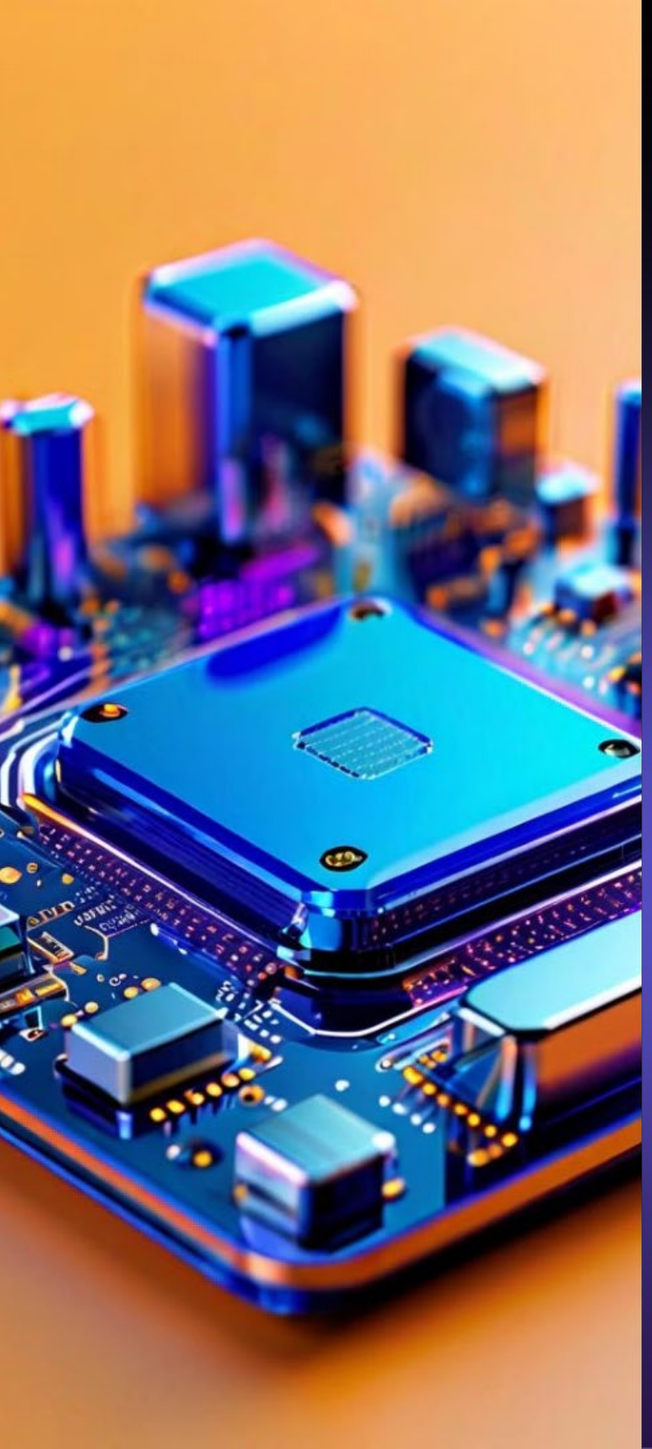
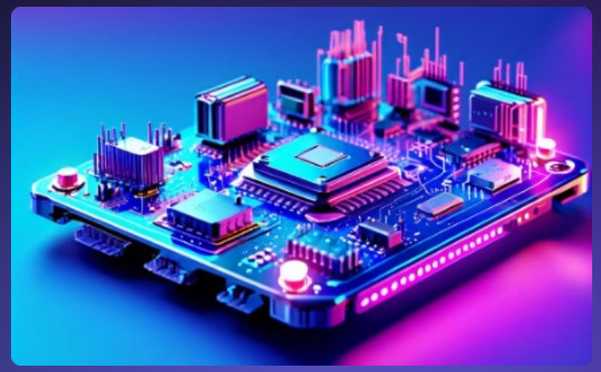
MICROPROCESSOR PROJECT
4조

학번	이름
2021245151	정용의
2021245153	유승민
2019253084	전유성



목차

프로젝트 개요 및 소개

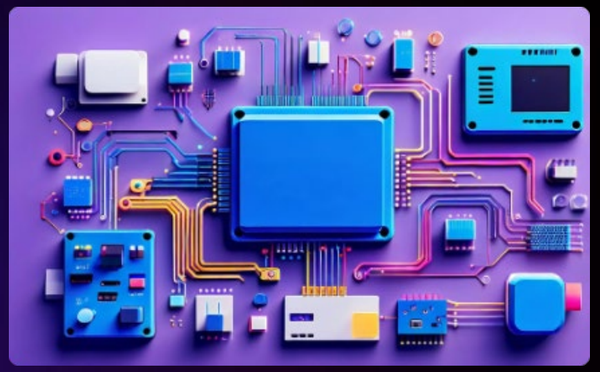


Arduino 기반 기억력 테스트 게임

Arduino mega를 사용하여 기억력 테스트 게임을 구현합니다.

사용자의 기억력 도전

사용자의 기억력을 도전하고 단계별로 난이도를 높이는 게임을 제공합니다.



다양한 입출력 장치 사용

LED, FND, Buzzer, Button, TEXT TEXT LCD 등 다양한 입출력 장치를 사용합니다.

게임 규칙

정해진 패턴 맞추기

이 게임은 정해진 패턴을 제한 시간 내에 맞추는 것이 목표입니다. 사용자는 LED, FND, Buzzer 등의 모듈을 통해 표시되는 패턴을 기억하고 정확하게 재현해야 합니다.

단계별 난이도 상승

게임은 1단계부터 15단계까지 난이도가 점점 높아집니다. 사용자가 각 단계를 성공적으로 클리어하면 다음 단계로 진행됩니다.

성공/실패 표시

사용자가 패턴을 정확히 맞추면 "GOOD" 표시와 함께 다음 단계로 진행됩니다. 실패하면 "FAIL" 표시가 나타납니다.

모듈 - LED

LED는 게임의 패턴을 표시하는 역할을 합니다. 각 단계에 맞는 패턴을 LED를 를 통해 점등하여 사용자가 기억할 수 있도록 합니다.

모듈 - LED 기능

getStageLedDuration

각 스테이지의 LED 시간 유지 기능입니다. 1~5단계계는 0.7초, 6~10단계는 0.5초, 11~15단계는 0.3초초 동안 LED를 유지합니다.

showPattern

LED 패턴을 표시하는 역할을 합니다. LED를 온오프프하여 사용자가 기억할 수 있는 패턴을 보여줍니다.

모듈 - FND

- 역할: FND(Seven Segment Display)는 각 단계의 성공 및 실패 여부를 부를 표시하는 역할을 합니다.
- 성공시 "GOOD" 표시: 사용자가 정해진 패턴을 정확히 맞추면 FND에 "GOOD"이 표시됩니다.
- 실패시 "FAIL" 표시: 사용자가 패턴을 틀리면 FND에 "FAIL"이 표시됩니다.
- 난이도별 대기 시간 출력: 각 난이도 단계마다 사용자가 패턴을 입력할 수 있는 대기 시간이 FND에 표시됩니다.

모듈 - FND 기능

1 **FndInit**

FND(Seven Segment Display)를 초기화하는 기능입능입니다. 게임 시작 시 FND를 초기화하여 준비 상태로 만듭니다.

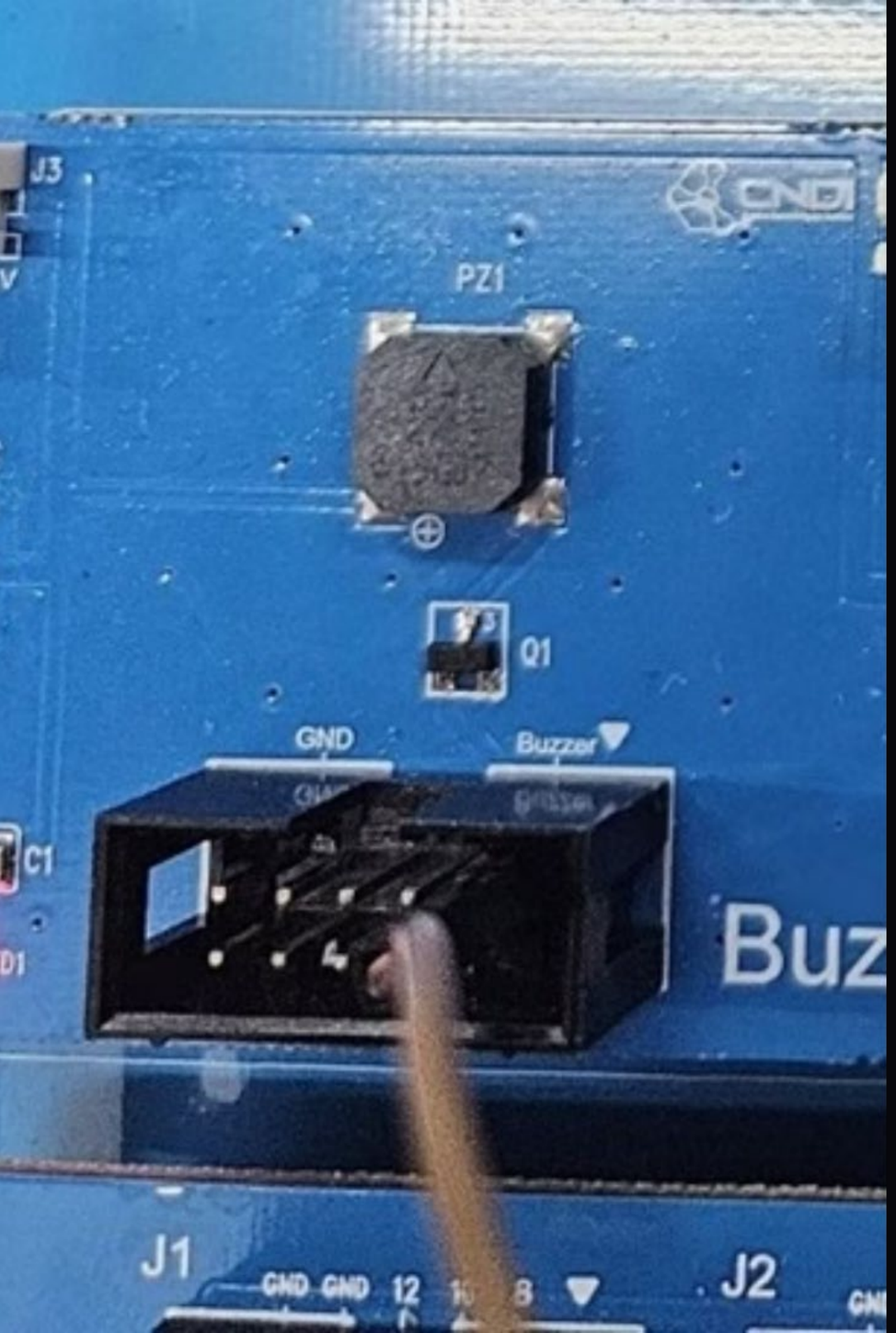
2 **FndSelect**

출력할 FND를 선택하는 기능입니다. 각 단계의 성공/실패 여부를 표시하기 위해 FND를 선택합니다.

3 **Fndoff**

출력 후 FND를 종료하는 기능입니다. 각 단계의 성공/실패 표시가 끝나면 FND를 끄고 다음 단계를 준비합니다.

```
94  // FND 초기화
95  void FndInit() {
96    int i;
97    for (i = 0; i < MAX_FND; i++) {
98      pinMode(FndSelectTable[i], OUTPUT);     // FND Sel Pin OUTPUT Set
99      pinMode(FndPinTable[i], OUTPUT);        // FND Data Pin OUTPUT Set
100   }
101 }
102
103 // FND 선택
104 void FndSelect(int Position) {
105   int i;
106   for (i = 0; i < MAX_FND_NUM; i++) {
107     if (Position & (1 << i)) {
108       digitalWrite(FndSelectTable[i], LOW);
109     } else {
110       digitalWrite(FndSelectTable[i], HIGH);
111     }
112   }
113 }
114
115 // FND 끄기
116 void FndOff() {
117   int i;
118   for (i = 0; i < MAX_FND; i++) {
119     digitalWrite(FndSelectTable[i], HIGH);
120   }
121   delay(50);
122 }
```



모듈 - Buzzer

역할

Buzzer는 게임의 성공 및 실패 상황을 알리는 음향 신호를 출력하는 역할을 합니다.

성공 시

사용자가 정해진 패턴을 정확히 맞추면 329Hz, 440Hz, 554Hz, 659Hz, 880Hz의 주파수로 음을 출력합니다.

실패 시

사용자가 패턴을 틀리면 880Hz의 주파수로 음을 출력합니다.

대기하는 5초 동안

사용자가 패턴을 입력할 수 있는 대기 시간 동안 700Hz의 주파수로 음을 출력합니다.

모듈 - Button

Button은 게임의 패턴을 입력하는 역할을 합니다. 사용자는 LED를 통해 표시되는 패턴을 보고 알맞게 Button을 입력해야 합니다.

모듈 - Button

getUserInputWithTimeout

제한된 시간 안에 키보드 입력을 받는 기능입니다. 사용자가 LED 패턴에 맞는 버튼을 입력할 수 있도록 합니다.

시간 표시

사용자가 버튼을 입력할 수 있는 남은 시간을 FND(Seven Segment Display)에 표시합니다. 이를 통해 사용자가 제한 시간 내에 패턴을 입력할 수 있도록 돕습니다.

패턴 입력

사용자가 LED 패턴에 맞는 버튼을 입력하도록 합니다. 사용자의 입력이 정확한지 확인하여 게임의 성공 여부를 판단합니다.

```
bool getUserInputWithTimeout(int* userInput, int length, int tim
  int count = 0;
  unsigned long startTime = millis();
  bool buzzerPlayed[3] = {false, false, false}; // 3, 2, 1 초에서

  while (count < length) {
    unsigned long elapsedTime = (millis() - startTime) / 1000;
    int remainingTime = timeout - elapsedTime;

    if (remainingTime >= 0) {
      DrawNumberFnd(remainingTime); // 남은 시간 FND에 표시
    }

    if (remainingTime == 3 && !buzzerPlayed[0]) {
      tone(buzzerPin, 700, 200); // 3초에 도달하면 Buzzer 소리
      buzzerPlayed[0] = true;
    }
    if (remainingTime == 2 && !buzzerPlayed[1]) {
      tone(buzzerPin, 700, 200); // 2초에 도달하면 Buzzer 소리
      buzzerPlayed[1] = true;
    }
    if (remainingTime == 1 && !buzzerPlayed[2]) {
      tone(buzzerPin, 700, 200); // 1초에 도달하면 Buzzer 소리
      buzzerPlayed[2] = true;
    }

    if (remainingTime <= 0) { // 제한 시간 초과 확인
      noTone(buzzerPin); // Buzzer 끄기
      return false;
    }

    for (int i = 0; i < 16; i++) {
      if (digitalRead(buttonPins[i]) == LOW) {
        userInput[count] = i;
        digitalWrite(LedPinTable[buttonToLedMap[i]], HIGH); // 켜
        ledStatus[buttonToLedMap[i]] = true;
        count++;
        delay(300); // 디바운싱을 위한 지연
        while (digitalRead(buttonPins[i]) == LOW); // 버튼이 떼어
```

모듈 – TEXT LCD

TEXT LCD는 각 단계를 사용자에게 알려주는 역할을 합니다. 마지막 단계까지 성공적으로 클리어하면 "You've "You've all cleared" 및 "Congratulations"라는 메시지를 출력합니다.

모듈 – TEXT LCD

1 **lcdPrint**

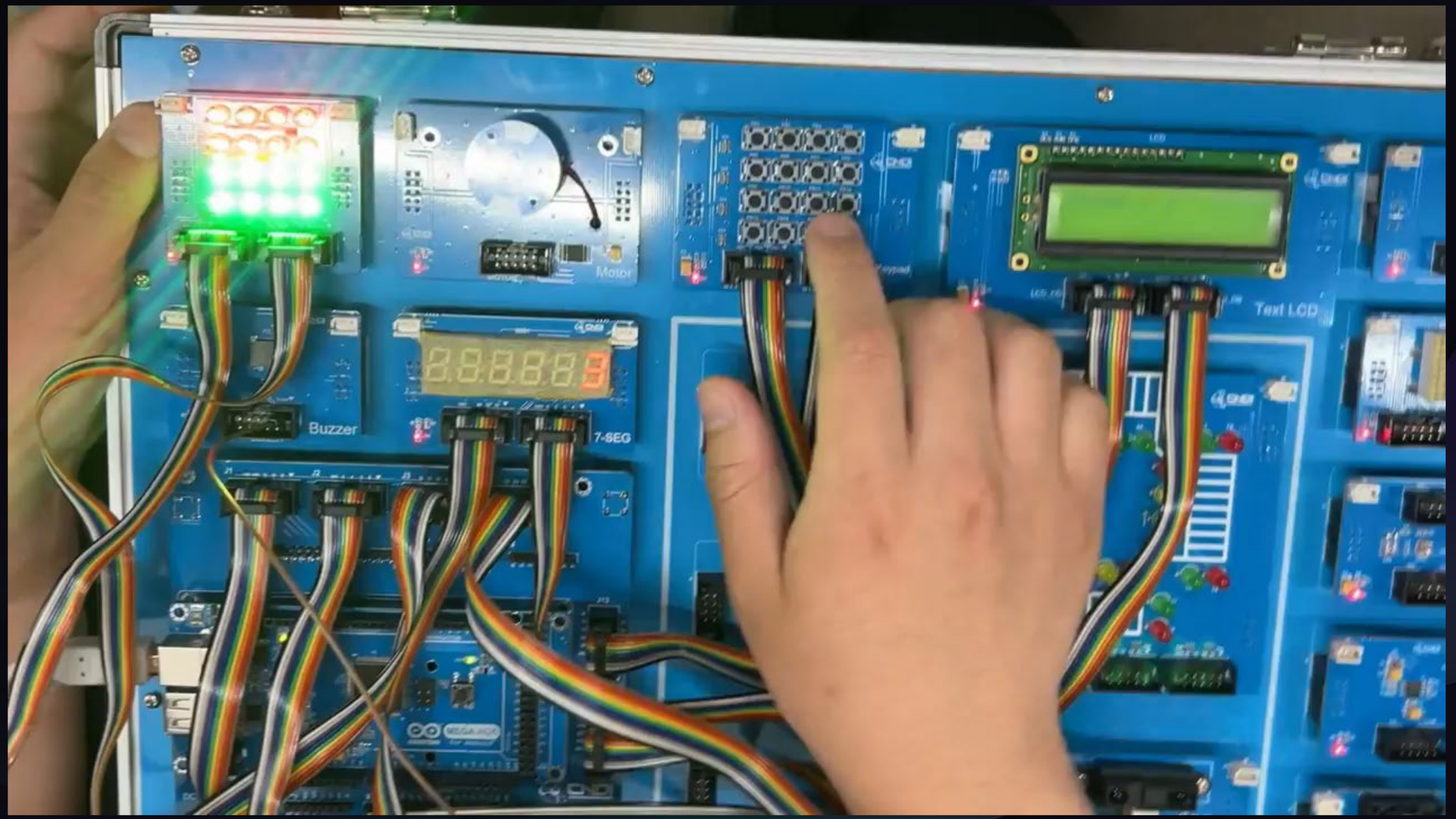
LCD에 문자열을 출력하는 기능입니다. 각 문자를 lcdData를 사용하여 전송합니다.

2 **displayStage**

현재 stage를 LCD에 출력하는 기능입니다. 조건문에 맞는 stage 번호 호를 출력합니다.

```
444  // LCD에 텍스트 출력
445  void lcdPrint(const char* str) {
446    while (*str) {
447      lcdData(*str++);
448    }
449  }
450
451  // LCD에 현재 스테이지를 출력하는 함수
452  void displayStage(int stage) {
453    lcdCommand(0x01); // Clear display
454    delay(2); // Wait for the clear command to execute
455
456    lcdPrint("Stage : ");
457    if (stage < 10) {
458      lcdData('0' + stage); // 한 자리 숫자
459    } else {
460      lcdData('0' + stage / 10); // 첫 번째 자리 숫자
461      lcdData('0' + stage % 10); // 두 번째 자리 숫자
462    }
463  }
```

시연 동영상



이상으로 4조 발표 마치도록 하겠습니다

감사합니다