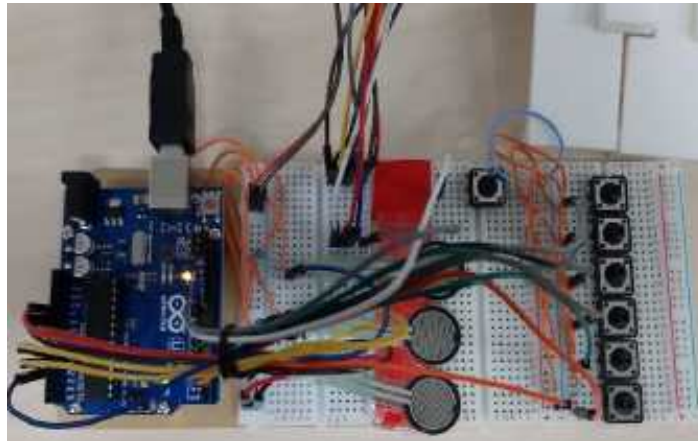


그림 13. 아두이노 장치 구현

[그림 13]은 MIDI Controller 아두이노 장치를 구현한 것이다. 압전센서 3개, 진동 센서 3개, 공명 버튼, 음 버튼 6개를 사용하였는데, 테스트 장치에서 대체된 부품을 원래 부품으로 교체하고, 음 버튼을 6개로 하여 장치를 구현하였다.

4.1.2. 1차 공명통 구현

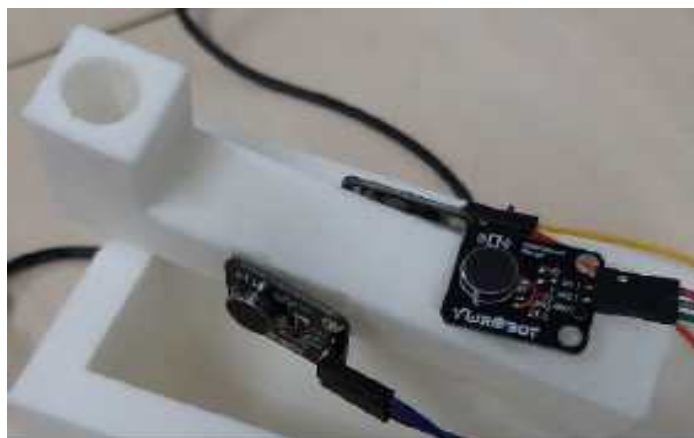


그림 14. 1차 공명통 구현

[그림 14]는 1차 공명통을 구현한 것이다. 3D 프린터를 이용하여 구현하였고, 필라멘트는 PLA를 사용하였으며, 채우기 밀도는 25%로 지정하였다. 필라멘트와 채우기 밀도는 가장 기본적인 설정으로 출력하였다. 진동을 고루 주기 위하여 외벽에 진동 센서 위치를 다르게 부착하였다.

4.2. 소프트웨어 구현

4.2.1. MIDI Message 전송 함수

```
//Send a MIDI note-on message. Like pressing a piano key
//channel ranges from 0-15
void noteOn(byte channel, byte note, byte attack_velocity) {
    talkMIDI( 0x90, note, attack_velocity);
}

//Send a MIDI note-off message. Like releasing a piano key
void noteOff(byte channel, byte note, byte release_velocity) {
    talkMIDI( 0x80, note, release_velocity);
}

//Plays a MIDI note. Doesn't check to see that cmd is greater than 127, or that data values are less than 127
void talkMIDI(int cmd, int note, int velocity) {

    digitalWrite(ledPin, HIGH);

    Serial.write(cmd);
    Serial.write(note);
    Serial.write(velocity);

    digitalWrite(ledPin, LOW);
}
```

그림 15. MIDI Message에 관한 코드

[그림 15]는 MIDI Message를 전송 및 재생하는 코드이다[10]. noteOn 함수와 noteOff 함수, talkMIDI 함수를 정의하였다. noteOn 함수는 MIDI 생성 및 전달하는 함수이고, noteOff 함수는 MIDI 전송을 중지하는 함수이다. 두 함수는 매개변수 값을 순서대로 byte channel, byte note, byte velocity 값을 받고, channel은 MIDI 음색 또는 note On Message, note Off Message 값이 들어가는 변수이며, note는 음계 값 변수, velocity는 Message가 전달되는 속도에 관한 변수이다.

talkMIDI 함수는 MIDI Message를 전송하는 함수로 byte cmd, byte data1, byte data2를 매개변수로 값을 받는다. talkMIDI 함수가 실행되면 트래픽 LED에 HIGH 값이 입력된다. MIDI Message 전송을 알리기 위함이다. 그 후 SoftwareSerial로 생성한 mySerial에 cmd와 data1값을 송신하는데 아스키코드가 전송된다. 보통 write() 함수는 시리얼 모니터에 표시할 때는 사용하지 않는데 통신을 할 때 문자 외에 직접 데이터값을 보내야 할 때 사용된다. 이때는 MIDI 데이터값을 직접 보내기 위해 사용하였다.

일부 명령에는 하나의 데이터 byte만 있을 때가 있다. 그 경우를 대비하기 위하여 data2 값을 입력받아 데이터 통신을 하는 코드를 if 문 안에 정의하였다.

마지막에는 MIDI Message 전송이 종료되었다는 것을 알려주기 위해 트래픽 확인용 LED에 LOW 값을 입력하는 것으로 talkMIDI 함수가 종료되도록 구현하였다.

5. 공명 제어 시스템 결과

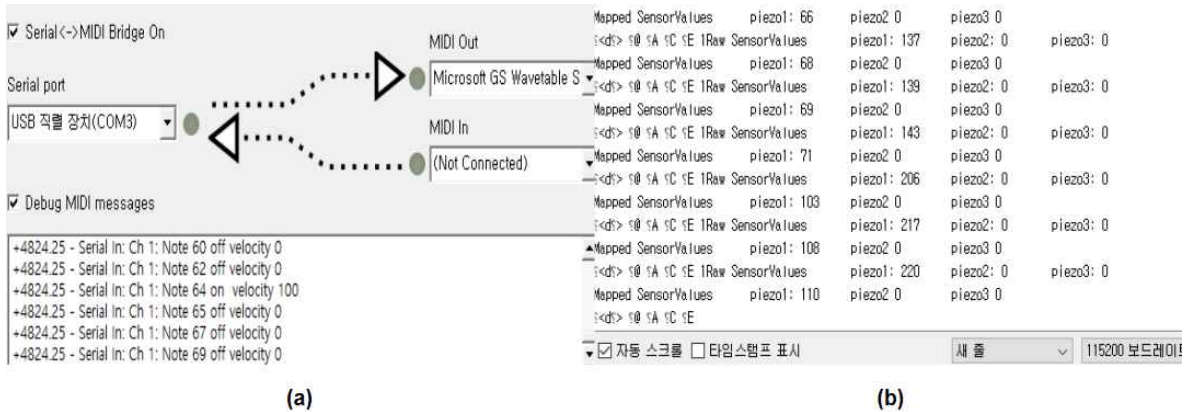


그림 16. MIDI – 시리얼 Bridge Program(a) 과 아두이노 시리얼 모니터(b) 결과 화면

[그림 16]은 공명 제어 시스템 시리얼 통신 결과 화면이다. (a)는 컴퓨터를 통해 시리얼 입력 값을 받아 MIDI로 출력해주는 프로그램 결과 화면이며, (b)는 아두이노 IDE에 내장되어있는 시리얼 모니터 결과 화면이다. (a)에서 Note 번호는 음을 뜻하며, on/off와 velocity 0/100은 MIDI 음이 출력되고 있는지 없는지를 나타내는 것이다. 따라서 velocity 0은 noteOn이고, 100은 noteOff이다. (b)는 시뮬레이션일 때와 동일한 의미를 가지는데, 압전센서의 원시 데이터가 먼저 출력되고, PWM으로 변환된 값이 출력된다. 시리얼 모니터 오른쪽 하단에 보면 11500 보드레이트로 되어있는데, 이것은 본 설계에서 사용한 MIDI – 시리얼 Bridge Program은 시리얼 통신을 11500 보드레이트일 때만 통신을 하므로 11500으로 설정하였다. 각 프로그램마다 통신 속도가 다르므로 프로그램마다 변경해 주어야 한다.

MIDI 명령어를 활용하여 피아노 음을 구현하였고, 공명 버튼 ON일 때 압전센서 값에 따라 진동 센서 입력 값이 변화하는 것을 확인할 수 있었다. 하지만 스피커의 부피로 인해 스피커와 공명통 연결이 불가하여 MIDI 명령어를 통해 공명 현상 확인이 불가하였다. 그렇지만 tone()함수를 이용하여 스피커 모듈에서 출력된 음은 1차 공명통을 통하여 진동을 주었을 때 미세하게나마 공명효과를 구현할 수 있었다. 이것으로 공명통에 의한 공명 현상의 가능성을 확인할 수 있었다.

6. 아두이노 외관 디자인 설계 및 구현

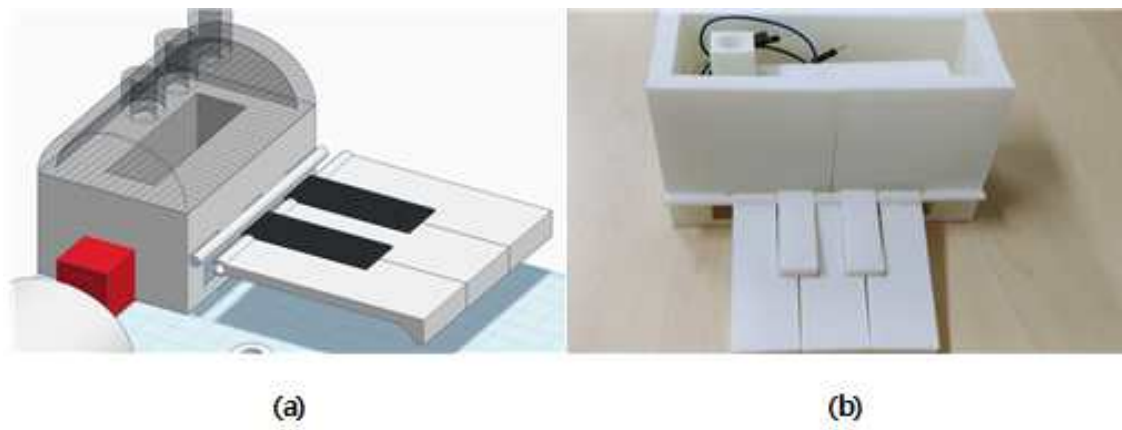


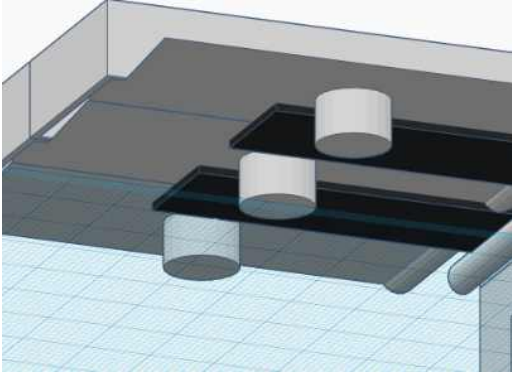
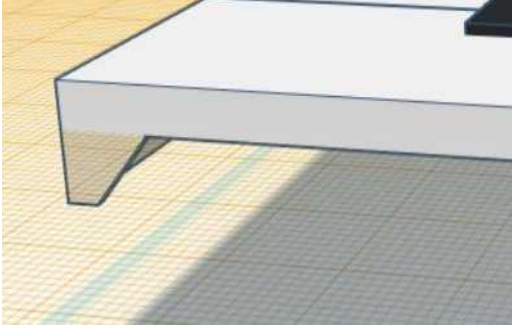
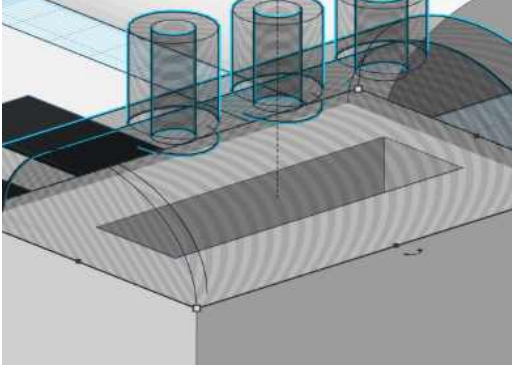
그림 17. 외관 디자인 설계(a) 및 구현(b)

[그림 17]은 MIDI Controller 외관 디자인 설계 및 구현한 것이다. Adobe 사에서 Online으로 제공하는 Tinker CAD를 이용하여 디자인하였다. [표 4]와 [표 5]는 디자인 요구사항과 설계 내용을 나열한 표이다.

[표 4] 외관 디자인 요구사항과 설계 내용

CAD 설계	요구사항
	<ul style="list-style-type: none">➤ 아두이노가 위치할 공간 필요➤ 아두이노와 건반을 담당할 버튼과 압전센서와 연결을 위한 공간 필요
	<ul style="list-style-type: none">➤ 건반이 상하로 움직여야 함➤ 흑건은 고정시키고 백건이 움직이도록 함

[표 5] 외관 디자인 요구사항과 설계 내용 - 기능적인 부분

CAD 설계	요구사항
	<ul style="list-style-type: none"> ➤ 건반을 누를 때 버튼이 누르게 설계
	<ul style="list-style-type: none"> ➤ 건반을 눌렀을 때 압전센서에 접촉하도록 설계
	<ul style="list-style-type: none"> ➤ 스피커에서 나오는 소리가 공명통을 통해 진동하며 전파되어야 함 ➤ 내부에 진동 센서와 스피커 설치 ➤ 원통형으로 공명통 구현

[표 4]는 아두이노가 위치하는 공간과 전선 연결, 기본적인 건반에 관한 내용에 관한 것이고, [표 5]는 MIDI 버튼을 누르고 소리를 내며 공명 제어할 수 있도록 기능적인 부분에 관한 내용이다.

7. 설계 구성요소 및 제한요소

[표 6] 설계 구성요소

설계 구성요소	내용
합성	<ul style="list-style-type: none"> ➤ 아두이노 + MIDI Controller → 아두이노로 구현한 저가 장비
분석	<ul style="list-style-type: none"> ➤ 하드웨어적 관점 → 공명통, 진동센서 이용 ➤ 소프트웨어적 관점 → aftertouch 기능 이용
제작	<ul style="list-style-type: none"> ➤ 압전센서 -> 스피커 -> 공명통 & 진동센서 → 공명 제어 ○ 시뮬레이션 : 포토레지스터 -> RGB LED
시험	<ul style="list-style-type: none"> ➤ 포토레지스터 값이 한 핀에서 오차범위의 RGB 값 변화 확인됨 ➤ 시리얼 보드레이트를 코드의 bps 이상으로 설정 시 시각화됨
평가	<ul style="list-style-type: none"> ➤ 대체 부품의 테스트 결과에서 음 구현에서 정확도가 떨어짐 ➤ 하드웨어적인 관점과 소프트웨어적인 관점 융합 필요
기타	<ul style="list-style-type: none"> ➤ 3D 프린터를 이용한 외관 출력 필요함

[표 6], [표 7]은 설계 구성요소 및 제한요소를 나타낸 것이다. [표 6]은 합성, 분석, 제작, 시험, 평가로 구성요소를 구분하여 분석 및 정리하였고, [표 7]은 원가, 안정성, 신뢰성, 미학, 내구성 요소로 구분하여 분석 및 정리하였다.

[표 7] 설계 제한요소

설계 제한요소	내용
원가	➤ 고가 장비는 50만 원인 것에 비해 10만 원대의 저가 장비임
안정성	➤ 하드웨어 관점에서 아두이노 기반 시스템 설계
신뢰성	➤ MP3 플레이어 쉴드 적층 시 정확도 높은 구현
미학	➤ 피아노 건반 디자인 유지 ➤ 공명통, 아두이노 적층 상자, 움직일 수 있게 설계한 디자인, 버튼을 누를 수 있도록 설계
내구성	➤ 3D 프린팅 출력 시 필라멘트 내구성과 설정에 따라 결정됨
기타	➤ 고가 장비에만 MIDI Controller 공명 기술이 탑재된다는 고정관념

III. 결론

1. 설계 결과 요약

본 설계에서는 하드웨어적 관점에서 아두이노를 활용한 MIDI Controller의 공명 제어 시스템 개발 방안을 제시하였다. 공명통에 부착된 진동 센서와 파이프 오르간 형태의 공명통을 이용하여 건반을 누를 때 압력변화로 공명을 형성하는 시뮬레이션을 하였다.

그 결과 한 버튼에서만 음이 구현되었다. 시뮬레이션 환경은 tone()의 특정 주파수가 소리를 형성하는지 확인하였다. MIDI Message로 MIDI를 형성하고 안정적인 스피커 출력을 MP3 플레이어 쉴드를 사용하면 해결될 것으로 추측된다. 본 시뮬레이션은 포토 레지스터의 빛을 완전히 차단하지 않았으며, 공명통 구현하지 않은 상태에서 실행하였다.

그러므로 하드웨어적 관점에서 공명 제어 시스템을 개발하는 것은 오히려 경쟁력이 떨어질 것으로 생각된다. 소프트웨어적 관점과 하드웨어적인 관점을 융합하여 새로운 시스템 프로세스의 필요성이 요구된다고 생각한다.

2. 개선 방안 및 향후 과제 방향

본 설계는 하드웨어적인 관점에서 공명 제어 시스템을 개발하려고 하였다. 하지만 실험 시뮬레이션 결과 문제점이 반복해서 발생하는 패턴이 나타났다. 또한, 확장성과 자유도가 낮은 문제점을 해결하기 위해서는 하드웨어뿐만 아니라 소프트웨어적 관점에서 새로운 프로세스의 필요성이 요구된다.

공명통을 이용한 하드웨어적 방안을 기반으로 MIDI 라이브러리 중 aftertouch Message를 이용한 소프트웨어적 개발 방안을 제안하며, 현재 MIDI 라이브러리는 진입장벽이 높아 사용자들의 접근성을 높일 수 API의 개발을 제안하고자 한다.

향후 개선 방안에 기초하여 하드웨어적 관점과 소프트웨어적 관점이 어울려진 시스템을 개발할 계획이다.

3. 설계 목표의 중요도 및 달성도

[표 7]은 설계 목표의 중요도 및 달성도이다. 표에 제시된 중요도는 시뮬레이션 상에서 나타난 수치로서, 시뮬레이션 설계 정도를 나타낸다. 달성도는 시뮬레이션 달성도를 말하며, 수행내용은 중요도와 달성도의 차이를 나타낸다.

[표 7] 설계 목표의 중요도 및 달성도

목표	중요도(%)	달성도(%)	수행내용
기반 시스템 설계	50	50	아두이노 회로도, UML, 공명통 설계
기반 기술 구현	35	30	음을 출력시켜주는 발주한 부품 도착하지 못하여 구현 확인 미흡
테스트 및 오류 수정	10	5	발주한 부품이 도착하면 미흡한 부분의 오류 수정 및 테스트 예정
외관 제작	15	10	기반 시스템 개발에 비중을 두어 외관 설계만 진행
평균	25	23.75	

IV. 참고문헌

- [1] MIDI Association, <https://www.midi.org/> (2018.09.14)
- [2] Roli, <https://roli.com/products/seaboard> (2018.09.14)
- [3] Arduino, <https://www.arduino.cc> (2018.09.15)
- [4] Center for Computer Assisted Research in the Humanities at Stanford University, <http://www.ccarh.org/courses/253/handout/midiprotocol/> (2018.11.18)
- [5] Center for Computer Research in Music and Acoustics, <https://ccrma.stanford.edu/~craig/articles/linuxmidi/misc/essenmidi.html> (2018.11.18)
- [6] LG 사이언스랜드, <http://lg-sl.net/product/infosearch/curiosityres/readCuriosityRes.mvc?curiosityResId=H0DA2006080045> (2018.11.24)
- [7] Arduino MIDI Library, http://fortyseveneffects.github.io/arduino_midi_library/a00008.html#af8bb3cf501a6b530be54ef36864af943a040c84d19624cf23edd3f54c2d432793 (2018.11.18)
- [8] Ji-Dum, <http://www.jidum.com/jidums/view.do?jidumKindCd=Yu> (2018.11.13)
- [9] 박기창, 이현철, 김은석. (2015). UML을 이용한 아두이노 어플리케이션 설계. 한국콘텐츠학회논문지, 15(8), 1-8.
- [10] 로보캅, 2012.02.04, 2018.11.07, <https://robobob.co.kr/81>






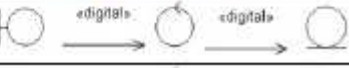

[부 록 1]

➤ UML을 이용한 아두이노 어플리케이션 설계

아두이노 어플리케이션은 C언어를 기반으로 개발되기 때문에 프로그램의 효과적인 설계양식을 제공하지 못하는 문제점이 있다. 선행 연구에서는 기존의 이해하기 힘든 구조도(Structure Chart)를 UML의 확장 매커니즘을 이용한 아두이노 어플리케이션 설계방법을 제안하였다. 여기서 UML은 객체 기술에 대한 표준화 기구에서 인정한 객체지향 분석, 설계를 위한 모델링 언어(모델을 표현하는 언어)이다.

UML에서 스테레오타입은 기존의 메타 클래스에서 추가적인 의미를 부여한 시각적인 표기법을 정의한 하나의 모델요소이다. 스테레오타입은 클래스, 관계, Use-Case, Actor 등 UML의 모든 구성요소에 적용할 수 있으며, 스테레오타입을 표기할 때에는 모델요소에 "<<스테레오타입 명 >>"형식을 붙여 확장된 의미를 부여한다.

아두이노 어플리케이션 설계를 위한 스테레오타입과 아이콘, 태그는 아래의 그림과 같이 정의한다.

스테레오타입	이미지
<<Arduino>>	 아두이노 보드를 표시
<<Digital Input>>	 디지털/아날로그 입출력 표시
<<Digital Output>>	
<<Analog Input>>	
<<Analog Output>>	
<<Value>>	 디지털값 : HIGH  디지털값 : Low  아날로그값 : 0 ~ 255
<<digital>>	
<<analog>>	

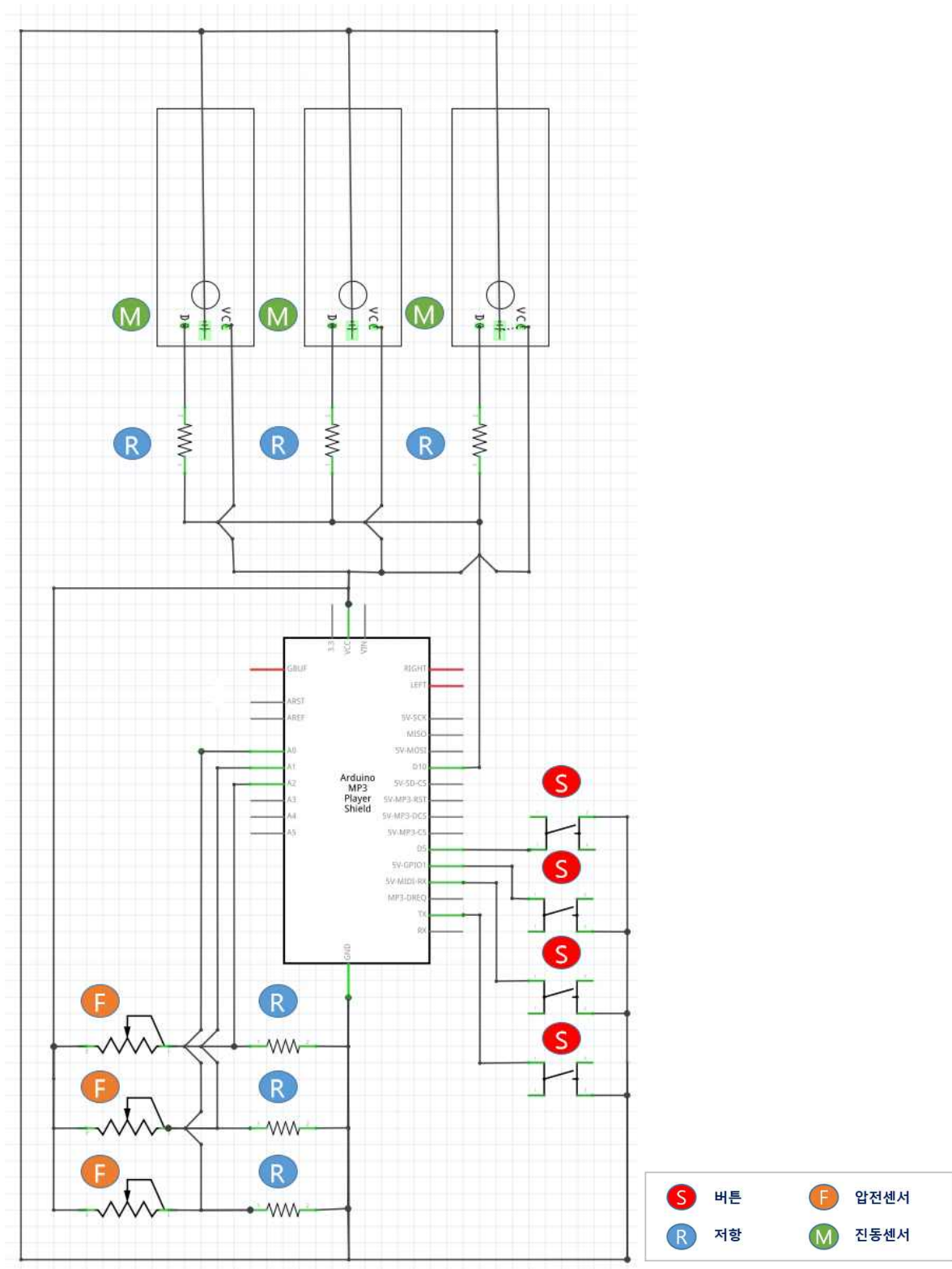
확장된 UML의 스테레오타입과 아이콘(a)

태그	적용대상	설명
pin	아두이노와 입출력 장치 사이의 연관 (Association)	<ul style="list-style-type: none"> • pinMode(), digitalWrite(), analogRead/Write()에 사용되는 파라미터 pin 번호를 표시 • 아두이노와 입출력 장치와 물리적으로 연결되는 핀 번호를 의미
value	출력장치와 출력값 사이의 연관 (Association)	<ul style="list-style-type: none"> • 출력 값을 나타내기 위한 조건 표시

확장된 UML의 태그 정의(b)

[부 록 2]

➤ 아두이노 회로도



[부 록 3]

➤ 시뮬레이션 Source Code

```
#include <SoftwareSerial.h>

SoftwareSerial mySerial(2, 3); // SW시리얼핀 정의 D3가 MIDI 전송용, D2는 미사용

#define speakerOut 7 // 스피커 출력

#define btn1 1 // 음 버튼 1 핀 번호
#define btn2 3 // 음 버튼 2 핀 번호
#define btn3 4 // 음 버튼 3 핀 번호

#define btn4 5 // 공명 버튼 핀 번호

#define piezo1 A0 // 음 버튼 1에 연결된 압전센서 핀 번호
#define piezo2 A1 // 음 버튼 2에 연결된 압전센서 핀 번호
#define piezo3 A2 // 음 버튼 3에 연결된 압전센서 핀 번호

#define vibration 10 // 진동센서 핀 번호

byte note = 0; // MIDI 연주될 note(음계)
byte ledPin = 13; // MIDI 트랙픽 표시 LED

boolean br1; // 음버튼 ON OFF 값 저장될 변수 선언
boolean br2; // 미하 동일
boolean br3;
boolean br4;

int piezo1Value = 0; // 압전센서 PWM 변환
int piezo2Value = 0;
int piezo3Value = 0;

int piezo1SensorValue = 0; // 압전센서 RAW 값
int piezo2SensorValue = 0;
int piezo3SensorValue = 0;

boolean vibrationValue = false; // 진동센서 ON/OFF
```

```

// start setup()
void setup() {
  Serial.begin(57900); // 시리얼포트 Open

  // MIDI Control 시리얼 포트 Open
  mySerial.begin(31250); // Open SoftwareSerial Port

  // 버튼 입출력 상태 정의
  pinMode(speakerIn, INPUT); // 스피커 입력모드 설정
  pinMode(speakerOut, OUTPUT); // 스피커 출력모드 설정

  // 스피커 초기 설정 LOW
  digitalWrite(speakerOut, LOW);

  // 버튼 입출력 상태 정의
  pinMode(btn1, INPUT_PULLUP); // 음 버튼1 풀업입력모드 설정
  pinMode(btn2, INPUT_PULLUP); // 음 버튼2 풀업입력모드 설정
  pinMode(btn3, INPUT_PULLUP); // 음 버튼3 풀업입력모드 설정

  pinMode(btn4, INPUT_PULLUP); // 공명버튼 풀업입력모드 설정

  // 센서 입출력 상태 정의
  pinMode(vibration, OUTPUT); // 진동센서 출력모드 설정

  pinMode(piezo1, INPUT); // 음 버튼 1에 연결된 압전센서 입력모드 설정
  pinMode(piezo2, INPUT); // 음 버튼 2에 연결된 압전센서 입력모드 설정
  pinMode(piezo3, INPUT); // 음 버튼 3에 연결된 압전센서 입력모드 설정

  // 센서 초기 설정 LOW
  digitalWrite(piezo1, LOW);
  digitalWrite(piezo2, LOW);
  digitalWrite(piezo3, LOW);
  digitalWrite(vibration, LOW);
  digitalWrite(ledPin, LOW);
} // end setup()

// start loop()
void loop() {

  br1 = digitalRead(btn1);
  br2 = digitalRead(btn2);
  br3 = digitalRead(btn3);
  br4 = digitalRead(btn4);

  // br4가 LOW이면서 vibrationValue가 true일 때 false로 변경 아니면 true로 변경
  if( br4 == LOW ) {
    if( vibrationValue ) {
      vibrationValue = false;
      // digitalWrite(vibration, LOW);
    } else {
      vibrationValue = true;
    }
  }
}

Serial.print(vibrationValue);

```

```

// 음버튼 br들의 결과값을 br에 저장
boolean br = (!br1 || !br2) || !br3;

if( vibrationValue && br ) {
    piezo1SensorValue = analogRead(piezo1);
    piezo2SensorValue = analogRead(piezo2);
    piezo3SensorValue = analogRead(piezo3);
    // piezo의 값을 시리얼통신으로 출력
    Serial.print("Raw SensorValues \t\t piezo1: ");
    Serial.print(piezo1SensorValue);
    Serial.print("\t\t piezo2: ");
    Serial.print(piezo2SensorValue);
    Serial.print("\t\t piezo3: ");
    Serial.println(piezo3SensorValue);
    // PWM으로 변환하며 piezoValue에 저장
    piezo1Value = piezo1SensorValue/4;
    piezo2Value = piezo2SensorValue/4;
    piezo3Value = piezo3SensorValue/4;
    //변환된 값을 시리얼통신으로 출력
    Serial.print("Mapped SensorValues \t\t piezo1: ");
    Serial.print(piezo1Value);
    Serial.print("\t\t piezo2 ");
    Serial.print(piezo2Value);
    Serial.print("\t\t piezo3 ");
    Serial.println(piezo3Value);
    // piezoValue값을 vibration핀에 보냄
    if( piezo1Value > piezo2Value ) {
        analogWrite(vibration, piezo1Value);
    } else if( piezo2Value > piezo3Value ){
        analogWrite(vibration, piezo2Value);
    } else {
        analogWrite(vibration, piezo3Value);
    }
}

if( br1 == LOW ) {
    tone(speakerOut, 261, 500);
} else {
    noTone(speakerOut);
}

if( br2 == LOW ) {
    tone(speakerOut, 293, 500);
} else {
    noTone(speakerOut);
}

if( br3 == LOW ) {
    tone(speakerOut, 320, 500);
} else {
    noTone(speakerOut);
}

delay(500);
digitalWrite(vibration, LOW);
} // end loop()

```


[부 록 4]

➤ 공명 제어 시스템 Source Code

```
#define btn1 2 // 음 버튼 1 핀 번호
#define btn2 3 // 음 버튼 2 핀 번호
#define btn3 4 // 음 버튼 3 핀 번호
#define btn4 5 // 음 버튼 4 핀 번호
#define btn5 6 // 음 버튼 5 핀 번호
#define btn6 7 // 음 버튼 6 핀 번호

#define btt 11 // 공명 버튼 핀 번호

#define piezo1 A0 // 음 버튼 1에 연결된 압전센서 핀 번호
#define piezo2 A1 // 음 버튼 2에 연결된 압전센서 핀 번호
#define piezo3 A2 // 음 버튼 3에 연결된 압전센서 핀 번호

#define vibration 12 // 진동센서 핀 번호

byte note = 0; // MIDI 연주될 note(음계)
byte ledPin = 13; // MIDI 트래픽 표시 LED

boolean br1; // 음버튼 ON OFF 값 저장될 변수 선언
boolean br2; // 이하 동일
boolean br3;
boolean br4;
boolean br5;
boolean br6;

boolean brr;

int bn[] = { 60, 62, 64, 65, 67, 69 }; // 음 선언

int piezo1Value = 0; // piezo 센서 값 저장될 변수 선언
int piezo2Value = 0; // 이하 동일
int piezo3Value = 0;

int piezo1SensorValue = 0; // PWM 변환된 값이 저장될 변수 선언
int piezo2SensorValue = 0; // 이하 동일
int piezo3SensorValue = 0;

boolean vibrationValue = false; // 진동센서 ON OFF 값 저장될 변수 선언
```

```

// start setup()
void setup() {
  Serial.begin(115200);

  // 버튼 입출력 상태 정의
  pinMode(btn1, INPUT_PULLUP);
  pinMode(btn2, INPUT_PULLUP);
  pinMode(btn3, INPUT_PULLUP);
  pinMode(btn4, INPUT_PULLUP);
  pinMode(btn5, INPUT_PULLUP);
  pinMode(btn6, INPUT_PULLUP);

  pinMode(btt, INPUT_PULLUP);

  // 센서 입출력 상태 정의
  pinMode(vibration, OUTPUT);

  pinMode(piezo1, INPUT);
  pinMode(piezo2, INPUT);
  pinMode(piezo3, INPUT);

  // 센서 초기 설정 LOW
  digitalWrite(piezo1, LOW);
  digitalWrite(piezo2, LOW);
  digitalWrite(piezo3, LOW);
  digitalWrite(vibration, LOW);
  digitalWrite(ledPin, LOW);
} // end setup()

```

[부 록 5]

➤ 사용 부품

분류	제품명	사용 이유	비고
저항	220 Ω	압전센서와 진동센서	
스위치	아두이노 탭트 스위치	음버튼과 공명버튼구현	
압전센서	압전센서 FSR402	공명구현	
진동센서	아두이노 진동모터 모듈 [ELB060416]	울림통에서 진동생성	