Uniwersytet Wrocławski

Wydział Fizyki i Astronomii

Bartłomiej Matyszkowicz

System do testowania i demonstracji regulatora temperatury PID w układzie z wentylatorem i elementem grzejnym

System for PID Controller’s testing and demonstration in a system with a fan and heating element

Praca inżynierska wykonana pod kierunkiem  
 dr inż. Radosława Wasielewskiego

Wrocław 2019

Streszczenie

Założeniem pracy jest zaprojektowanie i zaimplementowanie układu pomiarowego służącego studentom do poznania dwóch różnych regulatorów temperatury oraz metod ich strojenia. Wstępnie omówione zostały zastosowanie i różnice między regulatorem dwustawnym a regulatorem PID oraz założenia projektu.

Założenia projektu obejmują stworzenie samego układu pomiarowego oraz aplikacji komputerowej służącej do sterowania parametrami pracy układu. Układ pomiarowy składa się z komory pomiarowej i systemu wbudowanego regulującego pracę elementów sterowalnych, takich jak element grzewczy i chłodzący wewnątrz komory. System wbudowany oparty został o platformę sprzętową Arduino. Praca opisuje podłączenie elementów do platformy sprzętowej oraz cel i działanie każdego z komponentów wykorzystanego do budowy układu. Następnie szczegółowo opisane jest oprogramowanie systemu wbudowanego napisane w języku C++, odpowiadające za pracę regulatora i odczytywanie informacji z czujnika temperatury.

Aplikacja komputerowa została napisana w języku Python i jej zadaniem jest komunikacja pomiędzy uzytkownikiem a systemem wbudowanym oraz wizualizacja danych odbieranych zukładu w postaci wykresu temperatury. W pracy opisane jest wygląd interfejsu użytkownika oraz możliwości jakie oferuje, sposób w jaki komunikuje się z systemem wbudowanym, zewnętrzne biblioteki z jakich korzysta, sposób działania jej funkcji oraz powód użycia wielowątkowości.

Następnie opisane są różne metody strojenia regulatora PID oraz przykładowe ich wykorzystanie jako ćwiczenie związane z układem pełniącym rolę makiety dydaktycznej. Przeprowadzone zostały również testy metod strojenia w celu potwierdzenia poprawności ich zastosowania przy układu.

Abstract

Aim of the thesis is design and implementation of measurement system, serving students to learn two different temperature controllers and their tuning methods. The application and differences between the two-state controller and the PID controller and the assumptions of the project have been discussed initially.

The assumptions of the project include the creation of the measurement system and a computer application used to control the system parameters. The measurement system consists of a measuring chamber and an embedded system that regulates the workflow of controllable elements, such as the heating and cooling element inside the chamber. The embedded system was based on the Arduino hardware platform. The thesis describes the connection of elements to the hardware platform and the purpose and operation of each of the components used to build the system. Next, the software of the embedded system written in C++ is described in detail, which is responsible for the controller's work and reading information from the temperature sensor.

The computer application was written in Python language and its task is to communicate between the user and the embedded system and to visualize the data received in the form of a temperature chart. The thesis describes the appearance of the user interface and the options it offers, the way it communicates with the embedded system, the external libraries it uses, the way its functions work, and the reason for using multithreading.

Next, various methods of tuning the PID controller are described and, for example, their use as an exercise related to the system serving as a teaching model. The tests of tuning methods were also carried out in order to confirm the correctness of their application in the system’s tuning.

Spis treści

[1. Wstęp 5](#_Toc1857566)

[1.1. Regulacja dwustawna 5](#_Toc1857567)

[1.2. Regulator PID 6](#_Toc1857568)

[2. Cel i założenia pracy 7](#_Toc1857569)

[2.1. Układ pomiarowy 7](#_Toc1857570)

[2.2. Sterowanie układem 8](#_Toc1857571)

[3. Implementacja 10](#_Toc1857572)

[3.1. Układ pomiarowy 10](#_Toc1857573)

[3.1.1. Element grzejny 11](#_Toc1857574)

[3.1.2. Element chłodzący 11](#_Toc1857575)

[3.1.3. Czujnik temperatury 14](#_Toc1857576)

[3.2. Sterowanie układem 16](#_Toc1857577)

[3.2.1. Arduino IDE 16](#_Toc1857578)

[3.2.2. Python 19](#_Toc1857579)

[4. Pracownia studencka 23](#_Toc1857580)

[4.1. Dobór nastaw 23](#_Toc1857581)

[4.1.1. Strojenie ręczne 23](#_Toc1857582)

[4.1.2. Pierwsza metoda Zieglera-Nicholsa 24](#_Toc1857583)

[4.1.3. Druga metoda Zieglera-Nicholsa 26](#_Toc1857584)

[4.2. Przykładowe ćwiczenie 27](#_Toc1857585)

[5. Testy układu 28](#_Toc1857586)

[Bibliografia 32](#_Toc1857587)

[Dodatki 33](#_Toc1857588)

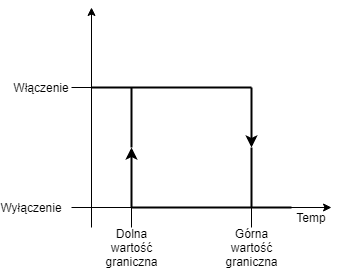
[Eksport 33](#_Toc1857589)

1. Wstęp

Wiele budynków użytku prywatnego, komercyjnego bądź publicznego wyposażona jest w instalacje umożliwiające zarówno ogrzewanie jak i klimatyzację wewnątrz budynku. Zadaniem takich instalacji jest zagwarantowanie odpowiednich warunków do komfortowego użytkowania obsługiwanych pomieszczeń, jednak priorytety różnych instalacji mogą być różne- jedne mogą skupiać się na jak najbardziej ekonomicznej pracy, inne mogą mieć za cel jak najszybszą odpowiedź na zmiany temperatury, z kolei jeszcze inne mogą priorytetyzować utrzymanie temperatury na zadanym poziomie z jak najmniejszymi różnicami od normy. Niezależnie jednak od przeznaczenia instalacji łączy je fakt, że każda z nich wymaga dostosowanego do warunków otoczenia w którym pracuje systemu regulacji, którego przeznaczeniem jest kontrola otoczenia i sterowanie instalacją. Sterowanie instalacją może być zrealizowane na dwa różne sposoby: regulacji dwustawnej oraz regulatora PID.

* 1. Regulacja dwustawna

Sterownik regulacji dwustawnej, pracuje w dwóch stanach – włączonym i wyłączonym. Parametrami jego pracy są dwie wartości graniczne definijące przedział temperatury zwany histerezą w jakim powinien pracować układ. Na rysunku niżej przedstawiona została charakterystyka pracy przykładowego regulatora ogrzewania pomieszczenia – gdy temperatura jest poniżej dolnej wartości granicznej regulator uruchamia ogrzewanie w celu wyrównania temperatury do wartości docelowej zdefiniowanej szerokością histerezy. Gdy temperatura przekroczy górną wartość graniczną ogrzewanie zostaje wyłączone i pomieszczenie zaczyna się schładzać. Gdy temperatura ponownie osiągnie dolną wartość graniczną cykl pracy zostaje powtórzony.



*Rysunek 1:* *Charakterystyka grzewczego regulatora dwustanowego*

Chłodzące regulatory dwustawne działają na odwróconej histerezie – układ chłodzący zostaje uruchomiony w momencie, gdy temperatura przekroczy górną wartośc graniczną i zostaje wyłączony po spadku poniżej dolnej wartości granicznej.

* 1. Regulator PID

Regulator PID, czyli regulator proporcjonalno-całkująco-rożniczkujący (ang. Proportional-Integral-Derivative) jest regulatorem, który minimalizuje różnicę, zwaną czasami uchybem, pomiędzy temperaturą docelową a zmierzoną za pomocą 3 działań opartych o człony regulatora, zwane nastawami. Suma tych nastaw jest sygnałem sterującym, za pomocą którego regulator kontroluje urządzenia grzewcze bądź chłodzące.

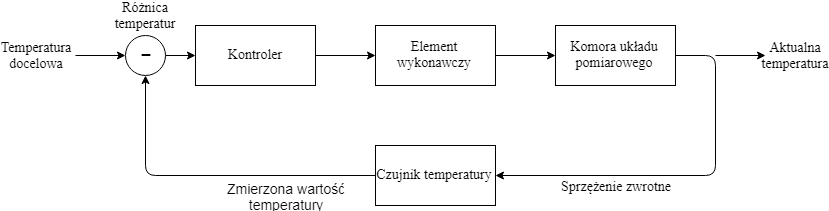
Każda z nastaw regulatora wykonuje inne działanie na sygnale sterującym. Nastawa proporcjonalna wykonuje działanie mnożenia na sygnale sterującym wraz ze wzrostem uchybu, czyli zwiększa odpowiedź układu im większa jest różnica temperatur. Nastawa całkująca sprowadza uchyb do zera dodając lub odejmując do sygnału sterującego pewną wartość zależną od uchybu. W przypadku gdy uchyb jest równy zeru, ta wartość również wynosi zero i sygnał sterujący przestaje się zmieniać, pozwalając układowi na utrzymanie zadanej temperatury. Nastawa różniczkująca wyznacza pochodną z uchybu i na podstawie szybkości zmiany uchybu w czasie modyfikuje sygnał sterujący w celu złagodzenia przestrzału czyli przekroczenia temperatury docelowej.

1. Cel i założenia pracy

Głównym zadaniem niniejszej pracy inżynierskiej jest stworzenie modelowego układu pomiarowego wykorzystującego system wbudowany pracujący jako regulator PID (proporcjonalno-całkująco-różniczkujący), który umożliwia osobie z niego korzystającej zrozumienie sposobu jego działania w praktyce. Taki układ mógłby być prostą makietą dydaktyczną na pracowni elektronicznej dla studentów, umożliwiającą studentowi zapoznanie się z konceptem regulatora PID oraz na poznanie różnych metod strojenia takiego regulatora. Zastosowanie różnych metod strojenia regulatora pozwala studentowi na ich porównanie i wyprowadzenie wniosków, która z metod sprawdza się najlepiej w zapewnionych warunkach. Główną motywacją do powstania pracy jest fakt szerokiego zastosowania regulatorów PID w przemyśle, z racji czego studenci powinni zapoznać się chociaż z podstawą działania takiego regulatora.

* 1. Układ pomiarowy

W założeniach układ pomiarowy, którego zadaniem jest kontrola i regulacja temperatury jest komorą wewnątrz której znajdują się z trzy główne komponenty kontrolowane przez system wbudowany: źródło sygnału wejściowego, element pomiarowy i element wykonawczy ze sprzężeniem zwrotnym. Element pomiarowy czyli czujnik temperatury, który stale poddany jest działaniu sygnału wejściowego, zbiera informacje o temperaturze wewnątrz komory i przekazuje je do kontrolera (systemu wbudowanego), który porównuje je z temperaturą docelową układu i na podstawie ich różnicy steruje elementem wykonawczym w celu zminimalizowania tej różnicy. W wyniku pracy elementu wykonawczego następuje zmniejszenie różnicy temperatur, wykonany zostaje kolejny pomiar i kontroler dostosowuje pracę do nowych warunków.



*Rysunek 2: Schemat regulatora*

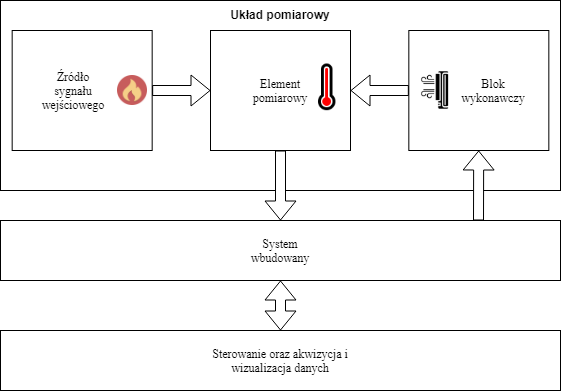
Taki układ jest uniwersalny pod względem dostosowania do potrzeb użytkownika – układ zbudowany z czujnika temperatury, elementu grzejnego i elementu wymuszającego przepływ powietrza pozwala na przeprowadzenie pomiarów w dwóch alternatywnych trybach pracy, w zależności od tego który element jest źródłem sygnału wejściowego, a który elementem wykonawczym. Jedynie czujnik temperatury zawsze będzie pełnił rolę elementu pomiarowego, niezależnie od zastosowanej kombinacji. Innymi słowy taka konstrukcja umożliwia pracę na dwa różne sposoby - gdy źródłem jest element grzejny to znaczy, że czujnik temperatury jest stale nagrzewany i zadaniem elementu wykonawczego jest jego schłodzenie i utrzymanie docelowej temperatury. Natomiast jeśli źródłem jest element chłodzący to celem elementu grzejnego jest dostarczanie odpowiedniej ilości ciepła, aby temperatura układu nie spadła poniżej docelowej.

Rolę kontrolera układu pomiarowego może pełnić system wbudowany pracujący jako regulator PID. Regulacja PID jest prostym systemem jednowymiarowej regulacji – układ steruje pojedyńczym elementem na podstawie danych otrzymywanych z jednego źródła. W przypadku omawianego układu źródłem danych jest czujnik temperatury, element wykonawczy jest natomiast zależny od wybranego sposobu pracy układu.

Czas reakcji i precyzja regulatora w pełni zależy od 3 nastaw. Są to kolejno składowa proporcjonalna, całkująca i różniczkująca. Odpowiedni dobór tych trzech parametrów pozwala na stabilną pracę układu i jest podstawowym problemem każdego regulatora PID.

* 1. Sterowanie układem

Regulacja i kontrola regulatora powinna odbywać się z poziomu aplikacji komputerowej umożliwiającej użytkownikowi odpowiedni dobór nastaw regulatora oraz temperatury docelowej układu. Zadaniem aplikacji jest też pobranie danych z czujnika temperatury i przedstawienie ich na wykresie, co powinno pozwolić na szybką analizę poprawności dobranych parametrów na podstawie zachowania temperatury.



*Rysunek 3: Schemat całego układu*

Jedną z zakładanych opcji aplikacji jest możliwość przełączenia układu z regulacji PID na pracę w trybie prostego przełącznika dwustawnego. W takim trybie regulator włączałby chłodzenie po przekroczeniu temperatury docelowej i wyłączał je gdy ta spadnie poniżej zadanego poziomu. Udostępnienie takiego trybu pracy pozwoli użytkownikowi na porównanie różnic między zachowaniem regulatora i wachaniami temperatury przy różnych sposobach regulacji, oraz na ocenienie w jakich sytuacjach prostszy regulator może być przydatniejszy od wymagającego regulacji kontrolera PID.

Sama aplikacja nie powinna bezpośrednio wpływać na to w jaki sposób zachowują się poszczególne elementy układu – cel ten spełnia system wbudowany sterowany z poziomu aplikacji. Aplikacja powinna przesyłać do niego dane takie jak temperatura docelowa oraz nastawy PID, zadaniem systemu jest natomiast na podstawie tych danych oraz temperatury odebranej z czujnika wyznaczenie odpowiednich parametrów pracy sterowanych komponentów w taki sposób, aby temperatura układu ustabilizowała się na poziomie temperatur docelowej. Aby system mógł spełniać swoje zadania platforma, na której jest on zbudowany musi mieć możliwość odbierania i wysyłania sygnałów by być w stanie sterować elementami i odczytywać dane z czujnika oraz potrafić przeprowadzać obliczenia konieczne do wyznaczania parametrów pracy sterowanych elementów. Spełnienie tych warunków pozwala systemowi wbudowanemu na bycie łącznikiem pomiędzy aplikacją sterowaną przez użytkownika a układem pomiarowym.

1. Implementacja
   1. Układ pomiarowy

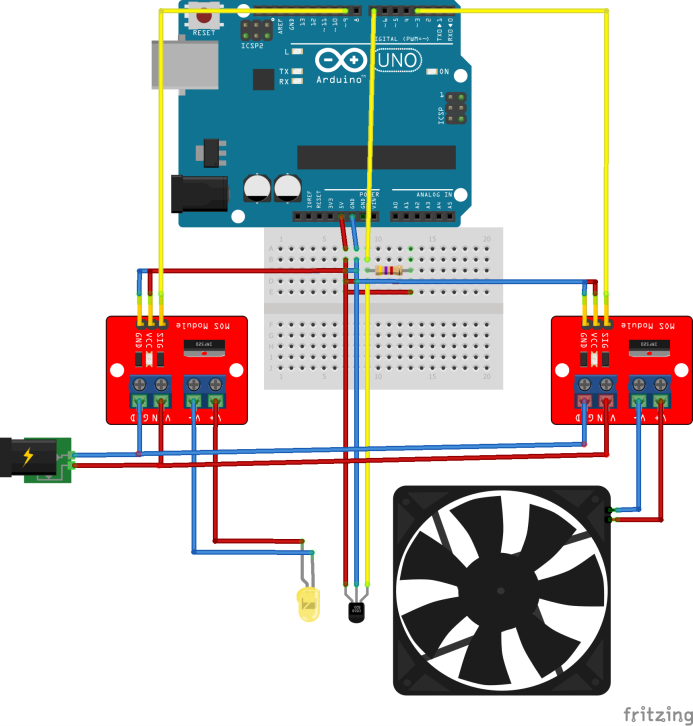
Ze względu na niewielką liczbę elementów potrzebnych do zbudowania układu, do jego stworzenia wykorzystany został mikrokontroler Arduino Uno w wersji 1. Jest to niewielka platforma z mikrokontrolerem umożliwiająca stworzenie systemu wbudowanego.

Do podłączenia wszystkich komponentów potrzebne są 2 cyfrowe wyjścia do sterowania mocą elementu grzejnego i chłodzącego oraz jedno cyfrowe wejście do odbioru danych z czujnika temperatury. Wybór Arduino Uno podyktowany jest tym, że udostępnia ono 14 cyfrowych wejść/wyjść, z czego 6 jest możliwych do wykorzystania jako kanały PWM (ang. Pulse Width Modulation) – układ potrzebuje dwóch takich kanałów by móc sterować prędkością kontrolowanych elementów.



*Rysunek 4: Arduino Uno*

Kolejną zaletą Arduino Uno jest prostota z jaką komunikuje się on z komputerem. Do wymiany informacji między użyty jest interfejs USB typu B umieszczony na płytce. Jego zadaniem jest wysłanie informacji odebranych z czujnika temperatury do komputera oraz odebranie sygnałów sterujących od użytkownika. Warty uwagi jest fakt, że interfejs USB jest magistralą szeregową – dane są wysyłane pod postacią jednego ciągu bitów, który musi dotrzeć do odbiorcy zanim następny pakiet danych zostanie wysłany. Wiąże się to z koniecznością chwilowego zatrzymania wysyłania informacji o mierzonej temperaturze w celu odebrania sygnałów sterujących z aplikacji.



*Rysunek 5: Implementacja projektu*

Interfejs USB za pomocą którego płytka jest podłączona do komputera jest jednocześnie jej złączem zasilającym, jego zastosowanie eliminuje więc konieczność użycia dodatkowego źródła zasilania podłączonego do układu.

* + 1. Element grzejny

Do symulowania wzrostu temperatury wewnątrz układu niezbędne jest zastosowanie elementu, który będzie wydzielał ciepło nagrzewające czujnik. W tym celu wykorzystana została żarówka halogenowa Decostar 35 12V o mocy 35 watów. Jest ona przymocowana na prowadnicy kulkowej umieszczonej wewnątrz tuby, pozwalając na manualną regulację odległości żarówki od czujnika temperatury. Odległość żarówki od czujnika wpływa na szybkość przyrostu notowanej temperatury – im żarówka jest dalej tym wolniej czujnik będzie notował zmianę.

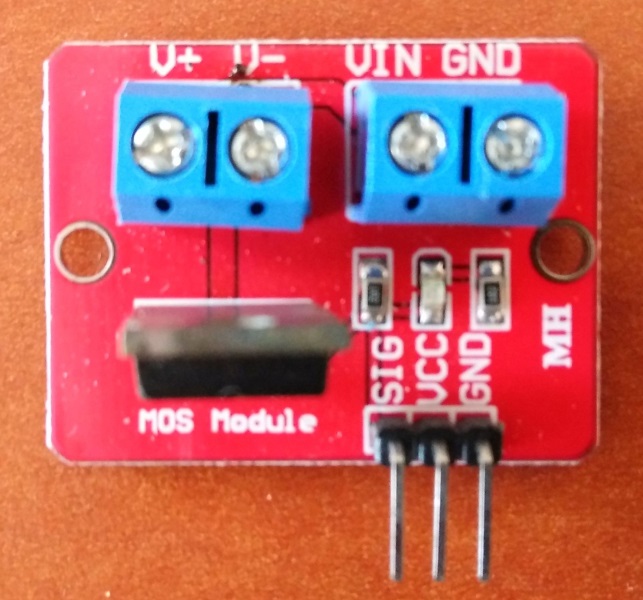
* + 1. Element chłodzący

Jako element chłodzący w układzie wykorzystany został wentylator Protechnic Magic o wymiarach 80x80mm zasilany prądem o natężeniu 12V i napięciu 0.15A. Rozmiar wentylatora jest niewielki, co umożliwiło wykorzystanie mniejszej niż początkowo planowano tuby ze szkła akrylowego, zmniejszając ogólny rozmiar układu, doprowadziło to jednak do nieoczekiwanego wcześniej problemu – gdy żarówka jest w minimalnej odległości od czujnika taki wentylator, nawet pracując z maksymalną mocą, nie jest w stanie odpowiednio schłodzić szybko nagrzewającego się czujnika aby sprowadzić jego temperaturę do punktu docelowego regulatora. Rozwiązania takiego problemu są trzy: można wykorzystać wentylator o większej mocy lub większego rozmiaru, wiązałoby się to jednak z koniecznością przebudowania całego układu, ponieważ do pełnego wykorzystania możliwości wentylatora o większej powierzchni chłodzącej należałoby wykorzystać tubę o odpowiednio większej średnicy. Innym rozwiązaniem problemu jest zwyczajne zwiększenie minimalnej odległości do czujnika w jakiej można ustawić żarówkę. Można także wykorzystać żarówkę o mniejszej mocy, prowadzi to jednak do pewnego wydłużenia czasu strojenia regulatora ze względu na dłuższy czas nagrzewania się czujnika.



*Rysunek 6: Wentylator Protechnic Magic*

Arduino Uno, do którego podłączony jest zarówno wentylator jak i żarówka halogenowa oferuje maksymalnie 5 V na złączu zasilającym, co jest wartością niewystarczającą dla obu elementów pracujących przy maksymalnym napięciu 12 V. W celu obejścia tego ograniczenia oba elementy zasilane są z zewnętrznego zasilacza 24 V do którego podłączone są za pośrednictwem modułu wykonawczego MOSFET IRF520. Jest to układ wykonawczy z tranzystorem mocy umożliwiający sterowanie urządzeniami o wyższym poborze prądu niż maksymalny dopuszczalny przez płytkę. Układ IRF520 pozwala na wykorzystanie napięcia zasilania w zakresie 0-24 V pobierając prąd do 5 A. Jest to bardzo przydatna cecha, ponieważ umożliwia ona rozbudowę układu o mocniejsze elementy grzejne i chłodzące w przypadku gdyby zaszła taka konieczność, nie jest jednak głównym powodem użycia takich modułów. Jest nim prostota z jaką można sterować mocą przekazywaną na sterowany element.



*Rysunek 7: Moduł MOSFET IRF520*

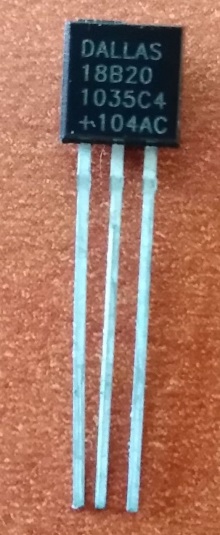
Moduł MOSFET posiada 7 wyprowadzeń, z czego 3 pary z nich są złączami zasilania – jedna odpowiada za podłączenie zasilania z modułu sterującego, druga za podłączenie zasilania z zewnętrznego źródła natomiast trzecia służy do podłączenia przewodów zasilających sterowany element. Ostatnie wyprowadzenie znajduje się pomiędzy złączami zasilania z płytki sterującej i służy ono do odbioru sygnału sterującego elementem. Zastosowanie takiego modułu pozwala zatem na zredukowanie liczby przewodów i rezystorów koniecznych do obsługi sterowania.

Moduł MOSFET IRF520 pozwolił także na rozwiązanie drobnego problemu powstałego przy zasilaniu i sterowaniu wentylatorem. Gdy ten korzystał ze zwykłego tranzystora MOSFET konieczne było podłączenie dodatkowej diody Schottky’ego między zasilaniem wentylatora a uziemieniem, która służyła do rozładowania wentylatora gdy ten powinien się wyłączyć po otrzymaniu zerowego sygnału PWM. Tak podłączona dioda pracuje jako dioda rozładowcza – wentylator zachowuje się niczym ładowana cewka i po odcięciu zasilania pracuje on dalej, zadaniem diody jest natomiast jego natychmiastowe rozładowanie. Zastosowanie modułu IRF520 pozwoliło na dalsze uproszczenie układu, ponieważ moduł posiada diodę rozładowczą w swojej konstrukcji.

Taki sam moduł został również wykorzystany do podłączenia elementu grzewczego. Nie jest on konieczny przy standardowym trybie pracy, polegającym na chłodzeniu nagrzewającego się elementu, jego wykorzystanie umożliwia jednak wprowadzenie dodatkowej koncepcji, według której układ może pracować – jeśli element grzejny stanie się elementem kontrolowanym przez układ, zaś element chłodzący będzie pracował ze stałą, maksymalną mocą to zadaniem układu przestaje być chłodzenie czujnika, by ten nie osiągał temperatur wyższych niż zadane. Jego nowym zadaniem jest nagrzanie czujnika do temperatury docelowej i utrzymanie tej temperatury na zadanym progu pomimo chłodzenia. Zmiana warunków pracy regulatora pozwala zarówno na odmienne spojrzenie na sposób kontroli temperatury oraz umożliwia przeprowadzenie doświadczenia mającego za cel wykazanie która metoda pozwala na skuteczniejsze utrzymanie temperatury na zadanym poziomie bez odchyłów w obu kierunkach.

* + 1. Czujnik temperatury

Do pomiaru temperatury użyty został czujnik temperatury DS18B20 pracujący na zasadzie interfejsu 1-wire. Jest to zarówno interfejs elektroniczy oraz protokół komunikacyjny między urządzeniami komunikującymi się między sobą korzystając z pojedyńczej szyny danych. Jedną z zalet wykorzystania takiego interfejsu jest fakt, że odbiornik (w przypadku układu przedstawionego w pracy jest to czujnik temperatury) jest zasilany za pomocą szyny danych – pozwala to na proste podłączenie takiego odbiornika bez konieczności podłączania go do zasilania dodatkowymi przewodami, sprawiając, że układ jest bardziej przejrzysty oraz prostszy w modyfikacji i utrzymaniu.



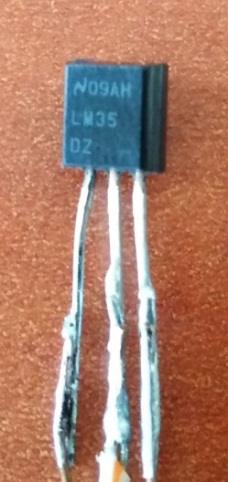
*Rysunek 8: Cyfrowy czujnik temperatury DS18B20*

Czujnik DS18B20 pracuje w zakresie od -55˚C do 125˚C z dokładnością ±0.5˚C w zakresie -10˚C do 85˚C. Zakresy te w zupełności wystarczają przy użytkowaniu układu korzystającego z żarówki halogenowej o mocy nieprzekraczającej 35 watów.

Alternatywnym czujnikiem, który mógłby zostać wykorzystany w układzie jest LM35. Pracuje on w zakresie temperatur od 0 do 100 stopni Celsjusza. W przeciwieństwie do wykorzystanego sensora DS18B20, LM35 jest czujnikiem analogowym, co oznacza, że wartość podawana na wyjściu jest wartością napięcia. LM35 jest czujnikiem liniowym o regulacji 10 mV/⁰C, co oznacza, że przy zmianie temperatury o 1⁰C napięcie wyjściowe wzrasta o 10 mV.

Sposób w jaki czujnik LM35 przekazuje temperaturę do kontrolera sprawia, że jej odczytanie jest bardzo proste: odczytaną wartość należy przemnożyć przez wartość napięcia referencyjnego (w przypadku użytego kontrolera Arduino jest to 5 V), wynik podzielić przez rozdzielczość kontrolera, na koniec zaś trzeba przeliczyć wartość z miliwoltów na wolty mnożąc wynik przez 100.

Prostota odczytania temperatury z czujnika LM35 w porównaniu do DS18B20, który wymaga użycia dostosowanej biblioteki jest dużą zaletą przy rozważaniach który czujnik powinien zostać użyty w układzie. Użycie DS18B20 jest jednak uzasadnione możliwością rozbudowy układu o kolejne czujniki umożliwiające zwiększenie dokładności pomiaru – interfejs 1-Wire pozwala na podłączenie kilku sensorów na jednej szynie danych, co nie jest możliwe dla zwyczajnych czujników analogowych. Chcąc rozbudować układ o kolejne LM35 konieczne jest zastosowanie większej ilości przewodów, które ze względu na położenie czujnika wewnątrz układu mogą zakłócać przepływ temperatury.



*Rysunek 9: Analogowy czujnik temperatury LM35*

* 1. Sterowanie układem

Sterowanie układem pomiarowym jest zaimplementowane w dwóch technologiach – zintegrowanym środowisku programistycznym Arduino oraz języku skryptowym Python. Program Arduino ma kontroluje wszystkimi elementami układu oraz wysyła i odbiera dane z aplikacji komputerowej. Sama aplikacja napisana jest w języku Python i implementuje elementy pozwalające na zdefiniowanie warunków docelowych układu, wysłanie ich do kontrolera oraz na graficzne przedstawienie temperatury układu.

* + 1. Arduino IDE

Jedną z przyczyn wyboru Arduino jako podstawy do systemu wbudowanego będącego regulatorem PID jest język w jakim pisane są funkcje wykorzystywane przez płytkę. Bazuje on na językach C i C++, ze względu więc na ich znajomość został wybrany do stworzenia oprogramowania systemu wbudowanego.

Głównym zadaniem, jakie wykonuje program jest obsługa sygnałów przychodzących z aplikacji i regulacja pracy regulatora PID. Do pełnienia swoich funkcji korzysta ona z trzech bibliotek umożliwiających jej kontakt z wykorzystywanymi urządzeniami. Biblioteki „OneWire” i „DS18B20” służą do nawiązania połączenia z czujnikiem temperatury DS18B20, natomiast biblioteka „PID\_v1” udostępnia różne funkcje umożliwiające stworzenia regulatora PID.

Kod można podzielić na trzy części: importowanie bibliotek wykorzystywanych przez program, funkcje systemowe niezbędne do działania kodu oraz funkcje użytkownika.

Zewnętrzne biblioteki służą do rozszerzenia możliwości oferowanych przez platformę Arduino bądź do wprowadzenia zupełnie nowych funkcjonalności. Kod wykorzystywany przez platformę użytą w układzie prezentowanym w pracy importuje trzy takie biblioteki: DS18B20, OneWire oraz PID\_v1. Biblioteka OneWire służy do obsługi protokołu OneWire poprzez stworzenie obiektu, za pomocą którego należy odwoływać się do czujnika temperatury z którego pobierany jest wynik pomiaru. Ponieważ interfejs OneWire umożliwia podłączenie kilku czujników temperatury wysyłających dane na tej samej magistrali, do poprawnego działania potrzebuje 64-bitowego numeru seryjnego. Biblioteka OneWire udostępnia funkcje umożliwiające odczytanie numeru seryjnego czujnika podłączonego do określonego wejścia/wyjścia płytki Arduino. Odczytany numer należy przekazać odpowiedniemu obiektowi, aby ten stworzył połączenie na interfejsie OneWire z używanym czujnikiem.

Następnie, za pomocą biblioteki DS18B20 definiuje się obiekt bezpośrednio reprezentujący sensor, który korzysta z referencji do stworzonego połączenia typu OneWire. Biblioteka DS18B20 oferuje szereg funkcji do obsługi podłączonego czujnika, z czego w kodzie obsługującym regulator wykorzystane zostały trzy: funkcja ***Begin()***, służąca do uruchomienia czujnika i zdefiniowania jego parametrów pracy takich jak rozdzielczość, funkcja ***readTemperature()***pobierająca z czujnika temperaturę oraz funkcja ***request()***, wysyłająca do czujnika polecenie wykonania pomiaru temperatury.

Do implementacji funkcjonalności regulatora PID użyta została biblioteka PID\_v1 pozwalająca stworzyć obiekt reprezentujący regulator. Do jego utworzenia potrzebna jest temperatura docelowa jaka będzie utrzymywana w układzie, obecna temperatura, adres na zmienną, która będzie przyjmowała sygnał PWM wyliczony tak, aby podana temperatura została utrzymana, wartości trzech nastaw regulatora oraz „kierunek działania” regulatora – wartość ta definiuje w jaki sposób regulator wylicza sygnał PWM wysyłany na sterowany element. Przy otrzymaniu polecenia DIRECT sygnał będzie zmniejszany gdy obserwowana wartość osiągnie docelową, polecenie REVERSE natomiast będzie ten sygnał zwiększać. Polecenia te można przyrównać do sposobu prowadzenia samochodu i chłodzenia lodówki – w przypadku prowadzenia samochodu jeśli dozwolona prędkość zostanie przekroczona należy zacząć ją zmniejszać, natomiast w przypadku lodówki jeśli temperatura zacznie przekraczać docelową należy mocniej chłodzić. Na potrzeby regulatora wykorzystany jest tryb pracy REVERSE.

Biblioteka PID udostępnia szereg funkcji do sterowania regulatorem. Do poprawnego działania wykorzystane są dwie z nich – funkcje ***SetTunings()*** oraz ***Compute()***. Pierwsza z nich otrzymuje jako parametry nowe nastawy z których powinien korzystać regulator i pozwala na ich zmianę w trakcie działania programu, natomiast druga funkcja służy do przeprowadzenia obliczeń, które prowadzą do wyznaczenia odpowiedniego sygnału PWM. Metoda obliczeń jest bardzo prosta – najpierw wyznaczane są różnice między wartością docelową i obecną oraz obecną i poprzednią wartością, które następnie wykorzystane są do wyliczenia trzech składowych sygnału docelowego. Składowa proporcjonalna jest iloczynem nastawy proporcjonalnej z różnicą między obecną i poprzednią wartością, składowa całkująca jest iloczynem nastawy całkującej z różnicą między wartością docelową a obecną, natomiast składowa różniczkująca jest wyznaczona podobnie do składowej proporcjonalnej, z wykorzystaniem nastawy różniczkującej. Trzy składowe są sumowane i w razie konieczności ograniczone do zakresu 0-255. Wyznaczony sygnał jest przypisany do zmiennej, której adres został podany w konstruktorze obiektu.

Kod, z którego ma korzystać płytka Arduino powinien posiadać dwie funkcje systemowe, pozwalające jej na sprawne działanie. Pierwszą z nich jest uruchamiana jednorazowo przy uruchomieniu płytki funkcja ***setup()***, której zadaniem jest przygotowanie platformy do działania poprzez inicjalizację portów wejścia/wyjścia oraz otwarcie portu szeregowego za pomocą którego odbywa się komunikacja płytki z komputerem. Ponadto, na potrzeby funkcjonowania regulatora PID, funkcja uruchamia jego obiekt utworzony z biblioteki PID\_v1.

Funkcja ***loop()***, uruchamiana po wykonaniu wszystkich instrukcji funkcji ***setup()***, jest główną pętlą programu, działającą nieprzerwanie do momentu wyłączenia płytki. W każdej iteracji pętli następuje wywołanie funkcji sczytującej dane z komputera, oraz zależnie od tego czy zostały odebrane nowe dane, aktualizowane są nastawy regulatora. Następnie funkcja sprawdza jaką komendę otrzymał regulator i na jej podstawie wywołuje odpowiednie funkcje kontrolera temperatury.

Zasada działania kontrolera temperatury oparta jest na prostej maszynie stanów zbudowanej na instrukcji warunkowej switch. W każdej iteracji głównej pętli sterującej Arduino program sprawdza w jakim stanie powinien pracować po czym wywołuje odpowiednią funkcję. Wybór stanu odbywa się z poziomu aplikacji komputerowej przy wykorzystaniu przycisków uruchamiających bądź wyłączających stany z nimi powiązane. Zastosowanie takiego wzorca projektowego uzasadnione jest prostotą, z jaką powinien zachowywać się układ – w danym momencie wykonuje on tylko jedną, konkretną funkcję, której zakres nie wykracza poza pojedyńczy stan. Ponadto umożliwia on w razie konieczności szybką i łatwą implementację dodatkowych stanów, bez konieczności modyfikacji kodu istniejących już istniejących.

Podstawowym stanem w jakim pracuje układ jest oczekiwanie – będąc w nim program stale wysyła do aplikacji informacje o zbieranej temperaturze oraz utrzymuje wentylator i lampę halogenową w stanie bezczynności poprzez nieustanne nadawanie zerowego sygnału PWM na oba elementy.

Drugim stanem regulatora jest stan prostej regulacji temperatury. Zasada jego działania bazuje na przełączniku dwustawnym. Pracując w nieskończonej pętli kontroluje on co 100 milisekund stan temperatury pobranej z czujnika i porównuje ją z zadaną temperaturą jaką powinien utrzymać. Sterowanie elementem chłodzącym odbywa się na zasadzie regulacji dwupołożeniowej sterowanej z poziomu aplikacji komputerowej – regulator otrzymuje pewną wartość kontrolną, która jest następnie wykorzystana do wyznaczenia symetrycznego zakresu wokół temperatury docelowej czyli histerezy. Element sterowany zostaje uruchomiony gdy temperatura przekroczy górny zakres lub wyłączony gdy spadnie poniżej dolnego. Taka mechanika pozwala na uniknięcie sytuacji gdy temperatura zaczyna oscylować wokół progu docelowego z niewielką amplitudą, powodując ciągłe włączanie i wyłączanie sterowanego elementu w minimalnych odstępach czasowych.

Na sam koniec iteracji pętli temperatura zostaje wysłana do aplikacji w celu zaktualizowania wykresu. Przerwanie tej pętli odbywa się za pomocą przycisku sterującego w aplikacji i powoduje powrót do stanu oczekiwania.

Najważniejszym stanem w jakim może pracować układ jest stan regulatora PID. Sposób jego działania jest bardzo zbliżony do stanu prostej regulacji, jednak zamiast zwykłego porównania temperatur wywołuje on na obiekcie kontrolera stworzonym w funkcji systemowej **setup()** funkcję obliczającą sygnał, jaki powinien zostać wysłany do wentylatora. Wartość sygnału jest sumą trzech składowych wyliczonych na podstawie nastaw otrzymanych z aplikacji.

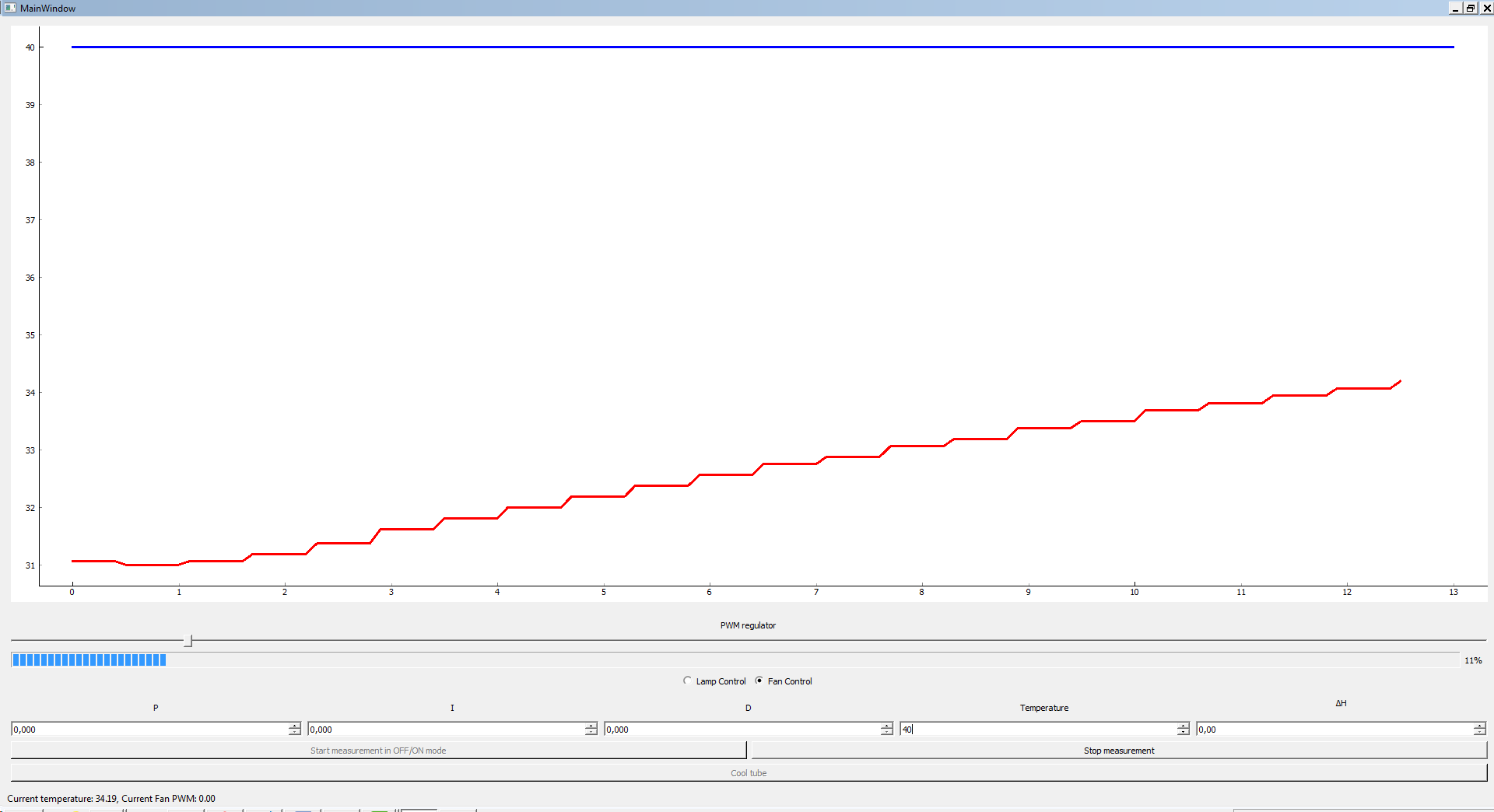
Ostatnim trybem pracy regulatora jest stan chłodzenia czujnika. Będąc w nim regulator operując w nieskończonej pętli utrzymuje lampę halogenową w bezczynności i wymusza na wentylatorze pracę z maksymalną mocą do momentu otrzymania sygnału przerwania. Tryb ten został zaimplementowany w celu przyspieszenia okresu oczekiwania pomiędzy skończeniem pomiaru dla jednego zestawu nastaw i temperatury docelowej a rozpoczęciem kolejnego, umożliwiając schłodzenie czujnika do temperatury w jakiej rozpoczynany był poprzedni pomiar.

* + 1. Python

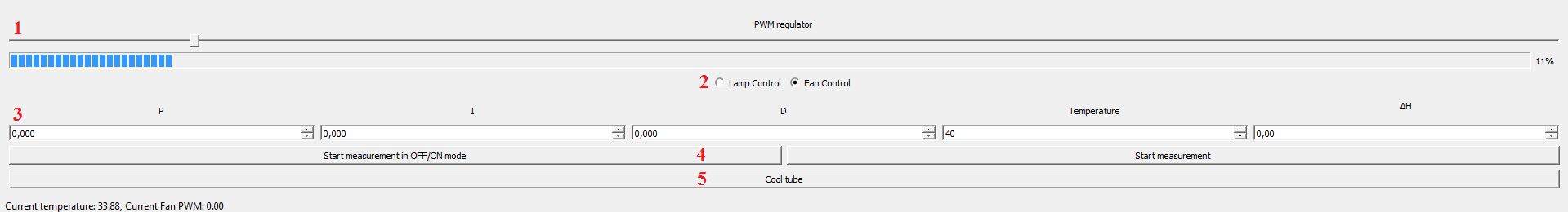
Główna aplikacja służąca do sterowania układem została napisana w języku skryptowym Python. Wybór ten podyktowany jest szerokim asortymentem różnorodnych bibliotek możliwych do zaimportowania przy pisaniu kodu, które m.in. ułatwiają komunikację z platformą Arduino bądź umożliwiają i znacząco upraszczają rysowanie wykresów na podstawie otrzymywanych danych. Biorąc pod uwagę konieczność odebrania danych z układu pomiarowego i wysyłania ich do niego, najważniejszą wykorzystaną biblioteką jest „pySerial” umożliwiająca nawiązanie połączenia z portem szeregowym obecnie używanym przez Arduino za pomocą jednej linii kodu. Biblioteka „PyQt5” służy do połączenia funkcji programu z graficznym interfejsem użytkownika GUI stworzonym w w programie QtCreator, natomiast biblioteka „PyQtGraph” jest użyta do połączenia się z polem stworzonym w oknie GUI i narysowania w nim wykresu na bazie otrzymanych danych.

Do komunikacji pomiędzy Arduino a aplikacją komputerową konieczne jest użycie biblioteki serial. W tym celu należy stworzyć obiekt połączenia szeregowego, który będzie używany przez program do odbierania i wysyłania danych. Gdy obiekt jest już utworzony pozwala on na wysyłanie danych na port za pomocą funkcji ***write()*** oraz na ich odbiór przy wykorzystaniu funkcji ***read()*** służącej do sczytania pojedynczego znaku bądź funkcji ***readline()***, czytającej całą linię. Należy pamiętać, że port szeregowy przesyła dane pod postacią pojedynczych bitów – oznacza to, że w celu wysłania bądź odebrania ciągu znaków należy je zakodować lub odkodować funkcjami ***encode()*** oraz ***decode()***. Jeżeli odbierane dane nie będą dekodowane, to zamiast oczekiwanego tekstu bądź wartości zapisane zostaną wartości ASCII odpowiadające przesyłanym symbolom. W przeciwieństwie do Arduino nie istnieje jednak konieczność odbierania każdego znaku z osobna i zapisywania go do tymczasowego buforu, który zostanie scalony z pozostałymi znakami w celu odtworzenia całej przesłanej wiadomości – funkcja ***readline()*** robi to za użytkownika, zapisując do zmiennej cały ciąg znaków zakończony symbolem końca linii.

Działanie aplikacji opiera się na oknie z interfejsem graficznym posiadającym wszystkie elementy służące do sterowania i obserwacji zachowania regulatora.



*Rysunek 10: Okno aplikacji*



*Rysunek 11: Panel kontrolny aplikacji*

Składa się ono z wykresu temperatury i panelu kontrolnego w skład którego wchodzą:

1. regulacja poziomu wyzwalania PWM z paskiem informującym o jego stanie
2. 2 pola typu radio button służące do wyboru elementu regulowanego przez układ
3. 4 spinnery służące do nadania wartości nastaw regulatora oraz docelowej temperatury
4. 2 przyciski do włączenia pomiarów w dwóch trybach – regulatora dwustawnego oraz regulatora PID
5. przycisk do schłodzenia układu

Regulacja poziomu wyzwalania PWM pozwala na wyznaczenie wartości odcięcia - gdy sygnał nadawany na sterowany element będzie mniejszy od tej wartości zostanie on zastąpiony zerem. Wymuszenie takiego zachowania na elemencie sterowanym podyktowane jest zastosowaniem wentylatora jako podstawowego elementu chłodzącego – w teorii wentylator powinien rozpocząć pracę nawet gdy dostanie najmniejszy możliwy sygnał – 1, jednak w praktyce taki sygnał jest niewystarczający do pokonania oporów wewnętrznych - wentylator pomimo dostarczanej mocy nie jest w stanie wykonać żadnej pracy, zaczyna się nagrzewać, co może prowadzić do jego uszkodzenia.

Od strony programistycznej okno jest klasą inicjalizującą wszystkie elementy interfejsu i w przypadku przycisków podłączającą funkcje mające się wykonać w momencie ich naciśnięcia. Funkcje te można podzielić w zależności od tego do jakiego typu przycisku są podłączone. Zadanie funkcji podłączonych do przycisków typu radio jest bardzo proste – zmieniają one wartość zmiennej odpowiadającej za to, którym elementem powinien sterować układ. Jest ona następnie wykorzystana w bardziej rozbudowanych funkcjach podłączonych do zwyczajnych przycisków typu dwustawna. Podstawową z trzech takich funkcji jest funkcja odpowiadająca za uruchomienie układu w trybie regulatora PID. Sczytuje ona ze wszystkich spinnerów wartości nastaw i temperatury docelowej, po czym scala je razem ze zmienną kontroli elementu i komendą rozpoczęcia pomiaru w jeden ciąg znaków i wysyła zakodowany na podłączony port szeregowy. Funkcja służąca do uruchomienia regulatora w trybie dwustawnym jest jej uproszczoną wersją – ponieważ regulacja dwustawna nie wymaga żadnego parametru poza temperaturą docelową, sczytana jest tylko ona, natomiast wysłane nastawy są ustawione na wartość zerową. Pomimo tego, że nastawy te nie są wykorzystane w tym trybie muszą one zostać wysłane ze względu na sposób w jaki funkcja służąca do odbioru danych po stronie Arduino je odczytuje i zapisuje – kolejność parametrów w ciągu znaków jest istotna i pominięcie chociaż jednego z nich może prowadzić do zapisania nieodpowiednich wartości w zmiennych używanych przez płytkę Arduino.

Trzecia funkcja, służąca do schłodzenia układu jest najprostszą ze wszystkich – odpowiada jedynie za nadanie odpowiedniej komendy sterującej, pozostałe parametry, podobnie jak w funkcji regulatora dwustawnego są wysłane jako zera.

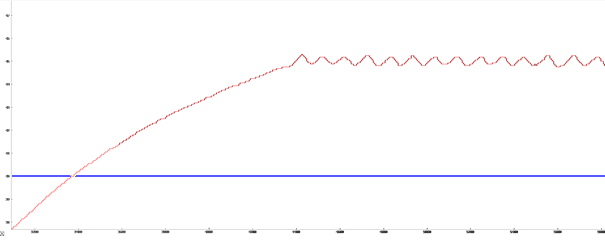
Użyta w programie biblioteka „PyQt5” oprócz możliwości wykorzystania zewnętrznego pliku do zdefiniowania interfejsu użytkownika oferuje jeszcze jedno potężne narzędzie. Są nim wątki, umożliwiające wykorzystanie wielowątkowości w programie. Jest to ważny aspekt płynności działania programu – jeden wątek jest w stanie wykonywać jedno zadanie na raz i w przypadku interfejsu użytkownika z wykresem graficznym, w trakcie pomiarów tym.zadaniem jest rysowanie wykresu. W przypadku braku wielowątkowości uruchomienie pomiarów sprawia, że aplikacja przestaje być responsywna, wątek na którym pracuje pracuje jedynie nad rysowaniem wykresu. Użycie wielowątkowości eliminuje ten problem – aplikacja tworzy główny wątek, którego zadaniem jest wysyłanie danych od użytkownika do systemu wbudowanego oraz drugi wątek poboczny, który odbiera dane z płytki Arduino i rysuje wykres. Wątki działają asynchronicznie, praca jednego nie jest więc blokowana przez pracę drugiego.

1. Pracownia studencka

Jednym z podstawowych założeń pracy jest dostosowanie układu w taki sposób, aby możliwe było jego użycie na pracowni studenckiej. Zachowanie układu zależne jest od trzech zmiennych warunków: doboru nastaw PID, odległości elementu grzejnego od czujnika oraz trybu pracy dwóch elementów sterowalnych układu. Bazując na różnych kombinacjach tych zmiennych możliwe jest stworzenie ćwiczeń wymagających od studenta np. dobrania odpowiednich parametrów i trybu pracy celem jak najszybszego ustabilizowania temperatury na zadanym poziomie bądź porównania różnych trybów pracy w stałych warunkach.

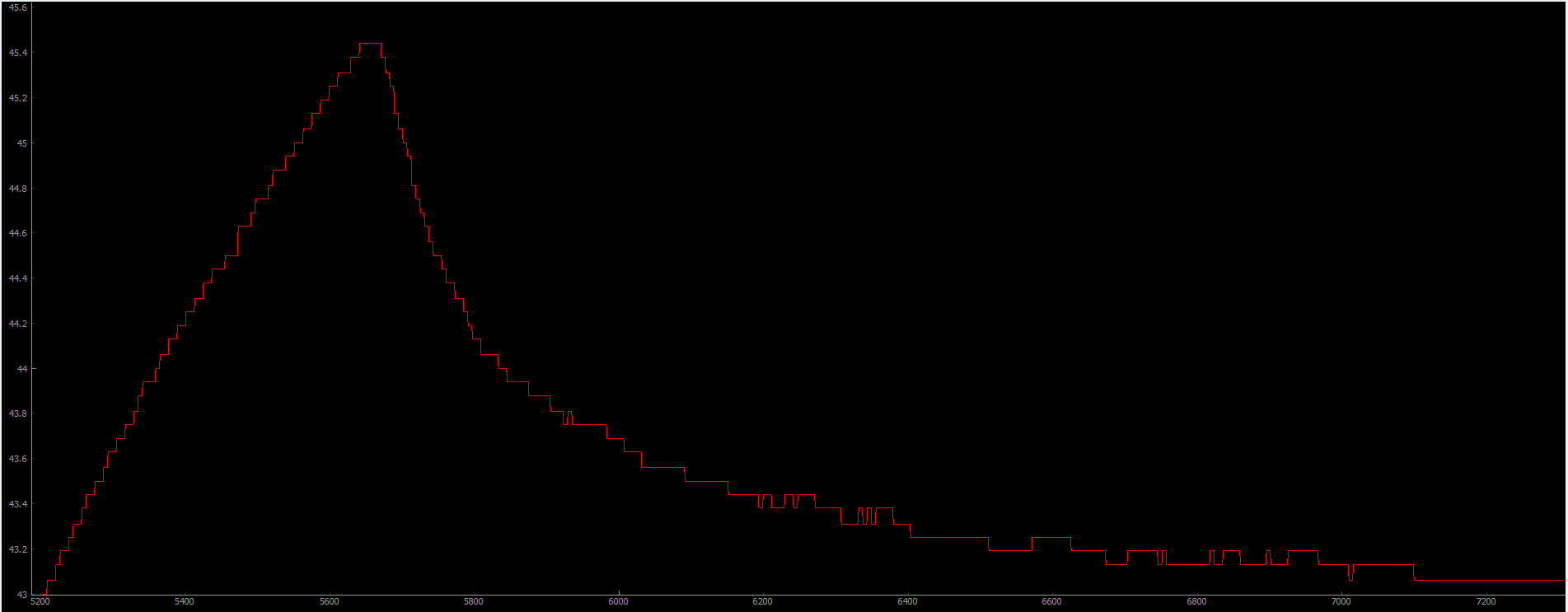
* 1. Dobór nastaw
     1. Strojenie ręczne

Metoda strojenia ręcznego polega na doświadczalnym dostosowaniu nastaw bazując na zachowaniu kontrolera w danej chwili. W pierwszym kroku resetuje się wszystkie trzy nastawy. Następnie należy stopniowo zwiększać składową proporcjonalną (nastawę P) do momentu, w którym temperatura badanego układu nie wejdzie w stan regularnych oscylacji o zbliżonej sobie amplitudzie.



*Rysunek 12: Ustabilizowane oscylacje przy zwiększaniu nastawy P*

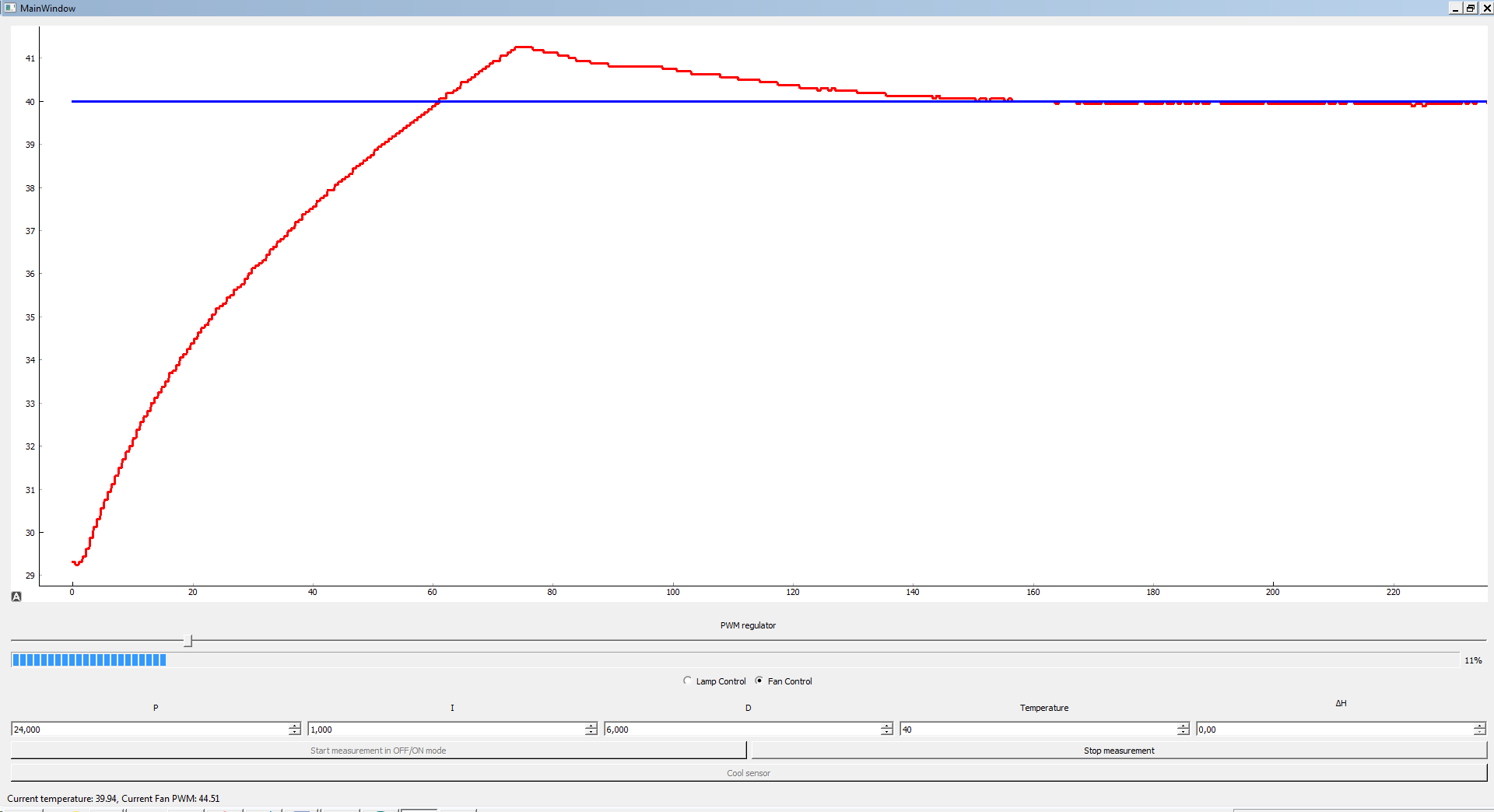
Gdy zachowanie układu będzie zadowalające rozpoczyna się powolne zwiększanie składowej różniczkującej (nastawę D). Sprawi to, że oscylacje układu zaczną być stopniowo tłumione. Zwiększanie nastawy D kontynuuje się tak długo, aż oscylacje nie zostaną całkowicie wygaszone. Przy tak dobranych nastawach P i D wykres temperatury badanego układu powinien zachować podobną charakterystykę do przedstawionego:



*Rysunek 13: Ustabilizowany układ z dużym przestrzałem temperatury*

Sposób w jaki zachowuje się układ przy takich parametrach jest zbliżony do końcowego efektu jaki należy osiągnąć, jednak czas w jakim układ będzie reagował na zmianę temperatury niekoniecznie jest satysfakcjonujący i może prowadzić do tzw. Przestrzału, czyli znaczącego przekroczenia temperatury ponadzadany próg. Aby go skrócić można powtarzać drugi i trzeci krok strojenia regulatora – najpierw następuje zwiększenie składowej proporcjonalnej do uzyskania stabilnych oscylacji, następnie zwiększa się składową różniczkującą do momentu ich wytłumienia. Takie strojenie jest możliwe do momentu gdy zwiększanie nastawy D przestanie tłumić oscylacje.

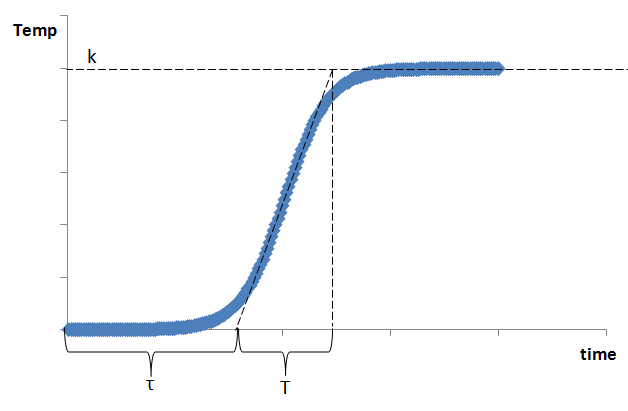
Ostatnim krokiem potrzebnym do nastrojenia układu jest uregulowanie składowej całkującej, czyli nastawy I. Zwiększa się ją tak długo jak błąd między temperaturą utrzymywaną przez układ a temperaturą docelową nie będzie równy zeru.



*Rysunek 14: Dostrojony układ z minimalnym przestrzałem*

* + 1. Pierwsza metoda Zieglera-Nicholsa

Pierwsza metoda Zieglera-Nicholsa polega na graficznym wyznaczeniu parametrów odpowiedzi skokowej układu na sygnał wejściowy. Charakterystyka przyrostu temperatury posiada dwa parametry potrzebne do wyznaczenia nastaw regulujących układ: opóźnienie czasowe *τ*, będące opóźnieniem pomiędzy otrzymaniem sygnału wejściowego a rozpoczęciem wzrostu temperatury oraz stałą czasową *T* wyjątkową dla każdego układu, definiującą czas przyrostu temperatury do momentu jej ustabilizowania na stałym poziomie. W celu graficznego wyznaczenia tych parametrów tworzy się wykres odpowiedzi skokowej i prowadzi dwie proste – pierwsza z nich jest styczną do ustabilizowanej temperatury i prostopadła do osi czasowej, druga jest styczna z punktem przegięcia funkcji odpowiedzi skokowej, czyli prowadzi się ją wzdłuż nachylenia funkcji. Następnie wyznacza się parametry *τ* i *T* według poniższego rysunku:

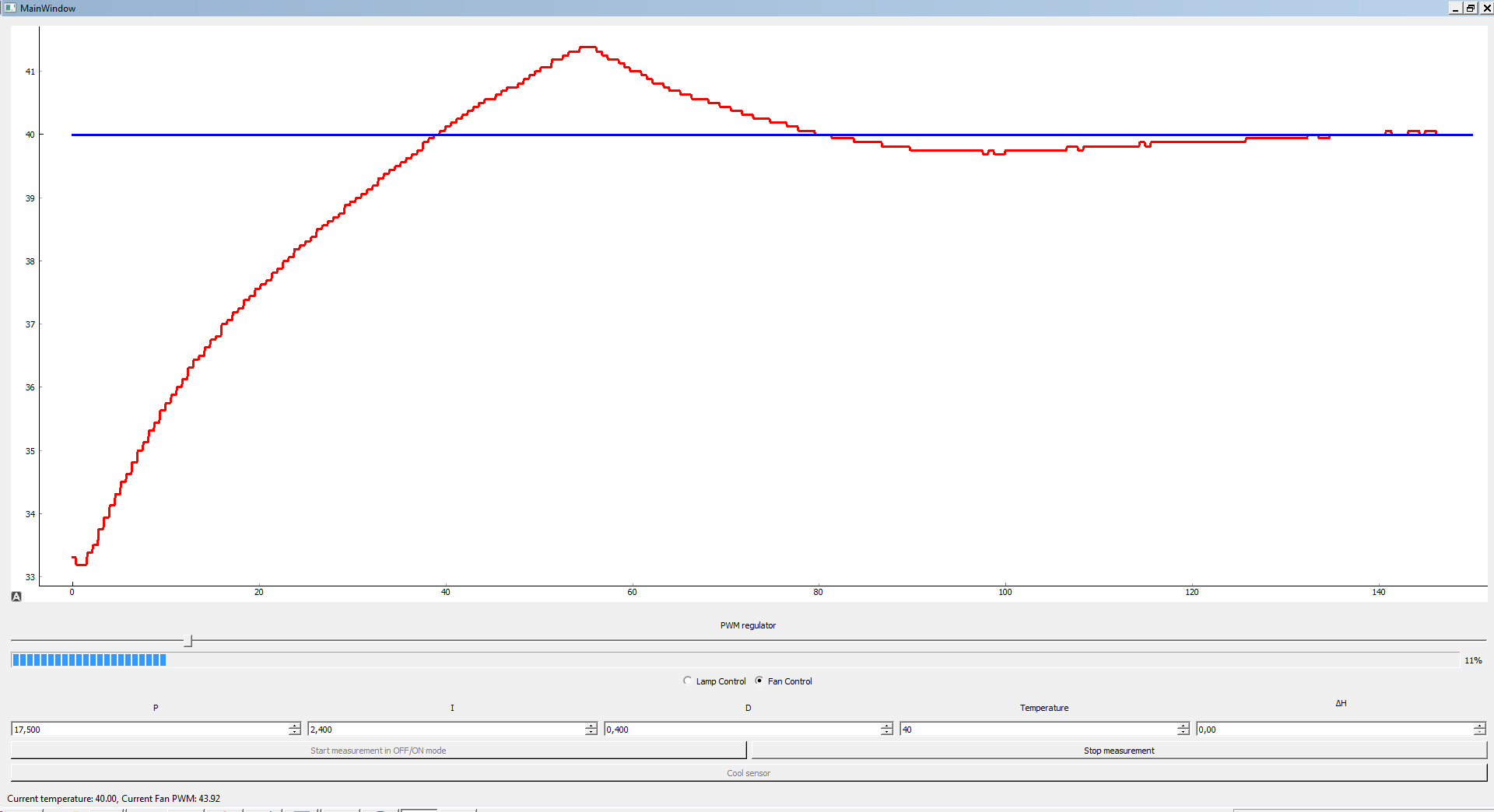


*Rysunek 15: Przykładowa charakterystyka odpowiedzi skokowej*

Po otrzymaniu parametrów wylicza się nastawy regulatora według poniższej tabeli:

*Tabela 1: Wartości nastaw według pierwszej metody Zieglera-Nicholsa*

|  |  |  |
| --- | --- | --- |
| **P** | **I** | **D** |
| 0.35 | 2.4*τ* | 0.4*τ* |



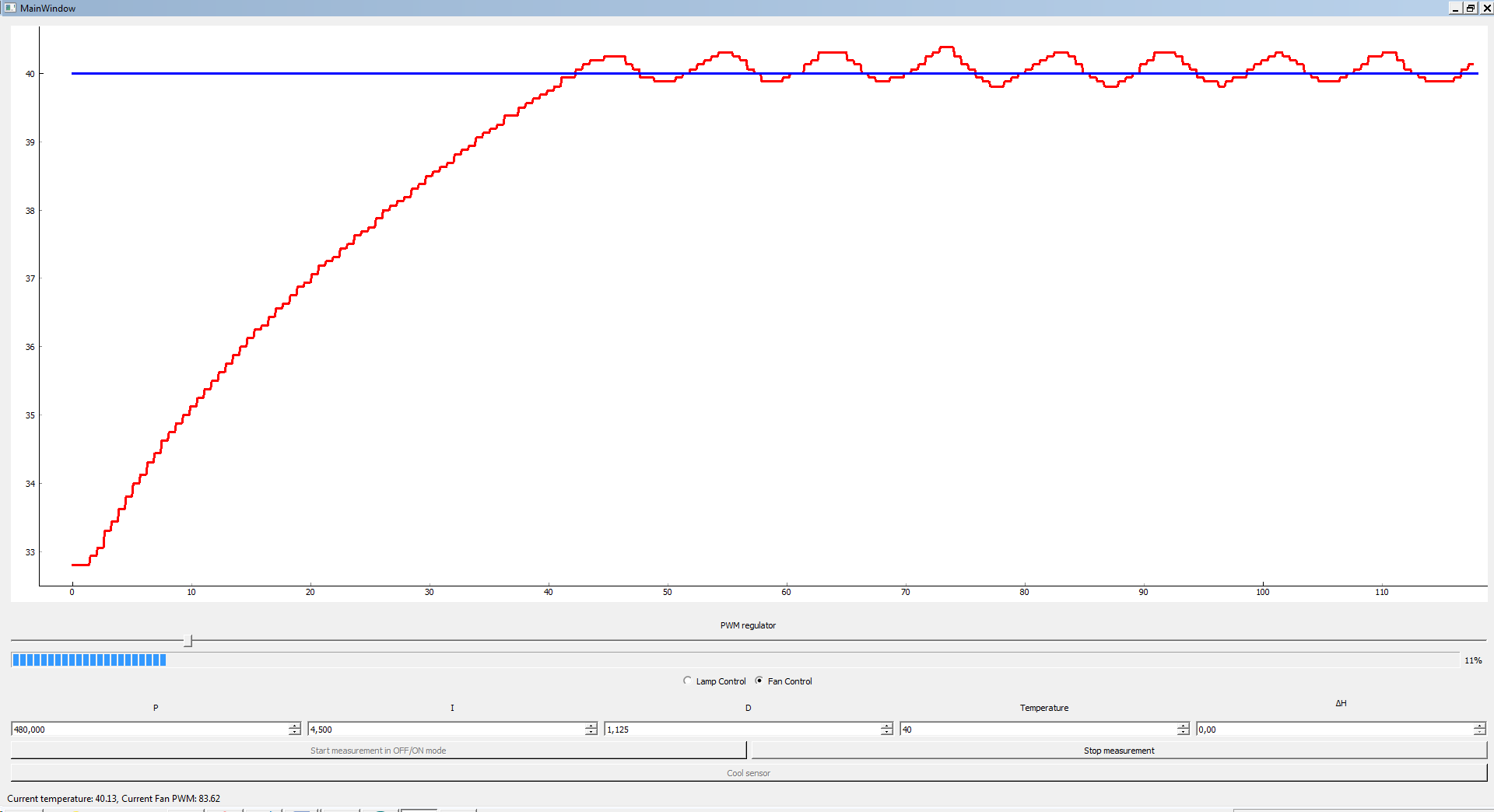
*Rysunek 16: Układ dostrojony pierwszą metodą Zieglera-Nicholsa*

* + 1. Druga metoda Zieglera-Nicholsa

Druga metoda Zieglera-Nicholsa jest metodą heurystyczną polegającą na matematycznym wyznaczeniu nastaw I i D na podstawie okresu oscylacji układu. W celu ich wyznaczenia należy wszystkim nastawom nadać wartość zerową, po czym zacząć zwiększać nastawę proporcjonalną do najwyższej możliwej wartości przy której występować będą stabilne, niegasnące oscylacje. Następnie odczytuje się okres oscylacji *Pu* oraz zapisuje się wartość składowej proporcjonalnej jako wzmocnienie krytyczne *Ku*, za pomocą których wyznacza się docelowe wartości nastaw według poniższej tabeli:

*Tabela 2: Wartości nastaw według drugiej metody Zieglera-Nicholsa*

|  |  |  |
| --- | --- | --- |
| **P** | **I** | **D** |
| 0.6*Ku* | *Pu*/2 | *Pu*/8 |



*Rysunek 17: Układ dostrojony drugą metodą Zieglera-Nicholsa*

* 1. Przykładowe ćwiczenie

Dzięki możliwości pracy układu w dwóch różnych trybach regulatora, które mogą kontrolować pracę zarówno elementu grzewczego jak i chłodzącego oraz 3 różnym metodom doboru nastaw regulatora PID, istnieje wiele różnych konfiguracji ćwiczeń, jakie mogłyby zostać przeprowadzonena układzie. Zadanie jakie otrzymałby student mogłoby wygladać następująco:

* Dla temperatury docelowej 40℃, odległości lampy od czujnika 7 cm i sterowania wentylatorem należy uruchomić układ w trybie dwustawnym.
* Stopniowo zwiększać zakres histerezy do momentu gdy oscylacje wokół temperatury docelowej nie będą przekraczać 1℃.
* Przełączyć element sterowany na lampę halogenową i powtórzyć pomiar.
* Za pomocą manualnej metody doboru nastaw nalezy dostroić układ, aby przy takich samych parametrach wstępnych jak w poprzednim pomiarze układ ustabilizował się na zadanym poziomie i przestrzał temperatury nie przekroczył 1℃.
* Porównać czas stabilizacji układu w różnych trybach pracy i ocenić, który z nich i dlaczego najlepiej nadaje się kontroli układu w zadanych warunkach

1. Testy układu

Testowanie układu polegało na sprawdzeniu jego zachowania w dostępnych trybach pracy przy wykorzystaniu przedstawionych metod doboru nastaw.

W pierwszej kolejności przeprowadzone zostały testy regulatora dwustawnego kontrolującego prędkość wentylatora przy stałym nagrzewaniu komory układu przez lampę halogenową oddaloną od czujnika temperatury o 7 cm. Układ uruchomiono z temperaturą docelową 40℃ i zakresem histerezy równym 0. Zgodnie z oczekiwaniami, wentylator był uruchamiany natychmiast gdy aktualna temperatura wewnątrz układu przekraczała temperaturę docelową i wyłączany gdy komora została schłodzona, wprowadzając układ w stan oscylacji o amplitudzie 0.5℃ i częstotliwości 1 sekundy. Następnie powtórzono testy dla ustawień histerezy równej 2 i 4. Dla takich ustawień układ uruchamiał wentylator gdy temperatura była większa o połowę histerezy od temperatury docelowej i wyłączał gdy była mniejsza o tą samą wartość. Układ pracował więc z oczekiwaną charakterystyką regulatora dwustawnego, próbując utrzymać temperaturę wewnątrz zakresu definiowanego przez histerezę.

Następnie dla takich samych warunków początkowych, czyli temperatury docelowej i odległości lampy od czujnika zostały przeprowadzone testy strojenia regulatora PID, których wyniki zostały przedstawione na rysunkach 14, 16 i 17 oraz w poniższej tabeli:

*Tabela 3: Wyniki testów doboru nastaw*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Czas stabilizacji [s]** | **Maksymalny uchyb**  **[**℃**]** | **P** | **I** | **D** |
| Metoda manualna | 80 | 1.42 | 24 | 1 | 6 |
| Pierwsza metoda Z-N | 85 | 1.38 | 17.5 | 2.4 | 0.4 |
| Druga metoda Z-N | - | 0.25 | 480 | 4.5 | 1.125 |

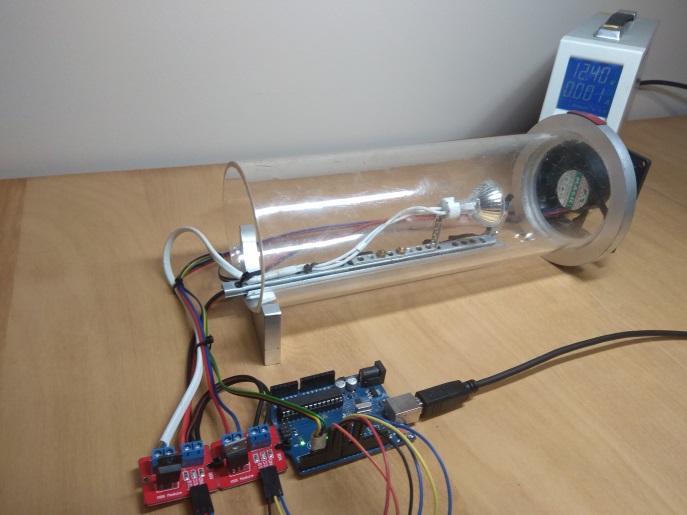
Uzyskane wyniki pokazują, że możliwe jest nastrojenie układu za pomocą wszystkich trzech przedstawionych metod – czasy stabilizacji układu od momentu przekroczenia temperatury docelowej są do siebie zbliżone z wyjątkiem strojenia drugą metodą Zieglera-Nicholsa – układ nastrojony w ten sposób nigdy nie osiągnął pełnej stabilizacji, wszedł natomiast w stan stałych oscylacji o amplitudzie 0.25℃ i okresie 10 sekund.

Wartości docelowe czasu stabilizacji układu i maksymalnych uchybów jakie wystąpią podczas regulacji są definiowane przez użytkownika – to użytkownik ukladu stwierdza, czy uzyskane warunki spełniają założenia postawione układowi pracującemu z ustalonymi warunkami początkowymi. Przeprowadzone testy nie mają zatem na celu przedstawienia jak dokładnie układ powinien się zachowywać, gdyż to zależy wyłącznie od oczekiwań użytkownika. Celem testów jest natomiast wykazanie, że układ jest w stanie pracować według postawionych mu założeń.

1. Podsumowanie

Głównym założeniem przedstawionej pracy było stworzenie układu pomiarowego do regulacji temperatury oraz aplikacji służącej do sterowania układem danym układem, który stanowiłby makietę dydaktyczną dostępną dla studentów, umożliwiającą poznanie sposobu działania dwóch różnych regulatorów temperatury i metod ich strojenia.

Stworzony układ składa się z 3 głównych komponentów zarządzanych przez system wbudowany komunikujący się z aplikacją komputerową, za pomocą której steruje się parametrami pracy układu. Trzy komponenty układu, czyli czujnik temperatury, lampa halogenowa będąca element grzewczy i wentylator będący elementem chłodzącym umieszczone są w tubie ze szkła akrylowego pełniącej funkcję komory pomiarowej układu. Zarowno element grzewczy jak i chłodzący sterowane są za pośrednictwem modułu wykonawczego MOSFET IRF520, umożliwiając sterowanie ich mocą. Jako, że jeden element jest elementem sterowalnym, drugi natomiast jest źródłem sygnału wejściowego, układ może pracować w dwóch różnych trybach zdefiniowanych przez to, którym elementem układ steruje – gdy kontrolowany jest wentylator komora nagrzewa się od stałej pracy lampy halogenowej i regulator sterując prędkością wentylatora chłodzi ją do temperatury docelowej. Natomiast gdy kontrolowana jest lampa, wentylator stale schładza komorę, a układ regulując mocą lampy ogrzewa ją do zadanego poziomu.



*Rysunek 18: Układ pomiarowy*

Sterowanie układem zrealizowane jest za pomocą systemu wbudowanego opartego na platformie sprzętowej Arduino Uno. Pozwala ona na podłączenie wszystkich wykorzystanych elementów do jednej płytki i sterowanie każdym z nich z osobna za pomocą poszczególnych kanałów wejścia/wyjścia przez które urządzenia są podłączane. Kod wgrany na wykorzystaną płytkę umożliwia pracę systemu jako regulator dwustawny oraz regulator PID – wszystkie obliczenia niezbędne do kontroli i regulacji temperatury wykonywane są przez mikrokontroler, w który płytka jest wyposażona .

Arduino Uno jest połączone z komputerem za pośrednictwem interfejsu USB typu B za pomocą którego płytka jest zasilana oraz komunikuje się z aplikacją komputerową. Sama aplikacja napisana jest w języku skryptowym Python i składa się z interfejsu graficznego zaprojektowanego w programie QT Designer, posiadającego wykres, na którym rysowana jest temperatura układu oraz kontrolki pozwalające użytkownikowi na dostosowanie parametrów takich jak: sterowany element, tryb pracy układu, temperatura docelowa, nastawy regulatora PID czy zakres histerezy dla regulatora dwustawnego. Sama aplikacja korzysta z dostępnych dla języka Python bibliotek pozwalających jej na komunikację z płytką Arduino oraz na proste rysowanie wykresów.

Końcowy projekt spełnia wszystkie cele, jakie zostały mu postawione - układ jest w pełni sprawny i gotowy do użytkowania przez studentów chętnych do poszerzenia wiedzy o informacje dotyczące regulatorów temperatury i ch strojenia.

Płytka Arduino Uno i wszystkie przewody,za pomocą których podłączone są elementy układu nie są zamknięte w żadnej obudowie. Istnieje zatem możliwość rozbudowania układu o dedykowaną obudowę wydrukowaną w technologi druku 3D, pozwalającą na zamknięcie elektroniki układu w środku, zabezpieczając ją przed niepożądanymi modyfikacjami. Same elementy czyli czujnik temperatury, lampa halogenowa i wentylator można łatwo wymienić w razie ich uszkodzenia lub w trakcie modyfikacji układu.

Zarówno aplikacja do kontroli układu jak i oprogramowanie płytki Arduino posiada strukturę funkcyjną, co oznacza, że poszczególne funkcje tworzące program nie komunikują się bezpośrednio ze sobą, co skutkuje możliwością rozbudowy programu o kolejne funkcjonalności. Przykładowo możnaby zamontować drugi czujnik temperatury, np. Wspomniany w podrozdziale 3.1.3 czujnik analogowy LM35, pozwalając na porównanie dokładności pomiaru pomiędzy dwoma czujnikami.

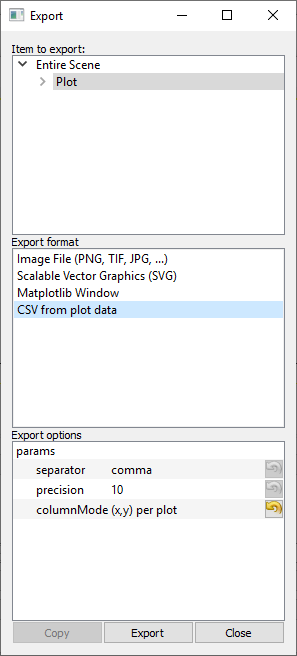
Bibliografia

* https://automatykab2b.pl/technika/46618-strojenie-pid-metody-doboru-nastaw-czesc-1, dostępne: 22 luty 2019
* https://playground.arduino.cc/Code/PIDLibrary , dostępne: 22 luty 2019
* http://akademia.nettigo.pl/ds18b20/, dostępne: 22 luty 2019
* https://github.com/milesburton/Arduino-Temperature-Control-Library, dostępne: 22 luty 2019
* http://www.asimo.pl/teoria/regulatory\_dzialanie\_ciagle.php, dostępne: 22 luty 2019
* http://pdf1.alldatasheet.com/datasheet-pdf/view/250767/VISHAY/IRF520.html, dostępne: 22 luty 2019

Dodatki

Eksport danych wykresu –metody Zieglera Nicholsa

W celu poprawnego wyznaczenia nastaw korzystając z metod Zieglera Nicholsa konieczna jest możliwość skorzystania z danych, by móc wyznaczyć oscylację układu bądź odczytać punkty przecięć stycznych. Aplikacja umożliwia wyeksportowanie danych do pliku CSV, które nastepnie można wykorzystać np. w programie Excel do stworzenia wykresu wyłącznie z niezbędnego zakresu danych. W tym celu należy na wykres kliknąć prawym przyciskiem myszy i z rozwiniętego menu wybraź opcję „export”. Otwarte zostanie następujące okno:



*Rysunek 19: Okno eksportu danych*

Następnie z okienka „Export format” wybrać opcję „CSV from plot data” i nacisnąć przycisk export. Dane zapisane w pliku nie nadają się jednak do natychmiastowego użycia w programie Excel, ponieważ wszystkie są zapisane w jednej kolumnie. W celu ich podzielenia konieczne jest zaznaczenie całej kolumny i wybranie opcji „Tekst jako kolumny” z zakładki „dane”. Otwarty zostanie kreator konwersji danych, w którm jako typ danych należy wybrać „Rozdzielany” a jako ogranicznik danych przecinek. Ostatnim krokiem jest podmiana wszystkich kropek w liczbach zmiennoprzecinkowych na przecinki, by Excel mógł poprawnie odczytać dane. By to zrobić należy na karcie „Narzędzia główne” z grupy „Edytowanie” kliknąć przycisk „Znajdź i zaznacz” i wybrać opcję „Zamień”. Otwarte okno pozwoli na zamianę wszystkich kropek na przecinki.