SPRAWOZDANIE Z PROJEKTU		Rok akademicki: 2024/25
Przedmiot: **Monitoring i Sterowanie w Inżynierii Środowiska**		
Temat projektu: **Mikroprocesorowy system sterowania i pomiaru. Regulator PID.**		Termin zajęć: **czwartek 9:45-11:15**
Wydział, kierunek, semestr, grupa: **WARiE, AiR, 7, 1**	Imię, nazwisko, numer indeksu: **Jędrzej Czujewicz, 151110 Adam Ciemiński, 151133**	Data wykonania: **10.12.2024**

1. Opis:

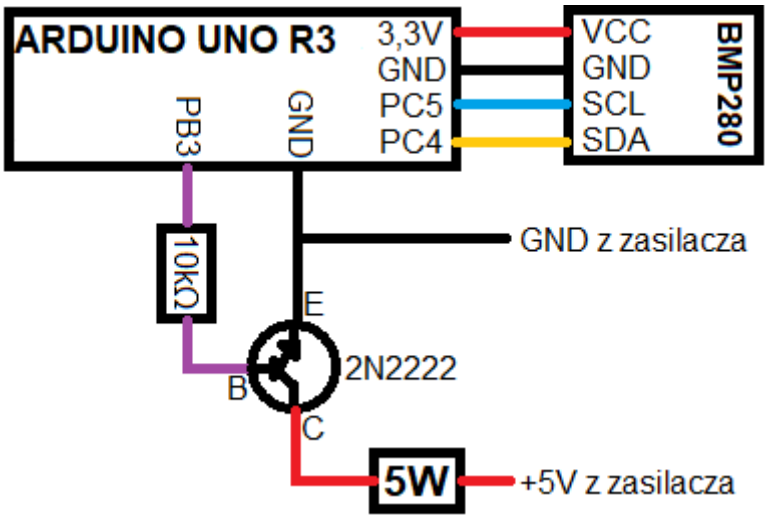
Tematem projektu jest pomiar temperatury generowanej przez rezystor mocy, a następnie sterowanie nią zgodnie z temperaturą zadaną przez użytkownika. Na podstawie pomiarów został wyznaczony przybliżony modelu obiektu, a następnie zostały dobrane optymalne parametry regulatora PID. Użytkownik ma możliwość zadania temperatury oraz podglądu kluczowych parametrów w prostym interfejsie graficznym.

Projekt zrealizowany został z użyciem mikrokontrolera Arduino UNO jako głównej jednostki zarządzającej. Do jego zaprogramowania użyte zostało środowisko Arduino IDE. Graficzny interfejs użytkownika oraz identyfikacja systemu zostały stworzone z wykorzystaniem Python'a.

Wszystkie napisane i użyte do projektu kody Arduino i Python zostały opublikowane w repozytorium Github - https://github.com/JedrzejCzujewicz/miswis_pro

2. Lista wykorzystanych elementów i ideowy schemat połączeń:

- 1x Arduino UNO R3
- 1x Płytka stykowa
- 1x Moduł zasilający do płytek stykowych MB102
- 1x Układ BMP280
- 1x Rezystor ceramiczny 5W 39Ω
- 1x Rezystor 10kΩ
- 1x Tranzystor NPN 2N2222
- kilka przewodów do zrealizowania połączeń



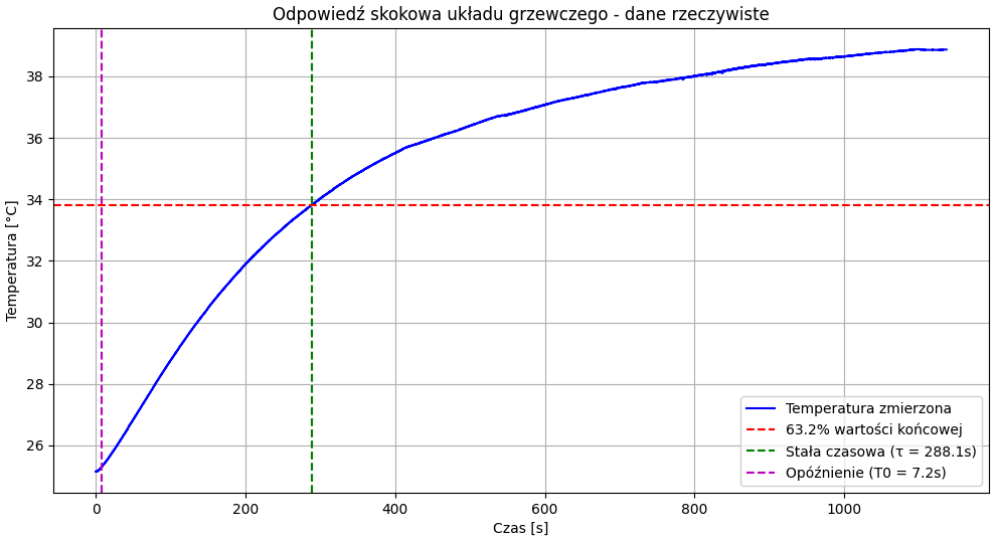
Schematyczny rysunek przedstawiający płytkę Arduino UNO połączoną ze wszystkimi elementami wykorzystanymi w projekcie.

3. Opis procesu programowania:

1. Pierwszym krokiem był pomiar i zapis odpowiedzi skokowej modelu. Aby to zrobić, w kodzie programu Arduino (data_collector.ino) sygnał sterujący został ustawiony na wartość maksymalną dla pinu Arduino w trybie PWM (PB3), czyli 255. Tym samym został umożliwiony pełny przepływ prądu przez rezystor. Następnie odczyty czujnika BMP280, były przesyłane poleceniem Serial.print(), gdzie dalej były przechwytywane i zapisywane do pliku .txt przy użyciu Python (save_data_to_txt.py)
2. Następnym krokiem była aproksymacja modelu na podstawie zapisanej w pliku .txt odpowiedzi skokowej (identification.py). Otrzymane parametry to:
   a. opóźnienie T_0=7,2 s,
   b. stała czasowa τ=288,1 s,
   c. wzmocnienie statyczne K=0.0538 °C/jednostkę PWM

   W dalszej części zostały wyznaczone parametry regulatora PID:
   a. Kp=892.44
   b. Ki=61.97
   c. Kd=3212.78
3. W głównym programie Arduino (main.ino) realizującym URA, należało zastosować anty-windup w związku z wysokimi wartościami parametrów regulatora PID, aby sygnał sterujący nie wykroczył poza dopuszczalny zakres wartości PWM (od 0 do 255).
4. Graficzny interfejs użytkownika, umożliwiający zadawanie temperatury oraz monitorowanie temperatury aktualnej i sygnału sterującego został stworzony w Python (main.py) z wykorzystaniem bibliotek: tkinter i matplotlib

4. Zrzuty ekranu z działającego projektu:

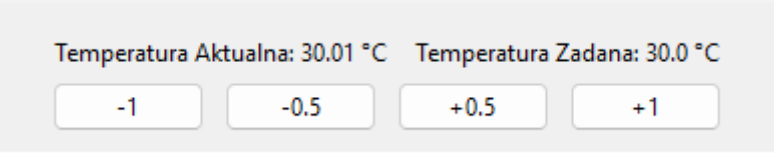


Zrzut ekranu z procesu identyfikacji obiektu.

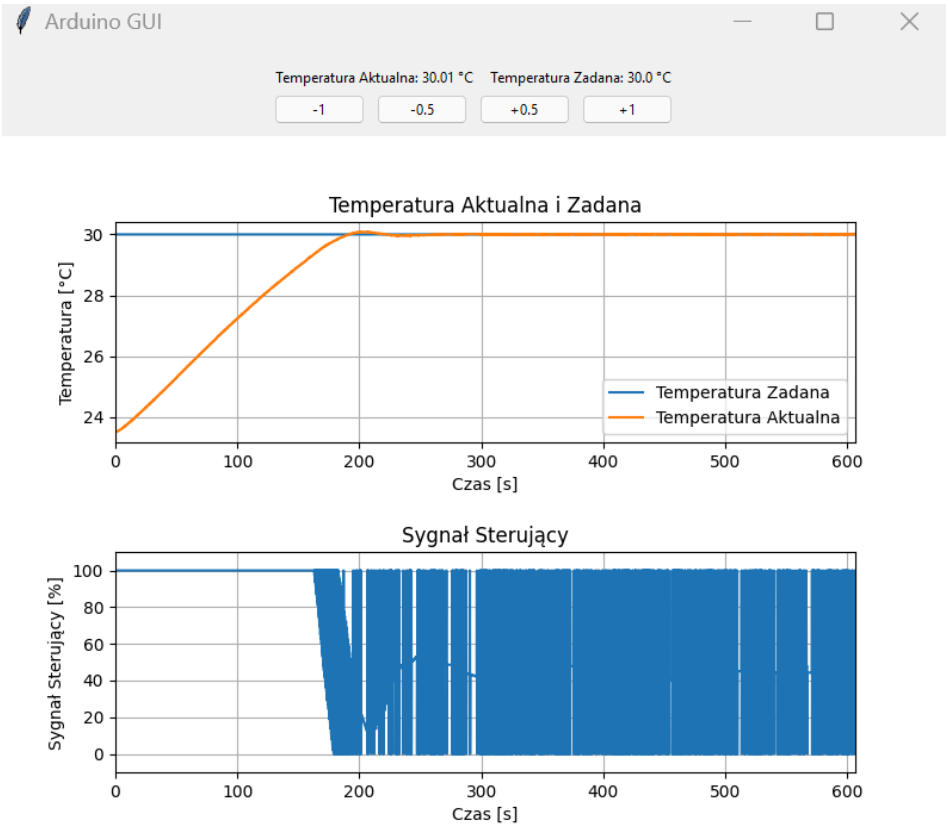
```
Parametry układu:
Temperatura początkowa: 25.15°C
Temperatura końcowa: 38.87°C
Stała czasowa (τ): 288.1 s
Wzmocnienie statyczne (K): 0.0538 °C/jednostkę PWM
Opóźnienie transportowe (T0): 7.2 s
```

```
Wyznaczone parametry regulatora PID:
Kp = 892.44
Ki = 61.97
Kd = 3212.78
```

Otrzymane parametry na podstawie przeprowadzonej identyfikacji.



W GUI możemy podglądać aktualny odczyt temperatury w postaci liczbowej i zadawać temperaturę z skokiem ±0.5 °C lub ±1.0 °C.



Zrzut ekranu całego okna GUI. Widać dwa wykresy: na pierwszym wykreślone są zmiany temperatur zadanej i aktualnej w czasie, natomiast na drugim zmiany sygnału sterującego w czasie (w stanie ustalonym widać wiele impulsów sygnału sterującego, które można by zniwelować dodając strefę nieczułości na drobne odchylenia temperatury aktualnej od zadanej).

5. Podsumowanie

W ramach zadania projektowego udało się zbudować działający układ regulacji automatycznej z regulatorem PID bazujący na mikroprocesorowym systemie sterowania i pomiaru temperatury z graficznym interfejsem użytkownika korzystając z Arduino i Python'a.

Potencjalne ścieżki rozwoju projektu:

- dodanie wentylatora, który pomagałby i przyspieszał proces zmniejszania temperatury
- podłączenie wyświetlacza (np. LCD, 7 segmentowych itp), jego implementacja w kodzie i wyświetlanie na nim takich informacji jak: temperatura aktualna i zadana
- dodanie w stanie ustalonym histerezy, aby zniwelować występowanie impulsów sygnału sterującego, przy minimalnych odchyleniach temperatury zmierzonej od temperatury zadane