Dokumentacija - Upravljanje višestrukim prekidima na Arduino Mega

Projekt uključuje:

3 eksterna prekida (INT0, INT1, INT5) za tipkala

Timer prekid za periodično očitavanje potenciometra

Upravljanje LED diodom kao vizualna indikacija aktivnosti

Serijsku komunikaciju za ispis događaja

2. Hardverska konfiguracija

Komponenta Arduino Pin Opis

Tipkalo 1 2 (INT0) Najviši prioritet

Tipkalo 2 3 (INT1) Srednji prioritet

Tipkalo 3 18 (INT5) Najniži prioritet

LED dioda 13 Indikacija aktivnosti

Potenciometar A0 Simulacija analognog senzora

3. Softverska dokumentacija

3.1. Popis prekida i njihovi prioriteti

Prekid Pin Prioritet Opis

INT0 2 Najviši Reagira trenutno, može prekinuti druge

INT1 3 Srednji Može prekinuti samo INT5

INT5 18 Najniži Može biti prekinut od strane INT0/INT1

Timer1 - Srednji Očitava senzor svakih 500ms

3.2. Funkcije i njihova uloga

Funkcija Opis

ISR\_button0() Prekidna rutina za INT0 - pali LED i ispisuje poruku

ISR\_button1() Prekidna rutina za INT1 - pali LED

ISR\_button5() Prekidna rutina za INT5 - pali LED

ISR(TIMER1\_COMPA) Timer prekid - čita potenciometar svakih 500ms

handleButtonEvents() Glavna obrada tipkala (gasenje LED, Serial ispis)

handleSensorReading() Ispisuje vrijednost potenciometra u Serial Monitor

4. Testiranje sustava

4.1. Testni slučajevi

Test 1: Pojedinačni prekidi

Pritisak tipkala na INT0 LED upali, Serial ispis: "INT0 prekid"

Pritisak tipkala na INT1 LED upali, Serial ispis: "INT1 prekid"

Pritisak tipkala na INT5 LED upali i 3x zablista, Serial ispis: "INT5 prekid"

Test 2: Višestruki prekidi (testiranje prioriteta)

INT5 pritisnut, zatim INT0 INT0 prekine INT5, LED odmah reagira na INT0

INT5 pritisnut, zatim INT1 INT1 prekine INT5, LED reagira na INT1

INT1 pritisnut, zatim INT0 INT0 prekine INT1, LED odmah reagira na INT0

Test 3: Timer prekid (senzor)

Akcija Očekivani rezultat

Okretanje potenciometra Svakih 500ms Serial ispis: "Vrijednost senzora: X"

Test 4: Normalan rad sustava

Bez pritiska tipkala Svake 2 sekunde Serial ispis: "Normalan rad sustava..."

5. Rezultati testiranja

Test slučaj

Pojedinačni prekidi ✓ LED i Serial rade kako treba

Višestruki prekidi ✓ INT0 uvijek ima prednost

Timer prekid (senzor) ✓ Stabilno očitavanje

Normalan rad ✓ Nema blokada

6. Zaključak

Sustav ispravno upravlja višestrukim prekidima s različitim prioritetima:

INT0 uvijek dobiva najveći prioritet

Timer stabilno radi u pozadini

Nema sukoba pristupa resursima (Serial, LED)

Debounce mehanizam sprječava lažne aktivacije

Kod:  
#include <Arduino.h>

// Definiranje pinova

const int buttonPins[] = {2, 3, 18};  // INT0, INT1, INT5

const int ledPin = 13;

const int potPin = A0;

// Globalne varijable

volatile bool buttonPressed[] = {false, false, false};

volatile int sensorValue = 0;

volatile bool timerFlag = false;

volatile bool ledState = false;

// Semafori

volatile bool serialInUse = false;

volatile bool sensorInUse = false;

// Prototipovi funkcija

void handleButtonEvents();

void handleSensorReading();

void normalOperation();

void setup() {

  // Inicijalizacija serijske komunikacije

**Serial**.begin(9600);

  // Inicijalizacija LED

  pinMode(ledPin, OUTPUT);

  digitalWrite(ledPin, LOW);

  // Inicijalizacija tipkala s pullup otpornicima

  for (int i = 0; i < 3; i++) {

    pinMode(buttonPins[i], INPUT\_PULLUP);

  }

  // Konfiguracija eksternih prekida

  attachInterrupt(digitalPinToInterrupt(buttonPins[0]), ISR\_button0, FALLING);

  attachInterrupt(digitalPinToInterrupt(buttonPins[1]), ISR\_button1, FALLING);

  attachInterrupt(digitalPinToInterrupt(buttonPins[2]), ISR\_button5, FALLING);

  // Konfiguracija timer prekida (Timer1)

  noInterrupts();

  TCCR1A = 0;

  TCCR1B = 0;

  TCNT1 = 0;

  OCR1A = 15624; // 1Hz (16MHz/1024/1Hz - 1)

  TCCR1B |= (1 << WGM12); // CTC mod

  TCCR1B |= (1 << CS12) | (1 << CS10); // 1024 prescaler

  TIMSK1 |= (1 << OCIE1A); // Omogući timer compare interrupt

  interrupts();

**Serial**.println("Sustav spreman");

}

// INT0 prekidna rutina (najviši prioritet)

void ISR\_button0() {

  static unsigned long lastInterruptTime = 0;

  unsigned long interruptTime = millis();

  if (interruptTime - lastInterruptTime > 200) {

    buttonPressed[0] = true;

    ledState = true;

    digitalWrite(ledPin, HIGH);

    // Simulacija kratke obrade

    for (volatile int i = 0; i < 500; i++);

**Serial**.println("INT0 prekid - najviši prioritet");

  }

  lastInterruptTime = interruptTime;

}

// INT1 prekidna rutina (srednji prioritet)

void ISR\_button1() {

  static unsigned long lastInterruptTime = 0;

  unsigned long interruptTime = millis();

  if (interruptTime - lastInterruptTime > 200) {

    buttonPressed[1] = true;

    ledState = true;

    digitalWrite(ledPin, HIGH);

    // Kratka obrada

    for (volatile int i = 0; i < 300; i++);

  }

  lastInterruptTime = interruptTime;

}

// INT5 prekidna rutina (najniži prioritet)

void ISR\_button5() {

  static unsigned long lastInterruptTime = 0;

  unsigned long interruptTime = millis();

  if (interruptTime - lastInterruptTime > 200) {

    buttonPressed[2] = true;

    ledState = true;

    digitalWrite(ledPin, HIGH);

  }

  lastInterruptTime = interruptTime;

}

// Timer1 interrupt rutina

ISR(TIMER1\_COMPA\_vect) {

  static unsigned long lastSensorRead = 0;

  unsigned long currentMillis = millis();

  if (currentMillis - lastSensorRead >= 500) {

    lastSensorRead = currentMillis;

    if (!sensorInUse) {

      sensorInUse = true;

      sensorValue = analogRead(potPin);

      sensorInUse = false;

      timerFlag = true;

    }

  }

}

void loop() {

  handleButtonEvents();

  handleSensorReading();

  normalOperation();

}

void handleButtonEvents() {

  // Obrada INT0 događaja

  if (buttonPressed[0]) {

    buttonPressed[0] = false;

    while (serialInUse);

    serialInUse = true;

**Serial**.println("Obrada INT0 događaja");

    serialInUse = false;

  }

  // Obrada INT1 događaja

  if (buttonPressed[1]) {

    buttonPressed[1] = false;

    while (serialInUse);

    serialInUse = true;

**Serial**.println("Obrada INT1 događaja");

    serialInUse = false;

  }

  // Obrada INT5 događaja

  if (buttonPressed[2]) {

    buttonPressed[2] = false;

    while (serialInUse);

    serialInUse = true;

**Serial**.println("Obrada INT5 događaja");

    serialInUse = false;

    // Blinkanje LED nakon INT5

    for (int i = 0; i < 3; i++) {

      digitalWrite(ledPin, HIGH);

      delay(200);

      digitalWrite(ledPin, LOW);

      delay(200);

    }

    ledState = false;

  }

  // Gašenje LED ako nije aktivna obrada

  if (!buttonPressed[0] && !buttonPressed[1] && !buttonPressed[2] && ledState) {

    digitalWrite(ledPin, LOW);

    ledState = false;

  }

}

void handleSensorReading() {

  if (timerFlag) {

    timerFlag = false;

    while (serialInUse);

    serialInUse = true;

**Serial**.print("Vrijednost senzora: ");

**Serial**.println(sensorValue);

    serialInUse = false;

  }

}

void normalOperation() {

  static unsigned long lastPrint = 0;

  if (millis() - lastPrint > 2000) {

    lastPrint = millis();

    while (serialInUse);

    serialInUse = true;

**Serial**.println("Normalan rad sustava...");

    serialInUse = false;

  }

}