به نام خدا

گزارش سوال سوم عملی درس سیستم های نهفته

پارسا آقاعلی ۴۰۰۵۲۱۰۷۲

مقدمه

هدف این تمرین استفاده از یک میکروکنترلر ESP32 برای شناسایی فشردن یک دکمه از طریق **وقفه خارجی (External Interrupt)**و کنترل یک LED است. این تمرین شامل پیاده‌سازی **دیبانس نرم‌افزاری (Software Debouncing)**می‌باشد تا از فشردن‌های ناخواسته جلوگیری شود. برای انجام این کار از شبیه‌ساز **Wokwi**به جای یک برد فیزیکی استفاده شده است.

مراحل انجام کار

1.راه‌اندازی شبیه‌سازWokwi

۱. وارد سایت Wokwiشدیم.

۲. یک تمرین جدید ESP32 ایجاد کردیم.

۳. قطعات مورد نیاز را به تمرین اضافه کردیم:

- **میکروکنترلرESP32**
- **دکمه فشاری(Push Button)**
- **LED**

۴. اتصالات زیر را در شبیه‌ساز انجام دادیم:

- **دکمه فشاری:**
  - یک پایه دکمه به **پایه GND** متصل شد.
  - پایه دیگر دکمه به **پایه GPIO4** (پین وقفه) وصل شد.

- **LED**:
  - پایه مثبت LED (آند) به **پایه GPIO2** وصل شد.
  - پایه منفی LED (کاتد) به GND متصل شد.

2.کدنویسی

کد تمرین با استفاده از زبان C در محیط **Arduino** نوشته شد. در این کد:

- از **تابع attachInterrupt()** برای تنظیم وقفه بر روی دکمه استفاده کردیم.
- از **میلی‌ثانیه‌ها (millis)** برای پیاده‌سازی دیبانس نرم‌افزاری استفاده شد.
- LEDبا هر بار فشردن دکمه روشن و خاموش می‌شود.

3.کد نوشته‌شده

کد کامل برنامه به شرح زیر است:

```
#include <Arduino.h>

#define BUTTON_PIN 4  // GPIO pin connected to the button
#define LED_PIN 2     // GPIO pin connected to the LED

volatile bool ledState = false;  // LED state (on/off)
volatile bool buttonPressed = false;  // Debounced button state

void IRAM_ATTR handleButtonInterrupt() {
  static unsigned long lastInterruptTime = 0;  // Timestamp of the last interrupt
  unsigned long currentTime = millis();

  // Debouncing: Only trigger if > 50ms since the last interrupt
  if (currentTime - lastInterruptTime > 50) {
    buttonPressed = true;  // Flag the button press
    lastInterruptTime = currentTime;
  }
}

void setup() {
```

```
  // Initialize the LED pin as an output
  pinMode(LED_PIN, OUTPUT);
  digitalWrite(LED_PIN, LOW);  // Turn LED off initially

  // Initialize the button pin as an input with an internal pull-up resistor
  pinMode(BUTTON_PIN, INPUT_PULLUP);

  // Attach interrupt to the button pin (falling edge triggers interrupt)
  attachInterrupt(digitalPinToInterrupt(BUTTON_PIN), handleButtonInterrupt,
FALLING);

  Serial.begin(115200);  // Initialize serial for debugging
}

void loop() {
  if (buttonPressed) {
    // Toggle LED state
    ledState = !ledState;
    digitalWrite(LED_PIN, ledState ? HIGH : LOW);

    Serial.println(ledState ? "LED ON" : "LED OFF");

    buttonPressed = false;  // Reset button press flag
  }

}
```

توضیحات کد

1.وقفه خارجی(External Interrupt)

- تابع attachInterrupt()برای اتصال وقفه به دکمه استفاده شده است. این تابع در حالت **FALLING**تنظیم شده است تا زمانی که دکمه به GND متصل شود (فشرده شود)، وقفه فعال شود.

2.دیبانس نرم‌افزاری

- از یک متغیر lastInterruptTimeبرای ثبت زمان آخرین وقفه استفاده کردیم.
- فاصله زمانی بین دو وقفه بررسی می‌شود تا از اجرای وقفه‌های ناخواسته (ناشی از نویز یا لرزش دکمه) جلوگیری شود.

3.تغییر وضعیتLED

- متغیر ledStateوضعیت روشن یا خاموش بودن LED را ذخیره می‌کند.
- در حلقه ()loop، با فشردن دکمه، این متغیر تغییر می‌کند و وضعیت جدید به LED اعمال می‌شود.

4.اجرای تمرین درWokwi

۱. قطعات را طبق اتصالات توضیح داده‌شده در محیط Wokwi شبیه‌سازی کردیم.

۲. کد بالا را در ویرایشگر کد Wokwi کپی کردیم.

۳. شبیه‌ساز را اجرا کردیم و صحت عملکرد بررسی شد:

- با فشردن دکمه، وضعیت LED تغییر می‌کند (روشن/خاموش).
- پیام‌های مربوط به وضعیت LED (ON/OFF) در مانیتور سریال نمایش داده می‌شوند.

نتیجه

- تمرین با موفقیت روی شبیه‌ساز Wokwi اجرا شد.
- با هر بار فشردن دکمه، وضعیت LED تغییر می‌کند (روشن/خاموش).
- دیبانس نرم‌افزاری به خوبی کار می‌کند و از نویز دکمه جلوگیری می‌کند.

منابع :

https://arduino.stackexchange.com/questions/66761/debouncing-a-button-with-interrupt

https://forum.arduino.cc/t/using-interrupt-to-detect-button-being-pressed/693556

https://community.st.com/t5/stm32-mcus-products/external-interrupt-and-button-debounce/td-p/722909