**[Z6] Wybrana platforma – UART**

|  |  |  |
| --- | --- | --- |
| **Grupa laboratoryjna:** L11  **Podgrupa: 4** | Paweł Kolec (155 873) |  |
| Adam Nowacki (155 838) |
| **Prowadzący zajęcia:** | **dr inż. Ariel Antonowicz** |

**ZDJĘCIA ZREALIZOWANEGO UKŁADU**

|  |
| --- |
|  |
|  |

**KOD ZREALIZOWANEGO ALGORYTMU - Arduino**

|  |
| --- |
| #include <OneWire.h>  #include <DS18B20.h>  #define ONEWIRE\_PIN 19  float mala[18] = {0.0};  short int numer = 0;  byte address[8] = {0x28, 0x8, 0xC, 0x79, 0x97, 0x2, 0x3, 0x84};  OneWire onewire(ONEWIRE\_PIN);  DS18B20 sensors(&onewire);  float mini(float mala[18]) {  float a = 1000000.0;  for (int i = 0; i < 18; i++) {  if (mala[i] < a) a = mala[i];  }  return a;  }  float maxi(float mala[18]) {  float a = -273.15;  for (int i = 0; i < 18; i++) {  if (mala[i] > a) a = mala[i];  }  return a;  }  float srednia(float mala[18]) {  float suma = 0;  for (int i = 0; i < 18; i++) {  suma += mala[i];  }  suma = (suma - mini(mala) - maxi(mala)) / 16;  return suma;  }  void setup() {  Serial.begin(9600);  sensors.begin();  sensors.request(address);  }  void loop() {  if (sensors.available()) {  float temperature = sensors.readTemperature(address);  mala[numer] = temperature;  numer++;  if (numer == 18) {  float avg\_temperature = srednia(mala);  Serial.println(avg\_temperature);  numer = 0;  }  sensors.request(address);  delay(1);  }  } |

**KOD ZREALIZOWANEGO ALGORYTMU – Wybrana platforma**

|  |
| --- |
| import Adafruit\_BBIO.UART as UART  import serial  import sqlite3  import time  # Ustalenie połączenia z UART  def setup\_uart():  UART.setup("UART1")  ser = serial.Serial(port="/dev/ttyS1", baudrate=9600, timeout=1)  return ser  # Tworzenie bazy danych (jeśli nie istnieje)  def create\_db():  conn = sqlite3.connect('temperatura.db')  c = conn.cursor()  c.execute('''CREATE TABLE IF NOT EXISTS temperatura  (id INTEGER PRIMARY KEY AUTOINCREMENT,  MEASURED\_AT DATETIME DEFAULT CURRENT\_TIMESTAMP,  temperatura REAL)''')  conn.commit()  conn.close()  # Wstawianie średniej temperatury do bazy danych  def insert\_db(avg\_temp):  conn = sqlite3.connect('temperatura.db')  c = conn.cursor()  c.execute("INSERT INTO temperatura (temperatura) VALUES (?)", (avg\_temp,))  conn.commit()  conn.close()  # Wyświetlanie danych z bazy  def show\_db():  conn = sqlite3.connect('temperatura.db')  c = conn.cursor()  c.execute("SELECT \* FROM temperatura")  measurements = c.fetchall()  for measurement in measurements:  print(f"ID: {measurement[0]} Znacznik czasowy: {measurement[1]} Temperatura: {measurement[2]} C")  conn.close()  # Funkcja do odczytu danych z UART i przetwarzania  def read\_uart\_data(ser):  if ser.isOpen():  message = ser.readline().decode("utf-8").strip() # Odczyt i oczyszczenie danych  return message  return None  # Główna funkcja programu  def main():  ser = setup\_uart()  create\_db()  while True:  message = read\_uart\_data(ser)  if message:  try:  # Załóżmy, że odczytana wartość to liczba zmiennoprzecinkowa  avg\_temp = float(message)  print(f"Odczytana temperatura: {avg\_temp} C")  # Zapis do bazy danych  insert\_db(avg\_temp)  # Wyświetlenie zawartości bazy  show\_db()  except ValueError:  print("Nieprawidłowy format danych temperatury")  time.sleep(1) # Przerwa między odczytami  if \_\_name\_\_ == "\_\_main\_\_":  main() |