Term Project

Mini-ADAS Development with Embedding System

SWIP 5G
[김영훈 / 이홍규 / 조형찬 / 홍세정]

HYUNDAI
MOTOR GROUP

Contents

Group Introduction

❖ 5 Group – Development Role & Contribution

	Design
	Development
	Documentation

Main role
-. SW development
(Ultrasonic / RGB / Buzzer System)

Sub role
-. Operation Verification
-. Documentation

Leader
Main role
-. Build SW architecture
-. Design System block diagram

Sub role
-. Presentation

Main role
-. SW development
(Button / LED System)

Sub role
-. Operation Verification
-. Documentation

Main role
-. SW development
(Motor / Potentiometer System)

Sub role
-. HW development

Member	Brainstorming	Building SW Architecture	Block diagram	SW Coding	HW Connection	Verify Operation	Documentation	Presentation
김영훈	30%	50%	50%	30%	20%	30%	-	100%
이홍규	30%	30%	30%	30%	10%	30%	50%	-
조형찬	20%	10%	10%	20%	60%	20%	-	-
홍세정	20%	10%	10%	20%	10%	20%	50%	-

Ⅰ. Development Introduction

◆ Purpose

- Development of Ultrasonic Sensor-based Collision Warning System during forward/backward driving
- Embedded System-Based HW/SW Interlocking

Fig 1. ADAS – BSD

◆ Functional Specification

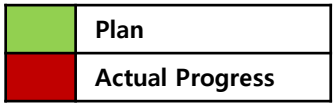
- Setting the direction of motor drive according to the driving mode (Gear - *D* / *R*)
- Changing Collision Warning System by driving mode (*D* - BSD / *R* - PCA)
- Visual/Audible Collision Warning System Operation based on Proximity Object Recognition
- Steering system Implementation with Switch Button & Potentiometer

	A	B	C	D	E	F	G	H	I	J	K	L
1	제목	내용	GPIO	LED	버튼(엣지)	가변저항	ADC	TIMER	PWM	부저	모터	초음파센서
5	핸들 조향각 기반 및 전방 차량 감지 LED 밝기 조절	핸들 (가변저항)을 돌리면 그 각도에 따라서, LED 왼쪽 오른쪽의 밝기를 조절, 초음파 센서를 이용하여 전방 차량이 감지되면 LED밝기를 낮추기	O	O		O	O	O	O			O

Fig 2.Recommanded Sample Project

II. Schedule

◆ Gantt Chart



Month		September		
Date		9th	14th	15th
Propose Design	Brainstorming			
	Building SW architecture			
	System block diagram			
Development	SW Coding			
	HW connection			
	Verify Operation			
Documentation	Documentation			
	Presentation			

Tab 1. Gantt Chart - Project schedule

Ⅲ. Development Process

① System Architecture

- Flow Chart

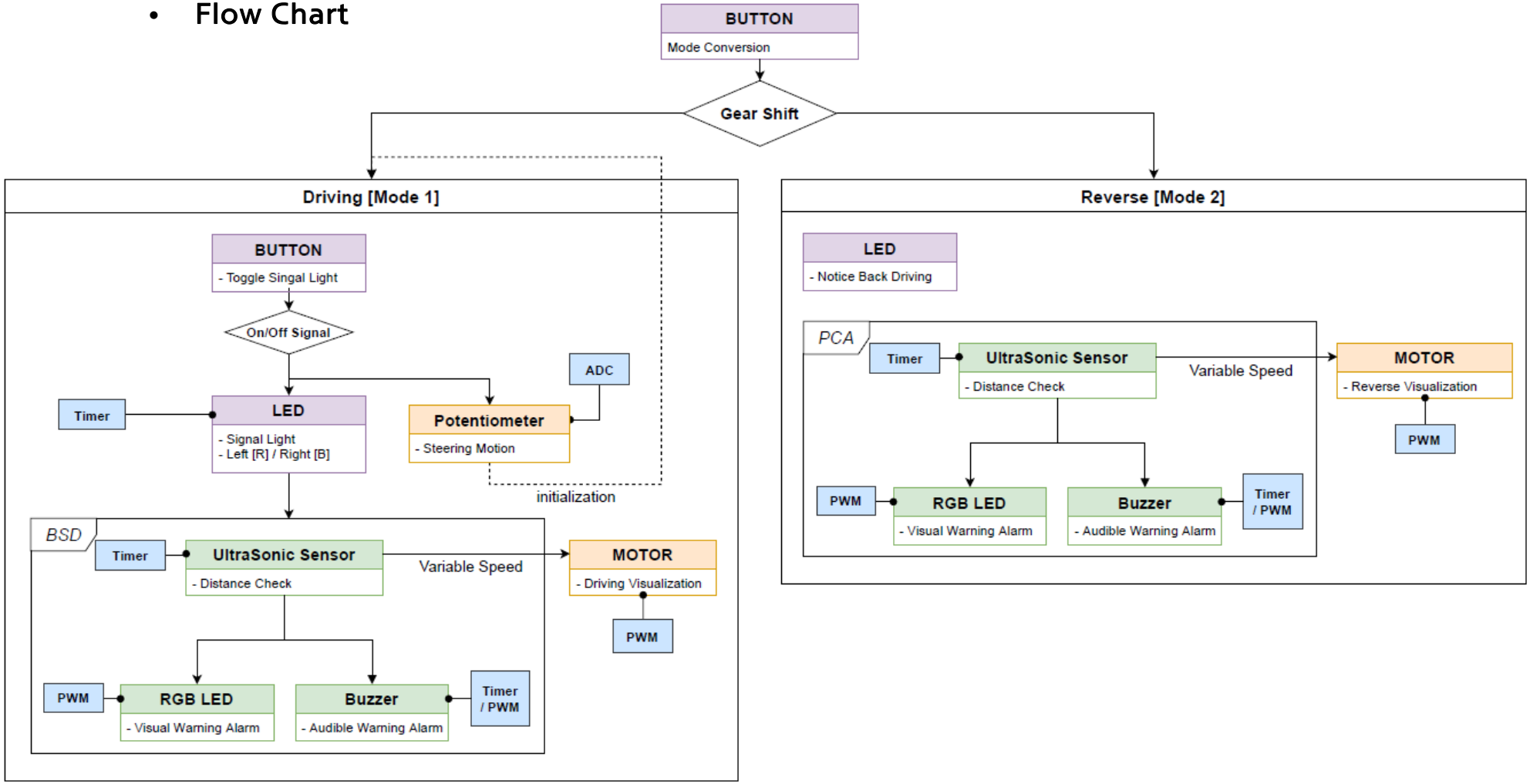


Fig 3. Flow Chart according to Driving Mode

Ⅲ. Development Process

② S/W Development

- Sequence Diagram [Dual Board]

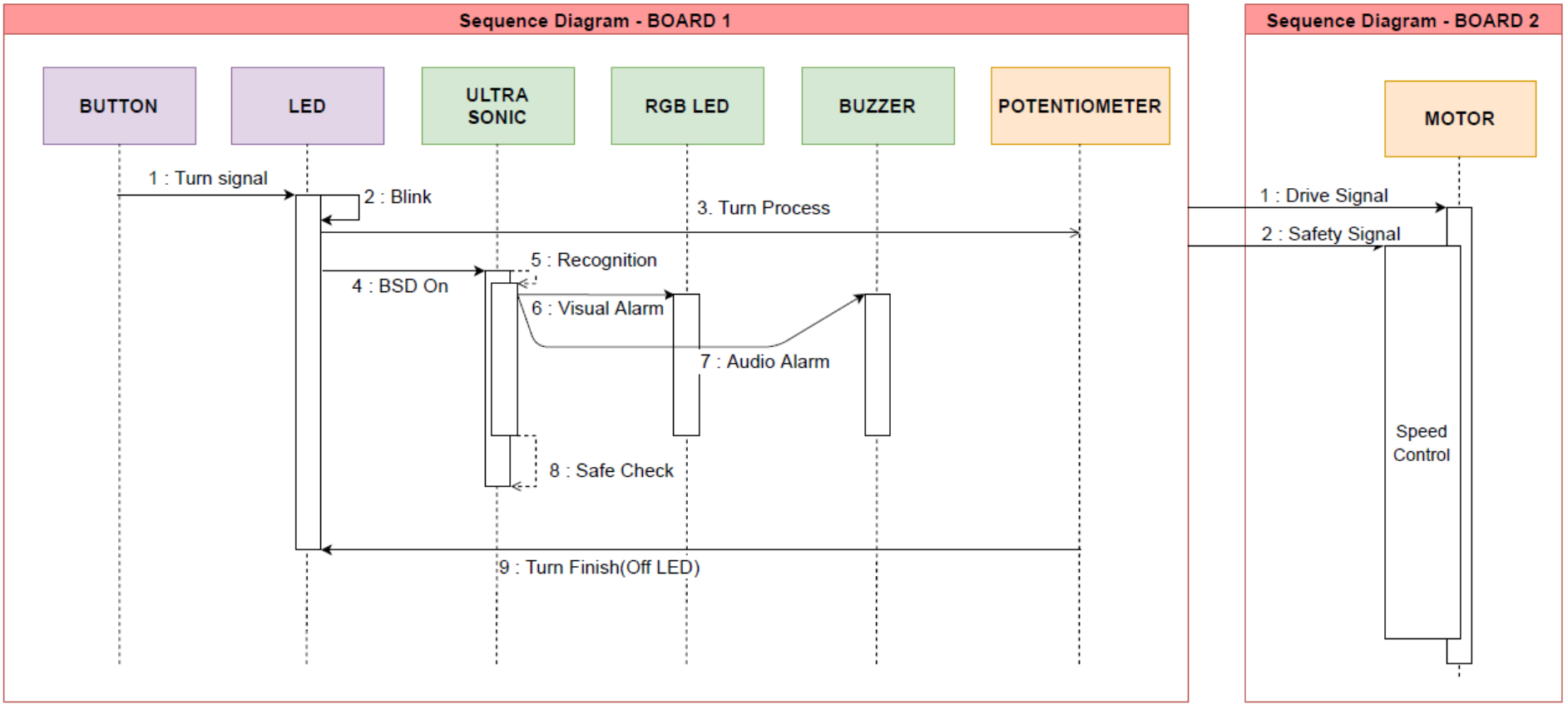


Fig 4. Sequence Diagram according to Board

Ⅲ. Development Process

② S/W Development

➢ New Port Open

1. Problem

- Designed to control 2 buttons and 1 ultrasonic sensor in TC275 board in parallel.

→ Can't control GPIO input and ERU Interrupt setting in single ERS.

→ Button (fixed ERS), Ultrasonic sensor (Need to avoid button ERS)

2. Solution

- Open New Port 11, pin 10 of TC275 board to receive echo signal from ultrasonic sensor.

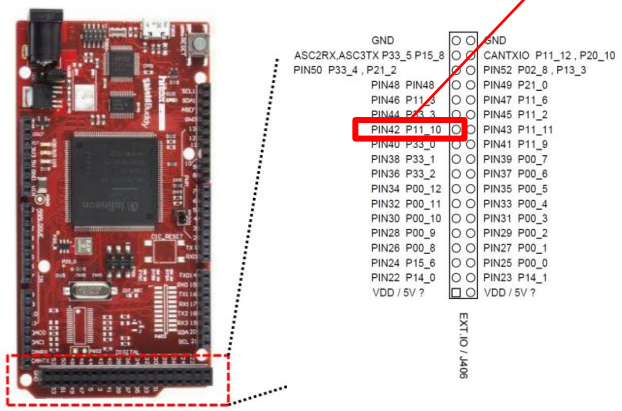


Figure 8 TC275 to Arduino EXT IO Connector Mapping

Table 13-18 Port 11 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select. Reg./Bit Field	Value
P11.10	I	General-purpose input	P11_IN.P10	P11_IOCR8. PC10	$0XXXX_B$
		GTM input	TIN99		
		QSPI1 input	SLSI1A		
		CAN node 3 input	RXDCAN3D		
		ERAY input	RXDB1		
		ETH input	ETHRXD0		
		MSC0 input	SDI00		
		ASCLIN1 input	ARX1E		
		SCU input	REQ12		
	O	General-purpose output	P11_OUT.P10		$1X000_B$
		GTM output	TOUT99		$1X001_B$
		Reserved	–		$1X010_B$
		QSPI0 output	SLSO03		$1X011_B$
		QSPI1 output	SLSO13		$1X100_B$
		Reserved	–		$1X101_B$
		Reserved	–		$1X110_B$
		CCU60 output	CC62		$1X111_B$

7.4.1.2 ERU Input Pin Connections

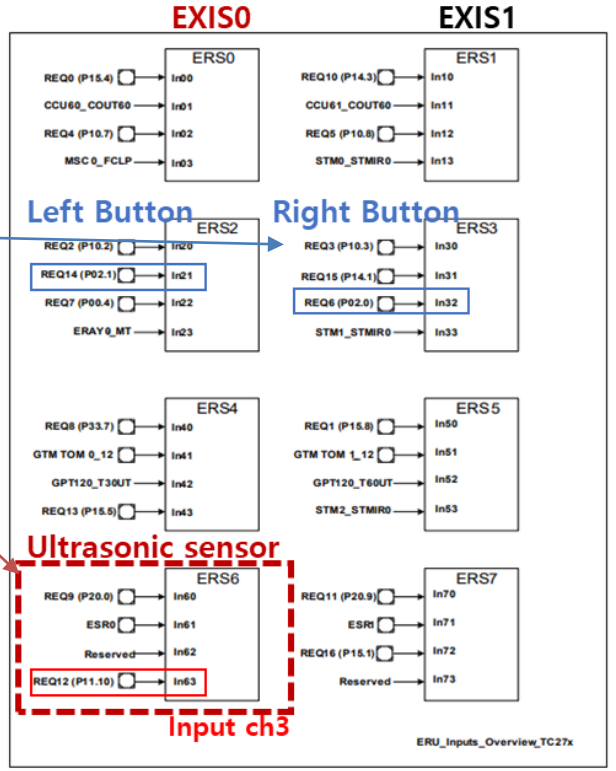
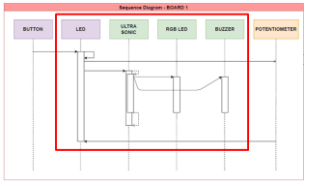


Figure 7-41 External Request Unit Input Connections for TC27x

Ⅲ. Development Process

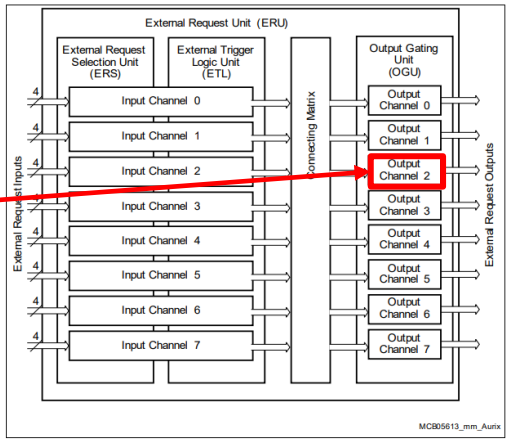


② S/W Development

➢ New Port Open

3. External Interrupt Process of Ultrasonic sensor

1) Confirm GPIO Push button connection in ERS : ERS6 REQ12 (P11.10)
2) Set ERU External Input Channel (EICR Register) : EICR3
3) Select channel (EXIS) : EXIS0
4) Generate Trigger event (ETL) : FEN0, REN0, EIEN0
5) Set trigger event Output channel (OGU) : 0X2
6) Set trigger event as ERU final output : IGP0 enable(0X1)



4. Result Code

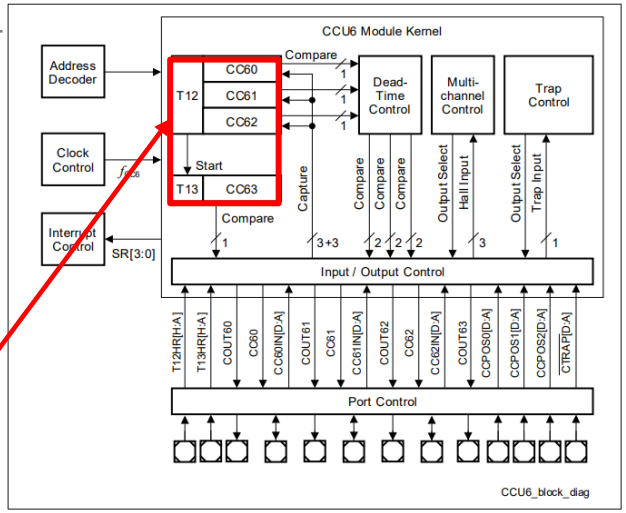
```
// @@@@@@@@@@@@ Ultrasonic Sensor @@@@@@@@@@@@@@@@
SCU_EICR3.B.EXIS0 = 0x3;
// ETL setting (Falling edge)
SCU_EICR3.B.FEN0 = 1;
// (BUZ - Rising edge)
SCU_EICR3.B.REN0 = 1;
// (BUZ - Falling edge)
SCU_EICR3.B.EIEN0 = 1;
// OGU - Output Gating Unit - Trigger Request
SCU_EICR3.B.INP0 = 0x2;
// IGCR - Input Trgigger Connect
SCU_IGCR1.B.IGP0 = 0x1;
// SRC Interrupt setting
SRC_SCU_SCU_ERU2.B.SRPN = 0x0A;
SRC_SCU_SCU_ERU2.B.TOS = 0x0;
SRC_SCU_SCU_ERU2.B.SRE = 1;
```

```
void initUSonic(void)
{
    P02_IOCR4.B.PC6 = 0x10;
    P11_IOCR8.B.PC10 = 0x01;

    P02_OUT.U &= ~(0x1 << P6_BIT_LSB_IDX);
}
```

Ⅲ. Development Process

T12 & T13 Both use



② S/W Development

➢ CCU6x Enable Segmentation

1. Problem

- LED (Constant pulse Timer), Ultrasonic sensor (Need to wait re-start signal after previous trigger pulse ends)

→ Can't control both constant and variable signal in single timer.

2. Solution

- Need to be set up for each module's purpose of use

* LED

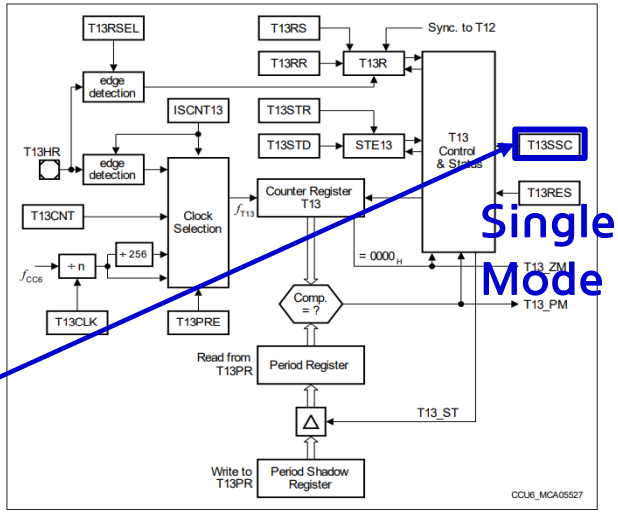
CCU60 T12 – LED Blink

* Ultrasonic sensor

CCU60 T13 – Generate trigger pulse

CCU61 T12 – Measure Echo pulse time

[Single Shot]



Single Shot Mode On

CCU60 — T12 → LED Blink; T13 → U.Sonic Trig

CCU61 — T12 → U.Sonic Echo

Ⅲ. Development Process

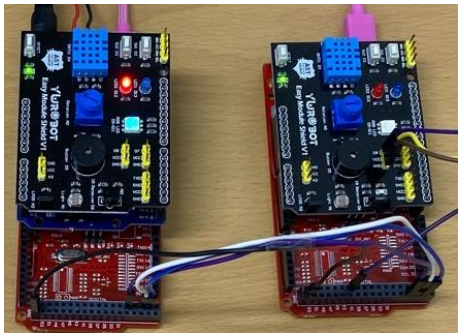
② S/W Development

➢ Dual Board – Control using GPIO signal

1. Problem

- To control the motor, three parameters are needed; Direction(P10,0), Speed(P02.1), Break(P02.7).

→ However, these pins overlap with LED2, switch2, RGB(R) and cannot be used together in the same board.

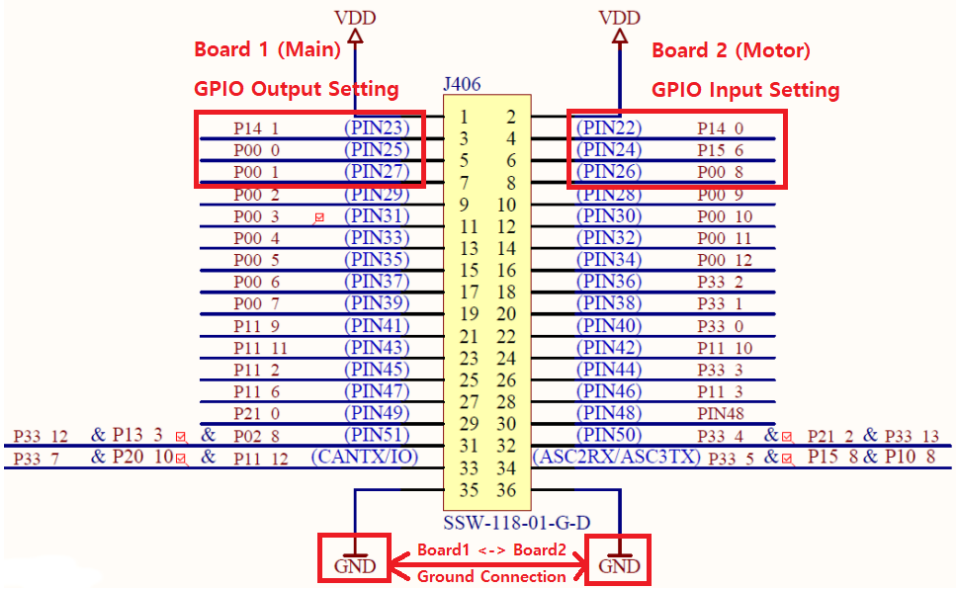


〈Board2〉 〈Board1〉

Channel	Motor Parameter	Easy shield Parametor	Pin Shield buddy
A	Direction	~~LED2~~	P10_1
	Speed(PWM)	~~SW2~~	P2_1
	Break	~~RGB(R)~~	P02_7

2. Solution

- Two boards are used Board 1 performs the main function and sends a motor control signal to board 2.
- Board 2 receives the motor control signal to control the motor.
- Communication between the two boards uses 3 GPIO PINs to create a total of 8 signal combinations.



Ⅲ. Development Process

② S/W Development

- Dual Board – Control using GPIO signal

3. External GPIO Connection between two boards.

1) PIN23 <-> PIN22 (Motor Direction bit)
2) PIN25 <-> PIN24 (Motor Speed Bit1)
3) PIN27 <-> PIN26 (Motor Speed Bit2)
4) PIN35 <-> PIN35 (GND)

Direction bit value	Direction
0	CW
1	CCW

Speed bit1	Speed bit2	Speed
0	0	Zero
0	1	Low
1	0	Middle
1	1	High

4. Result Code

```
// [Master] Motor Control Output to Slave Board
void initGPIOMaster()
{
    P14_IOCR0.B.PC1 = 0x10; // PIN23, P14.1 Output
    P00_IOCR0.B.PC0 = 0x10; // PIN25, P00.0 Output
    P00_IOCR0.B.PC1 = 0x10; // PIN27, P00.1 Output
}
```

```
// [Slave] Motor Control Input from Master Board
void initGPIOSlave()
{
    P14_IOCR0.B.PC0 = 0x02;     // PIN 22, P14.0 Input
    P15_IOCR4.B.PC6 = 0x02;     // PIN 24, P15.6 Input
    P00_IOCR8.B.PC9 = 0x02;     // PIN 26, P00.8 Input

}
```

Ⅲ. Development Process

② H/W Development

- Board 1
  - ✓ Main Board
  - ✓ Timer (CCU6x)
  - ✓ Ultra Sonic Sensor
  - ✓ Button
  - ✓ RGB LED / LED
  - ✓ Buzzer
  - ✓ Potentiometer

GPIO Signal

- Board 2
  - ✓ Motor Controller

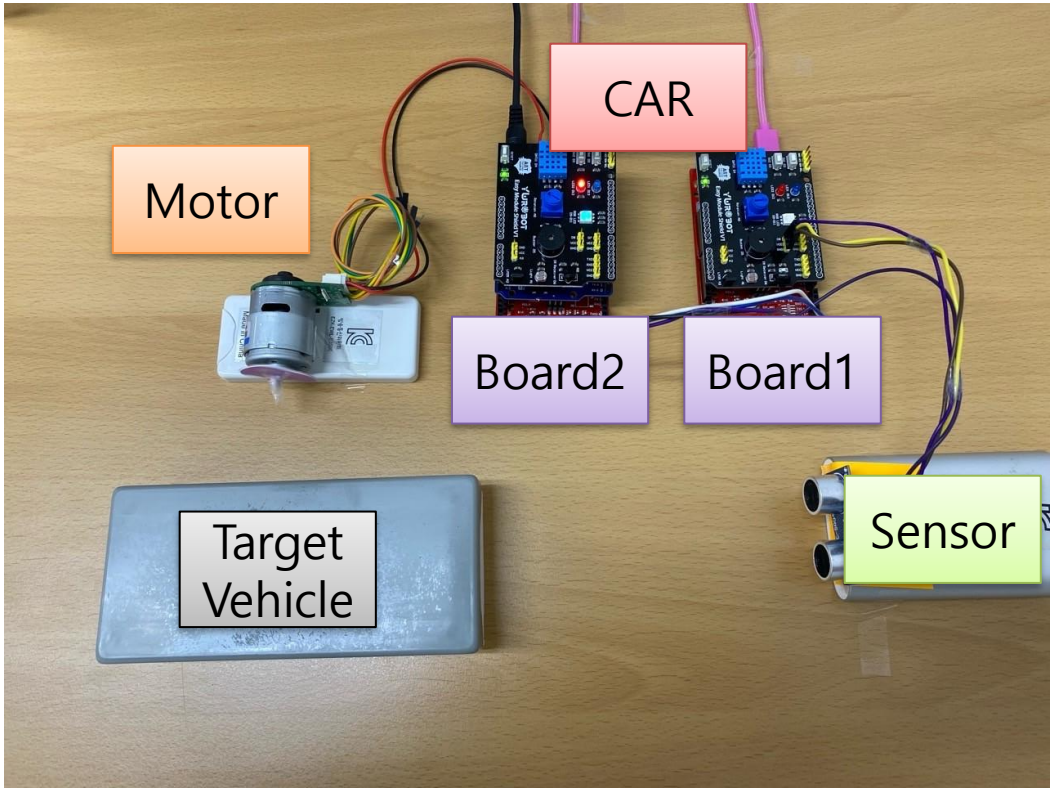
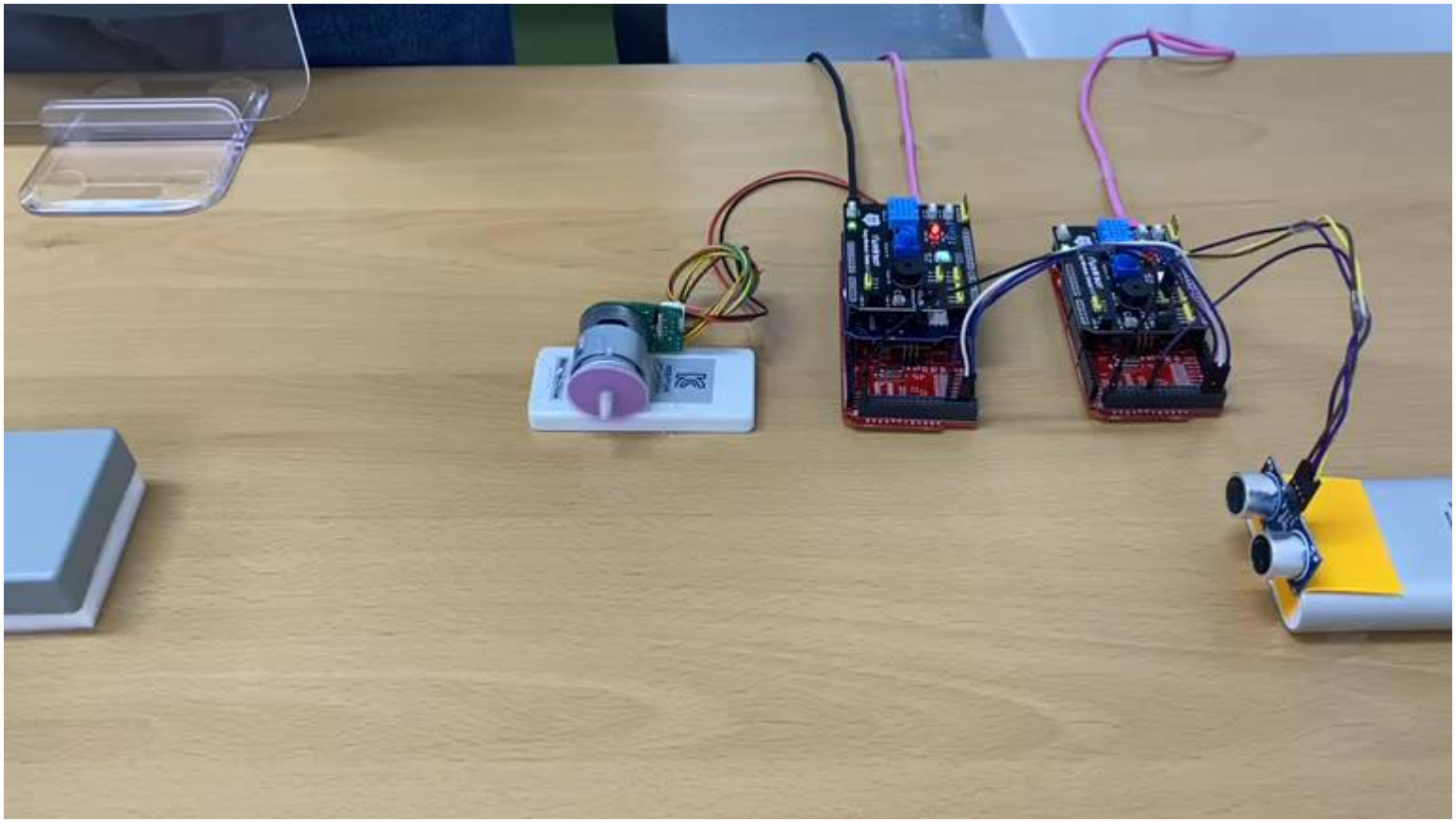


Fig5 . H/W Configuration

Ⅲ. Development Process

③ Result

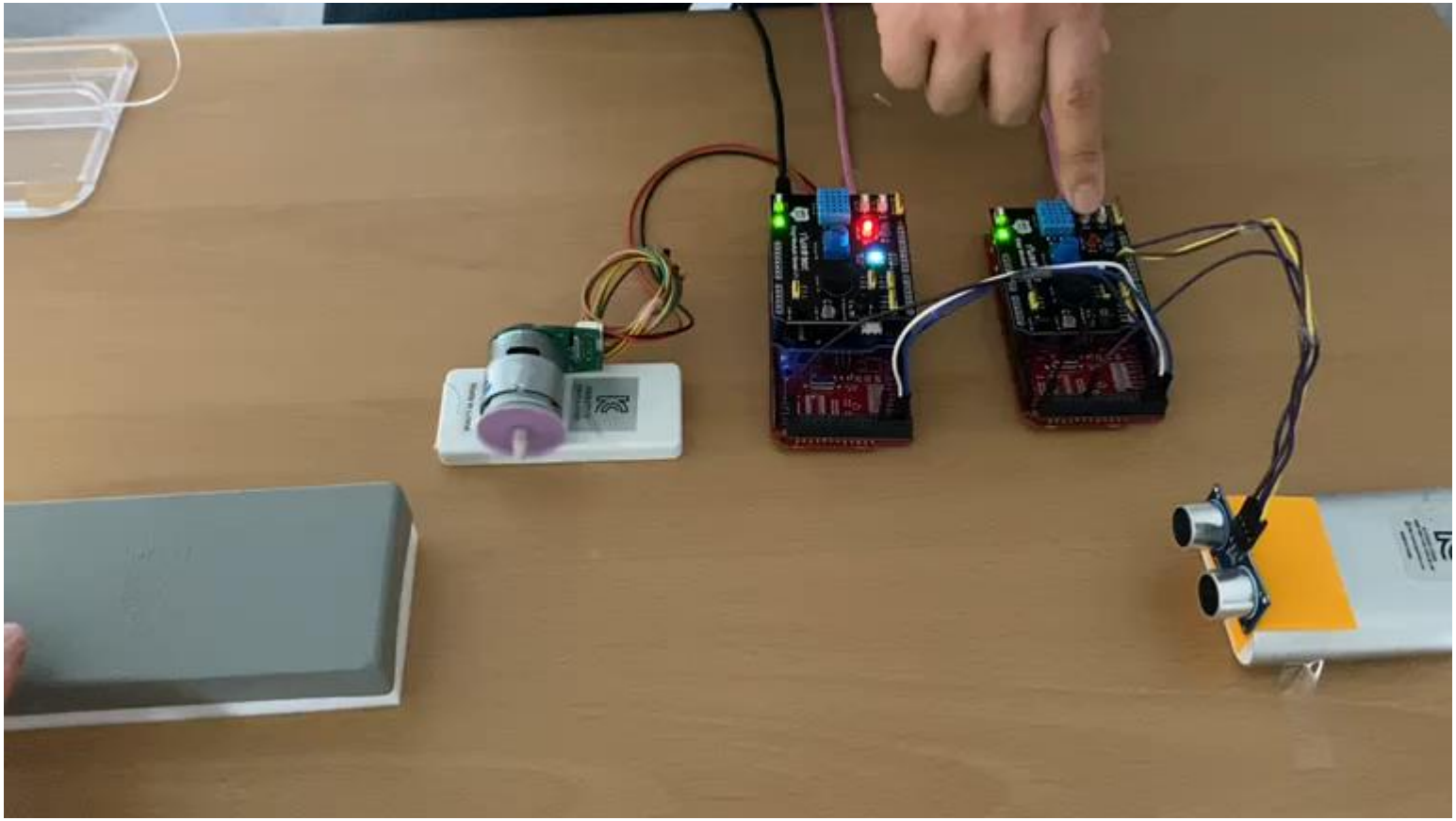
- 영상 1 – [D Gear] A.E.B



Ⅲ. Development Process

③ Result

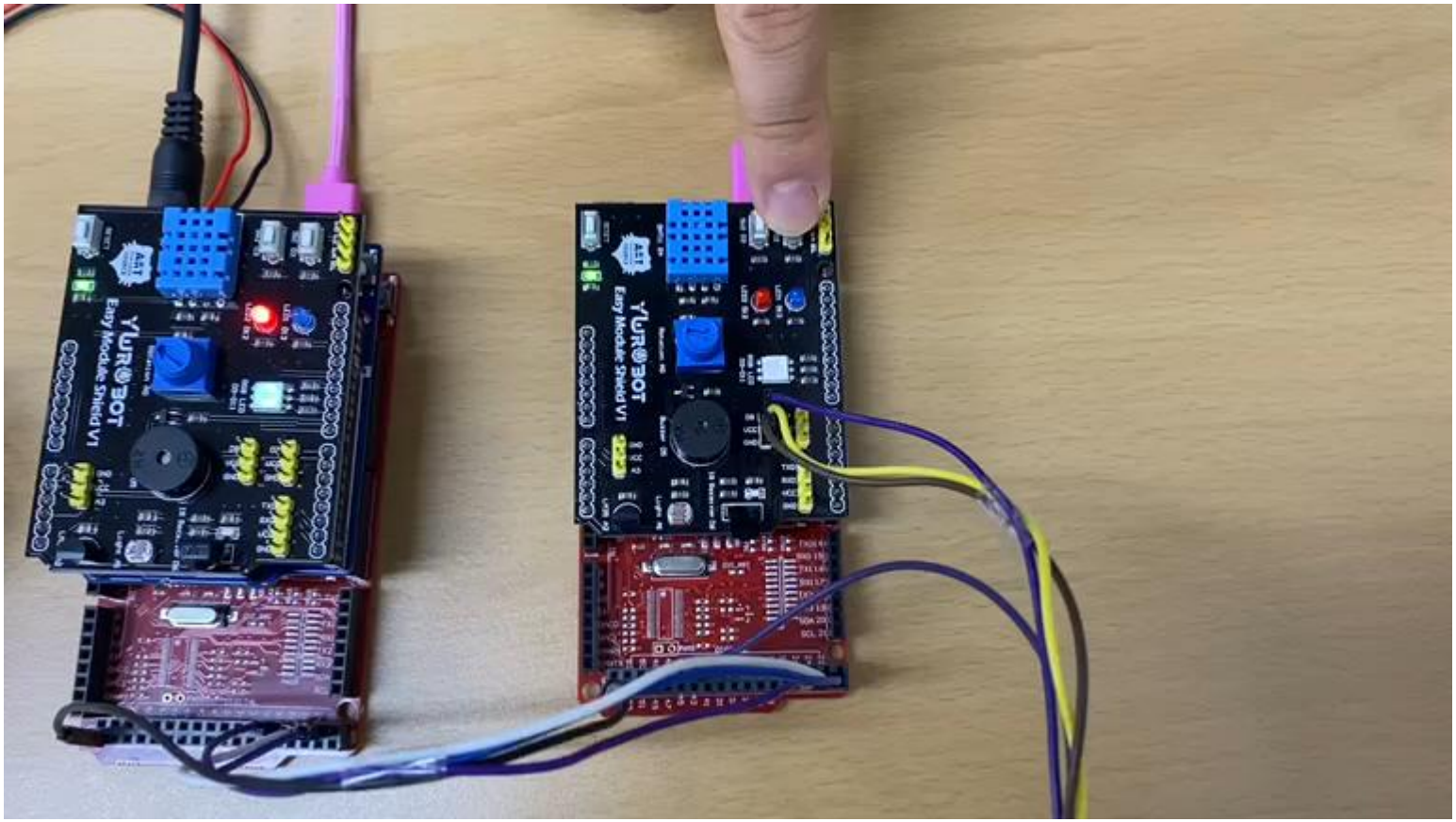
- 영상 2 – [D Gear] Signal Light + B.S.D



Ⅲ. Development Process

③ Result

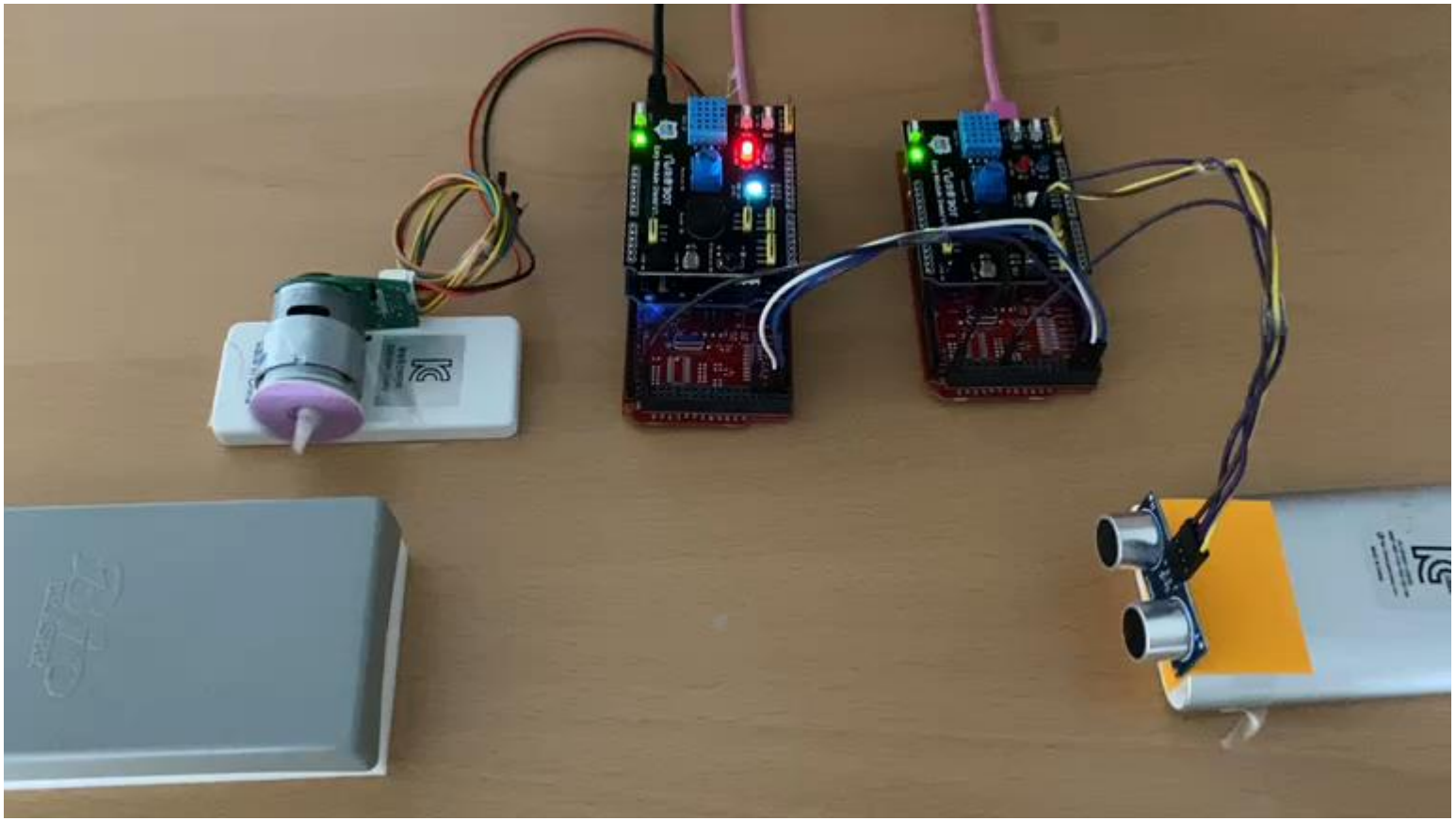
- 영상 3 – [D Gear] Signal Light + Steering



Ⅲ. Development Process

③ Result

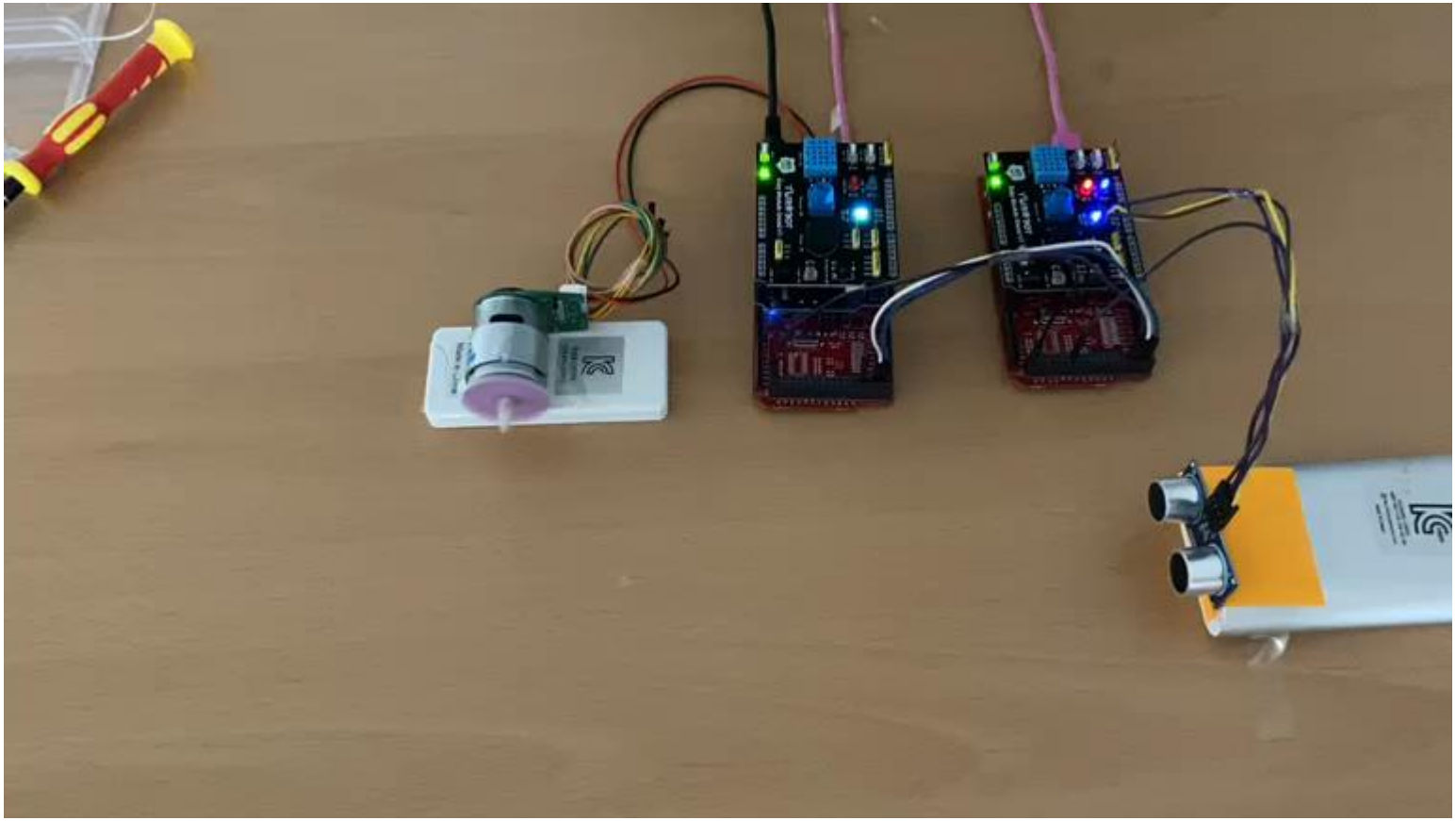
- 영상 4 - [D->R Gear] Gear Shift



Ⅲ. Development Process

③ Result

- 영상 5 – [R Gear] P.C.A



※ Reference

- AURIX TC27x-D User's Manual

- 교수님과 조교님의 ♥