```
#define Current_Pin       A0    // 전류측정 센서 핀
#define Acid_Pin          A1    // 연축전지 센서 핀
#define Scap_Pin          A2    // 슈퍼캐패시터 센서 핀
#define Process_Switch    6
#define Relay_Pin         7     // 슈퍼캐패시터 릴레이 제어 핀
#define Mosfet_Acid_Pin   8
#define Mosfet_Scap_Pin   9
int LedPin[]       = {2, 3, 4}; // Led 핀

#define V_Ref_Acid        12    // 연축전지 참조 전압
#define V_Ref_Scap        32    // 슈퍼캐패시터 참조 전압
#define Period            1000  // 시간 측정 주기
#define R1                31
#define R2                5
#define R_Load            10


float V_Soc;           // 상태 오류 보정
float V_Min;                          // 최소 전압
float V_Max;                          // 최대 전압
float Time_Current;                   // 현재 시간
int   Time_Flag;
float Time_Max       = 10000;         // 최대 시간
float Time_8Period   = 0;
float I_Switch     = 4.8;            // 스위치 전류
float I_Load;                        // 부하 전류

void setup() {
  Serial.begin(115200);              // 시리얼 통신 초기화
  pinMode(Relay_Pin, OUTPUT);        // 연축전지 릴레이 핀 설정
  pinMode(Mosfet_Acid_Pin, OUTPUT);     // 연축전지 릴레이 핀 설정
  pinMode(Mosfet_Scap_Pin, OUTPUT);     // 연축전지 릴레이 핀 설정
  pinMode(Process_Switch, INPUT);

  for(int i = 0; i < 3; i++)
    pinMode(LedPin[i], OUTPUT);      // LED 핀 설정

  mod_led(1);
  V_Min = V_Ref_Acid  - 1;           // V 최솟값 설정
  V_Max = V_Ref_Scap;                // V 최댓값 설정
}

float ma_filter(char Pin)
{
  float sum = 0;
  for(int i = 0; i < 100; i++)
  {
    sum += analogRead(Pin) * 5.0 / 1023.0;
    delay(10);
  }

  return sum / 100.0;
}

void mod_led(int flag)
{
  Serial.print("flag: ");
  Serial.println(flag);
```

pin 설정

setup 설정

process 1

이동평균필터

led 함수

```
  if(flag != 8)
  {
    for (int i = 0; i < 3; i++)
      digitalWrite(LedPin[i], (flag >> (2 - i)) & 1);
  }
  else
  {
    for (int i = 0; i < 3; i++)
      digitalWrite(LedPin[i], LOW);
  }

  delay(1000);
}

void time_measure()
{
  Time_Current = millis() / 1000;  // 현재 시간 측정
}

int soc_ocv(float Voltage)
{
  if (Voltage <= V_Min)
    return 0;  // 최소 전압 이하일 때 soc 0 반환
  else if (Voltage >= V_Max)
    return 100;  // 최대 전압 이상일 때 soc 100 반환
  else
    return (Voltage - V_Min) / (V_Max - V_Min) * 100;  // 최소 전압과 최대 전압 사이의 백분율로 soc 계산
}

void using_switch(int mode)
{
  switch(mode){
  case 0:
    digitalWrite(Mosfet_Acid_Pin, HIGH);
    digitalWrite(Mosfet_Scap_Pin, HIGH);
    time_measure();  // 시간 측정 함수 호출
    Time_Flag = 1;
    delay(1000);
    break;

  case 1:
    digitalWrite(Relay_Pin, HIGH);    // 연축전지 릴레이 활성화
    delay(1000);
    break;

  case 2:
    digitalWrite(Relay_Pin, LOW);    // 캐패시터 릴레이 활성화
    delay(1000);
    break;

  case 3:
    digitalWrite(Mosfet_Acid_Pin, LOW);
    digitalWrite(Mosfet_Scap_Pin, LOW);
    delay(100);
    break;
  }

}
```

시간측정함수

soc ocv 산출 함수

스위치 함수

nmos 사용 전력차단

process 4

```
void compare_value() // 동작 단계 4
{
  mod_led(4);
  if (I_Load < I_Switch) // (I_Load가 I_Switch보다 작을때)
    moving_by_acid(); // 동작 단계 5
  else // (if의 조건문이 아닐경우)
    moving_by_scap(); // 동작 단계 6
}
```

process 5

```
void moving_by_acid() // 동작 단계 5
{
  mod_led(5);
  using_switch(1);

  Serial.print(Time_Current);
  Serial.print(" ");
  Serial.println(Time_Max + Time_8Period);

  if (Time_Current >= Time_Max + Time_8Period) // 동작 단계 8
  {
    mod_led(8);
    using_switch(0);
  }
}
```

process 6

```
void moving_by_scap() // 동작 단계 6
{
  mod_led(6);
  if (V_Soc > V_Min) // 동작 단계 7(조건 1)
  {
    mod_led(7);
      using_switch(2);

    if (Time_Current >= Time_Max) // 동작 단계 8(조건 2)
    {
      mod_led(8);
      using_switch(0);
    }
    else // (조건 2가 아닐경우)
      compare_value();   // 동작 단계 4 호출(위의 void compare_value 함수로 이동)
  }

  else // V_Soc가 V_Min 보다 작거나 같을때
    moving_by_acid(); // 동작 단계 5
}
```

process 7

```
void loop() {

  if(digitalRead(Process_Switch) == HIGH)
  {
    using_switch(3);
    if(Time_Flag == 1)
    {
      Time_8Period += millis() / 1000 - Time_Current;
      Time_Flag = 0;
    }



    mod_led(2);
    float V_Acid = ma_filter(Acid_Pin) * (R1 + R2) / R2; // 동작 단계 2
    float V_Scap = ma_filter(Scap_Pin) * (R1 + R2) / R2; // 동작 단계 2
    float V_Load = ma_filter(Current_Pin)  * (R1 + R2) / R2;
    V_Soc = soc_ocv(V_Load) * V_Load / 100 ;
    I_Load = V_Load / R_Load;

    Serial.print(V_Acid);
    Serial.print(" ");
    Serial.print(V_Scap);
    Serial.print(" ");
    Serial.println(V_Min);

    mod_led(3);
    if (V_Acid >= V_Min && V_Scap >= V_Min); // 동작 단계 3(V_Acid 와 V_Scap이 둘 다 V_Min이상)
    {
      using_switch(1);
      compare_value();  // 동작 단계 4 호출(위의 void compare_value 함수로 이동)
    }

    if (V_Acid < V_Min && V_Scap < V_Min) // 동작 단계 8(V_Acid 와 V_Scap이 둘 다 V_Min보다 작음)
    {
      Serial.print(V_Acid);
      Serial.print(" ");
      Serial.print(V_Scap);
      Serial.print(" ");
      Serial.println(V_Min);
      mod_led(8);
      using_switch(0);
    }

  }

  else
  {
    using_switch(0);
    mod_led(8);
  }

}
```

슬라이드 스위치 on

정지 시간 측정

process 2

process 3

슬라이드 스위치 off

- 정지시 시간 계산
- 슬라이드 스위치 on
- 슬라이드 스위치 off
- 전압분배법 사용하여 전압 산출
- 사용 정지