



POLITECHNIKA ŚLĄSKA
WYDZIAŁ AUTOMATYKI, ELEKTRONIKI I INFORMATYKI

Praca dyplomowa magisterska

Rozproszony system sterowania w inteligentnym budynku

Autor: Andreas Gerono

Kierujący pracą: dr hab. inż. Robert Czerwiński prof. PŚ

Gliwice, marzec 2020

Spis treści

Spis treści	3
Rozdział 1 Wprowadzenie	5
1.1. Wstęp	5
1.2. Cel i zakres pracy	6
Rozdział 2 Analiza rozwiązań	8
2.1. Przegląd podobnych urządzeń na rynku.....	8
2.2. Analiza rozwiązań komunikacyjnych	12
2.3. Wybór technologii.....	28
2.4. Opis koncepcji.....	31
Rozdział 3 Opis części sprzętowej	33
3.1 Urządzenie bazowe	33
3.2 Urządzenia końcowe	37
Rozdział 4 Opis części programowej	46
4.1 Urządzenie bazowe	46
4.2 Urządzenia końcowe	60
Rozdział 5 Testy oraz badania prototypu systemu	67
5.1 Badania funkcjonalne prototypu.....	67
5.2 Badania zasięgu/opóźnień	69
5.3 Bilans mocy	71
Rozdział 6 Podsumowanie	73
Literatura	75
Dodatki	76
Schematy elektryczne.....	76
Projekty PCB	79

Rozdział 1

Wprowadzenie

1.1. Wstęp

Inteligentny dom (*ang. smart home*) składa się z urządzeń które mogą być ze sobą połączone za pomocą sieci lokalnej lub sieci internet. Umożliwia użytkownikom zdalne kontrolowanie i monitorowanie podłączonych urządzeń domowych z aplikacji, telefonu i innych urządzeń sieciowych niezależnie od tego czy są w domu, czy poza nim. Pozwala to na bardziej wydajne wykorzystanie energii, dodatkową wygodę przy wykonywaniu codziennych czynności oraz zapewnia dodatkowe bezpieczeństwo domu. Technologia *smart home* nie jest jednak pozbawiona wad. Do najpoważniejszych należą fragmentacja platformy oraz brak standaryzacji używanych technologii. Może to prowadzić do sytuacji, w których użytkownik nabiera urządzenie niekompatybilne z aktualnie używanym systemem.

Autor raportu konsumenckiego z 2016 roku [1] stwierdził, że w pierwszej piątce urządzeń najczęściej wymienianych na inteligentne znalazły się:

- systemy sterowania oświetleniem (40%),
- programowalne termostaty (40%) ,
- systemy monitoringu (33%),
- systemy bezpieczeństwa (30%),
- smart lodówki (30%).

Analizy runku przewidują, że do 2021 roku, przeciętny dom w Ameryce Północnej będzie posiadał 13 inteligentnych urządzeń.

1.2. Cel i zakres pracy

Celem pracy było zaprojektowanie i wykonanie systemu sterowania w tzw. inteligentnym budynku. Projekt i wykonanie prototypu systemu miał obejmować urządzenie bazowe oraz urządzenia wykonawcze i pomiarowe. Część projektową i wdrożeniową poprzedziły badania sposobów komunikacji w systemach inteligentnego budynku, protokołów komunikacyjnych z podziałem na ich zastosowanie. Dzięki temu zostały określone wady i zalety oraz wskazania projektowe dla realizowanego systemu.

Zakres pracy obejmował:

- przeprowadzenie analiz sposobów komunikacji i protokołów, wybór oraz uzasadnienie wybranej koncepcji,
- zaprojektowanie systemu: stacji bazowej i urządzeń wykonawczych i pomiarowych,
- realizację oprogramowania,
- wykonanie i uruchomienie prototypu,
- przeprowadzenie testów i badań systemu,
- opracowanie dokumentacji.

Założenia projektowe systemu:

a) Urządzenie bazowe:

- funkcja interfejsu użytkownika na bazie serwera i aplikacji internetowej,
- możliwość zdalnej kontroli urządzeń wykonawczych,
- możliwość podglądu danych z czujników,
- zapewnienie obsługi wielu użytkowników,
- zapewnienie sposobu autoryzacji użytkowników,
- możliwość zarządzania użytkownikami (dodawanie i usuwanie oraz przypisywanie konkretnych urządzeń do użytkowników),
- możliwość łatwego rozszerzania systemu (dodawania i usuwania kolejnych urządzeń wykonawczych i pomiarowych),
- zapewnienie bezproblemowej instalacji systemu w nowym miejscu,
- funkcja wyświetlania stanu łączności urządzeń w systemie.

- b) Urządzenia wykonawcze i pomiarowe:
- możliwość szybkiego i prostego podłączenia z systemem,
 - możliwość sterowania przekaźnikowego urządzeniami podłączonymi do sieci elektrycznej,
 - możliwość sterowania oświetleniem LED, w tym możliwość zmiany barwy oraz intensywności światła,
 - wysyłanie danych z czujników do urządzenia bazowego,
 - funkcja wyświetlania stanu łączności,
 - zasilanie sieciowe.

Rozdział 2

Analiza rozwiązań

2.1. Przegląd podobnych urządzeń na rynku

Liczba oraz rodzaj inteligentnych urządzeń dostępnych na rynku rośnie z roku na rok i nie sposób byłoby opisać je wszystkie, dlatego niniejszy przegląd obejmuje najczęściej wybierane opcje.

a) Przegląd urządzeń bazowych

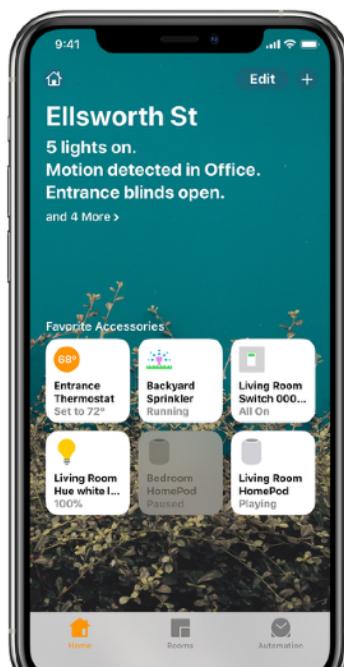
Pierwszym krokiem przy zakupie systemu inteligentnego dla domu, jest wybór urządzenia bazowego oraz ekosystemu. Stacja bazowa (Rys. 2.1.1) łączy wszystkie urządzenia systemu dlatego najważniejszym czynnikiem przy jej wyborze jest kompatybilność (liczba obsługiwanych urządzeń) oraz zakres funkcji które obsługuje. Obecnie na rynku najpopularniejsze są produkty amerykańskich gigantów technologicznych: Amazon oraz Google. Oferują one zaawansowane funkcje automatyzacji. Sterowanie odbywa się za pomocą komend głosowych, aplikacji mobilnej lub stacji bazowej z ekranem dotykowym (Rys. 2.1.2). Mimo względnie przystępnej ceny, systemy te mają bardzo poważną wadę, która często przesąduje o decyzji zakupu innych urządzeń. Chodzi tutaj o kontrowersyjną politykę prywatności Amazon'a oraz Google'a. Firmy te zbierają i przetwarzają dane na temat swoich użytkowników, na czym opierają swój model finansowy. Alternatywą są produkty mniejszych firm, które często są mniej dopracowane. Sterowanie u większości z nich również odbywa się za pomocą aplikacji mobilnej. Typowa aplikacja *smart home* (Rys. 2.1.3, Rys. 2.1.4) na głównym ekranie wyświetla dodane urządzenia (ich aktualny stan, odczyty z czujników), często pozwala dzielić je na kategorie jak np. pokoje w których się znajdują. Do dodatkowych funkcjonalności należą opcje edycji urządzeń oraz automatyzacje pozwalające na ustawianie zdarzeń według wcześniej ustalonego harmonogramu np. włączenie rano ekspresu do kawy. Funkcje te najczęściej znajdują się na dodatkowych ekranach by nie rozpraszać użytkownika.



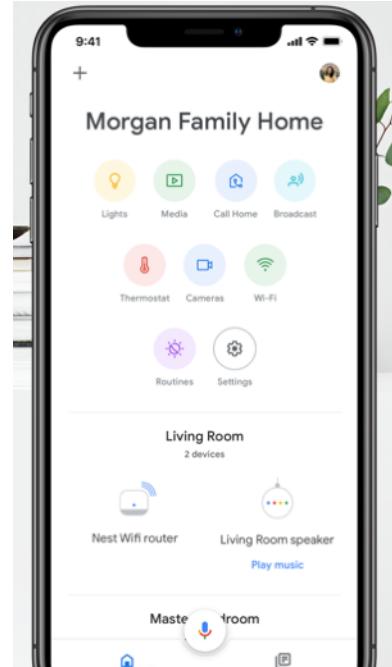
Rys. 2.1.1: Stacja bazowa bez ekranu



Rys. 2.1.2: Stacja bazowa z ekranem dotykowym



Rys. 2.1.3: Aplikacja smart home firmy Apple



Rys. 2.1.4: Aplikacja smart home firmy Google

b) Przegląd urządzeń wykonawczych i czujników

- inteligentne gniazdka

Inteligentne gniazdko służy do zdalnego włączania i wyłączania sprzętów elektrycznych. Dodatkowo może posiadać funkcję monitorowania zużycia energii. Urządzenie te może być wykonane jako wtyczka pośrednicząca między gniazdkiem w ścianie a sprzętem elektrycznym (Rys. 2.1.5) lub jako zamiennik standardowego gniazdko (Rys. 2.1.6). Pierwszy typ dużo łatwiej samodzielnie zainstalować w istniejącym mieszkaniu, drugi natomiast jest lepszym wyborem podczas budowy bądź modernizacji.



Rys. 2.1.5: Inteligentne gniazdko pośredniczące



Rys. 2.1.6: Inteligentne gniazdko podtynkowe

- inteligentne oświetlenie

Inteligentne oświetlenie może być wykonane w postaci żarówki, paska LED (Rys. 2.1.7), bądź jako gotowa do montażu lampa (Rys. 2.1.8). Podstawową funkcją inteligentnego oświetlenia jest możliwość zdalnej kontroli intensywności oraz barwy światła. Niektóre z nich umożliwiają programowanie sekwencji świetlnych.



Rys. 2.1.7: Inteligentny pasek LED



Rys. 2.1.8: Inteligentna lampa stołowa

- inteligentne czujniki

Aktualnie do najpopularniejszych należą czujniki pogodowe (Rys. 2.1.9) pozwalające mierzyć temperaturę, ciśnienie, wilgotność oraz jakość powietrza. Do kolejnej popularnej kategorii zaliczane są wszelkiego rodzaju czujniki bezpieczeństwa (Rys. 2.1.10). Pozwalają one na zdalne monitorowanie domu, jak np. czujniki obecności, czujniki dymu, zalania podłogi.



Rys. 2.1.9: Inteligentny czujnik pogodowy



Rys. 2.1.10: Inteligentny czujnik dymu i czadu

c) Systemy inteligentne wbudowane w budynek

Omawiając rynek urządzeń inteligentnych, nie można pominąć systemów wdrażanych na etapie budowania obiektu. Systemy te często oparte są na urządzeniach podłączonych przewodowo do centrali umieszczonej w skrzynce rozdzielczej (Rys. 2.1.11). Całość sterowana jest za pomocą dotykowego panelu operatorskiego umieszczonego na ścianie (Rys. 2.1.12). Rozwiązywanie to, w przeciwieństwie do wyżej opisanych systemów, nie umożliwia bezproblemowego dodawania nowych urządzeń. W przypadku szybko rozwijającego się rynku inteligentnych urządzeń jest to dużą wadą. Kolejną, jest stosunek ceny do jakości tych systemów.



Rys. 2.1.11: Skrzynka rozdzielcza inteligentnego domu



Rys. 2.1.12: Panel operatorowski inteligentnego domu

2.2. Analiza rozwiązań komunikacyjnych

Ważnym etapem przy projektowaniu systemu inteligentnego domu jest wybór protokołu komunikacyjnego, dlatego rozdział ten poświęcono analizie wiodących rozwiązań komunikacyjnych.

a) Zigbee

Zigbee jest jednym z najpopularniejszych protokołów używanych w budynkach inteligentnych, oparty jest na standardzie IEEE 802.15.4. Zigbee definiuje warstwę sieciową oraz aplikacji, a standard IEEE 802.15.4 warstwę fizyczną oraz łączy danych modelu sieci OSI.

Zigbee umożliwia urządzeniom komunikację bezprzewodową za pośrednictwem jednego z kilku możliwych topologii. Pakiety danych mogą być wysyłane między węzłami i być kierowane przez urządzenia pośredniczące do bardziej odległych węzłów, które w innym przypadku byłyby poza zasięgiem. Każde urządzenie, ma zarówno adres MAC, jak adres sieci Zigbee, podczas gdy cała sieć ma swój własny identyfikator PAN współdzielony przez wszystkie urządzenia.

W sieci Zigbee urządzenie może pełnić jedną z trzech рол:

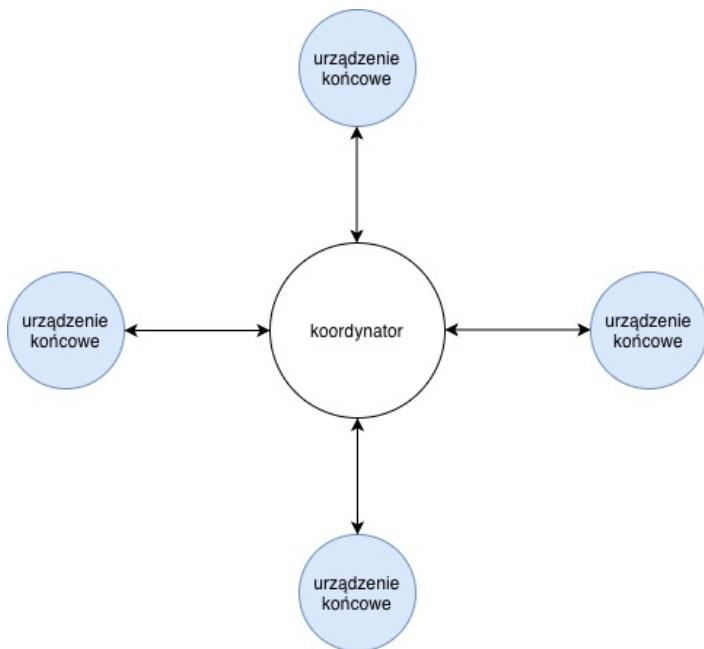
Koordynatora (ZC) - każda sieć posiada jednego koordynatora, tworzy on pierwszą gałąź. Inicjuje resztę sieci, wybiera częstotliwości pracy, identyfikator PAN oraz umożliwia innym węzłom dołączenie do sieci.

Routera (ZR) - routery nie są wymaganym węzłem w sieci. Odpowiadają za przekazywanie wiadomości do innych węzłów. Węzły mogą również łączyć się z siecią za pośrednictwem routera, router staje się wtedy ich węzłem nadziedzonym.

Urządzenia końcowego (ZED) - urządzenie końcowe jest węzłem, który może jedynie odpierać i wysyłać dane. Nie wykonuje innych specjalnych funkcji w sieci.

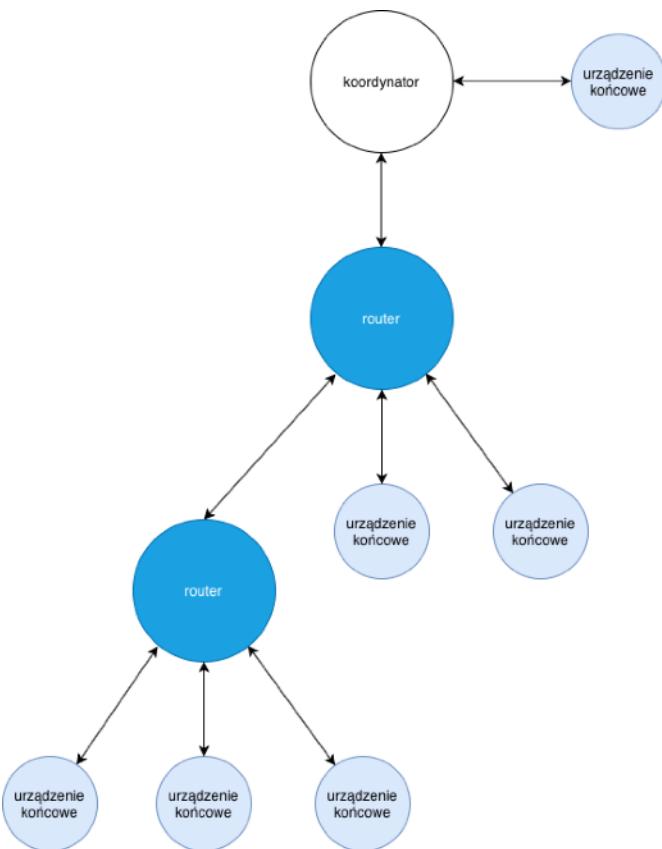
Sieci Zigbee mogą być zbudowane w oparciu o jedną z trzech różnych topologii. Wpływają one na sposób, w jaki wiadomości są przekazywane między urządzeniami. Do podstawowych topologii zaliczamy:

Topologia gwiazdy (Rys. 2.2.1) - jest to najprostsza topologia dostępna w Zigbee. Wszystkie urządzenia podłączone są do koordynatora i cała komunikacja odbywa się przez jeden węzeł. W tej topologii, przepustowość oraz zasięg sieci ograniczony jest do przepustowości i zasięgu koordynatora.



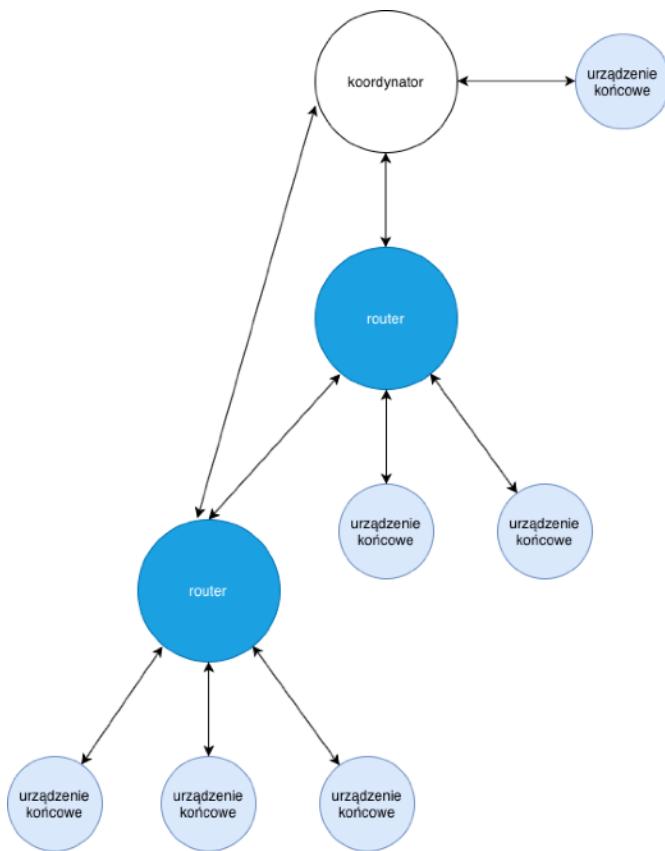
Rys. 2.2.1: Topografia gwiazdy w sieci Zigbee [2]

Topologia drzewa (Rys. 2.2.2) - koordynator tworzy główny węzeł drzewa, „liśćmi” są urządzenia końcowe a routery służą za węzły pośredniczące. Węzły komunikują się ze sobą poprzez przesyłanie komunikatów w górę drzewa w kierunku węzła docelowego. W tej topologii routery mogą rozszerzać zasięg sieci poza zasięg pojedynczego urządzenia. W przypadku awarii routera niektóre części sieci mogą zostać rozłączone.



Rys. 2.2.2: Topologia drzewa w sieci Zigbee [2]

Topologia siatki (Rys. 2.2.3) - topologia siatki jest podobna do wcześniej opisanej topologii, ale nie jest tworzona według sztywnej struktury drzewa. W tej topologii router może komunikować się bezpośrednio z dowolnym innym routерem lub koordynatorem jeśli znajduje się w zasięgu. Może istnieć wiele tras do istniejących węzłów, dlatego w przypadku awarii routera urządzenia końcowe mogą automatycznie połączyć się z innym routерem bądź koordynatorem.



Rys. 2.2.3: Topologia siatki w sieci Zigbee [2]

Sieć Zigbee rozróżnia dwa fizyczne rodzaje urządzeń - Full Function Device (FFD) i Reduced Function Device (RFD). RFD są często zasilane baterijnie, dzięki czemu mogą być usypane między transmisją danych w celu oszczędzania energii. Nie mogą one jednak pełnić funkcji routera ani koordynatora. Urządzenia FFD mogą spełniać dowolną funkcję w sieci.

Protokół Zigbee, dzięki wykorzystaniu topologii siatki bardzo dobrze nadaje się do dużych obiektów w których zasięg sieci Wi-Fi jest niewystarczający do stworzenia systemu inteligentnego. Do głównych zastosowań należy inteligentne oświetlenie oraz zasilane baterijnie czujniki.

b) KNX

KNX jest otwartym standardem (EN 50090, ISO / IEC 14543) dla automatyki budynków komercyjnych i domowych. Standard ten oparty jest na magistrali. Kolejną cechą systemu KNX jest jego zdecentralizowana struktura: nie ma potrzeby centralnej jednostki sterującej, ponieważ "inteligencja" systemu jest rozłożona na wszystkie jego urządzenia.

Istnieją dwa rodzaje urządzeń w systemie KNX:

Urządzenia końcowe - należą do nich czujniki, urządzenia sterujące i urządzenia wykonawcze.

Urządzenia systemowe - Urządzenia systemowe KNX to urządzenia, które wykonują przede wszystkim funkcje specjalne należą do nich: zasilacze, programatory, interfejsy, łączniki.

Do wymiany danych między urządzeniami w systemie KNX można wykorzystać następujące media:

Przewód skręcany (KNX TP) - Jest najczęstszym środkiem komunikacji w instalacji KNX. W tej konfiguracji przewód magistrali oprócz możliwości wymiany danych zapewnia również zasilanie. Ważną cechą KNX TP jest to że, sygnały przesyłane są symetrycznie, oznacza to zwiększoną odporność na zakłócenia.

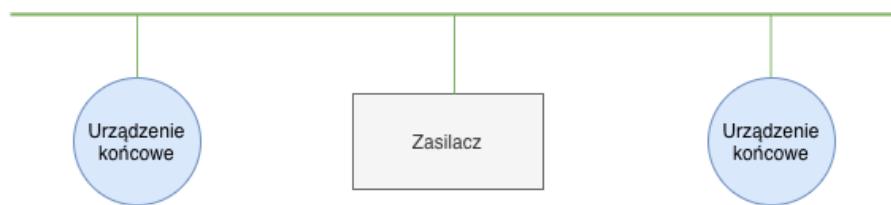
Instalację elektryczną (KNX PL) - Polega na wykorzystaniu istniejących kabli elektrycznych w budynku jako medium komunikacyjnego. Nie ma potrzeby instalowania dedykowanego kabla magistrali, opcja ta jest szczególnie przydatna w przypadku modernizacji starszych budynków.

Połączenie bezprzewodowe (KNX RF) - KNX RF jest głównie wykorzystywany w miejscach, do których nie można doprowadzić przewodów. Teoretycznie KNX RF pozwala na bezprzewodową komunikację całego systemu w budynku, ale w praktyce nie stosuje się tego rozwiązania.

Ethernet (KNX IP) - KNX IP jest wykorzystywany w celu zwiększenia możliwości systemu a nie jako substytut dla pozostałych metod komunikacji. Pozwala na łączenie ze sobą obszarów systemu oraz na zdalne zarządzanie budynkiem za pomocą sieci internet.

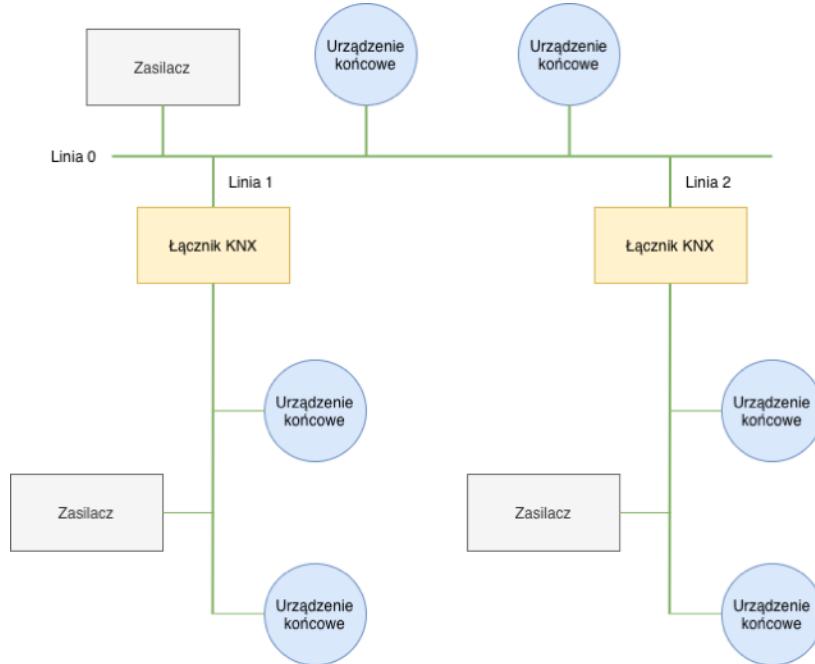
Systemy KNX można rozszerzać w miarę potrzeb i mogą składać się z kilku podsystemów opartych na różnych mediach komunikacyjnych. Aby zapewnić bezproblemową transmisję danych między poszczególnymi urządzeniami magistrali, systemy KNX muszą stosować się do określonej topologii.

KNX TP - Podstawową jednostką budującą instalację KNX TP jest topologia liniowa (Rys. 2.2.4). Składa się z zasilacza i zwykle, nie więcej niż 64 innych urządzeń. Zasilacz i przewód pełnią dwie funkcje: zasilają urządzenia oraz umożliwiają wymianę informacji.



Rys. 2.2.4: Topologia liniowa w sieci KNX [3]

Topografię liniową można rozszerzać tworząc nowe linie za pomocą łączników. Maksymalnie można ze sobą połączyć 15 linii tworząc w ten sposób obszar w systemie KNX (Rys. 2.2.5). Obszary podobnie jak linie mogą być ze sobą łączone tworząc kompletną instalację. Taki podział systemu pozwala na bardziej niezawodne działanie - wszystkie linie i obszary mają własne zasilanie. Pomaga to w zarządzaniu systemem oraz zmniejsza liczbę wysyłanych wiadomości wzdłuż każdej linii.



Rys. 2.2.5: Obszar w sieci KNX [3]

KNX PL - Topologia KNX PL jest podobna do KNX TP i składa się z linii i obszarów. Zamiast łączników liniowych stosuje się łączniki systemowe KNX PL. Poszczególne linie należy od siebie odseparować za pomocą filtrów pasmowo-przepustowych.

KNX RF - Urządzenia w instalacji KNX RF nie muszą być rozmieszczane hierarchicznie i można je zainstalować praktycznie wszędzie, pod warunkiem, że znajdują się w swoim zasięgu.

KNX IP - KNX IP jest stosowany do łączenia ze sobą obszarów bądź linii systemu, wymaga to użycia routerów KNX. Routery mogą być również używane do programowania urządzeń w sieci.

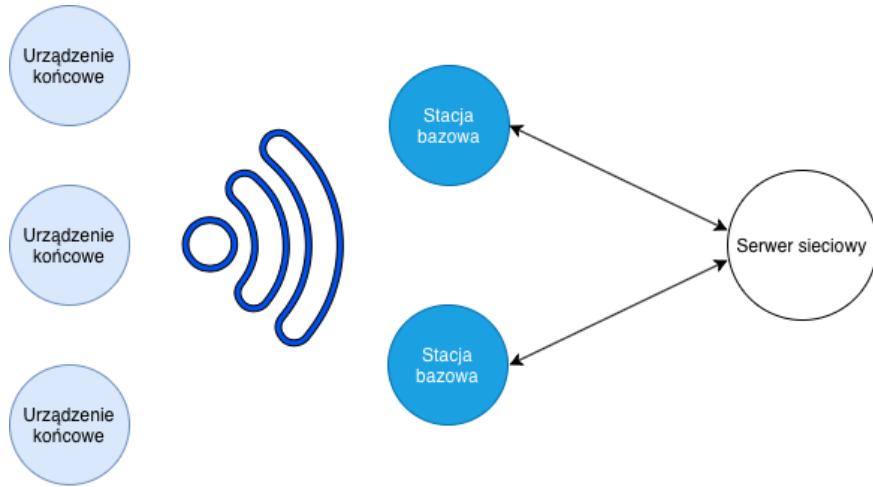
Wszystkie topologie różnych mediów, mogą być w razie potrzeby stosowane w połączeniu ze sobą. Wymaga to użycia dedykowanych urządzeń łączących.

Złożoność instalacji KNX powoduje, że najczęściej jest ona projektowana oraz instalowana przez specjalistów. Systemy te stosuje się głównie w dużych budynkach rodzinnych lub na obiektach komercyjnych takich jak hotele lub baseny. Dzięki swojej złożoności KNX oferuje głęboką integrację z budynkiem a lista rodzaju obsługiwanych urządzeń jest bardzo długa.

c) LoRa/LoRaWAN

LoRaWAN (Low Power Wide Area Network) jest protokołem przeznaczonym do bezprzewodowego łączenia urządzeń do sieci internet. Natomiast LoRa to warstwa fizyczna - modulacja bezprzewodowa wykorzystywana do tworzenia łączą komunikacyjnego dalekiego zasięgu.

Sieci LoRaWAN zbudowane są w oparciu o topologię rozszerzonej gwiazdy (Rys. 2.2.6). System składa się z trzech głównych komponentów: serwerów sieciowych, stacji bazowych i urządzeń końcowych. Urządzenia końcowe komunikują się za pośrednictwem stacji bazowych z serwerem sieciowym. Te zaś, zarządzają stacjami bazowymi za pomocą standardowej technologii IP.



Rys. 2.2.6: Topologia sieci LoRaWAN [4]

W LoRaWAN urządzenia końcowe nie są powiązane z określoną stacją bazową. Zamiast tego, przesyłane dane są zwykle odbierane przez wiele przekaźników. Każda stacja przekazuje otrzymany pakiet do serwera sieciowego, który filzuje nadmiarowe pakiety oraz wysyła potwierdzenia odbioru optymalną trasą. Serwer dostosowuje również prędkość przesyłu danych między urządzeniami bezprzewodowymi, jest ona wprost proporcjonalna do jakości sygnału. Umożliwia to urządzeniom bezprzewodowym niższe zużycie energii.

W sieci LoRa dostępne są trzy klasy urządzeń końcowych, każda dostosowana do konkretnych wymagań:

Klasa A - Urządzenia końcowe najniższej mocy. Jest to domyślna klasa, którą muszą obsługiwać wszystkie urządzenia końcowe. Komunikacja jest zawsze inicjowana przez urządzenia bezprzewodowe oraz jest w pełni asynchroniczna. Po transmisji w stronę serwera, następują dwa krótkie okna umożliwiające wysłanie danych przez serwer. Dzięki temu urządzenia końcowe mogą wchodzić w stan uśpienia na dowolnie długi czas. Klasa ta jest głównie wykorzystywana przez czujniki.

Klasa B - Urządzenia końcowe z deterministycznym opóźnieniem. Urządzenia klasy B są synchronizowane z siecią za pomocą określonych sygnałów, otwierając połączenie o zaplanowanych porach. Zapewnia to sieci deterministyczne opóźnienie kosztem dodatkowego zużycia energii. Klasa ta jest głównie wykorzystywana przez urządzenia wykonawcze.

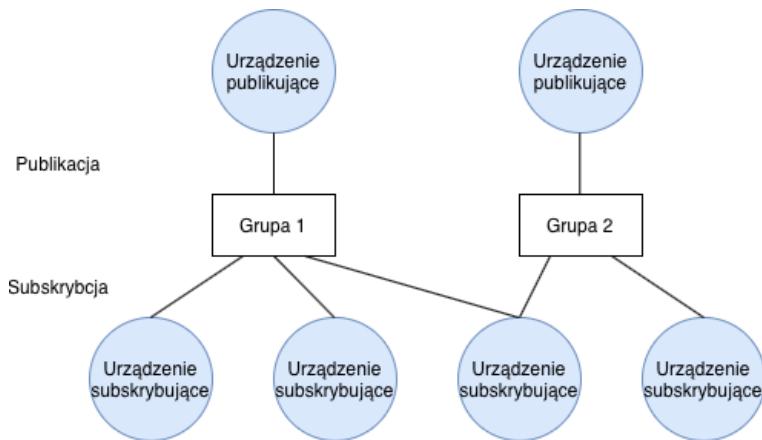
Klasa C - Urządzenie końcowe z niskim opóźnieniem. W klasie C opóźnienie w łączu w dół zostało zredukowane dzięki utrzymywaniu odbiornika w stanie ciągłego nasłuchu. Dzięki temu serwer może zainicjować połączenie w dowolnym momencie. Kompromisem jest pobór mocy odbiornika, dlatego klasa C nadaje się do urządzeń zasilanych z sieci elektrycznej.

Technologia LoRaWAN została zaprojektowana głównie do obsługi czujników przesyłających niewielką ilość danych kilka razy dziennie. Nie jest przeznaczona do obsługi aplikacji wymagających wysokich prędkości transmisji danych, takich jak audio lub wideo. Do głównych zastosowań tej technologii należą: inteligentne miasta, fabryki, dedykowane sieci do określonych branż, jak np. zarządzanie lotniskiem.

d) Bluetooth LE

Bluetooth Low Energy (BLE), czasem określany jako Bluetooth Smart, jest odmianą klasycznej wersji standardu Bluetooth, nawiązującą na rynek inteligentnych urządzeń. Standard ten cechuje znacznie zmniejszone zużycie energii oraz kosztów produkcji, przy zachowaniu podobnego zasięgu komunikacji. Początkowo, standard Bluetooth obsługiwał jedynie topologię punkt-punkt oraz gwiazdy. Umożliwiało to urządzeniom bazowym nawiązywanie połączenia z jednym lub kilkoma urządzeniami końcowymi. W 2019 roku dodano topologię siatki, w której każdy węzeł w sieci może komunikować się z każdym innym węzłem bezpośrednio lub poprzez komunikację *multi-hop*.

Standard Bluetooth Mesh jest oparty o model publikacji/subskrypcji (Rys. 2.2.7). Podczas publikowania, urządzenia mogą wysyłać wiadomości na określone grypy adresów unicast/multicast. Urządzenia mogą również być skonfigurowane by odbierać wiadomości wysłane na określony adres - adres subskrybcji. Wiadomości opublikowane na podanym adresie, będą odbierane przez wszystkie subskrybujące go urządzenia. Każdy węzeł w sieci posiada listę subskrybentów oraz adres unicast.



Rys. 2.2.7: Model publikacji/subskrypcji w sieci Bluetooth Mesh [5]

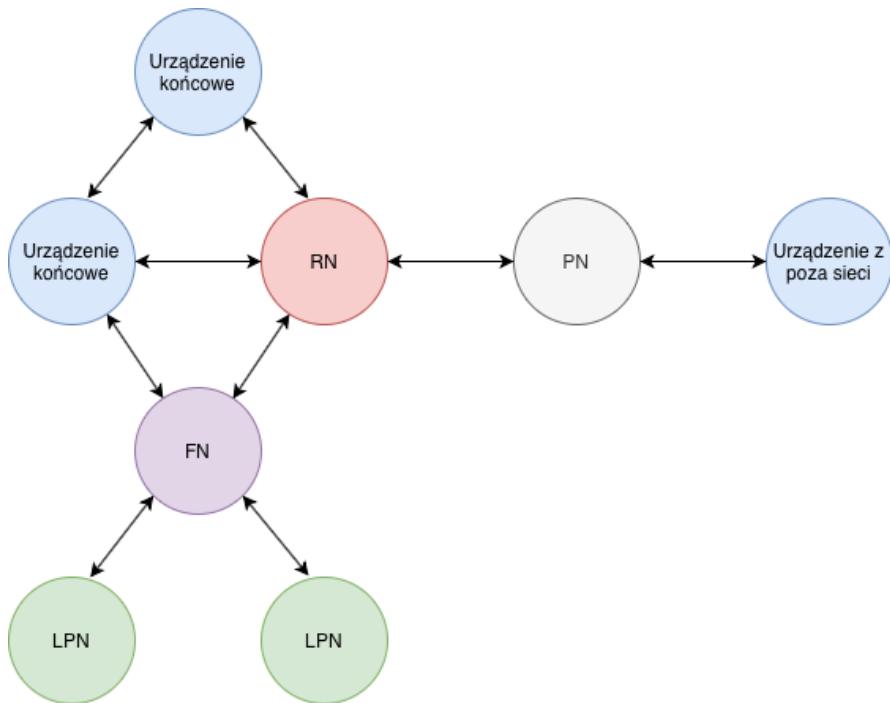
Zgodnie ze specyfikacją urządzenia w sieci Bluetooth Mesh mogą posiadać różne funkcje, które pozwalają węzłom wykonywanie określonych zadań w sieci. Zależności między urządzeniami pokazano na rysunku 2.2.8.

Proxy (PN) - Węzły proxy zapewniają kompatybilność wsteczną z urządzeniami, które nie obsługują standardu Bluetooth Mesh, np. umożliwiają interakcję urządzeń w sieci ze smartfonem.

Relay (RN) - Węzły te mogą retransmitować otrzymane wiadomości co umożliwia dostarczanie wiadomości do odległych urządzeń w sieci.

Friend (FN) - Węzły friend zostały stworzone, by asystować urządzeniu Low Power. FN przechowuje wiadomości zaadresowane do LPN i dostarcza je gdy LPD jest gotowy na odbiór wiadomości. Każdy węzeł Friend powinien być również PN.

Low Power Node (LPN) - Dzięki asyście FN urządzenia Low Power mogą spędzać większość czasu w trybie uśpienia, oszczędzając tym samym energię elektryczną. Najczęściej są to urządzenia zasilane baterijne np. czujniki.



Rys. 2.2.8: Topologia sieci Bluetooth Mesh [6]

e) Wi-Fi

Wi-Fi oparte jest na standardzie IEEE 802.11, który definiuje warstwę fizyczną oraz warstwę łącza danych w modelu sieci OSI. Jest to jedna z najpopularniejszych technologii używanych w komunikacji bezprzewodowej i obecna jest w każdym współczesnym domu. Większość stacji bazowych w systemach smart-home używa Wi-Fi w celu połączenia się z siecią internet, nawet jeśli używają innego protokołu do komunikacji z urządzeniami końcowymi.

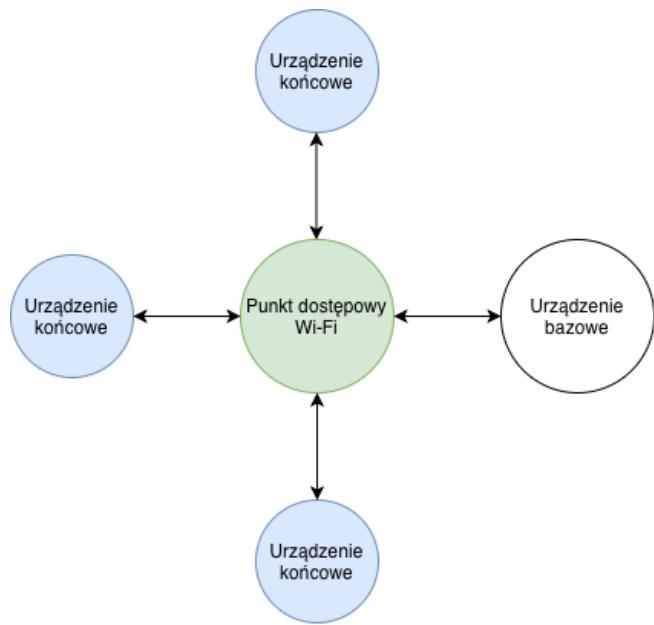
IEEE 802.11 definiuje dwa tryby pracy. Wszystkie stacje bezprzewodowe muszą wybrać tryb pracy przed próbą utworzenia sieci Wi-Fi lub przyłączenia się do niej.

Tryb ad-hoc - W tym trybie stacje bezprzewodowe komunikują się bezpośrednio ze sobą. Nie ma połączenia z innymi sieciami Wi-Fi lub siecią LAN. By komunikacja działała poprawnie wszystkie urządzenia muszą być w swoim zasięgu.

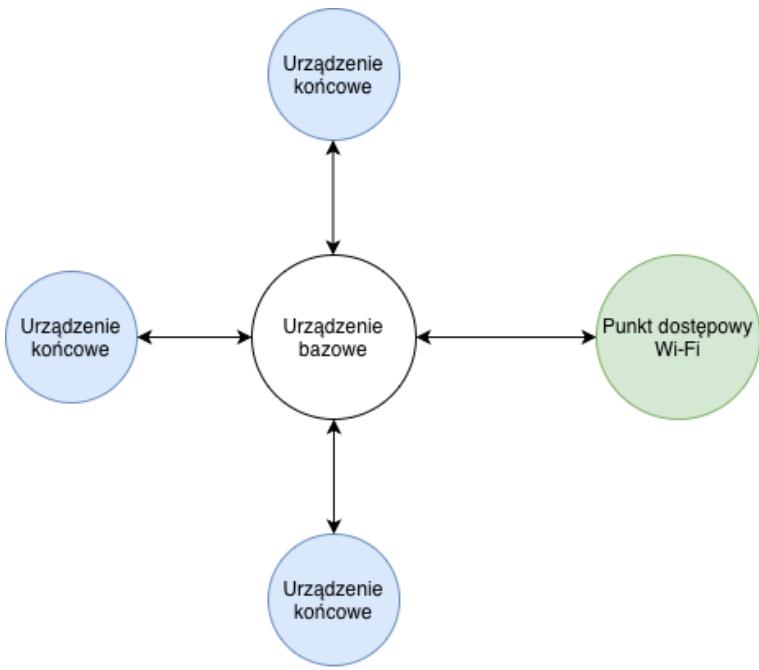
Tryb infrastruktury - Tryb ten wymaga aby sieć Wi-Fi zawierała conajmniej jeden bezprzewodowy punkt dostępu (AP). Wszystkie urządzenia próbujące dołączyć do sieci Wi-Fi muszą być powiązane z AP. Cała komunikacja bezprzewodowa w sieci przechodzi przez punkt dostępowy gdy sieć jest skonfigurowana w trybie infrastruktury.

Systemy inteligentne oparte na technologii Wi-Fi w większości korzystają z trybu infrastruktury (Rys. 2.2.9). Mogą one wykorzystywać istniejącą sieć bezprzewodową - urządzenie bazowe jak i urządzenia końcowe łączą się z domowym routerem Wi-Fi, bądź punktem dostępowym. Pozwala to w pełni wykorzystać domową infrastrukturę np. zdobywającą coraz większą popularność sieć Wi-Fi Mesh.

W drugim przypadku, system inteligentny tworzy własną sieć (Rys. 2.2.10) - tutaj urządzenie bazowe jest jednocześnie połączone z domowym routerem w celu zapewnienia łączności z siecią internet oraz służy jako punkt dostępowy dla pozostałych inteligentnych urządzeń. Zaletą takiego rozwiązania jest łatwiejszy sposób parowania - urządzenia końcowe mogą łączyć się automatycznie ze stacją bazową.



Rys. 2.2.9: System inteligentny oparty na istniejącej sieci Wi-Fi



Rys. 2.2.10: System inteligentny tworzący własną sieć Wi-Fi

f) Zestawienie technologii

Podrozdział ten poświęcono zestawieniu najpopularniejszych technologii komunikacyjnych. W tabeli 2.2.1 porównano kluczowe specyfikacje każdej technologii. Należy pamiętać, że wypisane wartości przedstawiają teoretyczny limit i mogą się różnić w zależności od używanego sprzętu oraz od środowiska. W tabeli 2.2.2 porównano specyfikację konkretnych modułów rozwojowych. Przy ich wyborze kierowano się dostępnością oraz ceną - starano się wybrać urządzenia podobnej klasy, by nie faworyzować którejś z technologii. W tabeli nie uwzględniono urządzenia implementującego protokół KNX RF, ponieważ nie znaleziono odpowiedniego modułu rozwojowego. W trzeciej tabeli porównano najważniejsze wady, zalety oraz zastosowanie opisanych protokołów w systemach inteligentnych.

Tabela 2.2.1: Zestawienie specyfikacji opisanych technologii [7][8][9]

	Zigbee	KNX	LoRa	Bluetooth LE	Wi-Fi
Częstotliwość pracy	2,4Ghz	868,3kHz	867-869Mhz	2,4Ghz	2,4Ghz, 5Ghz
Szerokość pasma	2Mhz	-	125/250KHz	1Mhz	20-160Mhz
Typowy zasięg	10-300m	150m	10-15km	10-100m	100-500m
Maksymalna prędkość transmisji	250kb/s	9.6kb/s	50kb/s	1Mb/s	1Gb/s
Maksymalna liczba urządzeń	65000	57375	62000	32767	250 na punkt dostępowy
Opóźnienie	30-200ms	-	> 1s	20-800ms	1-100ms
Maksymalny pobór prądu	30mA	-	30mA	15mA	116mA

Tabela 2.2.2: Zestawienie specyfikacji wybranych modułów [10][11][12][13]

Moduł	ESP-01	HM-10	Waveshare - 11212	RFM95
Standard	Wi-Fi 811b/g/n	Bluetooth 4.0 LE	Zigbee	LoRA
Układ	ESP8266	CC2541	CC2530	SX1276
Napięcie zasilania	3,3V	3,6-6V	2-3,6V	3,3-5V
Pobór prądu RX	50mA	17,9mA	24mA	9,9mA
Pobór prądu TX	120mA	18,2mA	29mA	20mA
Pobór prądu tryb czuwania	0,9mA	270uA	0,2mA	1,8mA
Cena	~15zł	~20zł	~50zł	~80zł
Zasięg	100m	20m	130m	500m

Tabela 2.2.3: Wady, zalety oraz zastosowanie opisanych technologii

	Zalety	Wady	Typowe zastosowanie
Zigbee	<ul style="list-style-type: none"> Niskie zużycie energii Duży zasięg Wysoka niezawodność Szyfrowanie Niskie opóźnienia 	<ul style="list-style-type: none"> Brak kompatybilności Niska prędkość transmisji Podatność na zakłócenia 	<ul style="list-style-type: none"> Inteligentne oświetlenie Inteligentne czujniki zasilane baterijnie
KNX	<ul style="list-style-type: none"> Integracja z budynkiem Kompatybilność Wybór urządzeń Szeroki zakres mediów komunikacyjnych 	<ul style="list-style-type: none"> Wysoka cena Poziom skomplikowania Nie wszystkie urządzenia obsługują szyfrowanie Urządzenia muszą być instalowane przez specialistów 	<ul style="list-style-type: none"> Automatyka zintegrowana w dużych budynkach prywatnych lub komercyjnych
LoRa	<ul style="list-style-type: none"> Bardzo duży zasięg Niskie zużycie energii Szyfrowanie Mało zatłoczone pasmo 	<ul style="list-style-type: none"> Wysoka cena Zamknięty standard Niska prędkość transmisji Duże opóźnienia 	<ul style="list-style-type: none"> Rozproszone czujniki w rozległych systemach jak np. Inteligentne miasta Systemy lokalizacji
Bluetooth LE	<ul style="list-style-type: none"> Niska cena Niskie zużycie energii Szyfrowanie Popularność Kompatybilność 	<ul style="list-style-type: none"> Średnia prędkość transmisji Podatność na zakłócenia Zasięg uzałączony od używanej topologii 	<ul style="list-style-type: none"> Inteligentne oświetlenie Inteligentne czujniki Inteligentne urządzenia ubieralne Inteligentne nagłośnienie
Wi-Fi	<ul style="list-style-type: none"> Niska cena Wysoka prędkość transmisji Popularność Kompatybilność Uniwersalność Szyfrowanie Niskie opóźnienia 	<ul style="list-style-type: none"> Wysokie zużycie energii Zasięg uzałączony od istniejącej infrastruktury 	<ul style="list-style-type: none"> Zastosowanie Wi-Fi w systemach inteligentnych jest bardzo rozległe - od inteligentnych czujników zasilanych baterijnie do systemów monitoringu

2.3. Wybór technologii

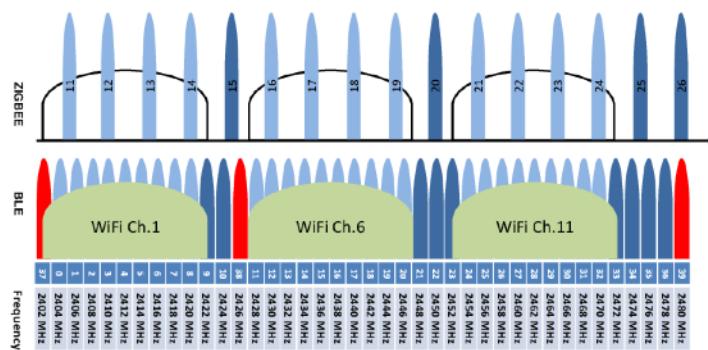
Technologię komunikacyjną dla projektowanego systemu intelligentnego dobrano drogą eliminacji. Główne kryteria jakie brano pod uwagę to:

- prędkość transmisji,
- opóźnienia,
- dostępność,
- cena,
- zużycie energii.

Pierwszą wyeliminowaną technologią jest KNX. Technologia ta okazała się za mało przystępna - trudno dostępne moduły rozwojowe uniemożliwiają tworzenie urządzeń prototypowych. Dodatkowo, protokół ten nie umożliwia łatwego zarządzania urządzeniami przez użytkownika końcowego, co jest sprzeczne z założeniami projektowymi. Wdrażanie tego systemu jest dobrym rozwiązaniem podczas budowy obiektu, kiedy może zostać z nim w pełni zintegrowany.

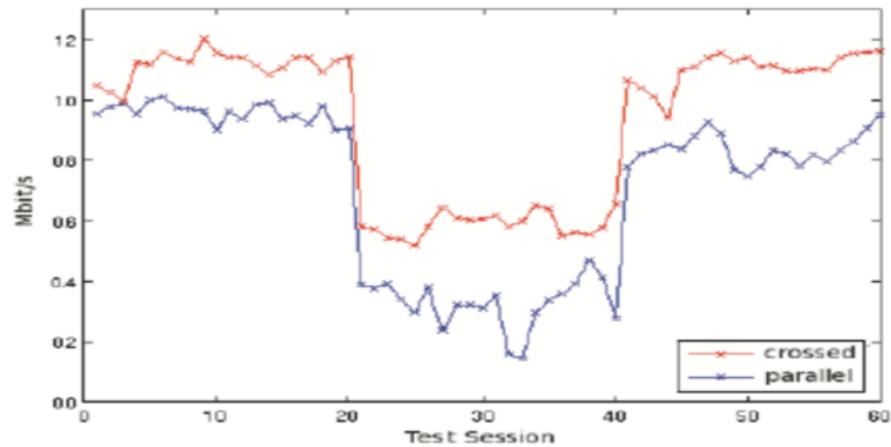
Kolejną wykluczoną technologią została LoRa. Mimo, że cechuje się bardzo niskim zużyciem energii oraz dalekim zasięgiem, protokół ten oferuje niską prędkość transmisji, co ogranicza jego zastosowania. Kolejnym wykluczającym czynnikiem jest cena stacji bazowych, które mogą sięgać kilkuset złotych. Dlatego technologię tę częściej spotkamy w komercyjnych systemach monitorujących np. poziom wody, niż w intelligentnych domach.

Technologie Bluetooth Low Energy oraz Zigbee zostały wyeliminowane z podobnych przyczyn. Podobnie jak Wi-Fi korzystają z nielicencjonowanego pasma 2,4Ghz (Rys. 2.3.1) i ich kanały mogą się wzajemnie nakładać, co prowadzi do wzajemnych zakłóceń. Przeprowadzone badania wykazały, że Bluetooth i Zigbee cierpią szczególnie z powodu obecności Wi-Fi. Na poziom zakłóceń największy wpływ miała odległość między urządzeniami oraz obciążenie sieci.

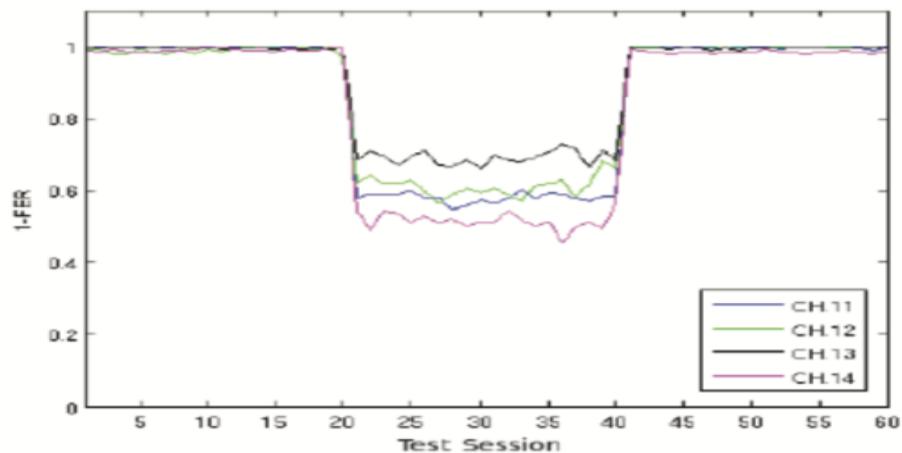


Rys. 2.3.1: Widmo technologii Wi-Fi, Zigbee oraz Bluetooth [14]

Rysunek 2.3.2 pokazuje zmniejszenie przepustowości sieci Bluetooth w obecności Wi-Fi. W dwóch próbach odnotowano spadek odpowiednio z 1,12 Mb/s na 0,59Mb/s oraz z 0,95 Mb/s na 0,30Mb/s. Na rysunku 2.3.3 przedstawiono stopę błędu FER (ang. *Frame Error Rate*) w sieci Zigbee podczas transmisji Wi-Fi. Można zauważyć, że błędy podczas transmisji danych zależą od używanego przez Wi-Fi kanału i mogą wynosić od 0,45 do 0,73.

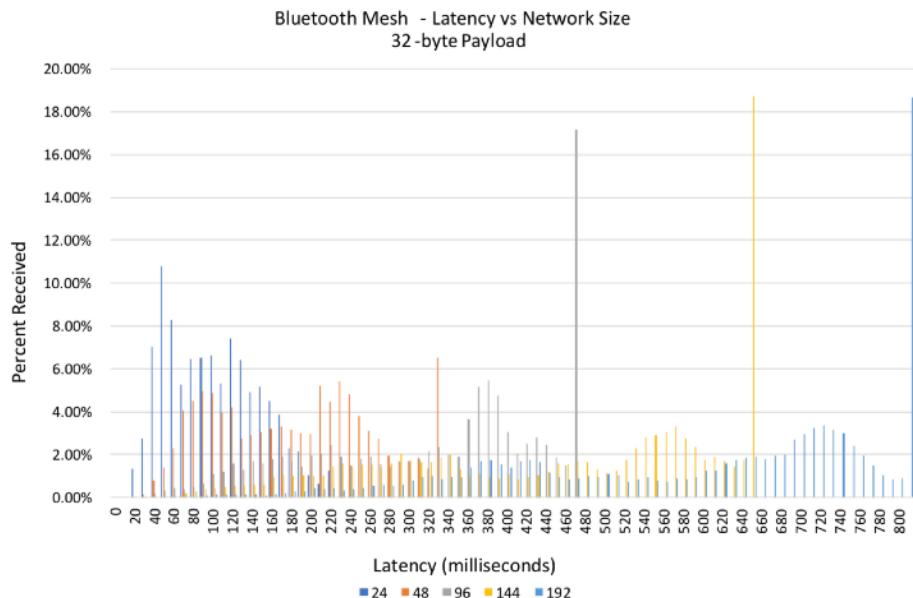


Rys. 2.3.2: Wpływ Wi-Fi na prędkość transmisji w sieci Bluetooth [15]

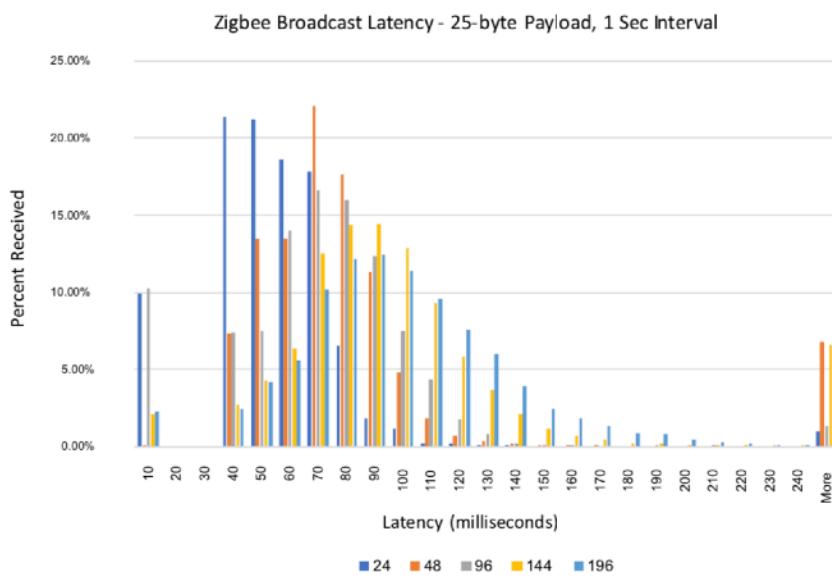


Rys. 2.3.3: Wpływ Wi-Fi na błędy transmisji w sieci Zigbee [15]

Kolejną przyczyną wyeliminowania tych technologii są opóźnienia, do których może dochodzić w większych instalacjach. Przyczyną ich powstawania jest zastosowanie topologii siatki, w której wiadomości często muszą przeskakiwać przez kilka węzłów nim dotrą do celu. Szczególnie podatna jest technologia Bluetooth, w której opóźnienia mogą przekraczać nawet 800ms. Zależność czasu opóźnienia od ilości urządzeń w sieciach Zigbee oraz Bluetooth pokazano na rysunkach 2.3.4 oraz 2.3.5.



Rys. 2.3.4 Wpływ wielkości sieci Bluetooth Mesh na opóźnienia [16]

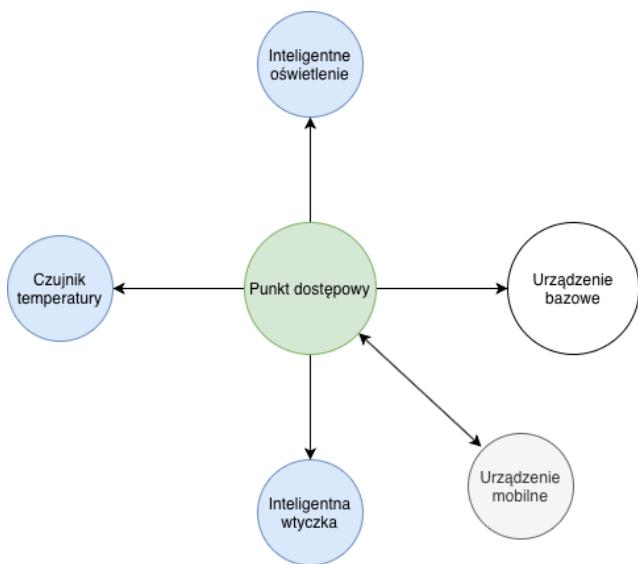


Rys. 2.3.5 Wpływ wielkości sieci Zigbee na opóźnienia [16]

Ostatecznie, zdecydowano się na użycie technologii Wi-Fi. Oferuje ona wysoką prędkość transmisji, niskie opóźnienia oraz odpowiedni zasięg do zastosowań w urządzeniach inteligentnych. Największą wadą tego rozwiązania okazuje się wyższe zużycie energii w porównaniu do konkurencyjnych technologii. Jednak, jak pokazały badania [17] komponenty Wi-Fi o niskiej mocy dostarczają wielu lat żywotności baterii, zapewniając jednocześnie łatwą integrację z istniejącą infrastrukturą dzięki wbudowanej kompatybilności z siecią IP. Ponowne wykorzystanie istniejącej infrastruktury Wi-Fi, zapewnia kluczowe oszczędności i szybsze wdrażanie systemów inteligentnych. Ponadto, technologia ta, dzięki dużej prędkości transmisji zapewnia większą funkcjonalność w porównaniu z pozostałymi rozwiązaniami jak np. możliwość przesyłu obrazu.

2.4. Opis koncepcji

Zaprojektowany system sterowania w inteligentnym budynku składa się z trzech urządzeń - urządzenia bazowego, inteligentnego oświetlenia LED RGB, inteligentnej wtyczki oraz czujnika temperatury (Rys. 2.4). Wszystkie elementy systemu mogą być umieszczone w dowolnej części budynku dzięki bezprzewodowej transmisji danych z wykorzystaniem istniejącej infrastruktury Wi-Fi. Najważniejszym elementem wybranej koncepcji jest urządzenie bazowe zbudowane w oparciu o minikomputer jednopłytkowy z systemem operacyjnym Linux. Służy on jako interfejs użytkownika dzięki zainstalowanemu wyświetlaczu dotykowemu oraz aplikacji internetowej. Aplikacja pozwala na sterowanie urządzeniami oraz wykonywanie czynności administracyjnych, takich jak: dodawanie/usuwanie urządzeń, zmiana nazw urządzeń, dodawanie/usuwanie użytkowników, przypisywanie użytkowników do urządzeń. Urządzenie bazowe służy również jako serwer WWW pozwalając użytkownikom na dostęp do systemu z poziomu urządzeń mobilnych oraz jako serwer TCP, dzięki któremu możliwa jest wymiana danych z urządzeniami końcowymi. Wykorzystano protokół TCP, ponieważ w przeciwieństwie do protokołu UDP, zapewnia poprawność przesłanych danych oraz potwierdzenie odbioru co jest niezbędnym elementem w przypadku sterowania urządzeniami. Urządzenia końcowe zostały zbudowane w oparciu o mikrokontroler oraz moduł Wi-Fi, każde zostało dodatkowo wyposażone w dwie diody LED wyświetlające informacje o stanie połączenia oraz w przycisk pozwalający sparować urządzenie z systemem. SSOT (ang. *Single Source of Truth*) w systemie jest baza danych uruchomiona w urządzeniu bazowym. Przechowywane są w niej informacje o użytkownikach, urządzeniach oraz relacje między nimi. Każda zmiana w systemie jest jednoznaczna ze zmianą danych w bazie.



Rys. 2.4 Koncepcja rozwiązania

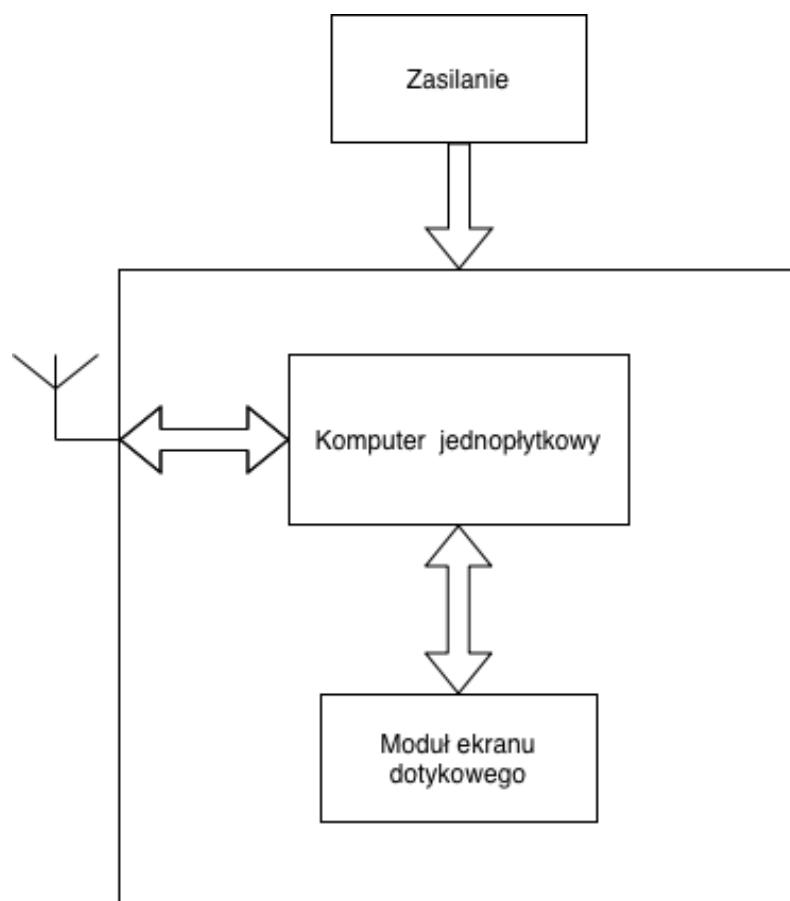
Rozdział 3

Opis części sprzętowej

Projektując część sprzętową systemu kierowano się optymalizacją liczby potrzebnych elementów, ich ceną oraz prostotą wykonania. Wszelkie skomplikowane mechanizmy: zabezpieczenia; eliminacja drgań styków; interfejs użytkownika; zaimplementowane zostały programowo.

Koncepcję działania urządzeń systemu intelligentnego przedstawiono na schematach blokowych (Rys. 3.1.1, Rys. 3.2.1), pod którymi znalazły się opisy poszczególnych bloków wraz ze schematami podłączeń.

3.1 Urządzenie bazowe



Rys. 3.1.1: Schemat blokowy urządzenia bazowego

a) Komputer jednoplatformowy

Przy projektowaniu urządzenia bazowego starano się jak najbardziej uniezależnić część sprzętową od części programowej, by w miarę rozwoju technologii móc modernizować urządzenie. Takie podejście pozwala z łatwością dodawać nowe funkcje oraz poprawiać doświadczenie użytkownika. Dlatego do budowy urządzenia bazowego wykorzystano komputer jednoplatformowy z systemem operacyjnym z rodziny Linux.

Linux został stworzony jako system modułowy, dzięki czemu może być uruchomiony na różnych platformach sprzętowych a większość oprogramowania jest niezależna od architektury procesora. Modułowa konstrukcja oznacza również, że jądro Linuxa jest niezależne od interfejsu graficznego. W rezultacie, awarie i luki w zabezpieczeniach aplikacji nie wpływają na cały system. Jest to niewątpliwie zaletą w przypadku urządzeń intelligentnych. Dodatkowo, Linux jest darmowy oraz otwarty - najczęściej używaną licencją jest GPL (ang. *GNU Public License*). Oznacza to, że można edytować kod źródłowy oprogramowania np. dostosowując go do potrzeb produktu bez ponoszenia opłat licencyjnych. W praktyce może się to przekładać na niższą cenę urządzenia.

Większość komputerów jednoplatformowych na rynku oferuje podobne możliwości dlatego przy wyborze kierowano się głównie stosunkiem ceny do jakości. Jedynymi wymaganiami była obecność złącza ethernet i interfejsu Wi-Fi, dzięki którym urządzenie mogłoby połączyć się z siecią oraz umożliwiać obsługę ekranu dotykowego. Stąd, jako platformę sprzętową wybrano Raspberry Pi 4B. Przy cenie poniżej 200 zł okazała się bezkonkurencyjna. Czterordzeniowy procesor 1,5 GHz oraz 2 GB pamięci RAM to więcej, niż wystarczająco do obsługi aplikacji internetowej oraz serwera TCP, pomoże to jednak uodpornić urządzenie na starzenie. Do kolejnych zalet należy obsługa nowoczesnych standardów takich jak: Bluetooth 5.0; USB typu C; dwuzakresowy moduł Wi-Fi; obsługa rozdzielczości 4K. Dodatkowo, dzięki dużej popularności oraz aktywnej społeczności rynek akcesoriów jak np. wyświetlacze czy kamery jest nieporównywalnie większy od konkurencji. Raspberry nie jest jednak pozbawione wad. Do największych należy brak wbudowanej pamięci stałej, przez co do działania wymagana jest karta SD bądź zewnętrzny dysk USB. Schemat podłączenia urządzenia przedstawiono na rysunku 3.1.3.

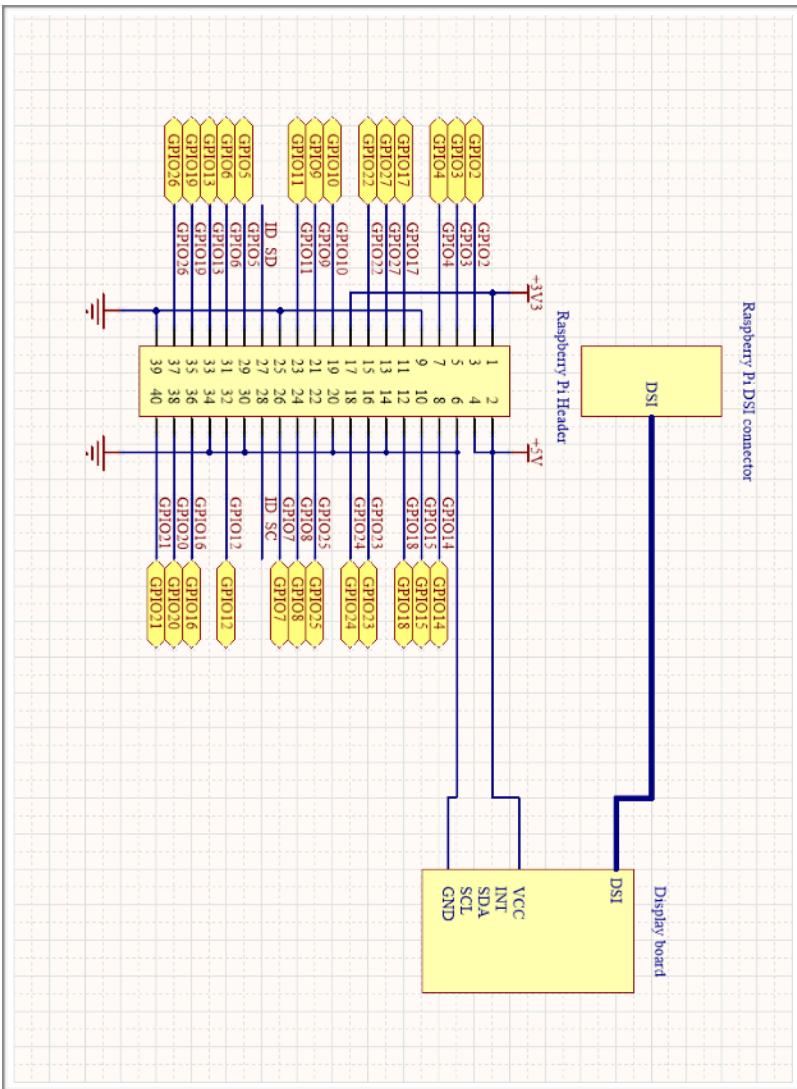
b) Ekran dotykowy

Do interakcji z urządzeniem zastosowano 7" ekran dotykowy dedykowany do Raspberry Pi. Charakteryzuje się on rozdzielcością 800 x 480 co przekłada się na zagęszczenie pikseli ~133PPI. Jest to wartość wystarczająca by przy typowym użytkowaniu pojedynczy piksel był nie do odróżnienia. Zastosowany ekran wykorzystuje technologię pojemnościową, która do rozpoznawania dotyku wykorzystuje zmianę pojemności elektrycznej, a nie nacisk palca dzięki czemu górną warstwą wyświetlacza może być wykonana ze szkła. Przekłada się to na trwałość oraz łatwość czyszczenia produktu. Kolejną zaletą wyświetlacza jest wsparcie dla technologii *multi-touch* oraz zintegrowanej z systemem klawiatury ekranowej zapewniającej pełną funkcjonalność bez fizycznej myszy i klawiatury.

Do komunikacji ekran wykorzystuje obsługujący przez Raspberry Pi szeregowy interfejs DSI. W tym celu między urządzeniami został podłączony 15-stykowy kabel taśmowy. Zapewnia on jednoczesny przesył obrazu oraz informacji z panelu dotyковego. Ekran zasilany jest napięciem 5V dostarczonym z 40-sto stykowego złącza Raspberry Pi. Pozostałe wyprowadzenia na płytce wyświetlacza pozostały niepodłączone. Dzięki dopasowanym otworom montażowym minikomputer został przymocowany w tylnej, niewidocznej części ekranu tworząc jednolite urządzenie.

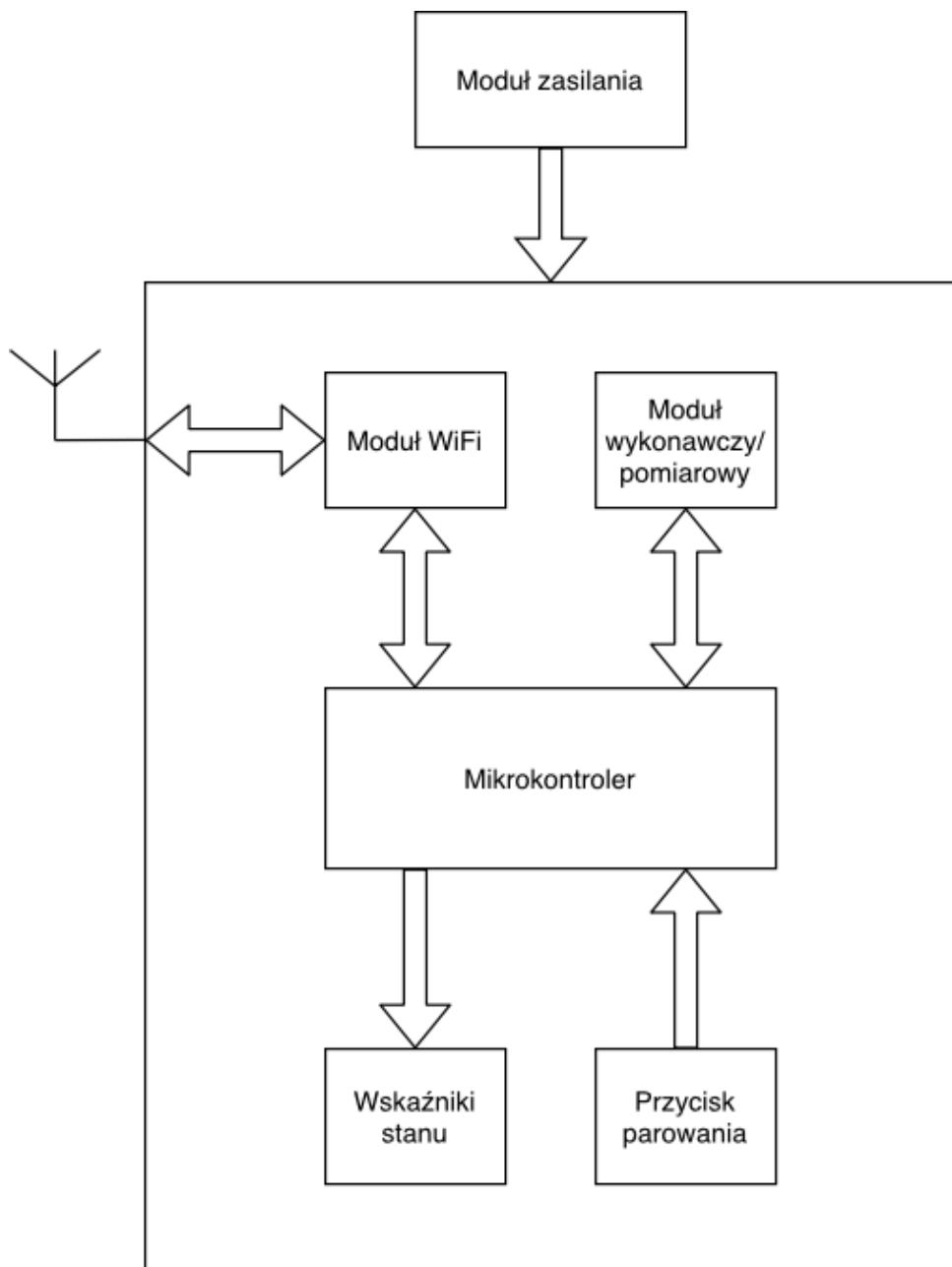
c) Zasilanie

Minikomputer zasilany jest napięciem 5V poprzez zewnętrzny zasilacz podłączony do gniazda USB typu C. Wykorzystanie standardu USB zapewnia kompatybilność z wszechobecnymi zasilaczami np. do telefonów komórkowych. Dzięki temu koszty tego urządzenia są niewielkie a wymiana w przypadku ewentualnej awarii jest bezproblemowa. Minimalna zalecana przez producenta wydajność prądowa zasilacza wynosi 3A.



Rys. 3.1.3: Schemat podłączenia urządzenia bazowego

3.2 Urządzenia końcowe

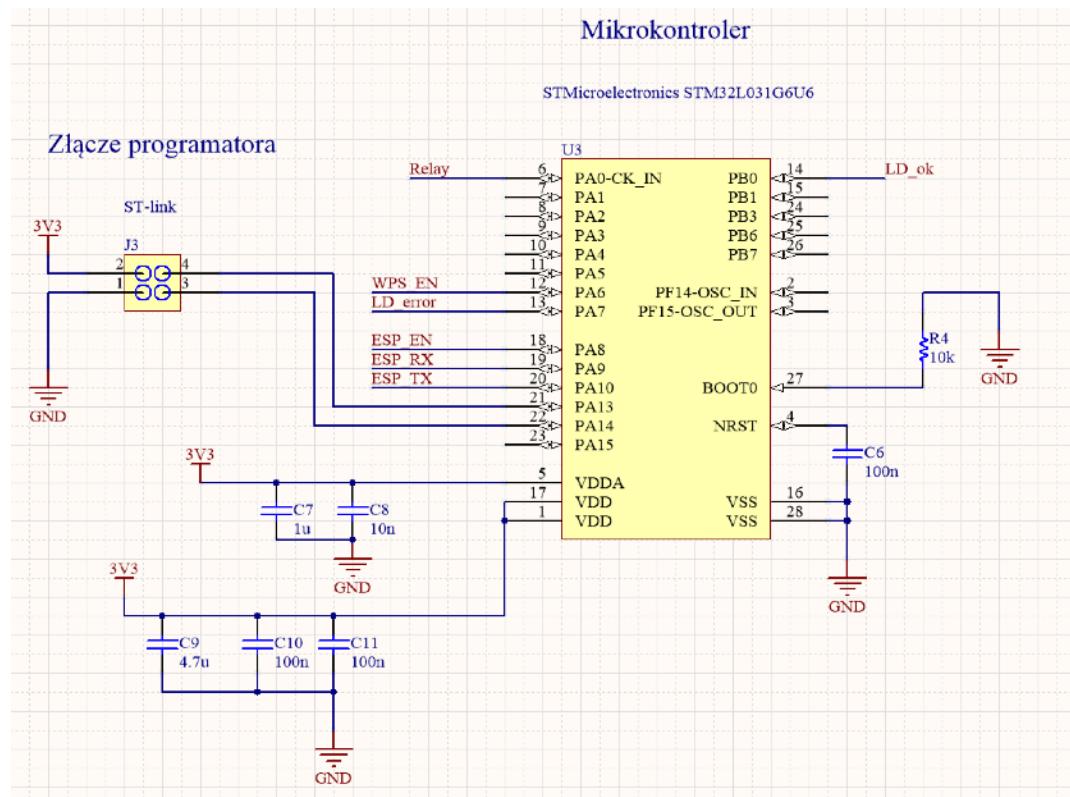


Rys. 3.2.1: Schemat blokowy urządzeń końcowych

Koncepcję działania wszystkich urządzeń końcowych przedstawiono na jednym schemacie blokowym (Rys. 3.2.1), ponieważ współdzielą one większość układów a zasada ich działania jest zbliżona. Różnią się jedynie modułem wykonawczym/pomiarowym oraz modułem zasilania - w zależności od spełnianej przez urządzenie funkcji. Ponowne wykorzystanie elementów pozwoliło zaoszczędzić czas przy projektowaniu części sprzętowej jak i programowej. Ułatwia to również wprowadzania poprawek i ulepszeń do produktu.

a) Mikrokontroler (Rys. 3.2.2)

Urządzenia końcowe zbudowano na bazie mikrokontrolerów wykorzystujących architekturę opracowaną przez firmę ARM (ang. *Avanced RISC Machine*). Układy te charakteryzują się wysoką wydajnością przy jednocześnie niskim poborze energii, dzięki czemu zdobyły bardzo dużą popularność na rynku urządzeń intelligentnych. Firma ARM jedynie licencjonuje technologię niezbędną do wykonania rdzeni procesorów, dlatego wybrano mikrokontroler produkowany przez STMicroelectronics - STM32L031. Seria L0 charakteryzuje się niskim zużyciem energii, rozbudowanymi funkcjami I/O oraz konkurencyjną ceną. Mikrokontroler zasilany jest napięciem 3.3V, na wszystkich liniach zasilających zastosowano kondensatory blokujące o standardowej pojemności 100nF [18]. W sekcji analogowej układu wykorzystano dwa, szeregowo podłączone kondensatory - C7 (1uF [18]) i C8 (100nF [18]), w celu poprawy precyzji działania przetwornika analogowo-cyfrowego. By zapewnić możliwość programowania urządzenia wprowadzono połączenia do zewnętrznego programatora/debuggera STlink.



Rys. 3.2.2: Schemat podłączenia mikrokontrolera

b) Wskaźniki stanu (Rys. 3.2.3)

Aby umożliwić wyświetlanie użytkownikowi informacji o stanie urządzenia zastosowano dwie diody elektroluminescencyjne - czerwoną oraz zieloną. Dokładny opis wyświetlanych sygnałów świetlnych znajduje się w rozdziale 4 - opis części programowej. Diody podłączono do mikrokontrolera poprzez rezystory R3, R5 ($1\text{k}\Omega$), prąd przewodzenia został obliczony ze wzoru 3.1, przyjmując napięcie zasilania 3,3V:

- dioda zielona ($UF = 2\text{V}$) $IF = 1,3\text{mA}$
- dioda czerwona ($UF = 1,75\text{V}$) $IF = 1,55\text{mA}$

$$IF = \frac{VDD - UF}{R} \quad (3.1)$$

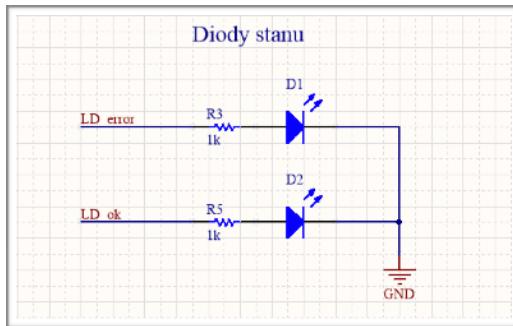
Gdzie:

IF - prąd przewodzenia diody

VDD - napięcie zasilania

R - rezystancja

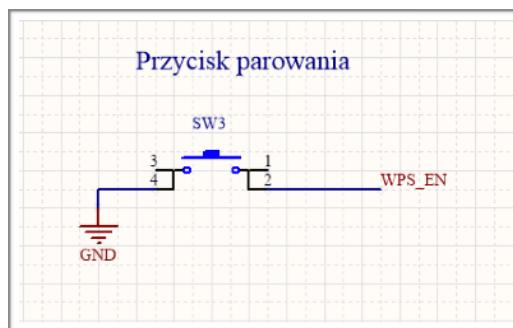
UF - napięcie przewodzenia diody



Rys. 3.2.3: Schemat podłączenia wskaźników stanu

c) Przycisk parowania (Rys. 3.2.4)

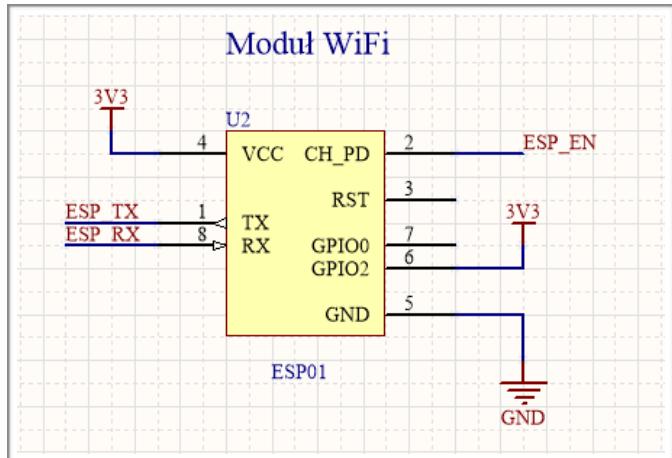
W celu uruchomienia procesu parowania urządzeń końcowych zastosowano przycisk typu *micro switch*. Został on bezpośrednio podłączony do wyprowadzeń mikrokontrolera, ponieważ eliminacji drgań styków dokonano w oprogramowaniu dzięki czemu zaoszczędzono miejsce na płytce drukowanej.



Rys. 3.2.4: Schemat podłączenia przycisku parowania

d) Moduł Wi-Fi (Rys. 3.2.5)

Aby umożliwić urządzeniom końcowym dostęp do sieci Wi-Fi, a tym samym komunikacje z urządzeniem bazowym, wyposażono je w moduły ESP-01, oparte na układzie EPS8266, który działa w standardzie Wi-Fi 802.11 b/g/n na częstotliwości 2,4GHz. W zaprojektowanych urządzeniach, obsługa częstotliwości 5GHz jest zbędna, ponieważ przesyłana jest mała ilość danych, większe znaczenie ma zasięg transmisji. Do komunikacji modułu z mikrokontrolerem wykorzystano interfejs UART. Dodatkowo, do mikrokontrolera zostało podłączone wyprowadzenie CH_PD (ang. *Chip Power Down*) w celu zapewnienia możliwości wyłączenia układu.

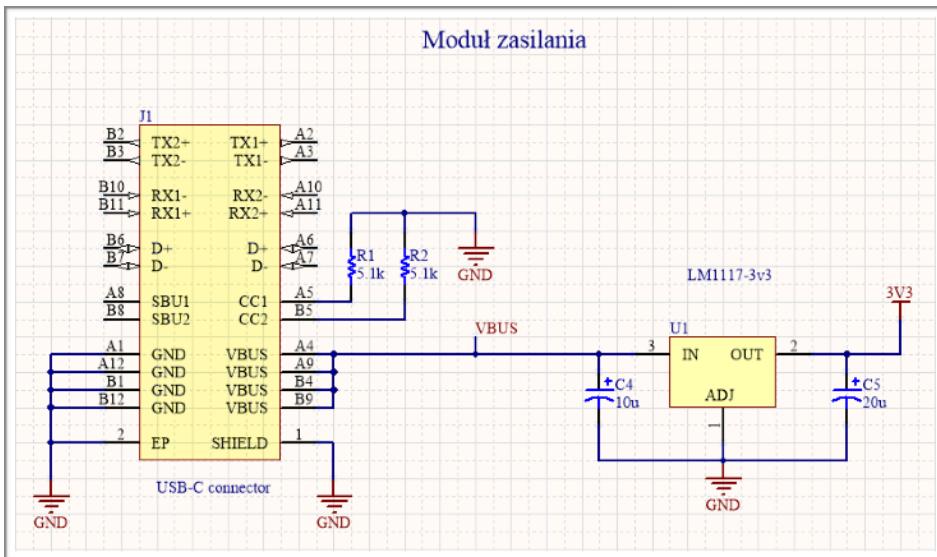


Rys. 3.2.5: Schemat podłączenia moduł Wi-Fi []

e) Moduł zasilania nr 1 (Rys. 3.2.6)

Moduł ten wykorzystano do zasilania dwóch urządzeń: inteligentnego oświetlenia RGB oraz czujnika temperatury. Składa się on z zewnętrznego zasilacza podłączonego do zainstalowanego na płytce PCB złącza USB typu C oraz liniowego regulatora napięcia. Największą zaletą zastosowania standardu USB jest kompatybilność z obecnymi na rynku przewodami i ładowarkami oraz zapewnienie wsparcia produktu w przyszłości - można wykorzystać ten sam zasilacz co dla Raspberry Pi.

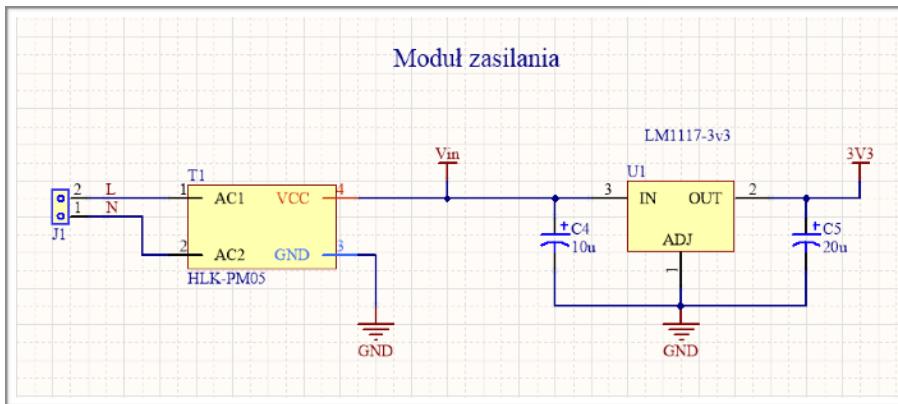
W celu włączenia funkcji ładowania 5V (maksymalna wydajność prądowa zależy od zastosowanego zasilacza), wyprowadzenia CC1 oraz CC2 złącza USB podłączono do masy układu poprzez rezystory polaryzujące o wartości $5,1\text{k}\Omega$ [19]. Ponieważ mikrokontroler oraz moduł Wi-Fi są zasilane napięciem 3,3V, wykorzystano liniowy regulator LM1117-3.3. Charakteryzuje się on małymi rozmiarami oraz niskim spadkiem napięcia, dzięki czemu nie wymaga zastosowania dodatkowego radiatorsa. Jego maksymalna wydajność prądowa wynosi 800mA, co jest wartością wystarczającą do zasilania układów. W celu filtrowania napięcia wejściowego oraz zapewnienia stabilizacji regulatora, zastosowano dwa kondensatory tantalowe według specyfikacji producenta: C4 o pojemności 10uF [20] na wejściu regulatora, C5 o pojemności 20uF [20] na wyjściu regulatora.



Rys. 3.2.6: Schemat podłączenia modułu zasilania nr 1

f) Moduł zasilania nr 2 (Rys. 3.2.7)

Moduł ten wykorzystano do zasilania inteligentnej wtyczki, która powinna być zainstalowana pomiędzy gniazdkiem, a urządzeniem sterowanym, dlatego najlepszym rozwiązaniem zasilania jest zainstalowanie zasilacza wewnętrz urządzienia. W projekcie wykorzystano układ przeznaczony do montażu na PCB HLK-PM05. Charakteryzuje się on szerokim zakresem napięć wejściowych - od 100V do 240V AC, napięciem wyjściowym 5V oraz maksymalną wydajnością prądową 1A. Podobnie jak w module 1, do zasilania mikrokontrolera STM32 oraz ESP-01 zastosowano liniowy regulator LM1117- 3.3V.



Rys. 3.2.7: Schemat podłączenia modułu zasilania nr 2

g) Moduł wykonawczy nr 1 (Rys. 3.2.9)

Ten moduł zastosowano w inteligentnym oświetleniu. Służy on do sterowania podłączoną do terminala J4 taśmą z diodami LED RGB. Zastosowano taśmę zasilaną napięciem 5V, w konfiguracji wspólnej katody. Regulacja natężenia oraz barwy oświetlenia odbywa się poprzez trzykanałowy sygnał PWM generowany przez mikrokontroler. Sygnał podłączono do wyprowadzeń taśmy kolejno: anody czerwonej; zielonej; niebieskiej poprzez klucze tranzystorowe. Zdecydowano się na użycie tranzystorów polowych z izolowaną bramką DMN1019, ponieważ charakteryzują się niską rezystancją przewodzenia ($R_{DS(ON)} = 10\text{m}\Omega$), co przekłada się na niską moc rozpraszana w postaci ciepła. Wybrany element porównano z tranzystorem bipolarnym PN2222A w tabeli 3.1. Moc wydzielaną przez pojedynczy tranzystor oraz temperaturę złącza obliczono kolejno ze wzorów 3.2 oraz 3.3. Do obliczeń przyjęto taśmę LED o długości 1m z zabezczaniem 30 diód na metr, dla której maksymalny prąd wynosi $I = 0.5\text{A}$.

Tab. 3.1: Porównanie tranzystorów PN2222A oraz DMN1019

	P_d	$T_J [^\circ\text{C}]$
DMN1019	5mW	25,8°C
PN2222A	500mW	125°C

$$P_D = I_D^2 \times R_{DS(ON)} \quad P_D = U_{CE(sat)} \times I_E \quad (3.2)$$

Gdzie:

P_d - moc rozpraszana

I_D - prąd drenu

$R_{DS(on)}$ - rezystancja dren - źródło

$U_{CE(sat)}$ - napięcie saturacji kolektor - emiter

I_E - prąd emitera

$$T_J = T_A + P_d \times R_{\theta JA} \quad (3.3)$$

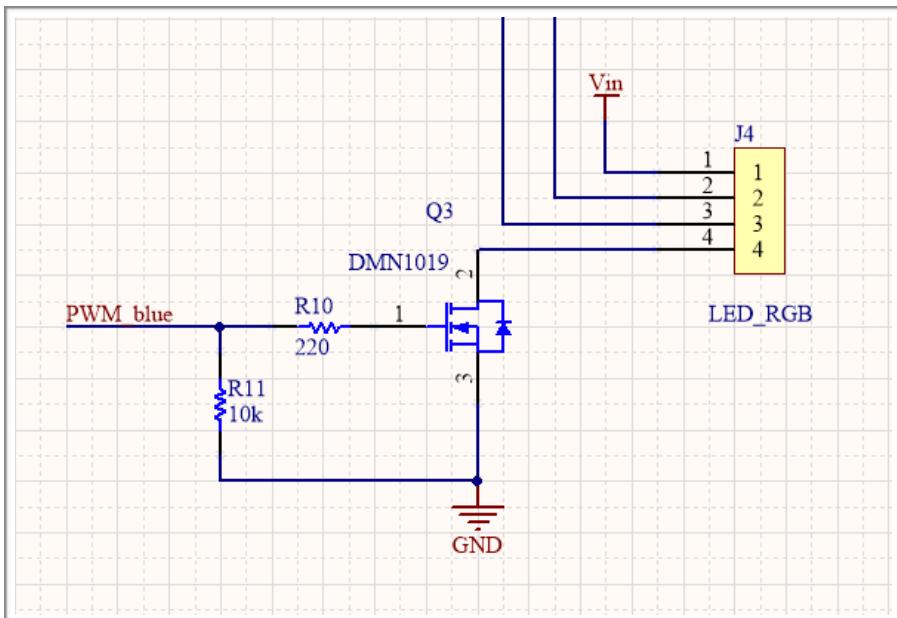
Gdzie:

T_A - temperatura otoczenia

T_J - temperatura złącza

$R_{\theta JA}$ - rezystancja termiczna złącze - otoczenie

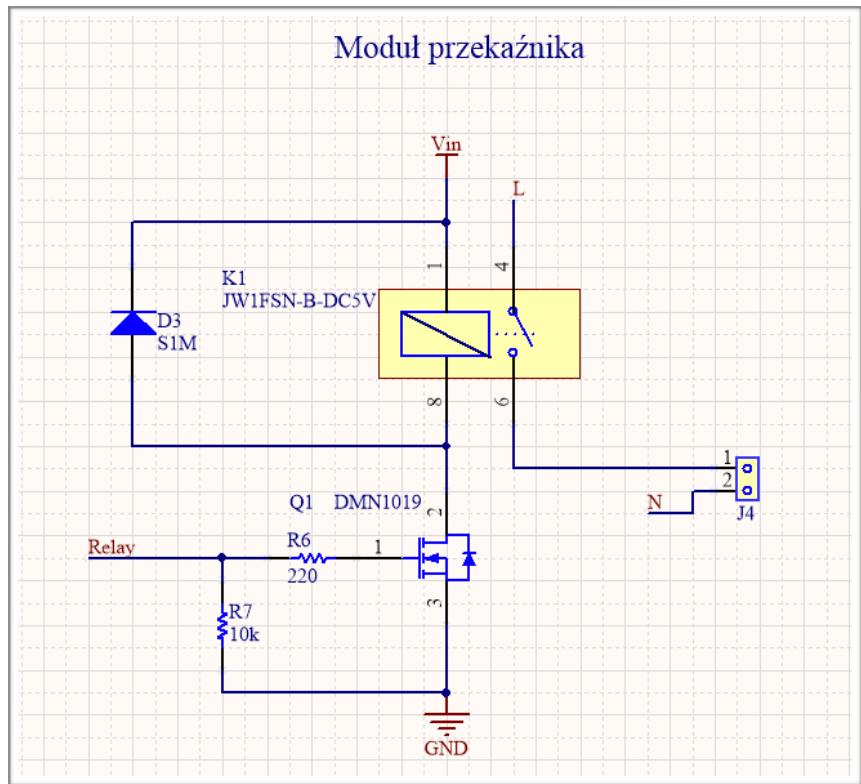
Na schemacie zastosowano rezystor polaryzujący R11, w celu utrzymania tranzystora w stanie nienasycenia podczas inicjalizacji mikrokontrolera. Aby ograniczyć maksymalny prąd bramki przy włączaniu tranzystora, zastosowano rezystor R10.



Rys. 3.2.9: Częściowy schemat podłączenia modułu wykonawczego nr 1

h) Moduł wykonawczy nr 2 (Rys. 3.2.10)

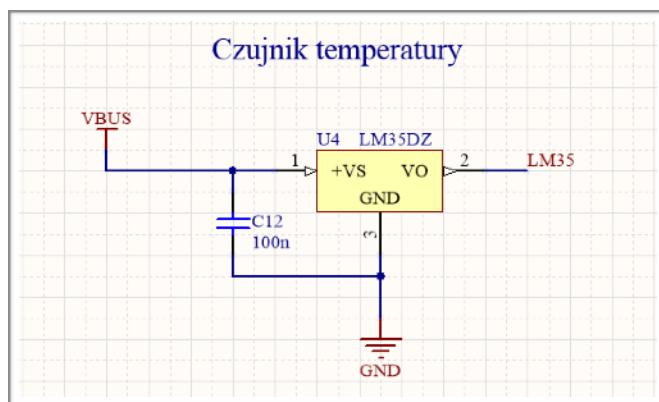
Moduł wykonawczy nr 2 zastosowano w inteligentnej wtyczce. Służy on do sterowania zasilaniem urządzeń podłączonych do wtyczki. Moduł składa się z przekaźnika podłączonego do mikrokontrolera poprzez klucz tranzystorowy. Zapewnia to odpowiednią izolację od sieci elektrycznej oraz zabezpieczenie na wypadek awarii układu sterowania, ponieważ przekaźnik działa w trybie normalnie otwartym bez podtrzymywania. Zastosowano standardowy przekaźnik z cewką 5V sterowany poprzez klucz tranzystorowy. Wykorzystano ten sam tranzystor polowy co w module wykonawczym nr 1 - DMN1019. Aby podczas wyłączania cewki skok napięcia nie zniszczył tranzystora zastosowano diodę D3, która rozładowuje energię zgromadzoną w cewce. Styki cewki wyprowadzono do terminala śrubowego J4, co umożliwia podłączenie wtyczki sieciowej.



Rys. 3.2.10: Schemat podłączenia modułu wykonawczego nr 2

i) Moduł pomiarowy (Rys. 3.2.11)

Moduł pomiarowy zbudowano w oparciu o analogowy czujnik temperatury LM35DZ, który w celu odczytywania pomiarów podłączono do 12 bitowego przetwornika analogowo-cyfrowego, wbudowanego w mikrokontroler. Dokładność tego rozwiązania zależy od jakości napięcia zasilania dlatego zastosowano kondensator filtrujący C_{12} o pojemności 100nF [21].



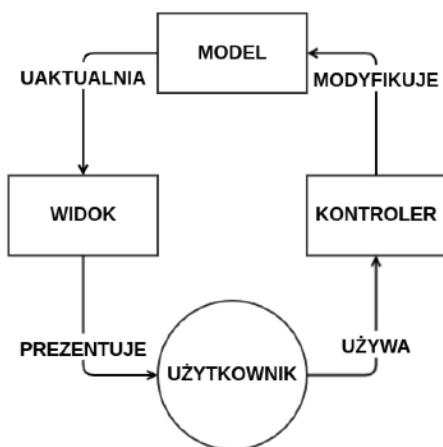
Rys. 3.2.11: Schemat podłączenia modułu pomiarowego

Rozdział 4

Opis części programowej

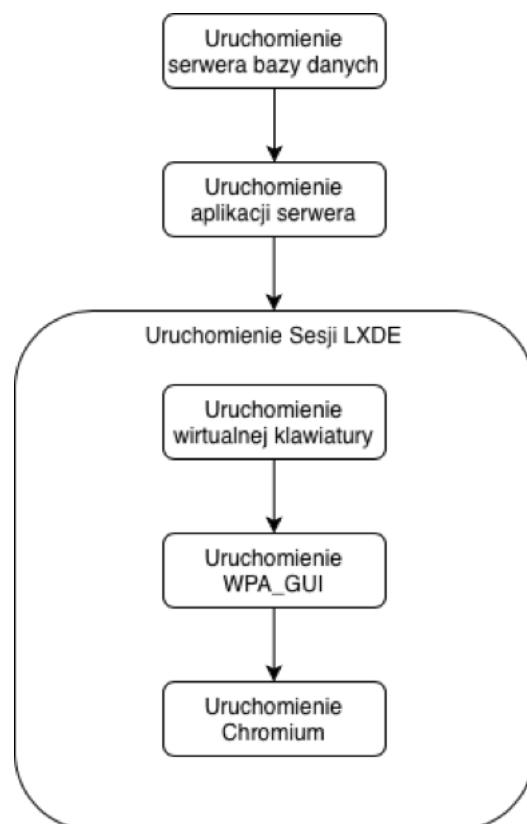
4.1 Urządzenie bazowe

Projektując aplikację do obsługi intelligentnego systemu skorzystano z wzorca architektonicznego *MVC* (Model Widok Kontroler) (Rys. 4.1.1). Dzięki temu, oddzielono dane używane w programie od komponentu wizualnego - aplikacji klienckiej. Model oraz widok oddziałuje tylko z kontrolerem zawierającym logikę programu. Zaletą tego rozwiązania jest łatwa rozbudowa programu o nowe funkcjonalności. Umożliwia to również generowanie niezależnych od siebie widoków. W dalszych podpunktach zawarto dokładne opisy poszczególnych elementów architektury.



Rys. 4.1.1: *Model Widok Kontroler*

Na rysunku 4.1.2 przedstawiono proces uruchamiania programów, które są niezbędne do działania inteligentnego systemu. Oprogramowanie uruchamiane jest automatycznie po włączeniu zasilania Raspberry Pi dzięki wykorzystaniu serwisów oraz skryptów startowych. W pierwszym kroku zostaje uruchomiona baza danych oraz aplikacja serwera. Następnie, ładowany jest skrypt LXSession uruchamiający aplikacje w LXDE - środowisku do obsługi aplikacji graficznych, cechujący się szybkością oraz energooszczędnością. W LXDE jako pierwsza uruchomiona zostaje aplikacja wirtualnej klawiatury obsługująca wprowadzanie tekstu na ekranie dotykowym. Następnie, ładowany jest program umożliwiający połączenie urządzenia z domową siecią WiFi - WPA_GUI. Po wyłączeniu okna WPA_GUI otwierana jest przeglądarka internetowa Chromium prezentująca aplikację kliencką. Aby zablokować użytkownikowi możliwość swobodnego przeglądania internetu, przeglądarkę uruchomiono w trybie kiosku, który ukrywa pasek adresu oraz uniemożliwia wejście w ustawienia. Dodatkowo, środowisko LXDE zostało ustawione tak by nie włączać wygaszacz ekranu oraz ukrywać wskaźnik myszy.



Rys. 4.1.2: Algorytm uruchamiania oprogramowania

a) Model

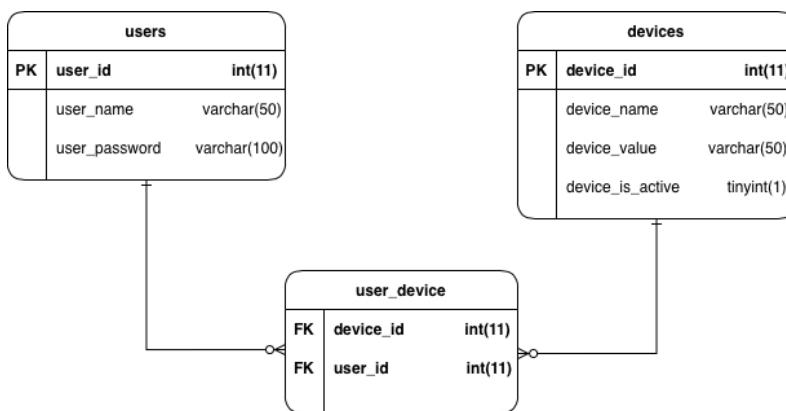
Model jest centralnym elementem architektury. Jest to dynamiczna struktura danych, niezależna od interfejsu użytkownika. W zaprojektowanej aplikacji rolę modelu pełni baza danych uruchomiona na lokalnym serwerze MySQL.

W bazie danych utworzone zostały trzy tabele przedstawione na rysunku 4.1.3.

Users - Tabela users służy do przechowywania informacji o utworzonych użytkownikach - unikatowej nazwy użytkownika oraz hasła. W celu bezpiecznego przechowywania hasła zapisywane są w formie nieodwracalnego skrótu (ang. *hash*). Klucz główny (ang. *Primary Key*) tabeli generowany jest automatycznie podczas dodawania użytkownika.

Devices - W tabeli devices zapisywane są wszystkie dodawane do systemu urządzenia. Każde posiada przypisaną nazwę, wartość oraz stan aktywności. Urządzenie identyfikowane jest unikatowym kodem nadawanym podczas programowania, dzięki czemu system może rozpoznawać jego typ. Rodzaj urządzenia zapisywany jest na ostatnich cyfrach kodu np. intelligentna wtyczka identyfikowana jest "0".

User devices - W celu zapisywania związku między urządzeniami, a użytkownikami (relacja wiele do wielu) utworzona została tabela, w której przechowywane są klucze obce (ang. *Foreign Key*) z tabel users oraz devices. Dzięki tej tabeli każde urządzenie może być przypisane wielokrotnie do różnych użytkowników.



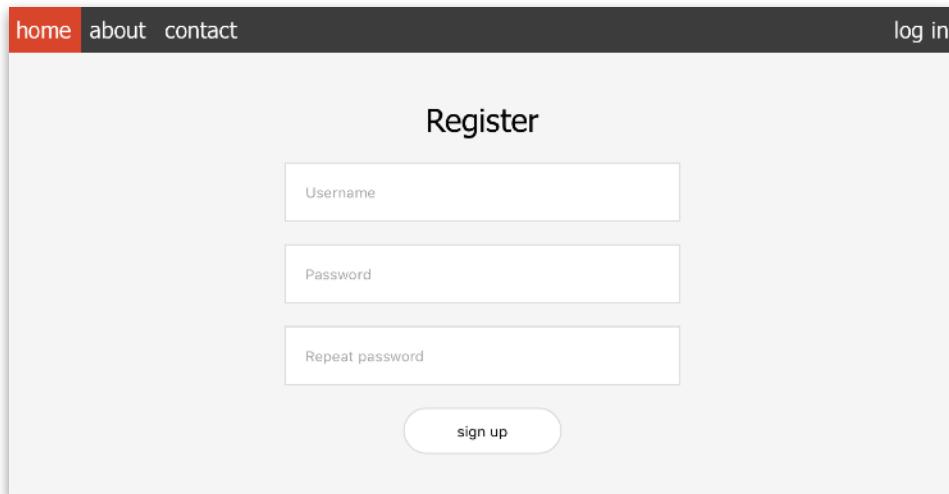
Rys. 4.1.3: Model bazy danych

b) Widok

Rolę widoków, a tym samym interfejsu użytkownika, pełni strona WWW uruchomiona w przeglądarce urządzenia klienckiego. Ważnymi punktami przy jej tworzeniu były: aspekt wizualny oraz zapewnienie responsywności. Ponieważ strona może być wyświetlana na urządzeniach mobilnych z niewielkim ekranem oraz na komputerach osobistych. Starano się również, by strona była niewielkich rozmiarów a tym samym szybko się wczytywała, dlatego nie użyto zewnętrznych bibliotek jak jQuery czy Bootstrap. Pożądane efekty udało się uzyskać dzięki wykorzystaniu funkcji wbudowanych w języki JavaScript oraz CSS.

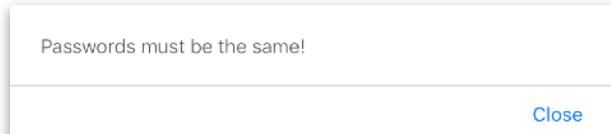
W zaprojektowanym systemie można wyróżnić cztery główne widoki:

Widok rejestracji (Rys. 4.1.4) - Składa się z formularza HTML pozwalającego wysłać żądanie HTTP do serwera o stworzenie nowego użytkownika. Na stronie znajduje się również skrypt, który pełni dwie funkcje - weryfikuje poprawność hasła i nazwy użytkownika (hasła w obu polach muszą się zgadać; hasło musi składać się z conajmniej 5 znaków; podana nazwa użytkownika nie może istnieć w bazie danych) oraz w przypadku poprawnej rejestracji przekierowuje użytkownika do widoku głównego. W razie niepowodzenia, użytkownikowi zostaje wyświetlony odpowiedni komunikat w postaci alertu JavaScript (Rys. 4.1.5).



The screenshot shows a user registration page. At the top, there is a navigation bar with links for 'home', 'about', and 'contact', and a 'log in' link on the right. The main content area has a title 'Register'. Below the title are three input fields: 'Username', 'Password', and 'Repeat password'. A 'sign up' button is located at the bottom of the form. The 'Username' field is currently selected, indicated by a red border around its input box.

Rys. 4.1.4: Widok rejestracji użytkownika



Rys. 4.1.5: Przykładowy komunikat niepowodzenia

Widok logowania (Rys. 4.1.6) - Prezentuje użytkownikowi formularz HTML umożliwiający wysłanie żądania o zalogowanie się do systemu. W przypadku poprawnej weryfikacji użytkownik zostaje przekierowany na widok główny. W razie niepowodzenia wyświetlany jest odpowiedni komunikat.

A screenshot of a login page. At the top, there is a dark header bar with white text. On the left, it says "home" (with a red background), "about", and "contact". On the right, it says "sign up". Below the header, the word "Login" is centered above a form. The form consists of two input fields: "Username" and "Password", separated by a horizontal line. Below the fields is a "log in" button with a rounded rectangle and a thin border.

Rys. 4.1.6: Widok logowania

Widok urządzeń - Pozwala użytkownikowi na interakcję z dostępnymi dla niego urządzeniami. Urządzenia wyświetlane są kategoriami obok siebie (Rys. 4.1.7) lub pod sobą (Rys. 4.1.8), w zależności od szerokości ekranu. Efekt ten uzyskano dzięki regule `@media max-width` oraz kontenerom *Flexible Box* wykorzystanych w stylach strony.

Do pobierania informacji o urządzeniach z serwera wykorzystano asynchroniczne żądania HTTP (AJAX), dzięki czemu widok może być odświeżany dynamicznie, bez przeładowania strony. Dane pobierane są w formacie JSON (ang. *JavaScript Object Notation*). Widok informowany jest o zmianach w modelu przy użyciu technologii *WebSocket* - synchronicznego kanału komunikacyjnego między klientem a serwerem, dzięki czemu zawsze jest aktualny.

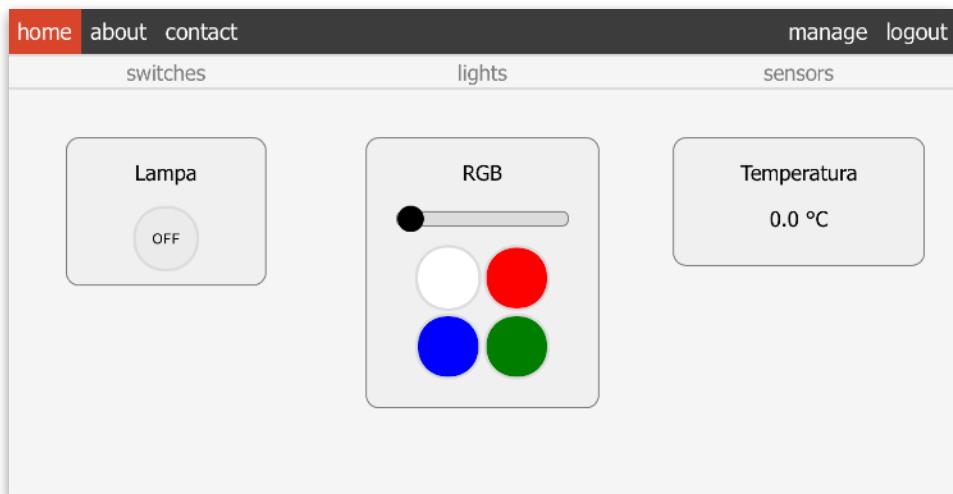
Dynamiczne odświeżanie widoku urządzeń ma szczególne znaczenie, gdy w tym samym czasie z systemu korzysta więcej niż jedna osoba.

Dla każdego urządzenia wykonawczego generowany jest odpowiedni element pozwalający na interakcję: przycisk ON/OFF w przypadku inteligentnej wtyczki; suwak do zmiany jasności i przyciski pozwalające na wybór koloru dla oświetlenia RGB. Użytkownik informowany jest o aktualnym stanie urządzeń dzięki akcentom graficznym jak np. szary kolor tła w przypadku zaniku aktywności w sieci. Przykładowy kod generujący przycisk przedstawiono na rysunku 4.1.9. Każdemu elementowi interaktywnemu, przypisana została funkcja obsługi zdarzeń: odpowiednio *onclick* dla przycisków oraz *onchange* dla suwaka. Jej zadaniem jest informowanie kontrolera o zmianie stanu urządzenia. Odbywa się to za pomocą technologii *WebSocket*. Dane przesyłane są w formacie JSON. Przykładowa wiadomość dla oświetlenia RGB:

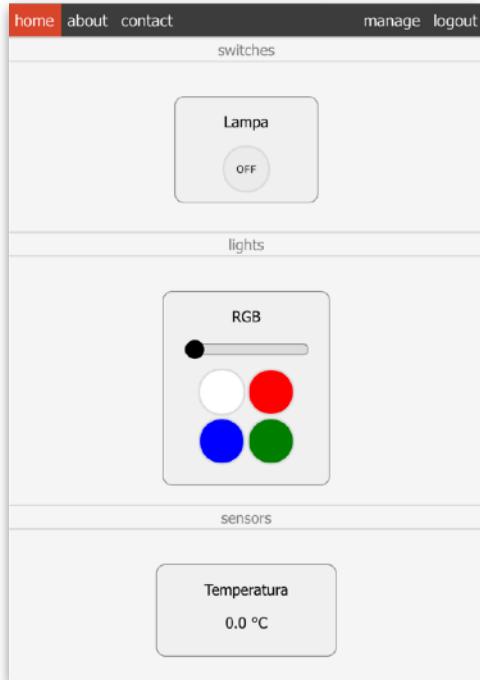
```
{'device_id': "22", 'device_value': "1099"} - ostatnia cyfra "2" w device_id oznacza typ urządzenia. W polu device_value zakodowane zostały informacje o stanie urządzenia: "1" - włączone oświetlenie; 0 - kolor biały; 99 - natężenia oświetlenia.
```

Kod obsługi zdarzenia *onclick* dla przycisku inteligentnej wtyczki przedstawiono na rysunku 4.1.10.

Dla urządzeń pomiarowych generowany jest element wyświetlający aktualny pomiar wraz z odpowiednią jednostką. Dodatkowo informacje z sensorów odświeżane są periodycznie - co 2 sekundy.



Rys. 4.1.7: Widok urządzeń - poziomo



Rys. 4.1.8: Widok urządzeń - pionowo

```
function createSwitch(element) {
    let button = document.createElement('button');
    let div = document.createElement('div');
    let para = document.createElement('p');
    if (element.device_is_active == 0) {
        div.className = 'switch disabled';
        button.disabled = true;
    }
    else{
        div.className = 'switch enabled';
    }
    para.textContent = element.device_name;
    button.value = element.device_value;
    button.textContent = parseInt(element.device_value) ? 'ON' : 'OFF';
    button.id = element.device_id;
    button.onclick = switchClick;
    div.appendChild(para);
    div.appendChild(button);
    devices_div[0].appendChild(div);
}
```

Rys. 4.1.9: Funkcja generująca przycisk inteligentnej wtyczki

```

function switchClick() {
    if (parseInt(this.value)) {
        this.textContent = 'OFF';
        this.value = 0;
    }
    else {
        this.textContent = 'ON';
        this.value = 1;
    }
    socket.send(formatData(this.id, this.value));
    console.log(formatData(this.id, this.value));
}

```

Rys. 4.1.10: Funkcja obsługująca zdarzenie onclick przycisku inteligentnej wtyczki

Widok zarządzania (Rys. 4.1.11) - Umożliwia dokonywanie czynności administracyjnych w systemie, dostęp do niego ma jedynie administrator. Składa się z tabeli, w której w pierwszej kolumnie pokazane są dostępne urządzenia, a w pierwszym wierszu wszyscy użytkownicy. Na skrzyżowaniu tych pól generowane są przyciski wyboru (ang. *check box*), które jednocześnie pozwalają połączyć użytkownika z urządzeniem oraz prezentują aktualny stan relacji. Widok ten pozwala również na edycję urządzeń - zmianę nazwy oraz usunięcie z systemu. Odbywa się to za pomocą okna dialogowego typu *prompt* pozwalającego na podanie nowej nazwy. W celu usunięcia urządzenia należy nadać mu nazwę “delete”. Opcja ta może wydawać się mało intuicyjna, jednak w systemach inteligentnych urządzenia usuwane są rzadko. Oprócz edycji urządzeń w widoku tym możliwe jest usuwanie kont użytkowników. Odbywa się to poprzez kliknięcie nazwy użytkownika w tabeli, administrator zostanie poproszony o potwierdzenie decyzji dzięki oknu dialogowemu typu *confirm*.

Tabela zarządzania, podobnie jak lista urządzeń w widoku głównym generowana jest dynamicznie na podstawie odbieranych informacji o urządzeniach i użytkownikach (Rys. 4.1.12). Dane pobierane są z serwera w formacie JSON dzięki asynchronicznym żądaniom HTTP. W widoku tym nie wykorzystano technologii WebSocket, ponieważ w danym momencie korzysta z niego tylko jeden użytkownik (administrator) - jest jedynym miejscem wprowadzającym zmiany do modelu. Każdemu elementowi do edycji danych przypisana została funkcja obsługi zdarzeń *onclick*. Jest ona odpowiedzialna za wyświetlenie odpowiednich dialogów oraz za wysłanie do serwera żądania zmiany danych w modelu. Przykładowa wiadomość żądania zmiany nazwy urządzenia wysłana pod adres “devices/names” metodą POST: {device_id: “identyfikator urządzenia”, device_name: “nowa nazwa” }.

home	about	contact		manage	logout
	Andreas	user	user1	user3	
Lampa	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	
RGB	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
Temperatura	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Rys. 4.1.11: Widok zarządzania

```

function drawTable(users, devices, userDevices) {
    let table = document.querySelector('table');
    let usersRow = table.insertRow();
    let firstCell = usersRow.insertCell();
    let text = document.createTextNode("");
    firstCell.appendChild(text);

    //Utworzenie wiersza z użytkownikami
    users.forEach(user => {
        let cell = usersRow.insertCell();
        let text = document.createTextNode(user.user_name);
        cell.id = user.user_id;
        cell.onclick = deleteUser;
        cell.appendChild(text);
    });

    //Utworzenie kolumny z urządzeniami
    devices.forEach(device => {
        let row = table.insertRow();
        let cell = row.insertCell();
        let text = document.createTextNode(device.device_name);
        cell.id = device.device_id;
        cell.onclick = editDevice;
        cell.appendChild(text);
    });

    //Utworzenie komórek z przyciskami wyboru
    for (let row of table.rows) {
        if (row.cells.length > 1) {
            continue;
        }
        for (let step = 1; step <= users.length; step++ ) {
            let device_id = row.cells[0].id;
            let user_id = table.rows[0].cells[step].id;
            let cell = row.insertCell();
            cell.id = user_id;
            let checked = checkIfUserDevice(user_id, device_id, userDevices);
            let checkBox = createCheckBox(checked, user_id, device_id);
            cell.appendChild(checkBox);
        }
    }
}

```

Rys. 4.1.12: Funkcja generująca tabelę zarządzania

Pasek menu (Rys. 4.1.13) - w celu poruszania się między widokami zastosowano pasek menu na którym umieszczone zostały przyciski z hiperlinkami do odpowiednich stron.



Rys. 4.1.13: Pasek menu

c) Kontroler

Kontroler odpowiedzialny jest za przetwarzanie danych oraz sterowanie aplikacją - obsługuje żądania pochodzące od użytkowników oraz pośredniczy w przesyłaniu danych pomiędzy widokiem, modelem i urządzeniami końcowymi. Jego głównymi elementami są trzy serwery - serwer WWW i WebSocket do obsługi aplikacji klienckiej oraz serwer TCP do komunikacji z urządzeniami wykonawczymi/pomiarowymi. Oprogramowanie kontrolera zostało stworzone przy użyciu środowiska Node.js. Środowisko to, służy do uruchamiania oprogramowania napisanego w języku JavaScript poza przeglądarką internetową. Node.js charakteryzuje się nieblokującymi funkcjami wejścia/wyjścia i asynchroniczną obsługą zdarzeń, dzięki czemu przetwarzanie żądań odbywa się bez opóźnień - przekłada się to na responsywność zaprojektowanej aplikacji. Dodatkową zaletą wykorzystania środowiska Node.js jest unifikacja języka programowania strony serwerowej i klienckiej.

Obsługa modelu - Kontroler korzysta z modelu dzięki wbudowanemu w środowisko Node.js API do obsługi serwera MySQL oraz napisaną funkcją dostępu/edycji tabel. Przykładową funkcję usuwającą powiązanie użytkownika z urządzeniem przedstawiono na rysunku 4.1.14.

```
function deleteUserDevice(user_id, device_id) {
  connection.query('DELETE FROM user_device WHERE device_id = ? AND user_id = ?',
  [device_id, user_id], (error, result) => {
    if (error){
      console.log(error)
    }
    else{
      console.log(`Removed device ${device_id} from user ${user_id}`)
    }
  });
}
```

Rys. 4.1.14: Funkcja usuwająca powiązanie użytkownika z urządzeniem

Serwer WWW - Zadaniem serwera WWW jest odpowiadanie na żądania klientów do określonych punktów końcowych (ang. *endpoint*). W tym celu zdefiniowane zostały funkcje odpowiadające metodom HTTP (GET, POST, DELETE). Odpowiedź serwera na żądania ściśle zależy od stanu zalogowania się klienta - niezalogowany ma dostęp do widoku rejestracji oraz logowania, zalogowany użytkownik może obsługiwać przypisane mu urządzenia, a tylko administrator ma możliwość wprowadzania zmian w systemie oraz dostęp do wszystkich urządzeń. Na rysunku 4.1.13 przedstawiono funkcję obsługującą żądanie metody GET pobrania listy urządzeń.

```

router.get('/devices', (request,response) => {
    if (request.session.username == "admin") {
        database.findDevices(results=>{response.json(results)});
    }
    else if (req.session.loggedin) {
        database.findUserDevices(request.session.username, results=>{response.json(results)});
    }
    else {
        res.sendFile(path.join(__dirname + '/public/404.html'));
    }
});
```

Rys. 4.1.13: Funkcja obsługująca żądanie pobrania listy urządzeń

System uwierzytelniania w aplikacji oparto na sesji, w której klient identyfikowany jest przy użyciu otrzymanego od serwera unikalnego id (ang. *session id*) zapisywanego w pliku cookie. Podczas procesu logowania nazwa użytkownika oraz stan zalogowania zostają skojarzone z identyfikatorem sesji, dzięki czemu serwer może odpowiednio realizować żądania. Funkcję realizującą logowanie klienta przedstawiono na rysunku 4.1.14.

```

router.post('/login', (request,response) => {

    let username = request.body.username;
    let password = request.body.password;

    database.findUser(username, result => {
        if (result.length>0) {
            if (passwordHash.verify(password,result[0].user_password)) {
                request.session.loggedin = true;
                request.session.username = username;
                response.redirect('/home');
                console.log(` ${username} logged in! `);
            }
            else {
                response.redirect('/home/incorrect');
            }
        }
        else {
            response.redirect('/home/incorrect');
        }
    });
});
```

Rys. 4.1.14: Funkcja obsługująca żądanie zalogowania do systemu

Serwer WebSocket - Dzięki serwerowi WebSocket klienci mogą nawiązywać synchroniczne połączenie z kontrolerem, jest ono niezbędne do poprawnego działania aplikacji - zapewnia zsynchronizowanie widoków z modelem. Klienci obsługując widok listy urządzeń wysyłają wiadomość w stronę serwera, w której zakodowano identyfikator oraz nową wartość urządzenia. Dzięki tym informacjom kontroler może odpowiednio zmodyfikować model oraz przesyłać wiadomość do pożądanego urządzenia za pomocą połączenia TCP. Jedyną wiadomością wysyłaną w stronę klientów jest żądanie odświeżenia widoku. Funkcję obsługującą odebranie wiadomości przez serwer WebSocket pokazano na rysunku 4.1.15.

```
client.on('message', message => {
  let data = JSON.parse(message);
  database.changeDeviceValue(data.device_id, data.device_value);
  tcpServer.sendToDevice(data.device_id, data.device_value);
  websocketServer.sendToClients('update_view');
  console.log(`Data from client: ${data}`);
});
```

Rys. 4.1.15: Funkcja obsługująca odebranie wiadomości od klienta

Serwer TCP - Stacja bazowa komunikuje się z urządzeniami końcowymi dzięki uruchomionemu serwerowi TCP. Wiadomości przesyłane są w formie tekstowej, a następnie dekodowane są do formatu JSON. Serwer w zależności do odebranej wiadomości uruchamia procedurę parowania lub jedynie aktualizuje dane. Przy każdej wprowadzonej zmianie, do widoku zostaje wysłane żądanie odświeżenia listy urządzeń. Aktywne połączenia (obiekty socket) przechowywane są w mapie, w której kluczem jest *device_id*, dzięki czemu kontroler może przesyłać wiadomości z widoków do odpowiednich urządzeń. Każda wiadomość przed wysłaniem jest kodowana by ułatwić proces parsowania - na początek dodawany jest znak '\$' a na koniec CR (ang. carriage return).

Dokładny przebieg obsługi odebranej wiadomości przedstawiono na rysunku 4.1.16.

W celu identyfikacji nieaktywnych urządzeń, każdemu połączeniu TCP został przypisany *timeout* po upływie którego uruchamiana jest funkcja resetująca stan urządzenia w bazie danych (Rys. 4.1.17).

```
socket.on('data', message => {
    console.log(`From: ${socket.id} ${message}`);
    let data = decode(message);
    if (data.value === PAIRING) {
        socket.id = data.device_id;
        conn_map.set(data.device_id, socket);
        database.addDevice(data.device_id);
        database.activateModule(data.device_id);
        websocketServer.sendToClients('update_view');
    }
    else if (data.value) {
        database.changeDeviceValue(socket.id, data.device_id);
        websocketServer.sendToClients('update_view');
    }
});
```

Rys. 4.1.16: Funkcja obsługująca odebranie wiadomości od urządzenia końcowego

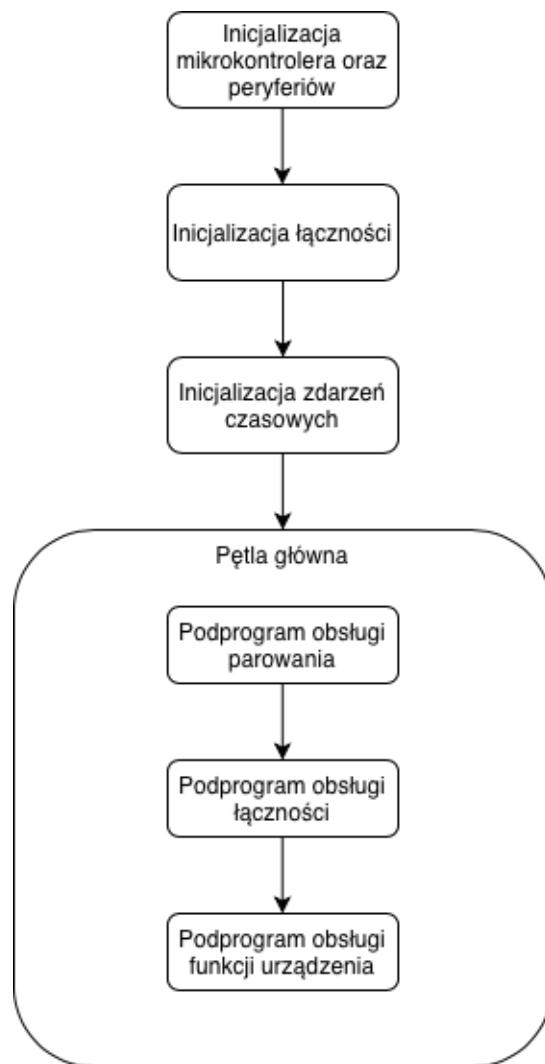
```
socket.on('timeout', () => {
    console.log(`Client: ${socket.id} left`);
    if (socket.id) {
        database.deactivateDevice(socket.id);
        database.resetDevice(socket.id);
        conn_map.delete(socket.id);
        websocketServer.sendToClients('update_view');
    }
});
```

Rys. 4.1.17: Funkcja obsługująca przekroczenie czasu braku aktywności

4.2 Urządzenia końcowe

Program urządzeń końcowych został napisany w języku "C", przy pomocy narzędzi STM32CubeMX oraz IDE Eclipse. CubeMx jest programem graficznym pozwalającym w wygodny sposób generować kod inicjujący oraz platformą dostarczającą funkcje wyższego poziomu do obsługi peryferiów mikrokontrolera.

Dzięki temu, że urządzenia końcowe zostały wykonane w sposób modułowy, na bazie mikrokontrolerów STM32, większość napisanego kodu - proces inicjalizacji lub parowania jest wspólny dla wszystkich urządzeń. Na rysunku 4.2.1 przedstawiono schemat blokowy działania programu urządzeń końcowych, w dalszych podpunktach zawarto dokładne opisy poszczególnych bloków.



Rys. 4.2.1: Schemat działania programu urządzeń końcowych

a) Inicjalizacja mikrokontrolera oraz peryferiów

Po włączeniu zasilania, pierwszymi instrukcjami jakie zostają wykonane są funkcje inicjujące mikrokontroler oraz jego peryferia. Skonfigurowany zostaje zegar systemowy, wyprowadzenia zostają ustawione jako wyjścia dla wskaźników stanu lub jako wyjście z rezystorem polaryzującym w przypadku przycisku parowania. Następnie zainicjowany zostaje interfejs UART działający z prędkością 115200b/s w trybie 8-bitowej ramki z jednym bitem stopu, co umożliwia poprawną komunikację z modułem Wi-Fi ESP-01. Wraz z interfejsem UART, zainicjowany został kontroler DMA (ang. *Direct Memory Access*), umożliwiający dostęp do pamięci wewnętrznej bez udziału CPU. Dzięki temu zabiegowi dane odbierane przez UART są dostępne w zdefiniowanym buforze cyklicznym. Dodatkowo, dla każdego typu urządzenia końcowego zostały uruchomione dodatkowe funkcje - dla inteligentnej wtyczki wyjście z rezystorem polaryzującym do sterowania przekaźnikiem; dla czujnika temperatury wejście analogowe w celu pomiaru; dla inteligentnego oświetlenia RGB układ zegara w trybie trójkanałowego PWM.

b) Inicjalizacja łączności

Ustanowienie łączności między modułem Wi-Fi, a urządzeniem bazowym jest niezbędne w celu zapewnienia poprawnego funkcjonowania urządzeń końcowych, dlatego jest to krytyczna część programu. Do sterowania modułem Wi-Fi wykorzystano komendy AT [22] wysyłane poprzez interfejs UART. W tym celu, do modułów ESP-01 wgrano dostarczone przez producenta oprogramowanie oraz napisano odpowiednie funkcje dla programu działającego na mikrokontrolerze.

Proces inicjalizacji przedstawiono na rysunku 4.2.2. W pierwszym kroku ustawiany jest status modułu Wi-Fi, na podstawie którego podejmowane są dalsze kroki, aktualny stan wyświetlany jest na diodach. W systemie przewidziano pięć statusów:

No_Status - Ustawiany jest domyślnie po włączeniu mikrokontrolera, może też oznaczać niepoprawne funkcjonowanie/brak modułu Wi-Fi. Sygnalizowany jest poprzez załączenie czerwonej diody LED. Podczas inicjalizacji, status ten powoduje wyłączenie trybu *pass-through* oraz próbę uzyskania statusu od modułu ESP.

AP_Disconnected - Oznacza brak ustanowionego połączenia z siecią Wi-Fi, sygnalizowany jest poprzez załączenie diody czerwonej oraz zielonej. Status ten powoduje przejście do dalszej części programu, w której użytkownik może uruchomić procedurę parowania.

AP_Connected - Oznacza poprawne połączenie z siecią Wi-Fi, sygnalizowany jest poprzez wyłączenie obu diód LED. Podczas inicjalizacji status ten powoduje wywołanie funkcji odpowiedzialnej za ustanowienie połączenia z serwerem TCP.

Server_Connected - Oznacza ustanowienie połączenia z serwerem TCP - stacją bazową. Sygnalizowany jest za pomocą zielonej diody LED. Podczas inicjalizacji status ten powoduje wywołanie funkcji odpowiedzialnej za włączenie trybu *passthrough*.

Passthrough - Oznacza, że moduł Wi-Fi działa w trybie przesyłania wiadomości między interfejsem UART a serwerem TCP. Umożliwia komunikację mikrokontrolera z urządzeniem bazowym. Status ten sygnalizowany jest za pomocą mrukania zielonej diody LED. Podczas inicjalizacji powoduje wywołanie funkcji odpowiedzialnej za sparowanie urządzenia z systemem. Parowanie odbywa się poprzez wysłanie do urządzenia bazowego wiadomości składającej się z ID oraz sygnału *pairing* - liczby 9999. Np. "319999".

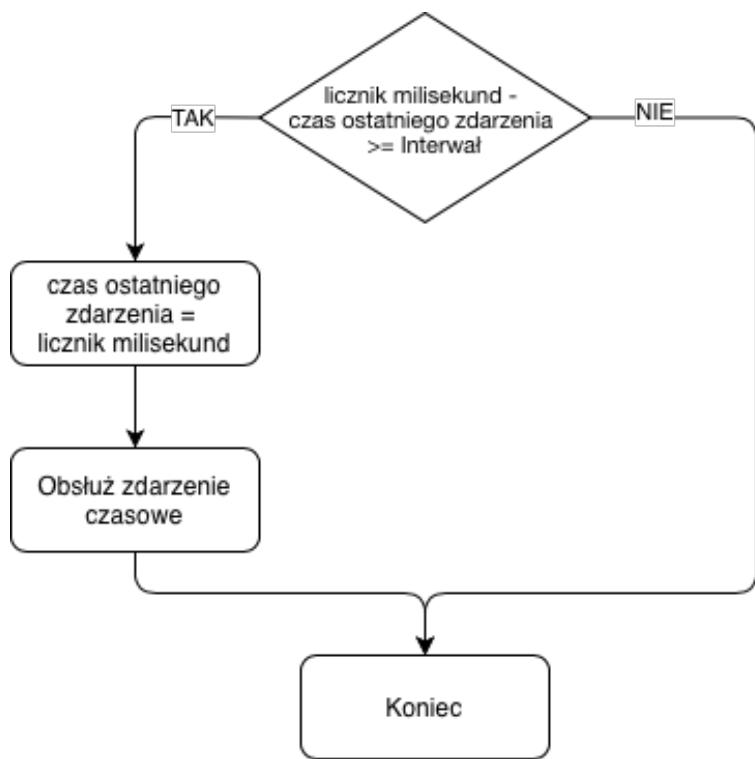
Podczas inicjalizacji po każdej wysłanej komendzie AT, mikrokontroler czeka na poprawną odpowiedź układu przez ścisłe określony czas - *timeout*. W przypadku braku odpowiedzi, moduł ESP-01 zostaje zrestartowany, a proces inicjalizacji zostaje uruchomiony od początku. Oznacza to, że mikrokontroler nie przejdzie do dalszej części programu przed ustanowieniem poprawnej komunikacji z modułem Wi-Fi.

```
void esp_initialize(){
    if (esp_status == No_Status){
        disable_passthrough();
        set_esp_status();
    }
    if (esp_status == AP_Connected)
        connect_to_server();
    if (esp_status == Server_Connected)
        enable_passthrough();
    if (esp_status == Passthrough)
        pair_device();
    if (esp_status == AP_Disconnected);
}
```

Rys. 4.2.2: Funkcja odpowiedzialna za inicjalizację modułu Wi-Fi

c) Inicjalizacja zdarzeń czasowych

Aby umożliwić periodyczne wykonywanie zadań przez mikrokontroler wykorzystano podprogram zdarzeń czasowych. Wyeliminowano w ten sposób potrzebę stosowania opóźnień blokujących w pętli głównej. Podprogram bazuje na strukturze, w której przechowywany jest czas ostatniego zdarzenia, interwał oraz wskaźnik do funkcji, która ma być okresowo wykonywana. Do odmierzania czasu posłużyono się systemowym licznikiem, który jest inkrementowany co 1ms, przez zegar *SysTick*. Algorytm sprawdzający czy obsłużyć dane zdarzenie czasowe przedstawiono na rysunku 4.2.3.



Rys. 4.2.3: Algorytm obsługi zdarzeń czasowych

d) Podprogram obsługi parowania

W celu połączenia urządzenia z domową siecią Wi-Fi, a tym samym sparowania go ze stacją bazową, wykorzystano wbudowaną w moduł ESP funkcję WPS (ang. *Wi-Fi Protected Setup*). WPS jest standardem pozwalającym dołączyć do sieci Wi-Fi bez potrzeby wpisywania nazwy sieci oraz hasła.

Funkcja obsługi WPS (Rys. 4.2.4) uruchamiana jest przez mikrokontroler po przytrzymaniu przycisku parowania przez conajmniej 5s. Następnie, do modułu ESP wysyłana jest seria komend - wyłączenie *passthrough*; włączenie trybu stacji; włączenie WPS. Po uruchomieniu funkcji WPS na urządzeniu końcowym należy ją również uruchomić na domowym routerze. Poprawne połączenie z siecią Wi-Fi sygnalizowane jest wiadomością “connecting” wyslaną przez moduł ESP, po czym uruchamiana jest inicjalizacja łączności. Parametry sieci Wi-Fi zapisywane są w pamięci modułu, dzięki czemu ponownie łączenie z siecią następuje automatycznie. W razie niepowodzenia, w ciągu 30s program zostaje zrestartowany.

```
void esp_wps(){
    disable_passthrough();
    send_command("AT+CWMODE=1");
    send_command("AT+WPS=1");
    wait_for_response("connecting", 30000);
    esp_status = AP_Connected;
    esp_initialize();
}
```

Rys. 4.2.4: Funkcja obsługi WPS

e) Podprogram obsługi łączności

W przypadku urządzeń wykonawczych, które jedynie odbierają wiadomości ze stacji bazowej podtrzymuje połączenie TCP wysyłając periodycznie wiadomość “OK”, dodatkowo dla wszystkich urządzeń kontroluje stan łączności. Monitoruje periodycznie bufor odbiorczy UART w poszukiwaniu wiadomości “ERROR” lub “Connected”, które moduł ESP wysyła w razie wystąpienia problemów z komunikacją. W przypadku błędu zostaje uruchomiona sekwencja inicjująca łączność.

f) Podprogram obsługi funkcji urządzenia

W pętli głównej mikrokontrolera periodycznie wykonywany jest podprogram realizujący funkcje specyficzne dla danego urządzenia końcowego. W przypadku urządzeń wykonawczych podprogram ten oparto na parserze. Analizuje on bufor odbiorczy UART w poszukiwaniu wiadomości, które zaczynają się znakiem "\$", a kończą znakiem CR. Parser zamienia ich treść na wartość liczbową a następnie kasuje je w buforze odbiorczym. Otrzymana wiadomość zależy od typu urządzenia:

Inteligentne oświetlenie RGB - otrzymana wartość składa się z czterech cyfr: SCBB, gdzie S - stan włączenia przyjmuje wartość 0 lub 1; C - zadany kolor 0-9; BB - zadana jasność 0-99. Funkcja sterująca (Rys. 4.2.5), manipuluje kanałami PWM mikrokontrolera w celu uzyskania pożądanych przez użytkownika parametrów oświetlenia.

Inteligentna wtyczka - otrzymana wartość składa się jedynie ze stanu włączenia na podstawie którego funkcja steruje wyjściem mikrokontrolera (Rys. 4.2.6).

```
void rgb_control(){
    uint32_t rgb_values = 0;
    if (esp_parse_value(&rgb_values)) {
        uint8_t brightness = rgb_values%100;
        Color color = (rgb_values/100)%10;

        _Bool rgb_state = rgb_values/1000;

        rgb_setBrightness(brightness);
        rgbSetColor(color);

        if (rgb_state)
            rgb_on();
        else
            rgb_off();
    }
}
```

Rys. 4.2.5: Funkcja sterująca oświetleniem

```

void relay_control(){
    uint32_t relay_state;
    if (esp_parse_value(&relay_state)) {
        if (relay_state == 1)
            WALL_GPIO_Port->BSRR = WALL_Pin;
        else
            WALL_GPIO_Port->BRR = WALL_Pin;
    }
}

```

Rys. 4.2.6: Funkcja sterująca wtyczką

Dla urządzeń pomiarowych - czujnika temperatury program periodycznie wysyła aktualny pomiar poprzez interfejs UART modułu Wi-Fi do urządzenia bazowego. Podczas inicjalizacji, przetwornik analogowo-cyfrowy został skonfigurowany do działania w trybie DMA wyzwalanym zegarem, dzięki temu aktualny pomiar znajduje się w utworzonej zmiennej. Przed wysłaniem wartość z przetwornika najpierw jest zamieniana na temperaturę w formacie stałoprzecinkowym. Ponieważ czujnik LM35 posiada liniową skalę +10-mV/°C, proces ten, sprowadza się do zamienienia wartości ADC na napięcie. Następnie temperatura formatowana jest na ciąg znaków. Funkcję formatującą pokazano na rysunku 4.2.7.

```

char* sensor_getTemp() {
    static char str[TEMP_LEN];
    uint32_t temp = (adcValue * ADCREF_MV) / ADCMAX;
    sprintf(str, "%02u.%1u", temp / 10, temp % 10);
    return str;
}

```

Rys. 4.2.7: Funkcja formatująca temperaturę

Rozdział 5

Testy oraz badania prototypu systemu

5.1 Badania funkcjonalne prototypu

W celu sprawdzenia poprawności działania projektu, zostały wykonane urządzenia prototypowe oraz, przy pomocy routera D-link GO-RT-AC750, utworzona została nowa sieć Wi-Fi. Do poprawnego działania systemu należało ustawić adres podsieci 192.168.0.0. Stacja bazowa posiada stały adres IP, dzięki czemu mogą się z nią połączyć urządzenia końcowe. Pierwszym etapem testów było wdrożenie oprogramowania stacji bazowej, ponieważ jest ono rozwijane poza serwerem roboczym. Do implementacji/wdrożenia wykorzystano system kontroli wersji - Git. W kolejnym kroku badano poprawność działania aplikacji z ekranem dotykowym stacji bazowej. Nie wszystkie elementy wejścia HTML działają tak samo z kursorem, co z dotykiem - obsługa włączania oświetlenia RGB przy pomocy wejścia suwaka (ang. *Slider-thumb*) nie działała poprawnie i zostało zastąpione. Następnie testowano funkcjonalność systemu symulując typowe warunki korzystania. Utworzono zostały dwa dodatkowe konta użytkowników: *home* zalogowany na stacji bazowej, oraz *user* zalogowany na smartfonie. Administrator za pomocą urządzenia PC zarządzał systemem. Do systemu podłączono zaprojektowane urządzenia końcowe. W tym celu uruchomiono na nich proces parowania poprzez przytrzymanie przycisku WPS oraz włączenie funkcji WPS PBC (ang. *Push-Button-Connect*) w ustawieniach routera D-link. Wszystkie urządzenia zostały automatycznie dodane do widoku głównego administratora w kolejności podłączania. Przy pomocy komputera sprawdzano poprawność odpowiedzi urządzeń wykonawczych na wprowadzane zmiany - testowano synchronizację stanu włączania inteligentnej wtyczki i stanu jasności/koloru oświetlenia. Po sprawdzeniu poprawności działania komunikacji z urządzeniami końcowymi testowano funkcje administracyjne oraz synchronizacje między klientami. Z poziomu widoku zarządzania dodano użytkownikom *home* oraz *user* dostęp do urządzeń wykonawczych. Następnie za pomocą stacji bazowej prowadzono interakcję z inteligentnym oświetleniem, obserwując jednocześnie synchronizację informacji u pozostałych klientów. Sprawdzano również poprawność działania systemu w sytuacji jednokrotnego wprowadzania zmian, w obu przypadkach systemy zachowywały poprawną synchronizację między klientami jak i urządzeniami końcowymi. Pozostałe funkcje administracyjne jak usuwanie użytkowników czy zmiana nazw urządzeń również działały bez zarzutu. Dodatkowo sprawdzano odpowiedź systemu w sytuacjach brze-

gowych jak np. wyłączenie stacji bazowej, awaria urządzenia końcowego lub zanik sieci Wi-Fi. We wszystkich sprawdzanych scenariuszach system działał poprawnie. Dzięki mechanizmom kontrolującym łączność jak *time-out* serwera TCP czy podprogram obsługi łączności w urządzeniach końcowych, wszelkie anomalie są wykrywane oraz odpowiednio sygnalizowane.

Istotnym etapem projektowania urządzeń jest zapewnienie ich jak najdłuższej pracy. Podrozdział ten poświęcono rozważaniom na temat wpływu starzenia na działanie urządzeń. Najsłabszym elementem zaprojektowanego systemu jest karta SD, na której uruchomiono system operacyjny stacji bazowej, ponieważ w przeciwieństwie do dysków SSD, karty SD nie są wyposażone w mechanizmy równoważenia zużycia (ang. *Wear Leveling*), dlatego ich żywotność w tych warunkach jest szacowana na 1-3 lata. Drugim elementem, który może ulec awarii przy długookaznym użytkowaniu jest przekaźnik w intelligentnej wtyczce. Jego minimalna trwałość jest przewidywana na 100 000 przełączeń, co przy intensywnym użytkowaniu np. 20/dzień wynosi rok i 4 miesiące. Pozostałe elementy urządzeń nie są narażone na nadmierne zużycie. W projekcie nie wykorzystano baterii jako źródła zasilania urządzeń końcowych, które po pewnym czasie należało by wymienić. Dodatkowym atutem projektu jest wykorzystanie najnowszych standardów takich jak USB, co pomoże uchronić urządzenia przed przedwczesnym zestarzeniem.

5.2 Badania zasięgu/opóźnień

W celu sprawdzenia wydajności działania urządzeń w typowym środowisku, przeprowadzono test opóźnień. Polegał on na badaniu wpływu umiejscowienia urządzeń końcowych w mieszkaniu, na siłę odbieranego sygnału od punktu dostępowego, a tym samym opóźnienia między stacją bazową. Do testów wykorzystano router D-link GO-RT-AC750, do którego podłączono jedynie urządzenia poddawane testom. Po zeskanowaniu środowiska elektromagnetycznego, router Wi-Fi został skonfigurowany do działania na kanele nr 1 sieci 2,4Ghz, o szerokości pasma 20Mhz. Ustawienia te pozwoliły zmniejszyć interferencję z sąsiednimi sieciami. Eksperyment przeprowadzono na piętrze domu jednorodzinnego (Rys. 5.2.1), w którym w centralnym punkcie umieszczono router (AP) oraz urządzenie bazowe (UB). Następnie w pięciu różnych miejscach sporządzono pomiary RTT (ang. Round-Trip Time) oraz RSSI (ang. Received Signal Strength Indication). Do badań wykorzystano narzędzie Ping oraz wbudowany w układ ESP8266 skaner sieci W-Fi. W każdym punkcie przeprowadzono 10 pomiarów RTT. W tabeli 5.2.1 umieszczono czas minimalny, maksymalny oraz średnią wyników. Wszystkie wysłane pakiety testowe zostały dostarczone. Pomiary RSSI (Tab. 5.2.2) wykonano jednokrotnie. Na podstawie wyników badania można stwierdzić, że system w typowym środowisku zachowuje odpowiednie parametry - średnie opóźnienie poniżej 100ms, nawet dla urządzeń umieszczonych w większej odległości od stacji bazowej oraz routera. Na jakość odbieranego sygnału zgodnie z oczekiwaniemi największy wpływ miał dystans między urządzeniami oraz ilość ścian na drodze sygnału Wi-Fi. Należy pamiętać, że zmierzone wartości mogą się różnić w zależności od używanego sprzętu oraz od środowiska.

Tab. 5.2.1 Wyniki pomiarów RTT

Pomiar	P1	P2	P3	P4	P5
Minimum [ms]	4,0	4,0	4,0	6,0	2,0
Maksimum [ms]	73,0	120,0	95,0	122,0	105,0
Średnia [ms]	15,0	65,6	32,0	53,9	53,2

Tab. 5.2.2 Wyniki pomiarów RSSI

Pomiar	P1	P2	P3	P4	P5
RSSI [dBm]	-57	-79	-85	-74	-75



Rys. 5.2.1: Plan piętra z rozmieszczeniem urządzeń

5.3 Bilans mocy

Pomimo, że urządzenia systemu zasilane są z sieci elektrycznej, zostały zaprojektowane z myślą o niskim zużyciu energii. Pomogło przy tym wykorzystanie zasilaczy impulsowych oraz układów scalonych małej mocy, jak mikrokontrolery STM32 z serii L0. W bilansie mocy dla wszystkich urządzeń brano pod uwagę najgorszy przypadek: załączone diody LED; zwarty przekaźnik; włączony wyświetlacz; moduł Wi-Fi działający w trybie nadawania. Pobór prądu mikrokontrolera został obliczony przy pomocy narzędzia STM32CubeMX, dane dla pozostałych elementów pochodzą z not katalogowych, wyniki umieszczone w tabelach 5.4.1 - 5.4.4.

Tab. 5.4.1 Bilans mocy urządzenia bazowego [23]

	Napięcie zasilania	Pobór prądu	Pobór mocy
Raspberry Pi 4B	5V	1.25A	6.25W
Moduł Wyświetlacza	5V	1A	5W
			Suma:
			11.25W

Tab. 5.4.2 Bilans mocy inteligentnej wtyczki [10]/[24]

	Napięcie zasilania	Pobór prądu	Pobór mocy
STM32L031	3,3V	1,5mA	4,95mW
ESP01	3,3V	120mA	366mW
LED czerwona	3,3V	1,55mA	5,12mW
LED zielona	3,3V	1,3mA	4,29mW
JW1FSN	5V	106mA	530mW
			Suma:
			911mW

Tab. 5.4.3 Bilans mocy inteligentnego oświetlenia [10][25]

	Napięcie zasilania	Pobór prądu	Pobór mocy
STM32L031	3,3V	1,5mA	4,95mW
ESP01	3,3V	120mA	366mW
LED czerwona	3,3V	1,55mA	5,12mW
LED zielona	3,3V	1,3mA	4,29mW
Taśma LED RGB 1m	5V	1,5A	7,5W
			Suma:
			7,88W

Tab. 5.4.4 Bilans mocy inteligentnego czujnika [21]

	Napięcie zasilania	Pobór prądu	Pobór mocy
STM32L031	3,3V	1,5mA	4,95mW
ESP01	3,3V	120mA	366mW
LED czerwona	3,3V	1,55mA	5,12mW
LED zielona	3,3V	1,3mA	4,29mW
LM35DZ	5V	60µA	300µW
			Suma:
			381mW

Rozdział 6

Podsumowanie

Celem pracy dyplomowej było zaproponowanie rozproszonego systemu sterowania w inteligentnym budynku. Prace projektowe poprzedziła dokładna analiza rynku, dzięki której ustalono założenia projektowe oraz wybrano typy urządzeń jakie zostaną wykonane: inteligentne oświetlenie RGB, inteligentna wtyczka, inteligentny czujnik temperatury oraz stacja bazowa sterowana panelem dotykowym, pozwalającą na interakcję z systemem. Według aktualnych trendów są to typy urządzeń, które użytkownicy najczęściej wymieniają na inteligentne odpowiedniki. W kolejnym etapie pracy dokonano analizy rozwiązań komunikacyjnych. Autor pracy porównał ze sobą cztery wiodące technologie: Zigbee, KNX, Bluetooth oraz Wi-Fi. Główne kryteria jakimi się kierowano to: prędkość transmisji, opóźnienia, dostępność oraz możliwości rozwoju. Żadna z opisanych technologii nie jest pozbawiona wad lecz zdecydowano się na Wi-Fi, ponieważ oferuje najlepszy kompromis pomiędzy porównywany kryteriami. W przyszłości należałoby rozważyć wykorzystanie więcej niż jednej technologii w systemie do interakcji z inteligentnymi czujnikami, co ułatwiłoby zasilanie baterijne, np. Bluetooth. Wybór Wi-Fi jako wiodącej technologii komunikacyjnej w systemie dał projektantowi możliwość wykorzystania standardowych sieciowych protokołów transmisji - TCP do komunikacji między urządzeniami końcowymi, HTTP i WebSocket do komunikacji między serwerem a widokami.

Największą zaletą zaprojektowanego systemu jest dobrze przemyślany interfejs użytkownika. Umożliwia on wszystkim klientom szybką interakcję z dostępnymi urządzeniami, a wszelkie zaawansowane funkcjonalności udostępniane są jedynie administratorowi systemu. Kolejnym silnym elementem projektu jest jego rozszerzalność - nowe urządzenia dodawane są do systemu automatycznie po poprawnym połączeniu z domową siecią Wi-Fi. Prostota oraz bezproblemowa obsługa urządzeń inteligentnych są cechami, dzięki którym użytkownicy częściej wybierają sprzęt danej marki. Dlatego autor przywiązywał do tego szczególną uwagę w trakcie procesu projektowania oraz testowania urządzeń.

Projekt nie jest pozbawiony wad. Wydajność systemu - zasięg/opóźnienia zależą od istniejącej infrastruktury - domowego routera bądź punktu dostępowego W-Fi. W przypadku większych obiektów może to oznaczać konieczność wymiany sprzętu na nowszy. Słabym punktem systemu jest również zasilanie sieciowe czujnika temperatury - zazwyczaj urządzenia te są wyposażone w baterię.

Mimo wymienionych wad zrealizowany system spełnia wszystkie postawione założenia projektowe. Dzięki modułowemu wykonaniu urządzeń koń-

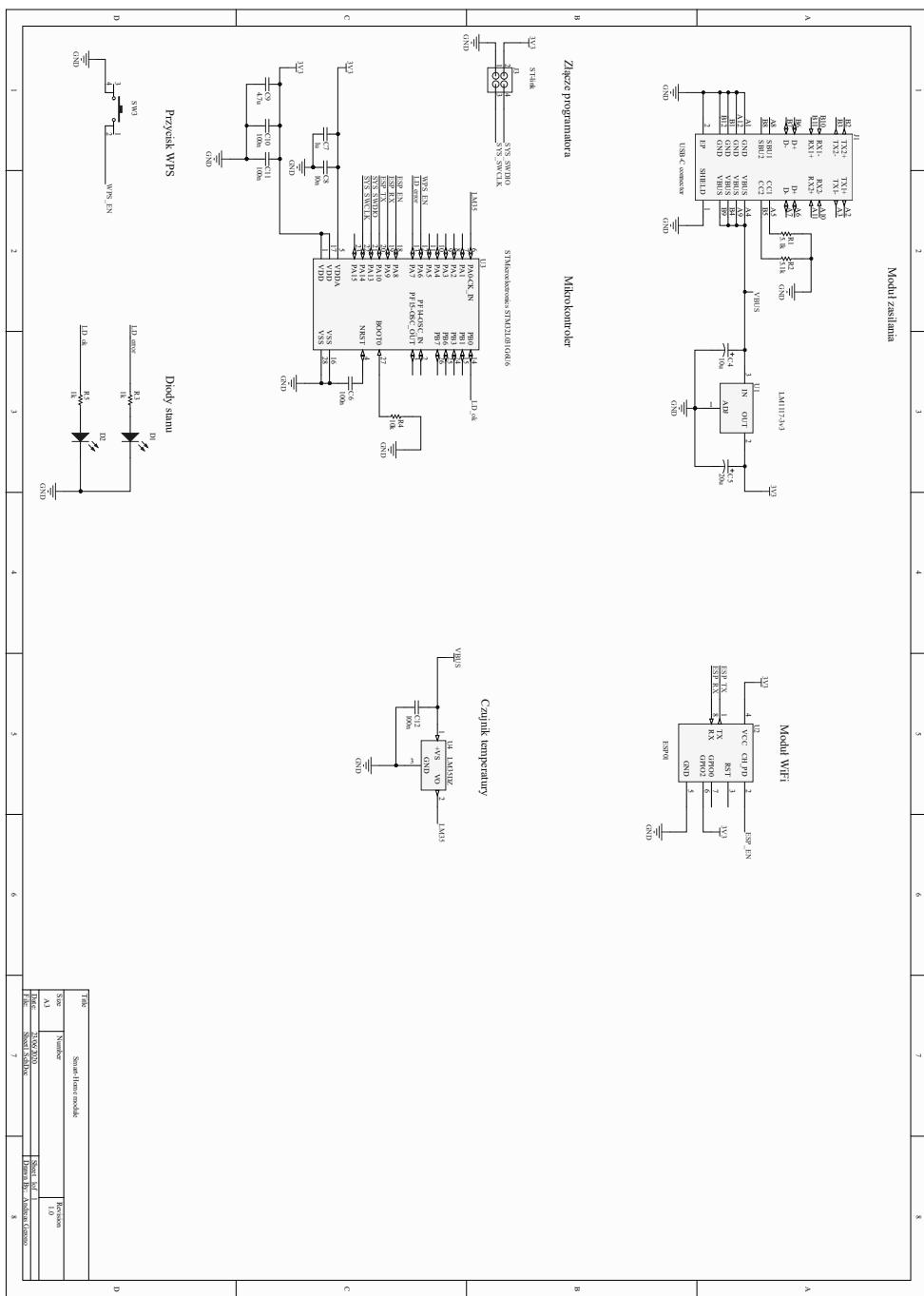
cowych oraz zapasowi wydajności użytych układów można go dalej rozwijać, na przykład poprzez dodatkowy widok prezentujący ostatnio używane/ulubione urządzenia. Przemyślana budowa oprogramowania umożliwia również dodanie do systemu nowych typów urządzeń, takich jak innego rodzaju czujniki. Dodając nowe możliwości do systemu należałoby uważać by nie naruszyć filozofii, w myśl której został on stworzony, czyli prostocie obsługi.

Literatura

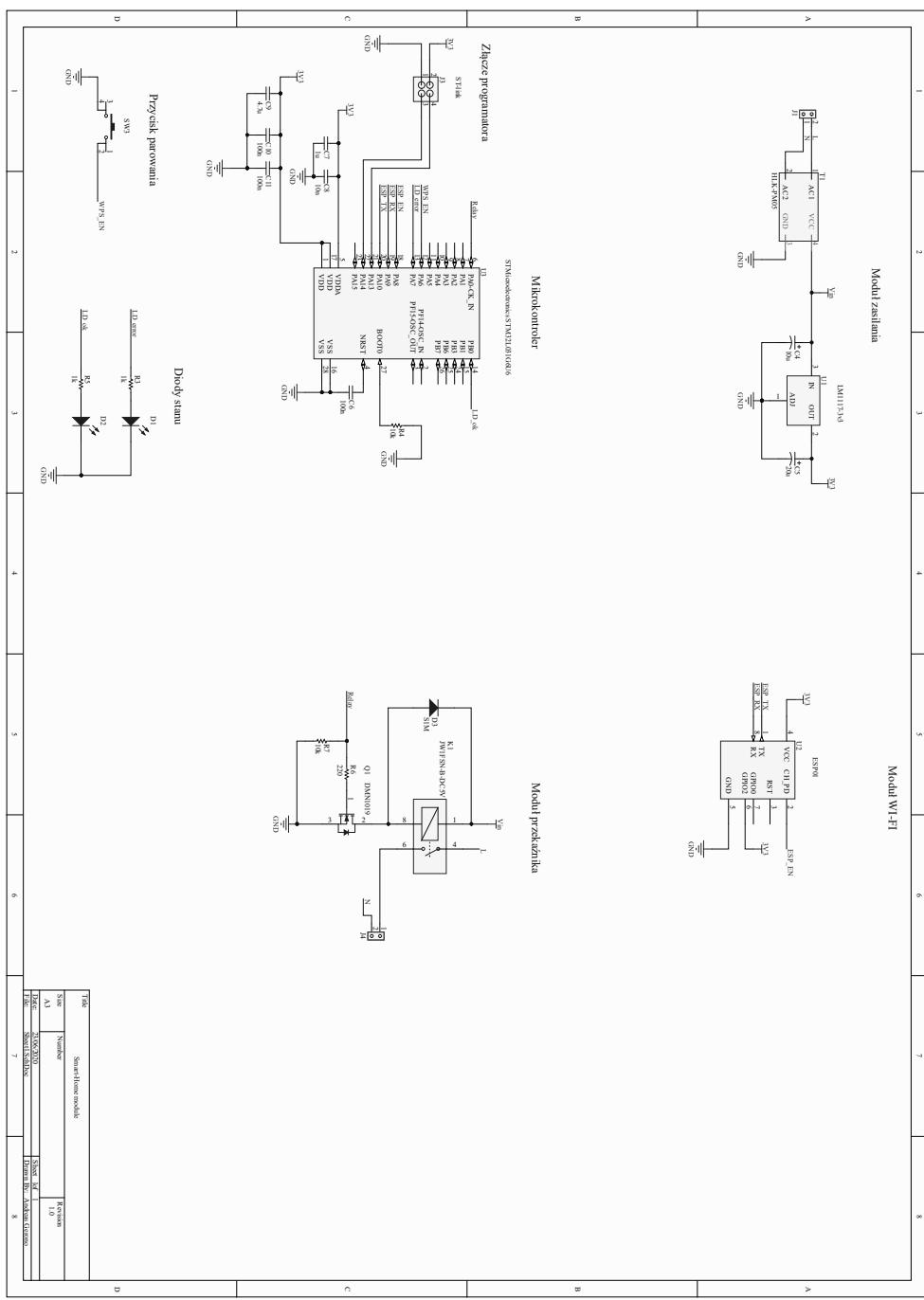
- [1] Deloitte, The Deloitte Consumer Review, 2016.
- [2] F-Secure, Zigbee Network an overview.
- [3] KNX Association, The KNX standard – the basics.
- [4] Lora Alliance, A technical overview of LoRa and LoRaWAN.
- [5] Bluetooth SIG, Bluetooth Mesh Networking / An Introduction for Developers.
- [6] R. Rondo'n, A. Mahmood, S. Grimaldi, M. Gidlund, Understanding the Performance of Bluetooth Mesh, 2019.
- [7] P. N. Borza, M. Machedon-Pisu, F. Hamza-Lup, Design of Wireless Sensors for IoT with Energy Storage and Communication Channel Heterogeneity, 2019.
- [8] A. Seferagic , J. Famaey , E. De Poorter, J. Hoebeke, Survey on Wireless Technology Trade-Offs for the Industrial Internet of Things, 2020.
- [9] A. Baviskar, J. Baviskar, S. Wagh, A. Mulla, P. Dave, Comparative Study of Communication Technologies for Power Optimized Automation Systems, 2015.
- [10] Espressif Systems, ESP8266EX Datasheet.
- [11] Texas Instruments, CC2541 Datasheet.
- [12] Texas Instruments, CC2530 Datasheet.
- [13] Semtech, SX1276 Datasheet.
- [14] Q. Duy La, D. Nguyen-Nam, M. V. Ngo, Hieu T. Hoang, T. Q.S. Quek, Dense Deployment of BLE-based Body Area Networks, 2018.
- [15] M. B. Shoemake, Wi-Fi and Bluetooth Co-existence Issues and Solutions for the 2.4GHz ISM Band, 2001.
- [16] Silicon Labs, AN1142: Mesh Network Performance Comparison.
- [17] S. Tozlu, M. Senel, W. Mao, A. Keshavarzian, Wi-Fi Enabled Sensors for Internet of Things: A Practical Approach, 2012.
- [18] STMicroelectronics, AN4467: Getting started with STM32L0xx hardware development.
- [19] Texas Instruments, LM1117 Datasheet.
- [20] Microchip, AN1953: Introduction to USB Type-C.
- [21] Texas Instruments, LM35DZ Datasheet.
- [22] Espressif Systems, ESP8266 AT Instruction Set
- [23] Raspberry Pi, <https://www.raspberrypi.org/documentation/faqs/#pi-power/>
- [24] Panasonic, JW1ASN Datasheet.
- [25] 5V RGB LED Strip,
<https://www.pcboard.ca/5v-RGB-LED-Strip-30-LEDs-per-Meter>

Dodatki

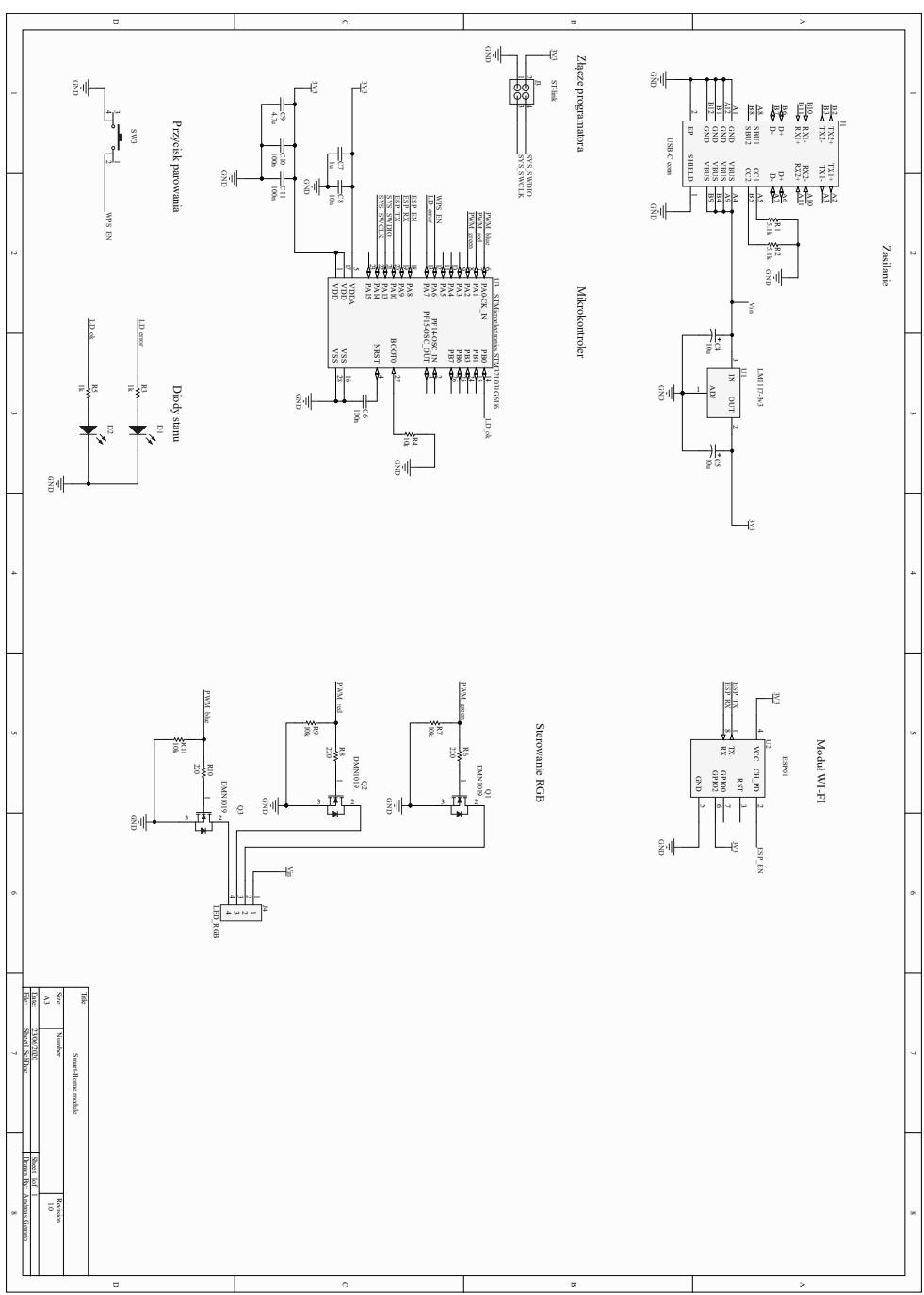
Schematy elektryczne



Rys. 6.1: Schemat elektryczny intelligentnego czujnika temperatury

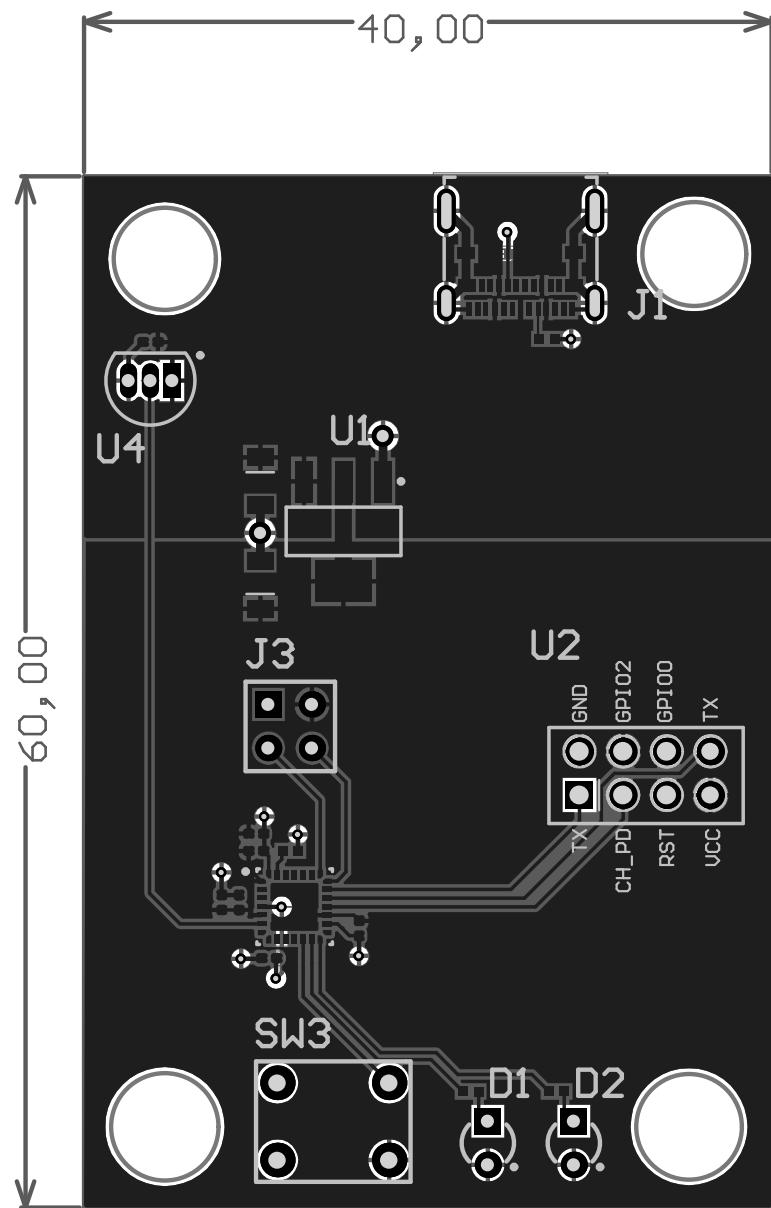


Rys. 6.2: Schemat elektryczny inteligentnej wtyczki

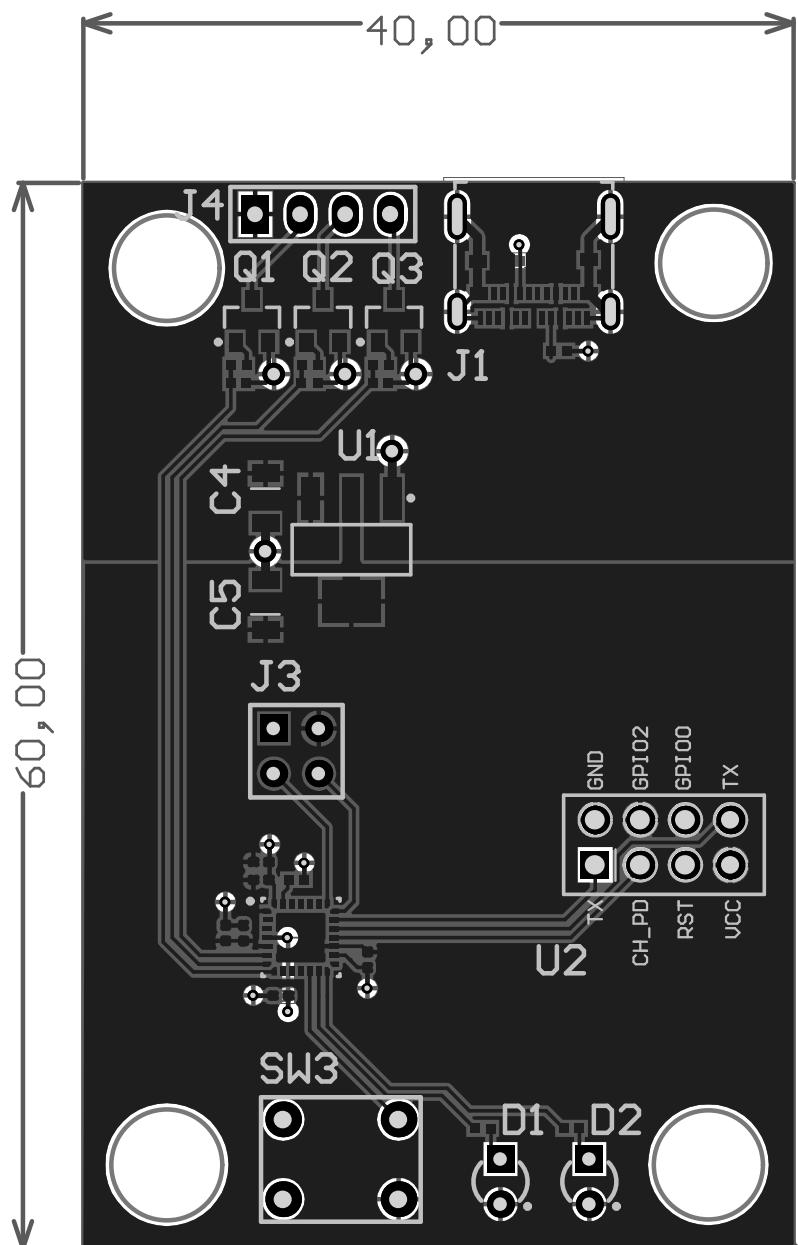


Rys. 6.3: Schemat elektryczny inteligentnego oświetlenia

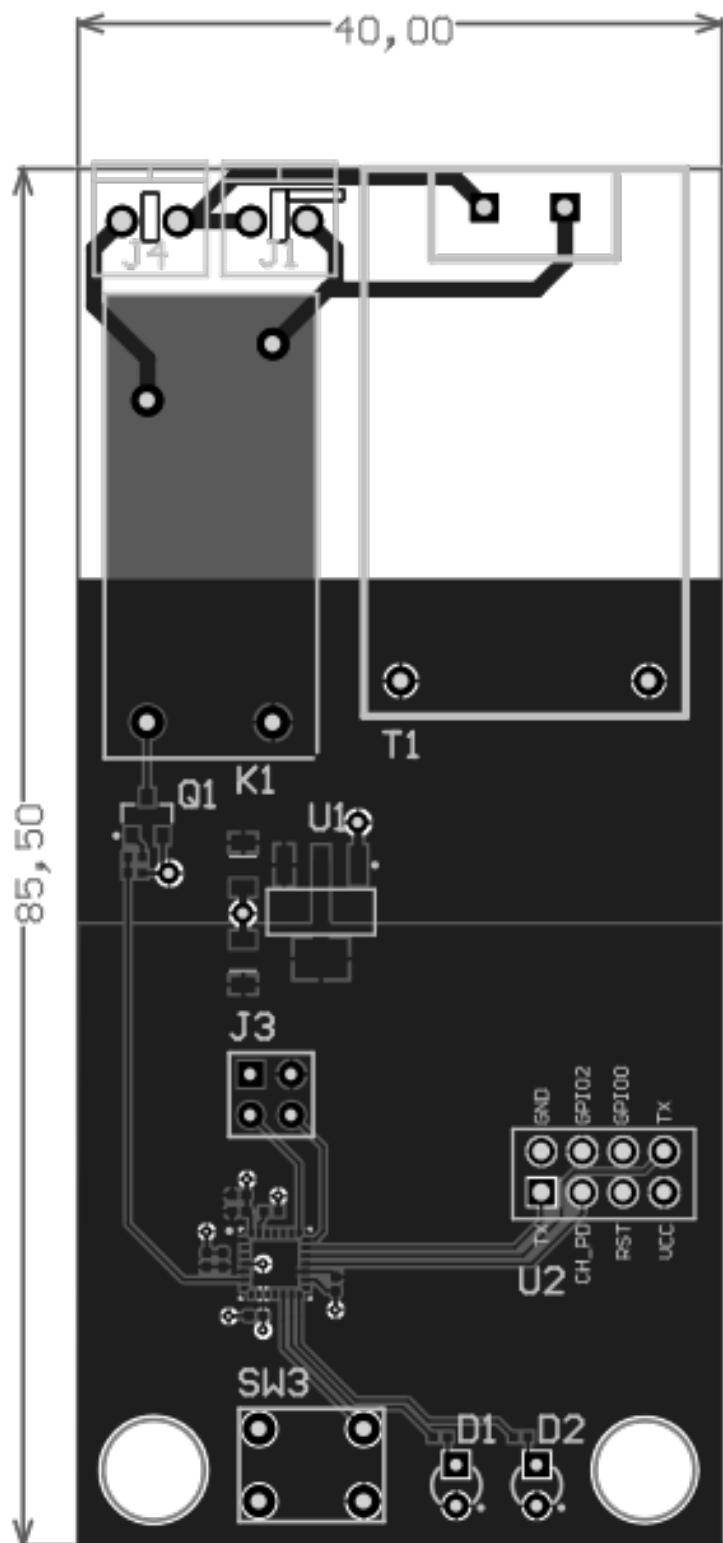
Projekty PCB



Rys. 6.4: Projekt PCB inteligentnego czujnika temperatury



Rys. 6.6: Projekt PCB inteligentnego oświetlenia



Rys. 6.5: Projekt PCB inteligentnej wtyczki