



# F-458

# (자율주행 전동차)

역할 분담 – 개인 자료

# Contents

## 01 H/W PART

- 구동계 재 설계
- BOM
- 시스템 아키텍처
- 배선작업
- ERRATA

## 02 S/W PART

- 업무 관리 및 자료 공유
- TEST 환경
- 주행 시나리오
- Firmware
  - FreeRTOS 이용한 MCU 제어
- ERRATA



# **H/W PART**

# 01 H/W PART

- 구동계 재 설계
- 기존 전동차 스펙은 유아 몸무게를 고려하였기 때문에 보통 성인남성 몸무게(70kg)로 가정 했을 경우 문제 발생 고려 (SERVO, DC모터 고장 및 기구 변형)
- 무게를 견딜 수 있게 구동계를 재 설계 해야함
- 기존 유아 전동차 구동계에서 크게 변경 하지 않고 기존 부품을 사용하도록 함
- 그에 맞는 SERVO 모터와 BLDC 모터로 제품을 선정함
- BLDC 모터와 맞물리는 기어박스는 힘을 많이 받을 것으로 예상 하여 금속 기어로 제작

# 01 H/W PART

- 유아 전동차 선정 및 요구 사항

- 전동차가 구동하기 위해 필요한 토크 산정
- 기존 후륜 좌측만 동작 -> 양쪽 모두 동작으로 구조 변경
- 속도를 표시 할 수 있도록 중공 축 엔코더 추가
- 방향과 동력을 제어 할 수 있도록 Servo모터와 BLDC모터 선정

- 제원



COLOR



고객선호도

DAEHOTOYS

Ferrari – berlinetta F12

₩368,000

6v 저전력 구동 시스템  
운전자 2단계 속도 조절 래버  
자동운전 시스템(리모컨 운전)  
실차느낌의 시동 키  
2세대 소프트 스타터 장착  
오토 브레이킹 시스템  
과류 방지 장치  
충전중 작동 차단장치  
스마트 전원 버튼  
고휘도 LED 헤드라이트, 방향지시 점등  
차체 고품질 ABS재제 사용  
크랙션 기능  
접이식 사이드 밀리터  
안전벨트 적용  
AUX 단자 연결 가능  
(mp3플레이어 및 휴대폰으로 음악 재생 가능)  
페라리사와의 정식 디자인 라이선스를  
통해 제작된 제품

제품크기(mm)	1115*572*460	사용시간	1~2.5
차체중량(kg)	11.3kg	충전시간	8~10
작용모터	후륜,싱글모터	배터리사양	6V 7AH
속도(km/h)	약 3km/h	권장연령	30개월~6세이만

# 01 H/W PART

- 전동차가 구동하기 위해 필요한 토크 산정
  1. 공차 중량 : 11.3kg
  2. 타이어 직경 : 21.5cm
  3. 차량 기어비 : 1/80
  4. 사람(70kg)이 탑승 할 경우 구동하기위한 최소 토크 = [타이어 반경(cm)  
X 차량의 중량(kgf)] / 기어비 =  $(10.75 \times 81.3) / 80 = 10.924\text{kgf.cm}$
- ☞ 구동계 및 기구의 마찰이나 효율, 타이어의 접지력, 노면이 마찰력 등을 고려하여 2배 값을 적용 : **20kgf.cm**
- ☞ Servo 모터의 경우 좌우 주행 시 토크가 더 작용하므로 고려하여 선정

# 01 H/W PART

- 기존 후륜 좌측만 동작 -> 양쪽 모두 동작으로 구조 변경



- 기존 구동계는 싱글 기어박스로 이루어져 있으며 동력이 전달되는 부분은 표시된 한곳 뿐임
- 방향을 제어하는 전륜 구동 축은 바퀴만 회전하는 방식임

# 01 H/W PART



- 양 쪽 바퀴가 같이 회전 할 수 있게 구동 축을 가공
- 구동 축을 지지 할 수 있으며 회전에 무리가 없도록 베어링 장착

# 01 H/W PART

## 후륜구동 축 가공



- 양쪽 바퀴를 구동 축과 고정 시키기 위해 양쪽 끝에 나사산 작업을 함
- 양쪽 바퀴를 구동 축과 함께 회전하기 위해 홀을 뚫어 니들 핀을 넣을 공간을 만듬

# 01 H/W PART

후륜구동 축 가공



# 01 H/W PART

## 후륜구동 축 가공

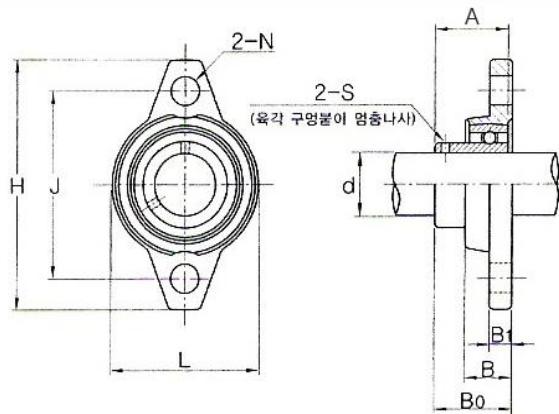


- 구동 축과 바퀴가 물리는 지점에 잘 잡아 주지 못하여 유격이 발생
- 각각 바퀴 바깥쪽과 안쪽에 유격이 발생 하지 않도록 고정 물 삽입함

# 01 H/W PART

## 후륜구동 축 가공

‰ UFL



- 마연특수열처리하여 인장도 강함.
- 좁은 공간에 설치가능
- 정밀하여 고속회전에 사용가능

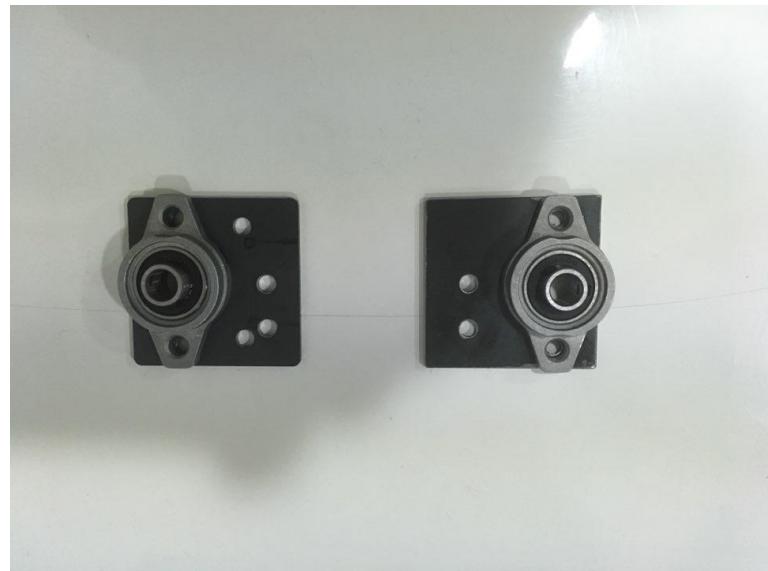
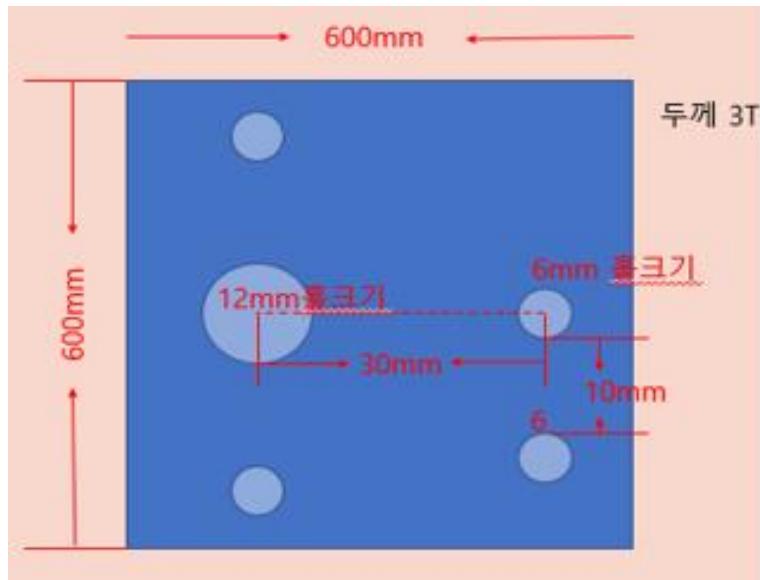
유니트 베이링 주요치수 (mm)

호칭번호 (Bearing Unit No.)	축경 (Bore diameter) mm d	주요치수 (Dimensions) mm								설치볼트 (Bolt used)
		H	J	B1	B	N	L	B0	A	
UFL 000	10	60	45	5,5	11,5	7	36	19	17,5	M6
UFL 001	12	63	48	5,5	11,5	7	38	19	17,5	M6
UFL 002	15	67	53	6,5	13	7	42	20,5	18,5	M6
UFL 003	17	71	56	7	14	7	46	22,5	20,5	M6
UFL 004	20	90	71	8	16	10	55	26,5	24,5	M6
UFL 005	25	95	75	8	16	10	60	27,5	25,5	M8
UFL 006	30	112	85	9	18	13	70	29	26,5	M10

- 후륜구동 축의 회전과 차량 무게를 지지할 수 있는 베어링 선정

# 01 H/W PART

## 후륜구동 축 가공



- 치수를 산정하여 전동차에 장착 할 수 있게 철판을 제작

# 01 H/W PART

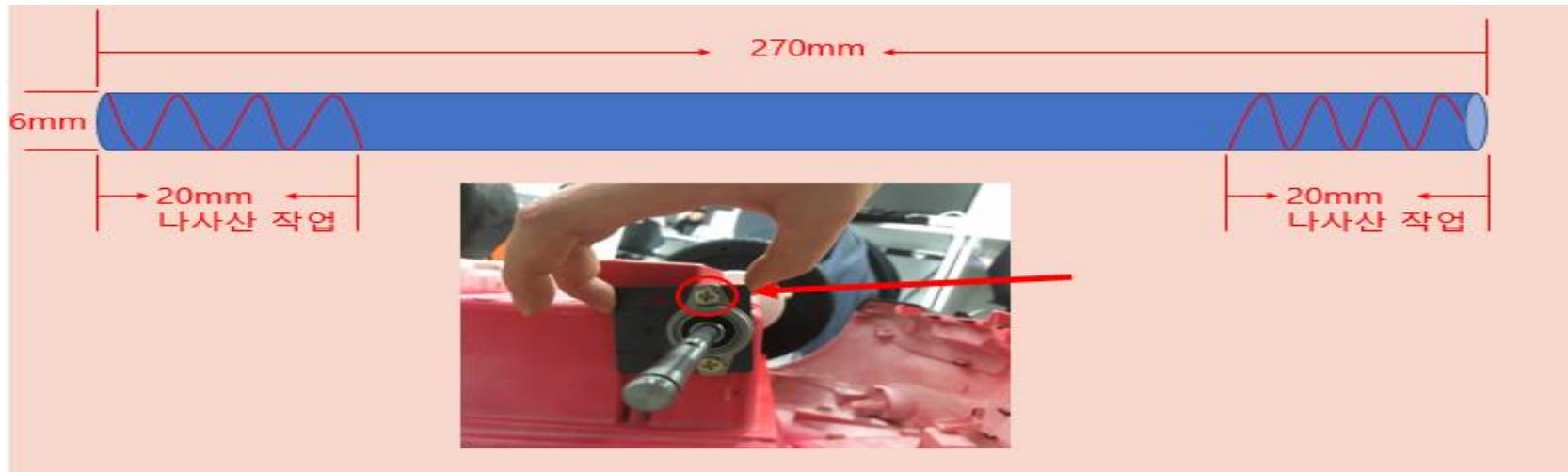
## 후륜구동 축 가공



- 양쪽 기구에 베어링 제작물  
장착

# 01 H/W PART

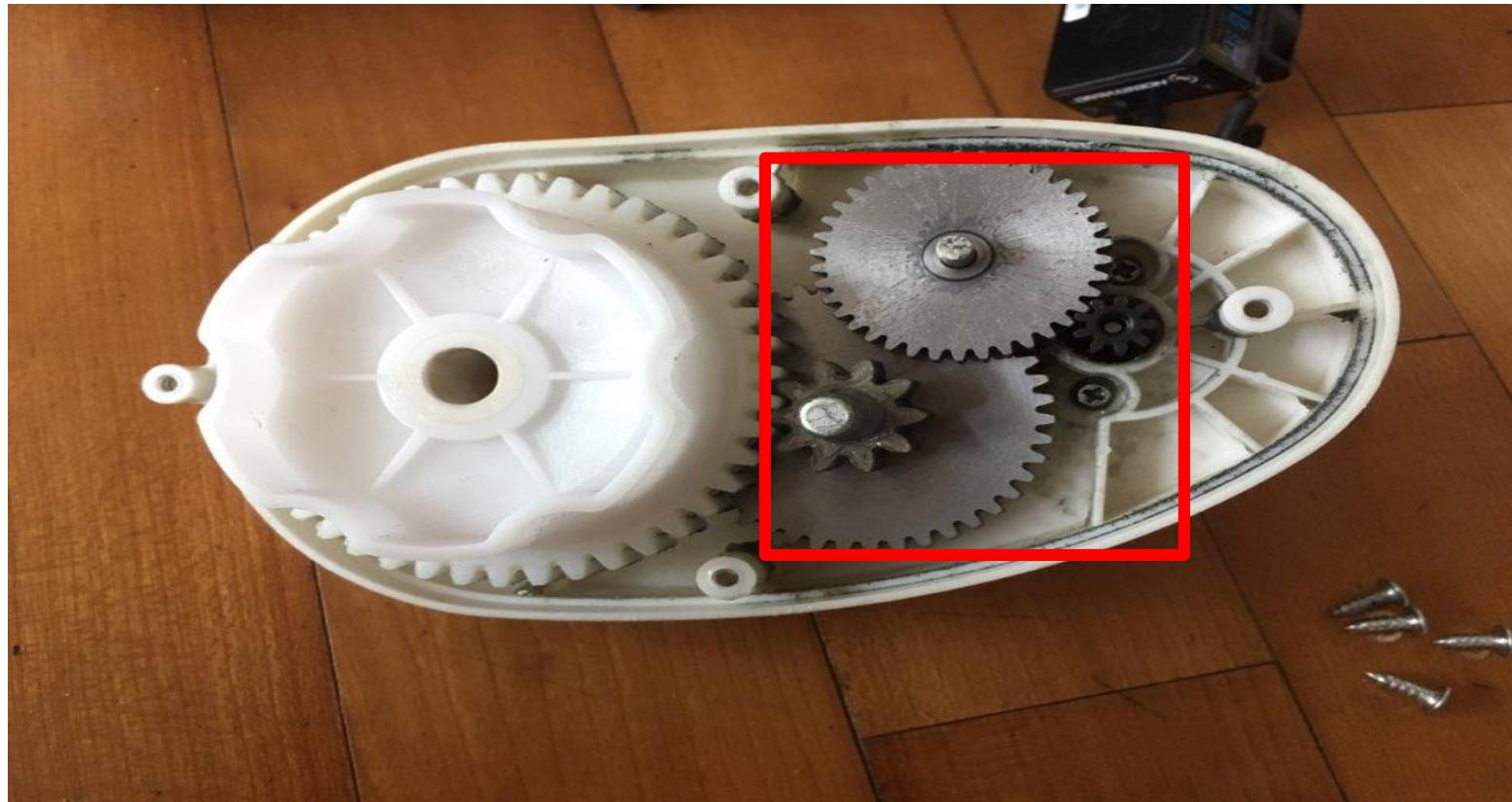
## 후륜구동 축 가공



- 베어링 제작물 비틀림 방지 지지대 제작

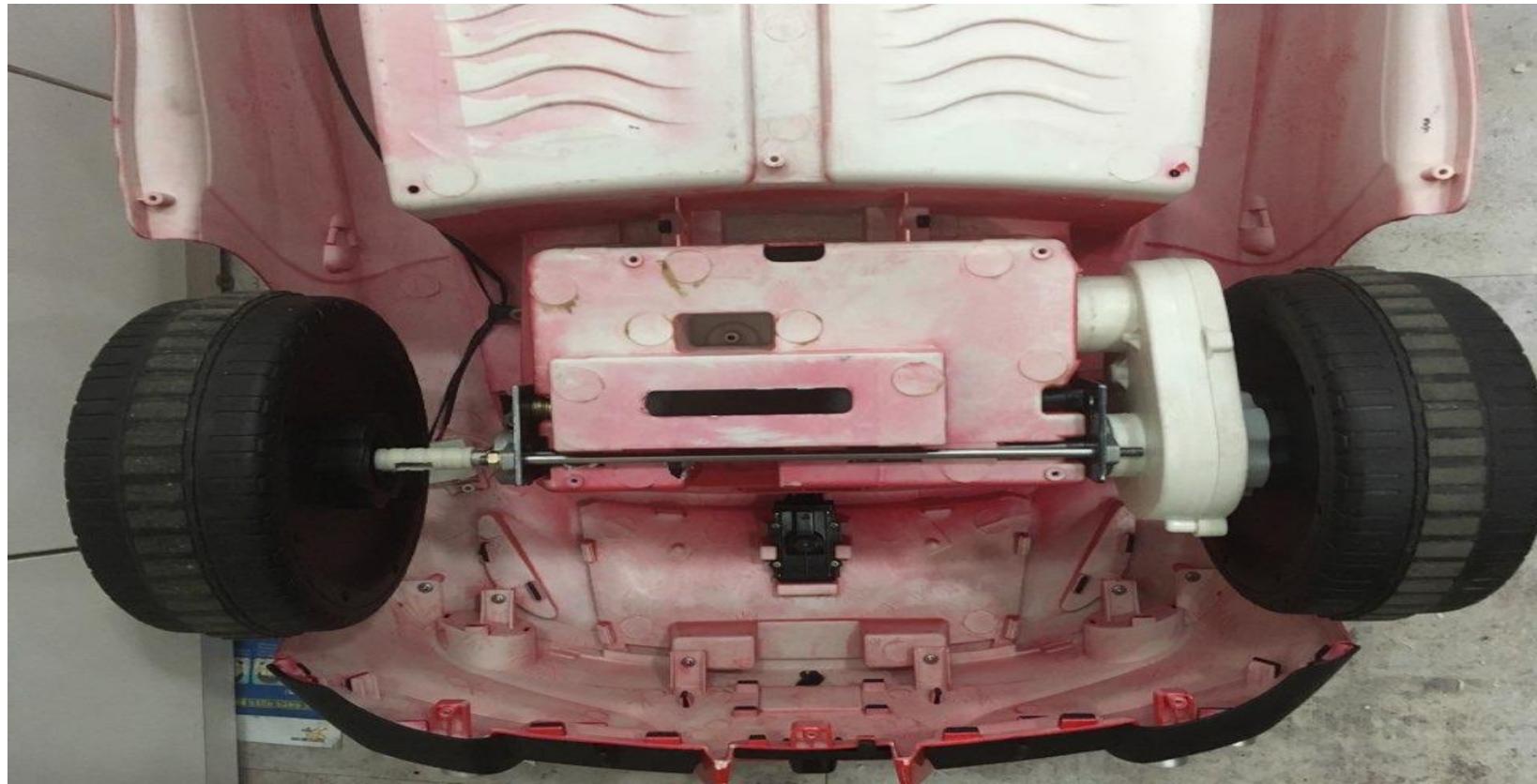
# 01 H/W PART

기어 박스 안에 기어는 높은 토크에 버틸 수 있도록 메탈 재질로 제작



# 01 H/W PART

후륜구동 축 장착 사진



# 01 H/W PART

- 속도를 표시 할 수 있도록 중공 축 엔코더 추가

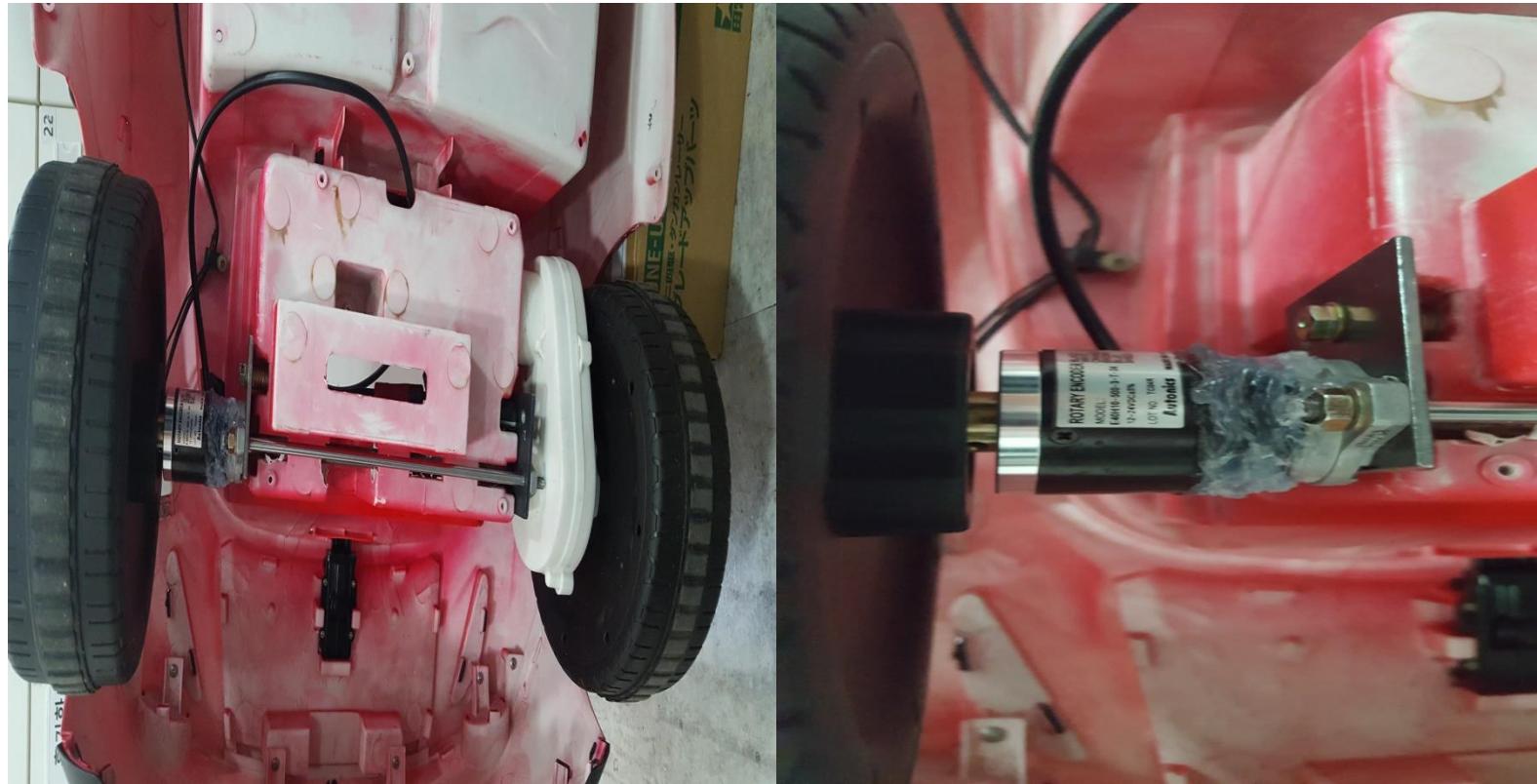


E40H10-500-3-T-24  
Autonics  
↑  
분해능

- 후륜 우측 구동 축에 장착하여 주행 시 속도를 표시하기 위한 용도

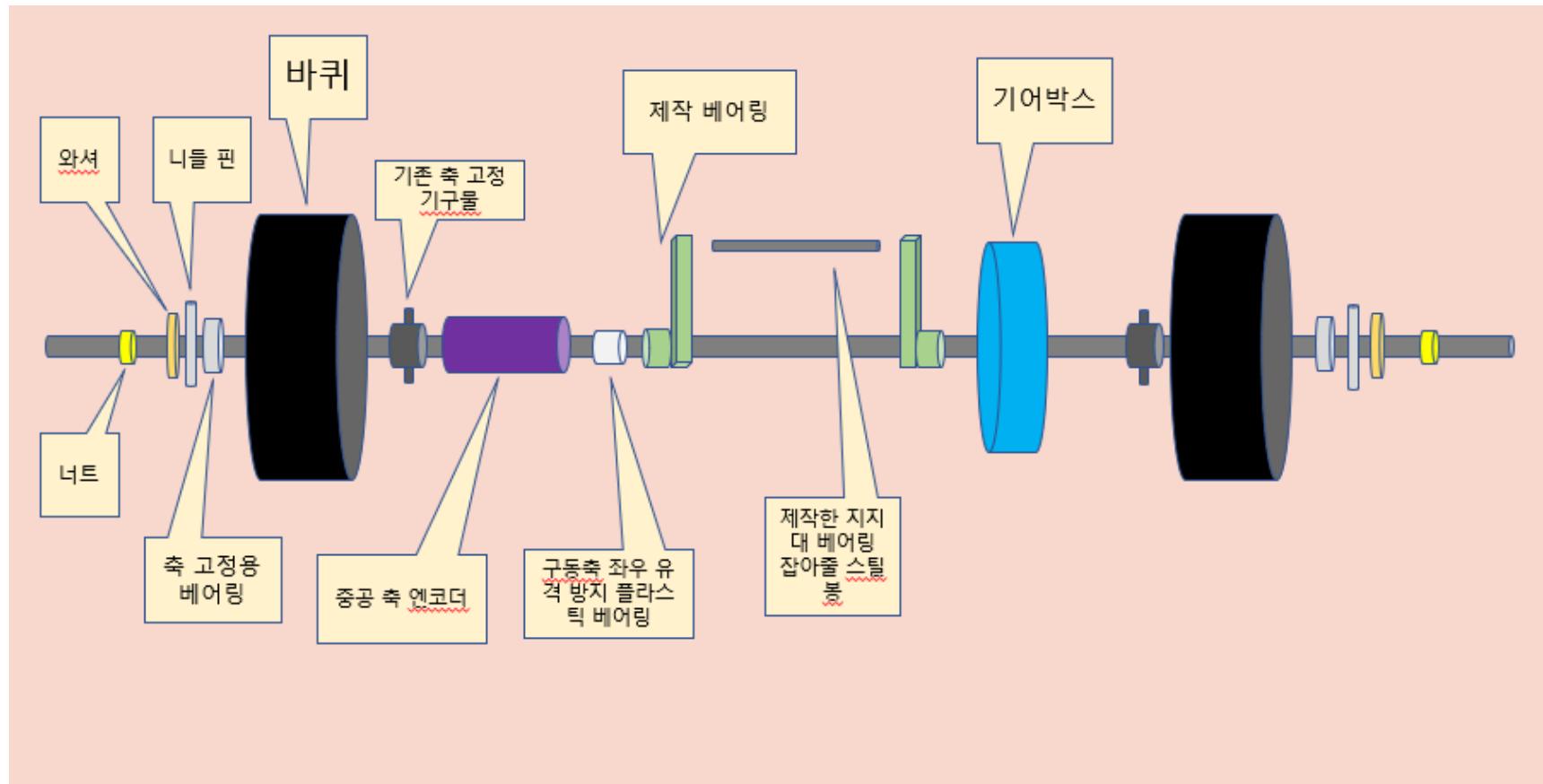
# 01 H/W PART

중공 축 엔코더 장착



# 01 H/W PART

## 구동 축 ASSEMBLY



# 01 H/W PART

- 방향과 동력을 제어 할 수 있도록 Servo 모터와 BLDC모터 선정



- HITEC 社 Servo제품 사용
- 유아 전동차가 구동하기 위한 최소 토크 20kgf.cm보다 큰 제품
- 동작 속도 및 토크 고려하여 선정

**WATER PROOF**  
**IP-67** **32-bit WV HR PRG SG Smart**  
**MEGA SCALE MONSTER TORQUE**

- 4.8V Torque : 32.50kg.cm
- 4.8V Speed : 0.26 sec/60°
- 6.0V Torque : 40.5kg.cm
- 6.0V Speed : 0.21sec/60°
- 7.4V Torque : 50.0kg.cm
- 7.4V Speed : 0.17sec/60°
- Operation Voltage : DC 3.5~8.4V
- Dimensions : 66 x 32 x 62 mm
- Weight : 227g

# 01 H/W PART

- 방향과 동력을 제어 할 수 있도록 Servo 모터와 BLDC모터 선정



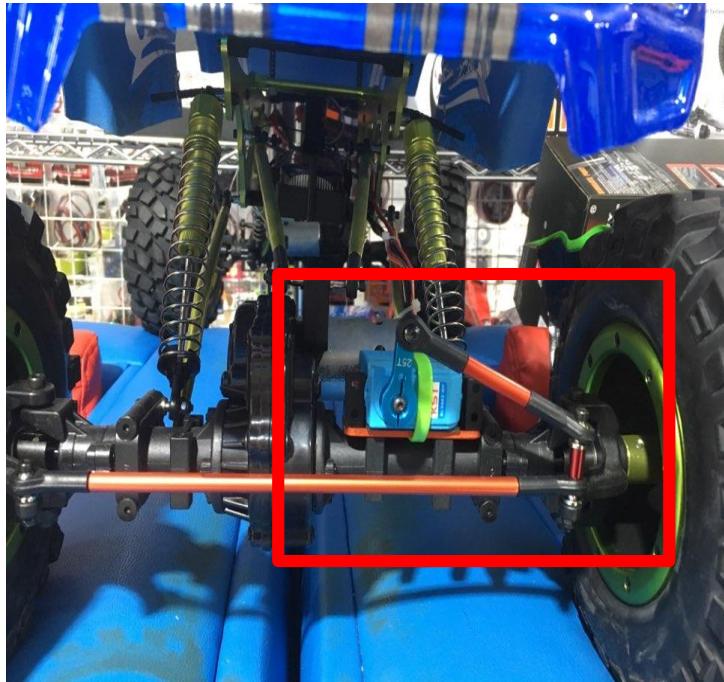
## Specifications:

- **ESC**
  - Scale: 1/8th 1/10th
  - Brushed/Brushless: Brushless
  - Sensored/Sensorless: Sensored
  - Waterproof: No
  - Cont./Peak Current: 140A/880A
  - Input: 2-4S Lipo/6-12 Cell NiMH
  - BEC Output: Switch Mode: 6V/7.4V, 6A
  - Input Wires: Red-12AWG-200mm\*1; Black--12AWG-200mm\*1
  - Output Wires: Blue -12AWG-200mm\*1; Yellow-12AWG-200mm\*1; Orange -12AWG-200mm\*1
  - Input Connectors: No
  - Output Connectors: No
  - Fan Size: 30x30x10mm
  - Fan Voltage Range: 5-7.4V
  - Fan Powered by: Powered by BEC
  - ESC Programming via LCD Program Box: Supported
  - ESC Programming via WiFi Module: Supported \*WiFi Module not included and sold separately: part# [HWA30503000](#)
  - Program Port: FAN/Program Port
  - Firmware Upgrade: Supported
  - Size: 54.1x37.2x36.1mm
  - Weight: 90.5g
- **Motors**
  - Kv (no load): 3100
  - LiPo: 2-3S
  - Resistance: 8.7
  - No-load Current: 2.9A
  - Motor Diameter: 36mm (1.417in)
  - Motor Length: 52.5mm (2.067in)
  - Shaft Diameter: 5mm (0.197in)
  - Shaft Length: 15mm (0.591in)
  - Poles: 4
  - Weight: 192g (6.60oz)
  - Applications: 1/10 light duty SCT, truck, monster truck

- HOBBYWING 社의 ESC+BLDC COMBO 제품 사용(3100kv , 3s배터리 사용)
  - $RPM = kv \times V = 3100 \times 11.1$ (배터리 전압) = 34410
  - kW(모터가 먹는 전력) = {A(모터의 최대 Amps) x V(사용할 배터리 전압)} / 1000(단위 맞춤) =  $(880 \times 11.1) / 1000 = 9.768\text{kW}$
  - 모터의 최대 Toque(kgf.cm) =  $(97400 \times \text{kW}) / RPM = (97400 \times 9.768\text{kW}) / 34410 = 27.6490\text{kgf.cm}$
  - 97400은 kW와 중력가속도를 kgf.cm으로 환산 할 때 사용되는 상수
  - 차량의 속도(km/h) =  $\{(rpm / (\text{기어비})) \times \text{바퀴 직경} \times 3.14\} / 1000 \times 60 = \{(34410 / 80) \times 0.215(\text{M로 환산}) \times 3.14\} / 1000 \times 60 = \text{대략 } 17.4226\text{ km/h}$

# 01 H/W PART

기구물에 장착



- 기존 RC카에서 사용하고 있는 조향 방식을 적용

# 01 H/W PART

장착 완료 전체 사진



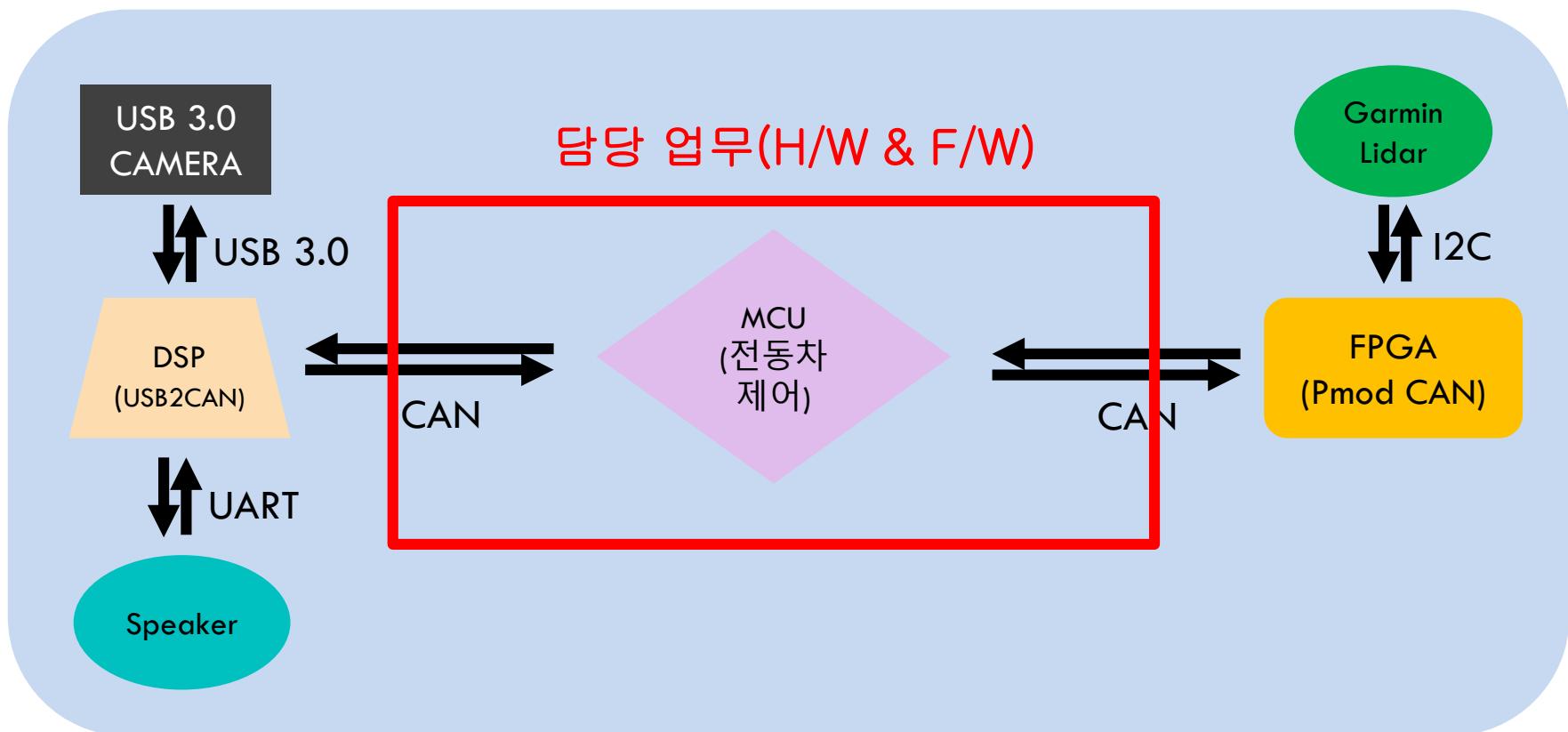
# 01 H/W PART

- BOM

No	구분	Part No	Description	Qty	Manufacture	단가(원)	날기	비고
1	MCU	TMDX570LC43HDK	Hercules Development Kit	1	TI	240,322		
2	FPGA	ZYBO Z7-10	Zynq-7000 ARM/FPGA SoC Development Board	1	Digilent	254,000		
3	DSP	TMDSEVM572X	AM572x Evaluation Module	1	TI	706,950		
4	Pmod	Pmod CAN module	SPI2CAN module	1	Digilent	28,300		
5	유아 전동차	파라미 F12 바디 키트	6V 1모터 구동	1	SMG	25,000		
6	GEAR	metal gear 1	기어박스 기어1(10:38)	1	태초 토이즈	70,000		
7	GEAR	ECX232028	피니언기어 MOD1 9T	1	ECX	14,800		
8	CAMERA	ELP-SUSB1080P01-L60	USB 3.0 CAMERA	1	ELP	65,852		
9	CANmodule	MW USB2CAN(VCP) v2	Can통신 모듈	1	NTRexLAB	64,680		
10	GEAR	metal gear 2	기어박스 기어2(10:43)	1	신진정밀	70,000		
11	BEARING	NSK ZZ 6000	구동축 고정 베어링	4		12,000		
12	BEARING	UFL200 + 지지대	우측 좌측 구동축 지지대 베어링	2	신진정밀			
13	shaft	body shaft	휠바퀴 구동축, 10파이 500mm	1	신진정밀			
14	mount	mount shaft	제작 베어링 고정용, 6파이, 270mm	1	신진정밀			
15	MOTOR+ESC	Xerun 3652 sensored brushless motor + Xerun XR8 SCT	3S, 3100kv, sensored 140/880A	1	hobbywing	204,000		
16	Servo Motor	HS-D845WP	7.4V Torque(50kg.cm), 0.17sec/60	1	HITEC	115,000		
17	Servo Mount	98022N	Servo Motor 지지대	1	HSP	11,620		
18	Screw	98052	xScrew 3*18	4	HSP	3,500		
19	Nuts	UR165300	M3 Nylon Locknuts	4	ULTIMATE RACING	4,800		
20	LINK	98010	Steering link(servo motor와 조향 지지대 연결용)	1	HSP	13,800		
21	Battery	EP Power Series 7600mAh	3S, 7600mAh, 60C, 모터용	1	EP POWER	69,000		
22	Encoder	E40H10-500-3T24	12-24V, 500분회전, 10파이	1	오토닉스	73,300		
23	Charger Cable	UP-XT150CH-T1	리튬 배터리 충전용 케이블	1	UP-KOREA	9,000		
24	Connector	XT150-R	배터리, 배 속기, 전원 공급용 커넥터	2	DRC	7,000		
25	Connector	XT150-BK	배터리, 배 속기, 전원 공급용 커넥터	2	DRC	7,000		
26	Connector	XT150/A5150	XT150 PARALLEL Y CABLE RED & BLACK	1	hobbywing, king	12,900		
27	DC converter	MP1584EN	감하형 DC-DC 소형 가변 컨버터	2	SMG	4,000		
28	DC converter	LM2598	감하형 DC-DC 가변 컨버터	1	SMG	4,500		
29	TR	2N2222A	NPN TR	2	-	1000		
30	MOSFET	IRFZ44N	N CH MOSFET	4	infineon	1800		
31	MOSFET DRIV	TC426	MOSFET DRIVER	2	microchip	1610		
32	RESISTOR	10KΩ	1/4W	4	-	40		
33	RESISTOR	15Ω	1/4W	2	-	20		
34	RESISTOR	20Ω	1/4W	2	-	20		
35	LED	Luxeon Star/C LED	POWER LED( LED, WHITE, ORANGE)	6	Philips	1800		
36	RF Module	T8FB/R8EF	8CH RF 송신기/수신기	1	RADIOLINK	80000		
37	Battery	[MOS] 22.2V 6000mAh 6s lipo	6S, 6000mAh(증고)	1	MOSWORTH	70000		
38	Battery	14.8V 7300mAH 4s lipo	4S, 7300mAh	1		88000		
					total	2,446,314		
기타부속		볼트 M6 3종, 너트 3종, 오션 3종, 니들비, 구동축 베어링 4ea						

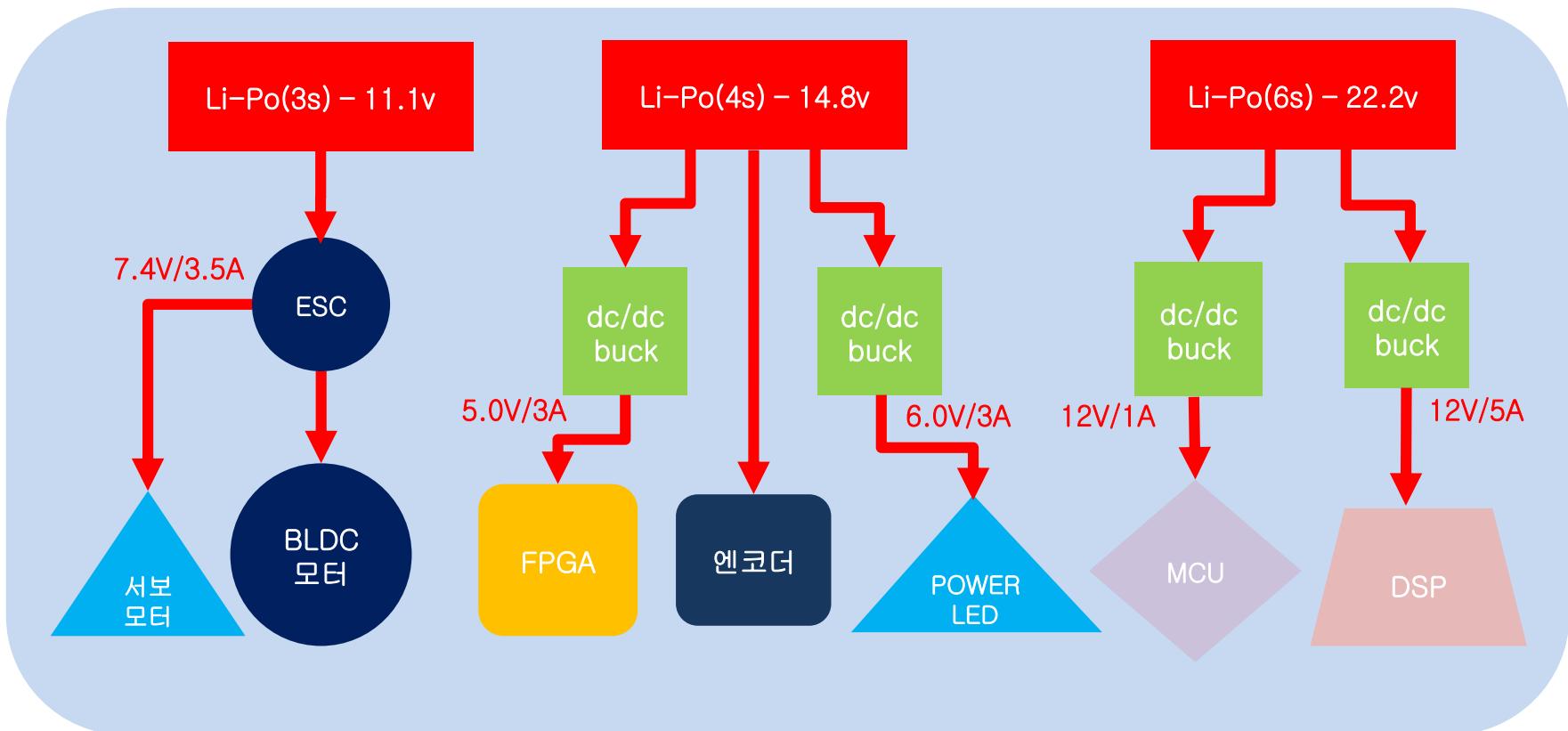
# 01 H/W PART

- 시스템 아키텍처
  - 통신 네트워크



# 01 H/W PART

- 배선 작업
  - 전원부 배선도



# 01 H/W PART

- 배선 작업

- SIGNAL LINE 배선

Device	↔	MCU [FOOT PRINT]
SERVO MOTOR – PWM SIGNAL		M1
BLDC MOTOR – PWM SIGNAL		B5
USB to CAN Module (DSP) – H,L,GND		CAN1
PmodCAN Module(FPGA) – H,L,GND		CAN2

# 01 H/W PART

POWER LED(W, O, R) – LED1,2,3,4,		A5, C2, E1, C1
ENCODER – A, B		U1, V2
RF RECIVER – ch3,4,5,6,7,8		W8, V8, G19, H18, N1, R2

# 01 H/W PART

- 배선 작업
  - 디바이스 장치 간 연결



# 01 H/W PART

- 배선 작업
  - 디바이스 장치 간 연결



# 01 H/W PART

## • ERRATA

- 제조사마다 유아 전동차 구동 축이 상이하여 개조 할 수 있는 기구를 찾느라 3대나 구입하게 됨. 비용 낭비 발생



- 베어링을 장착 할 수 없는 구조

# 01 H/W PART

- ERRATA



- 베어링을 장착 할 수 없는 구조

# 01 H/W PART

## • ERRATA

### 2. Servo 모터 선정하기 전 테스트 용도로 구매한 모터 고장 발생

- 전동차에 장착하여 사람이 탑승한 상태에서 좌 우로 만 방향 전환만 시행



- 기구적인 오차로 인한 토크 량 부족으로 고장 발생
- 더 큰 토크 량을 가진 제품으로 교체

Torque(6.0V): 23.0 kg-cm (319.4 oz/in)  
Torque(7.4V): 25.0 kg-cm (347.2 oz/in)  
Speed: 0.16 sec (6.0V) | 0.14 sec (7.4V)  
Operating Voltage : 6.0 ~ 8.4 DC Volts  
Weight: 72 g (2.54 oz)  
Bearing Type : Ball Bearing x 2  
Motor Type : DC Motor  
Gear Type : Copper & Aluminum  
Operating Temperature: -20°C ~ 60°C  
Size : 40.7 x 20.5 x 38.6 mm ( 1.60 x 0.81 x 1.52 in)

# 01 H/W PART

## • ERRATA

### 3. 기구 물 구조 문제로 사람을 탑승 시키지 못하게 됨

- 성인이 탑승하기에 전동차 기구물이 작음
- 기어박스 규격에 적합한 BLDC 모터 규격을 확인 해보니 사람을 탑승 시킬 만한 토크를 가진 모터를 구하기가 어려움



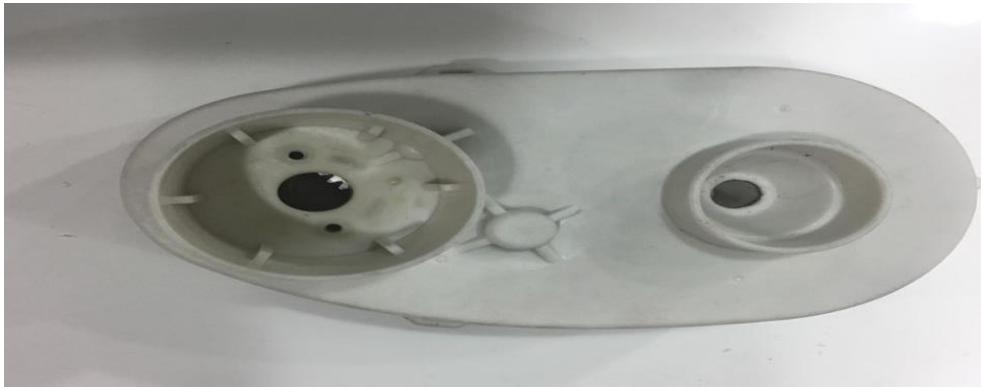
#### Specifications:

- **ESC:**
  - **Current:**
    - **Continuous:** 150A
    - **Peak:** 950A
  - **Motor Type:** Sensored/Sensorless Brushless Motor
  - **Applications:** 1/8 Scale On Road/Off Road, Competition Level
  - **Motor Limit:**
    - 4S LiPo/12 Cell NiMH: 3000Kv for 4274 size or smaller motor
    - 6S LiPo/18 Cell NiMH: 2400Kv for 4274 size or smaller motor
  - **LiPo/NiMH Cells:** 2-6S LiPo, 6-18 Cell NiMH
  - **BEC Output:** 6V/7.2V Switchable, Continuous Current of 6A (Switch Mode BEC)
  - **Cooling Fan:** Powered by BEC Output Voltage
  - **Connectors:** Open Post, must be soldered by the user
  - **Size (LxWxH):** 58.7x48x36.9mm
  - **Weight:** 127g (no wires)
  - **Programming Port:** FAN/PRG Port
- **Motor:**
  - **KV (No-load):** 2250
  - **Current No-load (A):** 5.4
  - **Motor Diameter x Length ( mm ) :** 42x74  
Shaft Diameter x Length ( mm ) : 5x10.5
  - **LiPo (S):** 2S-6S
  - **W (g):** 396g
  - **Application:** 1/8 Truggy, Monster Truck
  - **Sensored:** Yes

- 기존에 적용하려던 큰 토크를 가진 모터의 둘레 치수가 기어박스와 호환이 되지 않아 사용 할 수 없음
- 기어박스와 호환 되는 사이즈의 BLDC 모터로 교체

# 01 H/W PART

- ERRATA



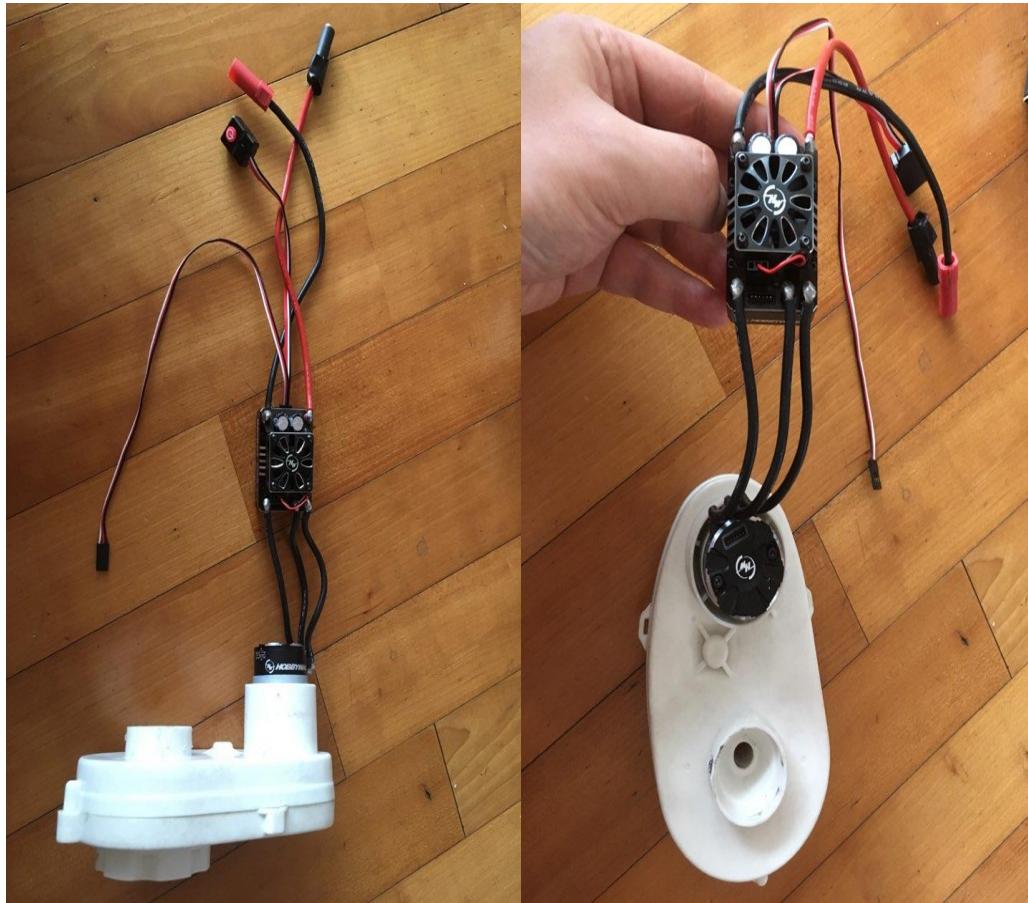
외부치수



- 기어박스 와 순정 DC모터  
치수

# 01 H/W PART

- ERRATA



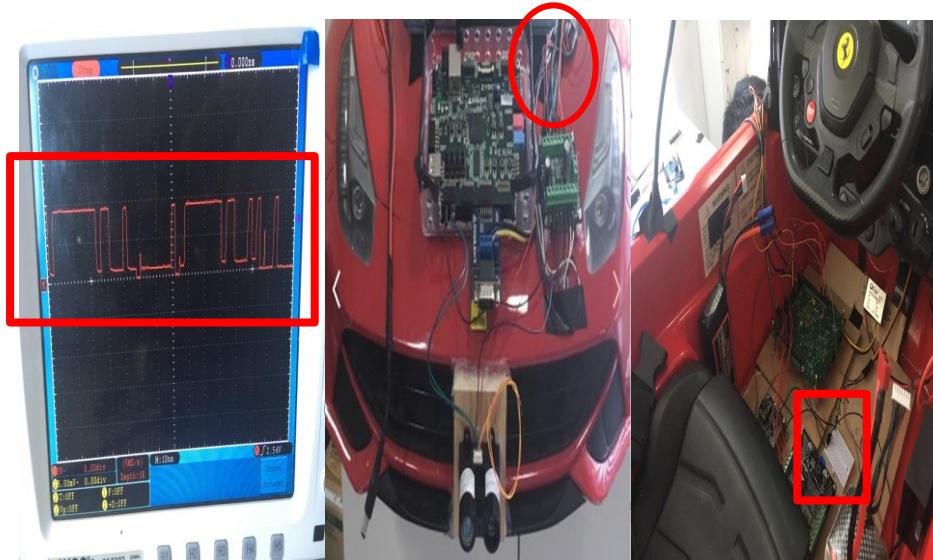
- 기어박스 규격에 맞는  
BLDC 모터 장착

# 01 H/W PART

## • ERRATA

### 4. 전원 신호 그라운드 분리

- 배선을 분리하지 않고 묶어서 작업
- 공통 그라운드를 브레드 보드로 연결함(그라운드 처리 문제)
- POWER 라인이 길어지다 보니 L성분으로 인한 역기전력이 발생 하여 기기 오작동을 일으킴



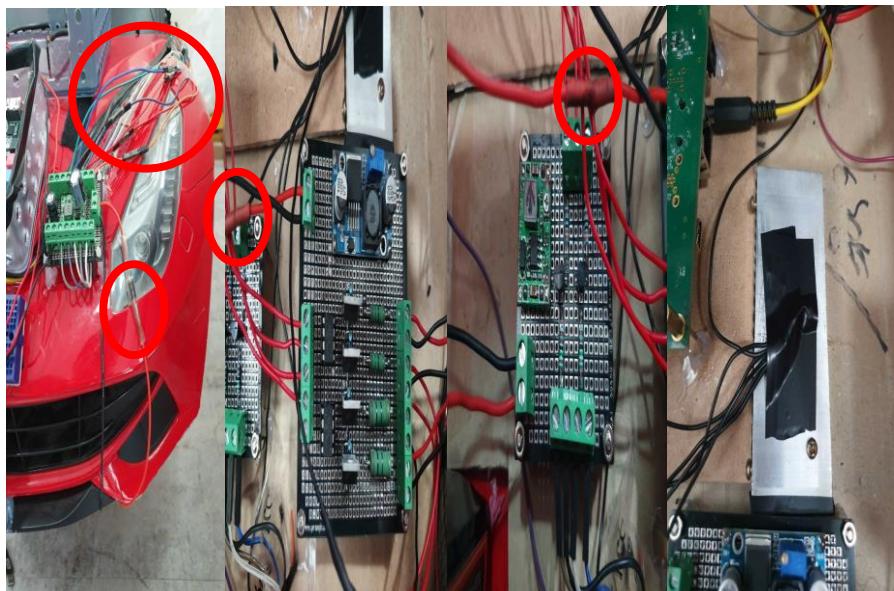
- FPGA↔MCU 간 CAN 통신 신호 라인 (노이즈 및 GND 레벨이 떠 있음 다른 신호들도 노이즈에 영향을 받을 것으로 추측)
- 신호, POWER, GND 라인을 겹쳐서 배선함 신호라인에 EMC 간섭이 있을 것으로 추측
- 공통 그라운드 라인이 좁음

# 01 H/W PART

## • ERRATA

### 4. 전원 신호 그라운드 분리

- 배선을 분리하지 않고 묶어서 작업
- 공통 그라운드를 브레드 보드로 연결함(그라운드 처리 문제)
- POWER 라인이 길어지다 보니 L성분으로 인한 역기전력이 발생 하여 기기 오작동을 일으킴



- 공통 그라운드는 철판으로 작업
- 각각 입력단에 역기전력 방지용 다이오드 작업
- 배선은 신호, POWER, GND 라인 분리 하여 작업



**S/W PART**



# 02 S/W PART

## • 업무 관리 및 자료 공유

- SLACK, TRELLO, GITHUB 를 활용하여 프로젝트 관련 문서 및 일정 공유

The screenshot shows a Slack interface for a channel named '#f458'. At the top, a message says "Slack needs your permission to enable desktop notifications." The channel has 89 unread messages and 0 notifications. There are buttons to "Add a topic" and "Search". The date is Tuesday, July 9th.

The left sidebar shows the user's profile (KOITT3) and a list of channels: # general, # github\_log, # random, # trello\_log, + Add a channel, Direct Messages, and a list of direct message recipients: Slackbot, HyeonseungKim (you), daeroro, Donghyeok Kim, gccompli3r, Godric Hong, Jangho Lee, KangMinSung, Kim99707, and Sungmin Yang.

The main channel area displays a waveform on an oscilloscope screen. A message from user gccompli3r at 6:10 PM asks for confirmation of signal levels. Below it, a link to a question on Electrical Engineering Stack Exchange about modeling long wires in circuits is shared.

Another message from gccompli3r at 6:10 PM asks about the proper way to model really long wires in circuits (18ga, ~1500 ft). The user is aware that over long lengths start to accrue non-negligible resistances and is wondering if they start to get other instances.

A message from pololu.com at 6:10 PM discusses understanding destructive LC voltage spikes, linking to a page on Pololu's website.

At the bottom, there is a message input field with the placeholder "Message #f458".

- SLACK(OS영향을 받지 않음-windows, ubuntu, mobile)을 이용하여 팀원 간에 간단한 파일 및 자료 공유

# 02 S/W PART

## • 업무 관리 및 자료 공유

- SLACK, TRELLO, GITHUB 를 활용하여 프로젝트 관련 문서 및 일정 공유

The screenshot shows a Trello board titled "Integrated Project". The left sidebar lists boards: "F458", "etc", and "TODO". The "TODO" board has five columns: "3rd\_log 1st\_week(19.05.27-05.31)", "3rd\_log 2st\_week(19.06.03-06.05)", "3rd\_log 3st\_week(19.06.10-06.14)", "3rd\_log 4st\_week(19.06.17-06.21)", and "3rd\_log 5st\_week(19.06.24-06.28)". Each column contains a card with a checklist of tasks. A modal window is open on the right, showing options like Attachment, Power-Ups, Actions (Move, Copy, Watch, Archive, Share), About This Board, Change Background, Search Cards, Stickers, More, Power-Ups, and Activity. The activity log shows a task being completed by Hyeonseung KIM.

- TRELLO을 이용하여 프로젝트 일정 및 개발 진행 상황 팀원 공유

# 02 S/W PART

## • 업무 관리 및 자료 공유

### - SLACK, TRELLO, GITHUB 를 활용하여 프로젝트 관련 문서 및 일정 공유

The screenshot shows a GitHub repository page for 'koittintegration3 / IntegrationProject'. The repository has a single branch named 'master' containing one commit from 'silenc3502' titled 'Merge pull request #151 from KimHS87/master'. Below the commit, there is a list of files and their descriptions. A terminal window is overlaid on the right side of the screen, showing a series of commands and their outputs related to the project's build and documentation process.

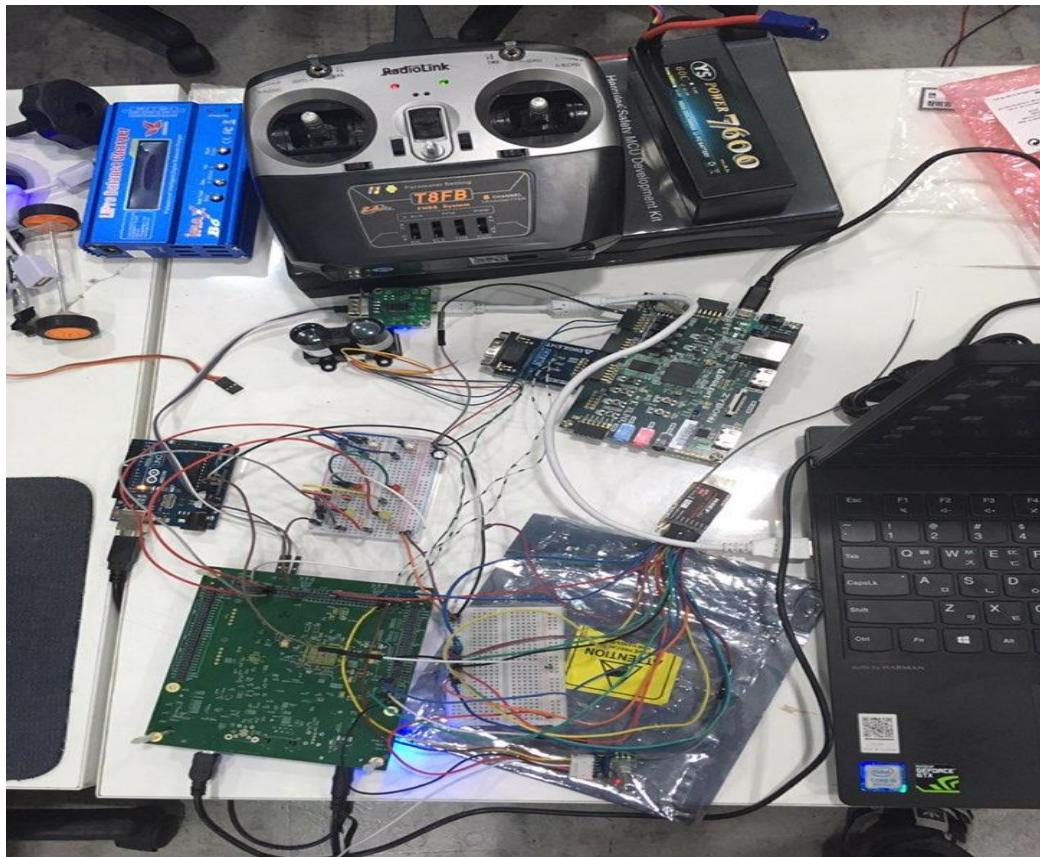
```
khs@khs-Lenovo-Legion-Y530-15ICH:~/gitfile2$ cd IntegrationProject/  
khs@khs-Lenovo-Legion-Y530-15ICH:~/gitfile2/IntegrationProject$ ls  
APST LICENSE asm member mixergi  
F458 README.md coffee_machine mercenary solenoid_braille  
khs@khs-Lenovo-Legion-Y530-15ICH:~/gitfile2/IntegrationProject$ cd F4  
58/  
khs@khs-Lenovo-Legion-Y530-15ICH:~/gitfile2/IntegrationProject/F458$  
ls  
1st_report_HW add FreeRTOS Testcode - 현승  
F458_자료모음 F458_자료모음 - 현승  
dsp Merge pull request #138 from Sunggyu...  
fpga final fpga code  
mcu pdf/ppt update  
others UFL000_유니트테어링  
2nd_week_F458_presentation.pptx 2nd_week_F458_presentation.pptx  
3rd_week_F458_presentation.pptx 3rd_week_presentation.pptx  
3차_1st_week_F458_presentation.pptx 3차_1st_week_F458_presentation.pptx f458_final.pptx  
3차_2nd_week_F458_presentation.pptx f458_fpga  
3차_3rd_week_F458_presentation.pptx mcu  
4th_week_F458_presentation.pptx others  
5th_week_F458_presentation.pptx  
khs@khs-Lenovo-Legion-Y530-15ICH:~/gitfile2/IntegrationProject/F458$  
khs@khs-Lenovo-Legion-Y530-15ICH:~/gitfile2/IntegrationProject/F458$  
git fetch upstream  
khs@khs-Lenovo-Legion-Y530-15ICH:~/gitfile2/IntegrationProject/F458$  
git merge upstream/master  
Already up-to-date.  
khs@khs-Lenovo-Legion-Y530-15ICH:~/gitfile2/IntegrationProject/F458$
```

This is Integration Repo of F458 Team

- GITHUB를 이용하여 대용량 데이터 공유(프로젝트 코드 및 결과물 공유)

# 02 S/W PART

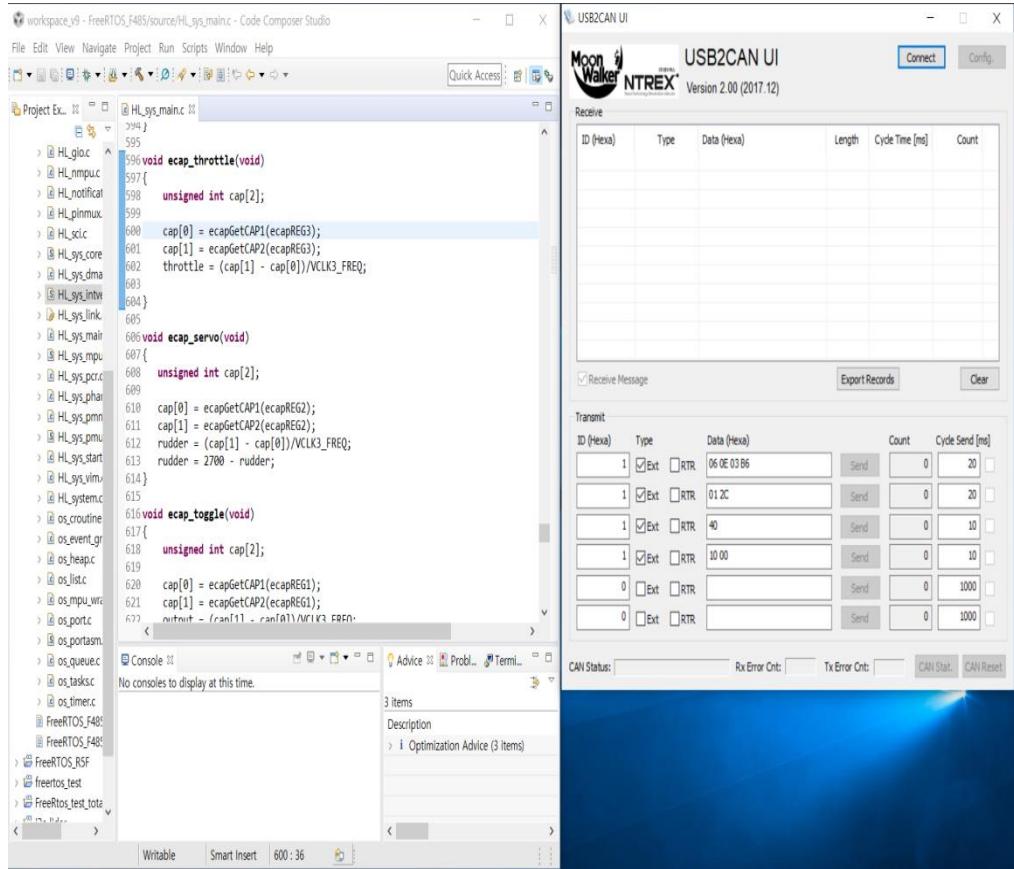
- TEST 환경[H/W]
  - 실제 전동차를 제어한다고 가정한 환경 구성



- PC(DSP대체)
- TMDX570LC43HDK
- RF 송수신기
- GARMIN LIDAR
- USB to CAN 통신 모듈
- FPGA BORAD
- PmodCAN
- ENCODER
- LED

# 02 S/W PART

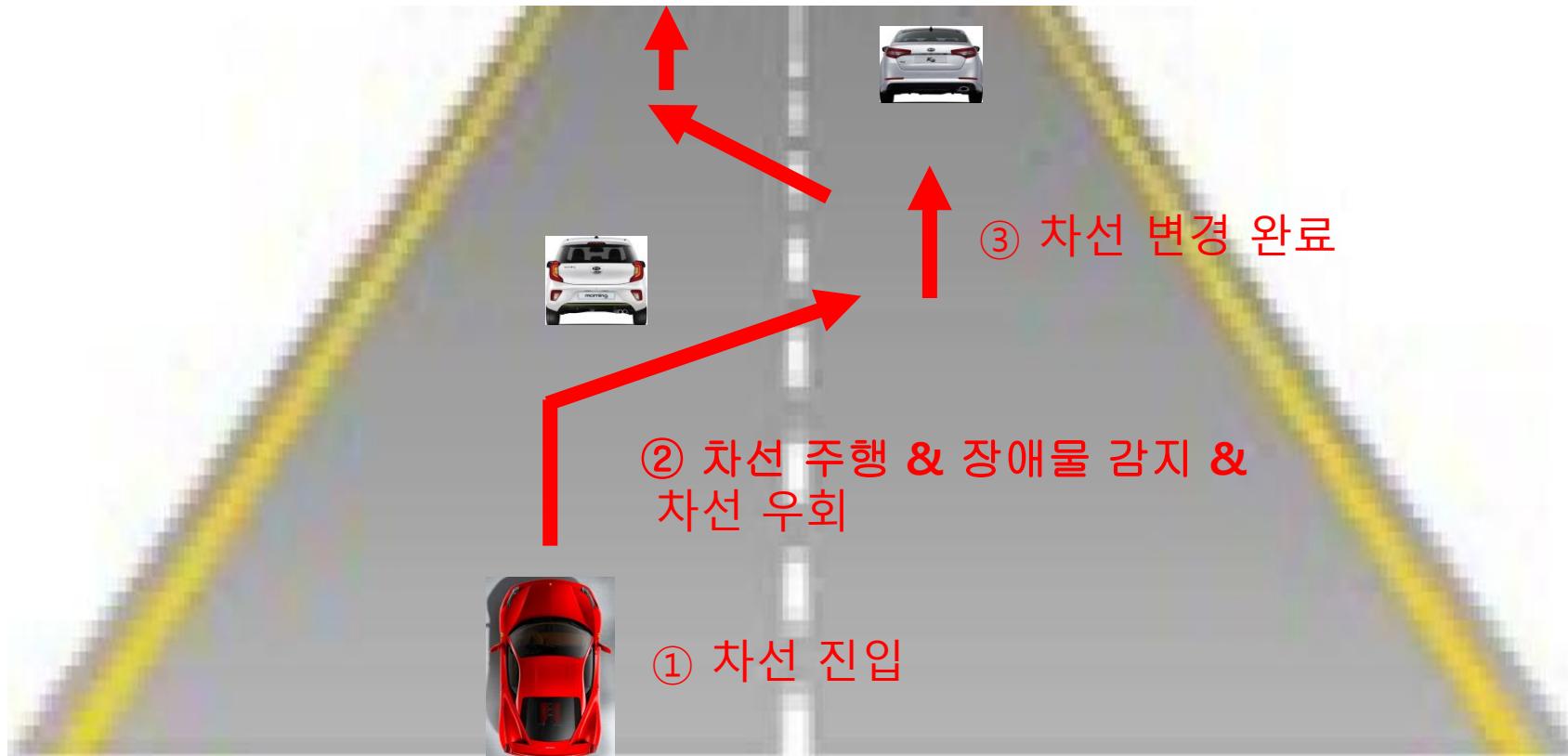
## • TEST 환경[S/W]



- WINDOW 용 CCS  
(우분투에서는 USBtoCAN 모듈 S/W 지원 하지 않음)
- MinGW(window에서 gcc, g++을 사용하는 C/C++ 개발환경 만들기 위함)
- USBtoCAN 전용 프로그램

# 02 S/W PART

- 시나리오
  - 주행 시나리오 1
- 차선 우회 주행



# 02 S/W PART

- 시나리오
  - 주행 시나리오 1
    1. 차선 진입
      - 1) 전조등 ON
      - 2) RF 송수신기를 이용(조종)하여 차선으로 진입하여 정지한다.
      - 3) FPGA-MCU간에 FPGA로 부터 Garmin lidar의 data 값을 CAN통신으로 항시 받는다.
      - 4) 전방 카메라가 차선을 인식하게 되면 우회 할 옆 차선의 throttle , rudder 데이터 값을 DSP-MCU 간에 CAN 통신으로 차선 변경 할 때 까지 항시 받는다.

★ 현재 TASK 상태 : Task0,1,2,4,5,6 모두 동작. 실제 주행 시 Task3 삭제 예정  
(디버깅용)

# 02 S/W PART

- 시나리오
    - 주행 시나리오 1
2. 차선 주행 & 장애물 감지 & 차선 우회
- 1) RF 송신기의 toggle switch(ch7)를 이용하여 High 신호로 변경하여 앞으로 전진한다.  
★ 현재 TASK 상태 : Task5의 toggle switch를 High로 변경함으로써 Task1(throttle), 2(rudder) 중지. 고정 값인 throttle(1550) , rudder(1250) 적용되어 천천히 전진 주행을 함.
  - 2) 전진 중에 전방에 장애물이 나타나 2M 범위 이내로 좁혀지면(garmin lidar data) 방향을 틀어 우회하기 시작함.
  - 3) Task5가 중지 됨에 따라 고정 값 throttle(1550) , rudder(1250) 사라지고 Task0(DSP-MCU)에서 CAN 통신으로 변수(dsp\_throttle , dsp\_rudder)에 저장했던 DSP의 우회 값을 적용 하여 좌측, 또는 우측으로 이동.
  - 4) 방향에 따라 좌, 우 방향 지시 등 ON.(수동)
- ★ 현재 TASK 상태 : Task 1(throttle),2(rudder),5(toggle) 중지

# 02 S/W PART

- 시나리오

- 주행 시나리오 1

- 3. 차선 변경 완료

- 1) 우회 함으로써 lidar의 장애물 인식 거리를 벗어나게 됨.

★ 현재 TASK 상태 : Task1(bldc),2(servo) 해제. Flag 변수가 바뀌면서 Task5(toggle) 도 해제.

- 2) RF 송수신기를 이용(조종)하여 우회한 차선에 일직 선상으로 진입 후 대기.(후미등 ON)

★ 현재 TASK 상태 : Task5(toggle) 해제 되면서 기존에 고정 값 throttle(1550) , rudder(1250) 에 의해 순간적으로 적용됨(방향을 꺾은 상태에서 직진함). 이때 RF 송신기의 toggle switch를 Low로 변경해야 함.

Task1,2 도 해제 되었기 때문에 RF 송수신기로 조종 하여 우회한 차선에서 일직선상으로 자리위치 시킴.

→ 2~3 시나리오를 반복 하면 앞선 장애물이 출현 할 때마다 우회하여 차선 변경 후 주행이 가능함.

# 02 S/W PART

- 시나리오
  - 주행 시나리오 2  
장애물 출현



# 02 S/W PART

- 시나리오
    - 주행 시나리오 2
1. 차선 진입
    - 1) 전조등 ON
    - 2) RF 송수신기를 이용(조종)하여 차선으로 진입하여 정지한다.
    - 3) FPGA-MCU간에 FPGA로 부터 Garmin lidar의 data 값을 CAN통신으로 항시 받는다.
    - 4) 전방 카메라가 차선을 인식하게 되면 우회 할 옆 차선의 throttle , rudder 데이터 값을 DSP-MCU 간에 CAN 통신으로 차선 변경 할 때 까지 항시 받는다.
  - ★ 현재 TASK 상태 : Task0,1,2,4,5,6 모두 동작.

# 02 S/W PART

- 시나리오
    - 주행 시나리오 2
2. 직진주행
- 1) RF 송수신기의 throttle을 동작(pwm 1650이상)시켜 전진 고속 주행을 한다.
- ★ 현재 TASK 상태 : Task 0,1,2,4,5,6 모두 동작



# 02 S/W PART

- 시나리오
  - 주행 시나리오 2

## 3. 급정지

1) RF 송수신기의 throttle을 동작(pwm 1650이상)시켜 전진 고속 주행을 한다.

★ 현재 TASK 상태 : Task 0,1,2,4,5,6 모두 동작

2) 주행 중에 전방에 장애물이 출현하여 Garmin Lidar 측정 범위(2M) 이내로 좁혀지면  
급 정지를 함. (후미등 ON, 비상등 동작 ON)

★ 현재 TASK 상태 : Task 1(bldc motor), 2(servo motor) 중지. 고정 값 적용으로  
throttle(1500), rudder(1250) 정지.

3) RF 송수신기의 toggle switch(ch7)를 이용하여 High 신호로 변경한다.

★ 현재 TASK 상태 : Task1,2,5 중지. flag\_accelerate 변수에 의해 Task0(dsp-mcu  
can통신)에서 dsp로 부터 데이터 값(throttle, rudder)을 받아 적용하여 좌측 또는  
우측으로 이동.

★ 현재 TASK 상태 : 장애물 판단 거리를 벗어나면 Task1,2,5 해제

4) RF 송수신기 toggle switch Low로 변경 (비상등 동작 OFF. 우회 방향에 따라 좌, 우  
방향 지시 등 ON)

# 02 S/W PART

- **Firmware**

- TMDX570LC43HDK 개발보드의 기능을 활용한다
- FREERTOS 운영체제를 이용하여 MCU를 제어한다

- 1) MCU 특징

- 2) 통합 개발 환경 설정(IDE)

- 3) 시나리오에 적합한 Task를 생성하여 운영체제가 스케줄링을 한다

- 4) 요구 사항에 맞추어 MCU를 제어한다

- 5) DSP 및 FPGA 와 CAN 통신을 한다

# 02 S/W PART

- **Firmware**

- 1) MCU 특징



Hercules™ Arm® Cortex®-R MCUs for  
**functional safety**

Simplify system safety certification with TÜV SÜD certified IEC61508 SIL3 and ISO26262 ASIL D microcontrollers for automotive and industrial applications.

- TI 社 의 Hercules 모델 Arm Cortex R series(cortex R5F)
- ISO 26262 규격 자동차 산업에서의 기능 안전에 관한 국제 표준 인증
- IEC61508 자동화 산업 분야의 기능 안전에 관한 국제 표준 인증
- 성장해 가는 자율주행 시장에 응용 할 수 있음

# 02 S/W PART

- Firmware

## 2) 통합 개발 환경 설정(IDE)

### - Code Composer Studio 사용

The screenshot shows the Code Composer Studio interface with the following details:

- Project Explorer:** Shows the project structure under "source". Files listed include: HL\_can.c, HL\_ecap.c, HL\_epcc.c, HL\_equep.c, HL\_errata\_SSWF021\_45.c, HL\_errata.c, HL\_esmc.c, HL\_etpwm.c, HL\_glo.c, HL\_nmpuc.c, HL\_notification.c, HL\_pinnmux.c, HL\_sci.c, HL\_sys.core.asm, HL\_sys\_dma.c, HL\_sys\_intvecs.asm, HL\_sys\_link.cmd, HL\_sys\_main.c, HL\_sys\_mpu.asm, HL\_sys\_prc.c, HL\_sys\_phantom.c, HL\_sys\_pmm.c, HL\_sys\_pmu.asm, HL\_sys\_startUp.c, HL\_sys\_vim.c, HL\_system.c, os\_croutine.c, os\_event\_groupsc, os\_heap.c, os\_list.c, os\_mpu\_wrappers.c, os\_port.c, os\_portasm.asm, os\_queue.c, os\_tasks.c, os\_timer.c, and FreeRTOS\_F485.dll.
- Code Editor:** The main window displays the content of `HL_sys_main.c`. The code includes various #include directives for hardware components like SCI, GPIO, ETPWM, ECAP, CAN, EQUEP, and SYSTEM. It defines constants for UART (sciREG1), DCNT (8), and UnitPeriod (10000000). The code also declares volatile variables for flags related to lidar, output, toggle, led, and accelerate. It includes assembly code for interrupt handlers and system calls.
- Terminal:** The bottom right panel shows the terminal window with the message "No consoles to display at this time."

- Firmware code 작성

# 02 S/W PART

- Firmware

## 2) 통합 개발 환경 설정(IDE)

- Code Composer Studio 사용

The screenshot shows the Code Composer Studio interface. The top menu bar includes File, Edit, View, Project, Tools, Run, Scripts, Window, and Help. The main window has tabs for Debug, Variables, Expressions, and Registers. The Registers tab is active, showing a table with columns for Name and Value. The table lists variables such as rx\_data, receive[0], disp\_udder, disp\_throttle, lidar, emergency\_light, tx\_data, deltapos, and flag\_eqep. A red box highlights the line of code: 715 while(!canIsRxMessageArrived(canREG1, canMESSAGE\_BOX2)). The code editor shows the following C code:

```
711 void canMessageNotification(canBASE_t *node, uint32_t messageBox)
712 {
713     if(node == canREG1)//dsp
714     {
715         while(!canIsRxMessageArrived(canREG1, canMESSAGE_BOX2))
716             canGetData(canREG1, canMESSAGE_BOX2, (uint8 *)&rx_data[0]);
717
718         canTransmit(canREG1, canMESSAGE_BOX1, (const uint8 *)&tx_data[0]);
719
720     }
721
722 }
723
724 if(node == canREG2)//fdea
725 {
726     while(!canIsRxMessageArrived(canREG2, canMESSAGE_BOX2))
```

The bottom status bar indicates Writable, Smart Insert, 715 : 1, and a three-dot ellipsis.

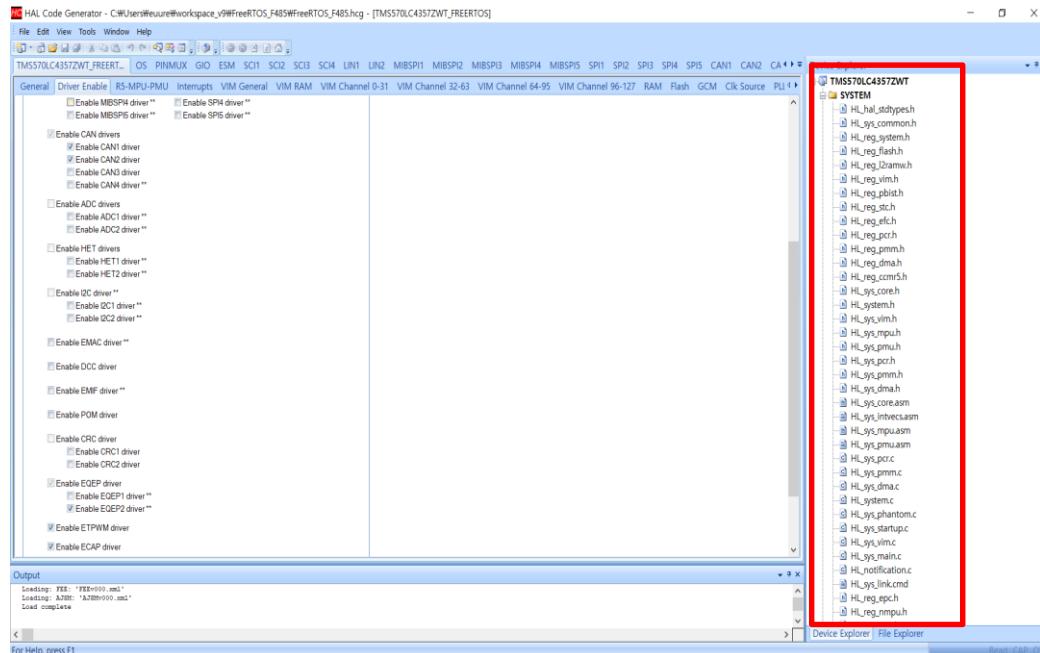
- Debug 모드로 전환하여 중간점을 이용하여 함수가 실행하는 중간에 데이터 값을 확인 할 수 있음

# 02 S/W PART

## • Firmware

### 2) 통합 개발 환경 설정(IDE)

#### - Halcogen 사용



- Halcogen 프로그램으로 사용자 편의를 제공함  
(MCU의 사용할 기능을 활성화 하여 헤더파일을 생성시킴)

# 02 S/W PART

## • Firmware

### 3) 시나리오에 적합한 Task를 생성하여 운영체제가 스케줄링을 한다

```
159 int main(void)
160 {
161 /* USER CODE BEGIN (3) */
162     Init_all();
163
164     disp_set("!!!!!!!!!!!!!!Init_all complete!!!!!!!!!!!!!");
165
166     xSemaphore = xSemaphoreCreateBinary();
167     xSemaphoreGive(xSemaphore); //semaphore initial
168
169     if(xTaskCreate(task0, "can1", configMINIMAL_STACK_SIZE*2, NULL, 5, &dsp_can_handle)!=pdPASS)
170     {
171         for(;;)
172             ;
173     }
174     if(xTaskCreate(task1, "bldc", configMINIMAL_STACK_SIZE, NULL, 5, &bldc_handle)!=pdPASS)
175     {
176         for(;;)
177             ;
178     }
179     if(xTaskCreate(task2, "servo", configMINIMAL_STACK_SIZE, NULL, 5, &servo_handle)!=pdPASS)
180     {
181         for(;;)
182             ;
183     }
184
185     if(xTaskCreate(task3, "sci", configMINIMAL_STACK_SIZE*2, NULL, 5, &sci_handle)!=pdPASS)
186     {
187         for(;;)
188             ;
189     }
190     if(xTaskCreate(task4, "can2", configMINIMAL_STACK_SIZE*2, NULL, 5, &fpga_can_handle)!=pdPASS)
191     {
192         for(;;)
193             ;
194     }
195     if(xTaskCreate(task5, "toggle", configMINIMAL_STACK_SIZE, NULL, 5, &toggle_handle)!=pdPASS)
196     {
197         for(;;)
198             ;
199     }
200     if(xTaskCreate(task6, "led", configMINIMAL_STACK_SIZE, NULL, 5, &led_handle)!=pdPASS)
201     {
202         for(;;)
203             ;
204     }
205     if(xTaskCreate(task7, "eqep", configMINIMAL_STACK_SIZE, NULL, 5, &eqep_handle)!=pdPASS)
206     {
207         for(;;)
208             ;
209     }
210
211     vTaskStartScheduler();
```

Task 스케줄러 실행

- 각 기능에 해당하는 Task를 생성하여 스케줄링 실행을 한다
- 각 Task는 우선순위가 같으며 라운드로빈 방식으로 실행된다

# 02 S/W PART

- Firmware

## 3) 시나리오에 적합한 Task를 생성하여 운영체제가 스케줄링을 한다

```
159 int main(void)
160 {
161 /* USER CODE BEGIN (3) */
162     Init_all();
163
164     disp_set("!!!!!!!!!!!!!!Init_all complete!!!!!!!!!!!!!");
165
166     xSemaphore = xSemaphoreCreateBinary(); → 세마포어 함수를 사용하기 위한 선언
167     xSemaphoreGive(xSemaphore); // semaphore initial
168
169     if(xTaskCreate(task0, "can1", configMINIMAL_STACK_SIZE*2, NULL, 5, &dsp_can_handle)!=pdPASS)
170     {
171         for(;;)
172             ;
173     }
174     if(xTaskCreate(task1, "bldc", configMINIMAL_STACK_SIZE, NULL, 5, &bldc_handle)!=pdPASS)
175     {
176         for(;;)
177             ;
178     }
179     if(xTaskCreate(task2, "servo", configMINIMAL_STACK_SIZE, NULL, 5, &servo_handle)!=pdPASS)
180     {
181         for(;;)
182             ;
183     }
184
185     if(xTaskCreate(task3, "sci", configMINIMAL_STACK_SIZE*2, NULL, 5, &sci_handle)!=pdPASS)
186     {
187         for(;;)
188             ;
189     }
190     if(xTaskCreate(task4, "can2", configMINIMAL_STACK_SIZE*2, NULL, 5, &fpga_can_handle)!=pdPASS)
191     {
192         for(;;)
193             ;
194     }
195     if(xTaskCreate(task5, "toggle", configMINIMAL_STACK_SIZE, NULL, 5, &toggle_handle)!=pdPASS)
196     {
197         for(;;)
198             ;
199     }
200     if(xTaskCreate(task6, "led", configMINIMAL_STACK_SIZE, NULL, 5, &led_handle)!=pdPASS)
201     {
202         for(;;)
203             ;
204     }
205     if(xTaskCreate(task7, "eqep", configMINIMAL_STACK_SIZE, NULL, 5, &eqep_handle)!=pdPASS)
206     {
207         for(;;)
208             ;
209     }
210
211     vTaskStartScheduler();
212 }
```

- 각 Task의 실행 보장을 위해 세마포어를 생성한다
- 세마포어를 사용하게 되면 다른 Task에서 공유자원(MCU)을 선점 할 수 없게 된다

# 02 S/W PART

## • Firmware

### 3) 시나리오에 적합한 Task를 생성하여 운영체제가 스케줄링을 한다

```
222 void task0(void *pvParameters)//can(canREG1) dsp-mcu task port6
223{
224    for(;;)
225    {
226        if(xSemaphoreTake(xSemaphore, (TickType_t)0x01)==pdTRUE)
227        {
228            if(rx_data[0])//dsp_control can interrupt execute
229            {
230                dsp_throttle = rx_data[1]*1000 + rx_data[2]*100 + rx_data[3]*10 + rx_data[4];
231
232                if(rx_data[5] == 1)
233                {
234                    dsp_rudder = 900;
235                }else if(rx_data[5] == 2)
236                {
237                    dsp_rudder = 1700;
238                }
239
240                //dsp_rudder = rx_data[2]<<8;
241                // dsp_rudder |= rx_data[3];
242
243
244
245                // rx_data[0]= dsp_throttle >> 8
246                //rx_data[1] = dsp_throttle & 0xFF
247                //wait(100000);
248            }
249
250            xSemaphoreGive(xSemaphore); → 세마포어 해제
251            vTaskDelay(30);
252        }
253    }
254 }
```

- 각 Task마다 세마포어 사용
- Task 실행 완료 후 다른 Task 실행을 위해 세마포어 해제

# 02 S/W PART

- Firmware

## 4) 요구 사항에 맞추어 MCU를 제어한다

```
86 volatile int flag;
87 volatile int flag_lidar= 0;
88 volatile int flag_output = 0;
89 volatile int flag_toggle = 0;
90 volatile int flag_led = 0;
91 volatile int flag_accelerate = 0;
92
93 int throttle,rudder,data,output;
94 int dsp_throttle, dsp_rudder=0;
95 int front_led, rear_led, left_blink, right_blink, blink_led, emergency_light;
96
97
98 char buf0[64]={0};//toggle switch output
99 unsigned int buf_len0;//toggle switch output
100
101 char buf1[64]={0};//bldc throttle
102 unsigned int buf_len1;//bldc throttle
103
104 char buf2[64]={0};//servo rudder
105 unsigned int buf_len2;//servo rudder
106
107 char buf3[64]={0};//etpwmREG1->CMPA use
108 unsigned int buf_len3;//etpwmREG1->CMPA use
109
110 char buf4[64]={0};//etpwmREG2->CMPA use
111 unsigned int buf_len4;//etpwmREG2->CMPA use
112
113
114
115 uint8_t tx_data[8]={0};
116 uint8_t rx_data[DCNT]={0};
117 uint8_t receive[5]={0x01,0x2c};
118 uint16_t lidar=300;|
119 int deltaT=0;;
120 int deltapos=0;
121 volatile uint16_t flag_eqep;
122 volatile uint8_t dsp_flag;
123
124 void Init_all(void);
125 void sci_display(sciBASE_t *sci, uint8 *test, uint32 len);
126 void disp_set(char *str);
127 void wait(uint32);
128 void ecap_throttle(void); //ecap_control pwm
129 void ecap_servo(void); //ecap_control pwm
130 void ecap_toggle(void); //ecap_toggle switch
131 void ecap_channel15(void);
132 void ecap_channel16(void);
133 void ecap_channel18(void);
134 void bldc_back(void); //bldc back throttle
135 void bldc_back_low(void); //bldc back low throttle
136 void servo_control(void); //servo rudder control pwm
137
```

- 공유 자원(MCU, 변수) 선점 문제로 인한 변수 분리 및 디바이스 단위로 함수 구현

# 02 S/W PART

- Firmware

## 4) 요구 사항에 맞추어 MCU를 제어한다

```
547 void Init_all(void)//Initial function
548 {
549     sciInit();
550     gioInit();
551     gioSetBit(gioPORTA, 0 , 0);
552     gioSetBit(gioPORTA, 1, 0);
553     gioSetBit(gioPORTA, 2, 0);
554     gioSetBit(gioPORTA, 3, 0);
555     ecapInit();
556     ecapStartCounter(ecapREG1);
557     ecapStartCounter(ecapREG2);
558     ecapStartCounter(ecapREG3);
559     ecapStartCounter(ecapREG4);
560     ecapStartCounter(ecapREG5);
561     ecapStartCounter(ecapREG6);
562     ecapEnableCapture(ecapREG1);
563     ecapEnableCapture(ecapREG2);
564     ecapEnableCapture(ecapREG3);
565     ecapEnableCapture(ecapREG4);
566     ecapEnableCapture(ecapREG5);
567     ecapEnableCapture(ecapREG6);
568     etpwmInit();
569     etpwmStartTBCLK();
570     canInit();
571     canEnableErrorNotification(canREG1);
572     canEnableErrorNotification(canREG2);
573     QEPIInit();
574     eqepSetUnitPeriod(eqepREG2, UnitPeriod);
575     eqepEnableCounter(eqepREG2);
576     eqepEnableUnitTimer(eqepREG2);
577     eqepEnableCapture(eqepREG2);
578
579     wait(1000000);
580 }
```

- 초기화 함수를 main문에서 처리하면 운영체제가 초기화 도중 다른 작업을 수행함
- 함수로 한번에 묶어서 처리

# 02 S/W PART

## • Firmware

### 4) 요구 사항에 맞추어 MCU를 제어한다

```
222 void task0(void *pvParameters)//can(canREG1) dsp-mcu task port6
223 {
224     for(;;)
225     {
226         if(xSemaphoreTake(xSemaphore, (TickType_t)0x01)==pdTRUE)
227         {
228             if(rx_data[0])//dsp_control can interrupt execute
229             {
230                 {
231                     dsp_throttle = rx_data[1]*1000 + rx_data[2]*100 + rx_data[3]*10 + rx_data[4];
232
233                     if(rx_data[5] == 1)
234                     {
235                         dsp_rudder = 900;
236                     }else if(rx_data[5] == 2)
237                     {
238                         dsp_rudder = 1700;
239                     }
240
241                     //dsp_rudder = rx_data[2]<<8;
242                     // dsp_rudder |= rx_data[3];
243
244                     // rx_data[0]= dsp_throttle >> 8
245                     //rx_data[1] = dsp_throttle & 0xFF
246                     //wait(100000);
247                 }
248             }
249         }
250
251         xSemaphoreGive(xSemaphore);
252         vTaskDelay(30);
253     }
254 }
255 }
```

- DSP와 CAN 통신(8bit) 통하여 차선 인식 시 rx\_data[0]~[5] 값을 참조하여 PWM값을 변수에 저장
- `dsp_throttle` = bldc 모터 PWM 값
- `dsp_rudder` = servo 모터 PWM 값

# 02 S/W PART

## • Firmware

### 4) 요구 사항에 맞추어 MCU를 제어한다

```
258 void task1(void *pvParameters)//ecap-bldc_motor task
259 {
260     for(;;)
261     {
262         if(xSemaphoreTake(xSemaphore, (TickType_t)0x01)==pdTRUE)
263         {
264             ecap_throttle(); // RF 송신기의 ch3 기능을 사용하기 위한 ecap 정의 함수
265             if((throttle>=1200 && throttle<1450))
266             {
267                 bldc_back();
268             }
269             PWM 듀티에 따른 후진 동작 구현
270             else if(throttle<1200 && throttle>=900)
271             {
272                 bldc_back_low();
273             }
274             else if(throttle<900)
275             {
276                 etpwmREG1->CMPA = 1500;
277             }
278             PWM 듀ti에 따른 중립 동작 구현
279             else if(throttle>=1450 && throttle<=1550)
280             {
281                 etpwmREG1->CMPA = 1500;
282             }
283             else if(throttle>1550 && throttle<=1800)
284             {
285                 etpwmREG1->CMPA = throttle;
286             }
287             else
288             {
289                 etpwmREG1->CMPA = 1800;
290             }
291             xSemaphoreGive(xSemaphore);
292             vTaskDelay(30);
293         }
294     }
```

- BLDC + ESC 특성에 맞는 전진 후진 PWM 값 설정
- RF 송수신기의 throttle 채널의 low data 값을 새로 정의
- BLDC 모터의 PWM값은 1200~1800으로 제한

# 02 S/W PART

## • Firmware

### 4) 요구 사항에 맞추어 MCU를 제어한다

```
296 void task2(void *pvParameters)//ecap-servo_motor task
297 {
298     for(;;)
299     {
300         if(xSemaphoreTake(xSemaphore, (TickType_t)0x01)==pdTRUE)
301         {
302             ecap_servo(); // RF 송신기의 ch4 기능을 사용하기
303             servo_control(); // 위한 ecap 정의 함수
304
305             xSemaphoreGive(xSemaphore);
306             vTaskDelay(30);
307         }
308     }
309 }
310
688 void servo_control(void)
689 {
690     if(rudder>=900 && rudder<1200)
691     {
692         etpwmREG2->CMPA = rudder;
693     }else if(rudder < 900)
694     {
695         etpwmREG2->CMPA = 900;
696     }else if(rudder>=1200 && rudder<=1300)
697     {
698         etpwmREG2->CMPA = 1250;
699     }else if(rudder>1300 && rudder<=1800)
700     {
701         etpwmREG2->CMPA = rudder;
702     }else if(rudder>1800 && rudder<=2000)
703     {
704         etpwmREG2->CMPA = 1800;
705     }else
706     {
707         etpwmREG2->CMPA = 1250;
708     }
709 }
```

- Servo 모터 특성에 맞게 좌우 PWM 값 설정
- RF 송수신기의 rudder 채널 low data 값을 새로 정의
- Servo의 PWM값은 900~1800으로 제한

# 02 S/W PART

- Firmware

## 4) 요구 사항에 맞추어 MCU를 제어한다

```
311 void task3(void *pvParameters)//sci task
312 {
313     for(;;)
314     {
315         if(xSemaphoreTake(xSemaphore, (TickType_t)0x01)==pdTRUE)
316         {
317             sprintf(buf0, "!!!!!!ecap output!!!!!!! = %d\n\r\0", output);
318             buf_len0 = strlen(buf0);
319             sci_display(UART, (uint8 *)buf0, buf_len0);
320             sprintf(buf1, "bldc motor = %d ms\n\r\0", throttle);
321             buf_len1 = strlen(buf1);
322             sci_display(UART, (uint8 *)buf1, buf_len1);
323             sprintf(buf2, "servo motor = %d ms\n\r\0", rudder);
324             buf_len2 = strlen(buf2);
325             sci_display(UART, (uint8 *)buf2, buf_len2);
326             sprintf(buf3, "etpwmREG1->CMPA = %d ms\n\r\0", etpwmREG1->CMPA);
327             buf_len3 = strlen(buf3);
328             sci_display(UART, (uint8 *)buf3, buf_len3);
329             sprintf(buf4, "lidar = %d ms\n\r\0", lidar);
330             buf_len4 = strlen(buf4);
331             sci_display(UART, (uint8 *)buf4, buf_len4);

333             xSemaphoreGive(xSemaphore);
334             vTaskDelay(30);
335         }
336     }
337 }
```

- SCI 통신을 이용하여 각 Task의 실시간 데이터 값을 확인 할 수 있게 디버깅 용도로 사용

# 02 S/W PART

- Firmware

## 4) 요구 사항에 맞추어 MCU를 제어한다

```
339 void task4(void *pvParameters)//canREG2(mcu-FPGA) task port3
340 {
341     for(;;)
342     {
343         if(xSemaphoreTake(xSemaphore, (TickType_t)0x01)==pdTRUE)
344         {
345             lidar = receive[0]<<8;
346             lidar |= receive[1];
347             //2byte 씩 받아서 처리(256오버플로우)
348             if(lidar <= 250 && flag_accelerate == 0) //already dsp_choice(left, right)
349             {
350                 vTaskSuspend(bldc_handle);
351                 vTaskSuspend(servo_handle);
352                 //vTaskSuspend(sci_handle);
353                 vTaskSuspend(toggle_handle);
354
355                 //canTransmit(canREG1, canMESSAGE_BOX1, (const uint8 *)&tx_data[0]);
356
357                 etpwmREG1->CMPA = dsp_throttle;
358                 etpwmREG2->CMPA = dsp_rudder;
359                 //wait(2000000);
360                 flag_lidar = 1;
361                 dsp_flag = 1;
362
363             }
364             if(lidar <= 250 && flag_accelerate == 1)//scenario 2
365             {
366                 vTaskSuspend(bldc_handle);
367                 vTaskSuspend(servo_handle);
368                 //vTaskSuspend(toggle_handle);
369                 etpwmREG1->CMPA = 1500;
370                 etpwmREG2->CMPA = 1250;
371                 gioSetBit(gioPORTA, 2, 1);
372                 gioSetBit(gioPORTA, 3, 1);
373                 dsp_flag = 1;
374
375             }
376             if(lidar > 250 && flag_lidar == 1)
377             {
378                 vTaskResume(bldc_handle);
379                 vTaskResume(servo_handle);
380                 //vTaskResume(sci_handle);
381                 //wait(1000000);
382                 flag_toggle = 1;
383                 flag_lidar = 0;
384             }
385             if(lidar > 250 && flag_toggle == 1)
386             {
387                 vTaskResume(toggle_handle);
388                 flag_toggle = 0;
389                 dsp_flag = 0;
390             }
391         }
392         xSemaphoreGive(xSemaphore);
393         vTaskDelay(30);
394     }
395 }
```

2byte 씩 받아서 처리(256오버플로우)

장애물 판단하는 시점에  
DSP로 부터 받은 데이터  
값으로 PWM 적용

- FPGA와 CAN 통신을 통하여 Lidar 거리 데이터 값을 실시간 저장함
- 설정한 거리이내 장애물 감지 시 시나리오에 맞게 Task가 동작함

# 02 S/W PART

## • Firmware

### 4) 요구 사항에 맞추어 MCU를 제어한다

```
397 void task5(void *pvParameters)//toggle switch output task
398 {
399     for(;;)
400     {
401         if(xSemaphoreTake(xSemaphore, (TickType_t)0x01)==pdTRUE)
402         {
403             ecap_toggle();           → RF 송신기의 ch7 기능을 사용하기
404             if(output>1900)        위한 ecap 정의 함수
405             {
406                 vTaskSuspend(bldc_handle);
407                 vTaskSuspend(servo_handle);
408                 //vTaskSuspend(sci_handle);
409                 //vTaskSuspend(dsp_can_handle);
410
411                 etpwmREG1->CMPA = 1560;
412                 etpwmREG2->CMPA = 1250;
413
414                 flag_output = 1;
415                 flag_accelerate = 0;
416             }
417             if(output<=1900 && flag_output == 1)
418             {
419                 vTaskResume(bldc_handle);
420                 vTaskResume(servo_handle);
421                 flag_output = 0;
422                 flag_accelerate = 1;
423             }
424
425             xSemaphoreGive(xSemaphore);
426             vTaskDelay(30);
427         }
428     }
429 }
```

- RF송신기의 ch7 toggle 스위치를 이용하여 고정 값 주행 및 수동 주행 설정

# 02 S/W PART

## • Firmware

### 4) 요구 사항에 맞추어 MCU를 제어한다

```
431 void task6(void *pvParameters)//led control task
432 {
433     for(;;)
434     {
435         if(xSemaphoreTake(xSemaphore, (TickType_t)0x01)==pdTRUE)
436         {
437             ecap_channel5();
438             ecap_channel6(); → RF 송신기의 ch5,6,8 기능을 사용하기
439             ecap_channel8();
440             if(epwmREG1->CMPA == 1500)
441             {
442                 gioSetBit(gioPORTA, 1, 1);
443             }else
444             {
445                 gioSetBit(gioPORTA, 1, 0);
446             }
447             if(front_led > 1800)
448             {
449                 gioSetBit(gioPORTA, 0, 1);
450             }else
451             {
452                 gioSetBit(gioPORTA, 0, 0);
453             }
454             if(blink_led >= 10 && blink_led < 1000)
455             {
456                 gioSetBit(gioPORTA, 2, 1);
457                 flag_led = 0;
458             }
459             if(blink_led < 10)
460             {
461                 gioSetBit(gioPORTA, 2, 0);
462                 gioSetBit(gioPORTA, 3, 0);
463                 flag_led = 0;
464             }
465             if(blink_led >= 1400 && blink_led <= 1600)
466             {
467                 gioSetBit(gioPORTA, 2, 0);
468                 gioSetBit(gioPORTA, 3, 0);
469                 flag_led = 1;
470             }
471             if(blink_led >1900)
472             {
473                 gioSetBit(gioPORTA, 3, 1);
474                 flag_led = 0;
475             }
476             if((emergency_light > 1900 ) && (flag_led == 1))//emergency light display
477             {
478                 gioSetBit(gioPORTA, 2, 1);
479                 gioSetBit(gioPORTA, 3, 1);
480                 wait(10000000);
481                 gioSetBit(gioPORTA, 2, 0);
482                 gioSetBit(gioPORTA, 3, 0);
483                 wait(10000000);
484                 flag_led = 0;
485             }
486             xSemaphoreGive(xSemaphore);
487             vTaskDelay(30);
488         }
489     }
490 }
```

- RF송신기의 ch5 toggle 스위치를 이용하여 좌우 방향 지시등 제어에 사용
- Ch6 스위치를 이용하여 비상등 제어에 사용
- Ch8 스위치를 이용하여 전조등 제어에 사용

# 02 S/W PART

- Firmware

4) 요구 사항에 맞추어 MCU를 제어한다



# 02 S/W PART

## • Firmware

### 4) 요구 사항에 맞추어 MCU를 제어한다

```
492 void task7(void *pvParameters)//velocity display task
493 {
494     for(;;)
495     {
496         if(xSemaphoreTake(xSemaphore, (TickType_t)0x01)==pdTRUE)
497         {
498             if((eqepREG2->QFLG & 0x800) == 0x800)
499             {
500                 flag_eqep = (eqepREG2->QEPSTS & 0x20) >> 5;
501             }
502             deltaT = 0; → QEPSTS 레지스터를 통해 정방향 인지
503                                         역방향인지 알 수 있음
504             deltapos = eqepReadPosnLatch(eqepREG2);
505
506             if(deltapos < 0)
507             {
508                 deltapos = eqepReadPosnLatch(eqepREG2);
509                 deltapos = -deltapos;
510             }
511             if(deltapos < 15)
512             {
513                 deltapos = 0;
514             }
515
516             deltaT = eqepREG2->QUPRD; //per second
517
518
519             eqepClearInterruptFlag(eqepREG2, QEINT_Uto);
520
521             tx_data[3] = dsp_flag;//flag status
522             tx_data[2] = deltapos % 100; //dsp setting data
523             tx_data[1] = (deltapos / 100) % 100;
524             tx_data[0] = deltapos / 10000;
525
526             xSemaphoreGive(xSemaphore);
527             vTaskDelay(30);
528         }
529     }
530 }
```

- DSP와 CAN통신을 통해 LCD에 속도를 표시하기 위해 Eqep 기능을 사용하여 엔코더의 카운팅 값을 계산

DSP에 전송할 엔코더의 카운팅 값을 각 인덱스에 계산하여 저장

# 02 S/W PART

## • Firmware

### 5) DSP 및 FPGA 와 CAN 통신을 한다

```
709 void canMessageNotification(canBASE_t *node, uint32_t messageBox)
710 {
711     if(node == canREG1)//dsp
712     {
713         while(!canIsRxMessageArrived(canREG1, canMESSAGE_BOX2))
714         ;
715         canGetData(canREG1, canMESSAGE_BOX2, (uint8 *)&rx_data[0]);
716
717         canTransmit(canREG1, canMESSAGE_BOX1, (const uint8 *)&tx_data[0]);
718
719         → DSP로 부터 rx_data[] 데이터(throttle, rudder값)를
720             받으면 바로 tx_data[](엔코더 카운팅 값) 데이터를
721             보냄
722     }
723
724     if(node == canREG2)//fpga
725     {
726         while(!canIsRxMessageArrived(canREG2, canMESSAGE_BOX2))
727         ;
728     }
729 }
```

→ DSP로 부터 rx\_data[] 데이터(throttle, rudder값)를 받으면 바로 tx\_data[](엔코더 카운팅 값) 데이터를 보냄

→ FPGA로 부터 Lidar 거리 값을 계속 받음

- DSP ↔ MCU ↔ FPGA 간에 CAN통신은 인터럽트 핸들러로 처리
- Task 동작 중 인터럽트 우선순위로 인해 CAN 인터럽트가 먼저 실행 됨.
- 인터럽트가 시작되는 시점에 실행되는 Task는 컨택스트 스위칭을 하게 되고 핸들러 실행이 끝나면 다시 컨택스트 스위칭이 일어난 시점으로 돌아와 stack에 저장한 정보를 불러와 Task가 정상 동작 함

# 02 S/W PART

- **ERRATA**

- #### - 공유 자원 선점 문제로 인한 Task 비정상 동작

실제 전동차를 구동을 위해  
관련 Task (ecap-bldc,  
ecap-servo)만 생성하여  
간단한 동작에는 문제가  
없지만 i2c-Lidar간에 통신  
Task를 생성한 후에는  
servo 값이 정상 출력 하지  
못함

# 02 S/W PART

## • ERRATA

### - 공유 자원 선점 문제로 인한 Task 비정상 동작

```
HL_sys_main.c
242 }
243
244 void task1(void *pvParameters) //bldc motor control
245 {
246     unsigned int cap[2];
247
248     for(;;)
249     {
250         if(xSemaphoreTake(xSemaphore,(TickType_t)0x01)==pdTRUE)
251         {
252             cap[0] = ecapGetCAP1(ecapREG3);
253             cap[1] = ecapGetCAP2(ecapREG3);
254             throttle = (cap[1] - cap[0])/VCLK3_FREQ; //channel 3
255             sprintf(buf, "bldc motor = %d ms\nr", throttle);
256             buf_len = strlen(buf);
257             sci;display(UART,(uint8 *)buf, buf_len);
258             if((1200 <= throttle && throttle < 1450)//bldc_motor back
259             {
260                 bldc_back();
261             }else if(throttle<1200 && throttle >= 900)//bldc_motor back
262             {
263                 bldc_back_low();
264             }
265             }else if(throttle < 900)
266             {
267                 etpwmREG1->CMPA = 1500;
268             }else if(1450 < throttle && throttle <= 1550) //bldc_motor neutrality
269             {
270                 etpwmREG1->CMPA = 1500;
271             }else if(1550 < throttle & throttle <= 1800) //bldc_motor forward
272             {
273                 etpwmREG1->CMPA = throttle;
274             }else//bldc_motor forward
275             {
276                 etpwmREG1->CMPA = 1800;
277             }
278             xSemaphoreGive(xSemaphore);
279             vTaskDelay(30);
280         }
281     }
282 }

HL_scic
284 void task2(void *pvParameters)
285 {
286     char buf[128] = {0};
287     unsigned int buf_len;
288     unsigned int cap[2];
289
290     for(;;)
291     {
292         if(xSemaphoreTake(xSemaphore,(TickType_t)0x01)==pdTRUE)
293         {
294             cap[0] = ecapGetCAP1(ecapREG2);
295             cap[1] = ecapGetCAP2(ecapREG2);
296             rudder = (cap[1] - cap[0])/VCLK3_FREQ; //channel 2
297             sprintf(buf, "servo motor = %d ms\nr", rudder);
298             buf_len = strlen(buf);
299             sci;display(UART,(uint8 *)buf, buf_len);
300             rudder= 2700-rudder;
301
302             if(rudder>=900 && rudder<1200)
303             {
304                 etpwmREG2->CMPA = rudder;
305             }else if(rudder < 900)
306             {
307                 etpwmREG2->CMPA = 900;
308             }else if(rudder>=1200 && rudder <= 1300)
309             {
310                 etpwmREG2->CMPA = 1250;
311             }else if(rudder>1300 && rudder<=1800)
312             {
313                 etpwmREG2->CMPA = rudder;
314             }else
315             {
316                 etpwmREG2->CMPA = 1800;
317             }
318             xSemaphoreGive(xSemaphore);
319             vTaskDelay(50);
320         }
321     }
322 }
```

- Task1 과 Task2의 구조를 보면 ecap 신호로 얻은 Low data 값을 범위로 지정하여 PWM 값으로 사용함
- PWM값을 sci통신으로 출력을 하는데 이때 Task간에 race condition이 발생함

# 02 S/W PART

- ERRATA

- 공유 자원 선점 문제로 인한 Task 비정상 동작

```

95 int throttle,rudder,data;
96 /*char buf[128]={0};//task1 use
97 unsigned int buf_len; //task1 use
98 char buf2[128]={0};//task2 use
99 unsigned int buf_len2;//task2 use*/
100
101 char buf[128]={0};//only lidar use
102 unsigned int buf_len; //only lidar use
103 uint16 tmp; //only lidar use
104
105 char buf1[128]={0}; //only bldc use
106 unsigned int buf_len1; //only bldc use
107
108 char buf2[128]={0}; //only servo use
109 unsigned int buf_len2; //only servo use
110
111 char buf3[128]={0}; //only CMPA use
112 unsigned int buf_len3; //only CMPA use

113 uint8 receives[2];
114 uint8 bias_cnt;
115 uint8_t rx_data[DONT]={0};

116 void Init_all(void);
117 void lidar_enable(void);
118 void sci_display(sciBASE_t *sci, uint8 *test, uint32 len
119 void disp_set(char *str);
120 void wait(uint32);
121 void get_data(void);
122 void lidar_bias(void);
123 void use_lidar(void);
124 void ecap_throttle(void); //ecap_control pim
125 void ecap_servo(void); //ecap_control pim
126 void bldc_back(void); //bldc back throttle
127 void bldc_back_low(void); //bldc back low throttle
128 void servo_control(void); //servo rudder control pim

129 void task0(void *pvParameters);
130 void task1(void *pvParameters);
131 void task2(void *pvParameters);
132 void task3(void *pvParameters);

/* USER CODE BEGIN (4) */
1void task0(void *pvParameters) Lidar Task
2{
3    for(;;)
4    {
5        if(xSemaphoreTake(xSemaphore, (TickType_t)0x01)==pdTRUE)
6        {
7            use_lidar();
8            xSemaphoreGive(xSemaphore);
9            vTaskDelay(30);
10       }
11    }
12}
13void task1(void *pvParameters)
14{
15    for(;;)
16    {
17        if(xSemaphoreTake(xSemaphore, (TickType_t)0x01)==pdTRUE)
18        {
19            ecap_throttle();
20            if((throttle)>=1200 && throttle<1450)
21            {
22                bldc_back();
23            }
24            else if(throttle>=1200 && throttle<900)
25            {
26                bldc_back_low();
27            }
28            else if(throttle<900)
29            {
30                etpwmREG1->CMPA = 1500;
31            }
32            else if(throttle>=1450 && throttle<=1550)
33            {
34                etpwmREG1->CMPA = 1500;
35            }
36            else if(throttle>=1550 && throttle<=1800)
37            {
38                etpwmREG1->CMPA = throttle;
39            }
40            else
41            {
42                etpwmREG1->CMPA = 1800;
43            }
44            xSemaphoreGive(xSemaphore);
45            vTaskDelay(30);
46        }
47    }
48}
49void task2(void *pvParameters) Servo Task
50{
51    for(;;)
52    {
53        if(xSemaphoreTake(xSemaphore, (TickType_t)0x01)==pdTRUE)
54        {
55            ecap_servo();
56            servo_control();
57            xSemaphoreGive(xSemaphore);
58            vTaskDelay(30);
59        }
60    }
61}
62void task3(void *pvParameters) Data 출력 Task
63{
64    for(;;)
65    {
66        if(xSemaphoreTake(xSemaphore, (TickType_t)0x01)==pdTRUE)
67        {
68            sprintf(buf0, "Distance = %d\n\r", tmp);
69            buf_len0 = strlen(buf0);
70            sci_display(sciREG1, (uint8 *)buf0, buf_len0);
71            sprintf(buf1, "bldc motor = %d ms\n\r", throttle);
72            buf_len1 = strlen(buf1);
73            sci_display(UART, (uint8 *)buf1, buf_len1);
74            sprintf(buf2, "servo motor = %d ms\n\r", rudder);
75            buf_len2 = strlen(buf2);
76            sci_display(UART, (uint8 *)buf2, buf_len2);
77            sprintf(buf3, "etpwm = %d ms\n\r", etpwmREG1->CMPA);
78            buf_len3 = strlen(buf3);
79            sci_display(UART, (uint8 *)buf3, buf_len3);

80            xSemaphoreGive(xSemaphore);
81            vTaskDelay(30);
82        }
83    }
84}
85

```

디바이스 단위로 함수 구현

같은 기능 사용하더라도 나눠서 구분

- Race condition이 발생하지 않게 디바이스 단위로 함수를 구분하여 코드 작성
- 전역변수 주의

# 02 S/W PART

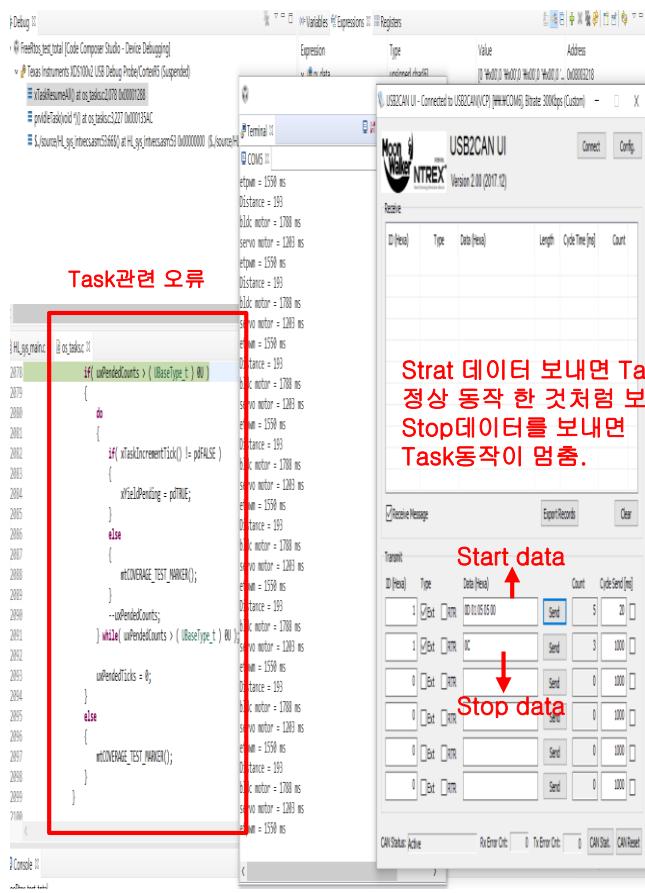
- ERRATA

- Timing 문제

```

510
511 void canMessageNotification(canBASE_t *node, uint32_t messageBox)
512{
513     while(!canIsRxMessageArrived(canREG1, canMESSAGE_BOX2))
514     ;
515     canGetData(canREG1, canMESSAGE_BOX2, (uint8 *)&rx_data[0]);
516
517     switch(rx_data[0])
518     {
519         case 13://can start data
520             vTaskSuspend(bldc_handle);
521             vTaskSuspend(servo_handle);
522             //vTaskSuspend(sci_handle);
523             data = rx_data[1] * 1000 + rx_data[2] * 100 + rx_data[3] * 10 + rx_data[4];
524             etpmREG1->XHPA = data;
525             etpmREG2->XHPA = 1250;
526
527             break;
528         case 12:
529             vTaskResume(bldc_handle);
530             vTaskResume(servo_handle);
531             //vTaskResume(sci_handle);
532
533             break;
534         default:
535             break;
536     }
537
538 // wait[100000];
539
540 memset(rx_data, 0, sizeof(rx_data));
541 data = 0;
542}
543 /* USER CODE END */

```



- DSP에서 명령을 CAN 통신으로 받아서 MCU에서 처리하기 위한 동작 구현을 PC->MCU로 하기 위해서 CAN 인터럽트를 추가하여 핸들러에서 명령어를 처리하려 했지만 동작 이후 task들이 동작하지 않는 문제 발생

## **02 S/W PART**

- **ERRATA**

## - Timing 문제

The screenshot shows the Code Composer Studio interface with the following components:

- Left Panel:** Shows the file `HL_sys_main.c` containing FreeRTOS task code. A red box highlights the section of code that handles CAN messages. The code includes tasks for lidar, servo, and bldc motors, as well as logic for receiving CAN data and sending commands.
- Top Bar:** Includes tabs for Variables, Expressions, and Registers, along with icons for Device Debugging, Terminal, and COM5.
- Right Panel:** The USB2CAN UI interface. It displays a logo for Moon Walker NTREX and the text "USB2CAN UI Version 2.00 (2017.12)". It has sections for Receive and Transmitter. The Receive section shows a table for incoming CAN messages. The Transmitter section shows a table for outgoing CAN messages, with one message currently selected and set to send.

A red box highlights the section of the code in `HL_sys_main.c` that handles CAN messages. The text in the highlighted area is as follows:

```
188 /* USER CODE BEGIN (4) */
189 void task0(void *pvParameters)
190 {
191     for(;;)
192     {
193         if(xSemaphoreTake(xSemaphore, (TickType_t)0x01)==pdTRUE)
194         {
195             use_lidar();
196             switch(rx_data[0])
197             {
198                 case 13://can start data
199                     vTaskSuspend(bldc_handle);
200                     vTaskSuspend(servos_handle);
201                     //vTaskSuspend(sci_handle);
202                     data = rx_data[1] * 1000 + rx_data[2] * 100 +
203                     etpwmREG1->CMPA = data;
204                     etpwmREG2->CMPA = 1250;
205                     break;
206                 case 12:
207                     vTaskResume(bldc_handle);
208                     vTaskResume(servos_handle);
209                     //vTaskResume(sci_handle);
210                     break;
211                 default:
212                     break;
213             }
214             // wait(100000);
215             memset(rx_data, 0, sizeof(rx_data));
216             data = 0;
217             xSemaphoreGive(xSemaphore);
218             vTaskDelay(30);
219         }
220     }
221 }
```

A red box also highlights the section of the code in `HL_sys_main.c` that handles CAN messages. The text in the highlighted area is as follows:

```
188 /* USER CODE BEGIN (4) */
189 void task0(void *pvParameters)
190 {
191     for(;;)
192     {
193         if(xSemaphoreTake(xSemaphore, (TickType_t)0x01)==pdTRUE)
194         {
195             use_lidar();
196             switch(rx_data[0])
197             {
198                 case 13://can start data
199                     vTaskSuspend(bldc_handle);
200                     vTaskSuspend(servos_handle);
201                     //vTaskSuspend(sci_handle);
202                     data = rx_data[1] * 1000 + rx_data[2] * 100 +
203                     etpwmREG1->CMPA = data;
204                     etpwmREG2->CMPA = 1250;
205                     break;
206                 case 12:
207                     vTaskResume(bldc_handle);
208                     vTaskResume(servos_handle);
209                     //vTaskResume(sci_handle);
210                     break;
211                 default:
212                     break;
213             }
214             // wait(100000);
215             memset(rx_data, 0, sizeof(rx_data));
216             data = 0;
217             xSemaphoreGive(xSemaphore);
218             vTaskDelay(30);
219         }
220     }
221 }
```

- 디버깅 모드에서 오류 관련 파일을 보면 Task 와 인터럽트 벡터 관련 오류 확인으로 CAN 인터럽트 핸들러 처리시 switch문을 처리하는 타이밍 및 인터럽트 복귀 주소 관련 오류로 파악 됨

# 02 S/W PART

- ERRATA
  - 통합테스트 중 심각한 문제 발생
- 갑작스러운 급발진으로 전동차가 파손 위험 발생
- POWER, GND, SIGNAL LINE이 길었으며 겹쳐서 배선 작업을 하여 EMC 관련 문제가 있었을 것으로 판단
- 또한 공통 GND 라인이 좁게 연결되어 있으며 노이즈에 대한 방지 대책이 없었음
- 공통 GND 라인을 철판으로 변경하고 브레드 보드용 와이어를 두꺼운 와이어로 변경하여 연결
- 디바이스로 들어가는 POWER Line에 다이오드를 추가하여 역기전력 방지
- 배선을 서로 겹치지 않게 분리하여 장착