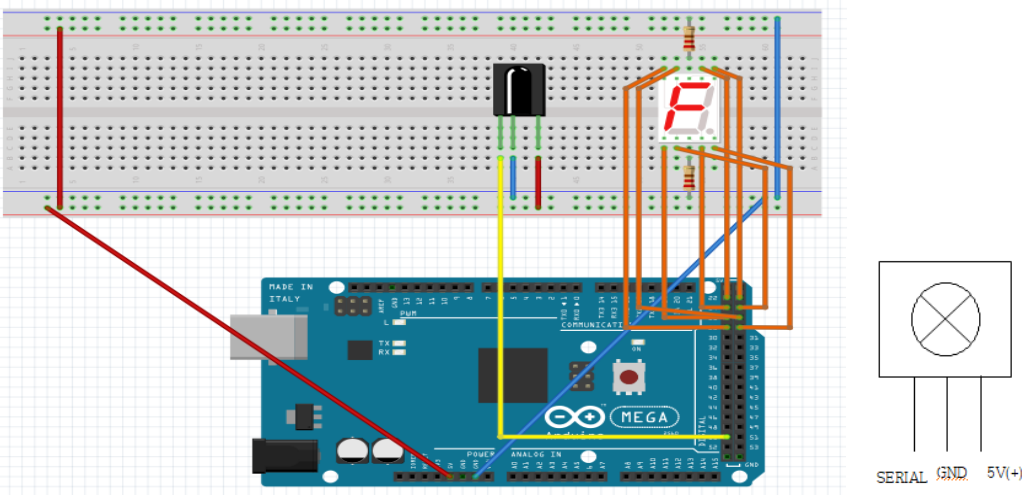


포트폴리오

프로젝트명	7 SEGMENT 와 IR 수신기/발신기를 이용한 숫자 카운트.
프로젝트 기간	2023.07.14~ 2023.07.14
상세내용	<p>1) 소 개: 7 SEGMENT 와 IR 수신기/발신기를 이용한 숫자 카운트.</p> <p>2) 주요 기능: 1. Arduino 의 Mega 2560 보드와 BreadBoard 를 연결. 2. 7 SEGMENT 와 IR 수신기를 BreadBoard 에 연결하여 회로를 구성. 3. Arduino IDE 를 이용하여 코드를 작성 후 업로드.</p> <p>3) 개발 환경 및 개발 언어: Arduino IDE, C 언어</p>
구성도	
상세 설명	<p>1. BreadBoard 에 7 SEGMENT 와 IR 수신기 연결.</p> <p>2. 7 SEGMENT 의 3 번과 8 번은 공통 GND 로 저항으로 GND 로 연결.</p> <p>3. Mega 2560 보드의 DIGITAL PIN 22 번~29 번 PIN 을 사용.</p> <p>4. A(7) -> 22PIN, B(6) -> 23 PIN, C(4) -> 24PIN, D(2) -> 25PIN, E(1) -> 26PIN, F(9) -> 27PIN G(10) -> 28PIN, DP(5) -> 29PIN 순으로 연결.</p> <p>5. IR 수신기의 '+'는 '+'에 연결, GND는 GND에 연결 후 SERIAL PIN은 DIGITAL PIN 50 번에 연결.</p> <p>6. 아두이노 IDE 에 IRremote 라이브러리를 다운 받은 후 헤더파일로 지정.</p> <p>7. 클래스를 만든 다음 생성자를 호출.</p> <p>8. 코드에서 비트 연산자를 이용 0~9 까지 (0b11111100, 0b01100000, 0b11011010, 0b11110010, 0b01100110, 0b10110110, 0b10111110, 0b11100000, 0b11111110, 0b11100110).</p>

사용 코드
(FOR)

```
#include <IRremote.hpp>
constexpr uint8_t IR_PIN{uint8_t (50U)};
class IRrecv irrecv {IRrecv(IR_PIN)}; //생성자 호출
class decode_results result; //default 생성자.
uint8_t FND[] {0b11111100, 0b01100000, 0b11011010,
               0b11110010, 0b01100110,
               0b10110110, 0b00111110, 0b11100100,
               0b11111110, 0b11100110};
uint8_t RC[] {22,12,24,94,8,28,90,66,82,74};
void setup() {
    // put your setup code here, to run once:
    Serial.begin(115200UL);
    pinMode(LED_BUILTIN, OUTPUT); //BUILTIN -> 13번 핀
    irrecv.begin(IR_PIN, LED_BUILTIN); //초기화 설정 -->

    Serial.begin(115200UL);
    for(int i=22; i<=29; ++i){
        pinMode(i, OUTPUT);
    }
}

void loop() {
    // put your main code here, to run repeatedly:
    if(irrecv.decode()){//decode == 해석하다.
        Serial.println(irrecv.decodedIRData.command, HEX); //HEX
        uint16_t remote_number = irrecv.decodedIRData.command;
        Serial.print(F("remote_number: "));
        Serial.println(remote_number);
        //irrecv.start(10);
        for(int j=0; j<=9; ++j){
            for(int i=2; i<=9; ++i){
                if(irrecv.decodedIRData.command==RC[j]){
                    if(bitRead(FND[j], 9 - i)){
                        digitalWrite(i + 20, HIGH);
                    }else{
                        digitalWrite(i + 20, LOW);
                    }
                }
            }
            delay(10UL);
        }
        irrecv.resume();
    }
}
```

사용 코드
(SWITCH)

```
#include <IRremote.hpp> //hpp -> C++에서 헤더파일. //# -> 매크로. 전처리.
constexpr uint8_t IR_PIN {50U};
const uint8_t FND[] = {0b11111100, 0b01100000, 0b11011010, 0b11110010, 0b01100110,
                      0b10110110, 0b01100110, 0b11100000, 0b11111110, 0b11100110};
class IRrecv irrecv {IRrecv(IR_PIN)}; //C++에서는 class 안써도 됨. 생성자 호출. irrecv
class decode_results result;

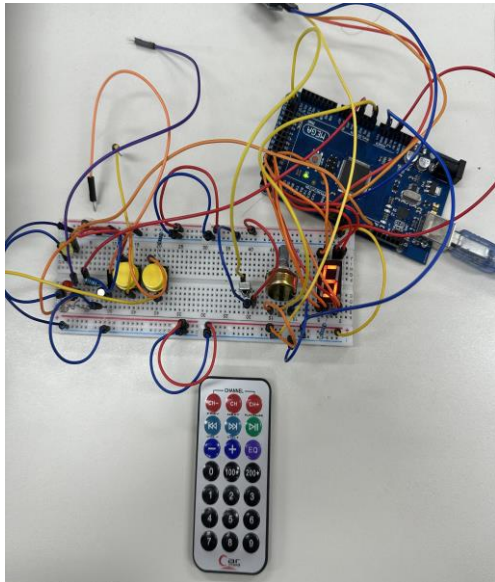
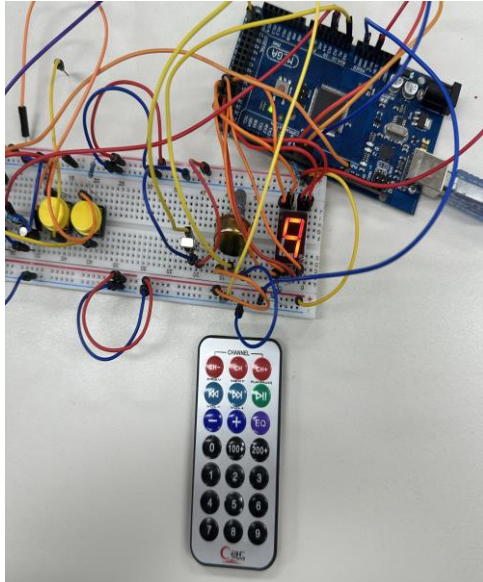
void setup() {
    // put your setup code here, to run once:
    Serial.begin(115200UL);
    pinMode(LED_BUILTIN, OUTPUT); //13pin 옆 LED
    irrecv.begin(IR_PIN, LED_BUILTIN); //하드웨어는 반드시 초기화 필수.
    for(int i = 22; i <= 29; ++i) {
        pinMode(i, OUTPUT);
    }
}

void loop() {
    if(irrecv.decode()) { //decode= 해석하다.
        uint16_t remote_number = irrecv.decodedIRData.command;
        Serial.print(F("Pressed number : "));
        Serial.println(remote_number);
        switch(remote_number)
        {
            case 22:
                Serial.println("0");
                for(int i = 2; i <= 9; ++i)
                {
                    if(bitRead(FND[0], 9 - i)) {
                        digitalWrite(i + 20, HIGH);
                    } else {
                        digitalWrite(i + 20, LOW);
                    }
                }
                break;

            case 12:
                Serial.println("1");
                for(int i = 2; i <= 9; ++i)
                {
                    if(bitRead(FND[1], 9 - i)) {
                        digitalWrite(i + 20, HIGH);
                    } else {
                        digitalWrite(i + 20, LOW);
                    }
                }
                break;

            case 24:
                Serial.println("2");
                for(int i = 2; i <= 9; ++i)
                {
                    if(bitRead(FND[2], 9 - i)) {
                        digitalWrite(i + 20, HIGH);
                    } else {
                        digitalWrite(i + 20, LOW);
                    }
                }
                break;

            case 94:
                Serial.println("3");
                for(int i = 2; i <= 9; ++i)
                {
                    if(bitRead(FND[3], 9 - i)) {
                        digitalWrite(i + 20, HIGH);
                    } else {
                        digitalWrite(i + 20, LOW);
                    }
                }
                break;
        }
    }
}
```

	<pre> case 8: Serial.println("4"); for(int i = 2; i <= 9; ++i) { if(bitRead(FND[4], 9 - i)) { digitalWrite(i + 20, HIGH); } else { digitalWrite(i + 20, LOW); } } break; case 28: Serial.println("5"); for(int i = 2; i <= 9; ++i) { if(bitRead(FND[5], 9 - i)) { digitalWrite(i + 20, HIGH); } else { digitalWrite(i + 20, LOW); } } break; case 90: Serial.println("6"); for(int i = 2; i <= 9; ++i) { if(bitRead(FND[6], 9 - i)) { digitalWrite(i + 20, HIGH); } else { digitalWrite(i + 20, LOW); } } break; </pre>	<pre> case 82: Serial.println("8"); for(int i = 2; i <= 9; ++i) { if(bitRead(FND[8], 9 - i)) { digitalWrite(i + 20, HIGH); } else { digitalWrite(i + 20, LOW); } } break; case 74: Serial.println("9"); for(int i = 2; i <= 9; ++i) { if(bitRead(FND[9], 9 - i)) { digitalWrite(i + 20, HIGH); } else { digitalWrite(i + 20, LOW); } } break; default: Serial.println(); } irrecv.resume(); delay(10UL); </pre>
	(2) IR 발신기에서 5를 누를 경우.	(2) IR 발신기에서 9를 누를 경우.
결과		
<p>처음에 FOR 문을 사용하여 코딩하였을 경우에는 IR 발신기에서 7 SEGMENT에 보고 싶은 숫자를 누를 경우 약간의 딜레이가 발생하여 동작하는 걸 발견하고 SWITCH 문으로 변경해본 결과 FOR 문에 비해서 딜레이가 없이 동작하는 것을 확인. IR 발신기에서 누른 숫자가 7 SEGMENT에 출력되는 것을 확인할 수 있습니다.</p>		