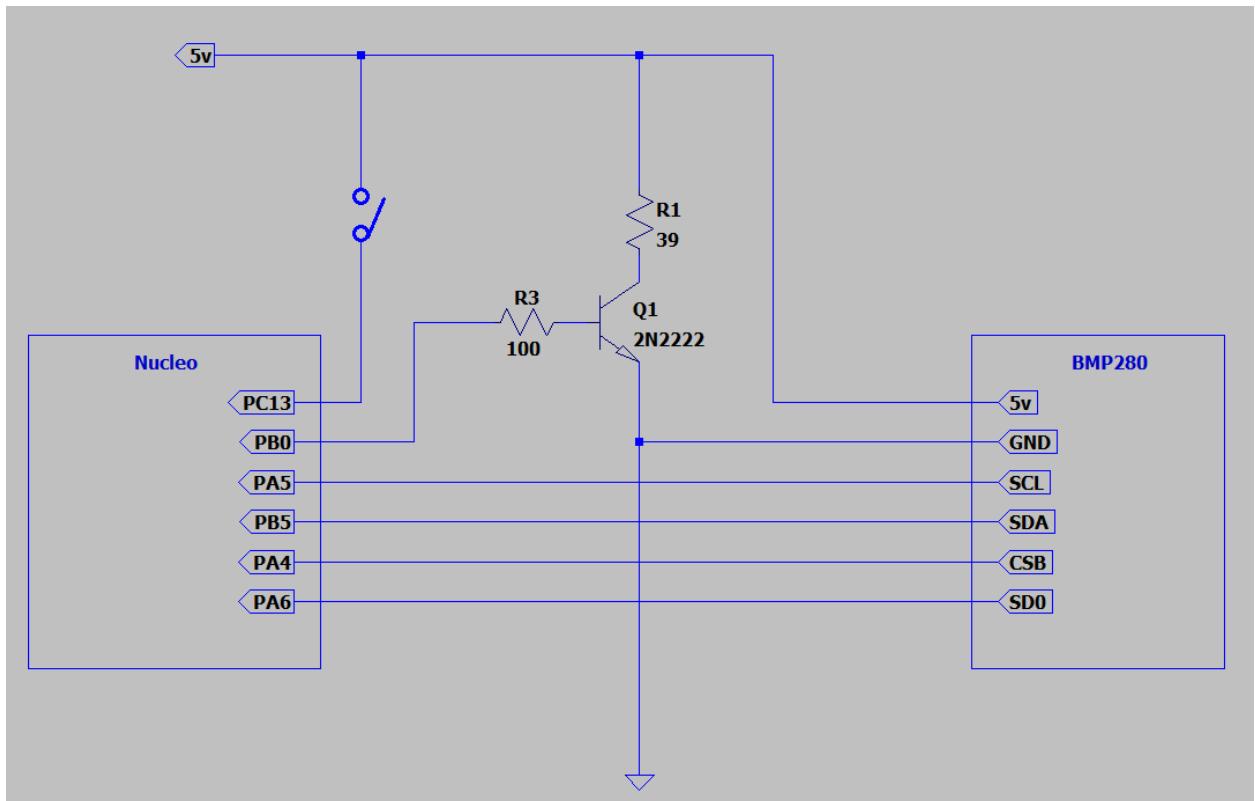


PROJEKT Z LABORATORIUM		Rok akademicki 2022/23
Przedmiot: SYSTEMY MIKROPROCESOROWE		
Temat Ćwiczenia: Mikroprocesorowy system sterowania i pomiaru. Regulator PID.	Nr zestawu: -	Termin zajęć: wtorek 15:10 – 16:40
Wydział, kierunek, semestr, grupa: WI, AiR, 5, A5-L10	Imię i Nazwisko: 1. Wojtek Karolczak 2. Robert Tomczyk	Punkty:
Data wykonania Ćwiczenia: 24.01.2023		

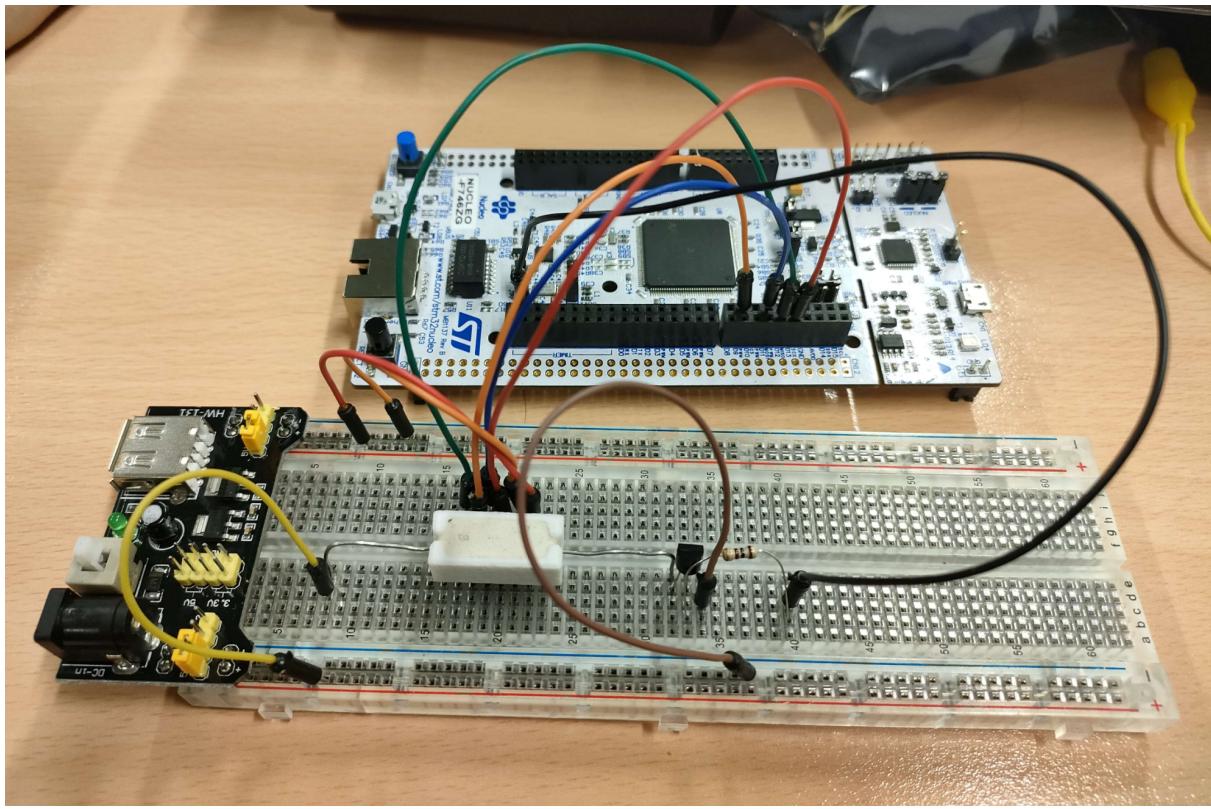
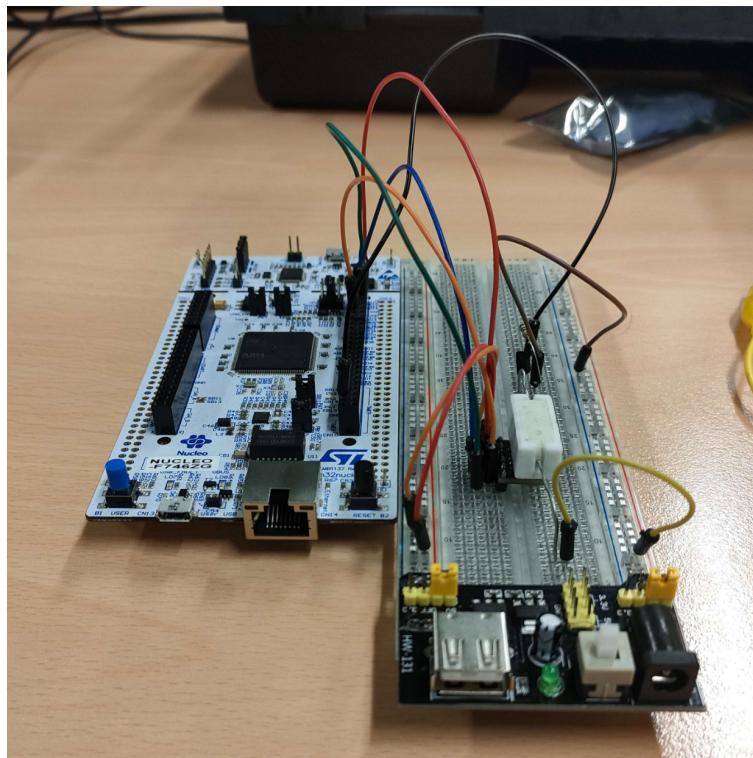
Schemat układu



Rys. 1 - Schemat układu, rezystor R3 = 100Ω

Opis schematu

Tranzystor NPN jest kluczowany z wykorzystaniem wyjścia GPIO PB0 (do którego jest podłączony rezystor 100 Ω). Aby załączyć funkcję nagrzewania dodane zostało zabezpieczenie w postaci przycisku załączającego (User B1), który jest wbudowany na płytce nucleo.



Wyznaczanie transmitancji obiektu i nastaw regulatora PID

Dane odpowiedzi obiektu zostały zebrane za pomocą Telemetry Viewer v0.7 oraz poniższego kodu na komunikację szeregową

```
float temperature;
int32_t pressure;

void transmit(char* stream, size_t length)
{
    HAL_UART_Transmit(&huart3, (uint8_t*)stream, length, 500);
    HAL_GPIO_WritePin(LD3_GPIO_Port, LD3_Pin, GPIO_PIN_SET);
    HAL_Delay(50);
    HAL_GPIO_WritePin(LD3_GPIO_Port, LD3_Pin, GPIO_PIN_RESET);
}

// ... //

while (1)
{
    // DATA READ
    BMP280_ReadTemperatureAndPressure(&temperature, &pressure);

    // SENDING DATA OUT
    sprintf(text, sizeof(text), "% .3f\n\r", (double)temperature);
    transmit(text, strlen(text));
}
```

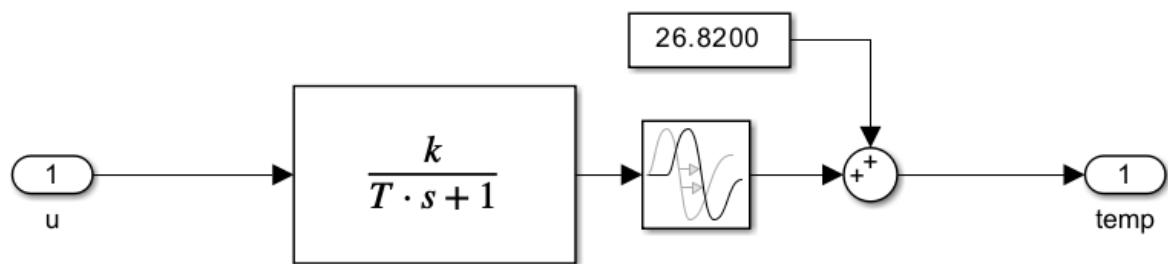
Przy użyciu zebranych danych oraz narzędzia curve fitting tool w środowisku matlab, wyznaczone zostały parametry k, T oraz T_o o wartościach:

-k = 202.4

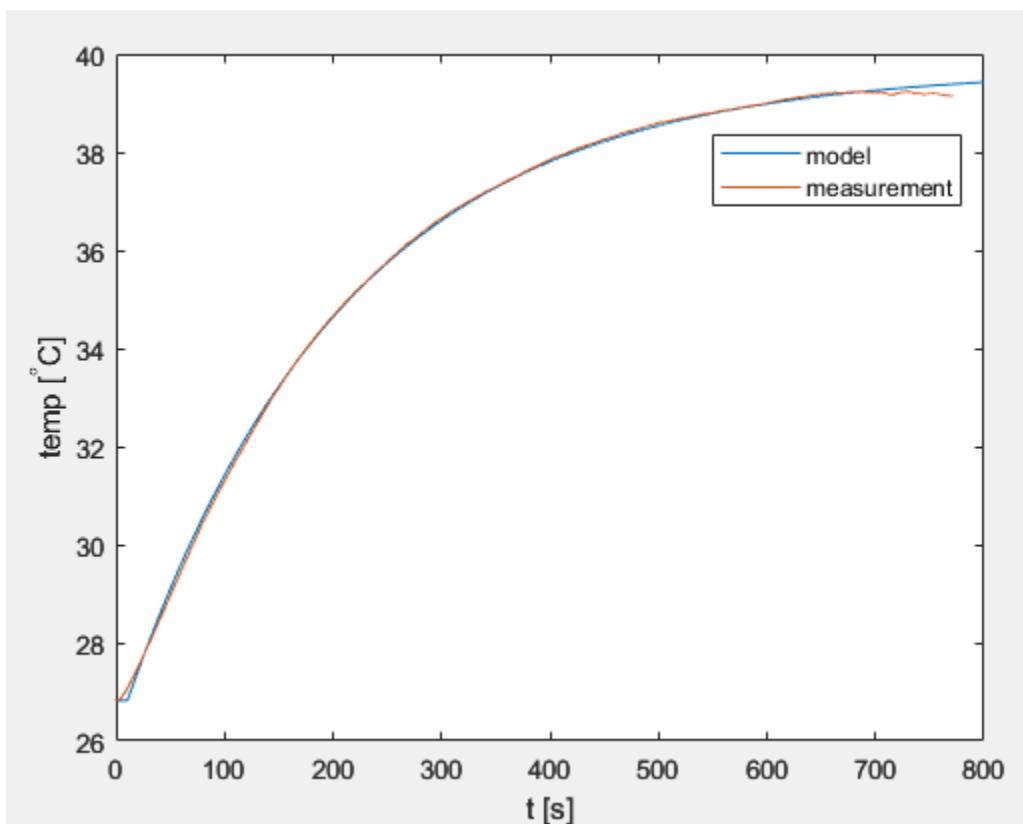
-T = 10.62

- T_o = 2.573

Następnie obiekt został zamodelowany w środowisku simulink aby potwierdzić wartości powyższych parametrów



Rys.2 Schemat obiektu w środowisku simulink

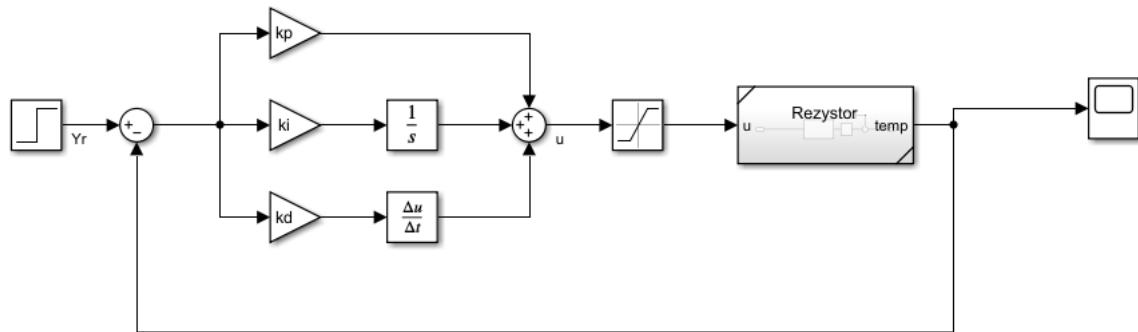


Rys.3 Wynik symulacji porównany z realnymi danymi

Maksymalna temperatura osiągana przez rezystor to niewiele poniżej 40 °C, a temperatura otoczenia wynosiło mniej więcej 26 °C. Na tej podstawie została wybrana nastawa $y_r = 33$ °C

Implementacja regulatora PID

Zamodelowany model został podłączony pod regulator PID na którego wyjściu znajduje się saturator który symuluje maksymalną wartość napięcia 5V.



Rys. 4 Schemat URA w środowisku simulink

Metodą aproksymacji eksperimentalnej zostały wybrane poniższe parametry:

- kp = 4
- ki = 0.006
- kd = 5

```
int set_value = 33;
float e_sum_pid;
float last_e_pid = 0;

float pid(float y, float set, float* e_sum, float* last_e)
{
    float kp = 4;
    float ki = 0.006;
    float kd = 5;
    float Tp = 0.005;

    float e = set - y;
    (*e_sum) = (*e_sum) + e * Tp;
    float e_diff = e - (*last_e);
    (*last_e) = e;

    float p = kp * e;
    float i = ki * (*e_sum);
    float d = kd * e_diff;

    return p + i + d;
}
```

Funkcja PID wywoływana jest w przerwaniu TIM4

```
void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef* htim)
{
    // PID
    float u = pid(temperature, (float)set_value, &e_sum_pid, &last_e_pid);
    if (u > 5) u = 5;
    if (u < 0) u = 0;

    u /= 5;

    if (!enabled)
    {
        u = 0;
        e_sum_pid = 0;
    }

    // SET U
    __HAL_TIM_SET_COMPARE(&htim3, TIM_CHANNEL_3, u * 19999);
}
```

Dodatkowa funkcjonalność programu

Czytanie komend użytkownika zostało zaimplementowane z użyciem komunikacji UART:

```
#define COUNTOF(_BUFF_) (sizeof(_BUFF_) / sizeof(*(_BUFF_)))
#define RESPOND(str) transmit(str, COUNTOF(str))

// ... //

uint8_t key[4];
HAL_StatusTypeDef status = HAL_UART_Receive(&huart3, key, 4, 100);

if (status == HAL_OK)
{
    RESPOND("Received ");
    transmit((char*)key, 5);
    RESPOND("\n\r");

    if (key[0] == 'S')
    {
        RESPOND("Trying to set the temperature\n\r");
        set_value = (key[1] - '0') * 100 + (key[2] - '0') * 10 + key[3] - '0';
    }
}
```

Regulator może zostać włączony lub wyłączony z użyciem przycisku.

Stan ten jest prezentowany z użyciem diody LED (LD2)

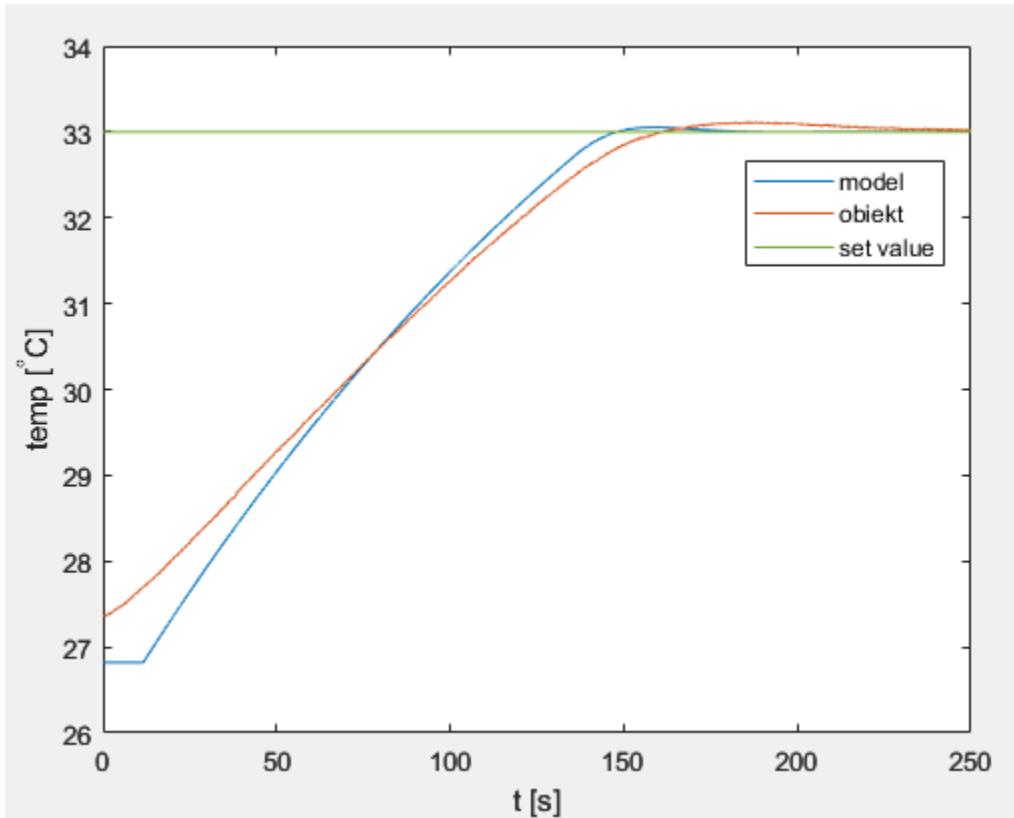
```
GPIO_PinState buttonState = HAL_GPIO_ReadPin(USER_Btn_GPIO_Port, USER_Btn_Pin);
if (buttonState == GPIO_PIN_SET)
{
    enabled = !enabled;
    HAL_Delay(100);
}
```

Akwizycja danych z użyciem Telemetry Viewer została rozszerzona o dodatkowe sygnały wyjściowe, wartość ustaloną, sygnał sterowania, całkę uchybu oraz pochodną

```
snprintf(text, sizeof(text), "%.3f,%d,%.3f,%.3f,%.3f\n\r", (double)temperature,
set_value, (double)u_DEBUG, (float)e_sum_pid, (float)e_diff_DEBUG);
transmit(text, strlen(text) + 1);
```

Porównanie symulacji z realnym obiektem

Symulacja lekko różni się od realnego obiektu ze względu na inną temperaturę początkową, nie idealne połączenie między rezystorem a sondą temperaturową oraz niedokładnościami w projektowanym modelu. Natomiast uzyskany wynik dalej mieści się w poniżej 1% uchybu.



Rys. 5 Porównanie symulacji z realnym obiektem w środowisku matlab