

POLITECHNIKA KRAKOWSKA
IM. TADEUSZA KOŚCIUSZKI

WYDZIAŁ INŻYNIERII ELEKTRYCZNEJ I KOMPUTEROWEJ
KIERUNEK INFORMATYKA

MICHAŁ PATYK

**Sterownik pieca kominkowego, w oparciu o
mikrokontroler (lub platformę
komputerową), zgodny ze szkicem
specyfikacji Web Thing API**

(praca inżynierska)

Promotor: dr inż. Radosław Czarnecki

Kraków 2020

Spis treści

| | |
|---|-----------|
| 1 Wstęp | 5 |
| 1.1 Rys historyczny - ujęcie problemowe | 5 |
| 1.1.1 Problem ogrzewania | 5 |
| 1.1.2 Integracja urządzeń | 6 |
| 1.2 Stan aktualnej wiedzy | 6 |
| 1.2.1 Sterowniki | 6 |
| 1.2.2 Internet rzeczy i WWW rzeczy | 7 |
| 1.3 Innowacyjność projektu na tle współczesnych rozwiązań | 7 |
| 1.4 Motywacja | 8 |
| 2 Cele pracy, zakres pracy, założenia | 9 |
| 2.1 Cele pracy | 9 |
| 2.2 Zakres pracy | 9 |
| 2.3 Założenia i wymagania | 10 |
| 2.3.1 Wykorzystane narzędzia | 10 |
| 2.3.2 Możliwości | 10 |
| 2.3.3 Wymagania | 10 |
| 2.4 Efekt końcowy | 11 |
| 2.5 Dalsze kierunki rozwoju | 11 |
| 2.5.1 Integracja z innymi urządzeniami | 11 |
| 2.5.2 Zmiany w oprogramowaniu sterownika | 12 |
| 2.5.3 Zmiany sprzętowe | 12 |
| 3 Charakterystyka metod obsługi pieca | 13 |
| 4 Przegląd istniejących rozwiązań | 14 |
| 4.1 Standardy integracji inteligentnych urządzeń | 14 |
| 4.2 Sterowniki kominkowe | 14 |
| 4.2.1 TECH STEROWNIKI | 15 |

| | |
|---|-----------|
| SPIS TREŚCI | 2 |
| 4.2.2 EUROSTER | 15 |
| 4.2.3 Kratki | 15 |
| 4.2.4 Podsumowanie | 15 |
| 5 Opracowanie koncepcji budowy sterownika pieca kominkowego | 16 |
| 6 Wybór podzespołów sterownika pieca kominkowego | 17 |
| 6.1 Mikrokontroler lub platforma komputerowa | 17 |
| 6.2 Wyświetlacz | 18 |
| 6.3 Termometr | 18 |
| 6.4 Termopara | 18 |
| 6.5 Układ poruszający przepustnicą | 18 |
| 6.6 Moduł Ethernet | 18 |
| 6.7 Zegar czasu rzeczywistego | 19 |
| 6.8 Urządzenie wskazujące | 19 |
| 6.9 Czujnik otwarcia drzwi paleniska | 20 |
| 6.10 Zasilanie awaryjne | 20 |
| 6.11 Źródło dźwięku | 20 |
| 6.12 Rodzaj połączenia czujników zewnętrznych | 20 |
| 7 Wykonanie prototypu sterownika pieca kominkowego na płytce stykowej | 21 |
| 7.1 Połączenie mikrokontrolera oraz LCD | 21 |
| 7.2 Podłączenie termometru i czujnika wilgotności DHT22 | 22 |
| 7.3 Podłączenie termometru Dallas | 22 |
| 7.4 Podłączenie układu MAX6675 - termopara | 22 |
| 7.5 Podłączenie serwomechanizmu | 23 |
| 7.6 Podłączenie czujnika otwarcia paleniska | 23 |
| 7.7 Podłączenie układu Ethernet W5500 Light | 23 |
| 7.8 Podłączenie brzęczyka | 23 |
| 7.9 Wykonanie okablowania | 23 |
| 8 Opracowanie oprogramowania - część pierwsza | 24 |
| 8.1 Utworzenie projektu | 24 |
| 8.2 Edycja pliku konfiguracyjnego projektu | 24 |
| 8.3 Szkielet do budowy aplikacji | 25 |
| 8.4 Dodanie bibliotek i podstawowa komunikacja z urządzeniami | 25 |
| 8.4.1 Dodanie portalu przechwytyjącego | 25 |
| 8.4.2 Dodanie biblioteki dla DHT22 | 26 |

| | | |
|-----------|--|-----------|
| 8.4.3 | Dodanie biblioteki dla MAX6675 | 26 |
| 8.4.4 | Dodanie biblioteki dla LCD | 26 |
| 8.4.5 | Dodanie biblioteki dla serwomechanizmu | 26 |
| 8.4.6 | Dodanie biblioteki dla PID | 26 |
| 8.4.7 | Dodanie biblioteki dla DS18B20 | 27 |
| 8.4.8 | Dodanie biblioteki dla W5500 Light | 27 |
| 8.4.9 | Dodanie biblioteki WebThings | 27 |
| 8.5 | Dodanie możliwości zdalnego uaktualniania oprogramowania | 27 |
| 8.6 | Dodanie odczytu napięć | 28 |
| 8.7 | Dodanie wykrywania dotyku | 28 |
| 8.8 | Dodanie obsługi brzeczyka | 28 |
| 8.9 | Testowanie sterownika | 28 |
| 9 | Wykonanie prototypu sterownika pieca kominkowego na płytce uniwersalnej | 29 |
| 9.1 | Rozmieszczenie elementów | 29 |
| 9.2 | Wykonanie panelu dotykowego | 30 |
| 9.3 | Połączenie elementów | 30 |
| 9.4 | Zmiany w projekcie | 31 |
| 9.4.1 | Dodanie dzielnika napięcia | 31 |
| 9.4.2 | Dodanie głównegołącznika zasilania | 31 |
| 9.5 | Testowanie prototypu | 32 |
| 10 | Wykonanie obudowy sterownika pieca kominkowego | 33 |
| 11 | Opracowanie oprogramowania - część druga | 35 |
| 11.1 | Rozbudowa portalu przechwytyującego | 35 |
| 11.2 | Rozbudowa obsługi termometru i czujnika wilgotności DHT22 | 35 |
| 11.3 | Rozbudowa obsługi termopary z modulem MAX6675 | 36 |
| 11.4 | Rozbudowa obsługi wyświetlacza LCD | 36 |
| 11.5 | Rozbudowa obsługi serwomechanizmu | 36 |
| 11.6 | Rozbudowa obsługi regulatora PID | 36 |
| 11.7 | Rozbudowa obsługi termometru Dallas DS18B20 | 37 |
| 11.8 | Rozbudowa obsługi modułu Ethernet W5500 | 37 |
| 11.9 | Rozbudowa obsługi odczytu napięć | 37 |
| 11.10 | Rozbudowa obsługi wykrywania dotyku | 37 |
| 11.11 | Rozbudowa obsługi brzeczyka | 37 |
| 11.12 | Rozbudowa obsługi WebThings | 38 |
| 11.12.1 | Napotkane problemy | 38 |
| 11.13 | Testowanie sterownika | 39 |

| | |
|--|-----------|
| 12 Zintegrowanie sterownika pieca kominkowego z Mozilla Gateway | 40 |
| 12.1 Przygotowanie lokalnej Bramy WebThings | 40 |
| 12.2 Dodanie rzeczy webowych do Bramy WebThings | 40 |
| 12.3 Wykorzystywanie interfejsu Bramy WebThings | 41 |
| 12.4 Tłumaczenie interfejsu Bramy WebThings | 41 |
| 12.5 Testowanie sterownika | 42 |
| 13 Efekt końcowy na tle koncepcji | 43 |
| 14 Podsumowanie | 45 |
| 14.1 Przegląd istniejących rozwiązań | 45 |
| 14.2 Opracowanie koncepcji budowy sterownika pieca kominkowego | 45 |
| 14.3 Wybór podzespołów sterownika pieca kominkowego | 46 |
| 14.4 Wykonanie prototypu sterownika pieca kominkowego na płytce stykowej | 46 |
| 14.5 Opracowanie oprogramowania - część pierwsza | 46 |
| 14.6 Wykonanie prototypu sterownika pieca kominkowego na płytce uniwersalnej | 46 |
| 14.7 Wykonanie obudowy sterownika pieca kominkowego | 47 |
| 14.8 Opracowanie oprogramowania - część druga | 47 |
| 14.9 Zintegrowanie sterownika pieca kominkowego z Mozilla Ga- teway | 47 |
| 15 Dalszy kierunek prac | 48 |
| Książki | 49 |
| Artykuły | 50 |
| Prace dyplomowe | 51 |
| Materiały konferencyjne | 52 |
| Pozostałe źródła | 53 |

Rozdział 1

Wstęp

1.1 Rys historyczny - ujęcie problemowe

1.1.1 Problem ogrzewania

Problem ogrzewania pomieszczeń towarzyszy człowiekowi od zarania dziejów [21]. Zwykle był to proces wymagający ciągłego nadzoru. Potrzeba automatyzacji wydaje się naturalną konsekwencją zmiany trybu życia człowieka. Wraz z pojawieniem się nowoczesnych kotłów, powstały pierwsze mechanizmy kontrolujące pracę urządzenia bez nieustannej konieczności doglądania go. Pojawienie się sterowników cyfrowych zaoferowało zupełnie nowe możliwości, takie jak:

- większą efektywność pracy systemu grzewczego
- zmniejszenie emitowanych zanieczyszczeń
- wzrost komfortu użytkowania
- poprawę bezpieczeństwa
- zdalne sterowanie (w nowszych modelach)

Sterowniki cyfrowe wydają się również odpowiedzią na coraz bardziej dostrzegany problem odpowiedzialnego wykorzystywania zasobów naturalnych [32][2], gdyż za ich pomocą do zapewnienia komfortu termicznego pomieszczeń potrzebna jest znacznie mniejsza ilość paliwa. Stanowią one również propozycję rozwiązania problemu ubóstwa energetycznego.

1.1.2 Integracja urządzeń

Sukces internetu spowodował zainteresowanie naukowców podłączeniem do sieci fizycznych urządzeń. Termin internet rzeczy powstał w 1999 roku wymyślony przez Kevina Ashtona [1] jednak jego popularność rozpoczęła się dopiero kilka lat temu, a w tej chwili przechodzi okres stagnacji. Możemy wyróżnić wiele przypadków zastosowania internetu rzeczy:

- bezprzewodowe sieci czujników i pomiary rozproszone,
- urządzenia ubieralne i urządzenia do pomiaru parametrów zdrowia,
- inteligentne domy i budynki,
- inteligentne miasta i sieci energetyczne,
- inteligentna produkcja przemysłowa,
- inteligentna logistyka i łańcuchy dostaw.

Przyjęte przez niewielkie grupy korporacji standardy internetu rzeczy podlegają ciągłej ewolucji spowodowanej rozwojem technologicznym, a zamknięte protokoły prowadzą do sytuacji, w której zmiana dostawcy rozwiązań jest mocno utrudniona i może powodować konieczność zmiany sprzętu. Standardy webowe, takie jak HTTP, JSON i Web Socket, zawdzięczają swoją popularność otwartości oraz brakowi opłat. Ich zastosowanie do integracji urządzeń i aplikacji w znacznym stopniu może ułatwić wchodzenie w interakcje z rzeczami.

1.2 Stan aktualnej wiedzy

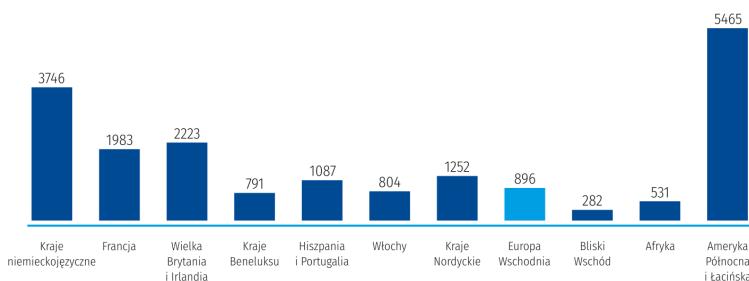
1.2.1 Sterowniki

Obecnie na rynku dostępny jest szeroki wachlarz rozwiązań, od prostych jednokomponentowych do bardziej zaawansowanych i złożonych. Począwszy od regulatorów pokojowych, które jedynie włączają ogrzewanie kiedy temperatura pomieszczenia obniży się poniżej nastawionej wartości, poprzez rozbudowane, umożliwiające programowanie temperatury zarówno w ciągu doby (niższa w nocy, wyższa po południu), jak i w wybrane dni tygodnia. Skończywszy na automatyce pogodowej, która dzięki wykorzystaniu czujnika zewnętrznego, umieszczonego na ścianie domu, przewiduje zwiększone

zapotrzebowanie na ciepło i wcześniej dostosowuje moc kotła. Coraz więcej dostępnych na rynku sterowników umożliwia zdalne nastawienie temperatury. W znakomitej większości standard komunikacji tych rozwiązań jest zamknięty, co utrudnia współpracę z innymi inteligentnymi urządzeniami.

1.2.2 Internet rzeczy i WWW rzeczy

W Europie Wschodniej występuje wciąż mała na tle konkurencji liczba firm oferujących rozwiązania IoT. Wzmożony rozwój tematu WWW rzeczy

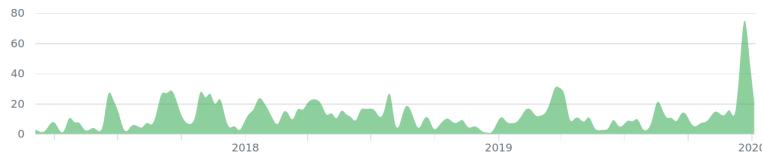


Rys. 1.1: Liczba firm oferujących rozwiązania IoT na poszczególnych rynkach. (źródło: [23])

zapoczątkowany został przez pracę doktorską Dominique'a Guinarda [16], a pogłębiło go wydanie książki [18] podsumowującej dotychczasowe prace autora w tym temacie [17] [15] [20] [16] [19]. W 2014 roku przy World Wide Web Consortium, grupie zajmującej się ustanawianiem standardów pisania i przesyłu stron WWW, powstała Web of Things Interest Group [29] która pracuje nad dokumentami opisującymi Rzeczy Webowe [47] oraz Architekturę [46]. W 2017 roku do projektowania Web of Things przyłącza się fundacja Mozilla [3] tworząc i rozbudowując projekt WebThings Gateway. Przyciągnął on liczną społeczność znacząco rozwijającą przedsięwzięcie. W następnych latach przybywa kolejnych prac naukowych na temat Web of Things [38] [41] [25].

1.3 Innowacyjność projektu na tle współczesnych rozwiązań

W odróżnieniu od szeroko stosowanych rozwiązań projekt zakłada użycie specyfikacji Web Thing API [48], która pozwala na ujednolicenie dostępu



Rys. 1.2: Liczba przesłanych fragmentów kodu przez społeczność WebThings Gateway. (źródło: [4])

do inteligentnych rzeczy za pomocą dodatkowej warstwy abstrakcji. Oznacza to, że system kreuje przestrzeń kooperacji urządzeń. Możliwe interakcje wpływają pozytywnie na koherencję systemu.

1.4 Motywacja

Pomysł na pracę zrodził się z potrzeby zmniejszenia ilości czasu oraz poświęcanej uwagi potrzebnych do obsługi pieca kominkowego, pracującego jako główne źródło ciepła w domu jednorodzinnym. Ponadto niekomfortowym ograniczeniem dotychczas stosowanych rozwiązań jest zmuszanie klienta do użytkowania produktów pochodzących od tego samego producenta, co w znaczącym stopniu ogranicza możliwość wyboru preferowanego sprzętu.

Rozdział 2

Cele pracy, zakres pracy, założenia

2.1 Cele pracy

Celem niniejszej pracy jest **opracowanie koncepcji sterownika pieca kominkowego** zgodnego ze szkicem specyfikacji Web Thing API oraz stworzenie **dedykowanego oprogramowania**.

2.2 Zakres pracy

Zakres pracy obejmuje:

1. przegląd istniejących rozwiązań - zarówno sprzętowych jak i programowych;
2. opracowanie koncepcji;
3. wybór podzespołów;
4. wykonanie prototypu na płytce stykowej;
5. stworzenie oprogramowania;
6. przetestowanie oprogramowania;
7. wykonanie prototypu na płytce uniwersalnej;
8. wykonanie obudowy;
9. zintegrowanie z Mozilla Gateway

2.3 Założenia i wymagania

Projekt sterownika zostanie wykonany zgodnie z dobrymi praktykami dotyczącymi programowania zawartymi między innymi w pracach Grębosza [14] i Francuza [13]. Część sprzętowa projektu zostanie opracowana na podstawie książek Monka [33] [34] [35], Kurczyka [28], Huanga [22] i Wallace'a [45] oraz norm Polskiego Komitetu Normalizacyjnego [36] [37].

2.3.1 Wykorzystane narzędzia

- środowisko programistyczne CLion
- język programowania C/C++
- ekosystem PlatformIO
- platforma monitoringu i kontroli urządzeń WebThing Mozilla

2.3.2 Możliwości

- monitorowanie pracy pieca kominkowego
- zdalne zadawanie temperatur
- informowanie o zdarzeniach

2.3.3 Wymagania

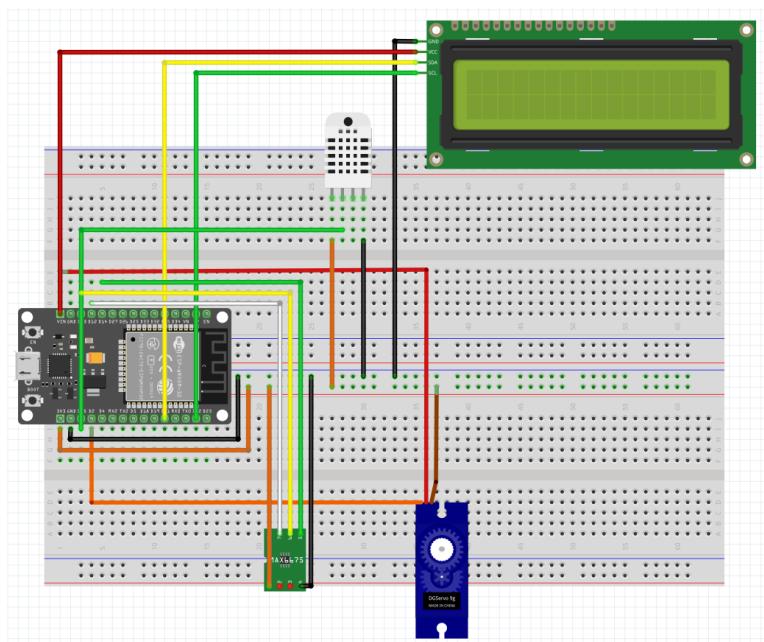
- zgodność z Web Thing API
- możliwość rozbudowy o dodatkowe czujniki i elementy wykonawcze
- przywracanie nastaw po utracie zasilania
- minimalizacja otwarcia przepustnicy w przypadku zaniku napięcia
- sygnalizacja uszkodzenia czujnika temperatury
- regulowana jasność wyświetlacza - zwiększana na czas zmiany ustawień
- sygnalizator dźwiękowy który informuje gdy temperatura wzrośnie do niebezpiecznego poziomu

2.4 Efekt końcowy

Planowanym efektem końcowym pracy będzie stworzenie sterownika pieca kominkowego, pozwalającego na bezobsługową pracę paleniska pomiędzy momentami uzupełniania paliwa.

Element wyróżniający wykonany sterownik stanowi wykorzystanie Web Thing REST API, który pozwala na wykorzystanie sieci, jako zunifikowanej warstwy abstrakcji, dla zdecentralizowanego internetu rzeczy.

Rysunek 2.1 przedstawia wizję struktury elementów sterownika.



Rys. 2.1: Wizja elementów sterownika. (opracowanie własne)

2.5 Dalsze kierunki rozwoju

W tej części zostaną opisane możliwe kierunki rozwoju sterownika.

2.5.1 Integracja z innymi urządzeniami

Wykorzystanie WebThing API umożliwi w przyszłości łatwą integrację sterownika z dodatkowymi urządzeniami, takimi jak:

- czujnik pogodowy
- systemem odzyskiwania ciepła ze spalin
- inne źródła ciepła

2.5.2 Zmiany w oprogramowaniu sterownika

Oprogramowanie sterownika może zostać rozbudowane o przykładowe komponenty:

- dodatkowe opcje konfiguracyjne
- programowanie czasowe
- zabezpieczenie hasłem
- dodatkowe tryby pracy pieca

Możliwą do wprowadzenia zmianą może być też zastąpienie frameworka Arduino, bardziej profesjonalnym Espressif IoT Development Framework, który lepiej współpracuje z mikrokontrolerem ESP32.

2.5.3 Zmiany sprzętowe

Proponowanymi zmianami w obrębie sprzętowym są:

- zaprojektowanie i wykonanie płytki drukowanej
- zaprojektowanie i wykonanie dedykowanej obudowy, np.: w technologii druku 3D

Rozdział 3

Charakterystyka metod obsługi pieca

Obsługa w pełni manualna wymaga stałego nadzoru nad przebiegiem spalania. Jest to bardzo absorbowujące i często nieekonomiczne ze względu na łatwość przekroczenia pożądanych temperatur. Obsługa manualna pogarsza również efektywność procesu spalania, co prowadzi do obniżenia ogólnej sprawności kotła, większego zapotrzebowania na opał, a w konsekwencji znacznie wyższych kosztów ogrzewania.

Obsługa półautomatyczna pozwala na samodzielną pracę urządzenia pomiędzy momentami dokładania paliwa. Można wyróżnić dwie jej odsłony:

- mechaniczna, w której głowica termostatyczna połączona jest z mechanizmem dźwigni. Na jej ramieniu znajduje się łańcuszek lub linka. Zaletą jest niski koszt;
- elektroniczna, gdzie stosuje się sterownik pokojowy z serwomechanizmem. Koszt tego rozwiązania jest umiarkowany oraz zapewnia szerokie możliwości regulacji, umożliwiając dokonywanie nastaw lokalnie lub zdalnie.

Najdroższe jak i najbardziej autonomiczne rozwiązanie stanowi obsługa automatyczna, w której kocioł z podajnikiem paliwa pracuje samodzielnie do kilku dni.

Rozdział 4

Przegląd istniejących rozwiązań

4.1 Standardy integracji inteligentnych urządzeń

W przypadku internetu rzeczy producenci nie zwracali uwagi na stworzenie otwartych systemów komunikujących się ze sobą urządzeń. Szczególnie ograniczone były działania podejmowane dla umożliwienia doraźnego współdziałania elementów składających się na internet rzeczy. Pomimo, że organizacje standaryzacyjne zaproponowały wiele protokołów, żaden z tych standardów nie zdobył na tyle dużej popularności by zdominować pozostałe rozwiązania. Coraz więcej urządzeń podłączanych do internetu udostępnia API, jednak każdy z nich został zaimplementowany z wykorzystaniem innych protokołów i używa innego modelu danych. Zamiast proponować kompletnie nowe normy komunikacyjne, Web of Things wykorzystuje istniejące, dobrze znane standardy webowe.

4.2 Sterowniki kominkowe

Na polskim rynku istnieje szeroka gama rozwiązań przeznaczonych dla kotłów na paliwo stałe, jednak stosunkowo niewielką ilość sterowników przeznaczono dla kominków. Istnieją wyłącznie modele z dodatkową przepustnią, co ogranicza możliwość regulacji dopływu powietrza pierwotnego i wtórnego.

4.2.1 TECH STEROWNIKI

Firma TECH ma w swojej aktualnej ofercie [44] trzy sterowniki do kominków. Dwa z nich są do kominków z płaszczyzną wodną. Trzeci [43] przeznaczony jest do kominków z dystrybucją ciepłego powietrza i mógłby zostać zastosowany przy piecu kominkowym. Czujnik mierzy temperaturę spalin w kominie. Aby mieć możliwość zdalnego dostępu do sterownika należy skorzystać z jednego z dwóch dodatkowych modułów komunikacyjnych - GSM lub Ethernet. Dostęp jest możliwy jedynie przez stronę producenta lub aplikację mobilną, brak publicznego API. Sam moduł Ethernet kosztuje około 500zł [42].

4.2.2 EUROSTER

Firma EUROSTER ma w swojej aktualnej ofercie [12] tylko jeden sterownik do kominków. Jest on [11] przeznaczony do współpracy z kominkiem z płaszczyzną wodną, więc nie może być zastosowany do pieca kominkowego, gdyż czujnik mierzy temperaturę wody obiegowej, która nie występuje w piecu kominkowym. Sterownik nie posiada możliwości zdalnego dostępu. Cena samego sterownika to około 300zł [10]. Dodatkowo należy zakupić przepustnicę.

4.2.3 Kratki

Firma Kratki ma w swoje ofercie [27] jeden sterownik do kominków występujący w kilku wersjach różniących się obudową i wielkością dołączoną do zestawu przepustnicy. Jest on przeznaczony do współpracy z kominkami każdego rodzaju, dlatego może zostać zastosowany do pieca kominkowego. Czujnik mierzy temperaturę w pobliżu paleniska. Sterownik nie posiada możliwości zdalnego dostępu. Cena samego sterownika to około 400zł [26].

4.2.4 Podsumowanie

Tylko jeden z wymienionych producentów pozwala na zdalny dostęp do sterownika, jednak jest on wyłącznie możliwy za pośrednictwem serwerów producenta. Brak publicznego API. Koszt tego rozwiązania jest wysoki. Tylko jeden z wymienionych sterowników posiada czujnik pozwalający na mierzenie temperatury spalin.

Rozdział 5

Opracowanie koncepcji budowy sterownika pieca kominkowego

Projektowanie koncepcji sterownika pieca kominkowego należy rozpocząć od skompletowania minimalnej wersji, na którą składa się: mikrokontroler, serwomechanizm oraz termopara. Do regulacji pracy pieca kominkowego jest potrzebne urządzenie potrafiące zmieniać położenie przepustnicy dolotu powietrza do komory spalania. Najbardziej optymalnym wyborem okazał się prosty serwomechanizm, którego dodatkową zaletą jest przystępna cena. Stopień otwarcia przepustnicy dobierany jest na podstawie temperatury spalin w kominie, mierzonej przez termoparę. Dodatkowymi elementami uzupełniającymi koncepcję są:

- termometr, mierzący temperaturę w pomieszczeniu, umożliwiający dobrą mocą pieca w stosunku do zapotrzebowania na ciepło
- wyświetlacz przekazujący informacje użytkownikowi
- czujnik otwarcia drzwi paleniska umożliwiający zmianę parametry pracy po dołożeniu drewna
- zdalny dostęp do sterownika w celu ułatwienia obsługi
- moduł Ethernet pozwalający na usunięcie niepotrzebnego ruchu WiFi
- zapasowe źródło zasilania, pozwalające na kontrolę pracy pieca w razie zaniku napięcia w sieci
- przyciski na sterowniku do zmiany podstawowych parametrów

Rozdział 6

Wybór podzespołów sterownika pieca kominkowego

6.1 Mikrokontroler lub platforma komputerowa

Jako serce sterownika można wykorzystać:

1. platformy komputerowe:

- RaspberryPi
- OrangePi

2. mikrokontrolery:

- Seria Arduino
- STM32F103C8T6
- ESP8266
- ESP32

Ze względu na koszt jak i rozdzielenie odpowiedzialności w pracy wybrano mikrokontrolera ESP32. Jest to układ System on Chip będący następcą ESP8266. Posiada on dwa rdzenie, moduł komunikacji WiFi oraz bluetooth. Wyróżnia go energooszczędność. Dużą zaletą jest stosunkowo niska cena (w Chinach około 25zł). Gotowa płytka deweloperska pozwala na wygodne prototypowanie.

6.2 Wyświetlacz

Aby sterownik sprostał potrzebom użytkowników o pogorszonym wzroku, wyświetlacz powinien być duży i kontrastowy. Ze względu na to, że wszystkie parametry będą dostępne przez stronę internetową, wybrany został wyświetlacz LCD 2x16, który pozwoli na prezentowanie tylko kilku z nich w jednej chwili.

6.3 Termometr

Wybrany układ termometru DHT22 pozwala na pomiar temperatury i wilgotności, jednocześnie zapewniając dobrą precyzję odczytów. Dodatkowo, sterownik rozszerzony został o moduł termometru Dallas DS18B20, który pozwala na pomiar temperatury wewnętrz urządzenia.

6.4 Termopara

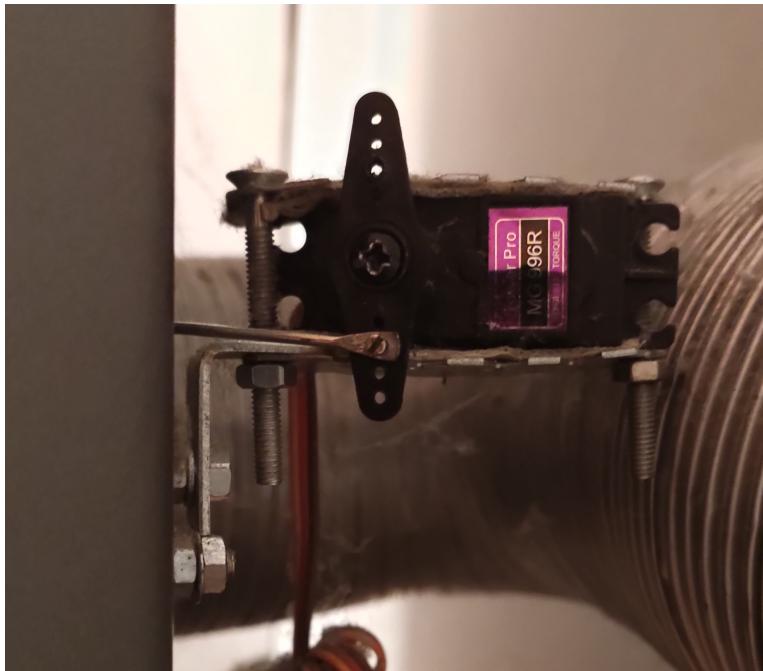
W pracy wybrano układ MAX6675, który jest jednym z najlepiej przetestowanych i najtańszych rozwiązań, ze względu na swoją długą obecność na rynku.

6.5 Układ poruszający przepustnicą

Posiadany piec kominkowy, który jest wzorcowy dla sterownika, potrzebuje przemieszczania liniowego przepustnicy, jednak odpowiednie serwomechanizmy są wciąż drogie i przewyższałyby swoją ceną koszt pozostałych podzespołów całego sterownika. Istnieje możliwość zaprojektowania i wydrukowania właściwego adaptera serwomechanizmu, lecz długi czas produkcji oraz testowania jego wytrzymałości przekraczałby zakres tej pracy. Ze względu na niewielką wymaganą precyzję ustawienia przepustnicy oraz zaistniałe opory, wybrano standardowej wielkości, analogowy serwomechanizm MG-995, bez adapterów przekształcających ruch obrotowy na liniowy. Zdjęcie 6.1 przedstawia serwomechanizm zainstalowany przy piecu kominkowym.

6.6 Moduł Ethernet

Zdecydowano się na dołączenie dodatkowego modułu Ethernet, ponieważ sterownik bez przerwy komunikuje się z siecią w celu wymiany informacji.



Rys. 6.1: Serwomechanizm zainstalowany przy piecu kominkowym. (opracowanie własne)

Wykorzystanie komunikacji WiFi generowałoby niepotrzebny smog elektromagnetyczny. Wybrany moduł W5500 Lite jest nieznacznie droższy, ale za to bardziej kompaktowy niż inne rozwiązania oparte o układ W5500.

6.7 Zegar czasu rzeczywistego

Pierwotnie planowano wykorzystać moduł czasu rzeczywistego z podtrzymaniem. Jednakże, sterownik przez większość czasu będzie podłączony do internetu, skąd może pozyskać aktualną godzinę, dlatego zrezygnowano z tego rozwiązania.

6.8 Urządzenie wskazujące

Pierwszym planowanym rozwiązaniem było wykorzystanie przycisków typu pushbutton. Mikrokontroler ESP32 posiada jednak specjalne piny umożliwiające rozpoznanie dotyku poprzez detekcję zmian pojemności, dlatego

zdecydowano się zastąpić pushbuttony panelem dotykowym.

6.9 Czujnik otwarcia drzwi paleniska

Czujnik otwarcia drzwi paleniska pozwala na wykrywanie momentów dokładania paliwa. W projekcie wykorzystano prosty przełącznik podłączony do pinu cyfrowego mikrokontrolera.

6.10 Zasilanie awaryjne

Najprostszym z rozważanych rozwiązań było zastosowanie dodatkowego, zewnętrznego modułu typu powerbank. Jednak jego wykorzystanie nie dawałoby żadnej kontroli nad trybem zasilania sterownika, co mogłoby skutkować nagłym przerwaniem dostaw energii, a w konsekwencji możliwością przegrzania pieca. Wybrano płytę rozszerzeń typu battery shield z dwoma akumulatorami litowo-jonowymi 18650 o pojemności około 3600mAh każdy oraz układem ładowania i układem zabezpieczającym przed całkowitym rozładowaniem.

6.11 Źródło dźwięku

Jako źródło dźwięku w projekcie wykorzystano brzęczyk pasywny. W odróżnieniu od aktywnego, umożliwia on generowanie dźwięków o zmiennej częstotliwości za pomocą sygnału PWM. Dla ułatwienia konstrukcji zdecydowano się wybrać moduł brzęczyka zawierający tranzystor sterujący, który chroni źródło sygnału przed przeciążeniem prądowym.

6.12 Rodzaj połączenia czujników zewnętrznych

Ze względu na łatwość i pewność połączenia zdecydowano się na wybranie standardu złącza 8P8C, wykorzystywanego w różnego rodzaju sprzęcie telekomunikacyjnym i komputerowym. Jest ono rozpowszechnione jako złącze do budowy sieci w standardzie Ethernet. 8P8C to złącze o ośmiu miejscach na styk i ośmiu stykach.

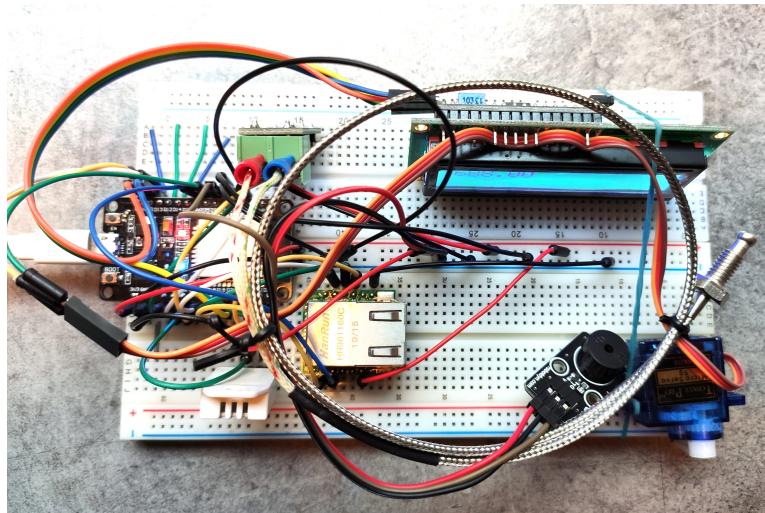
Rozdział 7

Wykonanie prototypu sterownika pieca kominkowego na płytce stykowej

W tym rozdziale opisany został proces prototypowania z wykorzystaniem płytki stykowej. Pozwala ona na łatwe połączenie elementów tworzonego układu. Podzespoły wpinane są w płytke, a następnie łączone elektrycznie przy pomocy specjalnych przewodów. Pierwszym elementem sterownika umieszczonym na płytce był mikrokontroler ESP32. Nie wymagał on dodatkowych połączeń, ponieważ może być zasilany przez port USB. Zdjęcie 7.1 przedstawia płytke stykową z podłączonymi modułami sterownika pieca kominkowego.

7.1 Połączenie mikrokontrolera oraz LCD

Użyty w projekcie wyświetlacz LCD 16x2 posiada dodatkowo konwerter I2C, który zmniejsza ilość linii niezbędnych do komunikacji z 16 do 4. Wykorzystana do komunikacji magistrala I2C jest szeregowa oraz dwukierunkowa. Wykorzystuje tylko dwie linie komunikacyjne - Serial Data i Serial Clock. Transfer danych może być zainicjowany tylko, gdy magistrala nie jest zajęta. Poza liniami transferu danych wyświetlacz potrzebuje także pary linii zasilania - masy oraz 5V.



Rys. 7.1: Płytki stykowe z podłączonymi modułami. (opracowanie własne)

7.2 Podłączenie termometru i czujnika wilgotności DHT22

Do komunikacji z mikrokontrolerem moduł termometru i czujnika wilgotności DHT22 używa magistrali 1-Wire, która wykorzystuje pojedynczą linię komunikacyjną. W zależności od długości przewodów należy wybrać właściwy rezystor podciągający. Mikrokontroler ESP32 posiada wbudowany opornik podciągający o wartości około 50k Ohmów. Jeśli przewody przyłączeniowe nie są długie, wtedy nie ma potrzeby wykorzystywania dodatkowego rezystora. Moduł termometru DHT22 potrzebuje również zasilania 3,3V.

7.3 Podłączenie termometru Dallas

Układ termometru Dallas DS18B20, do komunikacji z mikrokontrolerem, wykorzystuje magistrale 1-Wire opisaną w części 7.2. Dodatkowo potrzebuje zasilania 3,3V.

7.4 Podłączenie układu MAX6675 - termopara

Układ MAX6675 wykorzystuje do komunikacji magistralę SPI, posiadającą trzy linie komunikacyjne: SO - Master Input Slave Output, CS - Chip

Select, SCK - Serial Clock. Do zasilania układu potrzebne jest zasilanie 3,3V.

7.5 Podłączenie serwomechanizmu

Do podłączenia serwomechanizmu wystarczy jedna linia komunikacyjna przesyłająca sygnał PWM oraz zasilanie 5V.

7.6 Podłączenie czujnika otwarcia paleniska

Czujnik otwarcia drzwi paleniska, w postaci prostego przełącznika, został podłączony do pinu cyfrowego mikrokontrolera pojedynczą linią komunikacyjną. Linia ta będzie ustawiana w stan wysoki przy pomocy rezystora podciągającego. Otwarcie drzwi spowoduje zwarcie linii do masy.

7.7 Podłączenie układu Ethernet W5500 Light

Moduł W5500 wykorzystuje do komunikacji magistralę SPI, posiadającą 4 linie komunikacyjne: MO - Master Output Slave Input, SO - Master Input Slave Output ,CS - Chip Select, SCK - Serial Clock. Podczas podłączania układu napotkano problemy z komunikacją. Konieczna była zmiana wykorzystywanych pinów mikrokontrolera.

7.8 Podłączenie brzęczyka

Do podłączenia brzęczyka potrzebna jest jedna linia komunikacyjna przesyłająca sygnał PWM oraz zasilanie 5V. Podczas podłączania natknęto się na błędy komunikacyjne. Przyczyną okazało się błędne nazewnictwo pinów. Pin 32 został zamieniony z pinem 33 w opisie wyprowadzeń na płytce deweloperskiej mikrokontrolera ESP32.

7.9 Wykonanie okablowania

Podłączenie zewnętrznych czujników do sterownika wykonałem z wykorzystaniem skrętki ekranowanej, która zapewnia odporność komunikacji na zakłócenia transmisji.

Rozdział 8

Opracowanie oprogramowania - część pierwsza

W tym rozdziale opisano proces opracowywania pierwszej części oprogramowania, stanowiącej podstawę komunikacji z modułami sterownika. Dalsza część procesu programowania została opisana w rozdziale 11.

8.1 Utworzenie projektu

Do programowania sterownika wykorzystano środowisko programistyczne CLion wraz z ekosystemem PlatformIO. CLion to wieloplatformowe zintegrowane środowisko programistyczne (IDE) przeznaczone dla języków C/C++. PlatformIO Core jest sercem całego ekosystemu PlatformIO ułatwiającego programowanie mikrokontrolerów. Dostarcza interfejs wiersza poleceń. PlatformIO Core napisana jest w języku Python. Projekt oprogramowania sterownika został zainicjowany w pustym katalogu poleciением:

```
platformio init --ide clion --board esp32doit-devkit-v1
```

Definiuje ono z jakiego IDE chcemy skorzystać oraz jaki mikrokontroler będzie programowany. Następnie projekt został otwarty w CLion, gdzie przebiega dalszy proces programowania.

8.2 Edycja pliku konfiguracyjnego projektu

W celu zmiany konfiguracji projektu należy poddać edycji plik platformio.ini. Daje on przede wszystkim możliwość wybrania framework'a, dodatkowych bibliotek, prędkości komunikacji z mikrokontrolerem przez monitor

portu szeregowego oraz parametrów wgrywania oprogramowania do mikrokontrolera.

8.3 Szkielet do budowy aplikacji

Spośród dwóch dostępnych dla mikrokontrolera ESP32 frameworków, ze względu na dużą ilość gotowych bibliotek do komunikacji z modułami, zdecydowano się na wykorzystanie platformy programistycznej Arduino. Program napisany dla tej platformy musi posiadać plik main, w którym są dwie funkcje: setup oraz loop. Setup wykonywany jest tylko raz po uruchomieniu mikrokontrolera, natomiast loop wykonuje się w nieskończonej pętli aż do momentu zakończenia pracy.

8.4 Dodanie bibliotek i podstawowa komunikacja z urządzeniami

Pisanie oprogramowania rozpoczęto od dodania bibliotek pozwalających na komunikację z urządzeniami, aby sprawdzić poprawność działania prototypu wykonanego na płytce stykowej. Dla każdego urządzenia stworzono osobny plik .h zawierający deklaracje funkcji oraz plik .cpp zawierający ich definicje. W dalszej kolejności zostało opisane użycie poszczególnych bibliotek.

8.4.1 Dodanie portalu przechwytyującego

Pierwszym krokiem, umożliwiającym łatwą konfigurację połączenia WiFi w sterowniku, było wykorzystanie biblioteki ESP Acync WiFiManager [8], która pozwala na automatyczne tworzenie sieci WiFi ad-hoc i portalu przechwytyującego. Daje to możliwość wyboru oraz połączenia się z docelową siecią WiFi, zapewniającą łączność sterownika z Internetem. Dla utworzenia WiFi Menagera należy konstruktorowi przekazać dwa obiekty: serwer web oraz server dns. Następnie powinno się zdefiniować funkcję zwrotną (callback) wykonywaną w momencie utworzenia przez WiFi Manger punktu dostępowego. Aby uruchomić Menagerera należy wywołać metodę autoConnect, przekazując jej, jako opcjonalne parametry, nazwę tworzzonego acces pointa oraz hasło. Po uruchomieniu Menager próbuje się połączyć z poprzednio zapamiętaną siecią WiFi. Jeśli jej nie odnajdzie, tworzy punkt dostępowy oraz portal przechwytyujący, w którym można wybrać inną sieć.

8.4.2 Dodanie biblioteki dla DHT22

Do komunikacji z czujnikiem temperatury i wilgotności wykorzystano bibliotekę DHT sensor library for ESPx [6]. Aby zainicjować działanie biblioteki należy skorzystać z metody setup podając jako parametry numer pinu, do którego podłączony jest czujnik, oraz model czujnika. Do odczytania pomiarów służą metody getTemperature oraz getHumidity.

8.4.3 Dodanie biblioteki dla MAX6675

Do komunikacji z termoparą w projekcie wykorzystano bibliotekę MAX6675 library [31]. Aby zainicjować moduł należy skorzystać z konstruktora, przekazując jako parametry numery pinów używanych do komunikacji z magistralą SPI. W celu odczytania temperatury należy skorzystać z metody readCelsius.

8.4.4 Dodanie biblioteki dla LCD

Do komunikacji z wyświetlaczem LCD, przy udziale magistrali I2C, wykorzystałem bibliotekę LiquidCrystal PCF8574 [30]. Adres wyświetlacza na magistrali, przekazany jako argument konstruktora pozwala zainicjować bibliotekę. Metoda setBacklight pozwala na włączenie podświetlenia wyświetlacza. Do wypisywania tekstu na wyświetlaczu służą metody print oraz println.

8.4.5 Dodanie biblioteki dla serwomechanizmu

Do komunikacji z serwomechanizmem wykorzystano bibliotekę ESP32Servo [7]. Aby rozpocząć pracę należy skorzystać z metody attach, podając jako argument numer pinu będącego wyjściem sygnałowym. W celu ustawienia mechanizmu na wybraną pozycję należy skorzystać z metody write. Metoda detach służy do przerwania komunikacji z serwomechanizmem.

8.4.6 Dodanie biblioteki dla PID

Do sterowania wychyleniem serwomechanizmu posłużono się biblioteką PID [39]. Pozwala ona na wykorzystanie regulatora proporcjonalno-całkującoc różniczkującego do utrzymywania wartości wyjściowej na zadanym poziomie. Aby zainicjować bibliotekę należy skorzystać z konstruktora, przekazując jako parametr referencje do wartości wejściowej, wyjściowej, zadanej oraz trzy parametry regulatora. Przy pomocy funkcji SetSampleTime należy ustalić

jak często regulator ma obliczać wartość wyjściową. Funkcja SetOutputLimits służy do określenia granic parametru wyjściowego. Natomiast funkcja SetControlDirection określa kierunek sprzężenia wejścia z wyjściem. Funkcja SetMode(AUTOMATIC) uruchamia działanie regulatora.

8.4.7 Dodanie biblioteki dla DS18B20

Do komunikacji z termometrem Dallas wykorzystano bibliotekę DallasTemperature [5]. Aby zainicjować bibliotekę należy skorzystać z konstruktora, przekazując jako parametr referencję do obiektu oneWire. W celu odczytania temperatury należy skorzystać z metody requestTemperatures, a następnie getTempCByIndex.

8.4.8 Dodanie biblioteki dla W5500 Light

Do komunikacji z modułem W5500 Light wykorzystano bibliotekę Ethernet Library for Arduino [9]. Aby zainicjować bibliotekę należy użyć metody init, podając jako parametr numer pinu Chip Select. Następnie należy skorzystać z metody WizReset wykonującej sekwencję resetowania modułu. W celu rozpoczęcia połączenia Ethernetowego należy użyć metody begin, podając jako parametry adres MAC, adres IP, adres DNS, adres bramy oraz maskę.

8.4.9 Dodanie biblioteki WebThings

Do przekształcenia sterownika w rzecz webową wykorzystano bibliotekę webthing-arduino [49]. Tworzy ona prosty serwer implementujący Web of Things API. Aby zainicjować bibliotekę należy skorzystać z konstruktora, przekazując jako parametry nazwę oraz adres IP. Następnie należy utworzyć instancję urządzenia oraz dodać do niego jego właściwości. Aby rozpocząć pracę serwera należy skorzystać z metody begin. Do przekazywania wartości właściwości urządzenia służą metody setValue oraz update.

8.5 Dodanie możliwości zdalnego uaktualniania oprogramowania

Mechanizm Over The Air Update pozwala na wgrywanie nowych wersji oprogramowania przez sieć LAN lub Internet (jeśli urządzenie posiada publiczny adres IP). W celu uruchomieniu OTA należy, w pliku konfiguracyjnym projektu jako protokół wgrywania, wybrać espota, a następnie podać

adres urządzenia. Dla zabezpieczenia przed nieuprawnioną zmianą oprogramowania należy ustawić hasło blokujące dostęp.

8.6 Dodanie odczytu napięć

Mikrokontroler ESP32 posiada dwa 12 bitowe przetworniki ADC, z których drugi nie może być używany, gdy wykorzystywane jest WiFi. Domyślona rozdzielcość, 12 bitów (0 – 4095), może być zmniejszona do 9 bitów (0 – 511). Przy pomocy metody pinMode można określić numer pinu oraz tryb pracy (INPUT). Następnie należy skorzystać z metody analogRead, aby odczytać wartości napięć pinów mikrokontrolera.

8.7 Dodanie wykrywania dotyku

Mikrokontroler ESP32 posiada wbudowane 10 pojemnościowych sensorów dotyku, ale wykorzystana w projekcie płytką deweloperską ma wyprowadzone ich tylko 9. Odczyt sensorów dotyku jest bardzo prosty - wystarczy użyć funkcji touchRead jako argument, podając numer pinu.

8.8 Dodanie obsługi brzęczyka

Do wytwarzania dźwięku brzęczyk potrzebuje sygnału PWM. Do generowania sygnału PWM, przez mikrokontroler, wykorzystano moduł LED Control z framework'a. W pierwszej kolejności należy skorzystać z funkcji ledcSetup, podając trzy argumenty: kanał, częstotliwość oraz rozdzielcość. Następnie, aby rozpocząć generowanie należy skorzystać z funkcji ledcAttachPin, podając dwa argumenty: numer pinu, do którego podłączone jest wejście sygnałowe brzęczyka oraz kanał. Aby zaprzestać generowania należy skorzystać z funkcji ledcWrite, podając dwa argumenty: kanał oraz 0 (cykl pracy).

8.9 Testowanie sterownika

Po każdym etapie dodawania kolejnych modułów projektu, działanie sterownika było testowane z wykorzystaniem monitora portu szeregowego poprzez odczytanie mierzonych przez sterownik wartości.

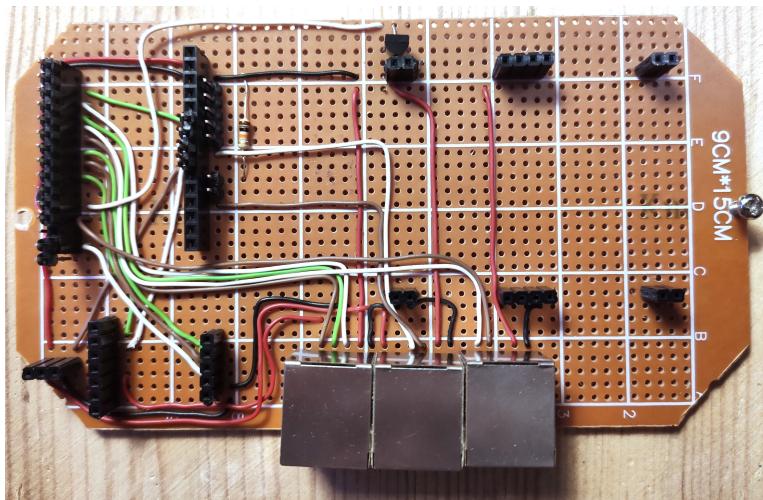
Rozdział 9

Wykonanie prototypu sterownika pieca kominkowego na płytce uniwersalnej

W tym rozdziale opisano przeniesienie prototypu z płytki stykowej na płytkę uniwersalną. Wykorzystanie płytki uniwersalnej pozwala na pewniejsze połoczenie podzespołów układu, jednocześnie pozostawiając wciąż pewną elastyczność do zmian w projekcie, co umożliwia dalsze jego doskonalenie. Zdjęcie 9.1 przedstawia płytke uniwersalną z gniazdami i połączeniami.

9.1 Rozmieszczenie elementów

Dla wygodnego połączenia elementów wybrano płytke uniwersalną o rozmiarach 9cm x 15cm ze standardowym rastrem otworów 2,54mm. Na początku na płytce umieszczono gniazda pod największe elementy: moduł zasilania o wymiarach 9,8cm x 2,9cm oraz mikrokontroler o wymiarach 5,5cm x 2,8cm. Gniazda pod kolejne elementy rozmieszczone tak, aby porty komunikacyjne oraz zasilania znajdowały się na dwóch krawędziach płytki. Największą trudność sprawiło umieszczenie na płytce portów sieciowych, gdyż wyprowadzenie ich połączeń sygnałowych nie jest zgodne z rastrem 2,54mm, a dodatkowo potrzebne są otwory montażowe.



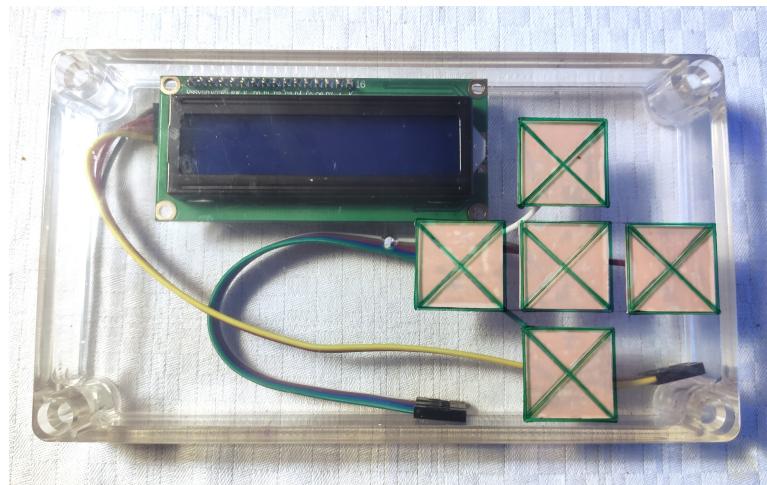
Rys. 9.1: Płytki uniwersalne z gniazdami i połączeniami. (opracowanie własne)

9.2 Wykonanie panelu dotykowego

Panel dotykowy wykonano z blaszek miedzianych o wymiarach 2cm x 2cm, przyklejonych do pokrywy sterownika, przylutowanych do przewodów zakończonych gniazdami. Podczas wycinania elementów należało zachować szczególną staranność dla zapewniania estetyki wykonania. Zdjęcie 9.2 przedstawia wieczko wraz z panelem dotykowym i wyświetlaczem LCD.

9.3 Połączenie elementów

Gniazda oraz goldpiny wszystkich elementów zostały połączone jednożutowymi przewodami miedzianymi w izolacji. Przewody rozmieszczono tak, aby łatwo można było prześledzić drogę połączenia, a następnie przylutowano je do płytka uniwersalnej. Różne kolory izolacji przewodów ułatwiają identyfikację połączeń. Zasilanie do złącz RJ-45 zostało doprowadzone tak, aby oba przewody znajdowały się w jednej parze skrętki, co pozwala na zmniejszenie ilości generowanych zakłóceń. Sprawdzenia poprawności połączeń dokonano przy pomocy miernika uniwersalnego. Zdjęcie 9.3 przedstawia spodnią stronę płytka uniwersalnej z widocznymi połączeniami lutowanymi.



Rys. 9.2: Wieczko wraz z panelem dotykowym i wyświetlaczem LCD. (opracowanie własne)

9.4 Zmiany w projekcie

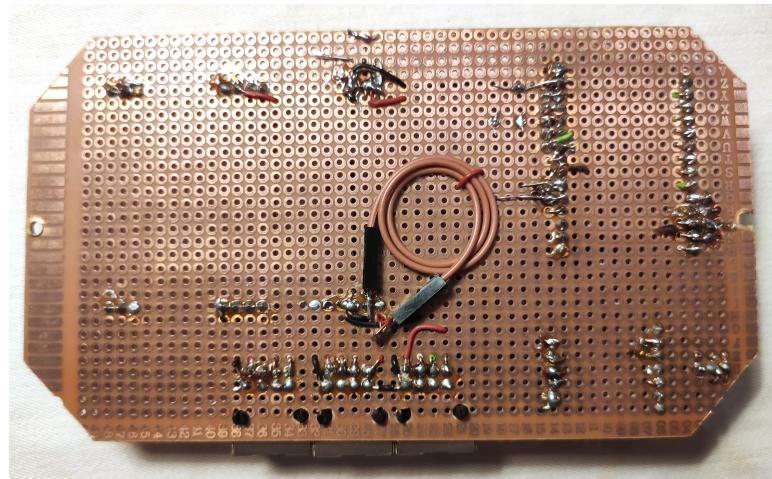
Na etapie wykonywania prototypu na płytce uniwersalnej postanowiono wprowadzić kilka zmian do projektu sterownika.

9.4.1 Dodanie dzielnika napięcia

W celu wykrywania aktualnego źródła zasilania sterownika, zdecydowano się na pomiar napięcia zasilania diod sygnalizacyjnych w układzie modułu zasilania awaryjnego. Aby dostosować napięcia do zakresu bezpiecznego dla mikrokontrolera, zastosowano dzielniki napięcia składające się z rezystorów 1k Ohm.

9.4.2 Dodanie głównegołącznika zasilania

Dla umożliwienia łatwiejszego włączania zasilania zdecydowano się umieścić w projekcie dodatkowy przełącznik zasilania podłączony do modułu zasilania awaryjnego. Do małego przełącznika, znajdującego się w module zasilania, przylutowano dwa goldpiny. Duży przełącznik podłączono do przewodów zakończonych gniazdami szpilkowymi pasującymi do goldpinów.



Rys. 9.3: Spodnia strona płytki uniwersalnej. (opracowanie własne)

9.5 Testowanie prototypu

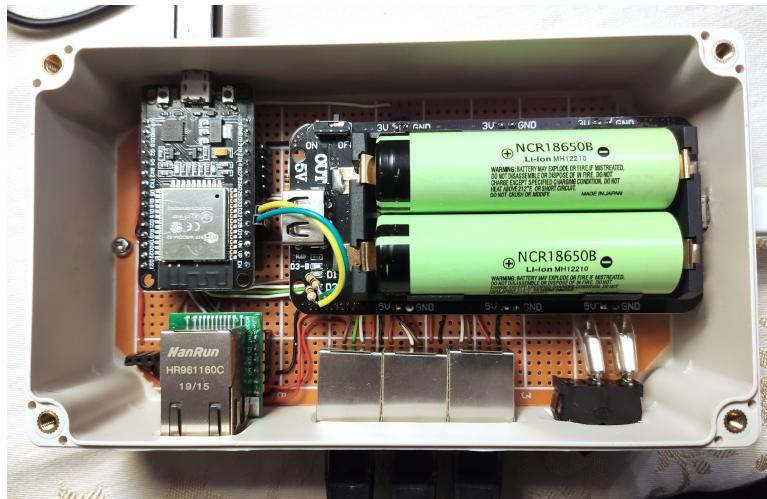
Po podłączeniu wszystkich elementów sterownika do płytka uniwersalnej dokonano testu działania z wykorzystaniem monitora portu szeregowego.

Rozdział 10

Wykonanie obudowy sterownika pieca kominkowego

Pierwotnie, w oparciu o pracę Francuza [24] zamierzano wydrukować obudowę w drukarce 3D. Jednak wielkość potrzebnej obudowy oraz czas niezbędny do wykonania projektu przesądziły o wykorzystaniu gotowej skrzynki, wykonanej z plastiku o przeźroczystym wieczku w rozmiarze 15,8cm x 9cm x 6cm, pozwalającej na swobodne zmieszczenie wszystkich komponentów sterownika. Płytką uniwersalną została przyjęta na rogach, ponieważ obudowa posiada tam miejsca na śruby. W skrzynce wykonano 5 otworów: pierwszy na górnej powierzchni obudowy (względem docelowego umiejscowienia sterownika) na port micro USB, który jest przeznaczony na wypadek potrzeby bezpośredniego podłączenia do mikrokontrolera. Drugi na prawej płaszczyźnie, również na port micro USB, przeznaczony do zasilania podczas normalnej pracy. Natomiast na dolnej powierzchni obudowy znajdują się trzy otwory: z lewej strony wejście na gniazdo 8p8c, przeznaczone do połączenia z internetem. W środkowej części potrójne gniazdo 8p8c, służące do komunikacji z czujnikami zewnętrznymi oraz serwomechanizmem. Po prawej stronie znajduje się miejsce na główny włącznik. Po wykonaniu otworów i zamontowaniu komponentów sterownika przystąpiono do montowania panelu dotykowego. Poszczególne pola panelu zostały przyklejone do przezroczystego wieczka obudowy na kształt "+", każdy w odległości około 4mm od drugiego, co pozwala na minimalizację zakłóceń. Ostatnim etapem wykonania obudowy było wklejenie ekranu LCD, w lewym górnym rogu przeźroczystego wieczka sterownika. Po umieszczeniu w obudowie i podłączeniu wszystkich elementów sterownika dokonano testu działania z wykorzystaniem monitora portu szeregowego. Zdjęcie 10.1 przedstawia spodnią część obudowy wraz z kom-

ponentami sterownika pieca kominkowego.



Rys. 10.1: Spodnia część obudowy wraz z komponentami sterownika. (opracowanie własne)

Rozdział 11

Opracowanie oprogramowania - część druga

W niniejszym rozdziale opisano dalszą część tworzenia oprogramowania sterownika, skupiając się na doskonaleniu i rozbudowie poszczególnych modułów oprogramowania o nowe funkcje i udogodnienia dla potencjalnego użytkownika.

11.1 Rozbudowa portalu przechwytyjącego

W portalu przechwytyjącym zaprogramowano funkcję, wywoływaną w pętli głównej, podejmującą próbę połączenia z zapamiętaną siecią WiFi w przypadku utraty łączności.

11.2 Rozbudowa obsługi termometru i czujnika wilgotności DHT22

W pliku dht.cpp stworzono funkcję pozwalającą na przekazanie indeksu cieplnego do zmiennej. Postanowiłem wykorzystać go jako punkt odniesienia dla sterownika zamiast temperatury, gdyż wcześniej oddaje on warunki termiczne panujące w pomieszczeniu. Dla zabezpieczenia poprawnej pracy sterownika, w przypadku niepowodzenia odczytu lub wątpliwych wartości temperatury i wilgotności (na podstawie których obliczany jest indeks cieplny), funkcja zwraca informację o błędzie.

11.3 Rozbudowa obsługi termopary z modułem MAX6675

W celu zabezpieczenia poprawnej pracy sterownika funkcja zwraca wartość 999 stopni Celsjusza oraz wypisuje komunikat o błędzie w przypadku: braku odczytu temperatury, wartości odbiegającej od stanu faktycznego w porównaniu z poprzednimi pomiarami. Zwrócenie przez funkcję tak wysokiej temperatury spowoduje szybkie zamknięcie przepustnicy przez regulator PID, co będzie chroniło piec przed przegrzaniem.

11.4 Rozbudowa obsługi wyświetlacza LCD

Z obsługi LCD usunięto metodę testową. Dodano funkcję wyświetlającą podstawowe nastawy sterownika: zadaną temperaturę w kominie, aktualną temperaturę w kominie, zadany indeks cieplny w pomieszczeniu, aktualny indeks cieplny w pomieszczeniu oraz procentowe otwarcie przepustnicy. Napisano również funkcję, pozwalającą na włączanie podświetlenia LCD i wyłączanie go po zadanym czasie.

11.5 Rozbudowa obsługi serwomechanizmu

Obsługę serwomechanizmu uzupełniono o wyłączenie sygnału sterującego po zmianie pozycji, co w znacznym stopniu przyczyniło się do poprawy kultury pracy. Często występującym problemem było blokowanie się przepustnicy, ze względu na duże opory, przy nieznacznej zmianie wychylenia. Po wyłączeniu sygnału PWM serwomechanizm zaprzestaje prób usytuowania się w zadawanej pozycji i zostaje w tej, którą osiągnął wcześniej. Zapobiega to niepotrzebnemu poborowi energii oraz redukuje głośny dźwięk wydawany przez urządzenie.

11.6 Rozbudowa obsługi regulatora PID

W celu poprawy komfortu termicznego w pomieszczeniu, wykorzystano dwa regulatory PID. Pierwszy z nich, na podstawie aktualnej temperatury w pomieszczeniu i zadanej temperatury, oblicza zadaną temperaturę dla spalin w kominie. Drugi, na podstawie aktualnych odczytów temperatury z komina i temperatury zadanej przez pierwszy z regulatorów, oblicza wychylenie serwomechanizmu, a co za tym idzie, stopień otwarcia przepustnicy

powietrza dolotowego do komory spalania. Dla większej odporności na błędy, dodano funkcje sprawdzające poprawność wartości wejściowych oraz tych obliczanych przez regulatory. Dodatkowo stworzono funkcję wypisującą powyższe wartości na port szeregowy.

11.7 Rozbudowa obsługi termometru Dallas DS18B20

Obsługę termometru Dallas uzupełniono o sprawdzanie odczytywanej wartości oraz informowanie o błędny odczycie.

11.8 Rozbudowa obsługi modułu Ethernet W5500

Wbrew wcześniejszym założeniom, nie udało się w przyjętym czasie przełożyć komunikacji z radiowej na kablową. Biblioteka arduino-webthings, przy pracy z mikrokontrolerem ESP32, domyślnie wykorzystuje tylko połączenie WiFi. Aby umożliwić wykorzystanie połączenia Ethernetowego, potrzebne są zwiększone nakłady pracy. Obsługa modułu Ethernet W5500 została jedynie poszerzona o funkcje z biblioteki, sprawdzające rodzaj wykrytego urządzenia oraz status połączenia kablowego.

11.9 Rozbudowa obsługi odczytu napięć

Obsługę odczytu napięć, uzupełniono o funkcję rozpoznającą źródło zasilania, która na podstawie dobranych doświadczalnie wartości progowych, cyklicznie przekazuje wyniki do części programu obsługującej WebThings.

11.10 Rozbudowa obsługi wykrywania dotyku

Moduł rozpoznawania dotyku uzupełniono o funkcję automatycznie dobierającą wartości progowe, zaraz po uruchomieniu sterownika. Dodano funkcję włączającą wyświetlacz LCD po wykryciu dotyku. Ponadto, utworzono funkcję, wysyającą cyklicznie odczytywane przez detekcję dotyku wartości do części programu obsługującej WebThings.

11.11 Rozbudowa obsługi brzęczyka

Do obsługi brzęczyka dodano funkcję włączającą oraz wyłączającą alarm.

11.12 Rozbudowa obsługi WebThings

Serwer WebThings uzupełniono o 6 różnych urządzeń, dla których określono właściwości zgodne z WoT Capability Schemas [52]:

- Sensor DHT22 posiada dwie własności: temperaturę (TemperatureProperty) i wilgotność (LevelProperty).
- Sensor termopary posiada jedynie własność temperatury (TemperatureProperty).
- Sensor PID posiada 6 własności: stopień otwarcia przepustnicy pieca (LevelProperty), temperaturę pomieszczenia (TemperatureProperty), zadaną temperaturę pomieszczenia (TargetTemperatureProperty), własność grzanie/chłodzenie (HeatingCoolingProperty), temperaturę w kominie (TemperatureProperty) oraz zadaną temperaturę w kominie (TemperatureProperty).
- Sensor Dallas posiada jedną własność: temperaturę (TemperatureProperty).
- Sensor dotyku posiada 5 własności (OnOffSwitch), po jednej dla każdego pola sensora.
- Sensor zasilania posiada dwie własności (OnOffSwitch): zasilanie zewnętrzne oraz bateria pełna.

Po określeniu właściwości, należało także przypisać każdej z nich odpowiednie atrybuty, takie jak: tytuł, wielokrotność, jednostkę, tryb odczytu. Ostatecznie, do każdego z sensorów, dodano funkcję aktualizującą wartości w serwerze WebThings.

11.12.1 Napotkane problemy

Pierwszym ze znalezionych problemów było zwracanie niepełnego opisu rzeczy webowej w formacie JSON przez serwer WebThings. Przyczyna tkwiła w definiowaniu przez bibliotekę webthing-arduino zbyt małego rozmiaru dokumentu JSON. Po zgłoszeniu problemu na stronie projektu, został on naprawiony poprzez umożliwienie użytkownikom wyboru zwiększonego bufora lub definiowania jego rozmiaru.

Drugim z napotkanych problemów było uaktualnianie tylko jednej z właściwości sensora, po wywołaniu funkcji update, z biblioteki webthing-arduino. Poszukiwanie rozwiązania przyniosło odpowiedź: przyczyną problemu jest

błąd w dodatku WebThing do Bramy Mozilli. Tymczasowym rozwiązaniem, pozwalającym na poprawne przekazywanie wartości do bramy, jest wywoływanie funkcji update po zmianie każdej własności z osobna.

11.13 Testowanie sterownika

Po opracowaniu kolejnej części oprogramowania, działanie sterownika było testowane z wykorzystaniem monitora portu szeregowego, poprzez odczytanie mierzonych wartości.

Rozdział 12

Zintegrowanie sterownika pieca kominkowego z Mozilla Gateway

W tym rozdziale opisany został proces integrowania sterownika pieca kominkowego z Bramą Mozilli.

12.1 Przygotowanie lokalnej Bramy WebThings

Projekt WebThings Gateway by Mozilla oferuje cztery możliwości instalacji: wykorzystanie obrazu dla Raspberry Pi, wykorzystanie obrazu Docker, użycie pakietu na Arch Linux lub samodzielna budowa ze źródła. Wybrano opcję wykorzystania gotowego obrazu dla Raspberry Pi. Po jego wgraniu i uruchomieniu urządzenia należy przejść do procesu konfiguracji [50], połączyć się z daną siecią WiFi, natomiast w portalu przechwytyującym wskazać połączenie z jakiego będziemy korzystać. Następnie należy wybrać subdomenę oraz utworzyć konto pierwszego użytkownika. Wtedy brama staje się gotowa do działania.

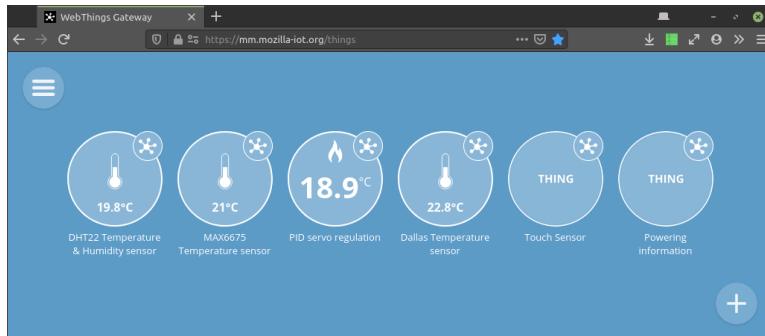
12.2 Dodanie rzeczy webowych do Bramy WebThings

Kolejnym etapem integracji było dodanie stworzonych przez sterownik rzeczy webowych do Bramy. Po włączeniu go i połączeniu z tą samą siecią

lokalną co Bramą, należy wybrać opcję dodawania urządzeń. Bramą automatycznie wykrywa rzeczy webowe i proponuje dodanie poszczególnych sensorów. Większość opcji można pozostawić jako domyślne, potwierdzając tylko dodawanie.

12.3 Wykorzystywanie interfejsu Bramy WebThings

Dzięki zapewnieniu przez Fundację Mozilli zdalnej usługi tunelowania istnieje możliwość zalogowania się do Bramy WebThings poprzez, wybrany na etapie przygotowania, adres subdomeny oraz login i hasło. Po zalogowaniu się widoczne są aktualne parametry pracy sterownika oraz istnieje opcja zmiany temperatury zadanej w pomieszczeniu. W zakładce "Logs", w interfejsie Bramy WebThings, jest możliwość dodawania wykresów czasowych poszczególnych własności sensorów sterownika. Natomiast zakładka "Rules" umożliwia tworzenie reguł, np.: informujących użytkownika o wzroście temperatury poprzez powiadomienia push. Zrzut ekranu 12.1 przedstawia interfejs Bramy WebThings.



Rys. 12.1: Interfejs Bramy WebThings. (opracowanie własne)

12.4 Tłumaczenie interfejsu Bramy WebThings

Aby poprawić dostępność i wygodę stworzonego rozwiązania, dla osób nie znających języka angielskiego, postanowiono dokonać tłumaczenia interfejsu Bramy WebThings na język polski. Projekt Bramy WebThings umożliwia dodawanie tłumaczeń z wykorzystaniem systemu Fluent [40], który zapewnia naturalne brzmienie przekładu. Przygotowane tłumaczenie zostało dodane

do projektu Bramy WebThings jako pull request [51]. Po dokonaniu poprawek, zostało zatwierdzone i będzie dostępne po opublikowaniu najnowszej wersji Bramy WebThings.

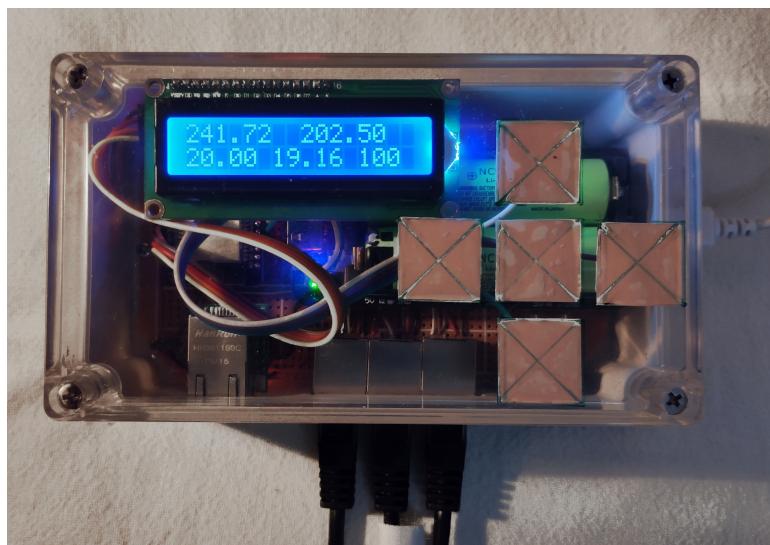
12.5 Testowanie sterownika

Po zintegrowaniu sterownika z Bramą WebThings, przeprowadzono testy manualne z wykorzystaniem interfejsu graficznego Bramy. Ułatwieniem w prowadzeniu testów była możliwość stworzenia wykresów parametrów sterownika w Bramie.

Rozdział 13

Efekt końcowy na tle koncepcji

Z góry narzucony termin wykonania zawsze pozostawia pewien niedosyt, co do zrealizowanego projektu, jednakże wstępny zarys koncepcji został, prawie całkowicie, zrealizowany. To, czego nie udało się osiągnąć, dotyczy braku możliwości informowania o zdarzeniach bezpośrednio przez sterownik, za to realizowane jest za pomocą reguły Bramy. Do niepowodzeń można zaliczyć także brak minimalizacji otwarcia przepustnicy w przypadku zaniku napięcia. Jednak sterownik spełnia swoje podstawowe zadanie, czyli pozwala na bezobsługową pracę paleniska pomiędzy momentami dokładania paliwa, a wykorzystanie serwera WebThings umożliwia korzystanie z Bramy WebThings Mozilli oraz łatwą interakcję sterownika z innymi urządzeniami lub usługami online. Zdjęcie 13.1 przedstawia kompletny sterownik pieca kominkowego.



Rys. 13.1: Kompletny sterownik pieca kominkowego. (opracowanie własne)

Rozdział 14

Podsumowanie

W niniejszej części zostaną opisane wnioski z pracy według kolejności wcześniejszych przedstawionych rozdziałów.

14.1 Przegląd istniejących rozwiązań

Przegląd istniejącej oferty w tym segmencie pokazał jak mała ilość rozwiązań w zakresie sterowania pieców kominkowych istnieje na polskim rynku, co dowodzi temu, że potrzebne są nowe propozycje produktów.

14.2 Opracowanie koncepcji budowy sterownika pieca kominkowego

Przy tworzeniu koncepcji budowy sterownika pieca kominkowego wyzwaniem było zrozumienie procesu poprawnego spalania drewna. Zadaniem sterownika jest tylko pomóc w utrzymywaniu właściwej temperatury spalin. Ważne jest, aby z jednej strony nie doprowadzić do przegrzania pomieszczenia, a z drugiej nie obniżyć temperatury panującej w piecu na tyle by powodować niedopalanie gazów drzewnych. Niestety, tu główna odpowiedzialność spoczywa na użytkowniku, dlatego ważne jest aby w dołączonej instrukcji obsługi opisać nie tylko samo użytkowanie sterownika, ale i uniwersalne, poprawne zasady palenia w piecu kominkowym.

14.3 Wybór podzespołów sterownika pieca kominkowego

Podczas wybierania podzespołów sterownika pieca kominkowego najwięcej trudność sprawiła wybór wśród mnogości dostępnych na rynku rozwiązań. Przeprowadzając analizę wielokryterialną wzięto pod uwagę nie tylko cenę i kompatybilność podzespołów, ale także stopień skomplikowania komunikacji oraz dostępne biblioteki tak, aby nie naruszyć ustalonego terminu oddania projektu. Głównym wnioskiem jest to, że przy mniejszej docelowej skali wytwarzania końcowych produktów należy wybrać podzespoły droższe, ale o większej skali integracji, na przykład gotową płytę deweloperską ESP32 z modułem Ethernet. Pomimo, że cena podzespołu jest dwukrotnie wyższa, jego wybór pozwoliłby na znaczną oszczędność czasu poświęconego na łączenie elementów.

14.4 Wykonanie prototypu sterownika pieca kominkowego na płytce stykowej

Wykorzystanie płytki stykowej do tworzenia prototypu nie jest procesem skomplikowanym. Jednak niedostateczna pewność połączeń może sprawiać trudne do wykrycia błędy w komunikacji. Przy dużej ilości elementów w projekcie należałoby łączyć podzespoły w grupy na osobnych płytach.

14.5 Opracowanie oprogramowania - część pierwsza

Największą trudność podczas opracowywania pierwszej - podstawowej - części oprogramowania przyniosło zaznajomienie się z ekosystemem PlatformIO oraz obsługą bibliotek wykorzystanych w projekcie. Aby sprawnie z nich korzystać należało zapoznać się z licznymi przykładami i wybrać te, które przedstawiają poprawne koncepcje projektowe.

14.6 Wykonanie prototypu sterownika pieca kominkowego na płytce uniwersalnej

Wykonanie prototypu na płytce uniwersalnej wymagało dobrych zdolności manualnych, precyzji wykonania oraz umiejętności posługiwania się

lutownicą. W ciasnym sąsiedztwie wylutowywanie błędnie umieszczonych elementów sprawiało dużą trudność. Wybór portów 8p8c, z wyprowadziami w rastrze 2,54mm, lub tych, umieszczonych na płytach drukowanych, znacznie ułatwiały proces łączenia ich z płytą uniwersalną. Praca włożona w konstrukcję prototypu na płytce uniwersalnej pozwoliła na niezawodne połączenie komponentów sterownika i dała większą pewność przy opracowywaniu oprogramowania.

14.7 Wykonanie obudowy sterownika pieca kominkowego

Tworzywo plastikowe, wykorzystane do stworzenia skrzynki, umożliwiło łatwe wycięcie otworów na porty komunikacyjne sterownika.

14.8 Opracowanie oprogramowania - część druga

W drugiej części opracowywania oprogramowania największą trudność sprawiło wykorzystanie biblioteki arduino-webthings, ponieważ jest to nowy zestaw komponentów programu, nie posiadający jeszcze doprecyzowanej dokumentacji. Podczas opracowywania kodu programu, przyczyną znaczącego spowolnienia pracy były błędy znajdujące się nie w kodzie użytkownika, a w bibliotece. Wykorzystywanie oprogramowania we wczesnym stadium rozwoju zawsze wiąże się z ryzykiem występowania niezidentyfikowanych jeszcze błędów i niekompletną dokumentacją, dlatego potrzebny jest znaczny zapas czasu na rozwiązywanie nieprzewidzianych problemów.

14.9 Zintegrowanie sterownika pieca kominkowego z Mozilla Gateway

Uaktualnianie poprzednich wersji Bramy WebThings doprowadziło do jej wadliwego działania. Podczas testów, stosowanie dużej ilości wykresów, a w konsekwencji zwiększonej liczby zapisów i odczytów z karty pamięci, doprowadziło do jej szybkiej degradacji i konieczności wymiany. Bardziej efektywnym rozwiązaniem byłoby zastosowanie innej platformy zamiast Raspberry Pi, która nie posiada wbudowanej pamięci, oraz skorzystanie z obrazu Dockera.

Rozdział 15

Dalszy kierunek prac

Posiadając wiedzę uzyskaną w trakcie tworzenia projektu podszedłbym inaczej do problemu opracowania koncepcji oraz wykonania sterownika pieca kominkowego. Przede wszystkim wykorzystałbym inne źródło zasilania awaryjnego, dające bezpośrednio więcej informacji o stanie baterii. Wykorzystałbym też płytę deweloperską mikrokontrolera ESP32 z większą ilością wyprowadzeń, tak aby pozostawić sobie pole do rozszerzenia liczby podłączonych czujników. W kwestii programowej, poświęciłbym dodatkowy czas, aby stworzyć testy jednostkowe, których zabrakło w tej wersji oprogramowania.

Podjętym bezpośrednio, dalszym kierunkiem pracy będzie poprawa bezpieczeństwa funkcjonowania sterownika tak, aby piec kominkowy nie wymagał nadzoru na dłuższy czas. Kolejnym krok stanowić będzie podział pracy pieca kominkowego na odrębne fazy i opracowanie do nich indywidualnych podprogramów. Następnie stworzone zostanie menu użytkownika wyświetlane na LCD. W dalszej kolejności komunikacja z internetem zostanie przeniesiona z WiFi na połączenie Ethernet.

Książki

- [2] A. Badyda and H. Mazurek, *Smog. Konsekwencje zdrowotne zanieczyszczeń powietrza*, W. L. PZWL., Ed. Warszawa: PZWL Wydawnictwo Lekarskie, 2018, p. 212.
- [13] T. Francuz, *Język C dla mikrokontrolerów AVR: od podstaw do zaawansowanych aplikacji*, G. W. Helion., Ed. Gliwice: Wydawnictwo Helion, 2015.
- [14] J. Grębosz, *Opus Magnum C++11 : programowanie w języku C++. Tom 1*, G. W. Helion., Ed. Gliwice: Wydawnictwo Helion, 2018, p. 607.
- [18] D. Guinard, V. M. Trifa, and P. Rajca, *Internet rzeczy: budowa sieci z wykorzystaniem technologii webowych i Raspberry Pi*, G. W. Helion., Ed. Gliwice: Wydawnictwo Helion, 2017, p. 384.
- [21] H. Hoppe, *Jak wymienić stary kocioł: (poradnik)*, W. i. H. K. KaBe., Ed. Krosno: Wydawnictwo i Handel Książkami "KaBe", 2018.
- [22] A. Huang and M. Baranowski, *Hardware hacker: przygody z konstruowaniem i rozpracowywaniem sprzętu*, W. N. P. W. N, Ed. Warszawa: Wydawnictwo Naukowe PWN SA, 2018.
- [24] A. Kaziunas France, *Świat druku 3D. Kompendium wiedzy o druku 3D!* G. W. Helion., Ed. Gliwice: Wydawnictwo Helion, 2014.
- [28] A. Kurczyk, *Mikrokontrolery STM32 dla początkujących*, B. T. C. Wydawnictwo, Ed. Legionowo: Wydawnictwo BTC, 2019.
- [32] H. Mazurek, *Smog: zagrożenie dla zdrowia czy moda na ekologię?* I. Publishing., Ed. Warszawa: ITEM Publishing, 2018.
- [33] S. Monk, *Arduino dla początkujących. Podstawy i szkice*, G. W. Helion., Ed. Gliwice: Wydawnictwo Helion, 2014.
- [34] S. Monk, *Arduino i Android: niesamowite projekty*, G. W. Helion., Ed. Gliwice: Wydawnictwo Helion, 2014.

- [35] S. Monk, *Elektronika z wykorzystaniem Arduino i Raspberry Pi. Receptury*, G. W. Helion., Ed. Gliwice: Wydawnictwo Helion, 2018.
- [45] S. P. Wallace and K. Matuk, *Płytki drukowane (PCB): nauka i projekty od podstaw*, G. W. Helion., Ed. Gliwice: Wydawnictwo Helion, 2019.

Artykuły

- [1] K. Ashton, “That Internet of Things Thing,” *RFID Journal*, vol. 4986, 2009.
- [41] D. Raggett, “Defragmenting the IoT with the Web of Things Enabling Open Markets of Services Defragmenting the Internet of Things,” no. May, 2018.

Prace dyplomowe

- [16] D. Guinard, “A Web of Things Application Architecture - Integrating the Real-World into the Web,” PhD thesis, 2011, p. 220.

Materiały konferencyjne

- [15] D. Guinard, “Mashing up your web-enabled home,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 6385 LNCS, 2010, pp. 442–446.
- [17] D. Guinard and V. Trifa, “Towards the Web of Things : Web Mashups for Embedded Devices,” in *Workshop on Mashups, Enterprise Mashups and Lightweight Composition on the Web (MEM 2009)*, 2009, pp. 1–8.
- [20] D. Guinard, V. Trifa, and E. Wilde, “A resource oriented architecture for the web of things,” in *2010 Internet of Things, IoT 2010*, 2010.
- [38] R. Pastor Vargas, M. Romero Hortelano, L. Tobarra Abad, J. Caño Carrillo, and R. Hernandez Berlinches, “Teaching cloud computing using Web of Things devices,” in *IEEE Global Engineering Education Conference, EDUCON*, vol. 2018-April, IEEE, 2018, pp. 1738–1745.

Pozostałe źródła

- [3] *Building the Web of Things*, 2017. [Online]. Available: <https://hacks.mozilla.org/2017/06/building-the-web-of-things/> (visited on 01/07/2020).
- [4] *Contributions to WebThings Gateway*, 2020. [Online]. Available: <https://github.com/mozilla-iot/gateway/graphs/contributors?from=2017-03-05&to=2020-01-07&type=c> (visited on 01/07/2020).
- [5] *Dallas Temperature library*, 2020. [Online]. Available: <https://github.com/milesburton/Arduino-Temperature-Control-Library> (visited on 01/06/2020).
- [6] *DHT sensor library for ESPx*, 2020. [Online]. Available: <https://github.com/beegee-tokyo/DHTesp> (visited on 01/06/2020).
- [7] *ESP32Servo library*, 2020. [Online]. Available: <https://github.com/madhephaestus/ESP32Servo> (visited on 01/06/2020).
- [8] *ESPAsyncWiFiManager library*, 2020. [Online]. Available: <https://github.com/alanswx/ESPAsyncWiFiManager> (visited on 01/06/2020).
- [9] *Ethernet library*, 2020. [Online]. Available: <https://github.com/arduino-libraries/Ethernet> (visited on 01/06/2020).
- [10] *EUROSTER Sterownik*, 2019. [Online]. Available: <https://sterownikitech.pl/sterowniki-euroster-11k,id119.html> (visited on 11/27/2019).
- [11] *EUROSTER Sterownik do kominka*, 2019. [Online]. Available: <https://www.euroster.pl/produkty/sterowniki/kominkowe/euroster-11k/293> (visited on 11/27/2019).
- [12] *EUROSTER Sterowniki*, 2019. [Online]. Available: <https://www.euroster.pl/produkty/sterowniki/kominkowe> (visited on 11/27/2019).

- [19] D. Guinard, V. Trifa, F. Mattern, and E. Wilde, “From the Internet of Things to the Web of Things: Resource-oriented Architecture and Best Practices,” in *Architecting the Internet of Things*, Springer Berlin Heidelberg, 2011, pp. 97–129.
- [23] “IoT W POLSKIEJ GOSPODARCE,” Tech. Rep., 2019. [Online]. Available: <https://www.gov.pl/attachment/82ad18f8-2ac1-4433-a1ea-f887b522e46b>.
- [25] E. Korkan, H. B. Hassine, V. E. Schlott, S. Käbisch, and S. Steinhorst, “WoTify: A platform to bring Web of Things to your devices,” 2019.
- [26] *Kratki Sterownik do kominka*, 2019. [Online]. Available: <https://kratki.com/sklep/pl/produkt/1435/sterownik-kominka-fi-100> (visited on 11/27/2019).
- [27] *Kratki Sterowniki do kominków*, 2019. [Online]. Available: https://kratki.com/sklep/pl/category/search?q=sterownik&category_id=0 (visited on 11/27/2019).
- [29] *Launching the Web of Things Interest Group*, 2014. [Online]. Available: <https://www.w3.org/blog/wotig/2015/01/20/launching-the-web-of-things-interest-group/> (visited on 01/07/2020).
- [30] *LiquidCrystal PCF8574 library*, 2020. [Online]. Available: https://github.com/mathertel/LiquidCrystal_PCF8574 (visited on 01/06/2020).
- [31] *MAX6675 thermocouple library*, 2020. [Online]. Available: <https://github.com/adafruit/MAX6675-library> (visited on 01/06/2020).
- [36] “PN-IEC 1131-1:1996/Ap1:1999 Sterowniki programowalne: postanowienia ogólne,” Warszawa, 1999.
- [37] “PN-M-42379:2000 Sterowniki programowalne: wytyczne dla użytkownika,” Warszawa, 2000.
- [39] *PID library*, 2020. [Online]. Available: <https://github.com/br3ttb/Arduino-PID-Library> (visited on 01/06/2020).
- [40] *Project Fluent*, 2020. [Online]. Available: <https://projectfluent.org/> (visited on 01/07/2020).
- [42] *TECH Moduł Ethernet ST-505*, 2019. [Online]. Available: https://sterownikitech.pl/modul-ethernet-st-505_id85.html (visited on 11/27/2019).
- [43] *TECH Sterownik do kominka DGP*, 2019. [Online]. Available: <https://www.techsterowniki.pl/p/st-3910-zpid> (visited on 11/27/2019).

- [44] *TECH Sterowniki do kominków*, 2019. [Online]. Available: <https://www.techsterowniki.pl/k/sterowniki-do-kominkow> (visited on 11/27/2019).
- [46] *Web of Things (WoT) Architecture*, 2019. [Online]. Available: <https://www.w3.org/TR/2019/CR-wot-architecture-20191106/> (visited on 01/07/2020).
- [47] *Web of Things (WoT) Thing Description*, 2019. [Online]. Available: <https://www.w3.org/TR/2019/CR-wot-thing-description-20191106/> (visited on 01/07/2020).
- [48] *Web Thing API*, 2020. [Online]. Available: <https://iot.mozilla.org/wot/> (visited on 01/07/2020).
- [49] *WebThing library*, 2020. [Online]. Available: <https://github.com/mozilla-iot/webthing-arduino> (visited on 01/06/2020).
- [50] *WebThings Gateway Getting Started*, 2019. [Online]. Available: <https://iot.mozilla.org/docs/gateway-getting-started-guide.html> (visited on 01/07/2020).
- [51] *WebThings Gateway Polish translation*, 2019. [Online]. Available: <https://github.com/mozilla-iot/gateway/pull/2281> (visited on 01/08/2020).
- [52] *WoT Capability Schemas*, 2020. [Online]. Available: <https://iot.mozilla.org/schemas/> (visited on 01/07/2020).