3비트 상향 카운터의 시각화 및 청각화

19010671 김택천

10cheon00@sju.ac.kr

1. 목표

아두이노를 이용하여 사용자가 버튼을 통해 조작 가능한 3비트 상향 카운터를 프로그래밍하고, 카운터의 값을 2진수로 나타내는 LED와 카운터의 값을 음계로 치환하여 소리를 내는 피에조 부저를 이용하여 시청각적으로 표현한다.

2. 구현 과정

2-1. 버튼 회로 구현 및 프로그래밍

사용자가 3비트 상향 카운터를 조작하기 위해서 아두이노와 버튼을 연결해야 한다.

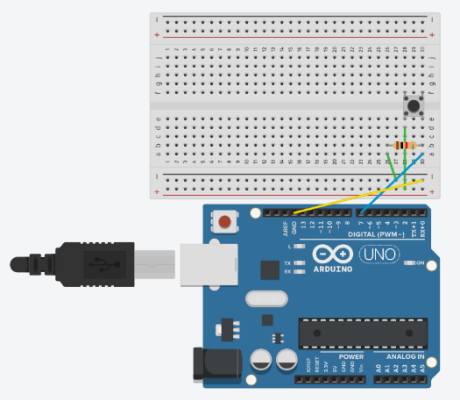

그림 1. 버튼 회로 구성

그림 1은 아두이노와 버튼을 연결하기 위해 회로를 구성한 모습이다. 버튼이 눌렸는가를 확인하기 위해 항상 전압이 가해져야 하므로 5V전원을 +버스에 연결하고, 전원이 연결된 +버스를 버튼이 설치된 IC영역과 연결한다.

만약 버튼을 누르게 된다면 IC영역 30번째 줄에 전원이 연결된다. 이 때 30번째 줄과 7번 핀이 연결되어 있으므로 7번 핀을 입력모드로 설정한다면 사용자가 버튼을 눌렀는지 알 수 있게 된다.

사용자가 버튼을 누를 때 카운터가 올라가도록 프로그래밍을 해야 한다. 첨부 #1 실행 코드에서 `loop()` 함수가 호출되면 `int readValue = digitalRead(PIN_BTN);`을 통해 버튼의 상태를 확인하고, 그 다음줄의 `int isButtonReleased = prevReadValue == HIGH && readValue == LOW;`을 통해 사용자가 버튼을 눌렀다가 뗐는지 확인한다. 사용자가 버튼을 눌렀다가 뗐을 때에만

`increaseCounter()` 함수를 호출하여 카운터를 증가시키고 있다. 만약 3비트 카운터의 최대값인 7에서 증가시키게 되는 경우에는 다시 0으로 초기화 시키도록 프로그래밍 되어 있다.

2-2. LED 회로 구현

카운터의 값을 시각화 하기 위해 카운터의 각 비트를 LED로 표현하여야 한다.

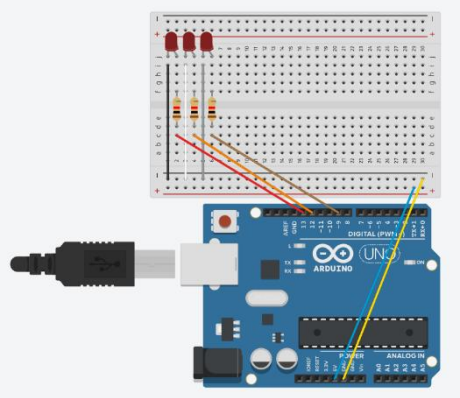

그림 2. LED 회로 구성

그림 2는 아두이노와 3개의 LED를 연결하기 위해 회로를 구성한 모습이다. 13, 12 그리고 9번 핀이 LED와 연결되어 있다. 각 핀을 입력모드로 설정한 후 출력을 내보낸다면 각 핀과 저항으로 연결되어 있는 LED가 점등될 것이다. 예를 들어 카운터의 값이 3이라면, 2진수로 나타내었을 때 $011_{(2)}$이므로 제일 왼쪽에 있는 LED부터 LOW, HIGH, HIGH 형태로 나타날 것이다.

첨부 #1 실행 코드의 `void updateLEDStateByCounter()` 함수 내부에서 `counter` 값의 각 비트 값에 따라 HIGH 또는 LOW 신호를 해당하는 핀으로 출력하고 있다.

2-3. 피에조 부저 회로 구현

카운터의 값을 청각화 하기 위해 카운터의 값을 '도레미파솔라시도'에 각각 대응시키고 그 음계를 피에조 부저로 표현하여야 한다.

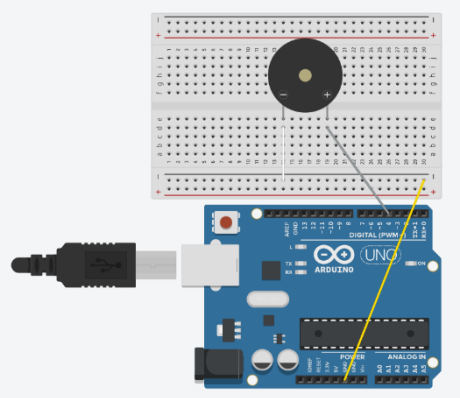

그림 3. 피에조 부저 회로 구성

그림 3은 아두이노와 피에조 부저를 연결하기 위해 회로를 구성한 모습이다.

피에조 부저의 -극은 -버스와, +극은 4번 핀과 연결하였다. 4번 핀에 출력 신호를 주게 된다면 피에조 부저에 전달된 주파수와 시간에 따라 소리가 출력된다.

첨부 #1의 실행 코드에서 카운터의 값을 '도레미파솔라시도'에 대응시키기 위해 `const int ARRAY_NOTE_PREQUENCY[8]`이라는 상수 값으로 미리 주파수 값을 설정한 다음, `void playSoundByCounter()` 함수 내부에서 해당하는 주파수 값을 출력시킨다. 매 루프마다 출력시키게 된다면 소리가 무한히 재생되므로 `void loop()` 함수 내부에서 버튼을 눌렀다가 뗐을 때에만 소리를 출력하도록 구현했다. 결과적으로 카운터의 값에 따라 '도레미파솔라시도'를 출력하도록 하여 청각화 시켰다.

3. 결과

그림 1, 그림 2, 그림 3을 모두 통합하여 회로도를 구현하면 아래의 그림 4처럼 완성된다.

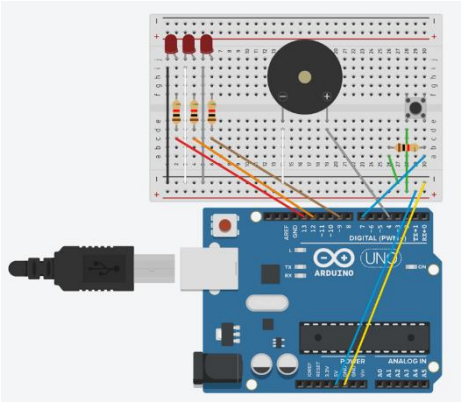

그림 4. 전체 회로 구성

사용자는 버튼을 눌러 카운터의 값을 증가시킬 수 있는데, 이를 LED를 통해 시각적으로 확인할 수 있다. 미리 정의된 주파수 값을 통해 카운터를 증가시킬 때마다 '도레미파솔라시도' 순으로 소리가 출력되어 카운터가 증가하고 있다는 것을 청각적으로 확인할 수 있다.

4. 결론

비록 논리회로를 통해 카운터를 구현한 것은 아니지만 카운터의 값을 2진수화하여 LED로 표현한 것과 카운터의 값에 따라 낮은 '도'부터 높은 '도'까지 올라갔다가 다시 낮은 '도'로 돌아오는 피에조 부저를 통해 시청각적으로 사용자가 카운터를 다루는 것처럼 느끼게 유도했다.

아두이노를 처음 다루어 보았기에 회로에 대한 이해가 부족했고 테스트를 하는 시간도 상당히 소요되었지만 인터넷에 공개된 자료가 방대하여 어렵지 않게 회로를 구현했다. 향후 소과제를 진행할 때에도 검색을 통해 쉽게 자료를 찾아 회로를 구현할 수 있다고 생각한다.

첨부 #1 : 실행 코드

```
const int COUNTER_BIT = 3;
const int COUNTER_MAX_VALUE = 1 << COUNTER_BIT;

const int PIN_BTN = 7;
const int PIN_LED_ARRAY[COUNTER_BIT] = { 13, 12, 8 };
const int PIN_SPEAKER = 4;

const int ARRAY_NOTE_FREQUENCY[COUNTER_MAX_VALUE] = { 262, 294, 330, 349, 392, 440, 493,
523 };

int counter = 0;
int prevReadValue = LOW;
bool isButtonReleased = false;

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

  for (int i = 0; i < COUNTER_BIT; i++) {
    pinMode(PIN_LED_ARRAY[i], OUTPUT);
  }
  pinMode(PIN_BTN, INPUT);
  pinMode(PIN_SPEAKER, OUTPUT);
}

void loop() {
  // update button state
  int readValue = digitalRead(PIN_BTN);
  int isButtonReleased = prevReadValue == HIGH && readValue == LOW;
```

```
    // increase counter when button released
    if (isButtonReleased) {
        increaseCounter();
    }
    // update LED state by counter
    updateLEDStateByCounter();
    // play sound only when button released
    // play sound with changed frequency by counter
    if (isButtonReleased) {
        playSoundByCounter();
    }
    // update button state
    // for trigger only released state.
    prevReadValue = readValue;

    // print only debug mode
#ifdef __DEBUG__
    Serial.println(counter);
#endif
}

void increaseCounter() {
    counter++;
    // if counter reached max value,
    // set counter to 0
    if (counter == COUNTER_MAX_VALUE) {
        counter = 0;
    }
}

void updateLEDStateByCounter() {
    int i = 0;
```

```
    while (i < COUNTER_BIT) {
        int status = (1 << i) & counter;
        digitalWrite(PIN_LED_ARRAY[i++], status ? HIGH : LOW);
    }
}

void playSoundByCounter() {
    tone(PIN_SPEAKER, ARRAY_NOTE_FREQUENCY[counter], 1000);
}
```

참고 문헌

[1] [아두이노기초-1 시작 전 필수로 알아야 할 사항], https://halfmoon.tistory.com/206, 2021년 12월 25일 23:17

[2] [05-1 푸시버튼으로 LED 켜고 끄기], https://wikidocs.net/30758, 2019년 5월 4일 17:02

[3] [13-1 피에조 스피커로 음계 표현하기], https://wikidocs.net/30897, 2019년 5월 5일 18:27