블루투스로 선풍기와 LED 전등을 원격으로 조작하기

김택천 19010671

10cheon00@sju.ac.kr

1. 목표

오렌지보드에 내장된 블루투스를 이용하여 DC모터로 구현된 선풍기와 RGB LED를 사용한 전등을 원격으로 조작하는 회로를 구현하고, 블루투스 통신 시스템을 구축한다.

2. 구현 과정

1) DC모터 및 RGB LED 회로 구현

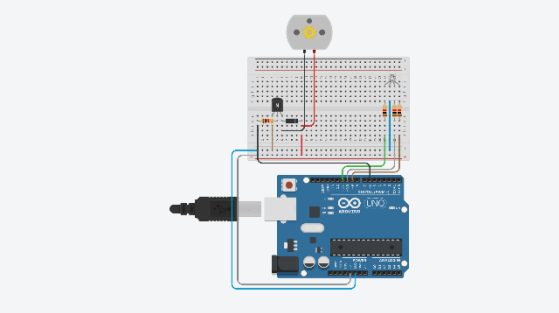

그림 1.A DC모터와 RGB LED 회로 구현

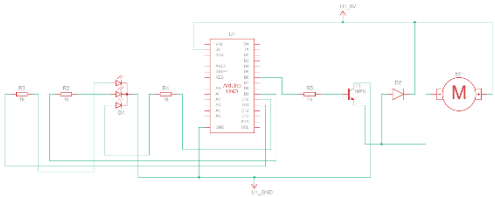

그림 2.B 그림 1.A 회로의 도식화

그림 1.A은 트랜지스터를 활용하여 베이스를 통해 선풍기의 세기를 조절할 수 있는 회로를 구현한 모습이다. 그림 1.B는 그림 1.A의 회로를 도식화한 것이다.

프로그래밍을 통해 6번 핀에 적절한 아날로그 신호를 출력할 때 DC모터가 회전하게 되고, 신호의 세기에 따라 모터의 회전 속력이 달라지게 된다. 이와 마찬가지로 11, 10 그리고 9번 핀에 적절한 아날로그 신호를 출력한다면 신호의 세기에 따라 RGB LED속에서 각 색상의 LED가 점등될 것이다. 만약 신호의 세기가 0이라면 LED가 점등되지 않으므로 불이 꺼진 것과 같게 된다.

2) JSON을 활용한 통신 시스템 프로그래밍

블루투스를 통해 원격 컨트롤러가 아두이노에게 명령을 내리게 되므로 적절한 명령을 내릴 수 있는 통신 시스템이 필요하다. 단순한 문자의 나열로 명령을 나타내기보다, 나름의 규칙을 갖고 대화할 수 있도록 JSON을 활용하기로 했다.

Key	Value
type	컨트롤러가 내리는 명령의 타입
value	컨트롤러가 내릴 명령의 구체적인 값

표 1. 통신 시스템 속 JSON 메시지의 구조

원격 컨트롤러는 표 1에 정의된 `type`과 `value`항목에 값을 담아 아두이노에게 전송하게 된다. 메시지가 전송된다면 매 루프마다 `BTSerial.available()`을 호출하여 받은 메시지가 있을 때에만 그 메시지를 해석하게 된다. 받은 메시지가 있다면, 첨부 #1에서 `readBTMessage()`함수와 `deserializeBTMessage()`함수를 호출하여 받은 메시지를 재조립한 후 해석한다.

예를 들어 컨트롤러가 LED의 밝기를 최대로 올리고 싶다면, `{“type”:2, “value”: 2}`의 텍스트를 아두이노에게 전송한다. 해석된 메시지는 `deserializeBTMessage()`함수 이후의 조건문에 의해서 메시지의 `type`값에 따라 각각의 명령을 수행한다. 명령이 수행된 후에는 `sendMessage()`함수를 통해 다시 컨트롤러에게 수행된 명령을 보고한다.

3) 원격 컨트롤러의 블루투스 어플리케이션 설정

원격 컨트롤러는 ‘2) JSON을 활용한 통신 시스템 프로그래밍’ 항목에서 정의한 JSON 메시지의 구조에 따라 메시지를 전송하여야 한다. 그러므로, 블루투스 통신 어플리케이션인 `nrf toolbox` 어플리케이션을 사용하여 통신하되, 통신 시스템 규칙을 따르는 JSON 메시지를 간편하게 전송할 수 있도록 내장된 매크로 기능을 이용한다.

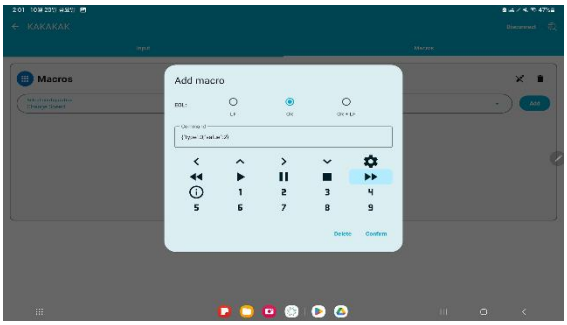

그림 2.A `nrf toolbox` 어플리케이션의 매크로 설정

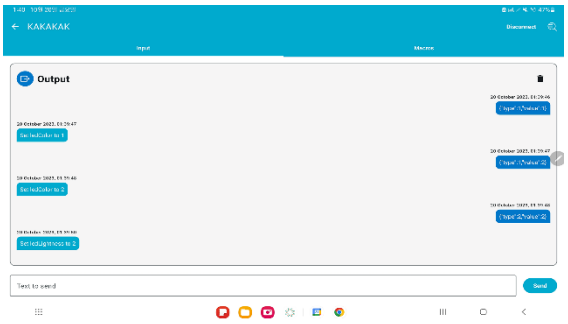

그림 2.B 통신 과정에서 발생한 메시지 로그

그림 2.A는 `nrf toolbox` 어플리케이션에서 매크로를 설정한 모습이다. 매크로가 지정된 버튼을 누르면 블루투스로 연결된 아두이노에게 지정된 명령 메시지가 전송된다. 명령이 적절히 수행되었다면 그림 2.B와 같이 수행된 명령이 보고된다. 아두이노가 컨트롤러에게 보고할 때에는 JSON을 이용하지 않았는데 JSON을 해석할 수 있는 컨트롤러를 만들기에는 새로 어플리케이션을 제작해야 하는 불편함 때문에 단순 문자열을 송신하는 방향으로 진행했다.

3. 결과

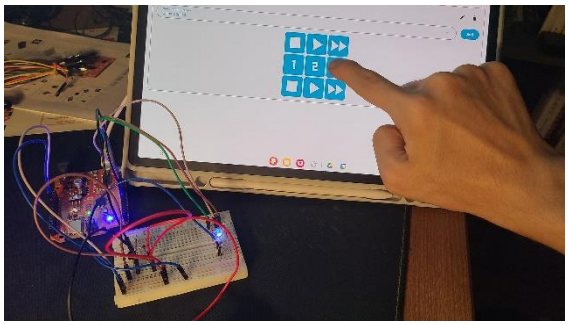

그림 3. 사용자가 컨트롤러를 통해 선풍기와 LED를 다루는 모습

그림 3과 같이 사용자는 아두이노의 회로에 연결된 버튼이나 다른 입력 장치를 필요로 하지 않고, 블루투스를 통하여 태블릿이나 스마트폰으로 아두이노에 연결된 선풍기와 LED를 컨트롤하게 된다.

4. 결론

RGB LED와 DC모터를 같이 활용하다가 트랜지스터 한 개를 망가뜨리게 되었다. 그 때문에 회로를 구현하는데 문제가 있었지만 여분의 트랜지스터로 해결했다.

통신 시스템을 구축하기 위해 JSON을 활용하였는데 처음 사용해보는 라이브러리여서 디버깅에 어려움이 있었다. 그렇지만 잘 정리된 API 문서 덕분에 문제를 해결할 수 있었다.

첨부 #1 : 실행 코드

```
#include <SoftwareSerial.h>
#include <ArduinoJson.h>

#define __DEBUG__
#define CAPACITY 256

// Bluetooth variables
SoftwareSerial BTSerial(4, 5);
String inputBuffer;
char messageBuffer[CAPACITY] = { 0 };
int btMessageType = 0;
int btMessageValue = 0;
enum BT_MESSAGE_TYPE {
   FAN_SPEED,
   LED_COLOR,
   LED_LIGHTNESS
};
const String BT_MESSAGE_KEY[3] = {
   "fanSpeed",
   "ledColor",
   "ledLightness"
};

// Arduino pin variables
const uint8_t PIN_FAN = 6;
const uint8_t PIN_LED_RED = 11;
const uint8_t PIN_LED_GREEN = 10;
const uint8_t PIN_LED_BLUE = 9;

// fan variables
```

```
const int FAN_SPEED_VALUE[3] = { 0, 127, 255 };
int fanSpeed = 0;

// led variables
uint8_t ledColor[3] = { 0 };
enum COLOR {
   RED,
   GREEN,
   PURPLE
};
const uint8_t LED_COLOR_VALUE[3][3] = {
   { 255, 0, 0 },
   { 0, 255, 0 },
   { 255, 0, 255 }
};
double lightness = 0.0;
const double LED_LIGHTNESS_VALUE[3] = {0.0, 0.5, 1.0};

void setup() {
#ifdef __DEBUG__
   Serial.begin(9600);
   while (!Serial)
      ;
#endif
   BTSerial.begin(9600);

   pinMode(PIN_FAN, OUTPUT);
   pinMode(PIN_LED_RED, OUTPUT);
   pinMode(PIN_LED_GREEN, OUTPUT);
   pinMode(PIN_LED_BLUE, OUTPUT);

   setLEDColor(COLOR::PURPLE);
```

```
    setLEDLightness(0);
    updateFanSpeed(0);
    updateLEDColorAndLightness();
}


void loop() {
    if (BTSerial.available()) {
        readBTMessage();
        deserializeBTMessage();
        if (btMessageType == BT_MESSAGE_TYPE::FAN_SPEED) {
            updateFanSpeed(btMessageValue);
        } else if (btMessageType == BT_MESSAGE_TYPE::LED_COLOR) {
            setLEDColor(btMessageValue);
            updateLEDColorAndLightness();
        } else if (btMessageType == BT_MESSAGE_TYPE::LED_LIGHTNESS) {
            setLEDLightness(btMessageValue);
            updateLEDColorAndLightness();
        }
        sendMessage();
    }
}

void readBTMessage() {
    inputBuffer = BTSerial.readString();
}

void deserializeBTMessage() {
    StaticJsonDocument<CAPACITY> doc;
    DeserializationError error = deserializeJson(doc, inputBuffer);

    if (error){
```

```
    Serial.println("Json parse error");
    return;
  }
  /*
   * Json Token Format
   * {
   *     "type": // this means BT_MESSAGE_TYPE
   *     "value": // this value will be applied differently by type
   * }
   */
  btMessageType = doc["type"];
  btMessageValue = doc["value"];

  sprintf(messageBuffer, "Set %s to %d", BT_MESSAGE_KEY[btMessageType].c_str(), btMessageValue);
}

void updateFanSpeed(int speedIndex) {
  speedIndex = clamp(speedIndex, 0, 2);

  fanSpeed = FAN_SPEED_VALUE[speedIndex];
  Serial.println(fanSpeed);
  analogWrite(PIN_FAN, fanSpeed);
}

void setLEDColor(int colorIndex) {
  colorIndex = clamp(colorIndex, 0, 2);

  ledColor[0] = LED_COLOR_VALUE[colorIndex][0];
  ledColor[1] = LED_COLOR_VALUE[colorIndex][1];
  ledColor[2] = LED_COLOR_VALUE[colorIndex][2];
}
```

```
void setLEDLightness(int lightnessIndex) {
    lightnessIndex = clamp(lightnessIndex, 0, 2);
    lightness = LED_LIGHTNESS_VALUE[lightnessIndex];
}

int clamp(int value, int low, int high) {
    return max(min(value, high), low);
}

void updateLEDColorAndLightness() {
    double enlightenRed = (double)ledColor[0] * lightness;
    double enlightenGreen = (double)ledColor[1] * lightness;
    double enlightenBlue = (double)ledColor[2] * lightness;
    analogWrite(PIN_LED_RED, enlightenRed);
    analogWrite(PIN_LED_GREEN, enlightenGreen);
    analogWrite(PIN_LED_BLUE, enlightenBlue);
}

void sendMessage() {
#ifdef __DEBUG__
    Serial.println(messageBuffer);
#endif
    BTSerial.write(messageBuffer);
}
```

참고 문헌

[1] [07-2 삼색(RGB) LED 사용하기], https://wikidocs.net/30791, 2019년 5월 4일 7:06 오후

[2] [오렌지보드 BLE와 채팅-안드로이드 채팅앱], https://kocoafab.cc/tutorial/view/527, 2015-10-08 14:17:08

[3] [Serial – Arduino Reference], https://www.arduino.cc/reference/en/language/functions/communication/serial/, 2023년 10월 19일

[4] [JsonParserExample.ino | ArduinoJson6], https://arduinojson.org/v6/example/parser/, 2023년 10월 19일