PROJECT RESULT

3조

팀원 : 김원, 양승지

목차

I. 개요 및 Motivation

I. 개요 및 Motivation

자율주행자동차 ADAS 기능 기반 아이디어 착안

심야 자율주행 택시

WE,ROBOT

자율주행 기술 단계별 분류

ADAS Sensor

-라이다 : LIDAR(LIght Detection And Ranging / Laser Imaging, Detection, and Ranging)

-레이더 : RADAR(Radio Detection And Ranging)

-카메라 + 인공지능 신경망(딥러닝)

아이디어 착안

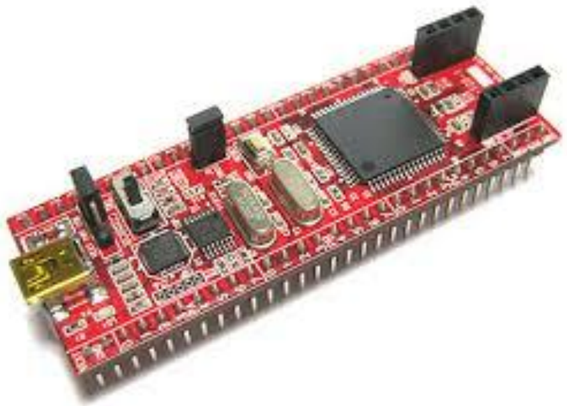
흰색 차선을 따라가는 MINI CAR (구동체)

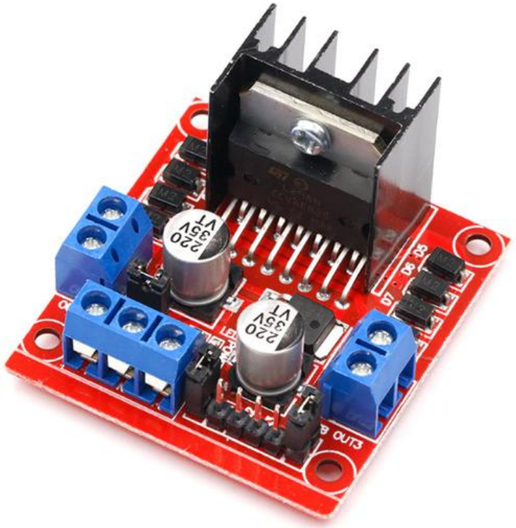
[출처(좌측부터) : 서울시, 테슬라, EVPOST]

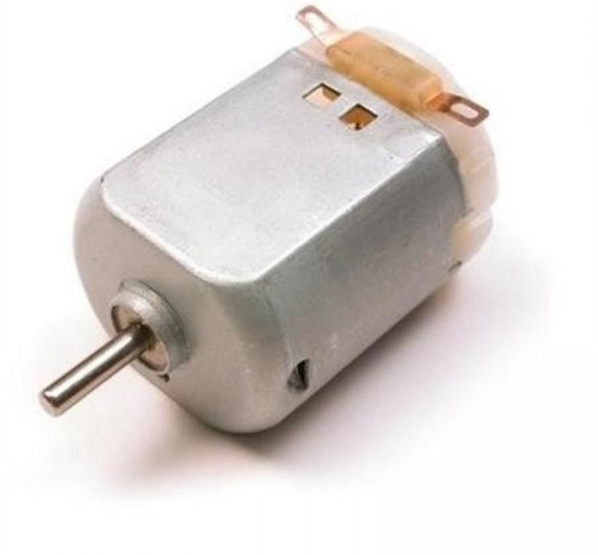
II. HARDWARE

II. HARDWARE

Atmega128, Motor Driver, Motor (2~4ea), Wheel, Infrared Sensor







[출처(좌측부터) :velog,bing, AliExpress , makerlab]

II. HARDWARE

Atmega128, Motor Driver, Motor (2~4ea), Wheel, **Infrared Sensor**

적외선 센서(Infrared Sensor)

적외선 센서 구성 및 기능 및 작동원리

1. 발광부

- 적외선 LED에서 적외선 방출

2. 수광부(Photo Diode)

- 외부 적외선 감지
- 감지 강도에 따라 크기가 다른 전기신호로 변환

활용 기능

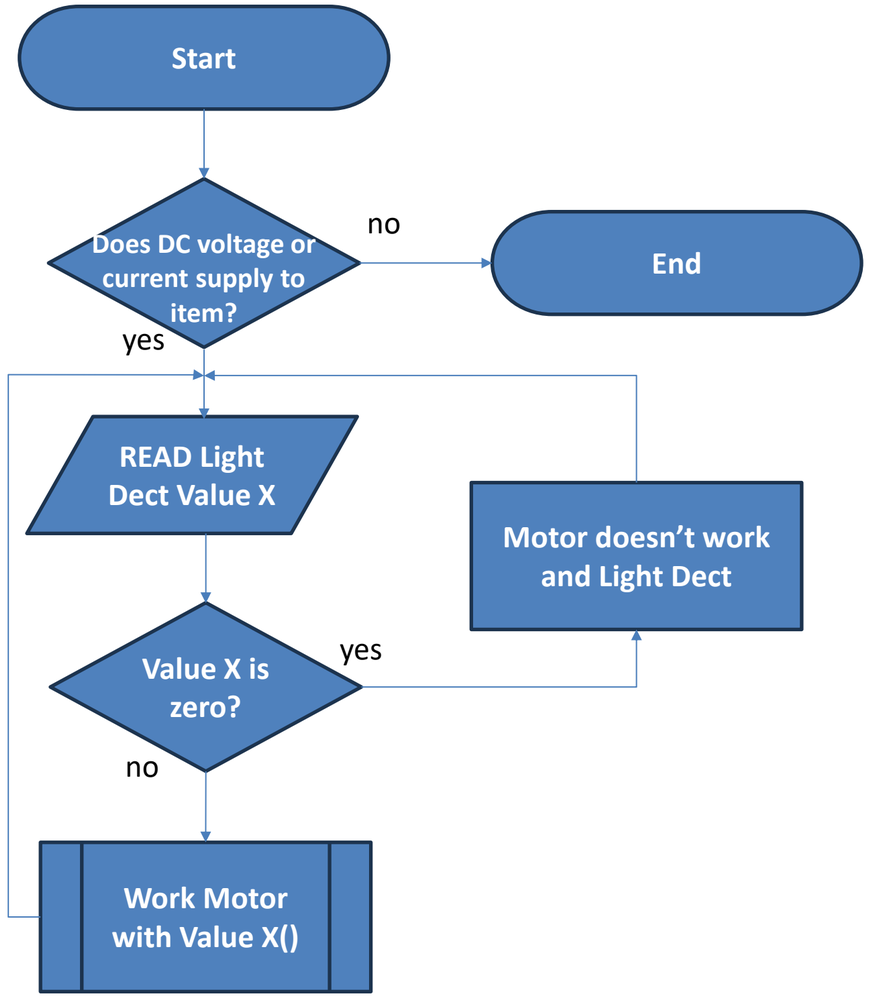
적외선 빛 방출 후 물체의 빛 반사 강도에 따라 감지하여 전기신호로 변환 후 모터 동작

[출처(좌측부터) :11번가]

III. SOFTWARE

III. SOFTWARE

예상 작동 Flow Chart



순서도 설명

1. 전원 연결 여부 판단
2. Photo Diode 빛 감지 양 읽음
3. 감지 양이 0이면 다시 읽음
4. 감지 양 존재하면 Motor 구동하는 함수 호출
5. 호출 끝나고 계속 반복

작동

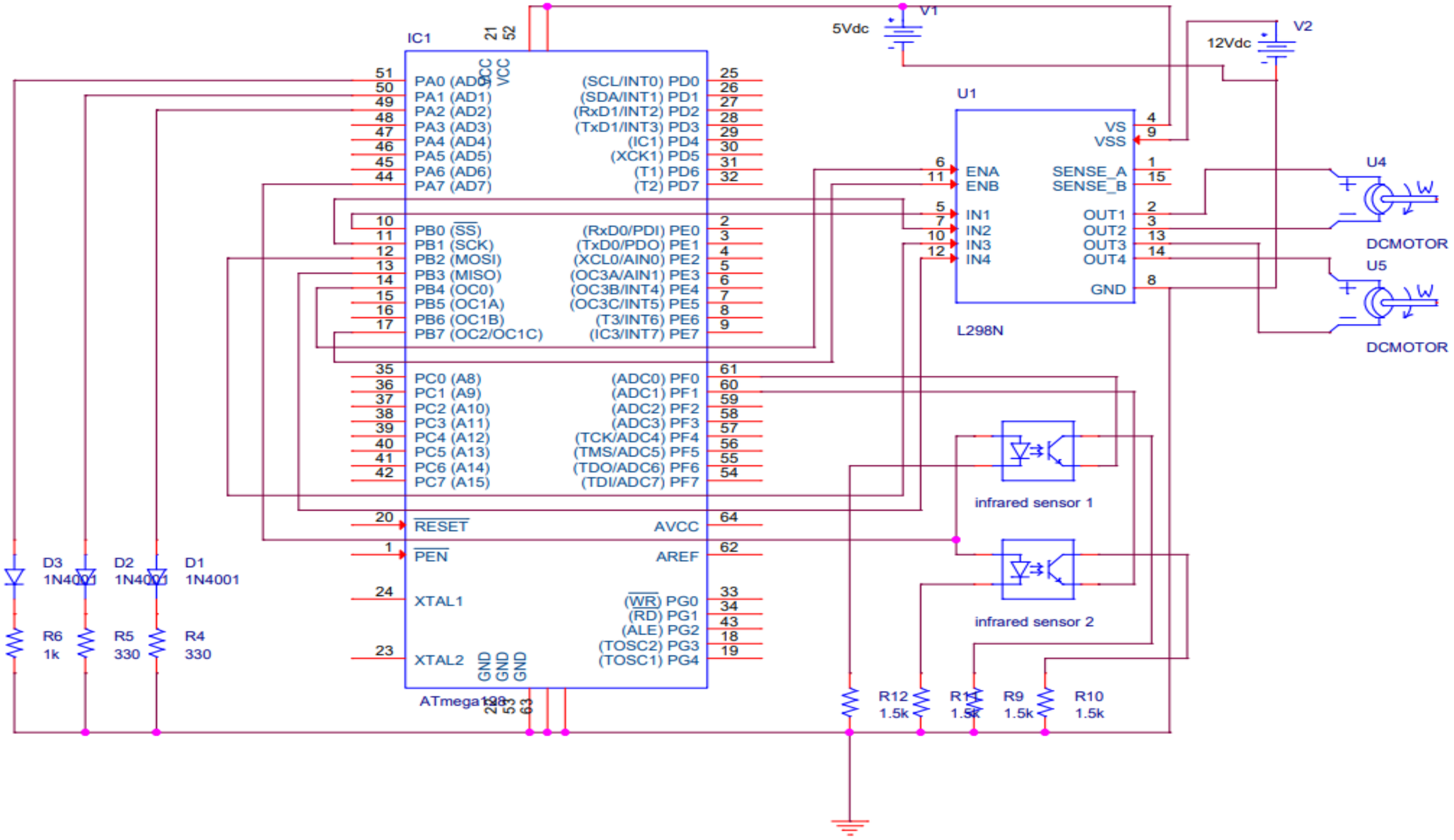
위의 순서도에 따라 흰색 차선에 발광부를 통해 빛을 조사 후 반사되는 빛을 Photo diode를 통해 감지한 빛의 양에 따라 모터를 제어하여 최종적으로, 흰색 차선을 따라가게 하는 **MINI CAR(구동체) 개발**

IV. RESULT

- 회로도
- 코드
- 작동
- 고장탐구
- 계획 및 고찰

IV. RESULT / 회로도

회로도



IV. RESULT / 코드

코드 (1 / 5)

```
#define F_CPU 16000000L
#include <avr/io.h>
#include <util/delay.h>

//함수 미리 선언
void init_adc();
unsigned short read_adc(unsigned char channel);
unsigned short smooth_adc(unsigned char channel);
void pwm_init();
void init_uart0();
void putchar0(char c);
void puts0(char *ps);

void motor_init()
{
	// Timer/Counter0 설정 (모터 A)/
	TCCR0 = (1 << WGM00) | (1 << WGM01) | (1 << COM01) | (1 << CS02); // Fast PWM, 비반전, 분주비 64
	// Timer/Counter2 설정 (모터 B)
	TCCR2 = (1 << WGM20) | (1 << WGM21) | (1 << COM21) | (1 << CS21) | (1 << CS20); // Fast PWM, 비반전, 분주비 64
}
```

IV. RESULT / 코드

코드 (2 / 5)

```
int main(){
	DDRF=0x00; // F port 모두 입력으로 설정
	DDRA=0xff; // A port 모두 출력으로 설정
	DDRB=0xff; // B port 모두 출력으로 설정
	motor_init(); //motor 초기화
	init_adc(); //ADC 초기화
	init_uart0(); // urat 초기화
	PORTB=0x95; //모터의 정방향 출력 (1001 0101) IN1(PB0) : 1, IN2 : 0, IN3 : 1, IN4 : 0, ENA : 1, ENB : 1

	char debug_buffer[50]; // adc 평균값 확인 및 디버깅용 버퍼
	while(1)
	{
		unsigned short value1 = smooth_adc(0);  // ADC 채널 0 평균값 읽기
		unsigned short value2 = smooth_adc(1);  // ADC 채널 1 평균값 읽기

		sprint(debug_buffer, "오른쪽센서: %d, 왼쪽센서: %d\r\n", value1, value2);
		puts0(debug_buffer);
		_delay_ms(100);

// 이론상 OCR0와 OCR2가 동일 값이면 같은 출력 전압과 속도는 같아야 하지만 Hardware 및 구조상 속도가 다르게 나와 ocr값을
   조정하였음.
		OCR0=54;   //Motor A out1,2 쪽(오른쪽 바퀴) 속도제어 63 3.5v, 54일때 2.75v
		OCR2=50;   //MOTOR B out3,4 쪽(왼쪽 바퀴) 속도제어 50 3.53v
		PORTA=0x80; // PA7번 HIGH -> 적외선센서의 발광부 ON
```

20page 에서 다룰 예정

IV. RESULT / 코드

코드 (3 / 5)

```
            if ((value1+value2) > 700) // 두 센서 중 하나가 하얀색 라인에 있을때
            {
                  if(value1>value2) // 오른쪽 센서가 하얀색 라인에 있을때
                  {
                        OCR0=54; // 기본값
                        OCR2=90; // 왼쪽 바퀴의 속도를 올려 우회전
                        PORTA = 0x81; //발광부는 항상 ON, PA0 LED ON
                  }
                  else // 왼쪽 센서가 하얀색 라인에 있을때
                  {
                        OCR0=94; // 오른쪽 바퀴의 속도를 올려 좌회전
                        OCR2=50; //기본값
                        PORTA = 0x82; //발광부는 항상 ON, PA1 LED ON
                  }
            }

            else if ((value1+value2) < 400) // 두 센서 모두 검정색 라인에 있을때 직진
            {
                  OCR0=54; // 기본값
                  OCR2=50; //out 3,4
                  PORTA = 0x84; //발광부는 항상 ON, PA2 LED ON
            }

      }
}
```

IV. RESULT / 코드

코드 (4 / 5)

```
void init_adc() {
	ADMUX = 0x40;  // AVCC(+5V) 기준 전압 사용, ADC0 채널 기본 선택
	ADCSRA = 0x87; // ADC 활성화, 프리스케일러 128분주
}

unsigned short read_adc(unsigned char channel) {
	ADMUX = (ADMUX & 0xF0) | (channel & 0x0F);  // 원하는 채널 선택
	ADCSRA |= (1 << ADSC);  // ADC 변환 시작
	while (!(ADCSRA & (1 << ADIF))) ;  // 변환 완료 대기
	ADCSRA |= (1 << ADIF);  // ADIF 플래그 클리어
	return ADC;  // 16비트 ADC 결과 반환
}

// ADC 이동 평균 필터
unsigned short smooth_adc(unsigned char channel) {
	unsigned long sum = 0;
	for (int i = 0; i < 10; i++) {
		sum += read_adc(channel);  // ADC 값 10번 읽기
		_delay_ms(1);
	}
	return (unsigned short)(sum / 10);  // 평균값 반환
}
```

IV. RESULT / 코드

코드 (5 / 5)

```
// UART 초기화 함수
void init_uart0() {
	UCSR0B = 0x18;  // RX, TX 활성화
	UCSR0C = 0x06;  // 8비트 데이터, 패리티 없음, 1스톱 비트
	UBRR0H = 0;     // 보레이트 상위 바이트
	UBRR0L = 103;   // 9600 보레이트 설정 (F_CPU = 16MHz)
}
// 문자 출력 함수
void putchar0(char c) {
	while (!(UCSR0A & (1 << UDRE0))) ;  // 송신 준비 완료 대기
	UDR0 = c;  // 문자를 송신
}
// 문자열 출력 함수
void puts0(char *ps) {
	while (*ps != '\0') {
	putchar0(*ps++);  // 문자열의 각 문자를 순차적으로 송신
	}
}
```

IV. RESULT / 작동

적외선 센서 값 디버깅 화면

[좌회전]

[직진]

[우회전]

IV. RESULT / 작동

적외선 센서 값에 따른 속도 제어 동작 확인

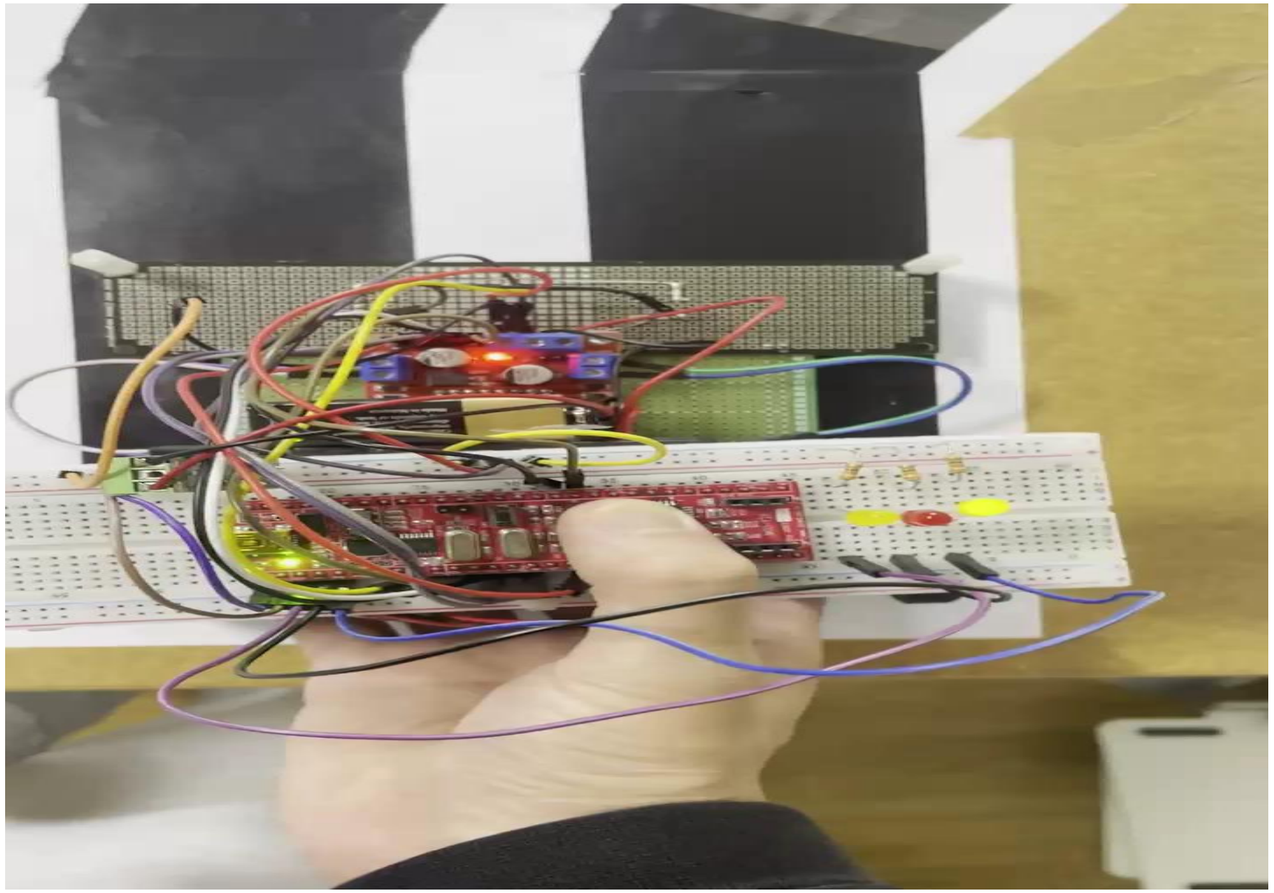
```
            if ((value1+value2) > 700)
            {
                  if(value1>value2)
                  {// 맨 왼쪽 LED

                        OCR0=54;
                        OCR2=90;
                        PORTA = 0x81;
                  }
                  else // 가운데 LED
                  {
                        OCR0=94;
                        OCR2=50;
                        PORTA = 0x82;
                  }
            }

            else if ((value1+value2) < 400)
            {// 맨 오른쪽 LED
                  OCR0=54;
                  OCR2=50;
                  PORTA = 0x84;
            }


      }
}
```

작동 영상

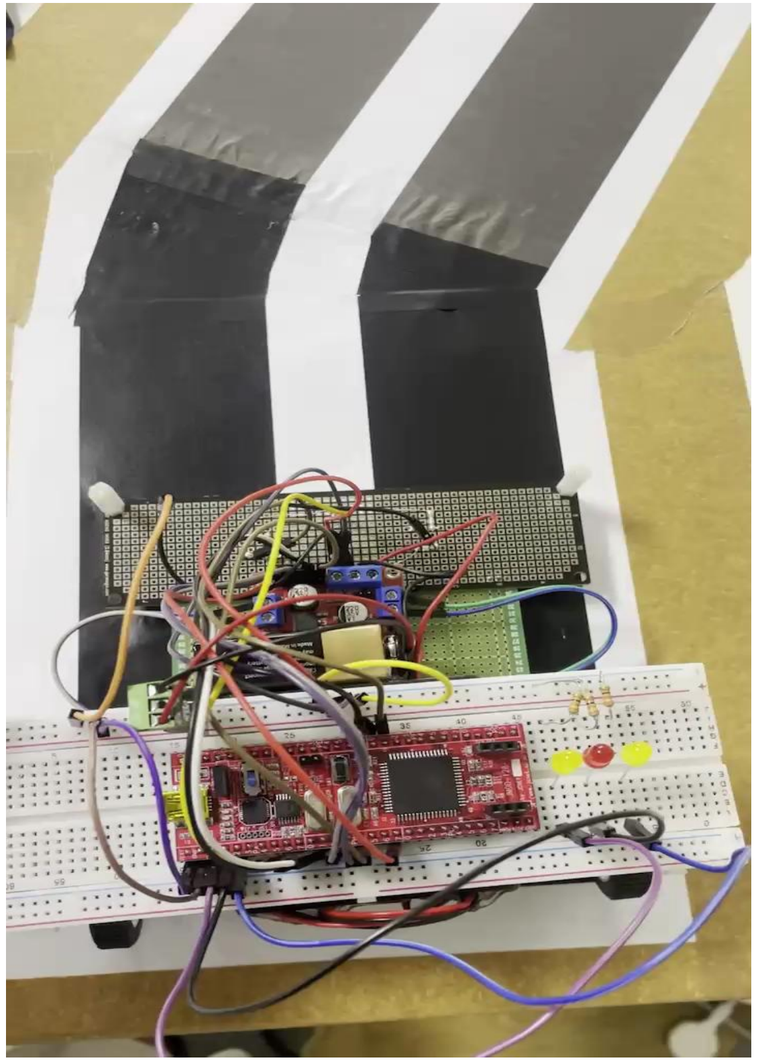


IV. RESULT / 작동

트랙 및 작동영상

run

작동 영상



IV. RESULT / 고장탐구

TCCR0 Register

- TCCR0

7	6	5	4	3	2	1	0
FOC0	WGM01	COM01	COM00	WGM00	CS02	CS01	CS00

WGM01	WGM00	설 명
0	0	일반(Normal) 모드
0	1	위상정정 PWM(Phase Correct PWM) 모드
1	0	CTC(Clear Timer on Compare match) 모드
1	1	고속 PWM(Fast PWM) 모드

WGM : Waveform Generation Mode

COM01	COM00	설명 (*일반 모드의 경우)
0	0	일반(Normal), OC0 미작동
0	1	비교 매치 시 OC0 토글(Toggle)
1	0	비교 매치 시 OC0=0
1	1	비교 매치 시 OC0=1

COM : Compare Output Mode

Clock Select

CS02	CS01	CS00	설명
0	0	0	클럭 입력 차단
0	0	1	1분주, No 프리스케일러
0	1	0	8 분주
0	1	1	32 분주
1	0	0	64 분주
1	0	1	128 분주
1	1	0	256 분주
1	1	1	1024 분주

Fast PWM,

비교 매치 bottom set (when fast pwm),

프리스케일러 64분주

IV. RESULT / 고장탐구

TCCR2 Register

TCCR2(Timer/Counter Control Register 2)

7	6	5	4	3	2	1	0
FOC2	WGM20	COM21	COM20	WGM21	CS22	CS21	CS20

(파형(waveform) 모드 선택)

WGM21	WGM20	설명
0	0	일반(Normal) 모드
0	1	위상정정 PWM(Phase Correct PWM) 모드
1	0	CTC(Clear Timer on Compare match) 모드
1	1	고속 PWM(Fast PWM) 모드

(비교 출력(Compare Output) 모드 선택)

COM21	COM20	설명 (고속 PWM 모드의 경우)
0	0	일반(Normal), OC2 미작동
0	1	예약(미사용)
1	0	비교 매치 시 OC2 클리어, Bottom에서 세트
1	1	비교 매치 시 OC2 세트, Bottom에서 클리어

(클록 분주(prescaler) 선택)

CS22	CS21	CS20	설명
0	0	0	클럭 입력 차단
0	0	1	1분주, No 프리스케일러
0	1	0	8분주
0	1	1	64분주
1	0	0	256분주
1	0	1	1024분주
1	1	0	T2 핀 입력 클록, 하승 에지
1	1	1	T2 핀 입력 클록, 상승 에지

Fast PWM,

비교 매치 bottom set (when fast pwm),

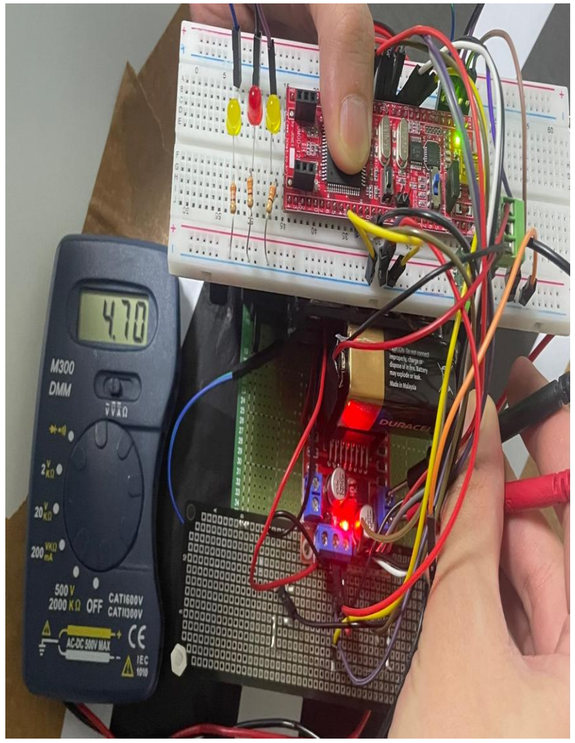
프리스케일러 64분주

IV. RESULT / 고장탐구

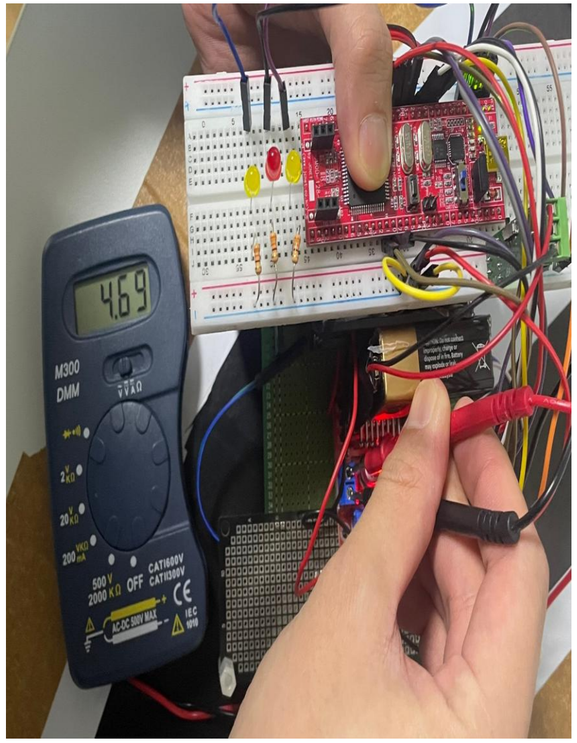
동일한 TCCR 설정 but OCR0, OCR2 같은 값이면 이론적으론 동일한 출력 전압

- 실제 DC전압 측정 값 (motor 연결 없이) OCR0 == OCR2 == 80

[A MOTOR]

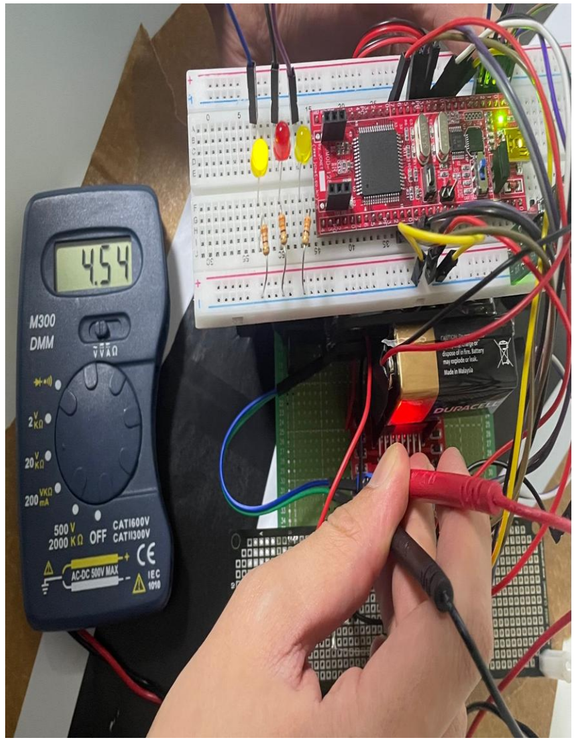


[B MOTOR]

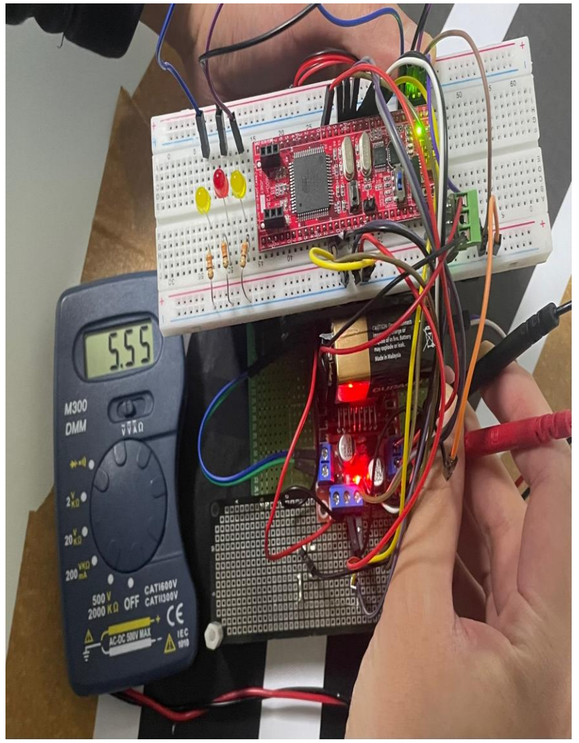


- 실제 DC전압 측정 값 (motor 연결) OCR0 == OCR2 == 80

[A MOTOR]



[B MOTOR]



결론

문제점이라 생각한 OC0와 OC2 간의 간섭은 없었으며 DC모터의 내부 회로에 대한 문제라고 판단되며

이로 인해, 속도를 제어하지 못하여 해당 프로젝트를 기간 내에 수행하지 못하였음.

IV. RESULT

향후 계획 및 고찰

향후 계획

1. 모터 교체 (단기)
2. 센서 값을 바로 읽어 OCR0, 2 속도로 보정하는 함수 구현 (중단기)
3. 카메라 모듈 및 블루투스로 속도 제어하여 주행 (장기)

고찰

이론과 실제는 변수가 존재하여 생각대로 작동 안하는 부분이 있어 여러 방면으로 설계 및 코드 구현 고려

V. Q & A

Any Questions?

THANK YOU