SPRAWOZDANIE Z PROJEKTU		Rok akademicki 2023/24
Przedmiot: **SYSTEMY MIKROPROCESOROWE**		
Temat ćwiczenia: **Projekt: Regulacja temperatury, regulator PID**		Termin zajęć: **czwartek 18:30 – 20:00**
Wydział, kierunek, semestr, grupa: **WARiE, AiR, 5, A1-L2**	Imię i Nazwisko: 1. Ignacy Chrobak 2. Jakub Czerwiński 3. Wojciech Czarnowski	Punkty:

1 Wstęp

Regulatory PID (Proporcjonalno-Całkująco-Różniczkujące) są jednymi z najczęściej stosowanych narzędzi w inżynierii sterowania.

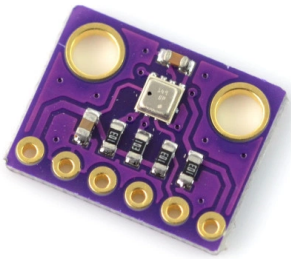
Dokumentacja przedstawia proces projektowania, analizy oraz implementacji regulatora PID dla obiektu fizycznego aproksymowanego jako układ inercyjny pierwszego rzędu z opóźnieniem transportowym. Głównym celem projektu jest stworzenie wydajnego systemu regulacji temperatury, opierającego się na czujniku BME280 i sygnale PWM

2 Opis obiektu

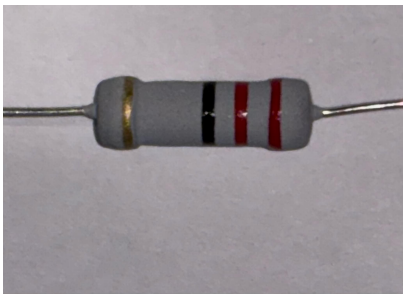
Obiekt, którego charakterystyka została zbadana, to prosty układ składający się z rezystora o wartości $22\,\Omega$. Rezystor ten pełni funkcję elementu grzewczego, który nagrzewa się w odpowiedzi na napięcie podawane na jego końcówki. Napięcie sterujące mieści się w zakresie $0 - 3.3\,\mathrm{V}$ i jest generowane przez mikrokontroler przy użyciu modulacji szerokości impulsów (PWM). Układ sterujący oparty jest na tranzystorze PN2222A, który umożliwia precyzyjne sterowanie przepływem prądu przez rezystor.

Zasilanie oraz implementacja całego systemu została zrealizowana na płytce rozwojowej Nucleo-F767ZI, która zapewniała niezbędną moc obliczeniową, interfejsy komunikacyjne oraz możliwości programowania.

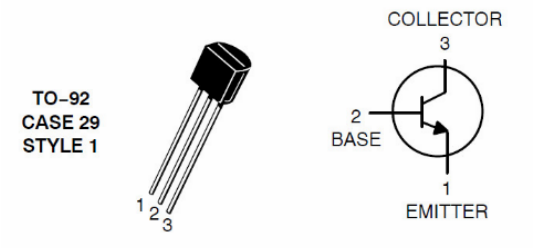
Temperatura rezystora, będąca wielkością wyjściową układu, jest mierzona za pomocą precyzyjnego czujnika BME280. Czujnik ten pozwala na odczyt temperatury w sposób szybki i dokładny, co jest kluczowe dla efektywnej implementacji regulatora PID.



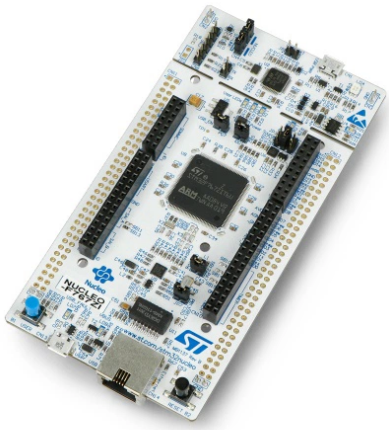
Rysunek 1: Czujnik temperatury i ciśnienia BME-280



Rysunek 2: Rezystor 22Ω



Rysunek 3: Tranzystor NPN PN2222a



Rysunek 4: Płytka Nucleo-F76ZI

3 Zbieranie danych do analizy modelu obiektu

Aby określić charakterystykę obiektu, na początku wymuszono na układzie skok jednostkowy napięcia, podając na wejście maksymalne napięcie 3.3 V. Następnie za pomocą czujnika BME280 rejestrowano zmiany temperatury w czasie.

Do zapisu danych wykorzystano skrypt napisany w języku Python. Skrypt ten automatycznie zbierał dane z czujnika i zapisywał je w formacie CSV, co umożliwiło ich późniejszą analizę. Proces ten zapewnił dokładne i powtarzalne wyniki, które były niezbędne do modelowania matematycznego obiektu.

```python
import serial
import csv
import time

ser = serial.Serial()
ser.baudrate = 115200
ser.port = "COM3"
ser.timeout = 1
try:
    ser.open()
    print("Serial port opened successfully.")
except Exception as e:
    print(f"Failed to open serial port: {e}")
    exit()
csv_file = "data.csv"

with open(csv_file, mode="w", newline="") as file:
    writer = csv.writer(file)
    writer.writerow(["Temperature (C)"])
    print("Start reading data...")
    try:
        while True:
            line = ser.readline().decode("utf-8").strip()
            if line:
                try:
                    temperature = float(line)
                    print(f"Temperature: {temperature}")
                    writer.writerow([temperature])
                except ValueError:
                    print(f"Malformed data: {line}")

    except KeyboardInterrupt:
        print("Data logging stopped.")
    except Exception as e:
        print(f"An error occurred: {e}")

ser.close()
print("Serial port closed.")
```

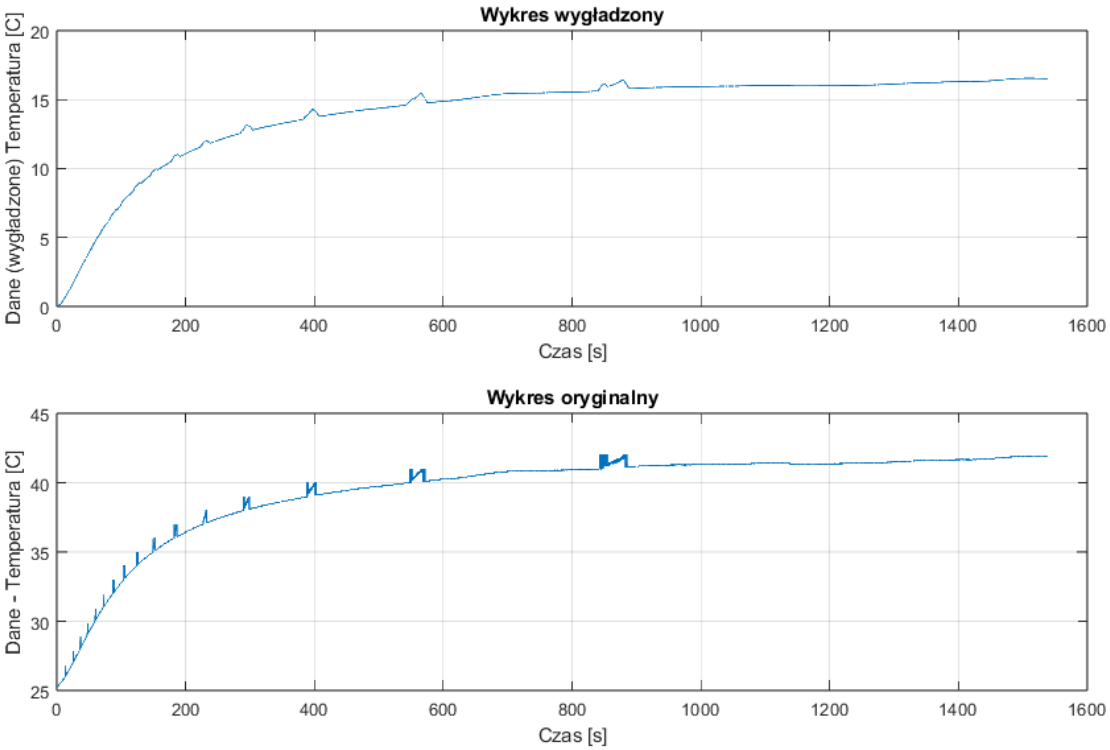
Listing 1: Zbieranie danych do pliku CSV

4 Analiza danych

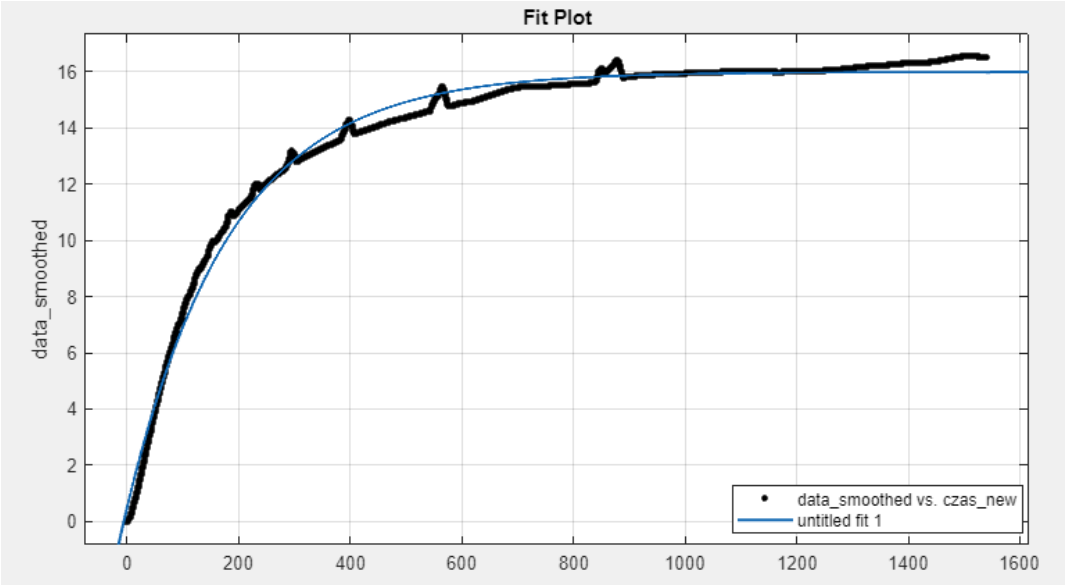
Po zebraniu danych przystąpiono do ich analizy w celu określenia parametrów modelu obiektu. Analiza wykazała, że obiekt może być aproksymowany jako układ inercyjny pierwszego rzędu z opóźnieniem transportowym. Model tego typu jest często stosowany w przypadku układów termicznych, gdzie występuje naturalne opóźnienie wynikające z przewodzenia ciepła.

Dane zostały wstępnie przetworzone w MATLAB-ie przy użyciu metody *movmean*, która pozwala na wygładzenie szumu pomiarowego. Następnie, korzystając z narzędzia *Curve Fitter*, dopasowano zebrane dane do modelu matematycznego. W wyniku tego procesu wyznaczono następujące parametry modelu:

- Stała czasowa T, określająca dynamikę układu,
- Wzmocnienie K, opisujące reakcję obiektu na zmianę napięcia wejściowego,
- Opóźnienie transportowe θ, wynikające z czasu potrzebnego na propagację ciepła.



Rysunek 5: Dane pobrane z pliku CSV, wygładzone oraz surowe



Rysunek 6: Aproksymacja nakładką curve fitter, z wykorzystaniem custom equasion: K * (1 - exp(-(x - theta) / T))

Wynikiem aproksymacji otrzymaliśmy następujące nastawy:

```
val =
     General model:
     val(x) = K*(1-exp(-(x-theta)/T))
     Coefficients (with 95% confidence bounds):
       K =       15.99  (15.99, 16)
       T =       186.9  (186.3, 187.6)
       theta =     -6.049  (-6.458, -5.64)
```

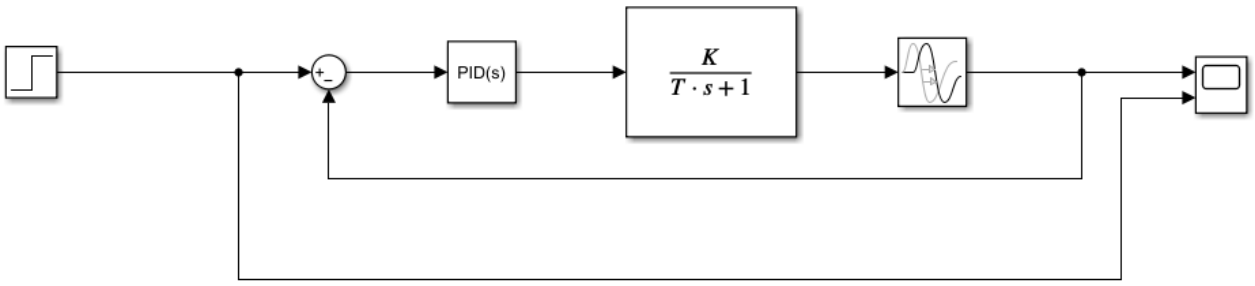
Rysunek 7: Wyniki Modelu Obiektu

Z dokładność aproksymacji wyniosła 95%

Coefficients and 95% Confidence Bounds

5 Dobór parametrów regulatora PID

Na podstawie uzyskanego modelu obiektu przystąpiono do projektowania regulatora PID. W celu dobrania parametrów regulatora stworzono model układu zamkniętego w środowisku *Simulink*.



Rysunek 8: Schemat blokowy układu zamkniętego

Narzędzie *PID Tuner*, dostępne w pakiecie *Control System Toolbox*, pozwoliło na szybkie i efektywne dobranie optymalnych wartości współczynników regulatora. Otrzymane wartości parametrów regulatora PID to:

- Wzmocnienie proporcjonalne $K_p = 1.09248012868956$,
- Wzmocnienie całkujące $K_i = 0.0187249852608489$,
- Wzmocnienie różniczkujące $K_d = 1.14899058414246$.

6 Dobór parametrów miktorkontrolera

- Użyty timer to TIM4 ustawiony w trybie przewań, działający co 100Hz.
- Komunikacja USART3 przebiega asynchronicznie w trybie Recive and Transmit, co umożliwia komunikację w obu kierunkach

7 Implementacja regulatora na mikrokontrolerze

Implementacja regulatora PID została zrealizowana na mikrokontrolerze STM32. Poniżej przedstawiono kluczowy fragment kodu odpowiedzialnego za realizację algorytmu PID:

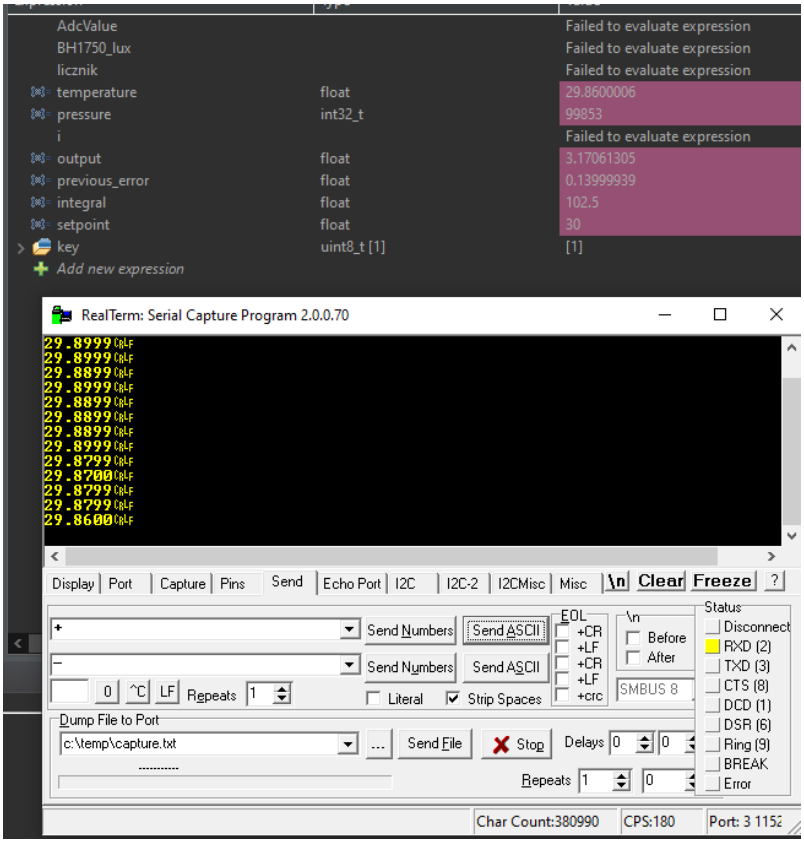
```c
void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim)
{
    BMP280_ReadTemperatureAndPressure(&temperature, &pressure);

    int temp_int = (int)temperature;
    int temp_frac = (int)((temperature - temp_int) * 10000);
    snprintf(mag, sizeof(mag), "%d.%02d\r\n", temp_int, temp_frac);
    HAL_UART_Transmit(&huart3, (uint8_t*)mag, strlen(mag), HAL_MAX_DELAY);

    float error = setpoint - temperature;

    integral += error;

    float derivative = error - previous_error;

    output = (Kp * error) + (Ki * integral) + (Kd * derivative);

    previous_error = error;

    if (output > 1000) {
        output = 1000;
    } else if (output < 0) {
        output = 0;
    }

    __HAL_TIM_SET_COMPARE(&htim3, TIM_CHANNEL_2, (int)output);
}
```

Listing 2: Implementacja regulatora PID na mikrokontrolerze

8 Sterowanie temperaturą

Sterowanie temperaturą w systemie odbywa się za pomocą dwóch przycisków: '+' i '-', które umożliwiają regulację temperatury o 0,5°C. Początkowa wartość temperatury jest zaokrąglana do najbliższej wielokrotności 0,5°C, co zapewnia stabilność i precyzyjność sterowania.

Komunikacja z systemem realizowana jest przez port szeregowy USART, co pozwala na interakcję za pomocą aplikacji takich jak RealTerm. Po odebraniu znaku '+' lub '-', wartość temperatury (setpoint) jest zwiększana lub zmniejszana o 0,5°C. Każda zmiana wartości jest następnie zaokrąglana do najbliższej wielokrotności 0,5°C, co zapewnia spójność

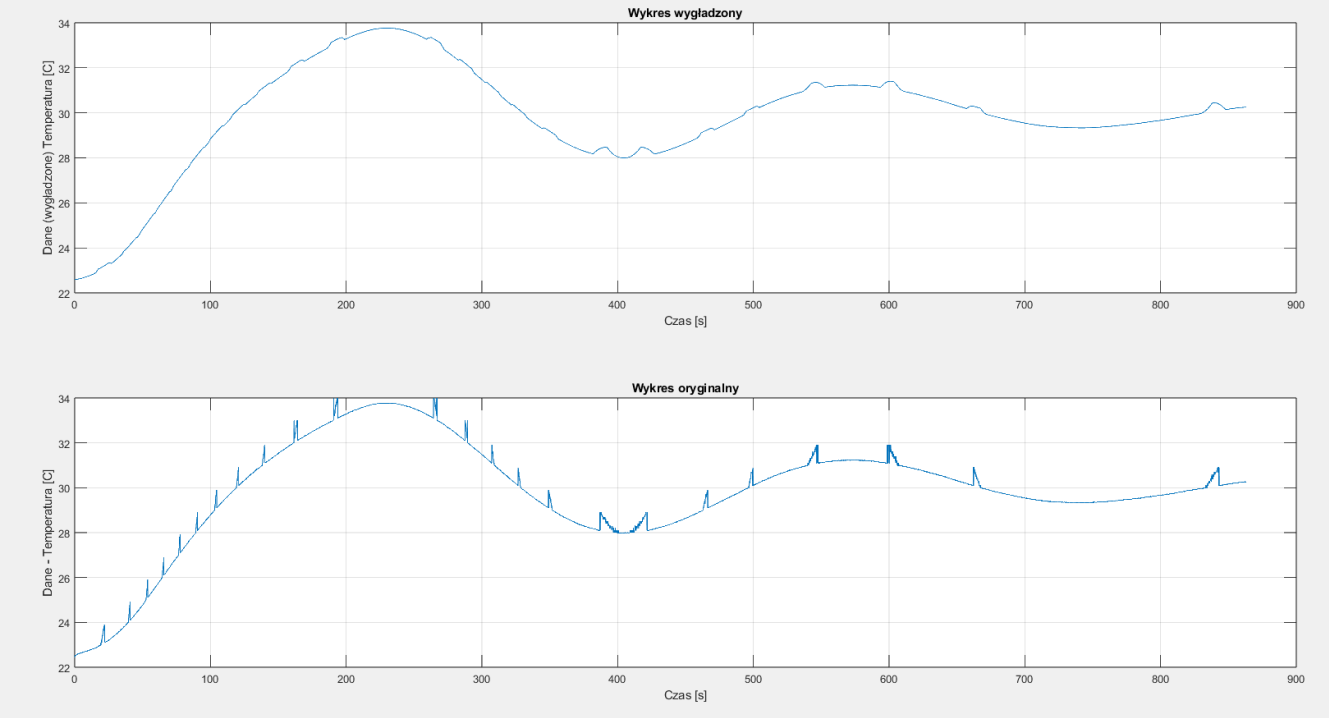


Rysunek 9: Zdjęcie obrazujące użycie terminala oraz fragment zakładki Live Expression z aplikacji STM32CubeIDE

```c
BMP280_ReadTemperatureAndPressure(&temperature, &pressure);

 setpoint = (float)( (int)(temperature / 0.5 + 0.5) ) * 0.5;


 /* USER CODE END 2 */

 /* Infinite loop */
 /* USER CODE BEGIN WHILE */
 while (1)
 {
   /* USER CODE END WHILE */
   /* USER CODE BEGIN 3 */
   HAL_StatusTypeDef status = HAL_UART_Receive(&huart3, key, 1, 1);
   if (status == HAL_OK)
     switch (key[0]) {
           case '+':
             {
               setpoint = setpoint + 0.5;
               break;
             }
           case '-':
             {
               setpoint = setpoint - 0.5;
               break;
             }
 }
 }
 /* USER CODE END 3 */
}
```

Listing 3: Sterowanie temperaturą

9 Odpowiedź Układu Regulacji



Rysunek 10: Odpowiedź układu zamkniętego z regulatorem PID

Układ jest stabilny, lecz charakteryzuje się oscylacjami i przeregulowaniem bliskim 50%. Wynika to z zastosowanego regulatora, który jest nieco zbyt agresywny, prowadząc do dużych wahań wokół zadanej wartości. Dostosowanie parametrów regulatora pozwoliłoby na uzyskanie bardziej płynnej i precyzyjnej regulacji, minimalizując oscylacje i stabilizując pracę systemu. Długa stabilizacja wynika z dużej stałej czasowej (ze względu na małą saturację sygnału wejściowego)

10 Podsumowanie

Udało się uzyskać uchyb ustalony bliski zera (po czasie rzędu 30 minut) jednak czas jego stabilizacji pozostaje, w mojej ocenie, zbyt długi. Na ten wynik wpływ miały przede wszystkim słabej jakości dane zebrane z czujnika oraz rozbieżność rzędu 5% już na etapie modelowania obiektu. Najistotniejszym czynnikiem ograniczającym szybkość stabilizacji jest jednak duża stała czasowa obiektu, wynosząca około 3 minut. Taki czas odpowiedzi obiektu znacznie utrudnia uzyskanie szybkiej reakcji systemu, co wskazuje na konieczność dalszej optymalizacji zarówno w zakresie poprawy jakości danych, jak i dopracowania modelu obiektu. Być może zastosowanie bardziej zaawansowanych algorytmów sterowania mogłoby poprawić czas stabilizacji, a także zwiększyć dokładność regulacji.

Ostatnią kwestią jest zmiana zbyt łatwa zmiana parametrów obiektu. W celu poprawy tego aspektu, obłożyłem obiekt papierem do pieczenia w sposób, który nie przemieszczał rezystora nad czujnikiem. Taki zabieg miał na celu także zmniejszenie stałej czasowej obiektu poprzez ograniczenie strat ciepła, co pozwoliło na szybszą reakcję systemu. Działanie to było próbą poprawienia stabilności obiektu i zredukowania opóźnień w odpowiedzi na zmiany temperatury.