SIMPLE FAN

1. 목표
2. FLOW CHART
3. PIN OVERVIEW
4. 상세 내용
5. 동작

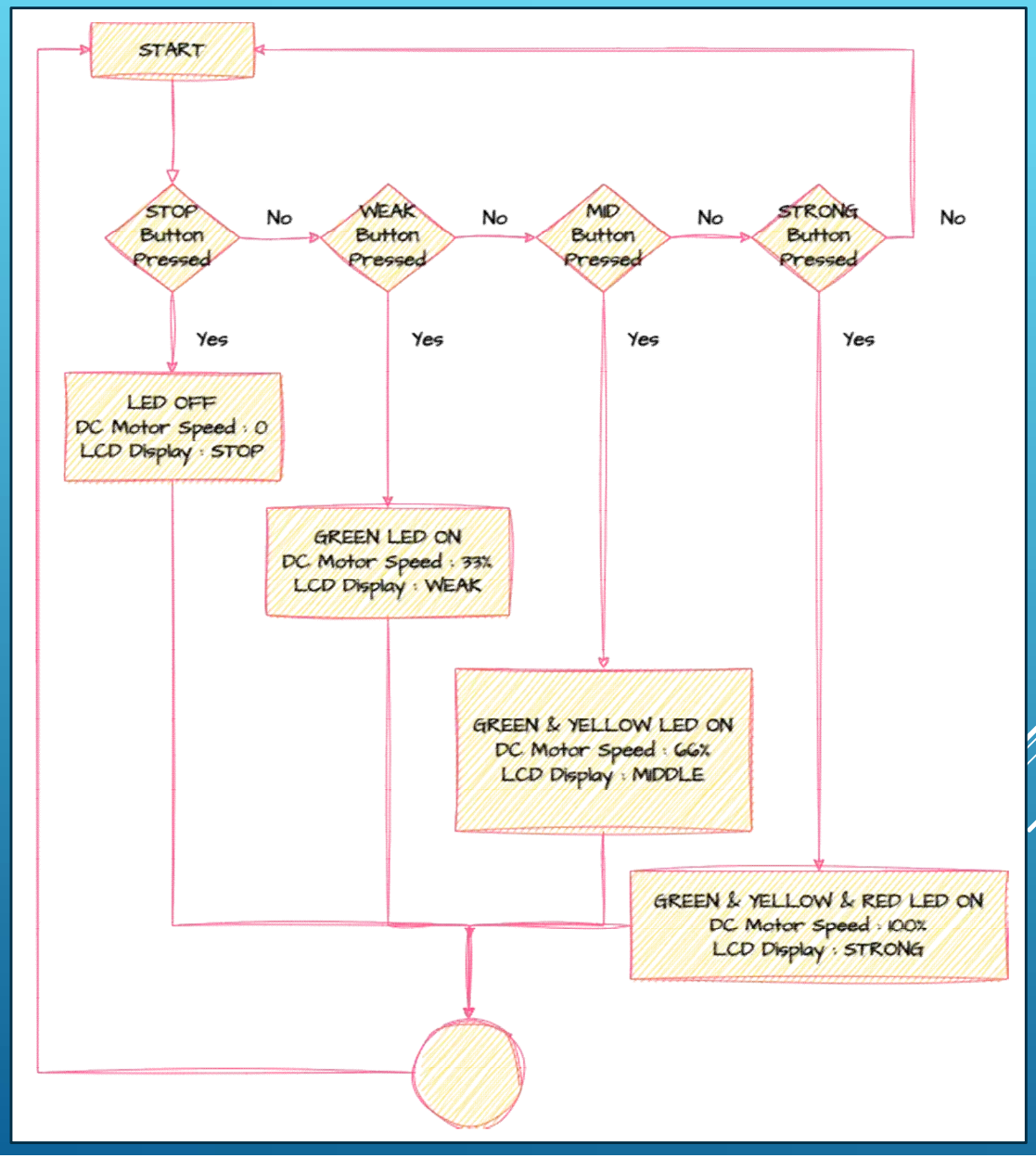
2023. 09. 27.

윤 희 승

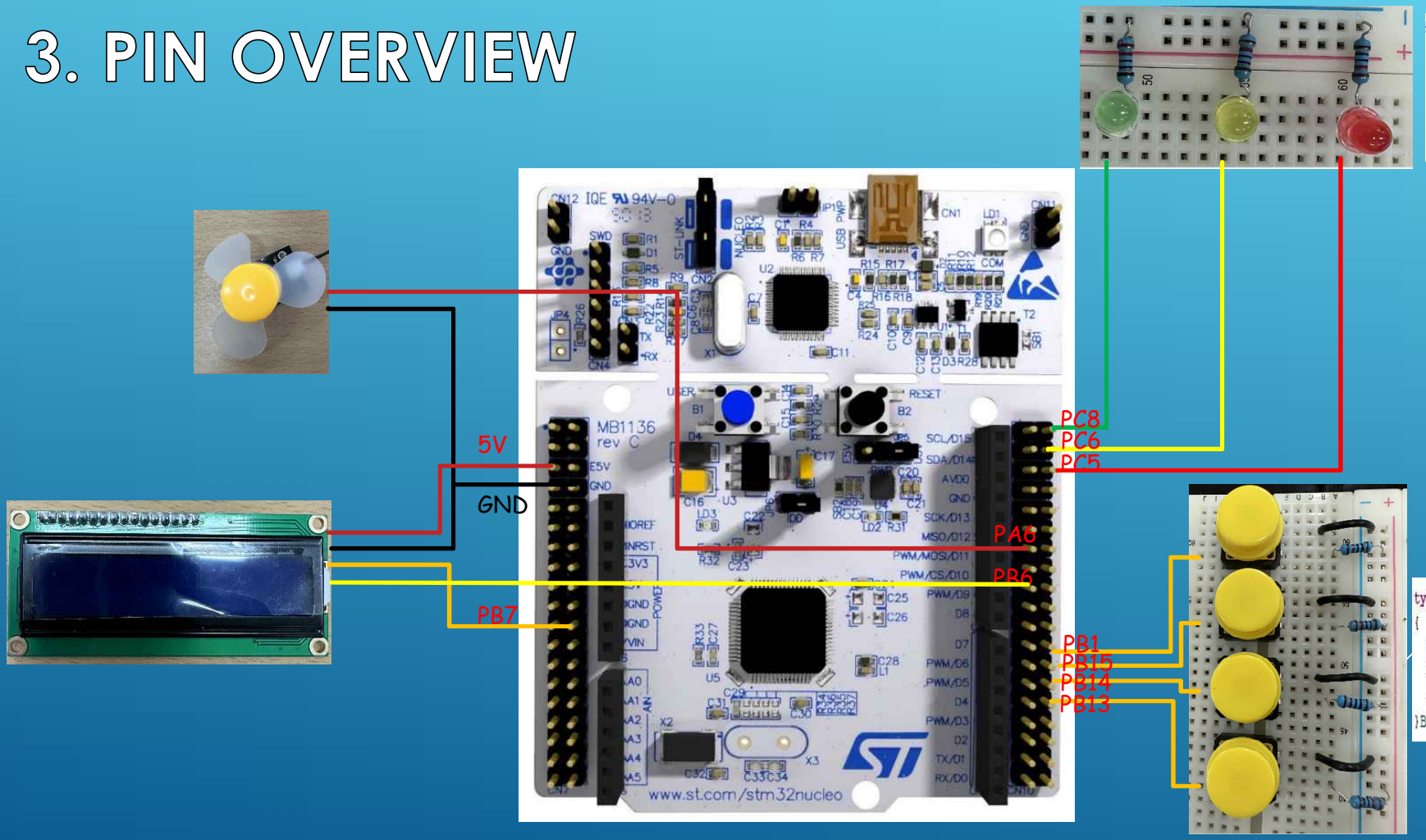
1. 목표

- 간단하게 동작하는 선풍기 설계 및 구현
  - 정지 버튼과 선풍기 회전 세기 조절 버튼(1단, 2단, 3단)
  - 회전 세기 별 LED 표시
  - LCD : 회전 세기 단계 표시

2. FLOW CHART


START
STOP Button Pressed
No
WEAK Button Pressed
No
MID Button Pressed
No
STRONG Button Pressed
No
Yes
Yes
Yes
Yes
LED OFF
DC Motor Speed : 0
LCD Display : STOP
GREEN LED ON
DC Motor Speed : 33%
LCD Display : WEAK
GREEN & YELLOW LED ON
DC Motor Speed : 66%
LCD Display : MIDDLE
GREEN & YELLOW & RED LED ON
DC Motor Speed : 100%
LCD Display : STRONG

3. PIN OVERVIEW




```
typedef struct
{
  GPIO_TypeDef *GPIOx;
  uint16_t GPIO_Pin;

}LED;
```

```
typedef struct _button
{
  GPIO_TypeDef *GPIOx; // Pin A, B, C ...
  uint16_t GPIO_Pin; // Pin number
  uint8_t preState; // Pin State
}Button;
```

4. 상세 내용

- 선풍기 회전 세기 조절
  - 크기가 4인 버튼 구조체 배열 fan_speed 생성 (정지, 1단, 2단, 3단)

```
enum {STOP, WEAK, MID, STRONG};
Button fan_speed[4];
```

  - 버튼 상태가 Falling Edge(ACT_PUSH)일 때 동작하도록 구성

```
uint8_t Button_getState(Button *btn)
{
  uint8_t curState = HAL_GPIO_ReadPin(btn->GPIOx, btn->GPIO_Pin);
  if(curState == PUSHED && btn->preState == RELEASED)
  {
    btn->preState = PUSHED;
    return ACT_PUSH;
  }
  else if((curState != PUSHED) && (btn->preState == PUSHED))
  {
    btn->preState = RELEASED;
    return ACT_RELEASE;
  }
  return NO_ACT;
}
```

4. 상세 내용

- BEFORE
  - 버튼을 외부 인터럽트로 설정하여 버튼이 눌러졌을 때, EXTI_Callback 함수를 호출
  - 버튼이 동작하지 않는 문제 발생
- AFTER
  - Polling 방식으로 변경하여 구현

BEFORE

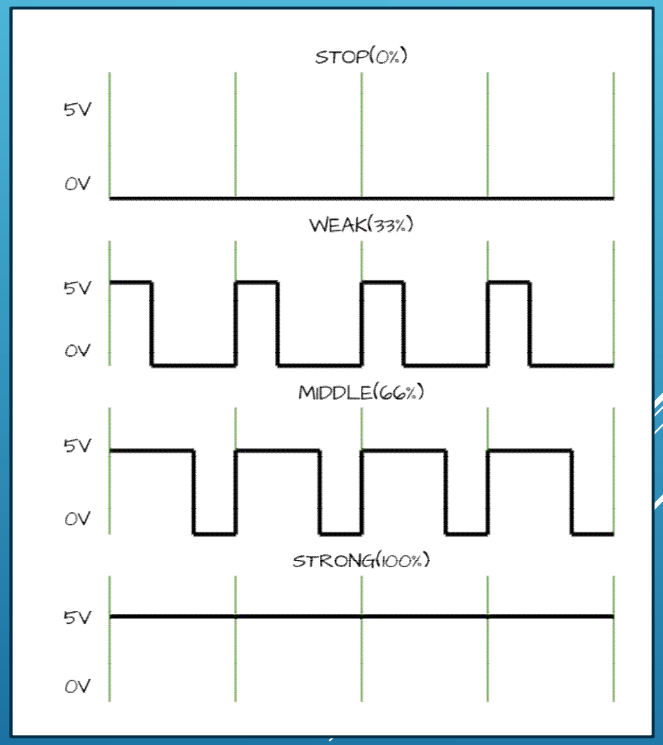
```
void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin)
{
  if(GPIO_Pin == GPIO_PIN_1)
  {
    // ...
  }
  else if(GPIO_Pin == GPIO_PIN_15)
  {
    // ...
  }
  else if(GPIO_Pin == GPIO_PIN_14)
  {
    // ...
  }
  else if(GPIO_Pin == GPIO_PIN_13)
  {
    // ...
  }
}
```

AFTER

```
while(1)
{
  if(Button_getState(&fan_speed[STOP]) == ACT_PUSH)
  {
    // ...
  }
  if(Button_getState(&fan_speed[WEAK]) == ACT_PUSH)
  {
    // ...
  }
  if(Button_getState(&fan_speed[MID]) == ACT_PUSH)
  {
    // ...
  }
  if(Button_getState(&fan_speed[STRONG]) == ACT_PUSH)
  {
    // ...
  }
}
```

4. 상세 내용

- 선풍기 날개(DC MOTOR)
  - PWM을 통해 DC MOTOR의 회전 세기를 조절
  - 회전 세기 별 CCR(PWM)은 ARR의 33%, 66%, 100%로 설정하여 주기마다 주어진 세기로 동작
  - TIM의 초기 ARR을 1,000으로 설정 – 카운터가 최대 1,000까지 증가
- 선풍기 회전 세기가 WEAK일 때, 전력이 너무 적어 회전을 잘 하지 못하는 문제 존재

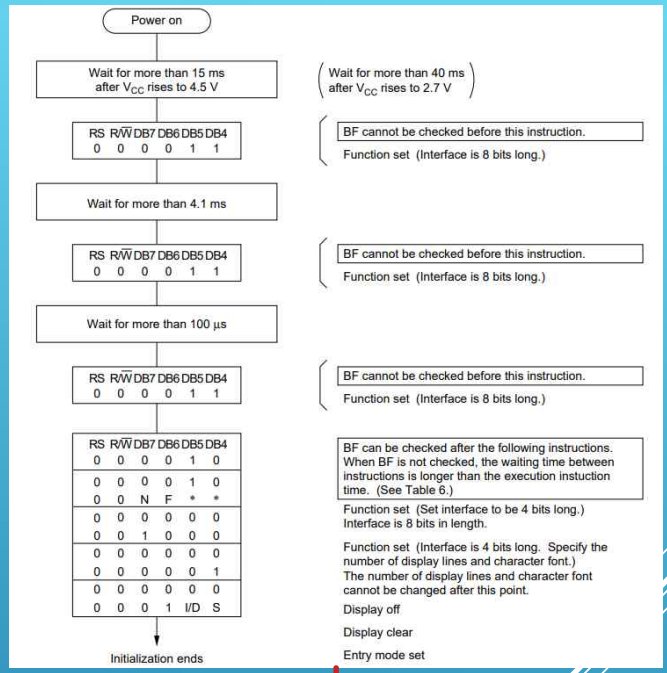


4. 상세 내용

- LED
  - 크기가 3인 LED 구조체 배열 생성 (1단, 2단, 3단)
  - 각 회전 세기 단계마다 동작하는 LED 개수 증가

- LCD
  - 현재 선풍기 회전 세기 상태(STOP, WEAK, MIDDLE, STRONG) DISPLAY

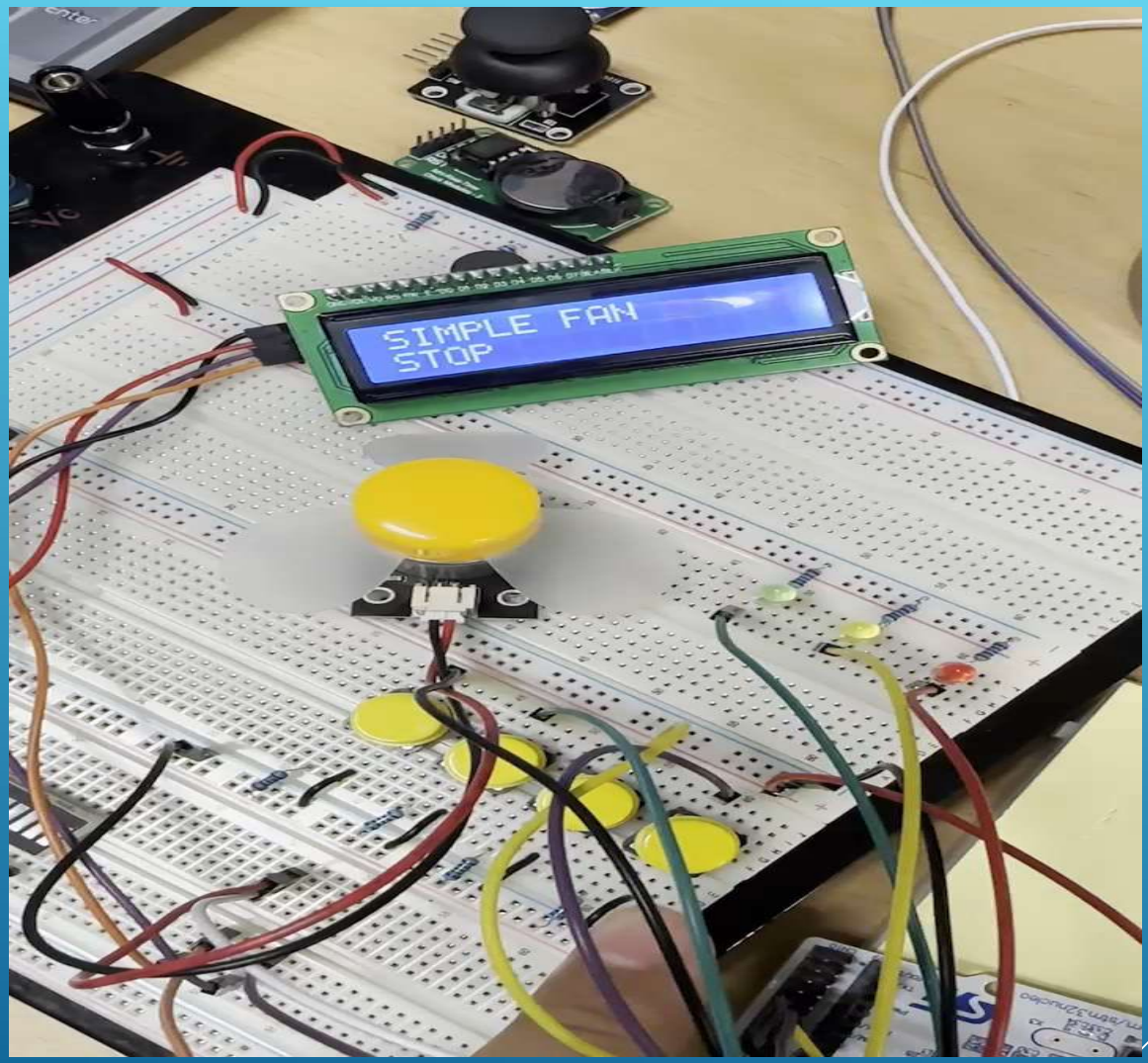
LCD 초기화 방법




```
void lcd_send_cmd(char cmd)
{
  char data_u, data_l;
  uint8_t data_t[4];
  data_u = (cmd & 0xf0);
  data_l = ((cmd<<4) & 0xf0);
  data_t[0] = data_u | 0x0C;    // en=1, reset=0,
  data_t[1] = data_u | 0x08;
  data_t[2] = data_l | 0x0C;
  data_t[3] = data_l | 0x08;
  HAL_I2C_Master_Transmit(&hi2c1, SLAVE_ADDRESS_LCD, data_t, sizeof(data_t), 100);

}
```

```
void lcd_init(void)
{
  HAL_Delay(40);
  lcd_send_cmd(0x30);
  HAL_Delay(5);
  lcd_send_cmd(0x30);
  HAL_Delay(1);
  lcd_send_cmd(0x30);

  HAL_Delay(10);
  lcd_send_cmd(0x20);  // set 4bits
  HAL_Delay(10);

  lcd_send_cmd(0x28);  // number of display lines and character font : 2row, 5*8 font
  HAL_Delay(1);
  lcd_send_cmd(0x08);  // display off
  HAL_Delay(1);
  lcd_send_cmd(0x01);  // display clear
  HAL_Delay(1);
  lcd_send_cmd(0x0C);  // display on
  HAL_Delay(1);
  lcd_send_cmd(0x06);  // entry mode set
}
```

5. 동작



THANK YOU!