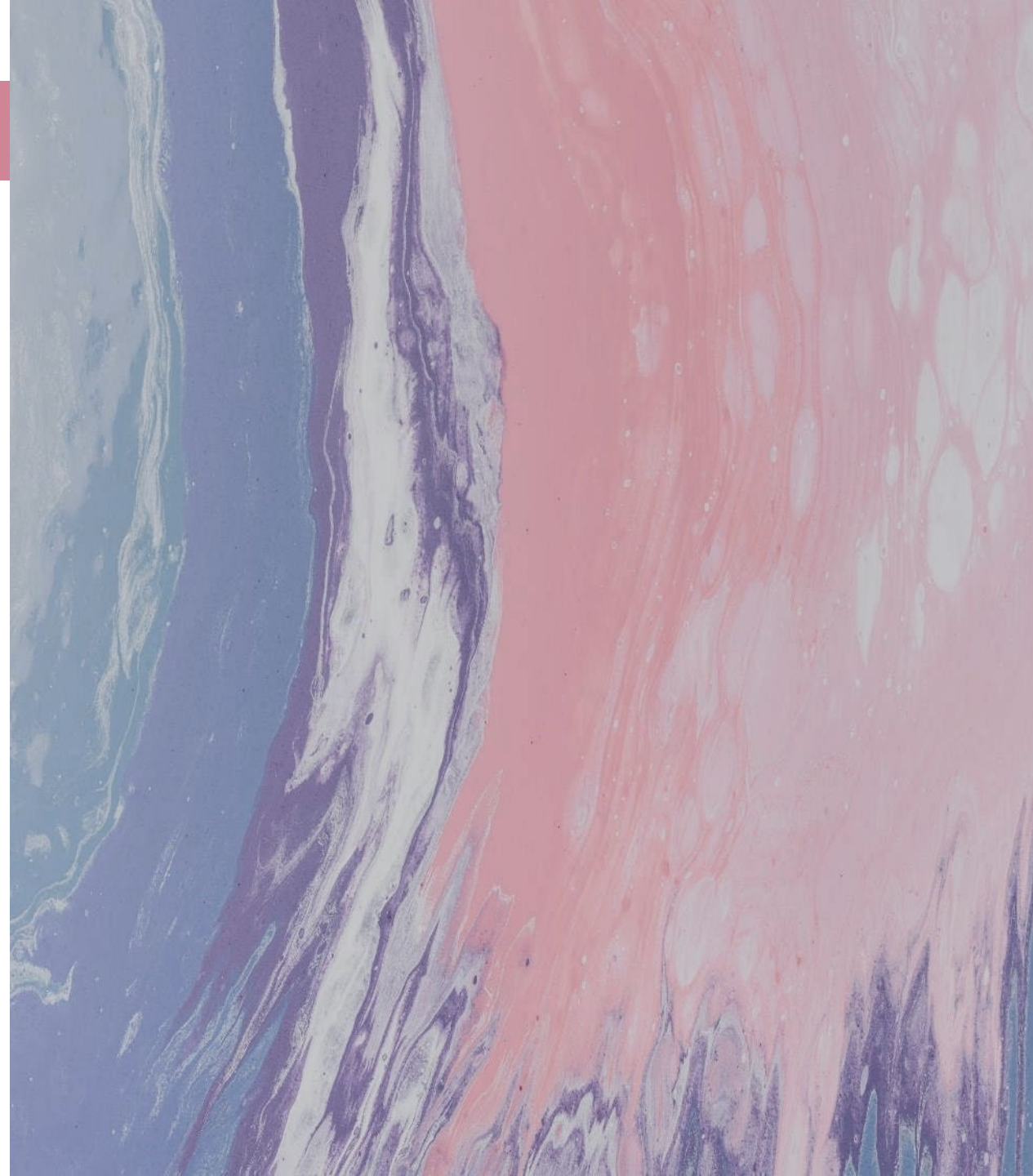
스마트 보안 시스템

- 팀명 : B팀
- 장성혁, 이호열
- 오세현, 한호석
- 정현규, 최지호
- 이이랑, 오두환

동의대학교 로봇·자동화공학

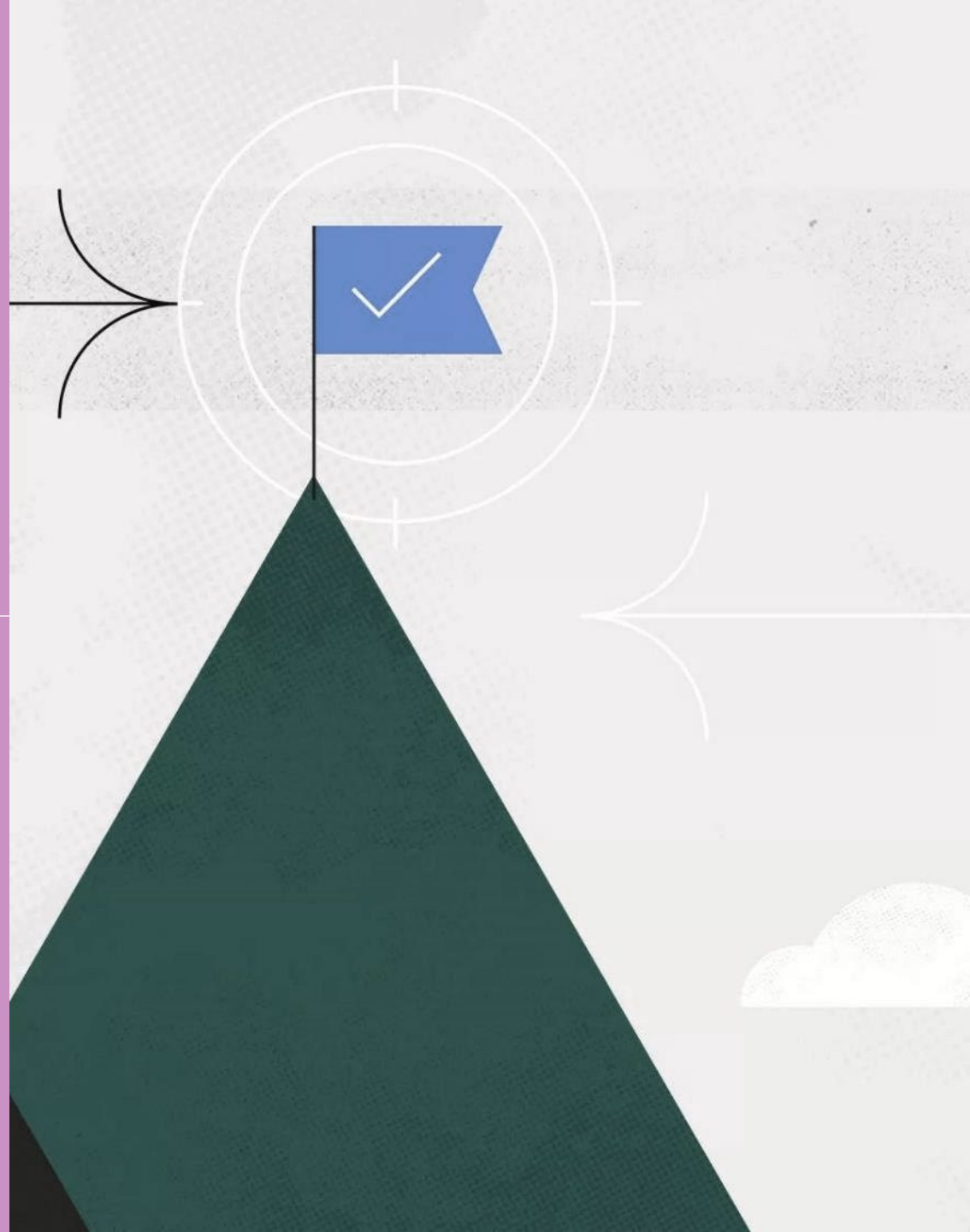
목차 a table of contents



Part 1,

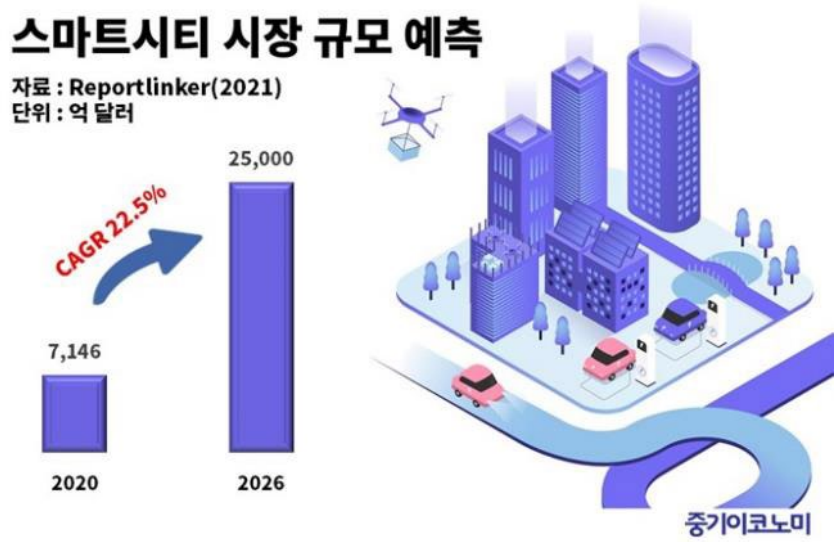
프로젝트 개요

- 프로젝트 선정배경
- 프로젝트 목표



Part 1,
프로젝트 선정 배경

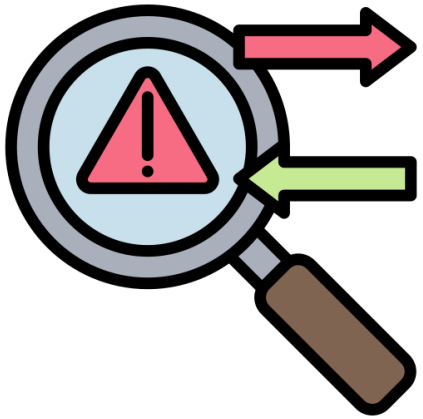
[포커스온] 편리한 스마트시티 속 커지는 보안 위협



- IOT를 활용한 스마트 시티의 증가로 인한 보안 시스템의 필요성이 증가
- 사용자가 집에 없는 상황에서도 원격으로 제어가 가능해야함
- 급속히 변화하는 환경에 맞추어 보안 체계도 스마트하게 변해야함

Part 1,
프로젝트 목표

보안강화



위험상황 감지

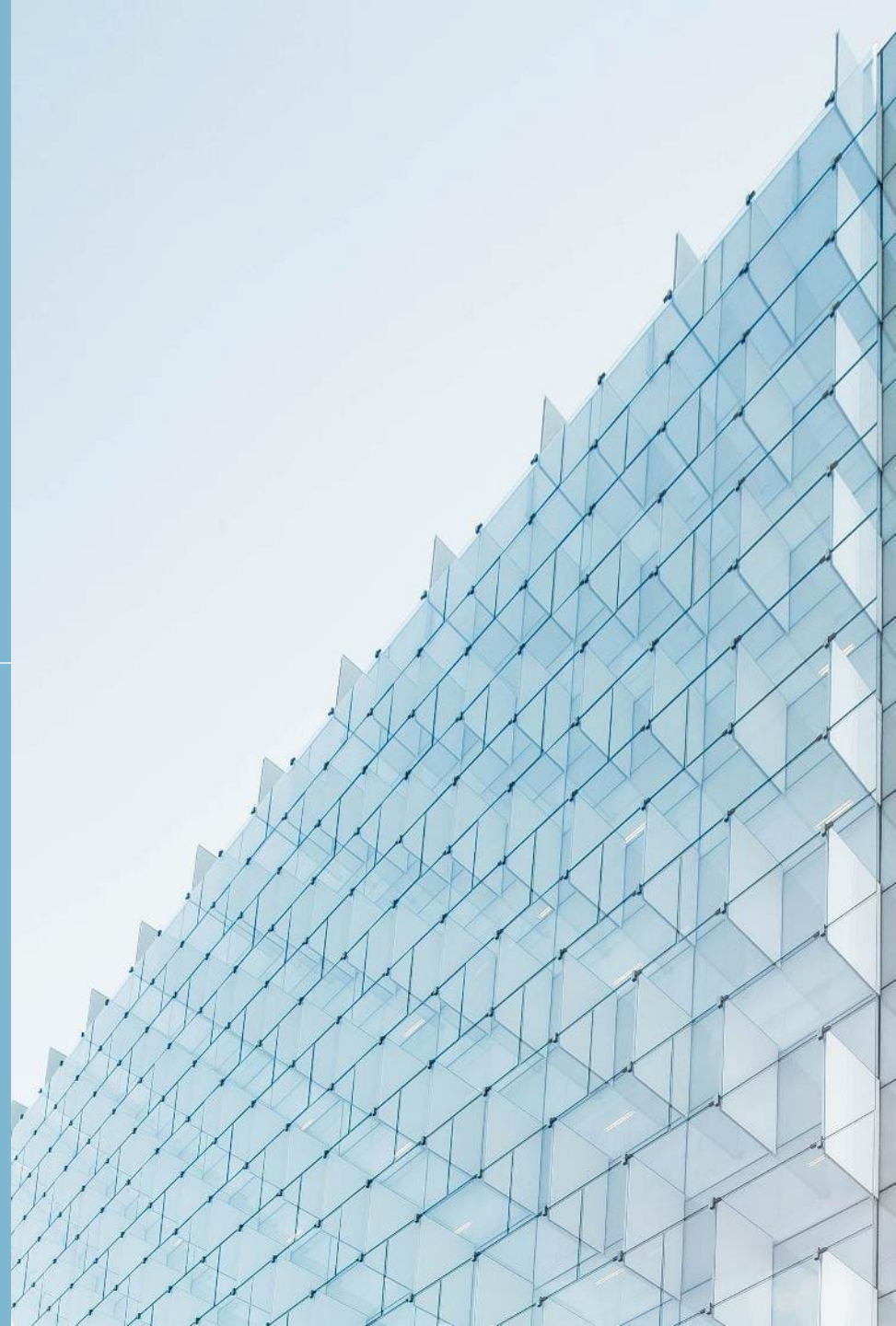
안전성 확보

- 보안 강화와 위험 상황 감지를 통한 안전성 확보
- 여러 센서를 사용하여 출입 감지를 통해 경보 및 알림을 울림
- 최종적으로 센서 퓨전을 통한 학습 능력 강화

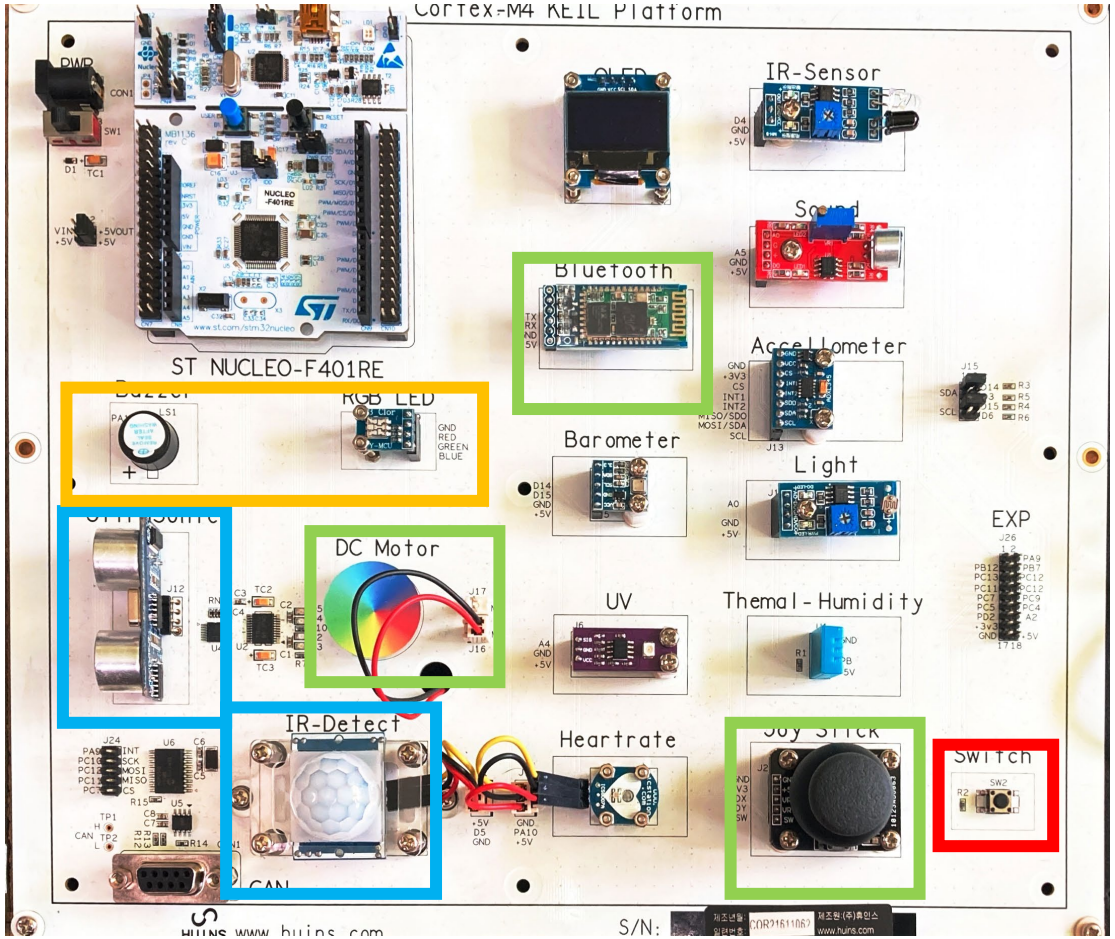
Part 2,

프로젝트 소개

- 구현 방법 및 사용센서
- 핀맵 구성도
- 프로젝트 플로우 차트
- STM32CubeIDE 설정
- 케일 코드 분석
- 동작 영상



Part 2,
구현 방법 및 사용센서



제어 및 통신 : 블루투스, 조이스틱 → DC 모터 제어

출입 감지 : 초음파 센서, 적외선 감지기 or 센서

경보 및 알림 : 부저, RGB LED

전체 시스템 ON/OFF : 스위치

Part 2,
프로젝트 역할 분담

▪ 장성혁, 이호열

블루투스 & 조이스틱 → DC 모터 제어 (문단속)

▪ 오세현, 한호석

스위치 → 시스템 ON/OFF OFF일 때 다른 시스템이 사용 X

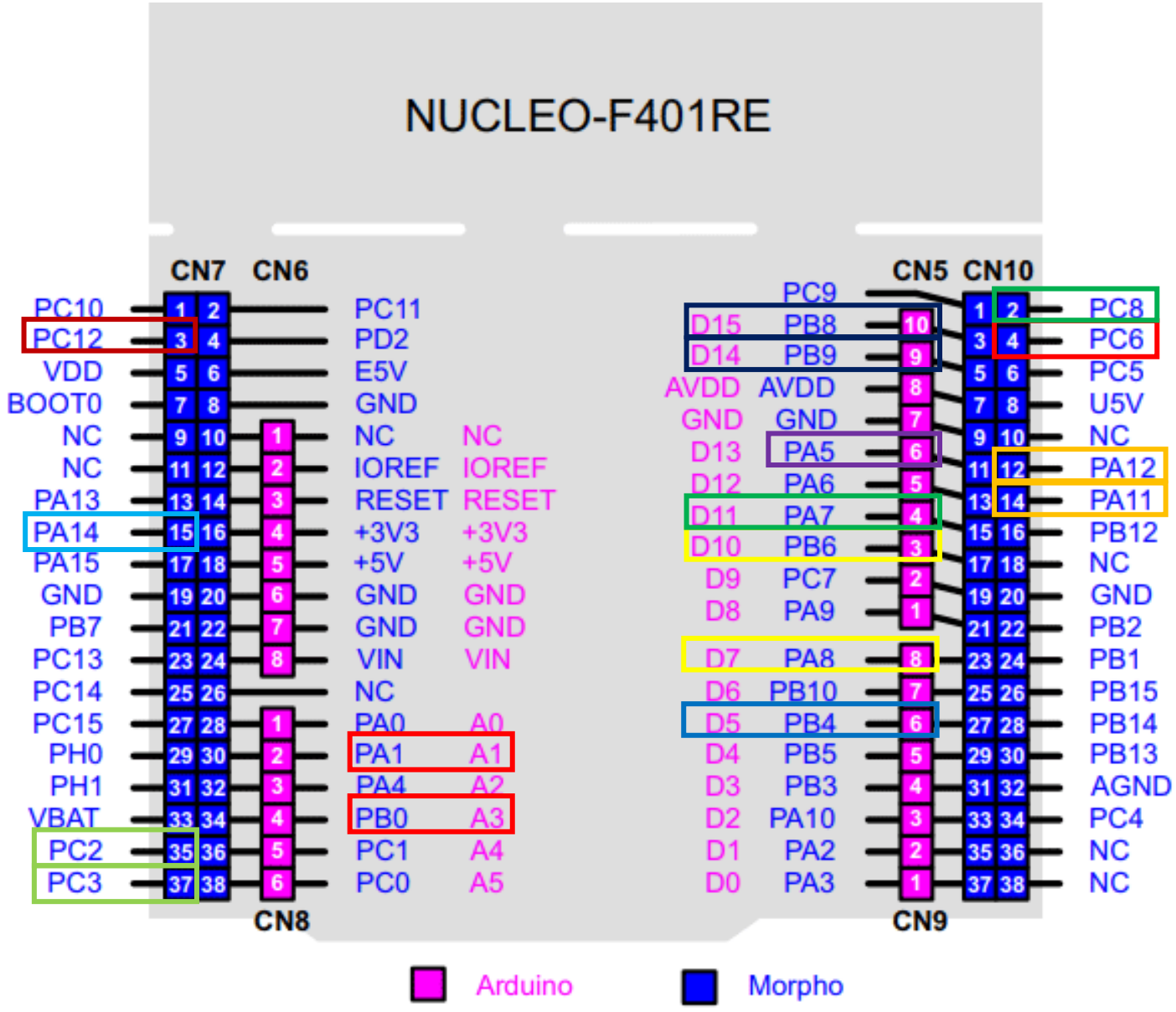
▪ 정현규, 최지호

적외선 감지기 → 감지되면 외부 LED 동작

▪ 이이랑, 오두환

초음파 센서 → 감지되면 RGB led와 부저 경고

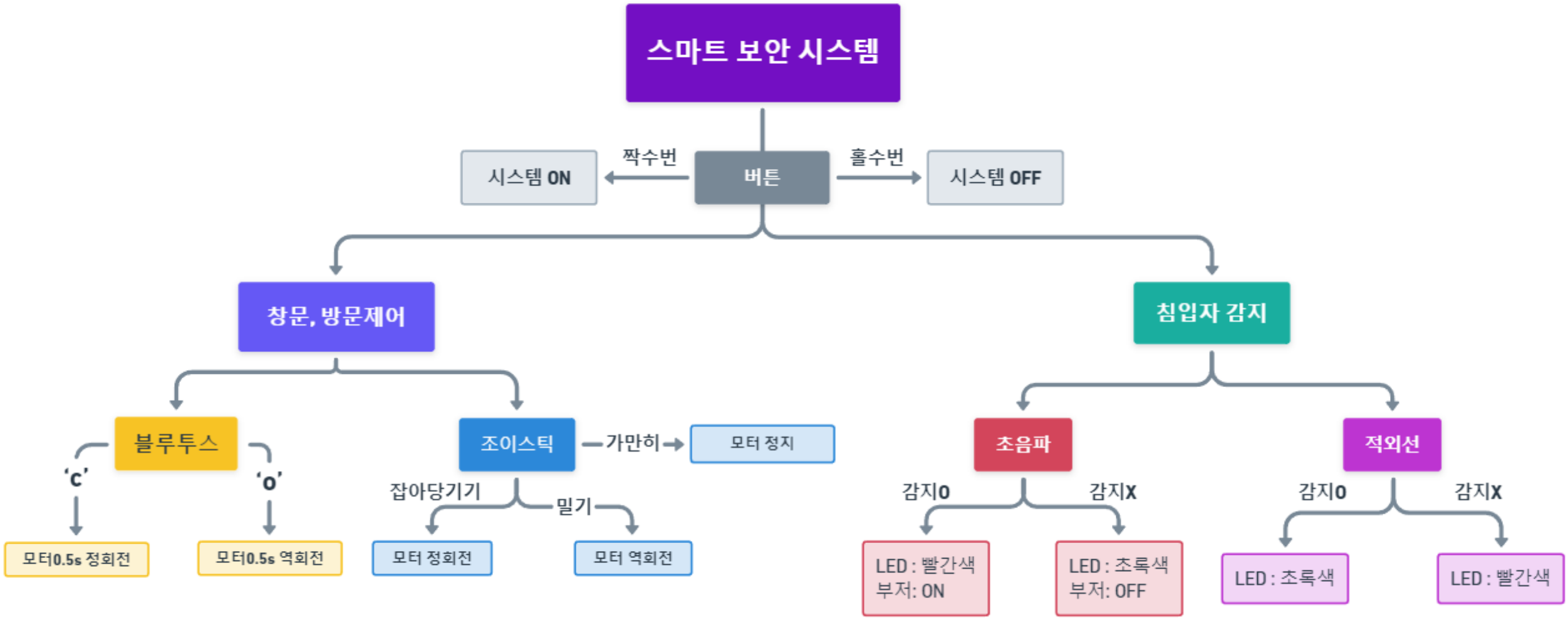
Part 2,
핀맵 구성도



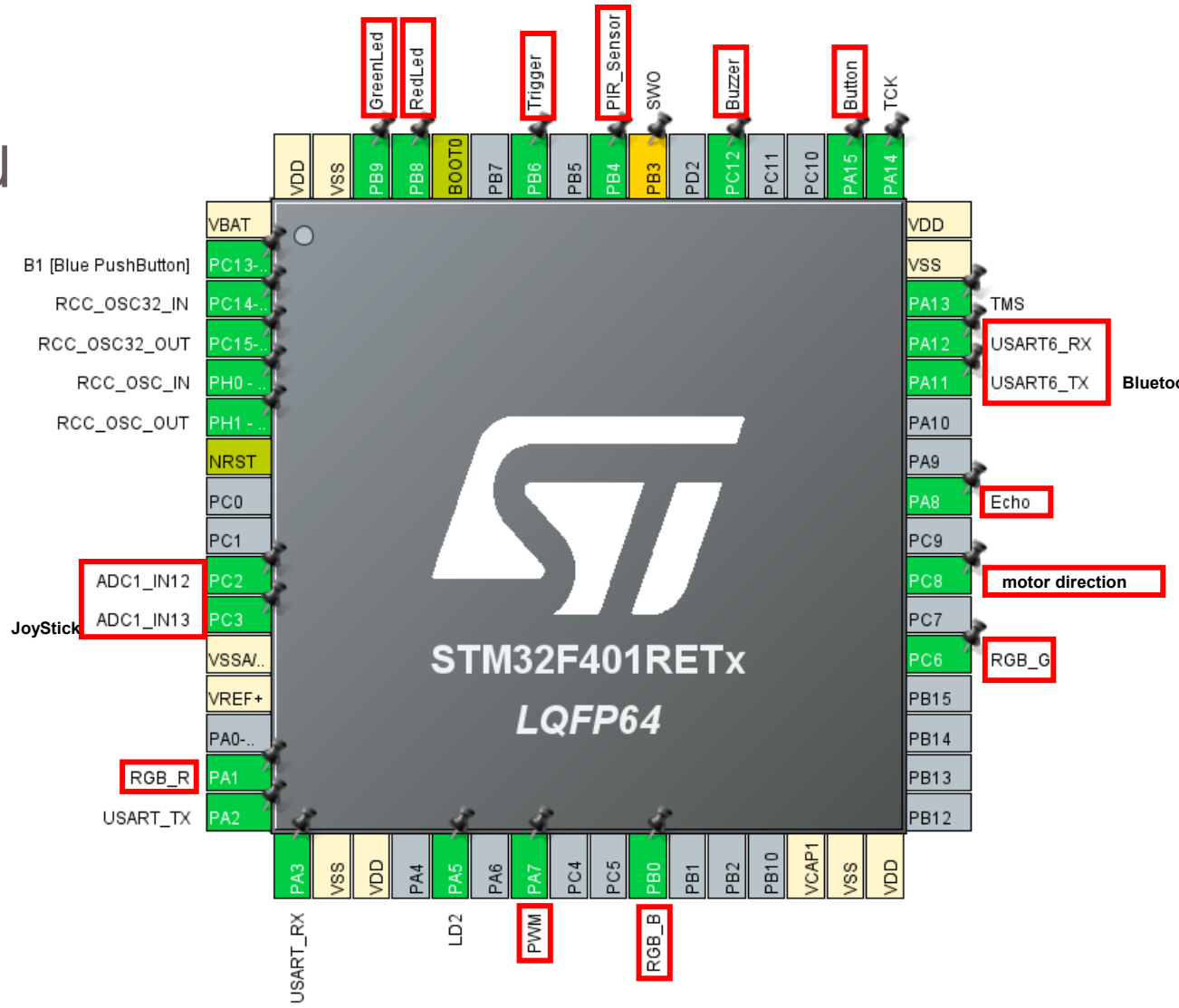
- RGB(DigitalOut) x 3
- Bluetooth(Serial) x 1 x 2
- JoyStick(AnalogIn) x 2
- Button(DigitalIn, PullUp) x 1
- In_LED(DigitalOut) x 1
- Ultra(Digital_X) x 1 x 2
- Buzzer(DigitalOut) x 1
- Motor(pwm, dir) x 1 x 2
- Ex_Led(DigitalOut) x 2
- PIR_Sensor(DigitalIn) x 1

Part 2,

프로젝트 플로우 차트

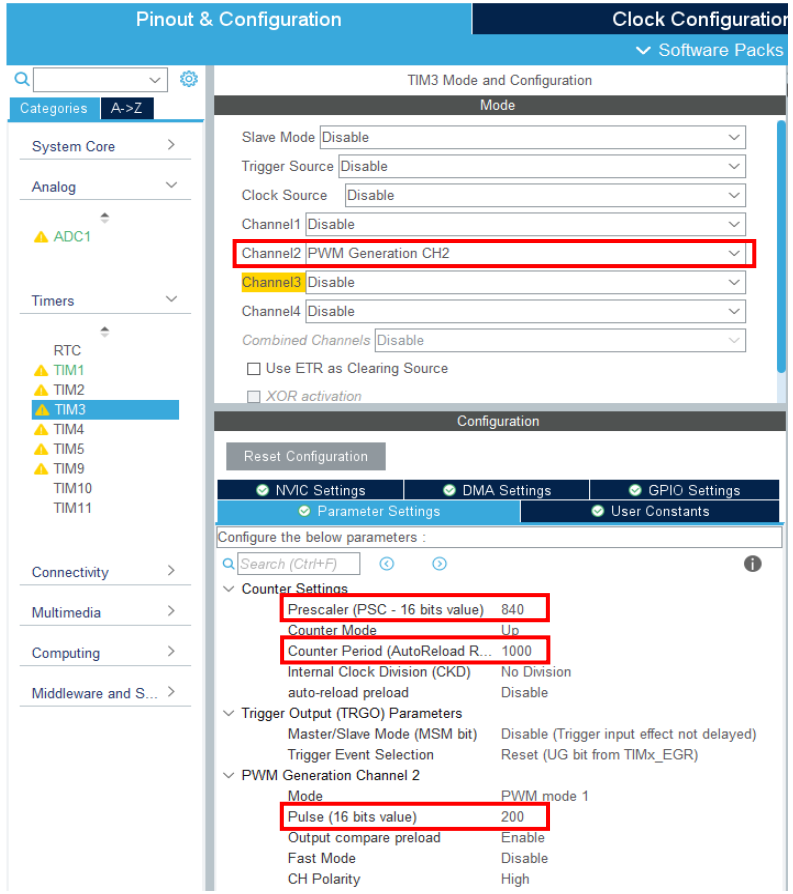


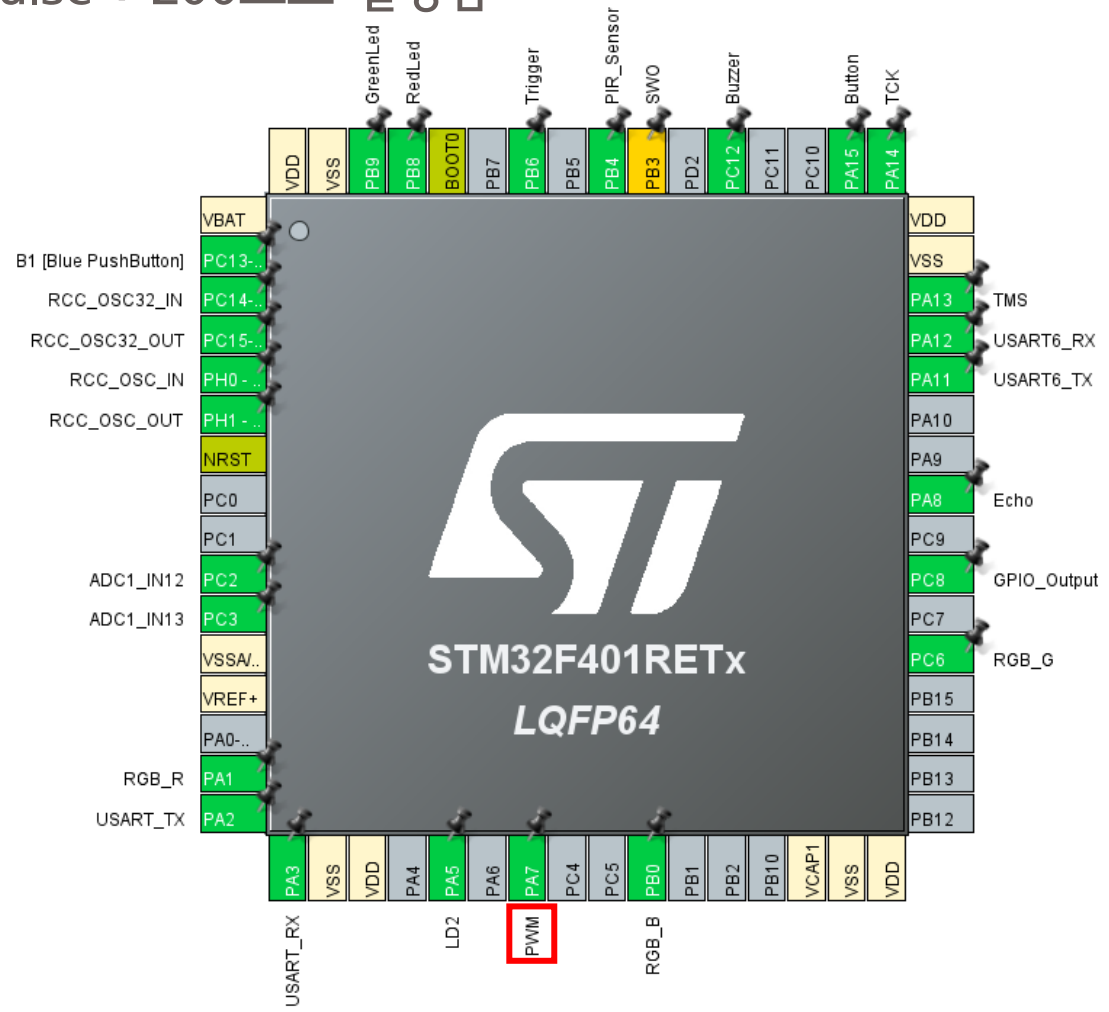
- 핀맵 설정
  - Bluetooth
  - 적외선 센서를 위한 Green, Red Led
  - 시스템 On/Off를 위한 Button
  - 경고를 위한 RGB와 부저
  - DC모터 제어를 위한 PWM
  - DC 모터 방향을 위한 Direction
  - PIR_Sensor
  - 초음파 감지를 위한 Echo, Trigger
  - 조이스틱 값 측정을 위한 ADC



Part 2,
Stm32CubeIDE 설정

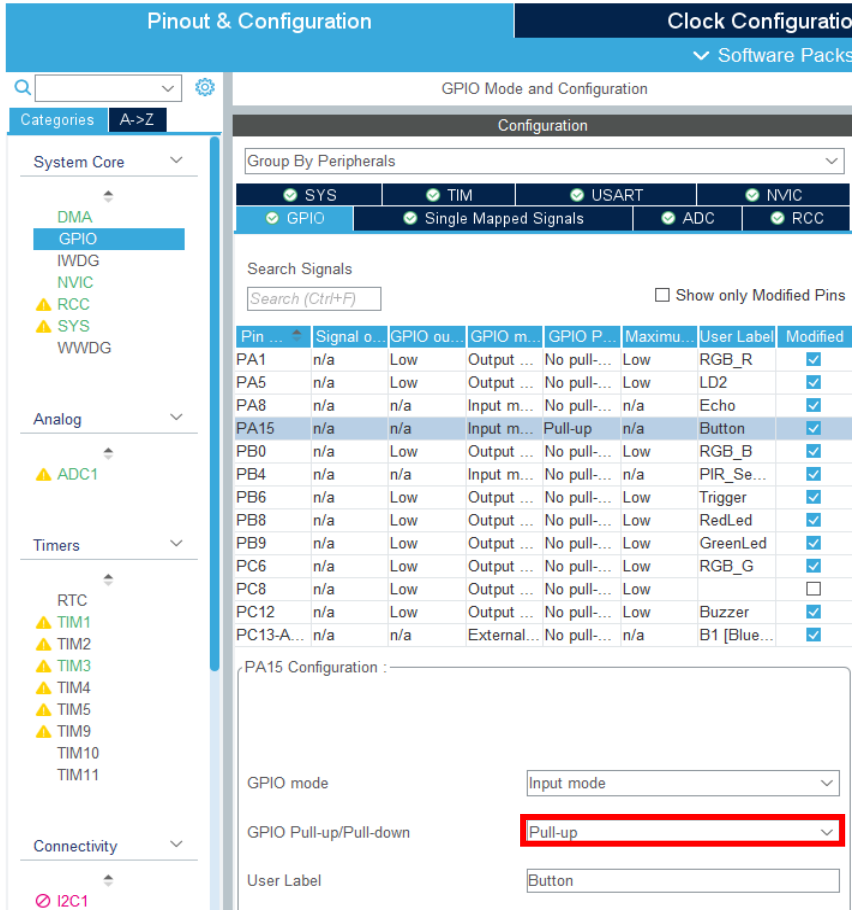
- TIM3 PWM 설정
  - Channel2를 PWM Generation CH2로 설정
  - PreScaler : 840, Counter Period : 1000, Pulse : 200으로 설정함

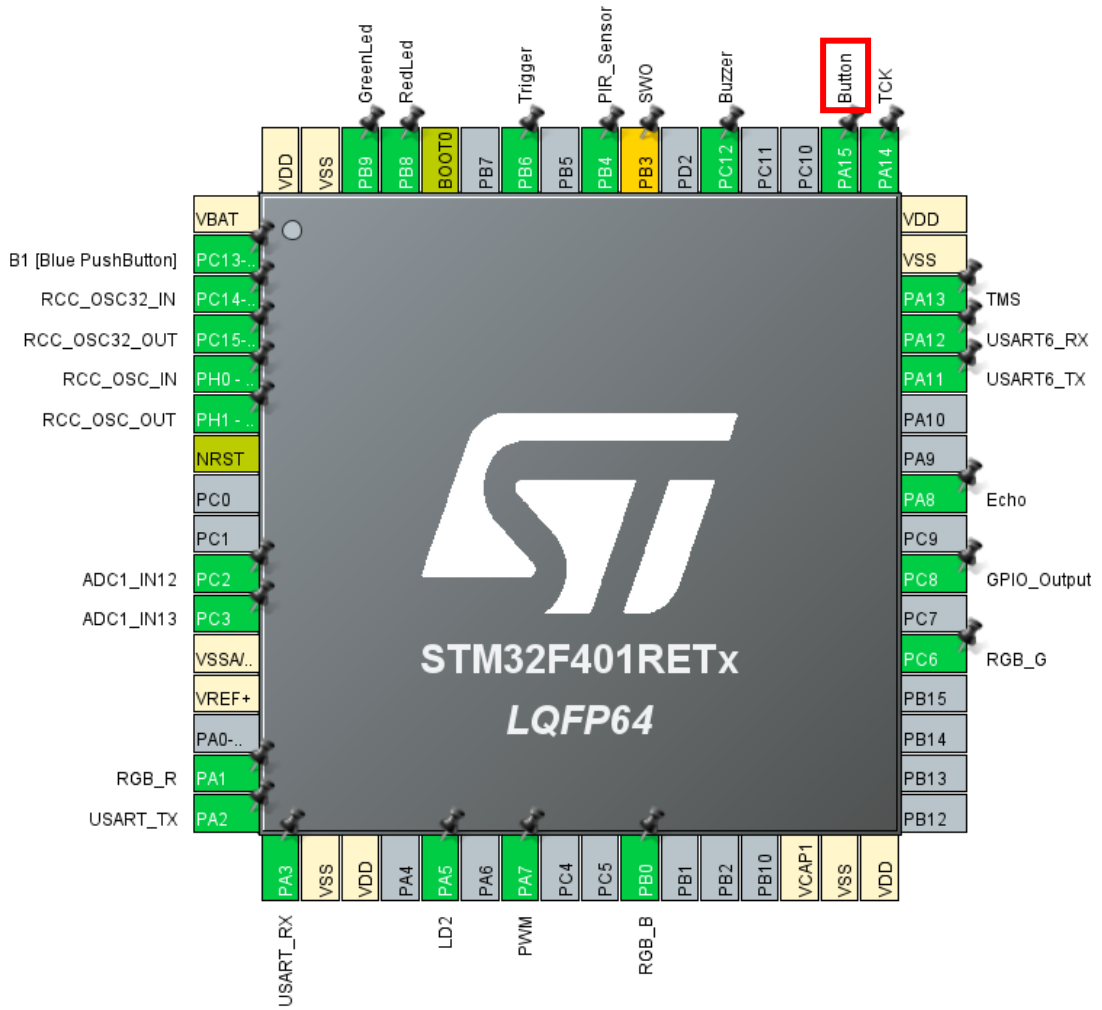




Part 2,
Stm32CubeIDE 설정

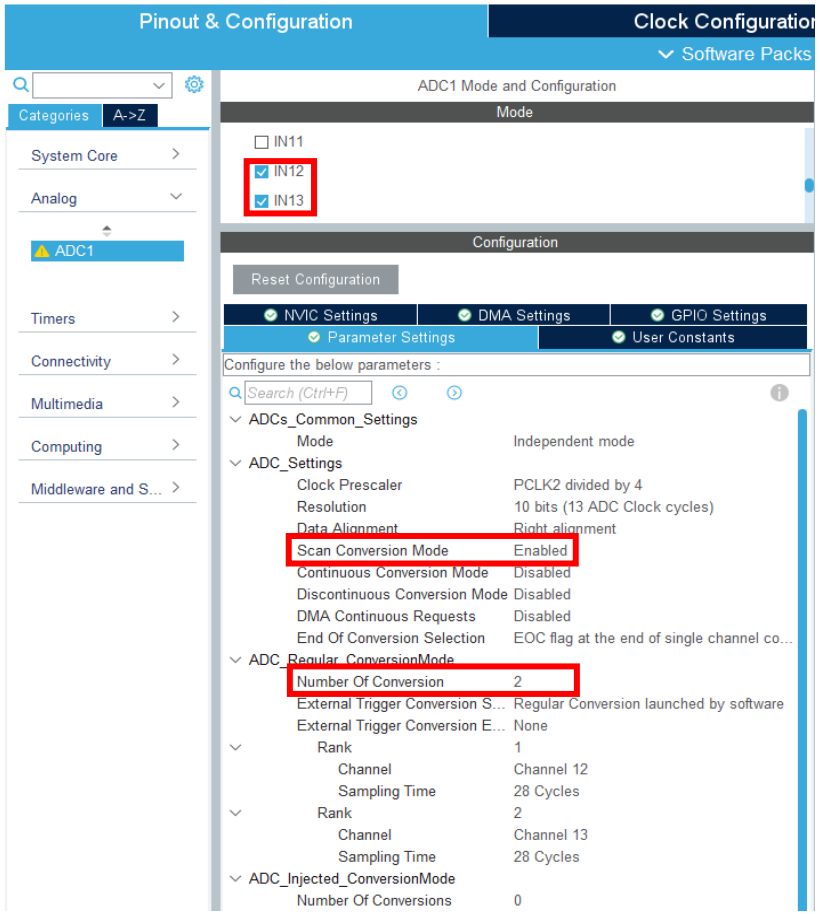
- 버튼 내부 풀업 설정
  - 시스템 제어를 위한 버튼을 외부에서 연결 후 내부 풀업 설정함

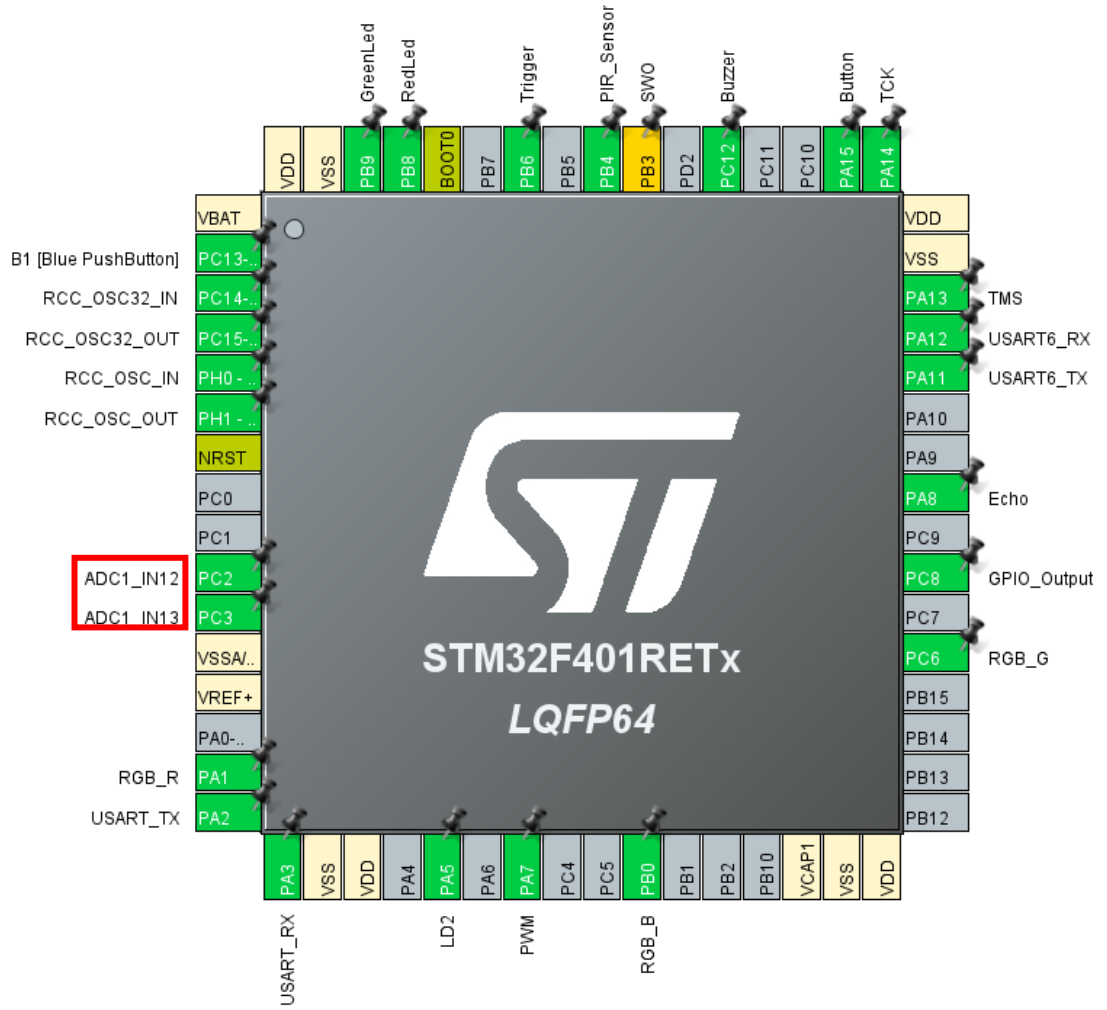




Part 2,
Stm32CubeIDE 설정

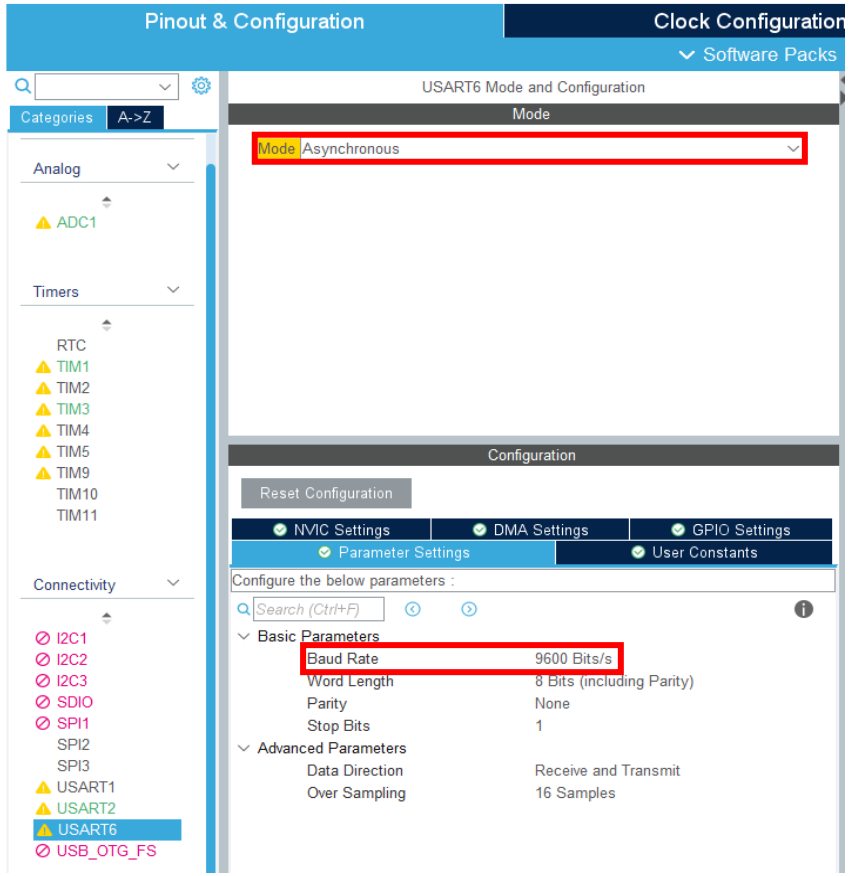
- ADC 설정
- 조이스틱의 x, y 값 측정을 위해 ADC 2개를 사용, Pooling 방식을 사용함으로 Scan Mode 만 Enable함

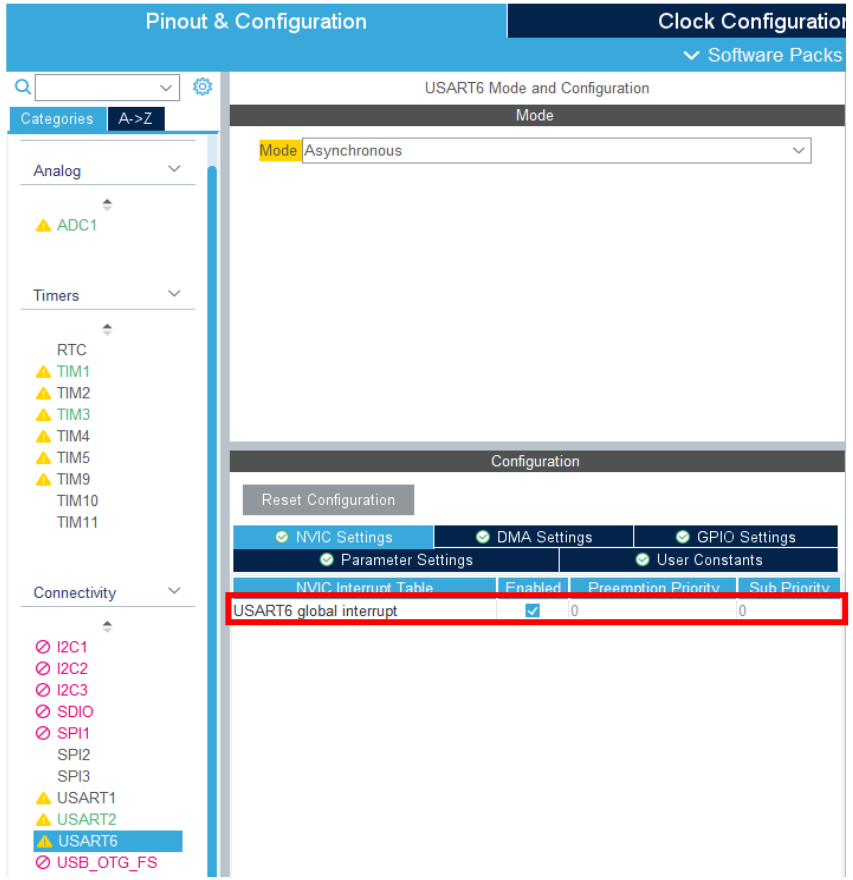




Part 2,
Stm32CubeIDE 설정

- USART6 Bluetooth 설정
  - Mode를 Asynchronous, Baud Rate를 9600 Bit/s, global interrupt 설정함





Part 2,
Stm32CubeIDE 코드 분석

- 코드 분석
  - 시리얼 출력을 위한 __in_putchat 정의
  - 초음파 변수 정의
  - 적외선 감지 변수 정의
  - 버튼을 통한 시스템 제어를 위한 변수 정의
  - 시리얼 통신 변수 정의
  - 조이스틱 변수 정의

```
#ifdef __GNUC__
#define PUTCHAR_PROTOTYPE int __io_putchar(int ch)
#else
#define PUTCHAR_PROTOTYPE int fputc(int ch, FILE *f)
#endif
PUTCHAR_PROTOTYPE
{
    HAL_UART_Transmit(&huart2, (uint8_t *)&ch, 1, 0xFFFF);
    return ch;
}

/* ultra sonic variable */
#define TRIG_PIN GPIO_PIN_6
#define TRIG_PORT GPIOB
#define ECHO_PIN GPIO_PIN_8
#define ECHO_PORT GPIOA

uint32_t pMillis;
uint32_t Value1 = 0;
uint32_t Value2 = 0;
uint16_t Distance  = 0;
int cnt = 0;

/* pir sensor variable variable */
int Pir_value = 0;

/* button variable */
int button_state = 0;
int state = 0;
int last_button_state = 1;

/* bluetooth and serial */
uint8_t rx6_data;
uint8_t rx2_data;

/* joystick */
uint32_t joy_value[2];
```

Part 2,
Stm32CubeIDE 코드 분석

- 코드 분석
  - 블루투스(USART6)를 연결함
  - c(99, close)를 보드로 보냄
  - 모터가 1초 동안 CW로 회전함
  - o(111,open)을 보드로 보냄
  - 모터가 1초동안 CCW로 회전함

```
void HAL_UART_RxCpltCallback(UART_HandleTypeDef *huart) {
    if(huart -> Instance == USART6){
        if(rx6_data == 111) {
            HAL_GPIO_WritePin(GPIOC, GPIO_PIN_8, 1);
            htim3.Instance->CCR2 = 200;
            for(int i=0; i<10000000; i++){
            }
        }
        else if(rx6_data == 99) {
            HAL_GPIO_WritePin(GPIOC, GPIO_PIN_8, 0);
            htim3.Instance->CCR2 = 200;
            for(int i=0; i<10000000; i++){
            }
        }
        HAL_UART_Receive_IT(&huart6, &rx6_data, 1);
    }
}
```

Part 2,
Stm32CubeIDE 코드 분석

- 코드 분석
  - UART6(블루투스), USART2(시리얼) 초기화
  - PWM_Timer3_channel2 초기화
  - PWM Pulse 초기화
  - 초음파 트리거 핀 초기화
  - 초음파 시간 측정을 위한 Tim1 초기화

```c
int main(void)
{

  HAL_Init();

  SystemClock_Config();

  MX_GPIO_Init();
  MX_DMA_Init();
  MX_USART2_UART_Init();
  MX_ADC1_Init();
  MX_USART6_UART_Init();
  MX_TIM1_Init();
  MX_TIM3_Init();
  /* USER CODE BEGIN 2 */

  HAL_TIM_Base_Start(&htim1);
  HAL_GPIO_WritePin(TRIG_PORT, TRIG_PIN, GPIO_PIN_RESET);
  HAL_UART_Receive_IT(&huart6, &rx6_data,1);
  HAL_UART_Receive_IT(&huart2, &rx2_data,1);
  HAL_TIM_PWM_Start(&htim3, TIM_CHANNEL_2);
  htim3.Instance->CCR2 = 0;
  /* USER CODE END 2 */

  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
```

Part 2,
Stm32CubeIDE 코드 분석

- 코드 분석
  - 조이스틱 값을 ADC Pooling 을 통해 측정 후 저장함
  - 조이스틱을 왼쪽으로 움직임(0~10 범위)
  - 모터가 CCW로 움직임
  - 조이스틱을 오른쪽으로 움직임(900~1024 범위)
  - 모터가 CW로 움직임
  - 조이스틱을 가만히 둠 (400~600 범위)
  - 모터가 멈춤

```
while (1)
{
  HAL_ADC_Start(&hadc1);
  HAL_ADC_PollForConversion(&hadc1, 10);
  joy_value[0] = HAL_ADC_GetValue(&hadc1);
  HAL_ADC_Start(&hadc1);
  HAL_ADC_PollForConversion(&hadc1, 10);
  joy_value[1] = HAL_ADC_GetValue(&hadc1);
  HAL_ADC_Stop(&hadc1);

  /* joy stick */

  if(joy_value[1] < 10) {
      if(state%2 == 0) {
          HAL_GPIO_WritePin(GPIOC, GPIO_PIN_8, 1);
          htim3.Instance->CCR2 = 200;
      }
  }

  if(joy_value[1] > 900) {
      if(state%2 == 0) {
          HAL_GPIO_WritePin(GPIOC, GPIO_PIN_8, 0);
          htim3.Instance->CCR2 = 200;
      }
  }

  if (joy_value[0]>450 && joy_value[0]<550 && joy_value[1]>400 && joy_value[1]<600){
      htim3.Instance->CCR2 = 0;
  }
```

Part 2,
Stm32CubeIDE 코드 분석

- 코드 분석
- Distance = (Value2-Value1)*0.034 / 2로 센서와 물체간의 거리를 구함

- 만약 Distance가 10cm 미만일 경우 HAL_GPIO_WritePin이 1ms마다 SET,RESET을 반복하며 500Hz의 진동수로 소리를 발생 시킴

```
/* ultra sonic to buzzer */
HAL_GPIO_WritePin(TRIG_PORT, TRIG_PIN, GPIO_PIN_SET);  // TRIG_PORT와 TRIG_PIN에 해당
__HAL_TIM_SET_COUNTER(&htim1, 0);  // 타이머 1의 카운터 값을 0으로 설정
while (__HAL_TIM_GET_COUNTER (&htim1) < 10);
HAL_GPIO_WritePin(TRIG_PORT, TRIG_PIN, GPIO_PIN_RESET);
pMillis = HAL_GetTick();
while (!(HAL_GPIO_ReadPin (ECHO_PORT, ECHO_PIN)) && pMillis + 10 >  HAL_GetTick());
Value1 = __HAL_TIM_GET_COUNTER (&htim1);
pMillis = HAL_GetTick();
while ((HAL_GPIO_ReadPin (ECHO_PORT, ECHO_PIN)) && pMillis + 50 > HAL_GetTick());
Value2 = __HAL_TIM_GET_COUNTER (&htim1);
Distance = (Value2-Value1)* 0.034/2;
HAL_GPIO_WritePin(GPIOA, GPIO_PIN_1, RESET);
HAL_GPIO_WritePin(GPIOC, GPIO_PIN_6, SET);
HAL_GPIO_WritePin(GPIOB, GPIO_PIN_0, RESET);
if(state%2 == 1){
    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_1, RESET);
    HAL_GPIO_WritePin(GPIOC, GPIO_PIN_6, RESET);
    HAL_GPIO_WritePin(GPIOB, GPIO_PIN_0, RESET);
}
printf("%d\r\n", Distance);
if (Distance < 10) //거리가 10cm이하일 때
{

    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_1, SET);
    HAL_GPIO_WritePin(GPIOC, GPIO_PIN_6, RESET);
    if(state%2 == 1)
    {
        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_1, RESET);
        HAL_GPIO_WritePin(GPIOC, GPIO_PIN_6, RESET);
    }
    while(cnt < 200) //약 1초동안 부저 울림
    {
        cnt++;
        if(state%2 == 0) {
            HAL_GPIO_WritePin(GPIOC, GPIO_PIN_12, SET);
            HAL_Delay(1);
            HAL_GPIO_WritePin(GPIOC, GPIO_PIN_12, RESET);
            HAL_Delay(1);
        }
    }
}
cnt = 0;
```

Part 2,
Stm32CubeIDE 코드 분석

- 코드 분석
  - PIR 센서는 사람이 감지하면 1, 평소는 0을 출력
  - 사람이 감지되면 빨간 불이 켜짐
  - 사람이 감지되지 않으면 파란 불이 켜짐

  - 버튼을 누를 시 state가 1씩 증가함
  - state % 2 == 0 일 시 시스템 ON
  - state % 2 == 1 일 시 시스템 OFF

```
/* pir sensor to led */
Pir_value = HAL_GPIO_ReadPin(GPIOB, GPIO_PIN_4);

if(Pir_value == 1) {
    HAL_GPIO_WritePin(GPIOB, GPIO_PIN_8, SET);
    HAL_GPIO_WritePin(GPIOB, GPIO_PIN_9, RESET);
    if(state%2 == 1) {
        HAL_GPIO_WritePin(GPIOB, GPIO_PIN_8, RESET);
        HAL_GPIO_WritePin(GPIOB, GPIO_PIN_9, RESET);
    }
}
else {
    HAL_GPIO_WritePin(GPIOB, GPIO_PIN_8, RESET);
    HAL_GPIO_WritePin(GPIOB, GPIO_PIN_9, SET);
    if(state%2 == 1) {
        HAL_GPIO_WritePin(GPIOB, GPIO_PIN_8, RESET);
        HAL_GPIO_WritePin(GPIOB, GPIO_PIN_9, RESET);
    }
}
```

```
/* button to control system */
button_state = HAL_GPIO_ReadPin(GPIOA, GPIO_PIN_15);
if(last_button_state == 1 && button_state == 0) {
    state++;
}
last_button_state = button_state;
```

Part 2,
케일 코드 분석

```
Serial pc(SERIAL_TX, SERIAL_RX);
Serial bt(PA_11, PA_12);

DigitalOut r(A1);
DigitalOut g(PC_6);
DigitalOut b(A3);

Motor A(D11, PC_8); // pwm, dir

AnalogIn x_axis(PC_2);
AnalogIn y_axis(PC_3);
DigitalIn button(PA_14,PullUp);

DigitalOut myled(LED1);

DigitalOut Trig(D10);
DigitalIn Echo(D7);
DigitalOut buzzer(PA_13);

DigitalIn pir_sensor(D5);
DigitalOut GreenLed(D14);
DigitalOut RedLed(D15);

Timer t;
```

Pin 지정

```
float i;
int x, y;

int state = 0;
int last_button_state = 1;
int button_state = 0;
```

변수 지정

```
/* joystick, bluetooth, UART to Motor */

x = x_axis.read() *1000;
y = y_axis.read() *1000;

if (bt.readable()) {
  char input_key=  bt.putc(bt.getc());

  if(input_key == 'o') {
    myled = 1;
    //bt.printf("Open the window");
    A.forward(0.4f);
    wait_ms(2000);
    A.stop();
  }

  if(input_key == 'c') {
    myled = 0;
    //bt.printf("Close the window");
    A.backward(0.4f);
    wait_ms(2000);
    A.stop();
  }
}
```

```
if(state%2 == 0) {
  if (y<10){
    A.forward(0.4f);
  }

  else if (y>900){
    A.backward(0.4f);
  }
}

if (x>450 && x<550 && y>400 && y<600){
  A.stop();
}
```

- 사용자의 휴대폰으로 ‘o’를 받으면 모터 역회전, ‘c’를 받으면 모터 정회전
- 조이스틱을 밀면 모터 역회전, 당기면 모터 정회전, 가만히 있으면 정지

Part 2,
케일 코드 분석

```
/* ultra sonic to Buzzer */

Trig = 1;
wait(0.00004);
Trig = 0;
int cnt=0;
while(!Echo);

t.reset();
while(Echo);
i = t.read_us();
i = i/58;
r=0;
g=1;
b=0;

if(state%2 == 1) {
  r=0;
  g=0;
  b=0;
}
if(i<10){
  r=1;
  g=0;
  b=0;
  if(state%2 == 1) {
    r=0;
    g=0;
    b=0;
  }
  while(cnt < 500){
    if(state%2 == 0) {
      buzzer=1;
      wait(0.001);
      buzzer=0;
      wait(0.001);
    }
      cnt++;
  }
}
```

```
/* pir sensor to led */
pir_value = pir_sensor.read();
if(pir_value == 1) {
  RedLed = 1;
  GreenLed = 0;

  if(state%2 == 1) {
    RedLed = 0;
    GreenLed = 0;
  }

}
else {
  GreenLed = 1;
  RedLed = 0;

  if(state%2 == 1) {
    RedLed = 0;
    GreenLed = 0;
  }

}

/* button to control system */
button_state = button.read();
if(last_button_state == 1 && button_state == 0) {
  state++;
}
last_button_state = button_state;


}
```

- 초음파 센서를 이용하여 RGB led와 부저 동작
- 적외선 감자기를 이용하여 외부 LED를 동작시킴
- 시스템 외부버튼을 이용하여 전체 시스템을 ON/OFF

감사합니다!