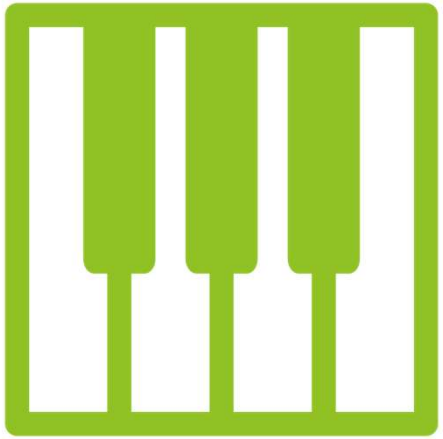


피아노 게임

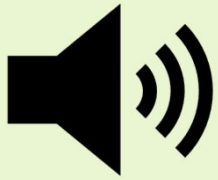
19100054 김시현
19100135 양희란

서울과학기술대학교 SEOULTECH SEOUL NATIONAL UNIVERSITY OF SCIENCE & TECHNOLOGY | 기계시스템디자인공학과

목차

- 주제 선정
- 주제 개요
- 부품 선정
- 회로 설계
- 알고리즘 계획
- 코드 작성
- 결과 영상
- 역할 분담 및 느낀 점

주제 선정



- 중간 과제로 제출한 음악 플레이어 알고리즘 확장

- 박자에 맞춰 정확한 음의 버튼을 눌러 점수를 획득하는 게임

주제 개요

1 대기모드

모드 입력 대기

1번 스위치

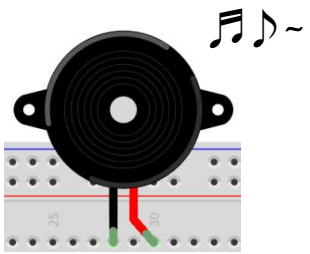
듣기모드

2번 스위치

게임모드

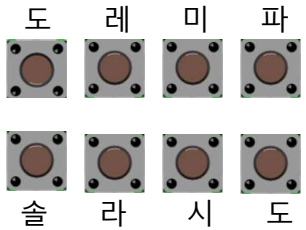
2 듣기모드

박자에 맞춰 노래가 흘러나온다.

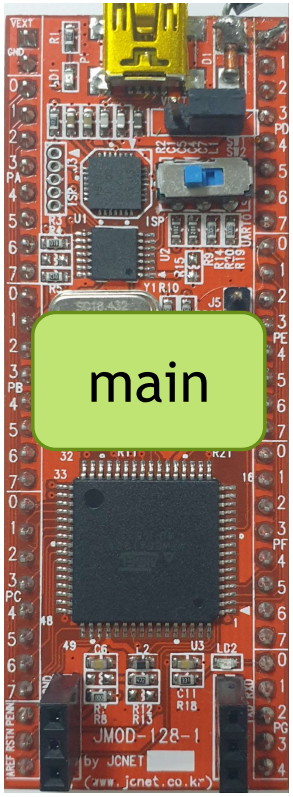


3 게임모드

듣기모드에서 들었던 노래를 박자에 맞추어
알맞은 음의 스위치를 누르고 점수를 매긴다.



부품 선정



ATMEGA128

ARDUINO UNO

- 마이크로컨트롤러

아트메가 메인 선정

-> 포트가 많음

스피커 2개 동시 사용

-> 하나의 보드로 제어 까다로움
-> 아두이노로 스피커 제어

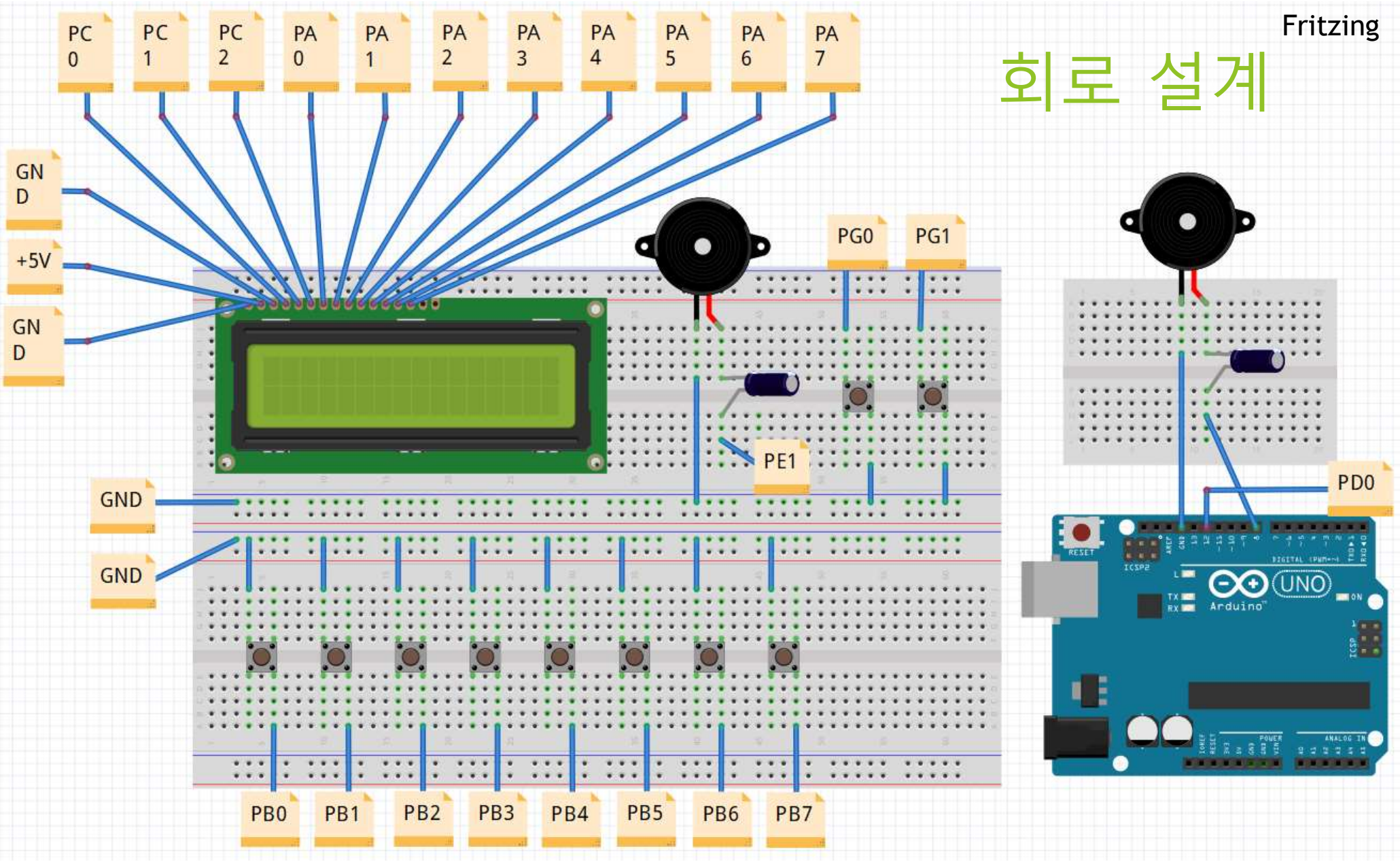
부품 선정

- 사용한 부품 목록

품목	개수	비고
ATMEGA128	1	메인 마이크로컨트롤러
ARDUINO UNO	1	박자 출력 스피커 제어
스위치	10	도~높은 도 입력(8개), 듣기, 게임 모드 입력(2개)
피에조 스피커	2	박자 출력(메트로놈 스피커), 음 출력(멜로디 스피커)
전해 커패시터(10μF, 5V)	2	스피커 연결
16*2 LCD 디스플레이 모듈	1	화면 출력

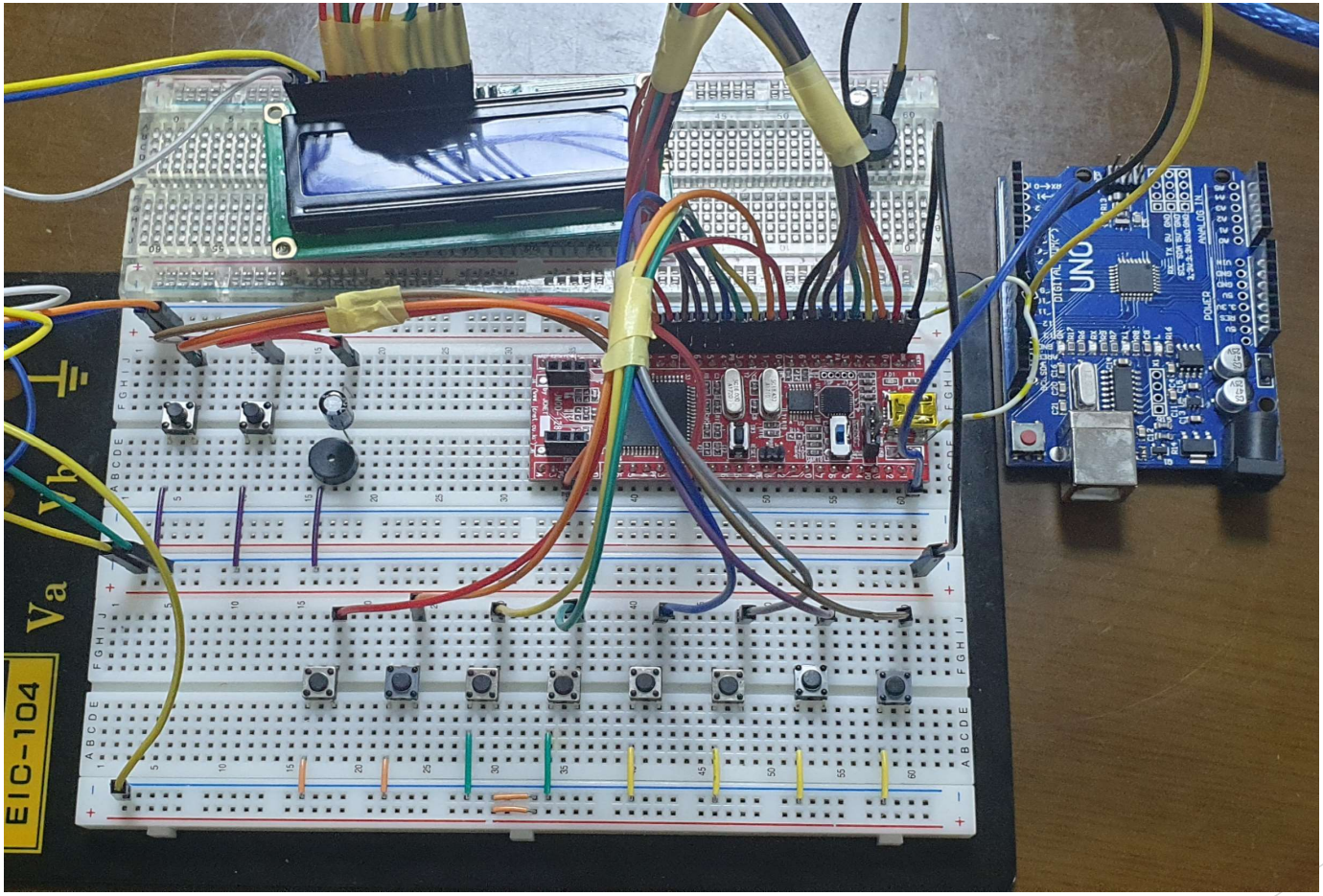
Fritzing

회로 설계



회로 설계

실제회로



알고리즘 계획

대기모드

- LCD 화면 출력
- 스위치 입력 대기

듣기모드

- LCD 화면 출력
- 스피커 출력

게임모드

- LCD 화면 출력
- 스피커 출력
- 타이머 인터럽트
- 스위치 입력 확인

<<대기모드>>

게임을 대기하고 있는 모드이다.

LCD 디스플레이에 문자 출력

LCD 와 ATmega128A 통신

- 8bit interfaece
- 11개 핀 사용

스위치 입력 확인

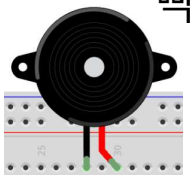
-> 스위치1번을 누르면 듣기모드 진입

-> 스위치 2번을 누르면 게임모드 진입

<<듣기모드>>

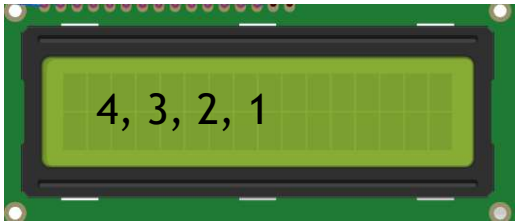
(1) 노래시작전 준비신호

삑 삑 삑 삑



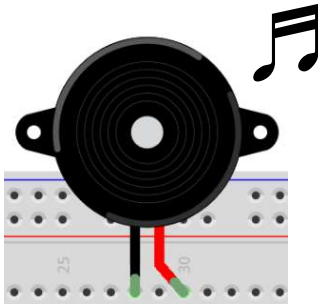
스피커1

스피커로 4박자 울림 and LCD화면에 4321숫자출력



(2) 노래시작

‘도레미파솔라시도’를 각각
‘C D E F G A B CC’ 로 표기

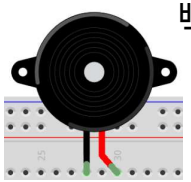


스피커2

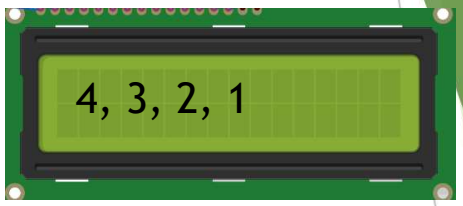
해당 음 밑에 ‘O’를 출력하여 위치 표시

(3) 노래 종료 후 다시 대기모드 진입

<<게임모드>>



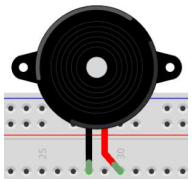
삑 삑 삑 삑



(1) 노래시작전 준비신호

:멜로디 스피커로 4박자 울림 and LCD화면에 4321숫자출력

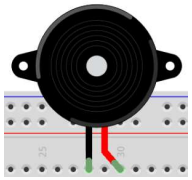
(2) 게임시작



메트로놈 스피커의 박자에 맞춰 기억해둔 음을 차례대로 누른다.

해당 음 밑에 'O' 또는 'X'를 출력하여 게임 진행상황 표시

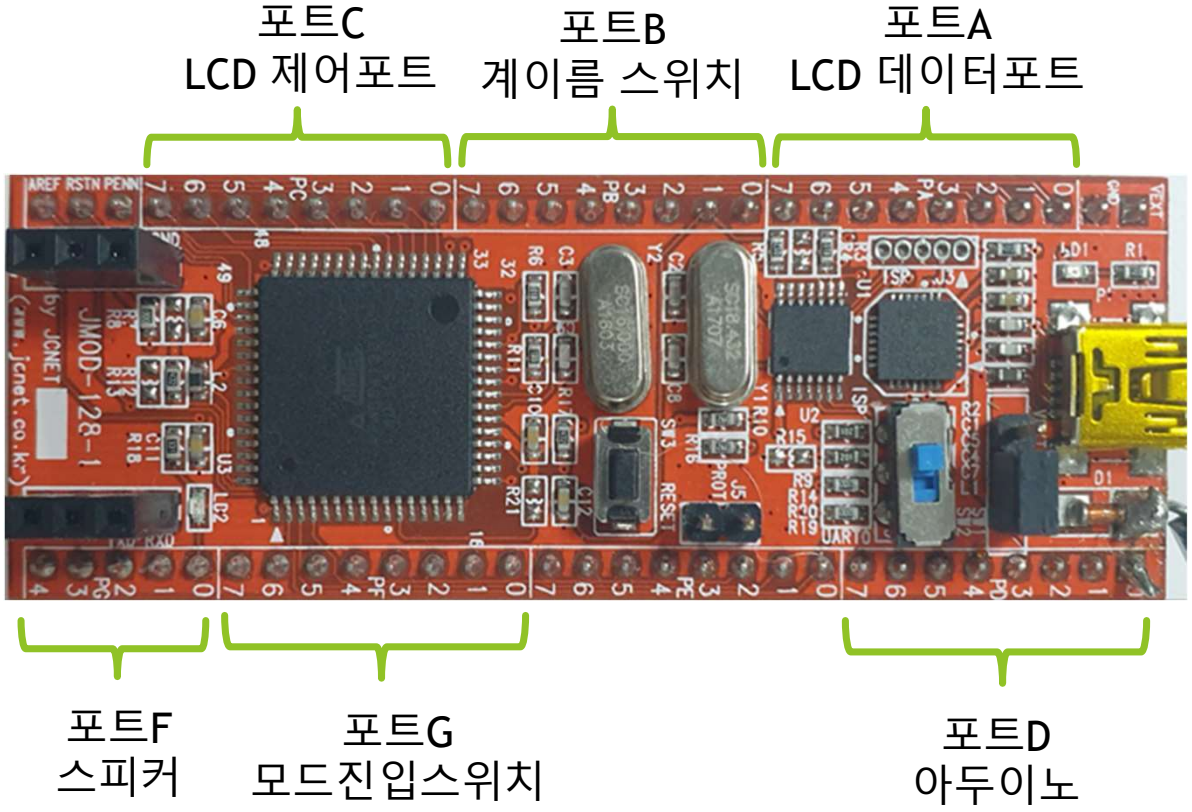
→'O'가 출력 될때 마다 점수 카운트



멜로디 스피커에서 자신이 누른 스위치에 해당하는 음 출력

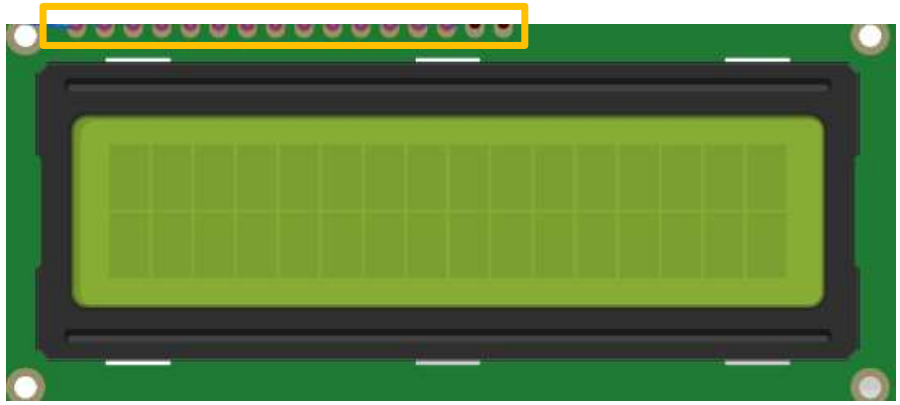
(3) 게임이 끝나면 점수가 LCD화면에 나타난다. 그리고 다시 대기모드로

ATmega128A 포트역할



LCD

Vss VDD VL RS RW E D0 D1 D2 D3 D4 D5 D6 D7 A K



0x 80	0x 81	0x 82	0x 83	0x 84	0x 85	0x 86	0x 87	0x 88	0x 89	0x 8A	0x 8B	0x 8C	0x 8D	0x 8E	0x 8F
0x C0	0x C1	0x C2	0x C3	0x C4	0x C5	0x C6	0x C7	0x C8	0x C9	0x CA	0x CB	0x CC	0x CD	0x CE	0x CF

LCD자리주소

핀 번호	기호	기 능	
1	Vss	0V	전원
2	V_{DD}	5V	
3	V_L	VR 10k	
4	RS	H : 데이터, L : 인스트럭션	
5	R / /W	H : 리드, L : 라이트	
6	E1	H : 인에이블 신호	
7	D0	데이터 버스 4비트 사용시 : D4~D7만 사용 상위 4비트 하위 4비트 8비트 사용시 : D0~D7사용	
8	D1		
9	D2		
10	D3		
11	D4		
12	D5		
13	D6		
14	D7		
15	A	LDE 백라이트 전원	
16	K		

LCD 각 핀 기능

main.c (1/1)

```
#include "piano.h"

ISR(TIMER0_OVF_vect) { //오버플로우 인터럽트 루틴
	TCNT0 = 6; //1ms
	count++;
	if (count > game_len[len_idx]) { //count가 음 길이(혹은 쉼 길이)보다 커지면
		len_idx++; //다음 인덱스로 이동
		mel_idx = (int)(len_idx / 2); //게임모드용 음 길이 인덱스는 쉼 길이와 합쳐져 있어서 인덱스가 2배
		count = 0; //카운트 초기화
	}
}

int main(){
	initialize();//포트 출력 모드 셋팅
	while(1){
		switch(mode){
			case mode_stay:
			stay(); break;//대기모드
			case mode_listen:
			listen(); break;//듣기모드
			case mode_game:
			game();//게임모드
		}
	}
}
```

piano.h (1/9)

```
#include<avr/io.h> //ATmega128 register정의
#define F_CPU 16000000UL //아트메가 주파수랑 맞추기
#define __DELAY_BACKWARD_COMPATIBLE__
#define HALF_CYC 500000UL
#include<util/delay.h>
#include<stdint.h>
#include<stdio.h>
#include<avr/interrupt.h>
#include "lcd.h" //lcd 헤더파일

#define DO  262//계이름
#define RE  294
#define MI  330
#define FA  349
#define SO  392
#define RA  440
#define SI  494
#define DDO 523
```

piano.h (2/9)

```
#define DBL 1208
#define FUL 623 //96bpm(한 박자 = 625ms)로 했으나 LCD write, interrupt 등으로 인한 딜레이로 약간의 수치 조정
#define HAFQUT 467
#define HAF 311
#define QUT 154
#define SIZE 31
#define ON 1
#define OFF 0

void initialize();//포트출력모드셋팅
void stay();//대기모드
void listen();//듣기모드
void game();//게임모드
void lcd_mode_stay();//대기모드 lcd화면
void lcd_mode_listen();//재생모드 lcd화면

void start_set();//듣기/게임모드 준비 함수(듣기/게임모드 시작할 때 스피커로 4박자, lcd에 숫자 출력)
void beep(uint16_t fr); //on time을 받아서 소리를 출력하는 함수
void tone(uint16_t fr); //주파수를 받아서 소리를 출력하는 함수
void tone3(uint16_t fr, uint16_t td, uint16_t tp); //소리를 원하는 길이만큼 출력하는 함수
int get_note(int pb); //스위치에서 음 가져오는 함수
```

piano.h (3/9)

```
unsigned int melody[]={MI, RE, DO, RE, MI, DDO, DDO, RA, SO, MI, DO, RE, MI, SO, MI, RE, MI, RE, DO, RE, MI, DDO, DDO, RA, SO, RA, DO, RE, MI, RE, DO}; //음 배열
unsigned int mel_len[]={HAFQUT, QUT, HAFQUT, QUT, FUL, FUL, DBL, DBL, HAFQUT, QUT, HAFQUT, QUT, FUL, HAFQUT, QUT, DBL+FUL, HAFQUT, QUT, HAFQUT, QUT, FUL, FUL, DBL, DBL, HAFQUT, QUT, HAFQUT, QUT, FUL, FUL, DBL+FUL}; // 음 길이
unsigned int pause[]={0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, FUL, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, FUL}; //쉼
unsigned int game_len[]={HAFQUT, 0, QUT, 0, HAFQUT, 0, QUT, 0, FUL, 0, FUL, 0, DBL, 0, DBL, 0, HAFQUT, 0, QUT, 0, HAFQUT, 0, QUT, 0, FUL, 0, HAFQUT, 0, QUT, 0, DBL+FUL, FUL, HAFQUT, 0, QUT, 0, HAFQUT, 0, QUT, 0, FUL, 0, FUL, 0, DBL, 0, DBL, 0, HAFQUT, 0, QUT, 0, HAFQUT, 0, QUT, 0, FUL, 0, FUL, 0, DBL+FUL, FUL}; //음 길이 + 쉼
unsigned int switch_num[]={2, 1, 0, 1, 2, 7, 7, 5, 4, 2, 0, 1, 2, 4, 2, 1, 2, 1, 0, 1, 2, 7, 7, 5, 4, 5, 0, 1, 2, 1, 0};//스위치 번호 순서
unsigned char lcd_mel[] =
{ 0xc4,0xc2,0xc0,0xc2,0xc4,0xce,0xce,0xca,0xC8,0xC4,0xc0,0xc2,0xc4,0xc8,0xc4,0xc2,0xc4,0xc2,0xc0,0xc2,0xc4,0xce,0xce,0xca,0xC8,0xCA,0xc0,0xc2,0xc4,0xc2,0xc0 }; //재생모드 lcd화면 해당 멜로디 표시
unsigned int note_freq[8] = {DO, RE, MI, FA, SO, RA, SI ,DDO};
unsigned char lcd_note[8] = {0xc0, 0xc2, 0xc4, 0xc6, 0xc8, 0xca, 0xcc, 0xce};

typedef enum {mode_stay, mode_listen, mode_game} MODE;//새로운형 MODE정의
MODE mode = mode_stay;
volatile int count = 0; int len_idx = 0; int mel_idx = 0; //음 길이. 음 인텍스
```

piano.h (4/9)

```
void initialize(){//초기포트셋팅함수
	DDRA = 0xff;//LCD 데이터포트(0~7번 사용)
	DDRB = 0x00; //스위치 8개 입력
	PORTB = 0xff; //내부 풀업
	DDRC = (1<<PORTC0) | (PORTC1) | (PORTC2);//LCD 제어포트(0~2번사용)
	DDRD = (1<<PORTD0); //메트로놈 스피커가 있는 아두이노에 신호를 주는 포트
	DDRF = (1<<PORTF1); //음 스피커 출력
	DDRG = 0x00; //스위치: 1번(재생모드),2번(게임모드) 입력
	PORTG = (1<<PORTG1) | (1<<PORTG2); //내부 풀업

	TCCR0 |= (1<<CS02);
	TIMSK |= (1<<TOIE0);
	TCNT0 = 6;
}

void stay(){ //대기모드함수
	lcd_mode_stay();//대기모드lcd화면
	while(1){
		if((PING & (1<<PORTG1))==0x00){//재생모드스위치on(포트G의 0번)
			LCD_wBCommand(0x01);//LCD화면 클리어
			mode=mode_listen;//재생모드로 바꿔준다.
			break;
		}
		else if((PING & (1<<PORTG2))==0x00){//게임모드스위치on(포트G의 1번)
			LCD_wBCommand(0x01);//LCD화면 클리어
			mode=mode_game;//게임모드로 바꿔준다.
			break;
		}
	}
}
```

piano.h (5/9)

```
void lcd_mode_stay() {
        LCD_Init();//LCD화면 초기화

        LCD_wBCommand(0x80);//시작지점 첫째줄
        LCD_wString("Switch1:listen");//문자출력
        LCD_wBCommand(0xc0);//시작지점 둘째줄
        LCD_wString( " Switch2:game " );//문자출력
}

Void listen(){
        start_set();//노래시작전 준비
        PORTD = 0x01;
        _delay_us(20);
        PORTD = 0x00;

        LCD_wBCommand(0x80);
        LCD_wString("C D E F G A B CC");// LCD화면 첫 줄에 계이름 출력해놓는다.
        for (int I = 0; I < SIZE; i++) { //melody 의 개수만큼 반복
                LCD_wBCommand(lcd_mel[i]);// 음에 해당하는 위치에 문자시작지점으로
                LCD_wString( " O");// 음의 위치에 O표시
                tone3(melody[i], mel_len[i], pause[i]); //소리 출력
                LCD_wBCommand(lcd_mel[i]);//음에해당하는 위치에 문자시작지점으로
                LCD_wString( "   " );// 한 음의 소리가 끝나면 O표시를 다시 빈칸으로

        }
        LCD_wBCommand(0x01);//LCD화면 클리어
        mode = mode_stay;
}
```

piano.h (6/9)

```
void game() { //게임모드
	len_idx = 0; mel_idx = 0; TCNT0 = 6; count = 0;
	int flag = 0; int score=0; int pb = 0; int note; char num[5];//PINB 저장 변수

	start_set();
	PORTD = (1<<PORTD0); //아두이노에 신호 보내서 메트로놈 스피커 on
	_delay_us(20);
	PORTD &= ~(1<<PORTD0);
	LCD_wBCommand(0x80);////////////////////////////////////////
	LCD_wString("C D E F G A B CC");////////////////////////////
	sei(); //전역 인터럽트 활성화

	while (len_idx < SIZE * 2) {
		if ((PINB & 0xff) != 0xff) { //스위치가 눌려있을 때
			if (flag == OFF) { //스위치가 OFF -> ON으로 바뀌면 flag를 ON (스위치 눌릴 때 최초 실행)
				flag = ON;
				pb = (PINB^0xff); //PINB 반전
				note = get_note(pb); //PINB로부터 어떤 음이 눌렸는지 계산
				LCD_wBCommand(lcd_note[note]);
				if (note == switch_num[mel_idx]) {
					//타이머 인터럽트에 의해 음 길이 마다 다음 인덱스로 넘어간다. 눌린 음과 현재 음이 같으면 성공
					LCD_wString("O");
					score++;
				}
				else LCD_wString("X");
			}
			tone(note_freq[note] + 2*note); //소리 출력
		}
```

piano.h (7/9)

```
                        else { //스위치가 눌려있지 않을 때
                                if (flag == ON) {
                                        flag = OFF;
                                        LCD_wBCommand(lcd_note[note]);
                                        LCD_wString(" ");
                                }
                        }
                }
        cli(); //전역 인터럽트 비활성화
                //게임모드가 끝나면 LCD화면에 맞은 개수 표시
                LCD_wBCommand(0x01);
                LCD_Init();
                LCD_wBCommand(0x80);
                LCD_wString("SCORE: ");
                        sprintf(num, "%d", score);
                LCD_wBCommand(0xc0);
                LCD_wString(num);
                _delay_ms(2000);
                LCD_wBCommand(0x01);
                mode = mode_stay;

}
```

piano.h (8/9)

```
void start_set(){
	LCD_Init();//////////////////////////
	char num[5]; //숫자->문자열을 담을 변수
	for (int i = 4; i > 0; i--) { //메트로놈 4번 반복
		sprintf(num, "%d", i); //숫자 -> 문자열 함수
		LCD_wBCommand(0x80);//LCD화면 1행1열에
		LCD_wString(num);//4321숫자출력
		for (int j = 0; j <821; j++) beep(95); //메트로놈의 주기와 반복 횟수는 항상 같기 때문에 상수
		_delay_ms(469);
	}

}

int get_note(int pb) { //PORTB 스위치 값에서 음 가져오는 함수
	int note = 0;
	while (pb != 1) {
		pb /= 2; note++;
	}
	return note;
}
```

piano.h (9/9)

```
void beep(uint16_t on_time) { //on time을 받아서 소리를 출력하는 함수
        PORTF |= (1<<PORTF1); _delay_us(on_time);
        PORTF &= ~(1<<PORTF1); _delay_us(on_time);
}

void tone(uint16_t fr) { //주파수를 받아서 소리를 출력하는 함수
        uint16_t on_time = 1./fr*500000.; //진동수 -> 반주기(us) 변환
        PORTF |= (1<<PORTF1); _delay_us(on_time);
        PORTF &= ~(1<<PORTF1); _delay_us(on_time);
}

void tone3(uint16_t fr, uint16_t td, uint16_t tp) { //소리를 원하는 길이만큼 출력하는 함수
        uint16_t on_time = 1./fr*500000.; //진동수 -> 반주기(us) 변환
        uint16_t n = td*1000./(2.*on_time); //duration time 동안 몇 번 반복할지 계산
        for (int i = 0; i < n; i++) beep(on_time);
        _delay_ms(tp);
}
```

lcd.h (1/5)

```
#ifndef LCD_H_
#define LCD_H_
#define F_CPU 16000000UL
#include <avr/io.h>
#include <util/delay.h>

#define sbi(x, y) (x |= (1 << y))  // x의 y 비트를 설정(1)
#define cbi(x, y) (x &= ~(1 << y)) // x의 y 비트를 클리어(0)

// CON 포트는 포트 C와 연결됨을 정의
#define LCD_CON       PORTC
// DATA 포트는 포트 A와 연결됨을 정의
#define LCD_DATA      PORTA
// DATA 포트의 출력 방향 설정 매크로를 정의
#define LCD_DATA_DIR DDRA
// DATA 포트의 입력 방향 설정 매크로를 정의
#define LCD_DATA_IN  PINA
// RS 신호의 비트 번호 정의
#define LCD_RS    0
// RW 신호의 비트 번호 정의
#define LCD_RW    1
// E 신호의 비트 번호 정의
#define LCD_E     2
```

lcd.h (2/5)

```
// 텍스트 LCD로 부터 상태(명령)를 읽는 함수
unsigned char LCD_rCommand(void){
	unsigned char temp=1;

	LCD_DATA_DIR = 0X00;

	cbi(LCD_CON, LCD_RS); // 0번 비트 클리어, RS = 0, 명령
	sbi(LCD_CON, LCD_RW); // 1번 비트 설정, RW = 1, 읽기
	sbi(LCD_CON, LCD_E);  // 2번 비트 설정, E = 1
	_delay_us(1);

	temp = LCD_DATA_IN;	// 명령 읽기
	_delay_us(1);

	cbi(LCD_CON, LCD_E);  // 명령 읽기 동작 끝

	LCD_DATA_DIR = 0XFF;
	_delay_us(1);

	return temp;
}

// 텍스트 LCD의 비지 플래그 상태를 확인하는 함수
char LCD_BusyCheck(unsigned char temp){
	if(temp & 0x80)		return 1;
	else		return 0;
}
```

lcd.h (3/5)

```
// 텍스트 LCD에 명령을 출력하는 함수 - 단, 비지플래그 체크하지 않음
void LCD_wCommand(char cmd){
        cbi(LCD_CON, LCD_RS); // 0번 비트 클리어, RS = 0, 명령
        cbi(LCD_CON, LCD_RW); // 1번 비트 클리어, RW = 0, 쓰기
        sbi(LCD_CON, LCD_E);  // 2번 비트 설정, E = 1

        LCD_DATA = cmd;          // 명령 출력
        _delay_us(1);
        cbi(LCD_CON, LCD_E);  // 명령 쓰기 동작 끝

        _delay_us(1);
}
// 텍스트 LCD에 명령을 출력하는 함수 - 단, 비지플래그 체크함
void LCD_wBCommand(char cmd){
        while(LCD_BusyCheck(LCD_rCommand()))
        _delay_us(1);
        cbi(LCD_CON, LCD_RS); // 0번 비트 클리어, RS = 0, 명령
        cbi(LCD_CON, LCD_RW); // 1번 비트 클리어, RW = 0, 쓰기
        sbi(LCD_CON, LCD_E);  // 2번 비트 설정, E = 1

        LCD_DATA = cmd;          // 명령 출력
        _delay_us(1);
        cbi(LCD_CON, LCD_E);  // 명령 쓰기 동작 끝

        _delay_us(1);
}
```

lcd.h (4/5)

```
// 텍스트 LCD를 초기화하는 함수
void LCD_Init(void){
	_delay_ms(100);
	// 비지 플래그를 체크하지 않는 Function Set
	LCD_wCommand(0x38);
	_delay_ms(10);
	// 비지 플래그를 체크하지 않는 Function Set
	LCD_wCommand(0x38);
	_delay_us(200);
	// 비지 플래그를 체크하지 않는 Function Set
	LCD_wCommand(0x38);
	_delay_us(200);

	// 비지 플래그를 체크하는 Function Set
	LCD_wBCommand(0x38);
	// 비지 플래그를 체크하는 Display On/Off Control
	LCD_wBCommand(0x0c);
	// 비지 플래그를 체크하는 Clear Display
	LCD_wBCommand(0x01);
}
```

lcd.h (5/5)

```
// 텍스트 LCD에 1바이트 데이터를 출력하는 함수
void LCD_wData(char dat){
        while(LCD_BusyCheck(LCD_rCommand()))
        _delay_us(1);

        sbi(LCD_CON, LCD_RS); // 0번 비트 설정, RS = 1, 데이터
        cbi(LCD_CON, LCD_RW); // 1번 비트 클리어, RW = 0, 쓰기
        sbi(LCD_CON, LCD_E); // 2번 비트 설정, E = 1

        LCD_DATA = dat;        // 데이터 출력
        _delay_us(1);
        cbi(LCD_CON, LCD_E);  // 데이터 쓰기 동작 끝

        _delay_us(1);
}

// 텍스트 LCD에 문자열을 출력하는 함수
void LCD_wString(char *str){
        while(*str)
        LCD_wData(*str++);
}

#endif /* LCD_H_ */
```

Arduino (1/1)

```
//계이름
#define DO 523
#define FA  698
#define SO  784

int freq[32] = {DO, DO, DO, DO, FA, FA, FA, FA, DO, DO, DO, DO, SO, SO, SO, SO, DO, DO, DO, DO, FA, FA, FA, FA, DO, DO, DO, SO, DO, DO, DO, DO};

void setup() {
        pinMode(8,OUTPUT); 메트로놈 스피커 출력 핀
        pinMode(12,INPUT); //아트메가 신호 입력 핀
}

void loop() {
        if (digitalRead(12)) { //신호가 들어오면 메트로놈 스피커 on
                for (int i = 0; i < 31; i++) {
                tone(8, freq[i], 156);
                delay(625);
                }
        }
}
```

결과 영상

유튜브 링크

https://www.youtube.com/watch?v=opjpkgV193c

역할 분담 및 느낀 점

- 김시현

역할: 대기모드, 듣기모드 알고리즘 구현, LCD함수코드작성 및 LCD모듈제어, 회로구성

느낀 점: ATmega128를 제어하는 방법을 더 잘 익힐수 있어서 좋았다. 두명이서 짜는 코드이다 보니 복잡한 코드들을 함수화 하는 과정이 중요하다는 것을 깨달았다. 수업시간에는 배우지 않아서 아쉬웠던 LCD모듈을 공부해서 문자출력 등 구동할 수 있게 되어서 뿌듯하다. I2c통신에 대해서 알게 되었는데 나중에 꼭 활용해보고싶다. 오류가 나는 원인을 분석하여 새로 고치는 과정이 흥미로웠다.

- 양희란

역할: 게임모드 알고리즘 구현, 코드 취합 및 최종 오류 수정, 회로 구성

느낀 점: 게임을 만드는 과정인 만큼 만들 때 재미있었다. 항상 혼자서 짜왔던 코드를 두 명이 나누어 맡으면서 파트 배분, 변수 호환 등 다수가 하나의 코드를 구성하는 방법을 익힐 수 있는 좋은 기회였다. 또한, 타이머 인터럽트에 대한 이해도를 크게 높일 수 있었다.

감사합니다.