I. 목차

II. 머리말

Tekbot 은 임베디드 시스템이 포함되어 있으며 미국의 Oregon State University 의 공과대학에서 개발된 프로그래밍이 가능한 무인 자동차 로봇이다. 이 프로젝트를 진행하면서 학생들이 전기•전자 공학 분야의 Soldering, 회로 구성을 공부하고 직접 경험함으로써 하드웨어 지식을 학습할 수 있게 되었고 납땜 및 조립을 완료한 Tekbot 프로그래밍을 진행함으로써 컴퓨터 공학 및 소프트웨어 분야의 기초적인 지식을 학습할 수 있다.

이 프로젝트의 학습 목표는 Tekbot 을 제작하고 구동하는 과정의 실습을 통해 전기 공학 및 컴퓨터 과학과 관련된 광범위한 분야에 흥미를 유발하여 시스템 엔지니어링에 대한 학생들의 관심을 유발하고 고무시키는데 있다.

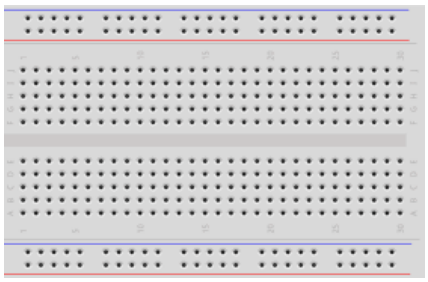
본 보고서는 2019 학년도 1 학기에 진행되었던 모바일 프로젝트 1 에서 진행되었던 Tekbot 실습을 바탕으로 작성되었다. Tekbot 을 조립하고 구동 시키기 까지의 진행 과정 및 발생, 그리고 실습을 진행하면서 겪었던 문제점 및 해결과정에 대해서 초점을 두고자 한다.

III. Hardware

1) 보드 및 부품의 역할과 특징

Tekbot 은 차제와 모터, 그리고 이를 제어하는 보드들로 구성된다. 이 항목에서는 모터와 각종 보드, 그리고 보드를 구성하는 부품의 역할과 특징에 대하여 기술한다.

① 브레드 보드(Bread Board)



[사진 1] 브레드 보드(Bread Board)

브레드 보드는 납땜이 필요 없는 보드이고, 프로토타이핑에 주로 이용된다. Tekbot 에서는 브레드 보드가 한 라인에 전기가 통한다는 점을 이용하여 결선하는 과정을 진행한다.

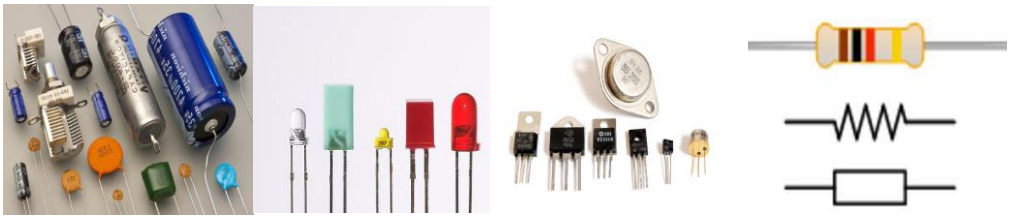
② 커패시터(축전지), 다이오드, 트랜지스터, 저항

커패시터: 전압이 높을 때 전하를 모으고 전압이 낮으면 전하를 방출하는 부품

다이오드: 전류의 방향을 안 쪽으로만 흐르게 하는 부품

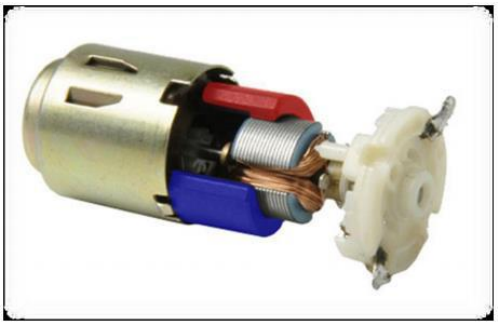
트랜지스터: 회로에서 전류나 전압의 흐름을 조절하며 증폭, 스위치 역할

저항: 전기의 흐름을 방해 및 회로 내에서 전류의 크기를 바꾸는 부품



[사진 3] 커패시터, 다이오드, 트랜지스터, 저항

③ 모터(Motor)



[사진 3] 모터

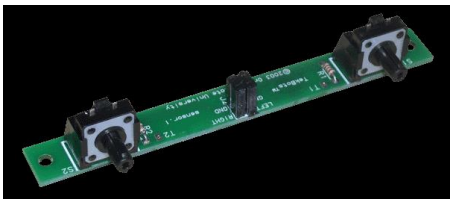
전류가 흐르는 도체가 자기장 속에서 받는 힘을 이용하여 전기 에너지를 회전 에너지로 바꾸는 엔진이며 전동기 라고도 불린다. 전원의 종류에 따라 DC 모터(직류 전동기)와 AC 모터(교류 전동기)로 나뉜다. Tekbot 에서는 [사진 1]의 모터가 사용되었다.

④ 모터 컨트롤 보드(Motor-control Board)

[사진 4] 모터 컨트롤 보드

Tekbot 에서 모터 컨트롤 보드는 Teensy 2.0 보드와 모터에 연결되어 모터 2 개를 개별적으로 제어하는 것을 가능하게 하는 보드이다.

⑤ 센서 보드



[사진 5] 센서 보드

센서보드는 Tekbot 이 장애물을 감지하는 것을 가능하게 하고, 이를 전기적 신호로 모터 컨트롤 보드에 전달하는 역할을 수행하는 보드이다. [사진 5]의 두 스위치는 독립적이며, Tekbot 이 장애물의 방향을 오른쪽 왼쪽으로 나누어서 감지할 수 있게 한다

실습을 진행할 때 센서가 더 잘 인식할 수 있도록 나무젓가락을 붙여주었다.

⑥ 배터리 차저(Charger) 보드

[사진 6] 배터리 차저 보드

배터리 차저 보드는 배터리를 충전하고, Tekbot 에 전원을 공급하기 위한 보드이다. 실제로 보드를 완성하는 실습은 진행하였지만, 이번 실습에서는 Tekbot 에 사용되지 않았다.

⑦ 틴지 보드(Teensy 2.0 Board)

[사진 7] Teensy 2.0 Board

위에서 설명한 모터 컨 트롤 보드, 센서보드를 프로그래밍을 통해 제어하는 역할을 하는 보드이다. 8 비트 기반 MCU 인 aTMEGA32u4 칩을 사용하며 아두이노 및 AVR Studio 를 통해 프로그래밍 할 수 있다. [사진 7]은 Tekbot 실습에 사용된 틴지 보드이다.

2) 조립과정

① 배터리 홀더 부착

배터리(건전지)를 고정하여 모터 컨트롤 보드 및 모터에 전원을 공급해주는 역할을 하는 배터리 홀더를 차체 밑 부분에 부착하였다. 볼트를 이용하여 헐거워지지 않도록 단단하게 조여주었다. 또한 배터리 홀더에 연결된 전선 끝부분의 피복을 벗기고 Female-Header 와 결합할 수 있는 Male-Header 를 납땜하여 연결하였다.

② 롤러볼 & 바퀴 부착

차체 아래 부분에 롤러 볼 1 개와, 바퀴 2 개를 부착하였다. 롤러 볼은 Tekbot 이 원활하게 움직일 수 있도록 해주는 역할을 한다

③ 보드 납땜

모터컨트롤 보드, 센서 보드, 틴지 보드, 배터리 차저 보드에 각각 필요한 부품들의 납땜을 진행하였다.

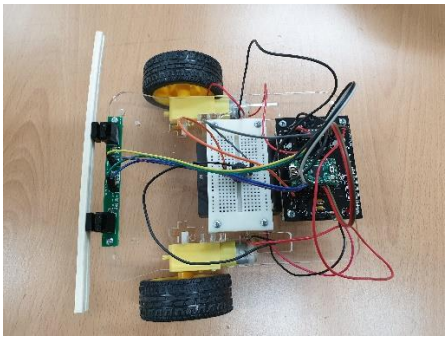
Male-Header, Female-Header, 다이오드, 커패시터, 트랜지스터, 저항 등을 납땜을 통해 연결하였으며 다이오드와 축전기 등의 극성이 있는 부품들은 (+)극과 (-)극을 유의하여 납땜하였다.

④ 와이어링 및 결선

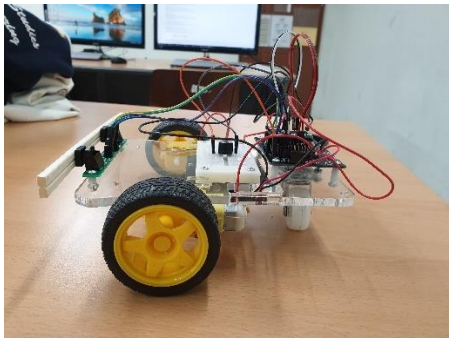
와이어링 과정에서는 틴지 보드를 통한 Tekbot 제어에 있어 동작 위치 및 상호 관계를 정하는 과정으로서 모터와 모터 컨트롤 보드, 센서보드와 모터 컨트롤 보드, 브레드 보드와 모터 컨트롤 보드 사이를 연결시키는 과정을 진행하였다

⑤ 완성된 모습

[사진 8], [사진 9]는 납땜 및 조립 과정을 거쳐 완성된 Tekbot 의 모습이다.



[사진 8] Tekbot 의 상단부



[사진 9] Tekbot 의 측면부

3) 하드웨어 측면의 문제점과 고찰

보드를 납땜하고 Tekbot 을 조립하는 과정에 있어 납땜과 관련된 문제점들이 발생하였다. 처음으로 경험해 본 납땜이 많이 미숙하였기에 발생한 문제들이 대부분이었다.

첫 번째로 발생했던 문제는 납땜 시 사용하였던 납의 양 부족으로 인하여 발생하였다. 모터 컨트롤 보드에 수많은 Female-Header 와 Male-Header 를 납땜하던 중 조교님께서는 납의 양이 부족하다며 충분한 양의 납을 사용해서 화산 모양이 되도록 만들어 달라고 하셨다. 그 말을 듣고서 납을 많이 이용하여 납땜을 진행하였는데 이 과정에서 옆 부분에서 진행된 납땜에 붙는 이중 납땜이 발생하였고 이러한 과정들을 해결하기 위해 Sucker 를 이용하여 납을 빨아들인 이후 다시 납땜을 진행하였는데 이 과정에서 많은 시간을 사용하여 Tekbot 조립에 있어서 다른 팀들 보다 늦어졌지만 차근차근 정확하게 납땜을 진행하여 보드를 완성하였다.

두 번째로 발생한 문제는 모터 컨트롤 보드 자체에서 발생하였다. 이 문제는 차체 조립을 하고 납땜을 하는 때에는 나타나지 않았지만 프로그래밍을 진행하고 코드를 틴지 보드에 다운로드하여 Tekbot 을 동작하였을 때 발생하였다. 간단하게 모터가

앞으로 구동하는 코드를 작성하고 Tekbot 에 결합된 상태인 틴지 보드를 컴퓨터에 USB 케이블로 연결하여 다운로드를 진행하면 모터가 잘 작동하였다. 하지만 이때 USB 케이블을 컴퓨터와 분리하면 Tekbot 이 구동하지 않는다는 것을 발견하였다. 이 문제점을 해결하기 위해 고민하던 중 2 가지의 방면으로 문제점을 예상하고 해결 방법을 찾아가기로 하였다.

첫 번째로는 전원공급의 문제였다. Tekbot 의 모터 컨트롤 보드는 기본적으로 배터리(건전지)에서 공급을 받고 이 전력을 이용하여 회로가 동작하고 모터가 움직이게 된다. 하지만 컴퓨터에 USB 케이블로 연결하면 미세한 전류가 흐르게 되어 모터는 구동되지 않지만 모터 컨트롤 보드에 있는 LED 에 불빛이 들어온다. 이러한 점을 이용하여 배터리에서 모터컨트롤 보드로의 전원공급이 불량한가에 대해서 의문을 가지고 USB 케이블의 연결을 해제하고 배터리만 연결하고 구동하였고, 이때 LED 의 불빛은 켜졌지만 Tekbot 은 구동하지 않았다. 또한 배터리 홀더의 전선이 잘 연결되었는지에 대해서도 확인하고 Male-Header 를 다시 납땜하였지만 여전히 Tekbot 은 컴퓨터에서 연결이 끊기게 되면 작동하지 않았다.

두 번째로는 모터 컨트롤 보드 납땜의 문제이다. 모터 컨트롤 보드를 납땜하여 조립하는 과정에서 정해진 회로 설계와는 다르게 잘못 설계했다고 가정하고 보드에 납땜 되어있는 부품들을 Sucker 를 이용하여 순차적으로 다시 납땜을 진행하였다. 이런 과정을 거쳤지만 여전히 문제는 해결되지 않았다.

세 번째로는 모터 컨트롤보드 자체의 문제이다. 납땜까지 처음부터 다시 진행하였지만 위의 문제가 해결되지 않자 조교님께 문제점을 직접 보여드리고 문제를 해결하기 위해 진행했던 과정들을 말씀드렸다. 조교님께서는 기존에 볼 수 없었던 문제였고 다시 여러 번 테스트를 진행한 후 보드 불량이라고 결론 내리시고 새로운 보드를 다시 지급해 주셨다. 이후 지급받은 새로운 보드를 다시 처음부터 납땜하여 완성시키고 Tekbot 에 결합하여 컴퓨터와의 연결을 해제시킨 후 구동하였을 때 정상적으로 작동하는 것을 확인하였다.

문제를 발견하고 이를 해결하기 위하여 몇 가지의 원인을 가정하고 그 가정들이 참인지를 확인하는 과정을 진행하였는데 결국은 모터 컨트롤 보드의 문제라는 것을 확인하였다. 비록 첫 번째와 두 번째 과정에서 문제점을 해결하지는 못하였지만 다시 한 번 Tekbot 을 점검하는 과정에서 많은 공부를 하였고 이런 과정을 통해서 문제를 해결하는 과정이 Tekbot 을 만드는 데 있어 상당히 의미가 있었다고 생각한다.

IV. Software

1) 소스 코드, 함수구현 및 설명

본 항목에서는 Tekbot 의 모터 제어, 스위치 제어, 부저 음 출력을 위해 작성되어진 코드와 그에 대한 설명을 한다

① 전역 지역에서의 선언

■ #define F_CPU 16000000 와 헤더파일, (열거형)tone_scale 와 tone_array 배열

```
#define F_CPU 16000000
#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>
#define OC1A_WIDTH OCR1A
#define OC1_PERIOD ICR1

enum tone_scale{
    No, Do, Rae, Mi, Pa, Sol, Ra, Si, Do_7};

uint16_t tone_array[9] = {0,1046,1174,1318,1396,1567,1760,1975,2093 };
//X, 도, 레, 미. 파, 솔, 도의 주파수
```

#define F_CPU 16000000	AVR studio에 내장되어 있는 delay 함수를 사용하기 위해 클럭을 정의함
avr/io.h	avr I/O (Input/Output) map에 대한 헤더파일
util/delay.h	delay 함수를 정의 해 놓은 헤더파일
avr/interrupt.h	Interrupt를 사용하기 위한 헤더파일
Enum tone_scale	각각의 음들에 대하여 0 ~ 8의 번호를 부여
tone_array[9]	0, 1046, ~ 은 각각 주파수에 해당

■ 함수 선언

```
void Forward(uint8_t speed);
void Backward(uint8_t speed);
void Rotate(uint8_t speed);
void Stop(uint8_t speed);
void LeftForward(uint8_t speed);
void RighttForward(uint8_t speed);

uint16_t frequency_return(uint16_t fre);

void tone0(uint8_t scale); //delay(10)인 음
void tone1(uint8_t scale); //delay(20)인 음
void tone2(uint8_t scale); //delay(40)인 음
void tone3(uint8_t scale); //delay(80)인 음

void startbuzzer();
void leftbuzzer();
```

```
void rightbuzzer();
void bothbuzzer();
```

- 실제 main 문에서 사용되어 질 함수들을 Global Area 에 선언한다

② 모터 제어 함수 – Tekbot 의 방향제어

1. Forward(직진) 함수

```
void Forward(uint8_t speed)//Function for moving forward robot. Speed : 0 ~ 255
{
        PORTB = 0b10001000;
        PORTD = 0b00000101;
        OCR0A = speed ;//right wheel
        OCR0B = speed ;//left wheel
}
```

- 바퀴가 앞 방향으로 굴러가도록 하는 틴지 보드의 핀 B3, D2를 설정한다.
- 매개 변수 speed를 통해 받은 값의 속도를 이용하여 전진한다.
- 양쪽의 바퀴가 동일한 속도로 굴러간다.

2. Backward(후진) 함수

```
void Backward(uint8_t speed) //Function for moving backward robot. Speed : 0 ~ 255
{
        PORTB = 0b10000100;
        PORTD = 0b00000011;
        OCR0A = speed; ;//right wheel
        OCR0B = speed; ;//left wheel
}
```

- 바퀴가 뒷 방향으로 굴러가도록 하는 틴지 보드의 핀 B2, D1을 설정한다.

- 매개 변수 speed를 통해 받은 값의 속도를 이용하여 후진한다.
- 양쪽의 바퀴가 동일한 속도로 굴러간다.

3. Rotate(180 도 회전) 함수

```
void Rotate(uint8_t speed) //Function for rotating robot. Speed : 0 ~ 255
{
        PORTB = 0b10001000;
        PORTD = 0b00000011;
        OCR0A = speed;//right wheel
        OCR0B = speed; ;//left wheel
}
```

- 차체가 180 도 회전할 수 있도록 양쪽의 바퀴가 다른 방향으로 회전하게 핀 B3, D1 을 설정한다.
- 양쪽의 바퀴가 같은 회전 수로 회전하여 왼쪽 방향으로 돌게 된다

4. LeftForward(좌회전) 함수

```
void LeftForward(uint8_t speed)
{ //Function for moving leftforward robot. Speed : 0 ~ 255
        PORTB = 0b10001000;
        PORTD = 0b00000101;
        OCR0A = speed-100;//right wheel
        OCR0B = speed + 100; ;//left wheel
}
```

- Forward 함수와 마찬가지로 핀 B3, D2을 설정한다.
- 오른쪽 모터의 회전수는 올리고, 왼쪽 모터의 회전수는 낮추어 좌회전한다

5. RightForward(우회전) 함수

```
void RightForward(uint8_t speed)
{ //Function for moving Rightforward robot. Speed : 0 ~ 255
        PORTB = 0b10001000;
        PORTD = 0b00000101;
        OCR0A = speed-100;//right wheel
        OCR0B = speed + 100; ;//left wheel
}
```

- Forward 함수와 마찬가지로 핀 B3, D2을 설정한다.
- 왼쪽 모터의 회전수는 올리고, 오른쪽 모터의 회전수는 낮추어 우회전한다

6. Stop 함수

```
void Stop(uint8_t speed)
{ //Function for moving Rightforward robot. Speed : 0 ~ 255
        PORTB = 0b00000000;
        PORTD = 0b00000000;
        OCR0A = speed-100;//right wheel
        OCR0B = speed + 100; ;//left wheel
}
```

- 핀 출력을 모두 0 으로 해서 멈추도록 하는 함수이다.

③ init 함수

```
void init()
{
    DDRB = 0b10101100; //PB7, PB5, PB3, PB2, PD0 : OUTPUT
    DDRD = 0b01000111; //PD6, PD2, PD1, PD0 : OUTPUT
    DDRF = 0x00;
    PORTF = 0b00000011; // INTERNAL PULL UP

    TCCR1A = 0b10000000; // PWM, OC1A On, Phase and Frequency Correct
    TCCR1B = 0b00010010; // prescaler 8

    TCCR0A = 0b11110001; // PWM, Phase and Frequency Correct
    TCCR0B = 0b00000100; // Prescaler : 256
    TCCR3A = 0b00000000; // Normal
    TCCR3B = 0b00000101; //prescaler 1024
    TIMSK3 = 0b00000001;
    TCNT3 = 0xffff-7812; // Every 1 second
}
```

- 핀의 입출력과 레지스터 설정을 위한 함수이다.
- PB2,3과 PD1,2은 모터와 관련된 핀이며 PB5는 부저와 연결된 핀이다. 따라서 출력 핀으로 설정하였다.
- PF0, PF1은 스위치 센서와 관련된 핀이다
- TCCR0A는 타이머 0의 PWM 제어, TCCR0B는 분주비 설정을 위한 레지스터이며, 각각
- 모드에 맞게 설정했다.
- TCNT3와 TCCR3B, TIMSK3는 타이머3을 이용한 인터럽트를 위한 레지스터이다.

④ Interrupt/Timer 구현

```
ISR(TIMER3_OVF_vect) // 1 초마다 Teensy 의 LED Blink
{
        TCNT3 = 0xffff - 7812; // Every 1 second
       PORTD |= (1 << 6); //D6 ON
       _delay_ms(10);
       PORTD |= (0 << 6); //D6 OFF
       _delay_ms(10);
}
```

- Tekbot 이 작동하면서 1 초마다 Teensy 의 LED 가 깜빡이도록 한다.

■ ISR(Interrupt Service Routine)의 기능

➔ 일반적인 폴링 Polling 함수를 사용할 경우, 프로그램 내 지정된 실행문을 차례대로 실행한다. 폴링의 문제점으로는 delay 의 시간 이 또한 배터리의 출력 등 다른 조건에 의해 바뀌어 정확하지 않다는 점이 있다. 반면 인터럽트 Interrrupt 의 경우 그 요인이 되는 조건만 충족된다면 프로그램 실행 도중 그 프로그램을 중단하고 인터럽트 처리 프로 그램을 실행한다

⑤ Switch 구현

```
while(a) { //두 개의 센서가 눌리면 Startbuzzer 실행 후 반복문 탈출
       if((PINF & 0b00000010) == 0){ Stop(0);
}
       else{ Stop(0);
}
       if((PINF & 0b00000001) == 0){ Stop(0);
}
       else{Stop(0);
```

```
}
        if((PINF & 0b00000011) == 0){
            startbuzzer();
            a--;
            break;}
}
while((a == 0)){

        _delay_ms(10);

        if((PINF & 0b00000011) == 0){ // both-switch - 뒤로 후진 후 180 도 회전
          Stop(0);
        _delay_ms(1);
        bothbuzzer();
        _delay_ms(5);
        Backward(130);
        _delay_ms(250);
        Rotate(100);
        _delay_ms(150);
}
        else if((PINF & 0b00000001) == 0) { //left switch - 뒤로 후진 후 우회전
        Stop(0);
        _delay_ms(1);
        leftbuzzer();
        _delay_ms(5);
        Backward(130);
        _delay_ms(150);
```

```
            //Right(200);
            RightForward(150);
            _delay_ms(100);
}
            else if((PINF & 0b00000010) == 0) { // Right switch - 뒤로 후진 후 좌회전
             Stop(0);
             _delay_ms(1);
            rightbuzzer();
            _delay_ms(5);
            Backward(130);
            _delay_ms(200);
            LeftForward(150);
            _delay_ms(100);
}
            else{ // 앞으로 전진
            Forward(100);
            _delay_ms(1);
}}
```

- 두 개의 스위치가 눌리면 Startbuzzer 함수 실행 후 출발하도록 한다.

- Tekbot 이 움직이다 장애물을 만나서 왼쪽 스위치가 눌리게 된다면 leftbuzzer 함수가 실행되고 이어서 Backward, RightForward 함수가 실행된다. 오른쪽 스위치가 눌리게 된다면 rightbuzzer 함수가 실행되고 이어서 Backward, LeftForward 함수가 실행된다.

  양쪽 스위치가 동시에 눌리게 된다면 bothbuzzer 함수가 실행되고 이어서 Rotating 함수가 실행된다.

2) 소프트웨어 측면의 문제점 및 고찰

Tekbot 의 납땜 및 조립을 마치고 코드를 작성하는 과정에서도 다양한 문제점이 발생하였다. C 언어를 기반으로 코드를 작성한다고 하였기에 익숙하게 다가올 줄 알았지만 군 생활을 마치고 2 년 만에 복학을 한 상황에서 C 언어는 낯설게 다가왔다.

이미 하드웨어 부분에서 모터 컨트롤러의 문제가 발생하였기 때문에 매주 진행되는 Tekbot 의 구현 코드를 배우고 실습하는 수업을 이해하는 데 있어서 어려움을 겪었다. 코드를 이해하고 직접 실행해보려 했지만 Tekbot 의 모터컨트롤 보드에서 발생한 문제를 해결하기 위해 원인을 분석하고 다시 납땜을 조립하는 과정을 진행하면서 약 2 주간의 시간 동안 제대로 코드 구현 및 실습을 진행하지 못하였었다.

이러한 문제점을 직면하고 하드웨어의 문제를 해결하는 과정과 동시에 프로그램 구현 수업을 이해하고 따라가기 위해서 코드를 구현하고 코드를 주입한 틴지보드를 다른 조원들의 Tekbots 에 결합하여 실습을 진행하였다. 이 과정을 통해서 하드웨어 문제 해결을 진행할 때 틴지 보드에는 문제가 없음을 확인하였다. 하지만 와이어링 및 결선이 달라 틴지 보드를 다른 조원들의 Tekbots 에 결합하여 실습하는 과정에도 한계가 존재하였다.

이후 하드웨어 문제를 해결하고 코드를 구현하는 데 있어서 가장 큰 문제점은 분명 같은 코드가 이전에는 적절히 실행이 되었는데 다른 기능을 추가적으로 구현하기 위해 약간 수정하여 틴지 보드에 주입하여 작동시키면 추가적으로 구현하고자 했던 기능만이 아니라 기존에 기능도 작동이 되지 않았다. 초기에는 이러한 문제들의 발생 요인을 찾지 못하여 원인을 찾는데 많은 시간이 소모되었다. 많은 시간과 노력을 들인 결과 코드들이 제대로 정렬되어 있지 않았으며 main 함수에서 선언했던 핀들의 입/출력을 각각의 부저 함수, 모터의 동작 함수에서 중복으로 선언이 되었다는 것을 발견하였다. 이런 중복 요소들을 제거하게 되니 정상적으로 작동하였다. 처음부터 세세한 부분들을 챙겨가며 코드를 작성해

내려갔다면 시간을 더 절약할 수 있었다고 생각했고 다시 한번 중요한 점을 배우게 되었다.

V. 최종고찰 및 맺음말

Teckbots 프로젝트를 시작하며, 기존에 경험해보지 못하고 배우지 못한 것들을 새롭게 배운다는 생각에 기대감이 컸다. 하지만 실제로 실습을 진행하면서 많은 시행착오와 어려움을 겪었다. 납땜, 와이어링, 하드웨어 등을 다루어 본 적이 없었고 코딩을 하는 과정에서도 2 년 만에 접하여 생소했던 C 언어와 비록 C 언어가 기억이 나더라도 차이점이 분명히 존재하였기에 궁극적 목표에 이르기까지 많은 문제점들을 직면하고 해결하는 과정에 연속이었다.

매주 수업에 앞서 진행이 되었던 Pre-lab 과 Main-Lab 을 작성하는 과정도 수월하다고는 할 수 없었다. 하지만 수업을 따라가기에 반드시 필요한 내용들이었기에 실습에 잘 참여할 수 있도록 능동적으로 찾아서 공부하면서 Tekbots 의 관한 이해가 훨씬 더 수월했다.

이 프로젝트를 진행하면서 다양한 문제들을 만나고 이 문제점들을 해결하는 과정을 통해 최종적으로 프로젝트를 마쳤을 때는 조금은 달라진 본인의 모습을 발견할 수 있었다. 어떤 문제를 직면하였을 때 그 문제를 해결하기 위해 다양한 원인들을 가정하고 그 가정의 참/거짓을 증명해 나가는 과정에서 그 문제와 관련된 것만이 아닌 다른 부분들도 자연스레 더욱 더 잘 알게 되었고 또 다른 문제를 직면하였을 때에도 큰 도움이 되었다.

본인 인생에 있어 많은 어려움을 겪었던 첫 프로젝트를 마무리함으로써 한 발짝 성장한 모습을 볼 수 있었고, 이보다 더 어려운 문제를 직면하더라도 문제 해결을 위해 노력하면 할 수 있다는 교훈을 얻었다. 앞으로 더 많이 직면하게 될 프로젝트, 졸업 후의 직장생활에서까지 이번 프로젝트 수업을 바탕으로 더욱 더 도전하는 자세로 임하고 문제를 해결할 것이다. 또한 그 것이 성공할 것이라 생각한다