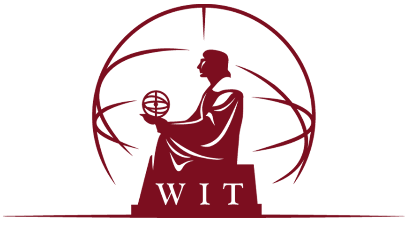
**Akademia WIT**

**pod auspicjami Polskiej Akademii Nauk**

****

**WYDZIAŁ INFORMATYCZNYCH**

**TECHNIK ZARZĄDZANIA**

STUDIA I STOPNIA (INŻYNIERSKIE)

**Laboratorium zastosowań elektroniki**

**Z-PEL-DB**

**Grupa TZS04IS**

**Laboratorium 10**

**Sprawozdanie**

**Opracował:**

*Ivan Ihnatsenkau*

*21595*

**WARSZAWA, 17.05.2024**

Spis treści

[1. Zadanie 1 – Czujnik temperatury LM35 3](#_Toc167369445)

[1.1. Wnioski 4](#_Toc167369446)

[2. Zadanie 2 – Czujnik pulsu – Laboratorium 5](#_Toc167369447)

[2.1. Wnioski 6](#_Toc167369448)

[3. Zadanie 3 – Pomiar pulsu – Laboratorium 7](#_Toc167369449)

[3.1. Wnioski 10](#_Toc167369450)

[4. Wnioski 11](#_Toc167369451)

# Zadanie 1 – Czujnik temperatury LM35

Opis doświadczenia

Obwód złożony z Przetwornik A/C Mikrokontrolera ATmega 32, czujnik temperatury LM35. Celem danego zadania jest zbudowaniu obwodu i napisanie kodu do mikrokontrolera. Ten kod odczytuje analogową wartość z czujnika temperatury LM35, przelicza ją na temperaturę w stopniach Celsjusza i wyświetla na monitorze szeregowym co 1 sekundę.

Изображение выглядит как электроника, Электронная техника, кабель, Компьютерный компонент

Автоматически созданное описание

Rysunek 1 Schemat do zadania 1

Wstawić program

#define LM35 A0

void setup(){

Serial.begin(9600);

}

void loop(){

int odczyt\_ADC = (analogRead(LM35));

Serial.print("Aktualna ADC = ");

Serial.println(odczyt\_ADC);

Serial.print("Wydrukowanie temperaturę");

float delta = 5.0/1024.0;

float Volt = float(odczyt\_ADC)\*delta;

float temp = Volt/0.01;

Serial.println(temp);

delay(1000);

}

## Wnioski

W trakcie lab 10 zbudowany obwód oraz napisany kod pozwalają na ciągły monitoring temperatury za pomocą czujnika LM35 i mikrokontrolera ATmega 32. Wyświetlanie wyników na monitorze szeregowym umożliwia łatwe śledzenie zmian temperatury w czasie rzeczywistym. Projekt ten ma wiele zastosowań praktycznych, zarówno w edukacji, jak i w przemyśle. Dzięki takiemu układowi można realizować projekty związane z kontrolą klimatu, np. automatyczne sterowanie wentylatorami, grzejnikami lub klimatyzatorami w zależności od aktualnej temperatury.

# Zadanie 2 – Czujnik pulsu – Laboratorium

Obwód złożony z przetwornika A/C mikrokontrolera ATmega 32 oraz sensora pulsu. Celem danego zadania jest zbudowanie obwodu i napisanie kodu do mikrokontrolera. Ten kod odczytuje analogową wartość z sensora pulsu, przelicza ją na odpowiednie wartości i wyświetla na monitorze szeregowym co 1 sekundę.

Изображение выглядит как электроника, Электронная техника, Компьютерное железо, Компьютерный компонент

Автоматически созданное описание

Rysunek 2. Schemat do zadania 2

Wstawić program

int SensorPin = 0;

int Pomiar;

void setup(){

Serial.begin(19200);

}

void loop(){

Pomiar = analogRead(SensorPin);

Serial.println(Pomiar);

delay(1000);

}

Изображение выглядит как снимок экрана

Автоматически созданное описание

Rysunek 3 Grafik pulsu z biegiem czasu.

Изображение выглядит как текст, рукописный текст, Шрифт, каллиграфия

Автоматически созданное описание

Rysunek 4 Realny grafik pulsu.

## Wnioski

Zbudowany obwód oraz napisany kod pozwalają na ciągły monitoring pulsu za pomocą sensora pulsu i mikrokontrolera ATmega 32. Wyświetlanie wyników na monitorze szeregowym umożliwia łatwe śledzenie zmian pulsu w czasie rzeczywistym. Sensor pulsu zapewnia dokładne i precyzyjne pomiary. Dzięki przetwornikowi A/C mikrokontrolera, uzyskujemy cyfrową reprezentację pulsu z wysoką rozdzielczością. Odczyty analogowe są bezpośrednio przetwarzane na wartości cyfrowe, co minimalizuje możliwość błędów obliczeniowych.

# Zadanie 3 – Pomiar pulsu – Laboratorium

Opis doświadczenia

Obwód złożony z mikrokontrolera Arduino oraz czujnika pulsu. Celem danego zadania jest zbudowanie obwodu i napisanie kodu do mikrokontrolera. Ten kod odczytuje wartość pulsu, przelicza ją na BPM (beats per minute) i wyświetla na monitorze szeregowym w czasie rzeczywistym. Dodatkowo dioda LED na Arduino miga w rytm bicia serca.

Изображение выглядит как электроника, Электронная техника, Компьютерный компонент, Компьютерное железо

Автоматически созданное описание

Rysunek 5. Shemat do zadania 3

Wstawić program

/\*  Getting\_BPM\_to\_Monitor prints the BPM to the Serial Monitor, using the least lines of code and PulseSensor Library.

 \*  Tutorial Webpage: https://pulsesensor.com/pages/getting-advanced

 \*

--------Use This Sketch To------------------------------------------

1) Displays user's live and changing BPM, Beats Per Minute, in Arduino's native Serial Monitor.

2) Print: "♥  A HeartBeat Happened !" when a beat is detected, live.

2) Learn about using a PulseSensor Library "Object".

4) Blinks the builtin LED with user's Heartbeat.

--------------------------------------------------------------------\*/

#define USE\_ARDUINO\_INTERRUPTS true    // Set-up low-level interrupts for most acurate BPM math.

#include <PulseSensorPlayground.h>     // Includes the PulseSensorPlayground Library.

//  Variables

const int PulseWire = 0;       // PulseSensor PURPLE WIRE connected to ANALOG PIN 0

const int LED = LED\_BUILTIN;          // The on-board Arduino LED, close to PIN 13.

int Threshold = 550;           // Determine which Signal to "count as a beat" and which to ignore.

                               // Use the "Gettting Started Project" to fine-tune Threshold Value beyond default setting.

                               // Otherwise leave the default "550" value.

PulseSensorPlayground pulseSensor;  // Creates an instance of the PulseSensorPlayground object called "pulseSensor"

void setup() {

  Serial.begin(9600);          // For Serial Monitor

  // Configure the PulseSensor object, by assigning our variables to it.

  pulseSensor.analogInput(PulseWire);

  pulseSensor.blinkOnPulse(LED);       //auto-magically blink Arduino's LED with heartbeat.

  pulseSensor.setThreshold(Threshold);

  // Double-check the "pulseSensor" object was created and "began" seeing a signal.

   if (pulseSensor.begin()) {

    Serial.println("We created a pulseSensor Object !");  //This prints one time at Arduino power-up,  or on Arduino reset.

  }

}

void loop() {

if (pulseSensor.sawStartOfBeat()) {            // Constantly test to see if "a beat happened".

int myBPM = pulseSensor.getBeatsPerMinute();  // Calls function on our pulseSensor object that returns BPM as an "int".

                                               // "myBPM" hold this BPM value now.

 Serial.println("♥  A HeartBeat Happened ! "); // If test is "true", print a message "a heartbeat happened".

 Serial.print("BPM: ");                        // Print phrase "BPM: "

 Serial.println(myBPM);                        // Print the value inside of myBPM.

}

  delay(20);                    // considered best practice in a simple sketch.

}

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

## Wnioski

Zbudowany obwód oraz napisany kod pozwalają na ciągły monitoring pulsu za pomocą sensora pulsu i mikrokontrolera Arduino. Wyświetlanie wyników na monitorze szeregowym oraz migająca dioda LED umożliwiają łatwe śledzenie zmian pulsu w czasie rzeczywistym. Projekt ten ma wiele zastosowań praktycznych, zarówno w edukacji, jak i w medycynie, fitnessie oraz sporcie. Sensor pulsu zapewnia dokładne i precyzyjne pomiary pulsu. Dzięki przetwornikowi A/C mikrokontrolera, uzyskujemy cyfrową reprezentację pulsu z wysoką rozdzielczości

# Wnioski

W ramach laboratorium przeprowadzono trzy zadania praktyczne, które pozwoliły na zdobycie wiedzy i umiejętności z zakresu pomiarów temperatury oraz monitorowania pulsu. Wszystkie przeprowadzone eksperymenty pokazały praktyczne zastosowania elektroniki i mikrokontrolerów w różnych dziedzinach życia. Określiłem zasad działania różnych typów sensorów oraz metod ich integracji z mikrokontrolerami.