

2021 ICNS Self-Driving Cart Manual

목차

1. Self-Driving Cart Specifications

2. Development Environment

- a. JAVA
- b. Cube MX
- c. TrueStudio

3. Project

- a. Generate Code
- b. Debug

4. Self-Driving Control

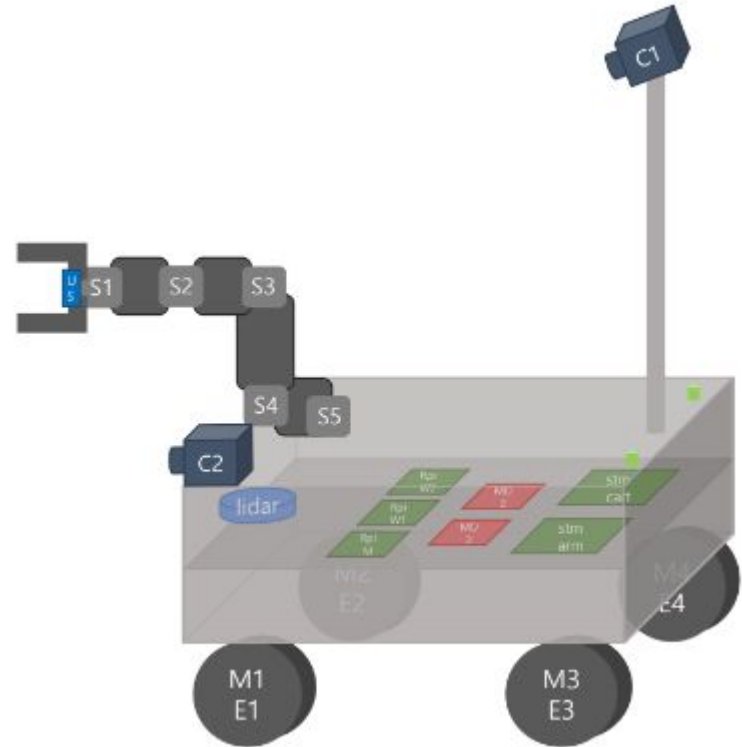
- a. Pin Configuration Diagram
- b. Mecanum Wheel Control
- c. Lidar Control
- d. PID Control

5. Serial Communication

6. Precautions

1. Self-Driving Cart Specifications

- 바퀴 : 96mm Omni Mecanum Wheel
- 모터 : 12V Speed Encoder Motor
- MCU : STM32F407VG
- 보드 : STM32F407G-DISC1
- 배터리
 - Encoder motor : 12v
 - STM : 5v



2. Development Environment

- Cube MX

Cube MX는 ST사에서 제작한 MCU(STM32 칩)의 초기설정을 쉽게 해주는 프로그램이다.

GUI를 통해 타이머, 통신(Uart, I2C, SPI), 인터럽트, DMA, GPIO 등을 설정할 수 있으며, 설정값에 대한 코드를 자동으로 생성해준다.

그 외에도 MCU의 핀 배열이나 동작 클럭을 확인할 수 있다.

- TrueStudio

STM32 칩을 컴파일 할 수 있는 무료버전의 IDE이다.



2. Development Environment - JAVA

- JAVA 설치

Cube MX를 사용하기 위해서는 먼저 **JAVA** 가 설치되어 있어야한다.

JAVA 설치 링크 : <https://www.java.com/ko/download/>



Windows용 64비트 Java

권장 사항 'Version 8 Update 291' (파일 크기: 80.7 MB)
릴리스 날짜: 2021년 4월 20일

Oracle Java 중요 라이선스 업데이트

Oracle Java 라이선스는 2019년 4월 16일 릴리스부터 변경되었습니다.

새로운 Oracle Java SE에 대한 Oracle Technology Network 라이선스 합의서는 이전 Oracle Java 라이선스와는 상당히 다릅니다. 새로운 라이선스는 개인 용도 및 개발 용도와 같은 특정 목적의 무료 사용을 허용하지만, 이전 Oracle Java 라이선스에서 관행이 부여된 기타 사용은 더 이상 허용되지 않습니다. 이 제품을 다운로드하여 사용하기 전에 약관을 자세히 검토하십시오. FAQ는 여기에서 확인할 수 있습니다.

상용 라이선스 및 지원은 저렴한 비용의 Java SE 구독을 통해 제공됩니다.

Oracle은 오픈 소스 GPL 라이선스에 따라 jdk.java.net에서 최신 OpenJDK 릴리스를 제공합니다.

사용자는 현재 Google Chrome을 사용 중이며 이 브라우저에서는 Java 플러그인을 사용하지 못할 수 있습니다. Chrome 버전 42(2015년 4월 릴리스)에서는 브라우저가 플러그인을 지원하는 표준 방식을 사용 안함으로 설정했습니다. [추가 정보](#)

동의 및 무료 다운로드 시작

Java를 다운로드하면 귀하가 Oracle Java SE에 대한 Oracle Technology Network 라이선스 합의서 를 읽고 이 조항에 동의하는 것으로 간주됩니다.

Java 설치 - 진행률



최고의 개발 플랫폼

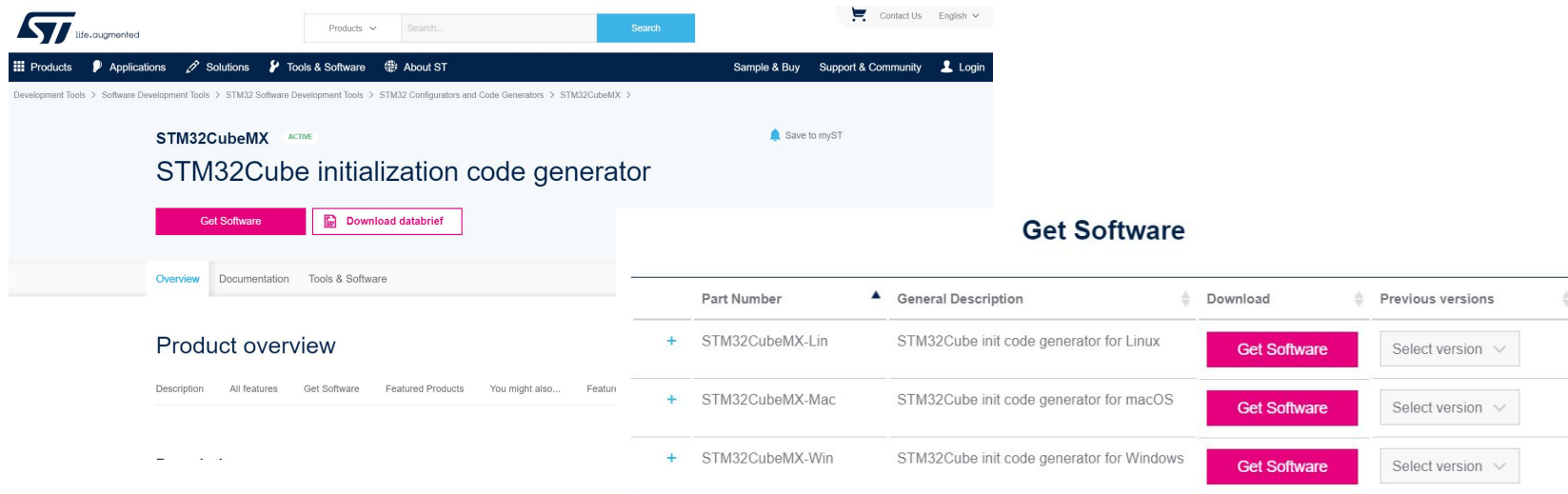
ORACLE

2. Development Environment - CubeMX

- CubeMX 설치

Cube MX 설치 링크 : <https://www.st.com/en/development-tools/stm32cubemx.html>

1. 링크로 이동하여 'Get Software' 클릭



The screenshot shows the STM32CubeMX website. The main heading is "STM32CubeMX" with a subheading "STM32Cube initialization code generator". There are two buttons: "Get Software" and "Download databrief". Below this, there is a "Product overview" section with tabs for "Description", "All features", "Get Software", "Featured Products", "You might also...", and "Features". The "Get Software" tab is selected, showing a table of download links.

| Part Number | General Description | Download | Previous versions |
|-----------------|---|------------------------------|-------------------|
| STM32CubeMX-Lin | STM32Cube init code generator for Linux | Get Software | Select version ▼ |
| STM32CubeMX-Mac | STM32Cube init code generator for macOS | Get Software | Select version ▼ |
| STM32CubeMX-Win | STM32Cube init code generator for Windows | Get Software | Select version ▼ |

2. Development Environment - CubeMX

- CubeMX 설치

2. OS에 따라 'Get Software'클릭 후 이메일 입력

Get Software

| Part Number | General Description | Download | Previous versions |
|-------------------|---|------------------------------|-------------------|
| + STM32CubeMX-Lin | STM32Cube init code generator for Linux | Get Software | Select version ▾ |
| + STM32CubeMX-Mac | STM32Cube init code generator for macOS | Get Software | Select version ▾ |
| + STM32CubeMX-Win | STM32Cube init code generator for Windows | Get Software | Select version ▾ |

Get Software

If you have an account on my.st.com, login and download the software without any further validation steps.

[Login/Register](#)

If you don't want to login now, you can download the software by simply providing your name and e-mail address in the form below and validating it.

This allows us to stay in contact and inform you about updates of this software.

For subsequent downloads this step will not be required for most of our software.

First Name:

Last Name:

E-mail address:

Please enter a valid e-mail address.

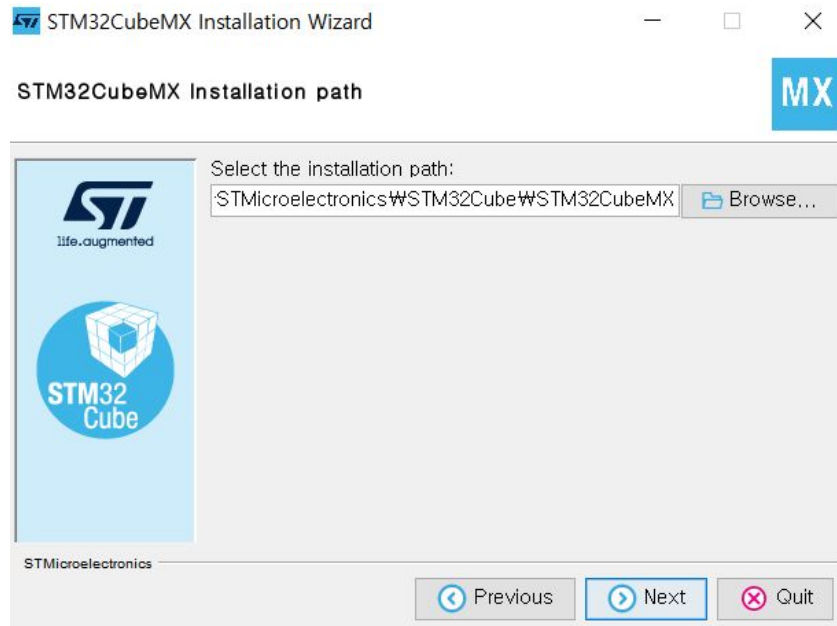
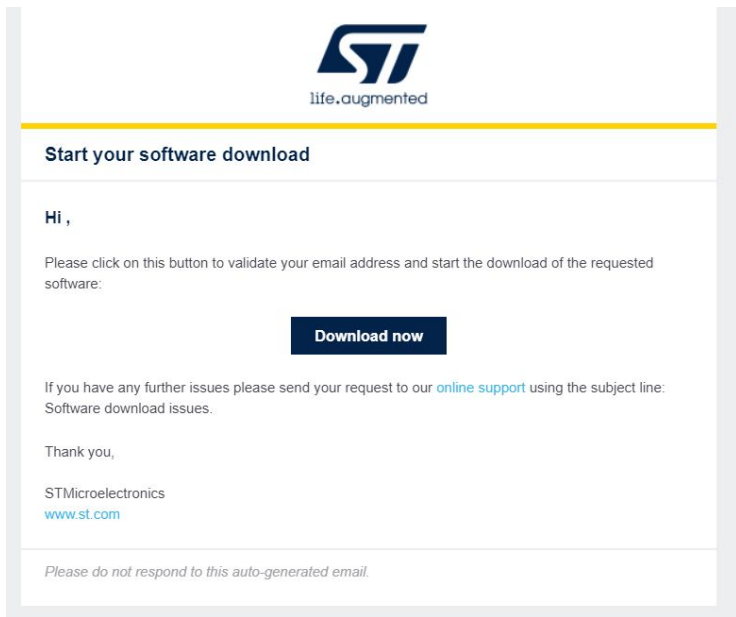
☒ I have read and understood the [Sales Terms & Conditions](#), [Terms of Use](#) and [Privacy Policy](#)

ST (as data controller according to the Privacy Policy) will keep a record of my navigation history and use that information as well as the personal data that I have communicated to ST for marketing purposes relevant to my interests. My personal data will be provided to ST affiliates and distributors of ST in countries located in the European Union and outside of the European Union for the same marketing purposes. [READ MORE >](#)

2. Development Environment - CubeMX

- CubeMX 설치

3. 받은 메일함으로 가서 'Download now' 클릭, 압축 해제 후 설치파일 실행



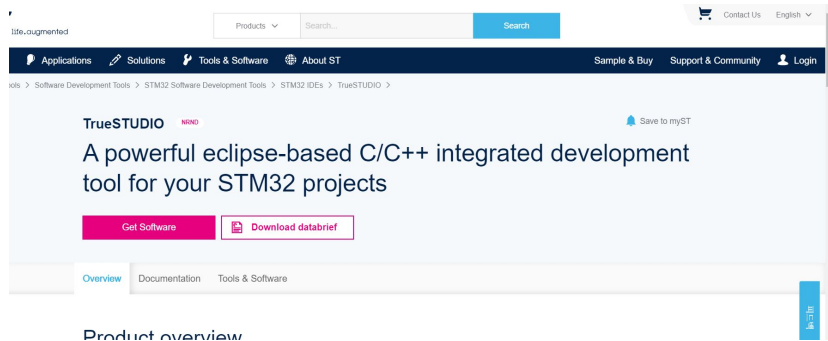
2. Development Environment - TrueStudio

- TrueStudio 설치

True Studio 설치 링크 :

https://www.st.com/content/st_com/en/products/development-tools/software-development-tools/stm32-software-development-tools/stm32-ides/truestudio.html

1. 링크로 이동 후 'Get Software' 클릭



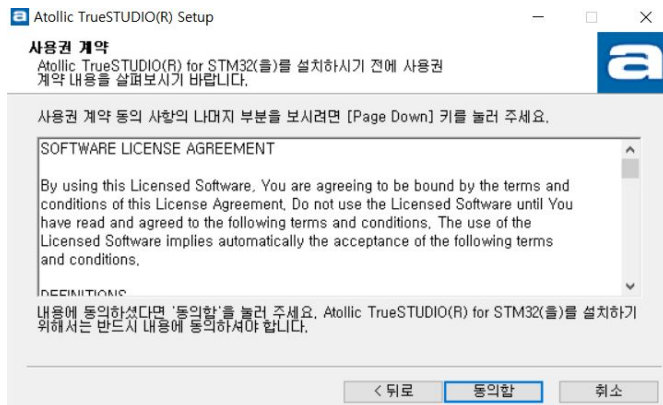
2. Development Environment - TrueStudio

- TrueStudio 설치

2. OS에 따라 선택 후 다운로드

Get Software

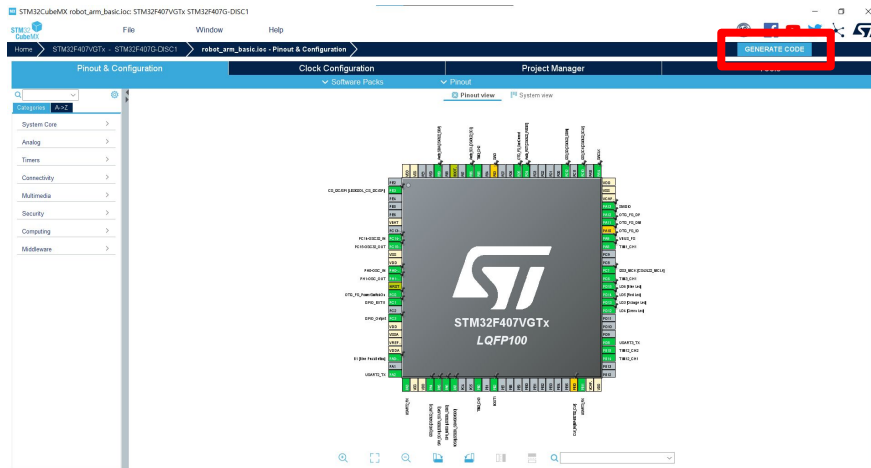
| Part Number | Supplier | Download |
|------------------|----------|--------------|
| + TrueSTUDIO_Lin | ST | Get Software |
| + TrueSTUDIO_Win | ST | Get Software |



3. Project - Generate Code

- 프로젝트 생성

1. CNS Robot Arm Github Repository 클론 <https://github.com/icns-distributed-cloud/Self-driving-project>
2. .ioc 파일 열기 (2021_self_driving_cart/cart)
3. 'GENERATE CODE' 클릭 (ICNS project의 설정을 그대로 가져올 수 있어 별도의 설정이 필요없음)

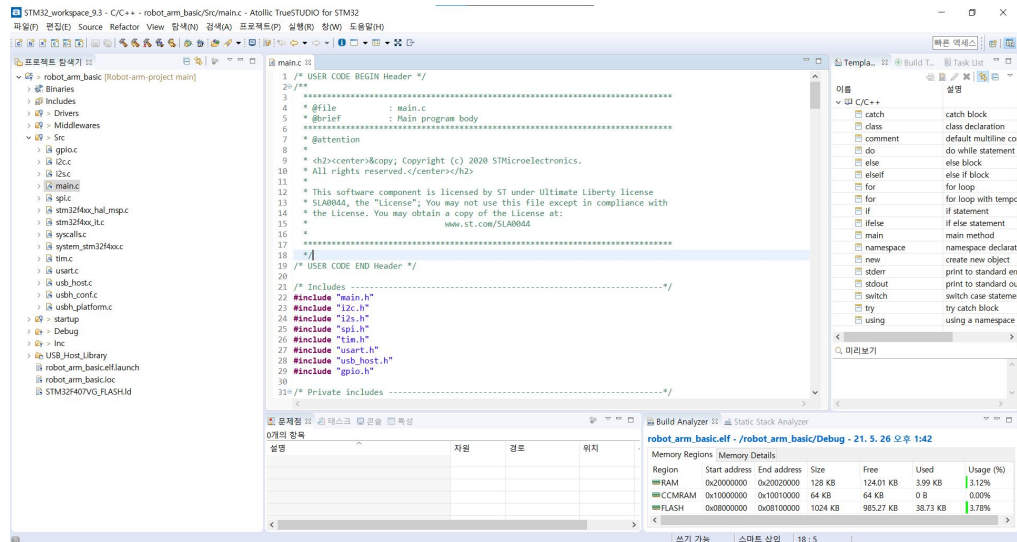
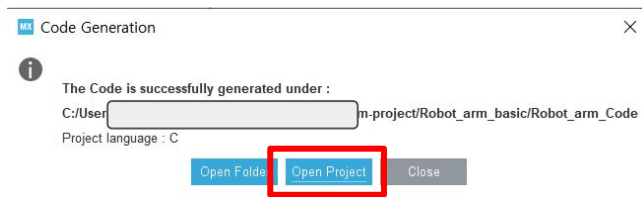


3. Project - Generate Code

- 프로젝트 생성

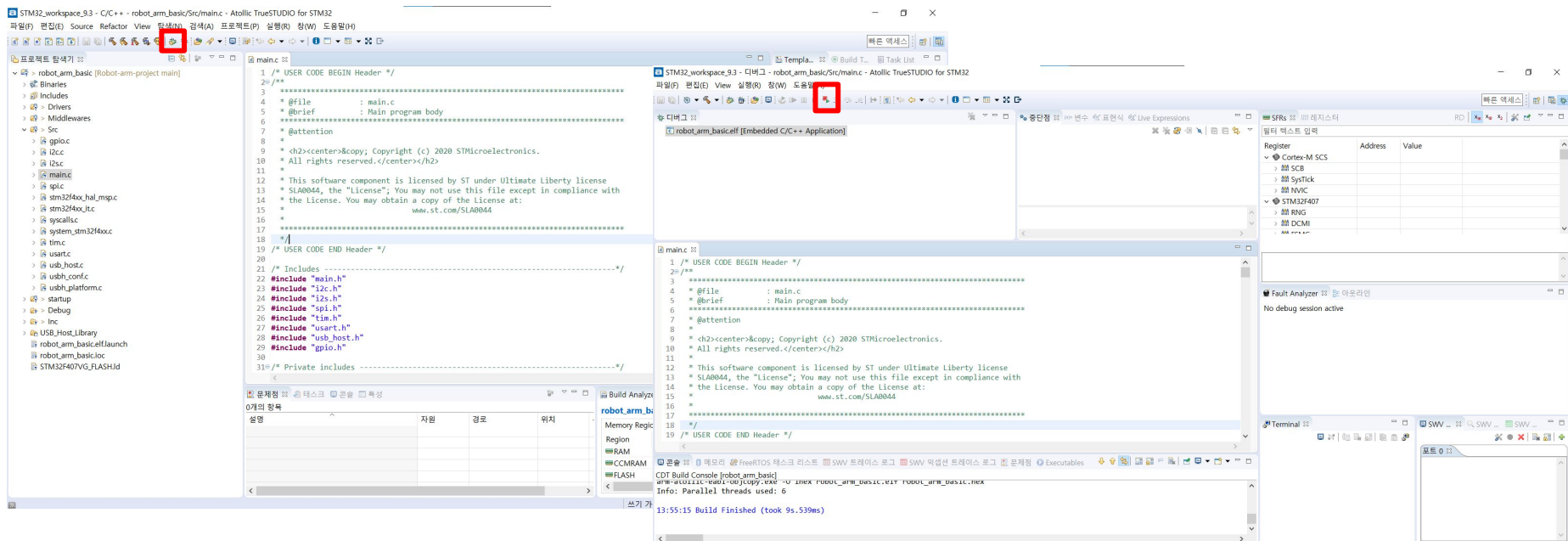
4. 'Open Project' 클릭 -> 생성완료

메인 코드 위치 : Src/main.c



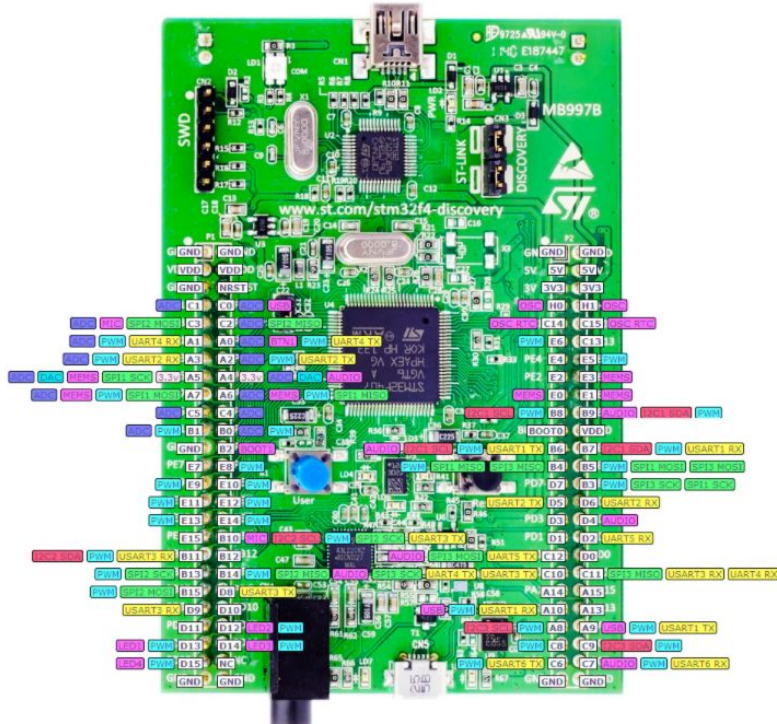
3. Project - Debug

- 프로젝트 열기 : **.cproject** 파일 열기
- 디버깅 : 디버그 아이콘 클릭 후 다음 화면에서 정지 아이콘 클릭

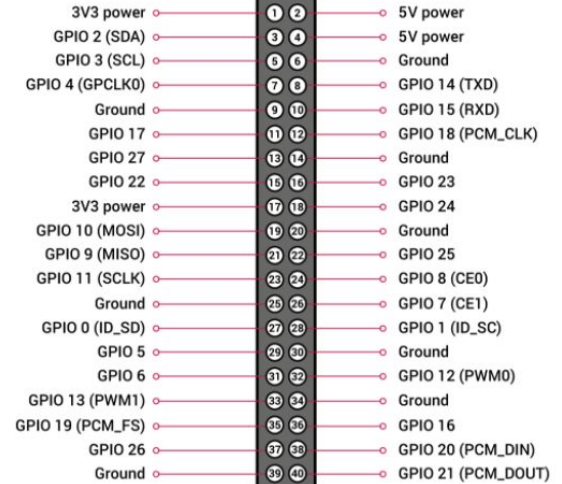
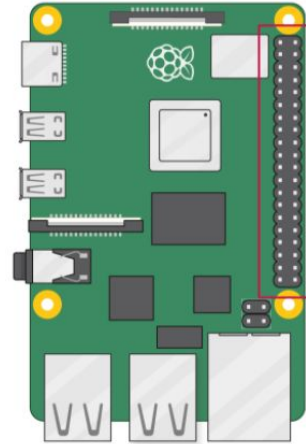


4. Self-Driving Control - Pin Configuration Diagram

- STM32407G



- 라즈베리파이4



4. Self-Driving Control - Mecanum Wheel Control

- PWM Start

HAL_TIM_PWM_Start(timer number, channel number)

```
HAL_TIM_PWM_Start(&htim1, TIM_CHANNEL_1);
```

```
HAL_TIM_PWM_Start(&htim1, TIM_CHANNEL_2);
```

```
HAL_TIM_PWM_Start(&htim1, TIM_CHANNEL_3);
```

```
HAL_TIM_PWM_Start(&htim1, TIM_CHANNEL_4);
```

4. Self-Driving Control - Mecanum Wheel Control

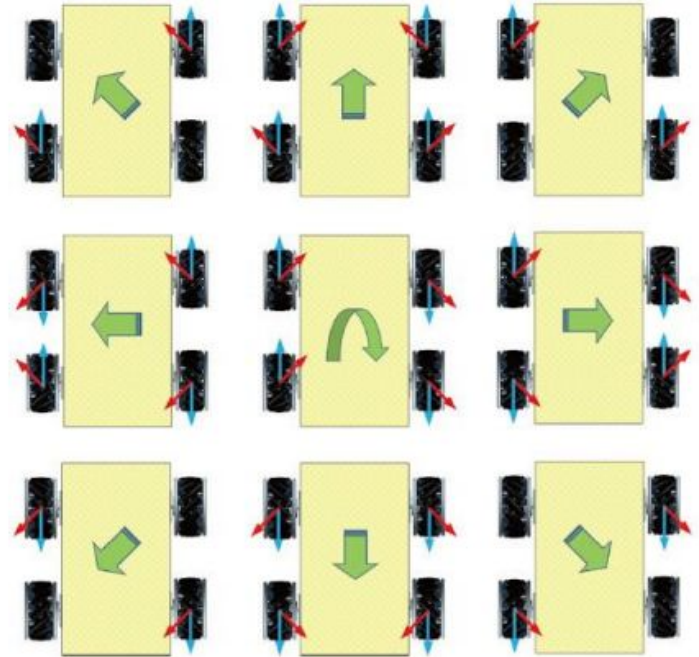
- 방향 제어

SET, RESET -> 정방향 회전

RESET, SET -> 역방향 회전

EX) 4바퀴 모두 정방향 회전

```
HAL_GPIO_WritePin(GPIOB, GPIO_PIN_12, SET);  
  
HAL_GPIO_WritePin(GPIOB, GPIO_PIN_13, RESET);  
  
HAL_GPIO_WritePin(GPIOB, GPIO_PIN_14, SET);  
  
HAL_GPIO_WritePin(GPIOB, GPIO_PIN_15, RESET);  
  
HAL_GPIO_WritePin(GPIOD, GPIO_PIN_8, SET);  
  
HAL_GPIO_WritePin(GPIOD, GPIO_PIN_9, RESET);  
  
HAL_GPIO_WritePin(GPIOD, GPIO_PIN_10, SET);  
  
HAL_GPIO_WritePin(GPIOD, GPIO_PIN_11, RESET);
```



4. Self-Driving Control - Mecanum Wheel Control

- 속도 제어

0 ~ 9999까지 가능

```
TIM1->CCR1 = 0;    //Left Front
```

```
TIM1->CCR2 = 0;    //Right Front
```

```
TIM1->CCR3 = 0;    // Left Back
```

```
TIM1->CCR4 = 0;    // Right Back
```

4. Self-Driving Control - Lidar Control

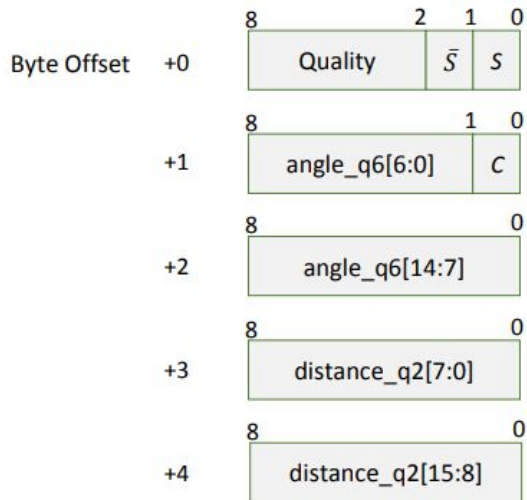
- 라이다 스캔 메커니즘

스캔 시작 명령을 주면 **7bytes** 헤더를 송신하고 **5bytes** 거리 데이터를 연속적으로 송신

* USART 통신의 송수신 기본단위는 8bit

- 스캔 시작

```
HAL_UART_Transmit(&huart3, &scan_command, 2, 100);
```



4. Self-Driving Control - Lidar Control

- 거리 계산 및 속도 변경

```
if(scan_start){  
    if(HAL_UART_Receive(&huart3, &rx3_data, 5, 10) == HAL_OK){  
        Q = rx3_data[0]>>2;  
        S = (rx3_data[0] & 0x01) ? 1 : 0;  
        angle = (rx3_data[2]<<7 | rx3_data[1]>>1)/64;  
        d = (rx3_data[4]<<8 | rx3_data[3])/4;  
        if(d >= 5000){  
            distance[angle] = 5000;  
        }  
        else{  
            distance[angle] = d;  
        }  
        if(S == 1){  
            avg_DIFF = array_avg_compare(distance);  
            MOTER_PWM[0] = PID_speed + avg_DIFF;  
            MOTER_PWM[1] = PID_speed - avg_DIFF;  
            MOTER_PWM[2] = PID_speed + avg_DIFF;  
            MOTER_PWM[3] = PID_speed - avg_DIFF;  
            memset(distance, 0, 360);  
        }  
    }  
    else{  
        if(HAL_UART_Receive(&huart3, &rx3_start, 7, 10) == HAL_OK){  
            if (array_element_of_index_equal(rx3_start, scan_response, 7)){  
                scan_start = true;  
            }  
        }  
    }  
}
```

1. 거리 값 저장
2. 360도 스캔 완료
3. 헤더 검사

4. Self-Driving Control - Lidar Control

- 평균값 계산

```
int16_t array_avg_compare(uint16_t distance[]){  
    uint32_t sum_R = 0;  
    uint32_t sum_L = 0;  
    uint8_t len_L = 0;  
    uint8_t len_R = 0;  
    uint16_t avg_R = 0;  
    uint16_t avg_L = 0;  
    int16_t avg_diff = 0;  
  
    for(int i=0; i<90; i++){  
        sum_R += distance[i];  
        if(distance[i]!=0){  
            len_R++;  
        }  
    }  
    avg_R = sum_R/len_R;  
  
    for(int i=270; i<360; i++){  
        sum_L += distance[i];  
        if(distance[i]!=0){  
            len_L++;  
        }  
    }  
    avg_L = sum_L/len_L;  
  
    avg_diff = avg_R - avg_L;  
  
    return avg_diff;  
}
```

4. Self-Driving Control -PID Control

- PID 제어 : 모터에 인가해준 목표 속도와 엔코더로 측정한 실제 측정 속도의 차를 이용하여 목표 속도에 빠르고 정확하게 도달하기 위한 기법
- 엔코더 특정 값을 속도로 변환 : $speed = 164.18 * \exp(0.01112 * encoder)$
- PID 계수 튜닝 후 튜닝한 속도값을 모터에 인가

```
uint32_t desired_speed = 3000;

void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim) //Timer interrupt every 20ms
{
    if(htim->Instance == TIM6){
        //HAL_GPIO_WritePin(GPIOC,GPIO_PIN_13,GPIO_PIN_SET);
        encoder_cnt[0] = TIM2->CNT;
        TIM2->CNT=0;

        encoder_cnt[1] = TIM3->CNT;
        TIM3->CNT=0;

        encoder_cnt[2] = TIM4->CNT;
        TIM4->CNT=0;

        encoder_cnt[3] = TIM5->CNT;
        TIM5->CNT=0;

        encoder_speed[0] = 164.18 * exp(0.0112*encoder_cnt[0]);
        encoder_speed[1] = 164.18 * exp(0.0112*encoder_cnt[1]);
        encoder_speed[2] = 164.18 * exp(0.0112*encoder_cnt[2]);
        encoder_speed[3] = 164.18 * exp(0.0112*encoder_cnt[3]);

        error_speed[0] = desired_speed - encoder_speed[0];
        error_speed[1] = desired_speed - encoder_speed[1];
        error_speed[2] = desired_speed - encoder_speed[2];
        error_speed[3] = desired_speed - encoder_speed[3];

        PID_speed[0] = old_PID_speed[0] + Kp*error_speed[0];
        PID_speed[1] = old_PID_speed[1] + Kp*error_speed[1];
        PID_speed[2] = old_PID_speed[2] + Kp*error_speed[2];
        PID_speed[3] = old_PID_speed[3] + Kp*error_speed[3];

        old_PID_speed[0] = PID_speed[0];
        old_PID_speed[1] = PID_speed[1];
        old_PID_speed[2] = PID_speed[2];
        old_PID_speed[3] = PID_speed[3];
    }
}
```

5. Serial Communication