Uniwersytet w Białymstoku

Wydział Matematyki i Informatyki

Instytut Informatyki

Konrad Trzeciak

Interfejs obsługi systemu inteligentny dom opartego na programowalnym sterowniku logicznym

Praca dyplomowa napisana
pod kierunkiem
dr. Mariusza Żynela

Białystok 2019

Składam serdeczne podziękowania
dr. Mariuszowi Żynelowi
za zaangażowanie, merytoryczną pomoc
i cenne wskazówki.
Dziękuję firmie Elektrokomplex
za wypożyczenie sterownika PLC Fatek.

Konrad Trzeciak

Spis treści

Wstęp

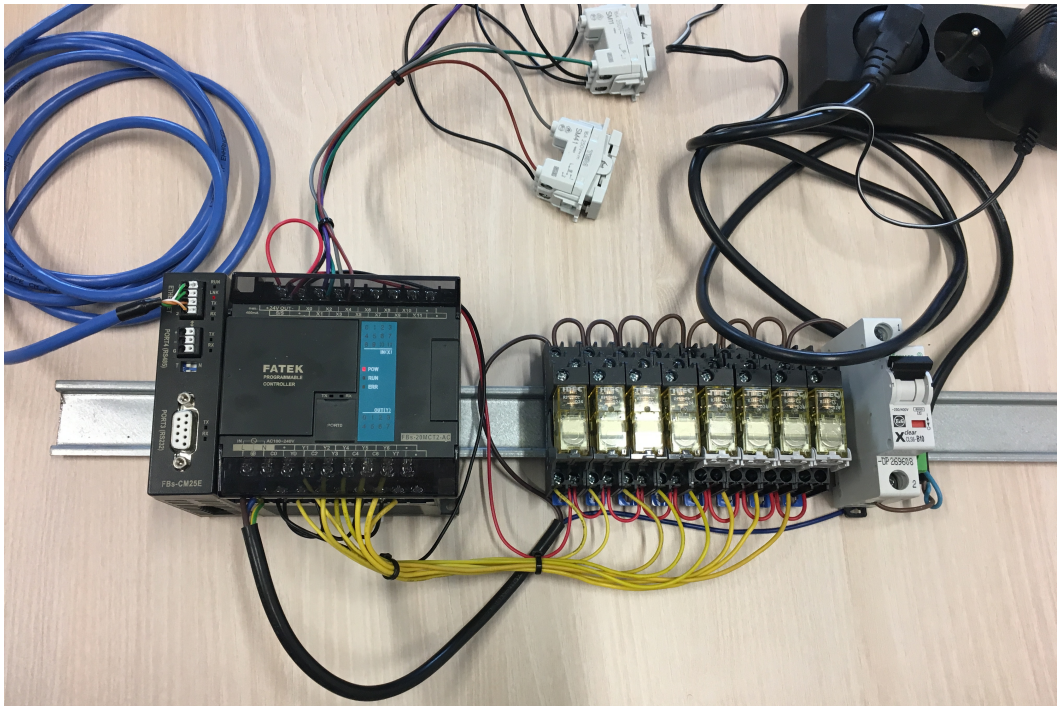
Już od dawna człowiek zastanawiał się jak może ułatwić sobie i bliskim życie poprzez udogodnienia związane z automatyzacją urządzeń w swoim domu. Każdy stara się o zapewnienie bezpieczeństwa i pełnego komfortu w swoich mieszkaniach. Osoby, które korzystają z systemów inteligentnego domu utwierdzają nas w fakcie, że technologie te są niezmiernie łatwe w obsłudze dla starszego jak i młodszego pokolenia oraz są przydatne w codziennym życiu każdego człowieka. Oczywiście ludzie radzili sobie przed pojawieniem się technologii inteligentnego domu. Wszelkie systemy w budynku miały swoje oddzielne zadania. Na przykład kamery i system alarmu strzegł bezpieczeństwa domowników. Kocioł grzewczy miał za zadanie utrzymać ciepło w mieszkaniu. Problem rozpoczął w przypadku gdzie domownicy zaczęli korzystać z nowocześniejszych instalacji typu automatyczne rolety na oknach, klimatyzacja, czy wentylacja. Aby zażegnać problem inżynierzy skonstruowali inteligentne technologie upraszczające centralne zarządzenie urządzeniami domowymi, dzięki czemu można z łatwością sterować każdym z urządzeń podłączonych do systemu.

Celem pracy jest implementacja interfejsu obsługi systemu inteligentny dom do kontroli urządzeń wchodzących w jego skład. System można podzielić na widoczny interfejs udostępniony klientowi, gdzie można wykonywać operacje związane z urządzeniami w mieszkaniu oraz warstwę administratora, który może zmieniać parametry urządzeń. Interfejs zaprojektowano pod konkretny budynek mieszkalny, ale można go rozbudować i dostosować do dowolnego mieszkania lub budynku, niekoniecznie mieszkalnego. Zazwyczaj technologia ta używana jest w domostwach o bogatszym wyposażeniu w zmechanizowane elementy codziennego użytku typu elektryczne żaluzje, czy elektrozawory sterujące obwodami grzewczymi.

Warto też podkreślić, że nie tylko w bogatszej technologicznie infrastrukturze można użyć system inteligentnego domu, ale również można używać tego systemu w ubogo wyposażonym mieszkaniu, gdzie mogą wystąpić tylko na przykład dwie lub trzy zmechanizowane instalacje. Nie wstając z wygodnego fotela, czy leżąc na łóżku można wyłączyć światło albo zmniejszyć temperaturę w pokoju za pomocą smartfona. Takie działanie jest możliwe wtedy, kiedy połączymy ze sobą trzy rzeczy: sterownik PLC, serwer HTTP, a także cały interfejs, do którego mamy wgląd oraz możliwość wykonania różnych operacji.

Interfejs użytkownika ukrywa szczegóły techniczne przed człowiekiem, pozwala wykonywać skomplikowane operacje mniej wprawionym użytkownikom nie wymagając od nich rozległej wiedzy na dany temat. Gdybyśmy musieli uruchomić aplikację do pisania programów na PLC, połączyć się ze sterownikiem, odszukać interesujący nas rejestr i zmienić jego wartość by zapalić światło w łazience to nie byłoby to fajne. Tego typu operacje ma nam ułatwiać tworzony interfejs.

Osoba zalogowana, będzie miała ukazane na stronie aplikacji poszczególne urządzenia domowe podłączone do systemu. Użytkownik patrząc na interfejs, będzie mógł jednym kliknięciem, na przykład, włączyć lub wyłączyć światło. Cała strona jest zaprojektowana tak, aby użytkownik nie miał żadnych problemów w poruszaniu się po interfejsie oraz korzystania z jego funkcji. Kod aplikacji jest na tyle przejrzysty, że dodawanie nowych urządzeń lub inne modyfikacje nie powinny sprawić kłopotu.



Rysunek 1: Sterownik PLC z dodatkowym modułem komunikacyjnym i przekaźnikami, na którym powstała praca.

Aby osiągnąć cel pracy niezbędny był sterownik PLC. Otrzymaliśmy go do testów z firmy Elektrokomplex specjalizującej się instalacjami oraz konserwacją inteligentnych budynków. Sterownik wraz z osprzętem pokazany jest na rysunku 1. Jest to urządzenie marki Fatek, model FBs-20MCT2-AC wraz z modułem komunikacyjnym FBs-CM25E. Dołączony moduł posiada najbardziej istotną rzecz dla nas, którą jest złącze Ethernetowe do komunikacji TCP/IP.

Rozdział 1

Zastosowane technologie

Interfejs użytkownika inteligentnego domu został opracowany przy użyciu kilku podstawowych, dobrze znanych, technologii do tworzenia witryn webowych. W kolejnych podrozdziałach przedstawimy krótko języki programowania zastosowane w projekcie.

1.1 HTML

HTML (Hypertext Markup Language) jest językiem znaczników służącym do tworzenia stron internetowych. Znaczniki są wykorzystywane w celu opisu struktury logicznej dokumentów hipertekstowych. Dzięki wbudowanym mechanizmom, w przeglądarkach odbywa się przekształcanie takiego dokumentu do postaci wizualnej. Podstawową ideą HTML, kiedy powstawał, była możliwość łatwego przechodzenia pomiędzy różnymi, powiązanymi ze sobą, dokumentami. Ta funkcja HTML przetrwała do dzisiaj, ale mało kto o tym pamięta.

Kod HTML zapewnia podział dokumentu na logiczne składowe takie jak: nagłówki, akapity tekstu, obrazki, tabele, listy numerowane, listy wypunktowane, formularze i inne. W HTML5 dodano wiele nowych elementów składowych. Warto wiedzieć, że HTML tylko wprowadza pewne znaczniki i atrybuty, natomiast twórca dokumentu może stosować własne znaczniki i atrybuty tyle, że nie zostaną one rozpoznane przez przeglądarki.

Kod HTML można porównywać do szkieletu człowieka, a kod CSS można nazywać skórą, czyli czymś co nadaje wygląd zewnętrzny (patrz [3]).

1.2 CSS

CSS (Cascading Style Sheets), czyli kaskadowy arkusz stylów, służy do opisu wyglądu witryny internetowej. Dzięki stylom twórca ma możliwość zmiany koloru, układu strony, odstępu między akapitami, wszelkich wymiarów,

marginesów, wielkości, czy rodzaju czcionki itp.. Arkusz stylów CSS daje możliwość dostosowania witryny do zróżnicowanych typów urządzeń, na których dokument będzie prezentowany: ekrany, drukarki, projektory. Umożliwia również dostosowanie strony do wyświetlania na różnych wielkościach i rozdzielczościach ekranu.

CSS umożliwia oddzielenie prezentacji strony internetowej od jej treści. Ułatwia to dostosowanie stron do różnorakich środowisk. Przy tworzeniu strony można umieścić zawartość kodu CSS w pliku HTML lub w osobnym pliku, współdzielonym przez wiele stron HTML. Umieszczenie CSS w oddzielnym pliku pozwala wykorzystać dany styl w dowolnej liczbie witryn internetowych (patrz [3]).

1.3 JavaScript

JavaScript jest to język programowania, który działa w przeglądarce internetowej, czyli po stronie klienta. Jest używany głównie w celu poszerzenia interakcji użytkownika ze stroną internetową. Pozwala dodawać do witryn internetowych zdarzenia, czyli umożliwia porozumiewać się z użytkownikami, pobierać dane z Internetu do wyświetlania na ekranie, rysowania grafik bezpośrednio na witrynie, itp. Jest on jednym z trzech języków front-endowych.

Poprzez JavaScript elementy CSS i HTML nabywają możliwość animacji i ruchu na stronie internetowej. Język ten używa się głównie do udoskonalania witryny, polepszenia sprawności i komfortu poruszania się po stronie przez użytkownika. JavaScript jest również używany do tworzenia gier, aplikacji mobilnych, czy animacji komputerowych (patrz [1]).

Myśląc o programowaniu obiektowym należy spojrzeć na warunki, które musi spełniać dany język programowania. Jednym z podstawowych wymagań jest, aby język programowania potrafił przedstawić program, za pośrednictwem obiektów. Celem programowania obiektowego jest zapewnienie większej elastyczności i łatwości rozwoju w programowaniu. JavaScript jest językiem zorientowanym obiektowo, ale różni się od innych języków. Zarzuca mu się brak niektórych elementów typowych dla języków programowania obiektowego. Programowanie obiektowe promuje większą elastyczność i łatwość rozwoju kodu.

Mówiąc o programowaniu obiektowym w JavaScript trzeba wspomnieć o standardzie ECMAScrpit. Jedną z implementacji ES jest właśnie JavaScript. Został on stworzony przez stowarzyszenie ECMA, które miało na celu ustandaryzować systemy informatyczne znajdujące się w Europie. ECMAScript określa semantykę języka. Definiuje również: typy danych, dziedziczenie, obsługę wyjątków, instrukcje warunkowe, pętle, obiekty i funkcje. Takie elementy jak model dokumentu albo wyspecjalizowane funkcje wejścia/wyjścia obsługi GUI (Graficzny interfejs użytkownika), nie należą do specyfikacji ECMAScript. ECMAScript2015 wnosi wiele pomocnych zmian, które czynią napisany kod jesz-

cze bardziej przejrzystym, bardziej wydajnym oraz czytelniejszym.

1.4 AJAX

AJAX (Asynchronous JavaScript And XML) jest nową techniką służącą do tworzenia lepszych i szybszych aplikacji skonstruowanych za pośrednictwem języka HTML, CSS i JavaScript. Podstawowym zadaniem AJAX jest aktualizacja zawartości strony asynchronicznie, czyli przeglądarka internetowa nie musi ponownie ładować całej strony, kiedy nieduży fragment zawartości strony musi zostać zmieniony. AJAX jest technologią przeglądarki internetowej, która jest niezależna od oprogramowania serwera sieciowego.

Technika AJAX bazuje na obiekcie `XMLHttpRequest` w JacvaScript. Posiada on metody umożliwiające wysłanie żądania do serwera HTTP i odebranie odpowiedzi. Są jednak pewne ograniczenia. Serwer HTTP musi być w tej samej domenie, co strona webowa, na której uruchamiany jest AJAX. Twórca witryny ma możliwość wykonać na serwerze HTTP niemal każde zadanie, pobrać jego wynik i uzyskane dane wykorzystać do zmodyfikowania treści lub wyglądu swojej aplikacji webowej – wszystko całkowicie dynamicznie (patrz [1]).

1.5 PHP

PHP (Hypertext Preprocessor) jest powszechnie stosowanym językiem skryptowym, którego głównym zadaniem jest dynamiczne tworzenie strony. W odróżnieniu od HTML, CSS i JavaScript, które obsługuje przeglądarka, kod PHP jest wykonywany po stronie serwera HTTP, co oznacza, że w przeglądarce widzimy jedynie wynik działania skryptu PHP.

Dzięki PHP można połączyć się z bazami danych MySQL, Postgres, Oracle i innymi. Obecne wersje PHP pozwalają wykonywać bardzo złożone zadania. Skrypty w PHP pisze się nie tylko wtedy, gdy tworzona jest strona webowa. Przy ich pomocy można rozwiązywać wiele problemów programistycznych, niezależnych od HTML i przeglądarek.

PHP jest językiem open source. Nie potrzebujemy specjalnej licencji, aby z niego korzystać. Działa na niemal wszystkich platformach: Unix, Linux, MacOS, Windows i innych (patrz [4]).

Rozdział 2

Sterowniki PLC

Programowalny kontroler logiczny (PLC) jest przemysłową jednostką komputerową, która została przystosowana do obsługi procesów produkcyjnych. Stosowany jest do sterowania i kontroli wielu różnych urządzeń przemysłowych począwszy od prostej pakowarki produktów spożywczych po zrobotyzowane linie montażowe samochodów. Spotkamy go tam, gdzie wymagana jest wysoka niezawodność i powtarzalność procesów oraz przejrzystość programowania i łatwość diagnozowania problemów.

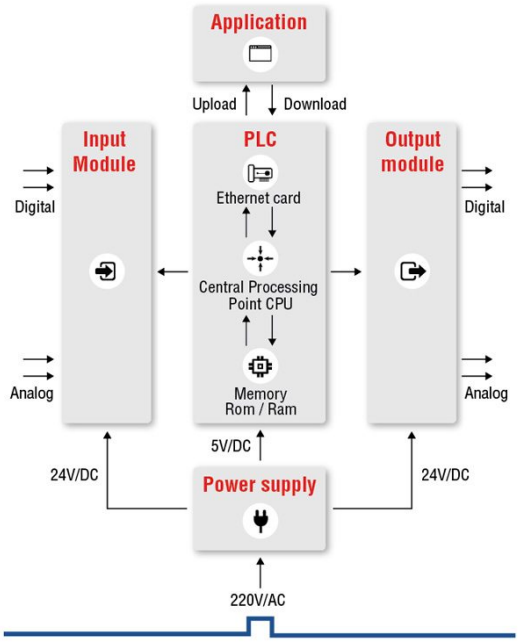
Sterownik PLC uzyskuje informacje z zamontowanych czujników i innych urządzeń wejściowych. Uzyskane z nich dane są przetwarzane, a następnie uruchamiane są wyjścia w oparciu o ustawione parametry. Schemat działania sterownika PLC został przedstawiony na rysunku 2.1.

Sterownik PLC ma możliwość monitorowania i rejestrowania danych czasu pracy, efektywności maszyny, temperatury pracy itp.. Może automatycznie uruchamiać i zatrzymywać proces, a także generować powiadomień w postaci alarmów w przypadku, gdy maszyna działa nieprawidłowo lub wykrywa jakieś problemy. Programowalny sterownik logiczny jest bezkonfliktowym i niezawodnym rozwiązaniem do sterownia różnymi procesami. Może być wykorzystywanym w wielu aplikacjach.

2.1 Zastosowania

Systemy automatyki zbudowane na podstawie sterowników PLC, cechują się modułową budową, wysoką sprawnością i niewielkimi rozmiarami. Wykorzystywane są w różnych typach aplikacji, które wymagają zrealizowania zarówno prostych, jak i rozbudowanych algorytmów logicznych, w szczególności z nastawieniem na bezpieczeństwo i skuteczność systemu.

Sterowniki PLC już od kilkunastu lat stosowane są w wielu dziedzinach codziennego życia. Cenione są przede wszystkim za mnóstwo posiadanych funkcji takich jak: zbieranie, udostępnianie i diagnostyka danych. Programowalne sterowniki wykorzystywane są w takich gałęziach przemysłu jak: przemysł

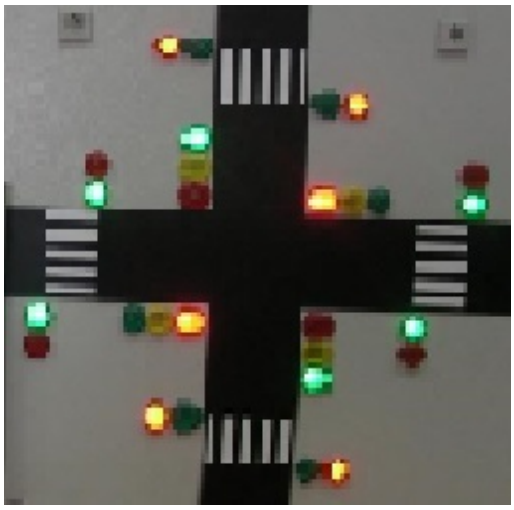


Rysunek 2.1: Schemat działania sterownika PLC.

maszynowy, przemysł chemiczny, przemysł spożywczy, energetyka, transport. Sterowniki PLC są stale doskonalone pod kątem funkcjonalności. Dzięki znakomitym możliwościom komunikacji, urządzenia te stają się łatwe w obsłudze jak i konfiguracji, czego efektem jest dalszy rozwój przemysłu. Sterownik PLC ma możliwość kontroli każdego przebiegu przemysłowego. Dzięki szerokiemu i stale powiększającemu się przekrojowi zastosowań sterownik PLC może zostać użyty w miejscach gdzie produkcja mogłaby być zautomatyzowana.

Jednym z przykładów użycia sterownika PLC jest sygnalizacja świetlna zastosowana w transporcie drogowym. W modelu sygnalizacji świetlnej skrzyżowania dróg, zaprojektowano dwa sygnalizatory świetlne na dwóch drogach, które były przeznaczone dla pojazdów, a także cztery sygnalizatory na przejściach dla pieszych. Schemat na rysunku 2.2 przedstawia sytuację, którą spotykamy na co dzień.

Następnym przykładem zastosowania automatyki przemysłowej z użyciem sterownika PLC jest oczyszczalnia ścieków. Zadaniem systemu automatyki jest przede wszystkim zarządzanie urządzeniami mechanicznego oczyszczania ścieków: przenośniki, piaskownik, krata, sito, praso płuczka. W ten sposób cała praca jest w pełni zautomatyzowana. Ponadto sterownik PLC ułatwia operatorom zadawanie parametrów technologicznych.



Rysunek 2.2: Sygnalizacja świetlna.

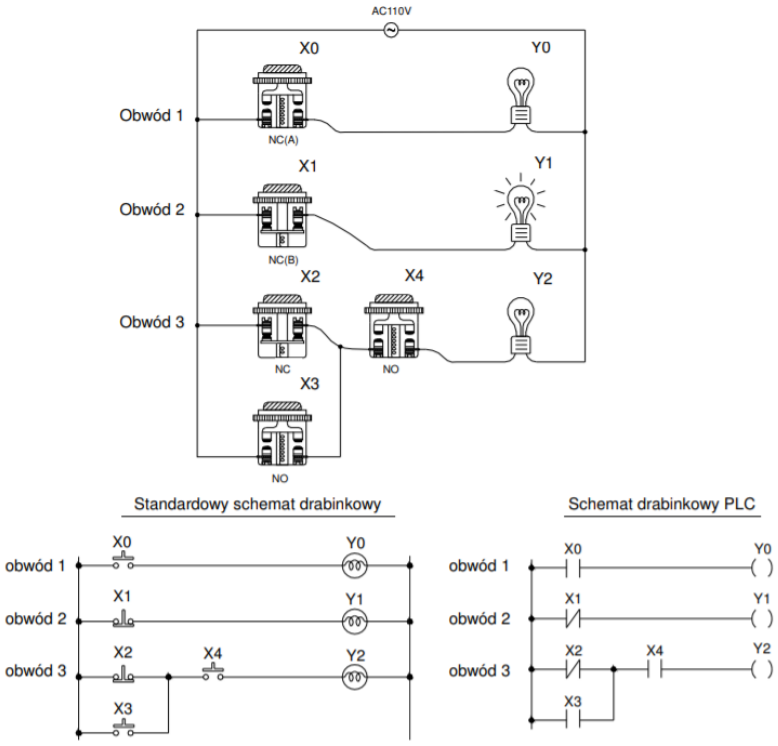
2.2 Programowanie drabinkowe

Mówiąc sterownikach PLC nie można pominąć programowania drabinkowego (ang. ladder programming). Należy je sobie wyobrażać jako rodzaj języka graficznego, który jest wykorzystywany w automatycznych systemach sterowania. Język drabinkowy składa się z takich elementów jak:

- styk A normalnie otwarty (domyślnie jest przerwa między zaciskami),
- styk B normalnie zwarty (domyślnie zaciski są zwarte),
- zegar,
- licznik,
- cewka wyjściowa.

Po pojawieniu się sterownika PLC opartego na mikroprocesorze pojawił się dodatkowo styk różnicowy. Można wyróżnić dwa systemy logiczne dostępne dla programowania drabinkowego: logikę sekwencyjną oraz kombinacyjną. W logice kombinacyjnej występuje obwód łączący jeden lub więcej elementów wejściowych w sposób równoległy lub szeregowy. Wyniki wysyłane są do elementów wyjściowych.

Rysunek 2.3: Oczyszczalnia ścieków.

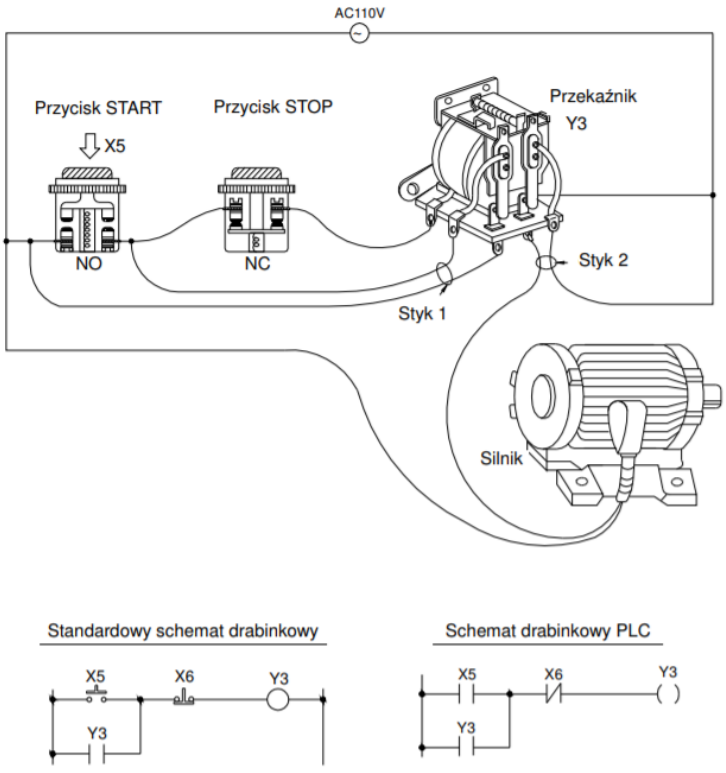


Rysunek 2.4: Schemat drabinkowy z wykorzystaniem logiki kombinacyjnej [6].

Na schemacie 2.4 ukazana jest logika kombinacyjna przy wykorzystaniu następujących trzech schematów: rzeczywistego połączenia, standardowego schematu drabinkowego i schematu drabinkowego PLC. W obwodzie pierwszym

widoczny jest rozwarty przełącznik, gdzie normalnie styk przełącznika i wskaźnik są wyłączone. Po naciśnięciu przełącznika stan stylu styk zmianie się na załączony i wskaźnik załączą się. W następnym obwodzie wykorzystywany jest przełącznik normalnie zamknięty. Zatem styk przełącznika i wskaźnik normalnie są załączone. Po wciśnięciu przełącznika stan styku i wskaźnika zmienia się na wyłączony. W trzecim obwodzie występują więcej niż jeden element wejściowy. Wskaźnik wyjściowy `Y2` włączy się, jeśli styk `X2` zostanie otwarty albo styk `X3` zostanie zamknięty przy równoczesnym włączeniu styku `X4`.

Jeśli chodzi o logikę sekwencyjną to mowa jest o obwodzie sterującym ze sprzężeniem zwrotnym (otrzymywanie przez układ informacji o własnym działaniu, czyli o wartości wyjściowej). Oznacza to, że na wyjściu obwodu następuje proces sprzężenia zwrotnego do wejścia tego obwodu. Stan wyjścia nie zmienia się, nawet w przypadku zmiany na pozycję początkową. Proces taki pokazuje schemat 2.5 obwodu sterownika silnika z dwoma przyciskami: start `X5` oraz stop `X6`.



Rysunek 2.5: Schemat drabinkowy z wykorzystaniem logiki sekwencyjnej [6].

Po podłaczeniu obwodu do źródła zasilania, przełącznik `X6` pozostanie załączony zaś `X5` wyłączony. Przekaźnik `Y3` również jest wyłączony. Styki wyj-

ściowe 1 i 2 są wyłączone ze względu na to, że należą do styku A, czyli takiego, który włącza się przy włączeniu przekaźnika. Silnik z tego powodu nie będzie pracował. Po kliknięciu przycisku start `X5`, styki wyjściowe zostaną włączone i silnik zacznie pracować. Jeśli włączony jest przekaźnik `Y3` oraz nastąpi wyłączenie przycisku `X5`, przekaźnik pozostanie w dotychczasowym stanie dzięki sprzężeniu zwrotnemu ze styku 1. Taka operacja nosi nazwę *obwód z podtrzymaniem*. W tabeli 2.6 przedstawione są procesy przełączania w opisanym przykładzie.

	Przycisk X5 (NO)	PrzyciskX6 (NC)	Stan silnika (przekaźnika)
①	Zwolniony	Zwolniony	WYŁ
↓			
②	Naciśnięty	Zwolniony	WŁ
↓			
③	Zwolniony	Zwolniony	WŁ
↓			
④	Zwolniony	Naciśnięty	WYŁ
↓			
⑤	Zwolniony	Zwolniony	WYŁ

Rysunek 2.6: Procesy przełączania w obwodzie z podtrzymaniem [6].

Zauważmy, że na różnych etapach sekwencyjnych, wyniki mogą się różnić od siebie pomimo tych samych stanów wejściowych. Na etapie 1 i 3 oba przyciski są rozłączone, ale na etapie 1 silnik nie pracuje, a na etapie 3 silnik pracuje. Sterowanie sekwencyjne ze sprzężeniem zwrotnym z wyjścia na wejście można uważać za unikalny opis obwodu z podtrzymaniem schematu drabinkowego.

2.3 Automatyka w domu mieszkalnym

Zaprojektowane z myślą wyłącznie o przemyśle, sterowniki PLC od kilku lat obecne są w domach i mieszkaniach. Głównym zastosowaniem sterowników PLC w domach mieszkalnych jest kontrola i zarządzanie urządzeniami znajdującymi się tam. Najprostszym rozwiązaniem jest sterowanie poprzez wciśnięcie przycisku lub aktywacja głosem. Bardziej zaawansowane systemy analizują dane wejściowe i na ich podstawie podejmują różne decyzje. Na przykład na podstawie aktualnej temperatury i wilgotności powietrza w pomieszczeniu, na

zewnątrz domu, temperatury pieca, ale także pory dnia i dnia tygodnia, PLC załącza lub wyłącza ogrzewanie. Można tutaj także uwzględnić upodobania użytkowników. Rano podłoga w łazience jest bardziej podgrzewana niż po południu.

Są to dość naturalne zastosowania sterowników PLC. Tam gdzie kiedyś ręcznie regulowaliśmy urządzenia, teraz zadanie to przejmuje automat. Powoli pojawiają się jednak jeszcze bardziej zaawansowane funkcje sterowników PLC w domach aby zapewnić komfort i bezpieczeństwo użytkowników. Dodatkowe czujniki i detektory pozwalają efektywnie wykorzystywać energię elektryczną i obniżać koszty eksploatacji. Wykrywają niebezpieczne substancje w wodzie i powietrzu dbając w ten sposób o bezpieczeństwo. Kontrolują napięcie w gniazdkach elektrycznych, wykrywają problemy w sprzęcie AGD oraz infrastrukturze sprawdzając ciśnienie gazu i wody w rurach (patrz [5]).

O ile dzisiaj sterownik PLC w domu czy mieszkaniu to pewna ekstrawagancja, to za jakiś czas należy się spodziewać, że stanie się normą, jednym z podstawowych elementów wyposażenia jak czajnik czy odkurzacz. Tak jak kiedyś komputery trafiły pod strzechy z centrów obliczeniowych i ośrodków naukowych, tak w niedalekiej przyszłości pod strzechy trafią sterowniki PLC. Tak więc inteligentny dom to temat bardzo aktualny.

Rozdział 3

Implementacja interfejsu

3.1 Integracja ze sterownikiem PLC

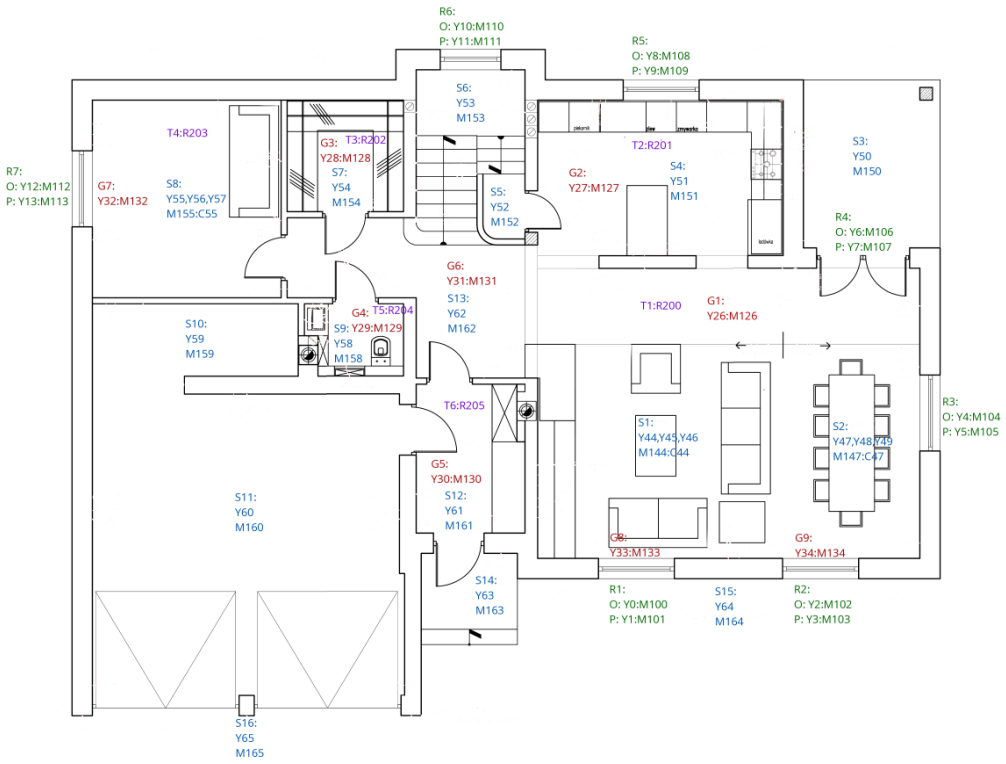
Jak każdy interfejs użytkowy, także i ten jest pewnym kompromisem pomiędzy oczekiwaniami użytkowników, a możliwościami systemu znajdującego się na jego zapleczu.

Na zapleczu naszego interfejsu znajduje się sterownik PLC odpowiadający za działanie wielu różnych urządzeń w domu: oświetlenia, ogrzewania oraz rolet okiennych. Jest to oczywiście tylko mała próbka tego, co do takiego sterownika można w domu podłączyć. Podłączenie na przykład systemu alarmowego daje nie tyle możliwość jego sterowania, ale korzystania z czujników ruchu, aby zapalać światło w pomieszczeniach. Do PLC można podłączyć sprzęt audio-video oraz AGD. W naszym projekcie ograniczamy się do bardzo podstawowych urządzeń, ale staramy się tworzyć go tak, aby można go było łatwo rozbudowywać o dodatkowe funkcje i urządzenia.

Implementacja interfejsu użytkownika ściśle związana jest danym systemem. Nie może zapewniać funkcjonalności, których ten system nie jest w stanie zrealizować. Z drugiej strony, interfejs może nie pozwalać zrealizować tego, co potrafi system, ale to zdecydowanie słaby punkt takiego interfejsu. W przypadku naszego interfejsu mocno jesteśmy zależni od programu działającego na sterowniku PLC. O tyle mamy ułatwione zadanie, że sami możemy ten program dowolnie modyfikować. Jest to niemal idealna sytuacje, gdzie w zasadzie jesteśmy ograniczeni tylko naszymi kompetencjami i kreatywnością.

Program PLC to, w uproszczeniu, zestaw rejestrów powiązanych ze sobą logicznymi zależnościami. Wykonanie jakichkolwiek operacji na urządzeniach podłączonych do sterownika PLC, sprowadza się do modyfikacji wartości odpowiednich rejestrów, czyli jakby zmiennych w klasycznym programie. Przy planowaniu interfejsu, a jeszcze bardziej samego programu PLC, bardzo ważne jest rozplanowanie, które rejestry mają za co odpowiadać. Ponieważ rejestry nie mają nazw jak zmienne w programie, tylko numeryczne adresy, ważne jest by te adresy były przypisane urządzeniom w jakiś w miarę regularny spo-

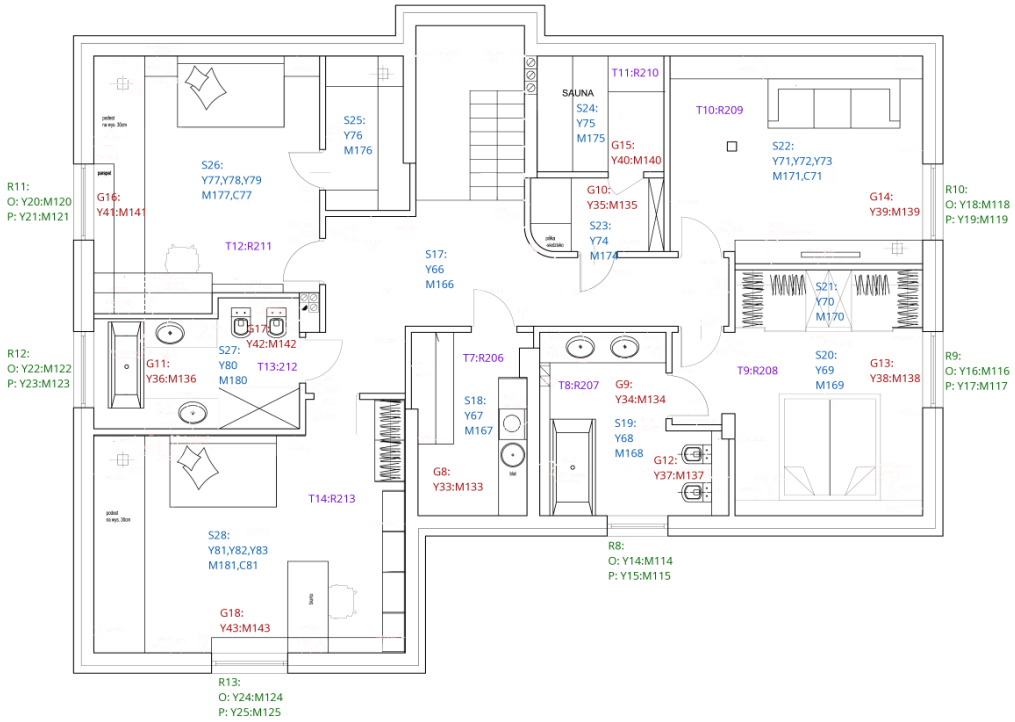
sób, pozwalający na rozbudowę i dodawanie nowych urządzeń i tak, aby sami autorzy systemu się w nim nie pogubili.



Rysunek 3.1: Plan rejestrów zastosowanych na parterze domu.

Pracę nad interfejsem i programem PLC zaczęliśmy od zaplanowania ile i jakich rejestrów będziemy potrzebować do sterowania urządzeniami. Kierowaliśmy się oszczędnością, aby nie marnować zbyt wielu cennych rejestrów do uzyskania danego efektu. W sterowniku PLC ilość rejestrów jest ograniczona. Często robione były uproszczenia byleby nie stracić na funkcjonalności. Na rysunkach 3.1 oraz 3.2 przedstawione są rejestry sterujące różnymi urządzeniami naniesione na plan domu. Taki sposób rezerwacji rejestrów przemawia do wyobraźni i ułatwia kojarzenie rejestrów z konkretnymi urządzeniami.

Druga ważna kwestia na etapie planowania i integracji ze sterownikiem PLC jest sposób w jaki zmiany wartości rejestrów wywołują odpowiednie zachowanie urządzeń. Można na przykład przypisać rejestr do żarówki tak, że świeci ona tak długo, jak wartość tego rejestru wynosi 1. Z punktu widzenia programu realizującego interfejs nie jest to problem, raczej kwestia umowy. Nie jest to jednak rozwiązanie oszczędne. W przypadku lampy potrójnej, gdzie mamy 3 żarówki i możemy zapalić jedną, dwie lub trzy normalnie potrzebny jest włącznik podwójny, jeden przycisk załącza jedną żarówkę, drugi dwie kolejne, a wciśnięcie dwóch przycisków powoduje załączenie wszystkich trzech żaró-



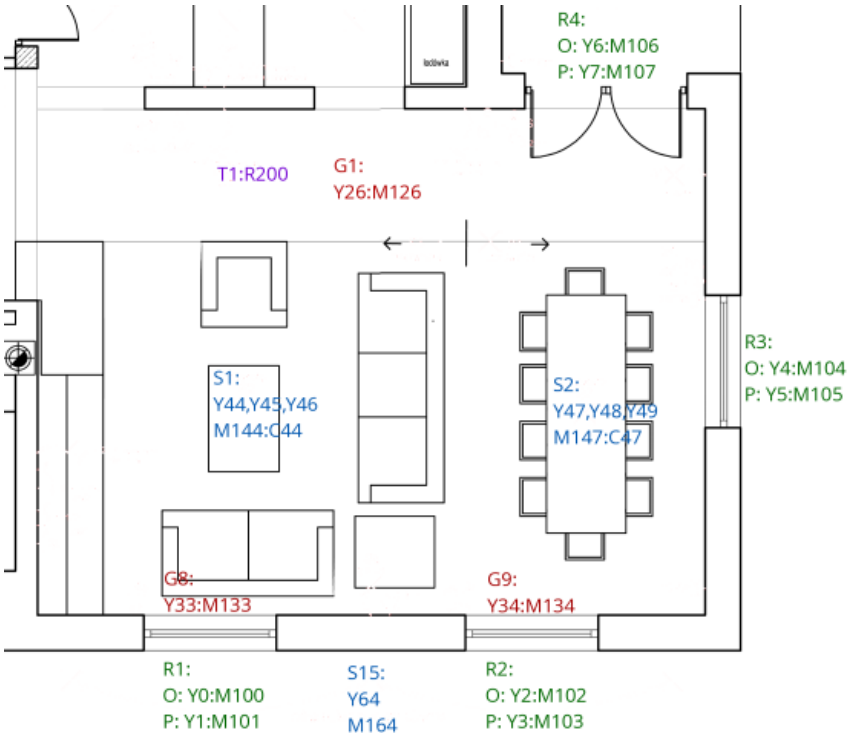
Rysunek 3.2: Plan rejestrów zastosowanych na piętrze domu.

wek. Taki włącznik odpowiada dwóm rejestrom, nie trzem. Można jednak to samo zrealizować jeszcze prościej, jednym przyciskiem dzwonkowym i jednym rejestrem plus licznik. Załączenie przycisku lub chwilowe ustawienie rejestru na 1 powoduje zwiększenie licznika o 1 i zapalenie kolejnej żarówki w lampie. Zauważmy, że nie jesteśmy tu ograniczeni do ilości żarówek w lampie. Taki przycisk można oprogramować również tak, że jego dłuższe przyciśnięcie powoduje albo zupełne zgaszenie światła, albo rozświetlenie wszystkich żarówek. Warto przy tej okazji zwrócić uwagę na to, ile możliwości daje zastosowanie sterownika PLC do tak banalnej operacji jak zapalanie światła.

Opisany tutaj sposób załączania urządzenia poprzez chwilowe ustawienie rejestru na 1, a gdy jest ono załączone, ponowne chwilowe ustawienie na 1 powoduje wyłączenie tego urządzenia nazywa się w żargonie elektryków *flip-flop*. Jest to typowe rozwiązanie stosowane w różnorakich interfejsach graficznych – jeden element wizualny po kliknięciu jest aktywowany, a po drugim kliknięciu jest deaktywowany. Dlatego zdecydowaliśmy, że wszystkie urządzenia jakie mamy będą załączany za pomocą rejestrów flip-flop. W praktyce, w programie PLC rejestr flip-flop wymaga, poza rejestrem wyjścia, 2 rejestrów pomocniczych, co oznacza, że nic nie oszczędzamy, a gorzej, marnujemy cenne rejestry. W sterownikach Fatek jest na szczęście funkcja `TOGG` realizujcą flip-flop.

3.2 Elementy interfejsu

Ponieważ ograniczyliśmy się w naszym programie PLC do sterowania wyłącznie oświetleniem, ogrzewaniem i roletami okiennymi, więc jest tylko kilka elementów interfejsu sterujących urządzeniami. Elementy te będziemy nazywać *kontrolkami*. Będę one powtarzać się w poszczególnych pomieszczeniach. Wydaje się, że rozmieszczenie kontrolek na planie domu będzie najbardziej intuicyjne i pozwoli łatwo je lokalizować. Najbardziej reprezentatywnym pomieszczeniem jest salon na parterze domu (rysunek 3.3), dlatego, że znajdują się tam urządzenia wszystkich możliwych typów. Na jego przykładzie omówimy przydział rejestrów oraz zasady sterowania urządzeniami.



Rysunek 3.3: Plan rejestrów zastosowanych w salonie.

3.2.1 Oświetlenie

Możliwe są pojedyncze lub potrójne punkty świetlne. Ilość wyjść PLC przypisanych do lampy zależy od ilość punktów świetlnych, żarówek, świetlówek lub lamp LED, w tej lampie.

Lampa pojedyncza opisana jest następująco:

```
S15:
Y64
M164
```

Etykieta S4 to identyfikator punktu świetlnego. Jest używana w komentarzach do rejestrów związanych z tym punktem świetlnym w programie PLC. Rejestr `Y` to wyjście sterownika połączone poprzez przekaźnik z odpowiednią lampą. Rejestr `M` działa na zasadzie flip-flop i steruje załączaniem i wyłączaniem lampy. Z lampą skojarzone jest też wejście `X`połączone z przyciskiem na ścianie pomieszczenia. Wartości liczbowe podane przy rejestrach to ich adresy w sterowniku.

Lampa potrójna opisana jest następująco:

```
S1:
Y44, Y45, Y46
M144 : C44
```

Mamy tutaj etykietę S1, trzy wyjścia `Y` każde połączone poprzez przekaźnik z poszczególnymi punktami świetlnymi w lampie. Licznik `C` zawsze zawiera ilość załączonych punktów świetlnych. Wartość 0 oznacza, że lampa jest wyłączona. Rejestr `M` steruje lampą na zasadzie flip-flop. Wielokrotne załączenie rejestru `M` powoduje załączanie kolejnych punktów świetlnych. Czwarte załączenie `M` powoduje wyłączenie lampy. Dłuższe przytrzymanie przycisku, co przekłada się na dłuższe załączenie rejestru `X`, powoduje całkowite wygaszenie lampy.

3.2.2 Ogrzewanie

W domu ogrzewanie realizowane jest za pomocą wężownic podłogowych oraz kaloryferów wiszących. Pomiędzy każdą wężownica jak i kaloryferem, a piecem znajduje się zawór elektrozawór kulowy. Załączenie zasilania w takim zaworze powoduje jego otwarcie. Pełne otwarcie zaworu trwa ok. 6 minut. Wyłączenie zasilania powoduje zamknięcie zaworu, na co potrzeba tyle samo czasu. Gwałtowne otwarcie lub zamknięcie zaworu powodowałoby skokową zmianę ciśnienia w obwodzie, co byłoby odbierane przez domowników jako stuki w rurach. Poza tym wpływa to niekorzystnie na obwody grzejne.

Ponieważ wszystkie elektrozawory są takie same, z punktu widzenia sterownika PLC nie ma znaczenia, czy steruje podłogówką, czy kaloryferem.

Grzejnik opisany jest następująco:

```
G1:
Y26 : M126
```

Do identyfikacji grzejnika służy etykieta G1. Wyjście `Y` połączone z elektrozaworem odpowiada za załączenie i rozłączenie zasilania. Rejestr `M` steruje zaworem na zasadzie flip-flop. Z uwagi na długi czas otwierania i zamykania w oprogramowaniu zaworów w PLC rejestr `M` jest blokowany na odpowiedni czas. Stan, czy dany zawór jest otwarty, czy zamknięty odczytuje się z wartości `Y`.

W większości pomieszczeń znajdują się termometry połączone ze sterownikiem PLC. Dzięki tym termometrom możemy kontrolować temperaturę w pomieszczeniach w sposób całkowicie automatyczny. Program PLC ma tak kontrolować odpowiednie elektrozawory, aby utrzymywać zadaną temperaturę w pomieszczeniu. Temperatura w danym pomieszczeniu może być ustawiona w zależności od pory dnia i dnia tygodnia. Na przykład w ciągu dnia, gdy nie ma domowników, temperatura może być niższa, ale w weekendy jest już inaczej. Wszystko kwestia zaprogramowania sterownika PLC.

Termometr opisany jest następująco:

```
T1: R200
```

Etykieta T1 identyfikuje termometr. W rejestrze `R` przechowywane jest wskazanie termometru. Ponieważ w PLC możemy przechowywać wyłącznie dane całkowite, wartość temperatury w rejestrze `R` to wskazanie termometru pomnożone przez 10. Daje to nam dokładność rzędu 1/10 stopnia Celsjusza. Z każdym termometrem powiązany jest jeszcze jeden rejestr `R` o adresie większym o 100 od podanego w opisie. W tym konkretnie przykładzie byłby to rejestr `R300`. Ten dodatkowy rejestr przechowuje ustawioną, żądaną temperaturę w danym pomieszczeniu.

3.2.3 Rolety okienne

Napęd rolety okiennej sterowany jest silnikiem, który obraca się w dwie strony: opuszcza lub podnosi roletę. Opuszczanie i podnoszenie załączamy przykładając napięcie do jednego z dwu sterujących przewodów elektrycznych. Dlatego do sterowania rolety potrzebujemy dwóch wyjść PLC. Jedno odpowiada za opuszczanie, drugie za podnoszenie rolety. Mechanizm rolety w naszym przypadku nie przewiduje możliwości sprawdzenia w jakim położeniu jest w danym momencie roleta. Dlatego PLC nie ma informacji zwrotnej pozwalającej odzwierciedlić ten stan w interfejsie inteligentnego domu.

Napęd rolety jest zabezpieczony tak, że po całkowitym opuszczeniu lub podniesieniu następuje rozłączenie zasilania silnika, aby go nie spalić. W oprogramowaniu PLC warto jednak zadbać o to by wyłączyć opuszczanie i podnoszenie, czyli wyzerować odpowiednie wyjście `Y`, po upływie pewnego czasu. Czas ten zależy od długości rolety, czyli od wysokości okna lub drzwi.

Roleta okienna opisana jest następująco:

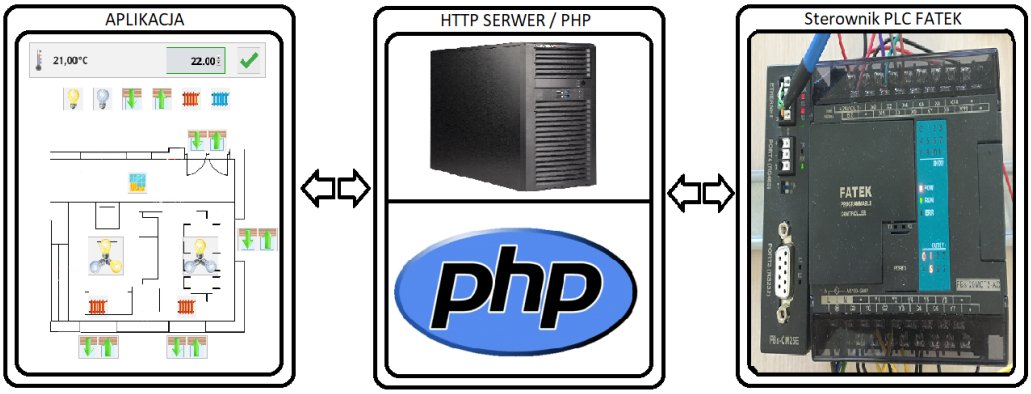
```
R1:
O: Y0 : M100
P: Y1 : M101
```

Etykieta R1 to identyfikator rolety. Litera `O` oznacza opuszczanie, natomiast `P` podnoszenie. Sterowanie napędu rolety połączone jest w wyjściami `Y`. Rejestry `M` typu flip-flop sterują roletami. Ponowne ustawienie rejestru `M` na 1, zanim

roleta zostanie całkowicie opuszczona lub podniesiona, powoduje jej zatrzymanie. Można w ten sposób częściowo tylko odsłonić okno. Przy każdym oknie znajduję podwójny przycisk dzwonkowy połączony z wejściami `X` sterownika.

3.3 Asynchroniczna wymiana danych

Działanie interfejsu inteligentnego domu oparte jest na przesyłaniu wartości rejestrów pomiędzy sterownikiem PLC, a aplikacją webową. Dzięki bibliotece `fatek.php`, powstałej równolegle z naszym interfejsem w ramach pracy [2], możemy łatwo odczytywać i zapisywać wartości rejestrów PLC wykorzystując język skryptowy PHP na serwerze HTTP. Serwer HTTP to warstwa pośrednicząca pomiędzy warstwą aplikacji a warstwą sterownika PLC jak to ukazana na rysunku 3.4.



Rysunek 3.4: Trzy warstwy systemu inteligentny dom.

Przeładowywanie strony, aby odczytać wartości rejestrów PLC i zaprezentować stan urządzeń w domu jest bardzo nieefektywne. Stan urządzeń nie zmienia się aż tak często, ale wypada to sprawdzać w dość małych odstępach czasu, aby interfejs odzwierciedlał aktualny stan urządzeń. Nie do przyjęcia byłoby na przykład, że aplikacja pokazuje zapalenie światła w kuchni z opóźnieniem 10 minut. Nawet 1 minuta to zbyt długi czas. Przeładowywanie całej strony tak często jest wręcz absurdalne. Dlatego w naszym projekcie zdecydowaliśmy się korzystać z asynchronicznej komunikacji aplikacji z serwerem HTTP zapewnianej przez technologię AJAX w JavaScript. Asynchroniczność sprowadza się do tego, że poszczególne elementy interfejsu mogą komunikować się z serwerem HTTP niezależnie od siebie, w tym samym czasie. Przesyłane są w ten sposób małe porcje danych i modyfikowane są tylko niektóre elementy interfejsu, tak aby ukazać aktualny stan urządzeń domowych.

Eksperymentalnie ustaliliśmy, że odświeżanie widoku powinno odbywać się w odstępach 2 – 3 sekundowych. Zbyt częste odświeżanie powoduje duże

obciążenie serwera i przeglądarki, a wcale nie poprawia reaktywności aplikacji. Z kolej zbyt duże przerwy w odświeżaniu powoduje, że aplikacja staje się mało reaktywna, po kliknięciu kontrolki trzeba czekać na reakcję. Bardziej nerwowi użytkownicy będą wtedy klikać ponownie i zamiast na przykład zapalić światło będą je gasić. Ostatecznie odświeżanie interfejsu jest ustawione na 3 sekundy, ale można to łatwo zmienić.

Druga bardzo ważna kwestia przy komunikacji pomiędzy aplikacją a sterownikiem PLC to ilość przesyłanych danych w jednym zapytaniu. Jeśli mamy na jednym widoku 30 kontrolek, każda z nich powiązana jest choćby tylko z jednym rejestrem i każdy rejestr odczytywany jest w osobnym zapytaniu do serwera, to na jedno odświeżenie przypada 30 zapytań pomiędzy aplikacją a serwerem HTTP. Jeśli korzystamy z aplikacji na smartfonie na słabym łączu to zanim skończy się jeden cykl odświeżania, zacznie się kolejny. Dlatego przyjęliśmy zasadę, że niezależnie od ilości rejestrów składających się na jeden widok, odświeżamy go za pomocą jednego tylko zapytania do serwera. Jest to duże ograniczenie podczas projektowania komunikacji w aplikacji, ale minimalizujemy ilość połączeń z serwerem oraz ilość przesyłanych danych.

Odświeżanie do tylko odczyt danych ze sterownika PLC. Trochę inaczej wygląda kwestia zapisu do sterownika. Zapis konieczny jest wówczas, gdy użytkownik uruchamia kontrolkę w aplikacji, na przykład, zapala światło w pokoju, opuszcza roletę albo ustawia temperaturę. W zdecydowanej większości wypadków zapisywana jest wówczas wartość tylko jednego rejestru. W naszym interfejsie tylko przy ustawianiu temperatury zapisywane są dwa rejestry: jeden to docelowa wartość temperatury, a drugi to uruchomienie automatycznej regulacji temperatury. Tak więc, przy zapisie problem ilości i częstotliwości przesyłanych danych nie jest taki poważny jak przy odświeżaniu.

3.4 Kod aplikacji

Aplikacja inteligentny dom została napisana w HTML, PHP, JavaScript i CSS. Najważniejsza jej część odpowiadająca za odświeżanie widoku to oczywiście JavaScript i AJAX. PHP jest używane do komunikacji ze sterownikiem PLC, do podłączenia z bazą danych, gdzie są hasła użytkowników oraz do obsługi sesji użytkowników. PHP pomaga również ujednolicić wspólne, powtarzalne części kodu aplikacji, na przykład nagłówek i stopkę stron HTML poszczególnych pomieszczeń. Nie korzystamy z żadnych frameworków.

3.4.1 Kod HTML

Najważniejszą częścią aplikacji są widoki pomieszczeń. W zależności od pomieszczenia w domu, w jego widoku może dodatkowo być panel ustawiania temperatury oraz panel sterowania grupami urządzeń, gdy jest ich więcej. Specyfikacja HTML i sposób działania przeglądarek (przynajmniej przeglądarki

Firefox) pozwalają na dodawanie w kodzie HTML własnych znaczników i atrybutów. Wykorzystujemy to w naszej aplikacji. Urządzenia reprezentowane są jako element `<div>`. Oświetlenie i ogrzewanie mają następujące atrybuty:

`switch` określa adres rejestru PLC odpowiadający za przełączanie urządzenia (załącz, wyłącz),

`state` określa adres rejestru PLC wskazujący stan urządzenia (ile punktów świetlnych jest zapalonych, czy urządzenie jest załączone, czy wyłączone).

Rolety mają dwa atrybuty `down` oraz `up`, które określają adresy rejestrów odpowiadających za opuszczanie i podnoszenie. W panelu sterowania temperaturą mamy atrybuty:

`gauge` określa adres rejestru PLC zawierającego aktualną temperaturę w pomieszczeniu,

`preset` określa adres rejestru PLC zawierającego temperaturę zadaną przez użytkownika,

`automatic` określa adres rejestru PLC, który włącza automatyczne sterowanie temperaturą.

W panelu grup urządzeń są tylko atrybuty `switch` określające adresy rejestrów PLC, które sterują tymi grupami.

Poniżej znajduje się kod HTML widoku salonu.

```
<div id="Salon" class="Room">
  <div id="T1" class="Control␣Temperature" gauge="R200"
      preset="R300" automatic="M300"></div>

  <ul class="Dashboard">
    <li><div id="C1" class="Control␣Button␣LightOn" switch="M0"></div></li>
    <li><div id="C2" class="Control␣Button␣LightOff" switch="M1"></div></li>
    <li><div id="C3" class="Control␣Button␣BlindsDown" switch="M2"></div></li>
    <li><div id="C4" class="Control␣Button␣BlindsUp" switch="M3"></div></li>
    <li><div id="C5" class="Control␣Button␣HeatOn" switch="M4"></div></li>
    <li><div id="C6" class="Control␣Button␣HeatOff" switch="M5"></div></li>
  </ul>

  <div id="S1" class="Control␣Light3" switch="M144" state="C44"></div>
  <div id="S2" class="Control␣Light3" switch="M147" state="C47"></div>

  <div id="G1" class="Control␣HeatingFloor" switch="M126" state="Y5"></div>

  <div id="G8" class="Control␣HeatingWall" switch="M133" state="Y33"></div>
  <div id="G9" class="Control␣HeatingWall" switch="M134" state="Y34"></div>

  <div id="R1" class="Control␣Blinds" down="M100" up="M101"></div>
  <div id="R2" class="Control␣Blinds" down="M102" up="M103"></div>
  <div id="R3" class="Control␣Blinds" down="M104" up="M105"></div>
  <div id="R4" class="Control␣Blinds" down="M106" up="M107"></div>
</div>
```

Niektóre z elementów jak panel temperatury, czy rolety po załadowaniu strony są uzupełniane dodatkowymi elementami przez kod w JavaScript. Pojawiają się wtedy nowe elementy z nowymi atrybutami.

```
<div id="T1" class="Control␣Temperature" automatic="M300" preset="R300"
        gauge="R200">
  <div class="Value"></div>
  <input class="Text" type="number" name="R300" autocomplete="on"
        min="0" max="30" step="0.1">
  <input class="Button" type="button">
</div>

<div id="R4" class="Control␣Blinds" up="M107" down="M106">
  <div class="Down"></div>
  <div class="Up"></div>
</div>
```

3.4.2 Kod CSS

Zgodnie ze swoim przeznaczeniem CSS odpowiada za wygląd elementów HTML w aplikacji. Pomieszczenia są elementami `<div>` klasy `Room`.

```
.Room {
    position: relative;
    width: 700px;
    height: 800px;
    margin: auto;
    background-repeat: no-repeat;
    background-position: 50% 0;
}
```

Kontrolki to elemeny `<div>` klasy `Control`.

```
.Control {
    position: relative;
    border: 1px solid rgba(128, 128, 128, 0.5);
    background-color: rgba(128, 128, 128, 0.1);
    background-position: 50% 50%;
    background-repeat: no-repeat;
}

.Control:hover {
    border: 1px solid rgba(128, 128, 128, 0.6);
    background-color: rgba(128, 128, 128, 0.2);
}
```

Każda konkretna kontrolka ma określoną swoją klasę, na przykład dla lamp potrójnych to `Light3`.

```
.Light3 {
    width: 90px;
    height: 85px;
    background-image: url('../assets/icon-light3-0.png');
}
```

Kontrolki rolet są trochę bardziej skomplikowane bo jedna kontrolka dzieli się na dwie: jedną do opuszczania drugą do podnoszenia. Kontrolki rolet są klasy `Blinds`. JavaScript odpowiada za utworzenie potomnych elementów odpowiadających przyciskom: opuść klasy `Down` oraz podnieś klasy `Up`.

```
.Blinds {
    width: 102px;
    height: 52px
}

.Blinds .Down {
    float: left;
    width: 50px;
    height: 50px;
    background-image: url('../assets/icon-blinds-down-0.png');
    background-position: 50% 50%;
    background-repeat: no-repeat;
}

.Blinds .Down:hover {
    background-image: url('../assets/icon-blinds-down-1.png');
}

.Blinds .Up {
    float: left;
    width: 50px;
    height: 50px;
    background-image: url('../assets/icon-blinds-up-0.png');
    background-position: 50% 50%;
    background-repeat: no-repeat;
}

.Blinds .Up:hover {
    background-image: url('../assets/icon-blinds-up-1.png');
}
```

Powyższy kod CSS jest wspólny dla wszystkich widoków. Poszczególne pomieszczenia różnią się jednak tłem, gdzie jest kontur pomieszczenia oraz rozmieszczeniem kontrolek. Poniżej fragment kodu CSS dla salonu.

```
#Salon {
    background-image: url('../images/d-salon.png');
}

#S1 {
    top: 260px;
    left: 200px;
}

#S2 {
    top: 175px;
    left: 440px;
}
```

3.4.3 Kod JavaScript

Dzięki ECMAScrpit można w JavaScript pisać program w klasycznym, obiektowym stylu. Przy bardziej złożonym projekcie, w którym mamy dość skomplikowaną strukturę danych, programowanie obiektowe daje czytelny kod. Dziedziczenie pozwala konstruować skomplikowane obiekty poprzez stopniowe uzupełnianie własności i metod w zależności od przeznaczenia klasy.

Podstawowa klasa to `Register` reprezentująca rejestr PLC. Jej własności to `type`, `symbol`, `address` oraz `value`. W klasie są następujące metody:

`details()` zwraca sformatowane własności do celów debugowania,

`uodate()` metoda abstrakcyjna, do nadpisania w klasach potomnych, bardziej szczegółowych, wołana w `read()` i `write()`, aby ewentualnie zaktualizować powiązaze z rejestrem inne obiekty,

`read()` odczytuje wartość rejestru ze sterownika PLC,

`write(aValue)` zapisuje wartość `aValue` w sterowniku PLC.

Jej definicja znajduje się w skrypcie `registers.js`. Prototypy klas i funkcji z tego skryptu są poniżej.

```
const register_cmd = 'register_cmd.php';
const registers_cmd = 'registers_cmd.php';

var registers = [];

class Register {
    constructor(aType, aSymbol, aAddress, aValue)

    get details()
    update()
    read()
    write(aValue)
}

function fatekRegistersGet(aRegisters, aCallback)
function fatekRegistersSet(aRegisters, aCallback)
```

W zmiennej globalnej `registers` znajduje się lista rejestrów dla danego widoku. Funkcje `fatekRegistersGet()` oraz `fatekRegistersSet()` służą do odczytu i zapisu wielu rejestrów jednocześnie. Argument `aCallback` w tych funkcjach to opcjonalna funkcja wołana po zakończeniu komunikacji ze sterownikiem PLC.

Drugi ważny skrypt to `control.js`, w którym znajdują się definicje kontrolek oraz funkcja odświeżająca widok. Poniżej znajdują się klasy z tego skryptu.

```
var interval;

var controls = [];
```

```
class Control {
    constructor(aElement)

    get registers() {}
    update() {}
}

class Button extends Control {
    constructor(aElement, aSwitch)

    toggle ()
}

class Toggle extends Button {
    constructor(aElement, aName, aSwitch, aState)

    get registers()
    update ()
}

class Light1 extends Toggle {
    constructor(aElement, aSwitch, aState)
}

class Light3 extends Toggle {
    constructor(aElement, aSwitch, aState)
}

class HeatingFloor extends Toggle {
    constructor(aElement, aSwitch, aState)
}

class HeatingWall extends Toggle {
    constructor(aElement, aSwitch, aState)
}

class Blinds extends Control {
    constructor(aElement, aDown, aUp)

    down()
    up()
}

class Temperature extends Control {
    constructor(aElement, aGauge, aPreset, aAutomatic)

    get registers()
    update ()
    accept ()
}
```

Klasa `Control` jest rodzicem wszystkich kontrolek. Jej konstruktor wiąże

ją z elementem HTML. Metoda `registres()` zwraca tablicę rejestrów, powiązanych z daną kontrolką, wymaganych do odczytu podczas odświeżania. Natomiast metoda `update()` odpowiada za odświeżenie tej kontrolki. Obie metody są abstrakcyjne i powinny być nadpisane w klasach potomnych, bardziej szczegółowych.

Klasa `Button` reprezentuje przycisk, którego naciśnięcie, czyli zdarzenie `click`, powoduje zapisanie wartości 1 do powiązanego rejestru PLC przekazanego jako argument `aSwitch` w konstruktorze.

`Toggle` zachowuje się jak `Button`, ale dodatkowo abstrahuje metody i własności dla kontrolek oświetlenia i ogrzewania. Argument konstruktora `aName` to nazwa identyfikująca plik tła kontrolki. Natomiast `aSwitch` i `aState` to rejestry z odpowiednich atrubutów w HTML danej kontrolki.

Klasy `Light1`, `Light3`, `HeatingFloor` i `HeatingWall` to uszczegółowienie klasy `Toggle`.

Klasa `Blinds` odpowiada kontrolce rolet. W konstruktorze ustawiane są rejestry służące do opuszczania `aDown` i podnoszenia `aUp`. Ich wartości pobierane są z odpowiednich atrybutów HTML. Obiekt `Blinds` w konstruktorze tworzy dwa nowe elementy HTML – przyciski do opuszczania i podnoszenia.

Panel sterowania temperaturą to klasa `Temperature`. Argumenty konstruktora odpowiadają atrybutom HTML tej kontrolki. W konstruktorze tworzone są trzy nowe elementy HTML: `<div>` zawierający aktualną temperaturę, `<input type"number">` pozwalający użytkownikowi wprowadzić nową temperaturę oraz `<input type"button">` przycisk załączający automatyczną regulację temperatury.

W skrypcie `control.js` znajdują się ponadto dwie bardzo ważne funkcje.

```
window.onload = init;

function init() {
    var elements;

    ...

    elements = document.getElementsByClassName('Light3');
    for (var i~= 0; i < elements.length; i++) {
        var e = elements[i];
        control = new Light3(e, e.attributes['switch'].value,
            e.attributes['state'].value)
        controls.push(control);
        registers = registers.concat(control.registers);
    }

    ...

    elements = document.getElementsByClassName('Blinds');
    for (var i = 0; i < elements.length; i++) {
        var e = elements[i];
        control = new Blinds(e, e.attributes['down'].value,
```

```
            e.attributes['up'].value)
        controls.push(control);
    }

    ...

    refresh();

    interval = setInterval(refresh, 3000);
}
```

Funkcja `init()` uruchamiana jest po załadowaniu strony. Skanuje dokument HTML, odszukuje definicje kontrolek i na ich podstawie tworzy odpowiednie obiekty. W tablicy `registers` spisywane są wszystkie rejestry potrzebne do aktualizacji widoku, czyli do aktualizacji wszystkich kontrolek w tym widoku. Tablica wszystkich kontrolek zawartych w widoku znajduje się w zmiennej globalnej `controls`. Na koniec inicjowane jest okresowe odświeżanie strony.

```
function refresh() {
    var request = new XMLHttpRequest();

    request.open('POST', x, true);
    request.setRequestHeader("Content-type",
        "application/x-www-form-urlencoded");
    request.onreadystatechange = function () {
        if (request.readyState == 4 && request.status == 200) {
            var r = JSON.parse(request.responseText);
            for (var i = 0; i < r.length; i++) {
                registers[i].value = r[i].value;
            }
            for (var i = 0; i < controls.length; i++) {
                controls[i].update();
            }
        }
    }
    request.send('c=g&r=' + JSON.stringify(registers));
}
```

Funkcja `refresh()` realizuje odświeżanie widoku. Tablica `registers` przesyłana jest metodą POST w formacie JSON do skryptu PHP na serwerze HTTP. Odpowiedź z serwera HTTP również jest w formacie JSON. W odpowiedzi otrzymywane są wartości rejestrów. Wartości te ustawiane są w rejestrach z tablicy `registers`. W następnej pętli aktualizowane są wszystkie kontrolki na widoku z tablicy `controls`.

3.4.4 Kod PHP

Istotne zastosowanie PHP w aplikacji to dwa skrypty do zapisu i odczytu wartości rejestrów sterownika PLC. Oparte są one na bibliotece `fatek.tex` napisanej w ramach pracy [2].

Poniżej przedstawiamy skrypt `registers_cmd.php` używny do odczytu i zapisu wartości wielu rejestrów sterownika PLC jednocześnie. Korzystamy z niego między innymi podczas odświeżania widoku.

```
<?php

define('ROOT_DIR', '.');

require_once(ROOT_DIR . '/lib/environ.php');
require_once('sanitizer.php');
require_once('fatek.php');


if (! isset($_SESSION['username']))
    exit('ERROR: Brak autoryzacji.');


$opt_command = array(
    'g' => 'fatekRegistersGet',
    's' => 'fatekRegistersSet'
);

$sanitizer = new SanitizerSelect($opt_command);
$command = $sanitizer->sanitize(get_post_data('c'));
if (is_null($command))
    exit('ERROR: Brak wymaganych danych: polecenie.');

$registers = json_decode(get_post_data('r'));

try {
    $fatek = new FatekPLC();
    $fatek->connect();
    switch ($command) {
        case 'g':
            $fatek->get_value(1, $registers);
            break;
        case 's':
            $fatek->set_value(1, $registers);
            break;
        default:
            exit('ERROR: Nieznane polecenie.');
    }
    $fatek->disconnect();
    echo json_encode($registers, JSON_PRETTY_PRINT);
} catch (Exception $e) {
    echo NULL;
}

?>
```

Skrypt `register_cmd.php`, podobny do powyższego, używany jest do odczytu i zapisu pojedynczego rejestru. Uruchamiany jest, gdy klikamy kontrolkę i ustawiamy wartość pojedynczego rejestru sterującego danym urządzeniem.

Rozdział 4

Obsługa interfejsu

4.1 Sterowanie urządzeniami w domu

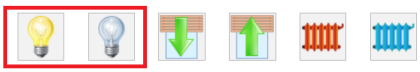
Aplikacja inteligentny dom daje możliwość sterowania oświetleniem, ogrzewaniem i roletami okiennymi. Działanie tych urządzeń opiszemy na przykładzie salonu, rysunek 4.1, gdzie występują wszystkie typy urządzeń.

Rysunek 4.1: Widok salonu w aplikacji inteligentny dom.

4.1.1 Oświetlenie

W oprogramowanym domu znajdują się dwa rodzaje lamp: pojedyncze i potrójne. Pojedyncze symbolizowane są za pomocą ikony pojedynczej żarówki, natomiast potrójne ikoną złożoną z trzech żarówek. Do załączania obu typów lamp używamy jednego stycznika. W aplikacji i programie PLC ten stycznik to rejestr 1-bitowy, na ścianie pomieszczenia natomiast mamy pojedynczy włącznik dzwonkowy.

W przypadku lampy pojedynczej zwarcie styku powoduje załączenie lampy, gdy jest wyłączona i wyłączenie, gdy jest załączona. W przypadku lampy potrójnej, gdy jest ona całkowicie zgaszona, pierwsze zwarcie stycznika zapala pierwszą sekcję. Wówczas, w aplikacji jedna z trzech żarówek na ikonce zapala się. Drugie zwarcie zapala drugą sekcję, a w aplikacji zapala się druga z trzech żarówek. Kolejne, trzecie zwarcie powoduje zapalenie trzeciej sekcji lampy, co w aplikacji pokazane jest poprzez zapalenie trzeciej żarówki na ikonce. Czwarte zwarcie stycznika wygasza wszystkie trzy sekcje lampy. Wtedy gasną wszystkie żarówki na ikonce. Jeśli przytrzymamy wciśnięty przycisk dzwonkowy na ścianie przez 3 sekundy, wówczas lampa zostanie całkowicie zgaszona.



Rysunek 4.2: Panel grupy urządzeń – oświetlenie.

Górny panel grupy urządzeń (rysunek 4.1 oraz 4.2) pozwala jednym przyciskiem załączyć lub wyłączyć wszystkie lampy w danym pomieszczeniu. Jest to wygodne, gdy na przykład, wychodzimy z domu i szybko chcemy zgasić wszystkie światła. Po naciśnięciu ikonki z zapaloną żarówką nastąpi zapalenie wszystkich świateł w danym pomieszczeniu. Nie ma znaczenia w tym momencie jakie lampy były załączone. Zapalone zostaje wszystko. Klikając drugą ikonkę, zgaszonej żarówki, zgasimy całkowicie światło w danym pomieszczeniu.

4.1.2 Rolety

Do sterowania jednej rolety w aplikacji mamy dwa przyciski ze strzałkami: dół, góra. Podobnie na ścianie pomieszczenia w pobliżu rolety mamy dwa przyciski dzwonkowe oznaczone góra/dół. Pojedyncze naciśnięcie odpowiedniego przycisku powoduje całkowite podniesienie lub opuszczenie rolety. Jeśli chcemy zatrzymać opuszczanie, czy też podnoszenie, na przykład, aby odsłonić okno tylko do połowy, należy ten sam przycisk wcisnąć drugi raz. W czasie podnoszenia lub opuszczania rolety drugi z przycisków sterujących jest nieaktywny.

Ikony widoku nie odzwierciedlają rzeczywistego stopnia odsłonięcia okna. Zastosowane mechanizmy rolet nie pozwalają na to. W zależności od wysokości

okna proces całkowitego opuszczania i podnoszenia rolety może trwać krócej lub dłużej. Dla typowego okna zajmuje to koło 20 sekund, a dla drzwi 30 sekund.

Rysunek 4.3: Panel grupy urządzeń – rolety.

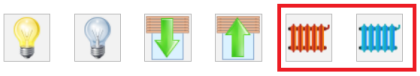
Dla wygody, w górnym panelu grupy urządzeń na widoku pomieszczenia (rysunek 4.1 oraz 4.3) umieszczono dwa przyciski rolet: dół oraz góra. Jednym z nich możemy zasłonić lub odsłonić wszystkie okna w tym pomieszczeniu.

4.1.3 Ogrzewanie

Jeśli chodzi o ogrzewanie, w domu występują dwa rodzaje ogrzewania: podłogowe oraz kaloryfery. Do uruchomienia jednego i drugiego typu grzejników służy pojedynczy stycznik. Stycznikiem tym w programie PLC oraz w aplikacji jest rejestr 1-bitowy. Grzejnikami można sterować wyłącznie za pomocą kontrolek w aplikacji.

W sytuacji, kiedy nastąpi zwarcie stycznika, uruchamiany jest elektrozawór odpowiedniego grzejnika. Przycisk w aplikacji działa na zasadzie flip-flop, gdy zawór jest zamknięty zwarcie powoduje otwieranie tego zaworu i na odwrót. Cały proces trwa około 6 minut.

Ikony z czerwonym kaloryferem i żółtą wężownicą podłogową oznaczają włączone ogrzewanie, czyli otwarte zawory. Natomiast ikony z niebieskim kaloryferem i niebieską wężownicą podłogową oznaczają wyłączone ogrzewanie, zamknięte zawory.



Rysunek 4.4: Panel grupy urządzeń – ogrzewanie.

Jeśli chcemy szybko podnieść temperaturę otwierając wszystkie zawory w pomieszczeniu mamy do dyspozycji przycisk czerwonego kaloryfera w górnym panelu grupy urządzeń (rysunek 4.1 oraz 4.4). Niebieski kaloryfer z kolei wyłączy całkowicie ogrzewanie.

W pomieszczeniach, gdzie są grzejniki jest możliwość automatycznego sterowania temperaturą. Aktualna temperatura wyświetlana jest na górze z lewej strony w panelu sterowania temperaturą (rysunek 4.1 oraz 4.5).

Aby ustawić żądaną temperaturę wpisujemy odpowiednią wartość w okienku zaznaczonym na rysunku 4.5 i zatwierdzamy znajdującym się z prawej

Rysunek 4.5: Tabelka z wpisywaniem temperatury

strony przyciskiem. Jeśli panująca obecnie temperatura w pomieszczeniu jest niższa od żądanej, ogrzewanie będzie załączone na tak długo, aż zostanie osiągnięta oczekiwana temperatura. Potem ogrzewanie jest wyłączane. Gdy temperatura spada poniżej żądanej wartości ponownie ogrzewanie jest załączane. W programie PLC temperatura załączenia ogrzewania to temperatura ustalona pomniejszona o pewną stałą, natomiast temperatura wyłączenia ogrzewania to ustalona temperatura powiększona o pewną stałą. Obie stałe są to dodatkowe, globalne rejestry PLC, które można modyfikować. Dzięki tym stałym uwzględniamy bezwładność układu grzewczego.

Gdy klikniemy którąkolwiek z ikonek ogrzewania, tryb automatycznego sterowania temperaturą jest przerywany, a dany grzejnik zostaje załączony lub wyłączony w zależności od jego aktualnego stanu.

4.2 Parametry programu PLC

Użytkownik może dopasować program uruchomiony na sterowniku PLC do urządzeń domowych oraz swoich upodobań. W tym celu w programie PLC zostało wyróżnianych klika rejestrów, które można modyfikować bez wchodzenia do programu WinLadderPro i bez restartowania sterownika. Modyfikacje parametrów można zrobić w dowolnym momencie za pomocą aplikacji. Odnośnik do formularza z parametrami jest na dole strony głównej.

Rejestr	Opis	Jednostka
R0	Czas na zgaszenie światła	ms
R1	Margines załączenia ogrzewania	°C
R2	Margines wyłączenia ogrzewania	°C
R3	Czas opuszczania rolet - małe okno	s
R4	Czas podnoszenia rolet - małe okno	s
R5	Czas opuszczania rolet - normalne okno	s
R6	Czas podnoszenia rolet - normalne okno	s
R7	Czas opuszczania rolet - drzwi	s
R8	Czas podnoszenia rolet - drzwi	s
R9	Czas otwarcia/zamknięcia zaworu	s

Tabela 4.1: Parametry programu PLC.

W tabeli 4.1 przedstawiamy listę dostępnych parametrów.

4.3 Logowanie i ograniczenie dostępu

Dostęp do aplikacji jest ograniczony i wymaga posiadania konta, aby z niej skorzystać. Nie chcielibyśmy, aby ktoś przypadkowy gasił nam światło albo regulował temperaturę w domu, to oczywiste. System uprawnień w obecnej wersji praktycznie nie istnieje. Każdy zalogowany użytkownik, może wszystko w aplikacji. W kolejnych wersjacch można by nie pozwalać na wykonanie pewnych operacji w aplikacji niektórym użytkownikom, na przykład dzieciom.

Przy próbie wejścia na którąkolwiek stronę aplikacji wykonywana jest autoryzacja dostępu z wykorzystaniem sesji w PHP i ciasteczek w przeglądarce. Jeśli sesja jest nieaktualna użytkownik przekierowywany jest na stronę z formularzem logowania.

Na dole każdej strony aplikacji jest podana informacja o nazwie zalogowanego użytkownika. Jest tam również odnośnik, aby się wylogować.

Spis rysunków

Spis tabel

Bibliografia

[1] Freeman E.T., Robson E., *Rusz głową! Programowanie w JavaScript*, Wydawnictwo Helion, 2014.

[2] Lewandowski G., *Interfejs w języku C i PHP do komunikacji komputera z programowalnym sterownikiem logicznym*, Uniwersytet w Białymstoku, Instytut Informatyki, praca licencjacka, 2019.

[3] Schafer S.M., *HTML, XHTML i CSS: Biblia*, Wydawnictwo Helion, 2011.

[4] `https://www.pickaweb.co.uk/kb/what-is-php/`
Dostęp: kwiecień/maj 2019.

[5] `https://scialert.net/fulltextmobile/?doi=jas.2010.1449.1454`
Dostęp: maj 2019

[6] `http://multiprojekt.pl/`
Dostęp: maj 2019.