AKADEMIA GÓRNICZO-HUTNICZA

IM. STANISŁAWA STASZICA W KRAKOWIE

Wydział Inżynierii Metali i Informatyki Przemysłowej

PRACA DYPLOMOWA INŻYNIERSKA

pt.

„Konstrukcja i zaprogramowanie opaski na rękę z przyciskiem do wzywania pomocy, wykorzystującej moduł GPS oraz sieć komórkową”

Imię i nazwisko dyplomanta:	**Kacper Pawlikowski**
Kierunek studiów:	**Informatyka Stosowana**
Nr albumu:	**286216**
Promotor:	dr inż. Piotr Kustra
Recenzent:	dr inż. Adam Mrozek

Podpis dyplomanta: Podpis promotora:

Kraków 2019

„Uprzedzony o odpowiedzialności karnej na podstawie art. 115 ust. 1 i 2 ustawy z dnia 4 lutego 1994 r. o prawie autorskim i prawach pokrewnych (t.j. Dz.U. z 2006 r. Nr 90, poz. 631 z późn. zm.): „Kto przywłaszcza sobie autorstwo albo wprowadza w błąd co do autorstwa całości lub części cudzego utworu albo artystycznego wykonania, podlega grzywnie, karze ograniczenia wolności albo pozbawienia wolności do lat 3. Tej samej karze podlega, kto rozpowszechnia bez podania nazwiska lub pseudonimu twórcy cudzy utwór w wersji oryginalnej albo w postaci opracowania, artystyczne wykonanie albo publicznie zniekształca taki utwór, artystyczne wykonanie, fonogram, wideogram lub nadanie.”, a także uprzedzony o odpowiedzialności dyscyplinarnej na podstawie art. 211 ust. 1 ustawy z dnia 27 lipca 2005 r. Prawo o szkolnictwie wyższym (t.j. Dz. U. z 2012 r. poz. 572, z późn. zm.) „Za naruszenie przepisów obowiązujących w uczelni oraz za czyny uchybiające godności studenta student ponosi odpowiedzialność dyscyplinarną przed komisją dyscyplinarną albo przed sądem koleżeńskim samorządu studenckiego, zwanym dalej "sądem koleżeńskim"”, oświadczam, że niniejszą pracę dyplomową wykonałem(-am) osobiście i samodzielnie i że nie korzystałem(am) ze źródeł innych niż wymienione w pracy.”

Kraków, dnia …… Podpis dyplomanta

Spis treści

Wstęp

Rozwój technologii cyfrowej sprawia, że urządzenia elektroniczne towarzyszą ludziom na każdym etapie życia. W ostatnich latach kładziony jest duży nacisk na to, aby nowoczesna technologia była dostępna dla osób starszych i wspierała ich w codziennym życiu. Nierozłącznie postęp techniczny wpływa na jakość oferowanej opieki zdrowotnej. Ponieważ w sytuacji nagłego pogorszenia zdrowia kluczowe jest udzielenie jak najszybszej pomocy, dużym zainteresowaniem cieszą się wszelkie systemy, które usprawniają skontaktowanie się z odpowiednimi służbami. Takie rozwiązania są szczególnie ważne dla osób starszych, które niejednokrotnie nie są w stanie wezwać pomocy samodzielnie. Z tego powodu na rynku pojawiają się nieustannie nowe urządzenia, które umożliwiają szybkie powiadomienie krewnych lub opiekunów o zaistniałych problemach. Oprócz jak najszybszego zaalarmowania odpowiednich osób ważne też jest łatwe zlokalizowanie wzywającego pomoc, dlatego stosuje się różne systemy lokalizujące oparte o technologię GPS. Takie urządzenia mogą też pomagać rodzinom osób cierpiących na Alzhaimera w zlokalizowaniu zagubionego krewnego.

Istotną kwestią jest wygoda osoby starszej, tak aby ta nie czuła się ograniczona przez system mający na celu ułatwienie jej życia. Z tego powodu opaska na rękę z przyciskiem do wzywania pomocy jest świetnym rozwiązaniem, które znacząco zwiększa bezpieczeństwo nie wpływając negatywnie na komfort użytkownika.

1. Teza i cel pracy

1.1. Teza

Współczesne rozwiązania z dziedziny elektroniki i informatyki mogą zostać wykorzystane do wsparcia opieki nad osobami starszymi. Zastosowanie technologii GPS oraz połączenia z siecią komórkową umożliwia stworzenie lokalizatora usprawniającego wzywanie pomocy przez seniorów.

1.2. Cel pracy

Celem pracy inżynierskiej jest budowa i zaprogramowanie opaski na rękę z przyciskiem do wzywania pomocy, która przesyła lokalizację osoby poszkodowanej odczytaną przy pomocy modułu GPS, przy użyciu sieci komórkowej.

2. Przegląd istniejących rozwiązań

W ostatnich latach na rynku pojawiło się wiele nowych rozwiązań w dziedzinie urządzeń do wzywania pomocy dla osób starszych. W tej chwili producenci prześcigają się w usprawnianiu swoich urządzeń i dodawaniu do nich nowych funkcji.

Jednym z rozwiązań dostępnych na rynku jest seria urządzeń wypuszczona pod marką King Pigeon[1]. Bardziej zaawansowane modele potrafią wykryć upadek właściciela i w takiej sytuacji automatycznie wezwać pomoc. Urządzenia z tej serii nie mają możliwości bezpośredniego połączenia się z siecią komórkową tylko komunikują się bezprzewodowo ze specjalnym panelem kontrolnym, który to wysyła powiadomienia do opiekunów/ratowników. Rozwiązania z tej serii nie są wyposażone w lokalizatory GPS i wymagają dwóch urządzeń, dlatego są one przeznaczone dla osób mniej aktywnych, które poruszają się tylko w obszarze miejsca zamieszkania.

Innym podejściem do urządzenia wzywającego pomoc jest zrealizowana przez firmę Comarch Bransoletka Życia[2]. Opaska ta posiada moduł GPS i jest w pełni autonomiczna – łączy się z Centrum Zdalnej Opieki Medycznej bez udziału pośredniczącego panelu kontrolnego. Ratownik znajdujący się w centrali ma możliwość nawiązania połączenia z użytkownikiem opaski oraz posiada wgląd w odczyty pulsometru wbudowanego w urządzenie.

Na rynku można znaleźć jeszcze wiele innych urządzeń opartych na podobnych założeniach. Istnieją również aplikacje, które umożliwiają lokalizowanie członków rodziny ale ich wadą jest fakt, że osoba starsza musi posiadać smartfona/smartwatch-a, na którym będzie możliwe zainstalowanie oprogramowania.

3. Funkcjonowanie systemu wzywania pomocy

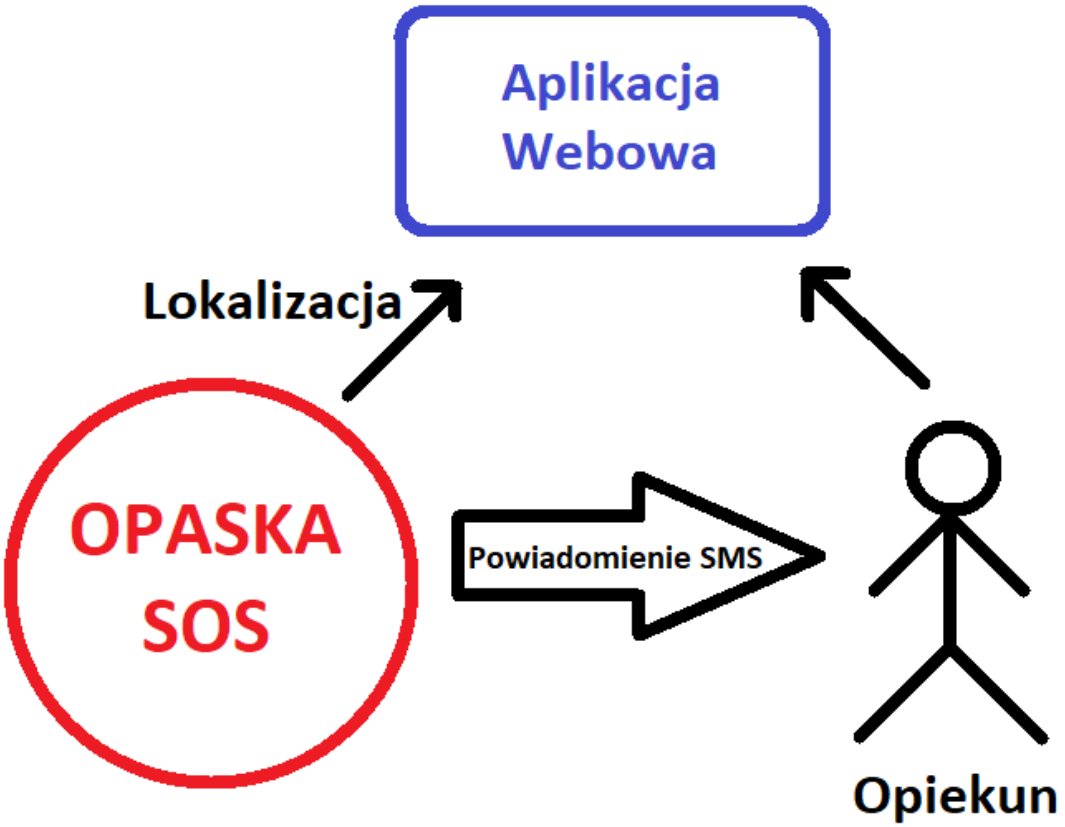
Opaska z przyciskiem do wzywania pomocy jest częścią większego systemu, który obejmuje aplikację webową umożliwiającą łatwe zlokalizowanie osoby starszej (rys. 3.1). Aplikacja ta została wykonana w ramach innej pracy inżynierskiej.

Opaska po naciśnięciu przycisku do wzywania pomocy wykonuje dwie podstawowe czynności:

- Wysyła powiadomienia SMS do wcześniej wskazanych opiekunów osoby starszej,

- Przesyła informacje o lokalizacji do aplikacji webowej.

Opiekun, po otrzymaniu powiadomienia na telefon, ma możliwość zalogowania się do aplikacji webowej i odczytania aktualnej lokalizacji swojego podopiecznego.



Rysunek 3.1 Działanie systemu wzywania pomocy

4. Dobór komponentów elektronicznych

Prototyp został oparty na układzie SIM808, ponieważ posiada on wszystkie niezbędne funkcje umożliwiające nawiązanie połączenia internetowego oraz określenie położenia w oparciu o GPS. Oprócz tego urządzenie to posiada wiele dodatkowych możliwości, które mogą zostać wykorzystane przy dalszym rozwoju projektu. Pozostałe komponenty do budowy opaski zostały dobrane pod kątem jak najlepszej współpracy z SIM808.

4.1. ATmega 328P-U DIP

Mikrokontroler wyprodukowany przez firmę Atmel. Został on wykorzystany w popularnej płycie Arduino Uno, która była używana przy wstępnym prototypowaniu

opaski. Układ scalony może być programowany przy użyciu płyty Arduino UNO bez potrzeby korzystana z dodatkowego programatora.

Mikrokontroler Posiada 32 kB pamięci flash oraz 2 kB pamięci RAM. Maksymalne taktowanie wynosi 20 MHz przy użyciu zewnętrznego źródła taktowania natomiast wewnętrzny oscylator kwarcowy zapewnia do 8 MHz. Do prawidłowego działania wymagane jest napięcie z zakresu 1.8 -5.5 V.

4.2. BK-SIM808 v2

Płytka PCB z układem SIM808 pozwalająca korzystać z jego podstawowych funkcjonalności.

4.2.1. Moduł SIM808

Układ umożliwiający nawiązywanie połączeń w sieci GSM i transmisję danych techniką GPRS, odczyt pozycji GPS czy też nawiązywanie połączeń Bluetooth. Dzięki swoim niewielkim rozmiarom (24x24x2.6 mm) idealnie nadaje się do budowy małych urządzeń mobilnych. Komunikacja z modułem odbywa się poprzez interfejs UART. SIM808 dodatkowo wspiera ładowanie baterii Li-ion.

4.2.2. Budowa płytki BK-SIM808 v2

Płytka Wyposażona jest w gniazdo na kartę Micro SIM, wyprowadzenia anten GSM, GPS i Bluetooth oraz 6 pin-ów umożliwiających zasilanie oraz komunikację z Modułem SIM808. BK-SIM808 nie pozwala niestety na wykorzystanie wsparcia modułu SIM808 dla ładowania baterii Li-ion. Zgodnie z dokumentacją[3] moduł powinien być zasilany prądem o napięciu z zakresu 5-10 V (z zaleceniem wykorzystania 5 V) ale w rzeczywistości urządzenie działa poprawnie dopiero zasilane napięciem powyżej 11 V. Za sprawą tej rozbieżności z dokumentacją, przy budowie prototypu trzeba było zastosować przetwornicę napięcia aby uzyskać wymagane parametry prądu.

4.3. Przetwornica step-up POLOLU-2117

Moduł, którego zadaniem jest podwyższenie napięcia wejściowego z baterii do wymaganego przez płytkę BK-SIM808. Napięcie wejściowe należy do zakresu od 2.5 V do 12 V a otrzymane napięcie wyjściowe wynosi 12 V.

4.4. Moduł tp-4056 do ładowania baterii typu Li-ion

Mała ładowarka do baterii typu Li-ion z interfejsem micro USB. Moduł pozwala na ładowanie baterii bez potrzeby wstrzymania pracy urządzenia.

5. Protokół do transmisji danych

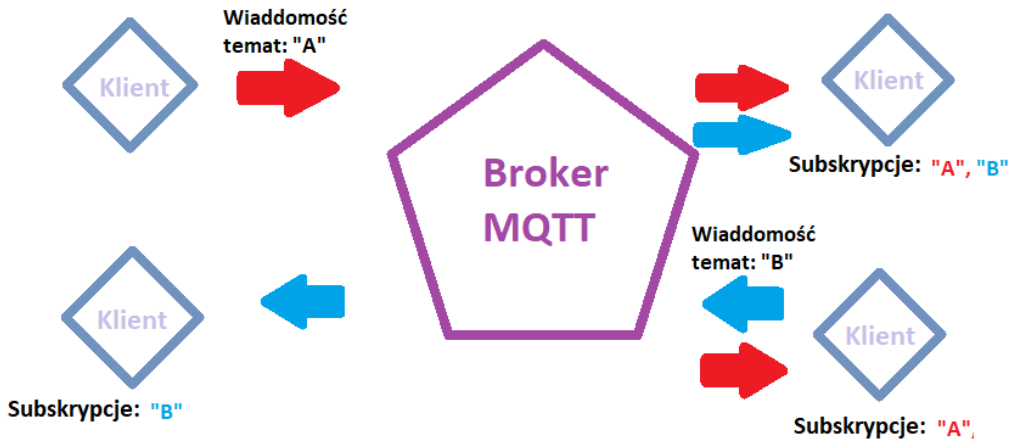
5.1. Porównanie popularnych protokołów internetowych

Wybór protokołu MQTT do realizacji tego projektu został dokonany na podstawie porównania kilku popularnych protokołów internetowych. Ze względu na specyfikę projektu głównymi kryteriami były prostota użycia oraz szybkość działania, która przekłada się na większą energooszczędność urządzenia.

5.1.1. MQTT

MQTT[4] to lekki protokół transportowy oparty na architekturze klient-serwer. Umożliwia on nawiązywanie połączeń zapewniających bezstratne przesyłanie danych z gwarancją zachowania odpowiedniej kolejności przesyłanych informacji. Komunikacja może odbywać się w obu kierunkach.

Schemat działania MQTT oparty jest na subskrypcjach, które umożliwiają dystrybucję wiadomości na zasadzie „jeden do wielu”. Każdy klient może mieć wiele subskrypcji. Aby wysłać wiadomość, klient musi podać nazwę tematu wiadomości a ta zostanie przesłana do wszystkich klientów, którzy subskrybują dany temat. Dystrybucją wiadomości do odpowiednich klientów zajmuje się specjalny serwer, który w MQTT nazywany jest brokerem. Prosty przykład przesyłania danych między klientami przedstawiono na rysunku 5.1.

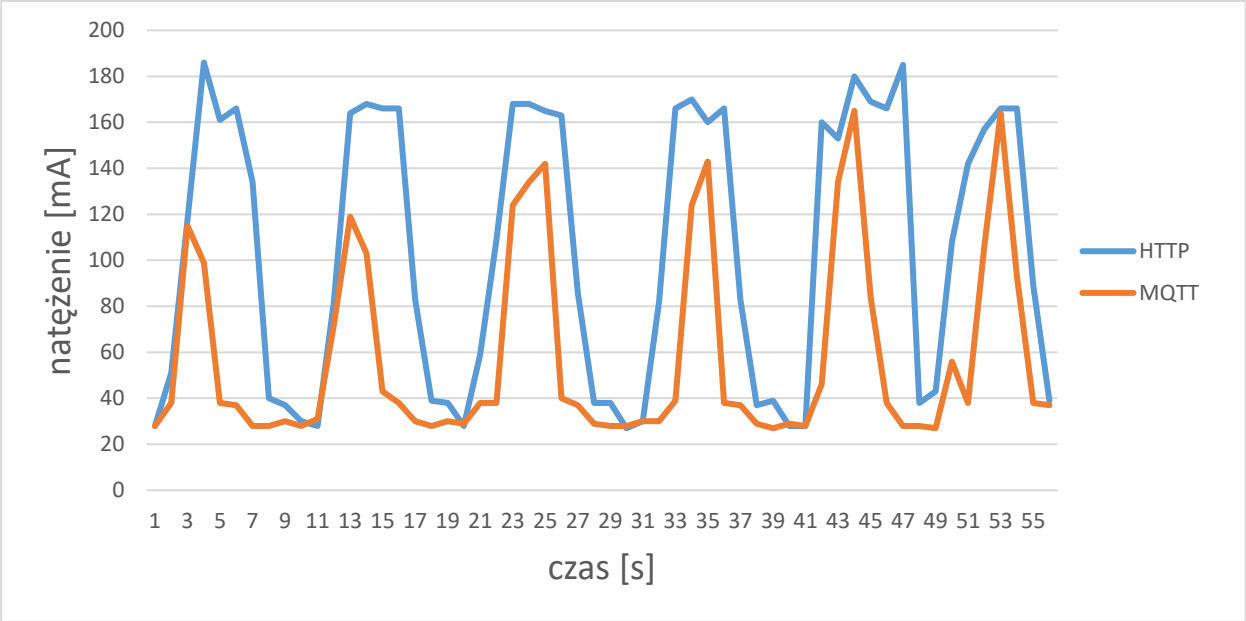


Rysunek 5.1. Prosty przykład przesyłania wiadomości pomiędzy klientami

Połączenie z serwerem jest zawsze nawiązywane przez klientów. Broker MQTT akceptuje żądania połączenia i pilnuje, aby do połączonych klientów docierały tylko wiadomości przesłane z tematami przez nich subskrybowanymi. Istnieje mechanizm, który umożliwia przechowywanie przez brokera ostatniej wiadomości przesłanej z danym tematem i ponowne wysyłanie jej za każdym razem gdy kolejny klient łączy się z brokerem i subskrybuje ten temat.

5.1.2. HTTP

Protokół HTTP w stosunku do MQTT znacząco zwiększa ilość konsumowanego przez urządzenie prądu. Na rys. 5.2 przedstawiono natężenie prądu pobieranego przez moduł SIM808 podczas przekazywania do serwera wiadomości w dziesięcio-sekundowych odstępach. Różnice w maksymalnym poborze prądu prawdopodobnie wynikają z małej częstotliwości próbkowania natężenia, które odbywało się co jedną sekundę, dlatego pomiaru tego nie należy traktować jako ilościową reprezentację pobieranego prądu a jedynie jakościowe przedstawienie różnicy między protokołami. Mimo tego, z przeprowadzonych pomiarów jednoznacznie wynika, że wysyłanie wiadomości przy pomocy protokołu MQTT zajmuje znacznie mniej czasu a zużycie prądu przez urządzenie jest mniejsze.



Rysunek 5.2. Natężenie prądu pobieranego przez moduł SIM808 w zależności od użytego protokołu

5.1.3. CoAP

Constrained Application Protocol (CoAP) jest protokołem przesyłu danych wyspecjalizowanym w komunikowaniu ze sobą urządzeń o niskich zasobach sprzętowych. CoAP został stworzony z myślą o mikrokontrolerach z niewielką ilością pamięci oraz opartych często na 8-bitowej architekturze[5]. CoAP charakteryzuje się niższym poborem prądu od MQTT[6]. Przesyłanie wiadomości oparte jest na architekturze request/response.

5.1.4. XMPP

Extensible Messaging and Presence Protocol (XMPP) został zaprojektowany pod kątem wymiany wiadomości w czasie rzeczywistym. Protokół ten wspiera zarówno architekturę publish/subscribe jak i request/response. Wiadomości przesyłane są w formacie XML, co wiąże się z potrzebą odpowiedniego przygotowania informacji a więc prowadzi do dodatkowego zużycia energii przez mikrokontroler. XMPP został ustandaryzowany przez IETF w 2004 roku i od tego czasu cieszył się bardzo dużą popularnością przy tworzeniu wszelkiego rodzaju komunikatorów internetowych. Ponieważ protokół ten z racji swojego wieku nie wspiera niektórych nowych rozwiązań, wiele firm odchodzi od jego stosowania. Przykładem jest Google, które ograniczyło swoje wsparcie dla XMPP[7] czy też Facebook, który tworząc popularną aplikację Messenger zrezygnował z tego protokołu, który był stosowany we wcześniejszych

wersjach komunikatora, na rzecz MQTT a decyzję swoją uzasadniali mniejszym opóźnieniem w dostarczaniu wiadomości i znaczną oszczędnością w kwestii zużycia prądu[8].

5.2. Wybór protokołu najlepiej pasującego do projektu

Opisane w rozdziale 3.1. protokoły występują w dwóch architekturach, request/response i publish/subscribe. Technologie opierające swoje działanie na metodologii publish/subscribe są wyposażone w gotowe rozwiązania, które mają za zadanie odpowiednie zarządzanie przesyłanymi informacjami tak, aby te trafiały do zainteresowanych stron. Z tego powodu ich zastosowanie jest dużo prostsze gdy nie jest możliwe bezpośrednie komunikowanie się urządzeń, ale potrzebny jest serwer, który będzie rozdysponowywał informacje.

Protokół HTTP został odrzucony, ponieważ jest wolniejszy od pozostałych co wiąże się z dużym poborem prądu. Dodatkowo architektura HTTP oparta na zasadzie request/response utrudnia implementację wymaganych rozwiązań.

Zastosowanie XMPP wiąże się z dodatkowym narzutem związanym z przygotowywaniem wiadomości w formacie XML. Oprócz tego, z racji wieku tego protokołu, nie można być pewnym dalszego wsparcia dla tej technologii.

Zarówno protokół XMPP jak i MQTT, za sprawą swojej lekkości, nadają się do zastosowania w projekcie opartym o mikrokontroler o ograniczonych zasobach oraz cechują się niskim poborem prądu, nieznacznie niższym w przypadku tego pierwszego. Do zastosowania w projekcie wybrano jednak MQTT, ponieważ jego architektura bardziej odpowiadała postawionym wymaganiom.

6. Budowa opaski

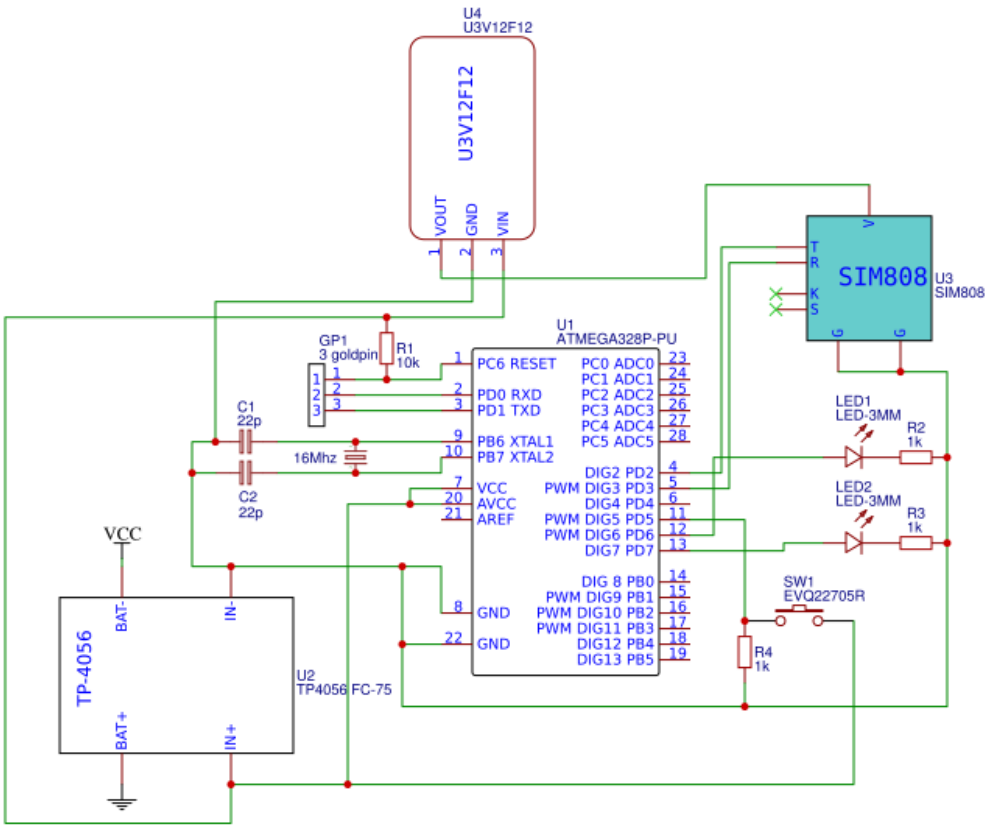
Przygotowanie prototypu obejmowało zarówno stworzenie działającego układu elektrycznego jak i zaprojektowanie i wykonanie ergonomicznej obudowy.

6.1. Układ elektryczny

Na rysunku 6.1 przedstawiono schemat urządzenia elektronicznego wewnątrz opaski z przyciskiem do wzywania pomocy. Porty 1, 2 i 3 (reset, RX i TX) połączone są z

wyprowadzeniem na płytce umożliwiającym programowanie mikrokontrolera. Do mikrokontrolera zostało dostarczone zewnętrzne źródło częstotliwości w postaci rezonatora kwarcowego (16 MHz). Nie jest to konieczne rozwiązanie, ponieważ Atmega ma wbudowany oscylator o częstotliwości 8 MHz, który jest wystarczający dla potrzeb projektu ale modyfikacja ta została dokonana aby można było w przyszłości rozwinąć projekt. Przycisk został podłączony do wyprowadzenia numer 11. Do portów 12 i 13 zostały podłączone diody, które są wykorzystywane w celu przekazywania informacji użytkownikowi.

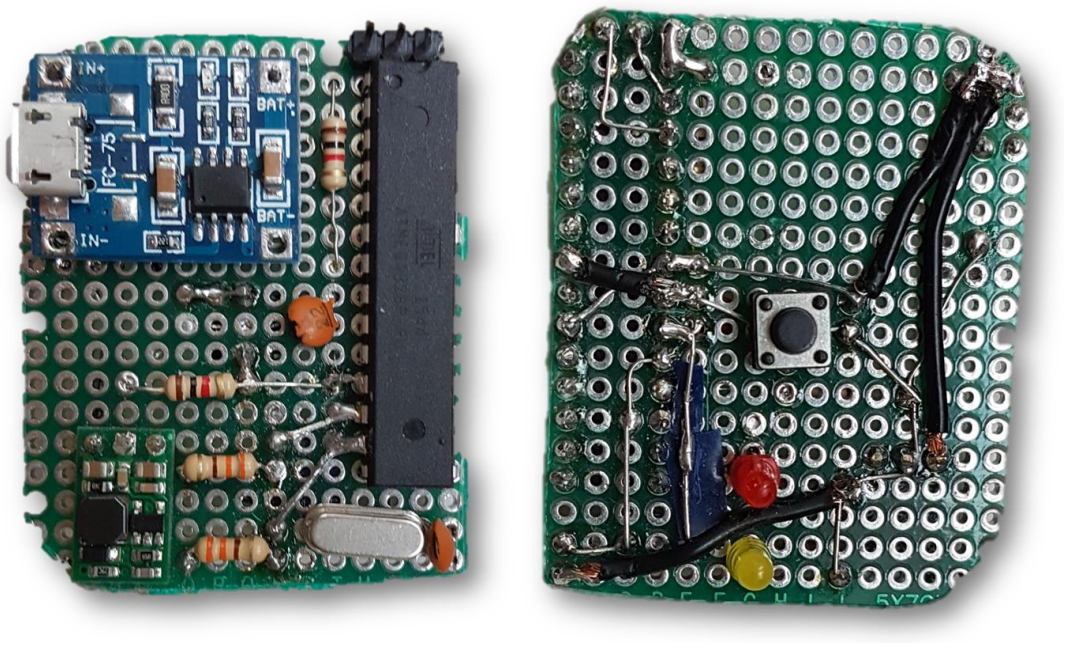
Mikrokontroler jest zasilany z baterii za pośrednictwem modułu ładującego TP-4056. Moduł GSM z racji, że wymaga napięcia 12 V, połączony jest z zasilaniem za pośrednictwem przetwornicy step-up podwyższającej napięcie do wymaganego.



Rysunek 6.1. Schemat układu elektrycznego

Prototyp układu został wykonany na dwustronnej płytce uniwersalnej, (rys. 6.2). Od wierzchniej strony znajdują się tylko przycisk oraz diody aby można je było umieścić w

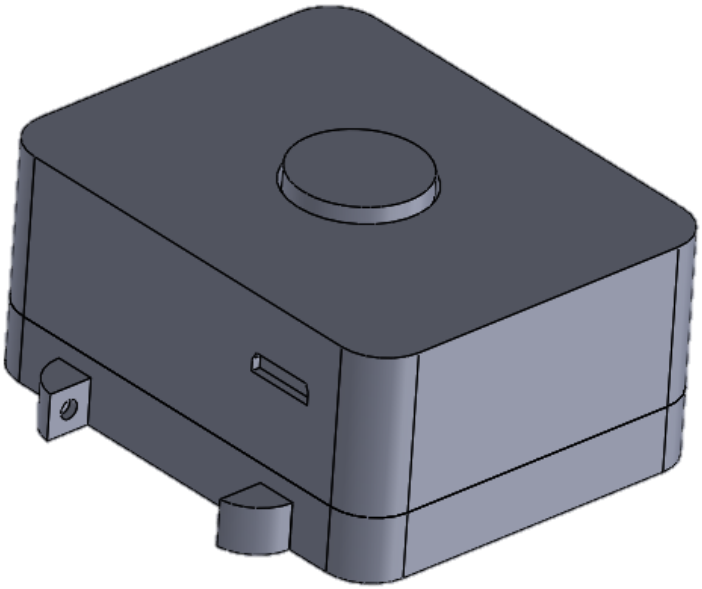
otworach na obudowie. Moduł BK-SIM808 z powodu swoich rozmiarów jest umieszczony osobno i połączony z resztą za pośrednictwem przewodów.



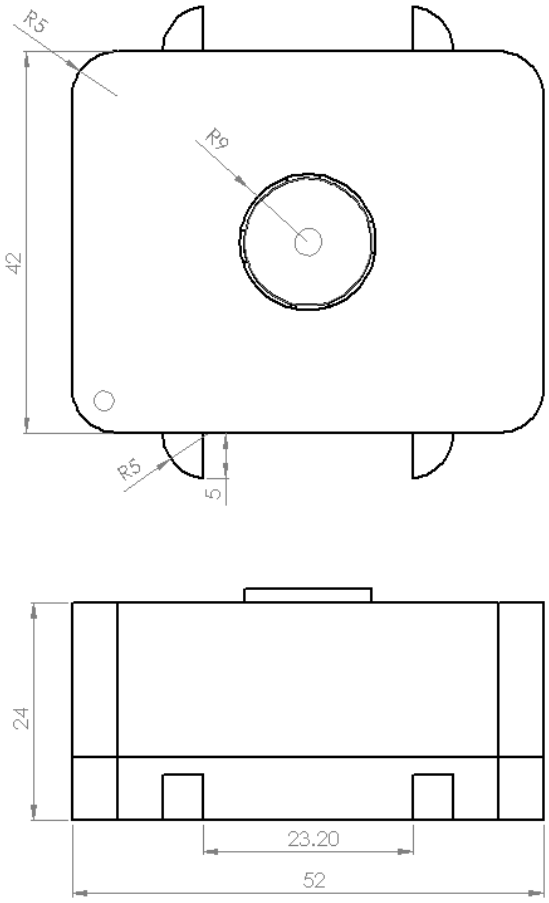
Rysunek 6.2 - prototyp układu elektrycznego

6.2. Obudowa

Obudowa została zaprojektowana w programie Solidworks 2017. Na rysunkach 6.3 oraz 6.4 przedstawiono wygląd obudowy oraz najważniejsze wymiary. Długość i szerokość urządzenia zostały dobrane tak aby jak najwygodniej przylegało do ręki. Grubość obudowy jest uzależniona rozmiarami układu elektronicznego i na obecnym etapie nie jest możliwe jej zmniejszenie bez zastosowania droższych rozwiązań.



Rysunek 6.3. Wizualizacja zaprojektowanej obudowy



Rysunek 6.4. Przedstawienie najważniejszych wymiarów obudowy

Prototyp obudowy został wykonany przy użyciu drukarki 3D. Konstrukcja składa się z trzech elementów (rys. 6.5) – podstawy, obudowy górnej oraz przycisku. Całość jest ze sobą skręcona za pomocą 4 śrub M2 znajdujących się w narożnikach konstrukcji i wkręcanych od spodniej strony urządzenia.

Podstawa posiada mocowania na standardowy pasek do zegarka o szerokości 22 mm, który można przytwierdzić przy pomocy teleskopów zegarkowych. Górna obudowa posiada z boku otwór umożliwiający ładowanie urządzenia, oraz z góry miejsce do mocowania przycisku.

Przycisk posiada specjalne wypustki pasujące do wcięć w obudowie. Mają one za zadanie zablokować możliwość wysunięcia się przycisku i dodatkowo stabilizować jego ruch aby nie dochodziło do przekrzywiania i blokowania tego elementu.

Rysunek 6.5. Elementy obudowy wydrukowane drukarką 3D

7. Oprogramowanie i biblioteki

7.1. Użyte oprogramowanie

Podczas tworzenia oprogramowania wykorzystano środowisko programistyczne Arduino IDE[9]. Program ten umożliwia pisanie kodu w językach C/C++, jego kompilację oraz

wgrywanie do mikrokontrolera za pośrednictwem płytki Arduino. Oprócz tego Arduino IDE zostało wykorzystane do wypalenia bootloadera umożliwiającego wgrywanie oprogramowania na mikrokontroler Atmega 328P bez potrzeby korzystania z zewnętrznego programatora.

Jako broker MQTT wykorzystano CloudMQTT[10]. Jest to wygodny i prosty w obsłudze broker dostępny online, który w darmowej wersji umożliwia połączenie się 5 klientów i pozwala na przesył danych do 10 Kbit/s. Na potrzeby budowy prototypu ograniczenia wersji darmowej nie stanowiły problemu.

Do testowania połączenia opaski z brokerem MQTT wykorzystano darmowy program MQTTBox[11]. Aplikacja ta umożliwia tworzenie własnych klientów, którzy mogą łączyć się z brokerem MQTT i subskrybować tematy oraz publikować wiadomości.

Aby ułatwić testowanie urządzenia korzystano z serwisu internetowego Google Maps[12], który był używany do sprawdzania poprawności wyznaczonych lokalizacji.

7.2. Wykorzystane biblioteki

Do implementacji programu odpowiedzialnego za funkcjonowanie opaski użyto dwóch bibliotek, bibliotekę pubsubclient[13], która implementuje interfejs ułatwiający komunikację z brokerem MQTT, oraz TinyGSM[14] ułatwiającą obsługę modułu SIM808.

7.2.1. Pubsubclient

Biblioteka Arduino, która umożliwia stworzenie klienta komunikującego się z brokerem MQTT. PubSubClient implementuje metody typowe dla klienta MQTT:

- *connect()*/*disconnect()* – umożliwiają nawiązanie/zerwanie połączenia z brokerem,
- *publish()* – publikuje wiadomość o określonym temacie,
- *subscribe()*/*unsubscribe()* – subskrybuje/przestaje subskrybować dany temat.

7.2.2. TinyGSM

Biblioteka Arduino ułatwiająca komunikację z różnymi modułami GSM, w tym ze stosowanym w projekcie modułem SIM808. Udostępnia ona interfejs typowy dla Arduino Client opakowując w ten sposób mniej przyjazną komunikację z modułem odbywającą się przy pomocy komend AT. TinyGSM umożliwia między innymi łatwe

przesyłanie danych przez TCP, wysyłanie SMS-ów i pozyskiwanie lokalizacji przy użyciu GPS.

8. Implementacja

8.1. Komunikacja z modułem Sim808

Komunikacja z modułem GSM odbywa się przy pomocy biblioteki TinyGsm. Ponieważ biblioteka obsługuje różne modele modułów GSM, wymagane jest zdefiniowanie używanego układu. Aby stworzyć obiekt klasy TinyGsm należy przekazać w konstruktorze port szeregowy, do którego podłączony jest moduł. Ponieważ porty sprzętowe, przeznaczone do komunikacji przez interfejs UART, są wykorzystywane przy testowaniu oraz debugowaniu oprogramowania, skorzystano z biblioteki SoftwareSerial w celu stworzenia wirtualnych portów RX i TX. Obiekt klasy TinyGSM zapewnia wiele metod służących do korzystania z wszystkich funkcjonalności modułu SIM808. Na fragmencie kodu 8.1. zaprezentowano tworzenie obiektu klasy TinyGsm oraz jej metody służące do restartowania modułu oraz wypisania podstawowych informacji na jego temat.

```
#include <SoftwareSerial.h>
#include <TinyGsmClient.h>

#define TINY_GSM_MODEM_SIM808

SoftwareSerial SerialAT(2, 3);

TinyGsm modem(SerialAT);

  modem.restart();
  SerialMon.print("Modem: ");
  SerialMon.println(modem.getModemInfo());
```

Fragment kodu 8.1.

8.2. Pozycja GPS

Dla większej dokładności każda wyznaczona pozycja jest efektem sześciu pomiarów, z których są odrzucane te, które mogą być efektem błędu pomiarowego a następnie z pozostałych współrzędnych liczona jest średnia.

8.2.1. Odczytywanie współrzędnych

Długość i szerokość geograficzna jest odczytywana przy pomocy metody getGPS() klasy TinyGsm dostarczonej przez bibliotekę TinyGSM. Po naciśnięciu przez użytkownika przycisku do wzywania pomocy opaska co 10 sekund dokonuje 6 pomiarów położenia i na ich podstawie, po odrzuceniu wyników, które mogą być nieprawidłowe z powodu błędu pomiarowego, przesyła lokalizację do klienta webowego. Oprócz tego, gdy przycisk alarmujący nie jest wciśnięty, urządzenie co pół minuty sprawdza aktualną pozycję, aby w przypadku gdy użytkownik wezwie pomoc będąc poza zasięgiem GPS, przesłać ostatnią zarejestrowaną lokalizację.

8.2.2. Odrzucanie potencjalnie niepoprawnych pomiarów

Aby wykluczyć pomiary, które znacznie różnią się od pozostałych skorzystano z testu Q-Dixona[15]. Test ten polega na obliczeniu parametru Q:

$$Q_n = \frac{różnica\ między\ sprawdzanym\ pomiarem\ a\ najbliższym\ wartością}{różnica\ między\ najmniejszym\ a\ największym\ pomiarem}$$

Tak obliczony współczynnik Q porównujemy z wartością tablicową (Tabela 8.1), która jest wyznaczona w zależności od ilości pomiarów i prawdopodobieństwa z jakim chcemy określić, czy dany pomiar jest błędny. Jeżeli obliczona wartość jest większa od tablicowej to sprawdzany pomiar jest uznawany za nieprawidłowy.

	Ilość pomiarów							
	3	4	5	6	7	8	9	10
$Q_{90\%}$	0.941	0.765	0.642	0.560	0.507	0.468	0.437	0.412
$Q_{95\%}$	0.970	0.829	0.710	0.625	0.568	0.526	0.493	0.466
$Q_{99\%}$	0.994	0.926	0.821	0.740	0.680	0.634	0.598	0.568

Tabela 8.1 Wartości graniczne testu Q-Dixona

Implementacja tego testu składa się z kilku etapów. Najpierw sprawdzane są szerokości geograficzne od najmniejszych wartości. Aby tego dokonać współrzędne są sortowane według szerokości geograficznej przy pomocy funkcji qsort(), ze standardowej biblioteki stdlib.h, a następnie sprawdzane są kolejne pomiary do momentu, w którym pomiar nie zostanie odrzucony. Taka sama procedura powtarzana

jest dla długości geograficznych (z pominięciem wcześniej odrzuconych wartości). Algorytm przedstawiono na fragmencie kodu 8.2.

```
qsort (coords, totalCoords, 2*sizeof(float), compareLat);

    *first=0;
    *last=totalCoords;
    float rangeLat, rangeLon;

    while(*first < *last - 1){

        rangeLat = coords[0][*last]-coords[0][*last];
        float QLat=fabs(coords[0][*first]-coords[0][*first+1])/rangeLat;
        if(QLat > Q99[totalCoords-3]){

            *first ++;
        }
        else{
            break;
        }
    }

    qsort (coords[*first], totalCoords-*first, 2*sizeof(float), compareLon);

    while(*first < *last - 1){

        rangeLon = coords[1][*last]-coords[1][*last];

        float QLon=fabs(coords[1][*first]-coords[1][*first+1])/rangeLon;

        if(QLon > Q99[totalCoords-3]){

            *first ++;
        }
        else{
            break;
        }
    }
```

Fragment kodu 8.2. Ocenianie poprawności pomiarów dla najmniejszych wartości współrzędnych

W analogiczny sposób wykrywane są błędne pomiary wśród największych wartości współrzędnych. Ponieważ w wyniku działania tego algorytmu, niepoprawne wartości zawsze będą znajdować się na początku lub na końcu tablicy współrzędnych, funkcja zwraca numery indeksów pierwszej i ostatniej poprawnej współrzędnej.

8.3. Połączenie z brokerem MQTT oraz publikowanie wiadomości z lokalizacją

Aby korzystać z protokołu MQTT zastosowano bibliotekę PubSubClient. Konstruktor klasy PubSubClient przyjmuje obiekt klasy Client[16]. W tym przypadku przekazano obiekt klasy TinyGsmClient, która dziedziczy po klasie Client i przeładowuje jej metody pod kątem współpracy z modułem GSM.

Połączenie z siecią komórkową jest nawiązywane przy pomocy metody gprsConnect(). Wymaga ona podania APN (Acces Point Name) oraz nazwy użytkownika i hasła. Parametry te są zależne od operatora sieci komórkowej, z którego usług korzystamy.

Aby połączyć się z brokerem MQTT należy podać adres serwera oraz port a następnie użyć metody connect() podając nazwę użytkownika oraz hasło. Na fragmencie kodu 8.3 zaprezentowano proces łączenia się z brokerem MQTT.

```
TinyGsmClient client(modem);
PubSubClient mqtt(client);


SerialMon.print("Connecting to ");
SerialMon.print(apn);
if (!modem.gprsConnect(apn, user, pass)) {
    SerialMon.println(" can't connect");
    return;
}

SerialMon.println(" device succesfully conected to network");

mqtt.setServer(broker, port);

boolean status = mqtt.connect("GsmClientName", mqttUser, mqttPassword);
if (status == false) {

    SerialMon.println(" conecting to MQTT failed");
    return;
}
```

Fragment kodu 8.3. Nawiązywanie połączenia z brokerem MQTT.

W celu wysłania wiadomości należy skorzystać z metody publish, która przyjmuje jako argumenty nazwę tematu oraz treść wiadomości, w tym przypadku składającej się z długości i szerokości geograficznej oddzielonych spacją.

8.4. Powiadomienia SMS

Podstawowym zadaniem opaski jest przesłanie powiadomienia w postaci smsa do opiekunów osoby starszej. W chwili naciśnięcia przez użytkownika urządzenia przycisku wywoływana jest funkcja, która przesyła wiadomości na wszystkie zapisane numery telefonów (fragment kodu 8.4). W przypadku gdy wysyłanie nie powiedzie się opaska ponowi próbę.

```
bool sendSMSNotifictions(){

    for(int i=0; i< numbersCount; i++){

        bool st=0;
        while(!st){

            st =modem.sendSMS(numbers[i], msg);
        }
    }
}
```

Fragment kodu 8.4. Przesyłanie wiadomości SMS

8.5. Przycisk i diody

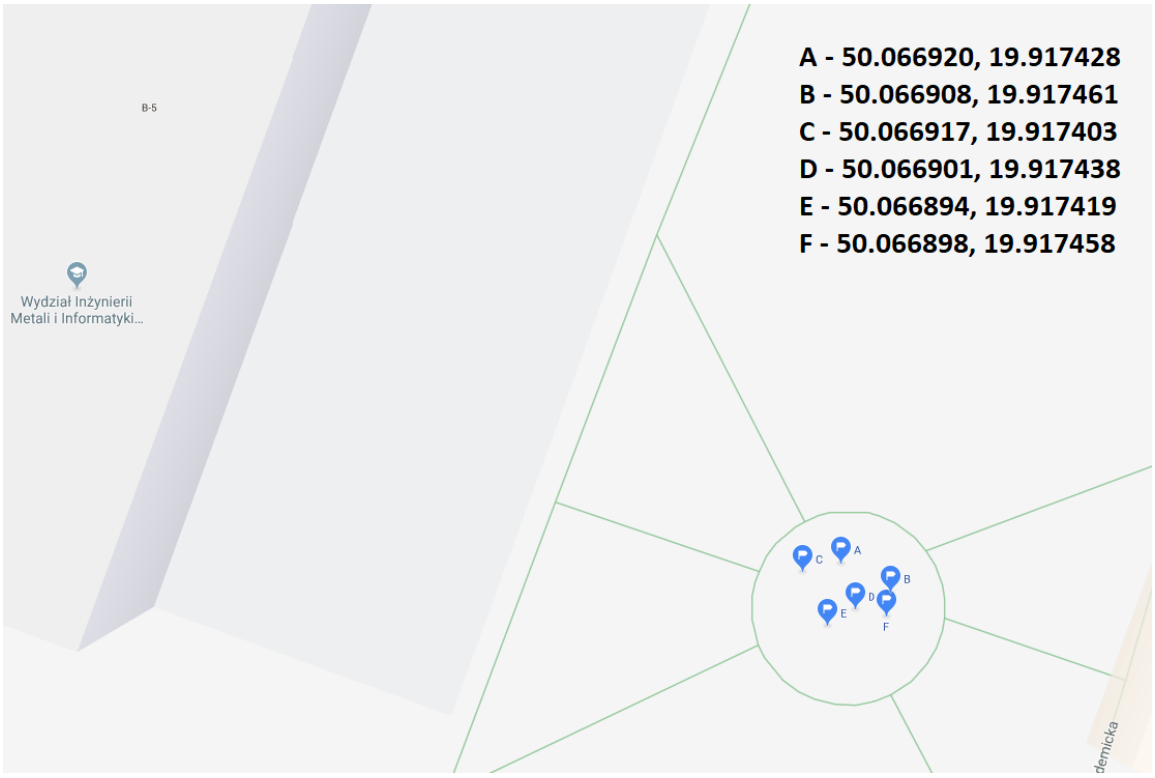
Urządzenie posiada przycisk i dwie diody, żółtą i czerwoną, które służą do sygnalizowania użytkownikowi podejmowanych akcji. Ciągłe świecenie zielonej diody oznacza prawidłowe funkcjonowanie opaski natomiast migające światło oznacza, że opaska próbuje nawiązać połączenie z siecią. Czerwona dioda miga po naciśnięciu przycisku alarmowego aby zasygnalizować, że urządzenie przeszło w tryb wzywania pomocy. Ponowne naciśnięcie przycisku sprawi, że opaska zaprzestanie wzywania pomocy.

9. Testowanie urządzenia

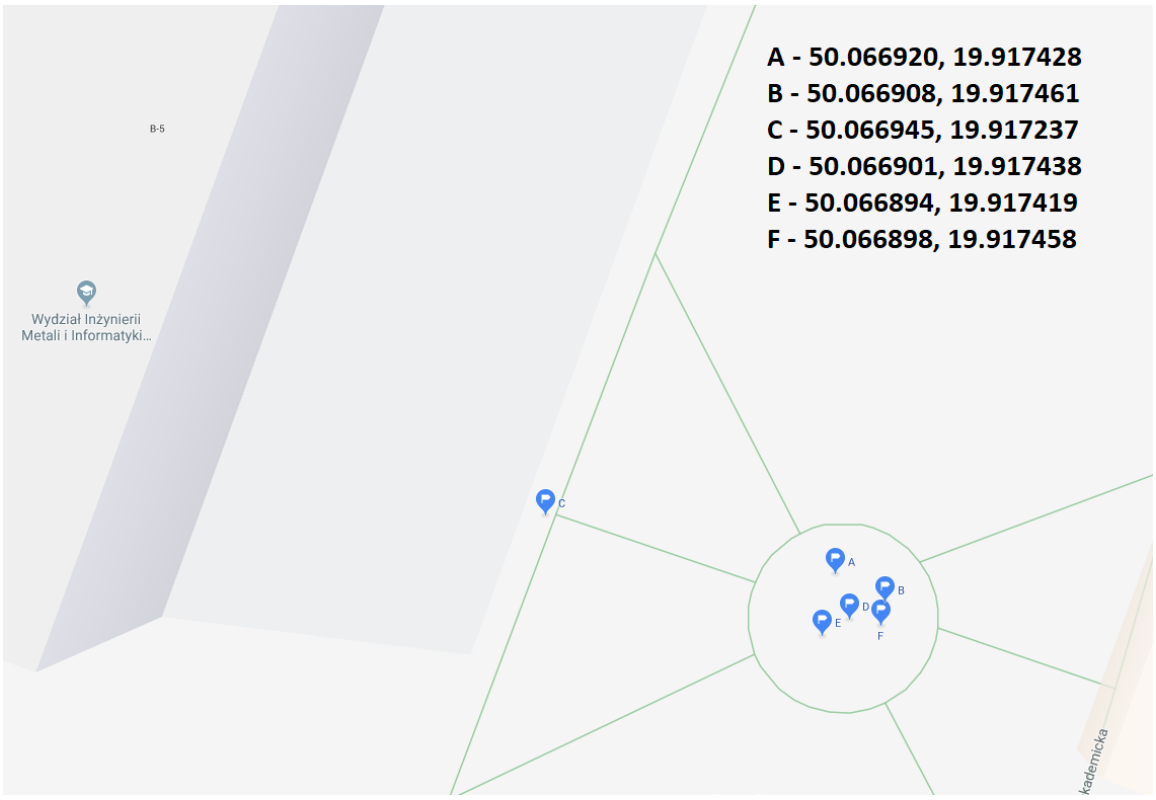
Aby uniknąć błędów w działaniu urządzenia, było ono testowane zarówno w trakcie procesu implementacji oprogramowania jak i po wykonaniu całego projektu.

9.1. Testy algorytmu odrzucającego błędne współrzędne

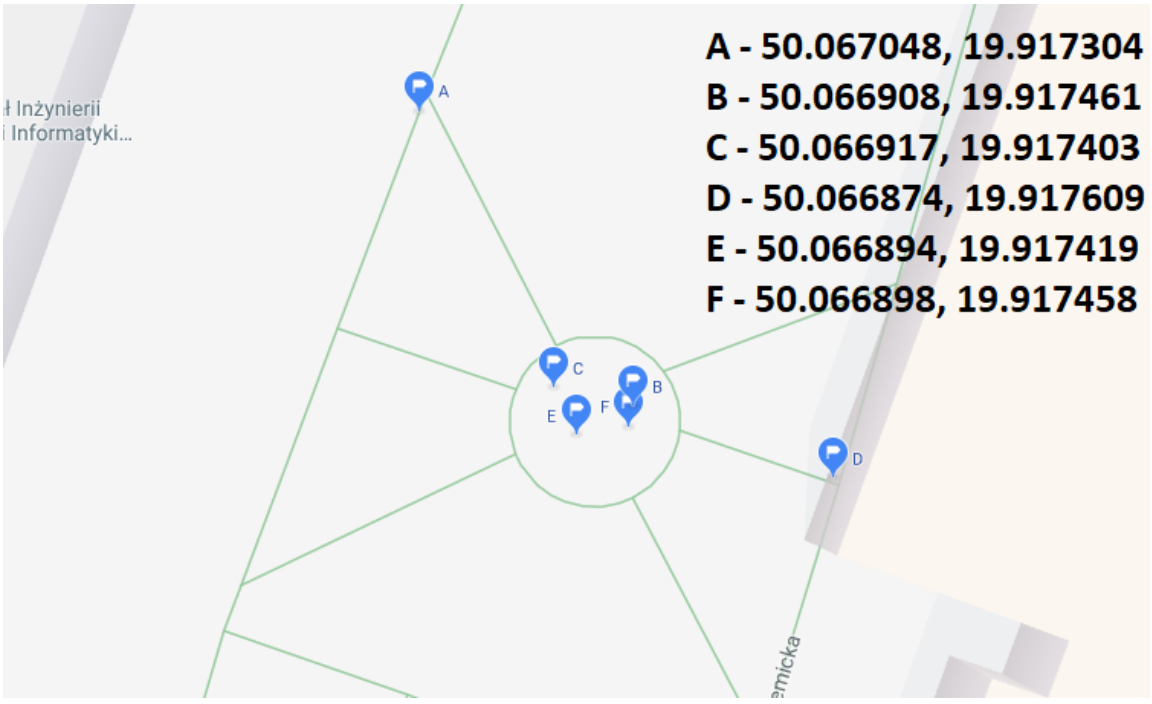
Algorytm stworzony na podstawie testu Q-Dixona został sprawdzony dla kilku możliwych wariantów. Przygotowano trzy zestawy do testowania, każdy zawierający po 6 współrzędnych. Pierwszy zestaw współrzędnych (rys. 9.1) zawierał współrzędne, wśród których żadna nie odstawała znacząco od reszty. Według założeń algorytmu żadna wartość nie powinna zostać odrzucona. Drugi zestaw współrzędnych posiadał punkt, oznaczony na rys. 9.2 jako C, znacznie oddalony w stosunku do odległości dzielących pozostałe współrzędne. Pomiar tren powinien zostać odrzucony przez algorytm. W ostatnim zestawie znajdowały się dwa punkty, oznaczone na rys. 9.3 jako A i D, które nie powinny być brane pod uwagę.



Rysunek 9.1 - zestaw testowy z poprawnymi współrzędnymi



Rysunek 9.2 - Zestaw testowy z jedną błędną współrzędną



Rysunek 9.3 - zestaw testowy z dwoma niepoprawnymi współrzędnymi

Wyniki testów zostały zebrane w tabeli 9.1. Dla wszystkich testów uzyskano prawidłowe rezultaty.

Numer testu	Testowane współrzędne	Odrzucone współrzędne
1	A - 50.066920, 19.917428 B - 50.066908, 19.917461 C - 50.066917, 19.917403 D - 50.066901, 19.917438 E - 50.066894, 19.917419 F - 50.066898, 19.917458	brak
2	A - 50.066920, 19.917428 B - 50.066908, 19.917461 C - 50.066945, 19.917238 D - 50.066901, 19.917438 E - 50.066894, 19.917419 F - 50.066898, 19.917458	C - 50.066945,19.917238
3	A - 50.067048, 19.917304 B - 50.066908, 19.917461 C - 50.066917, 19.917403 D - 50.066874, 19.917609 E - 50.066894, 19.917419 F - 50.066898, 19.917458	B - 50.066874,19.917609 A - 50.067048,19.917304

Tabela 9.1 – Wyniki testów algorytmu odrzucającego błędne pomiary

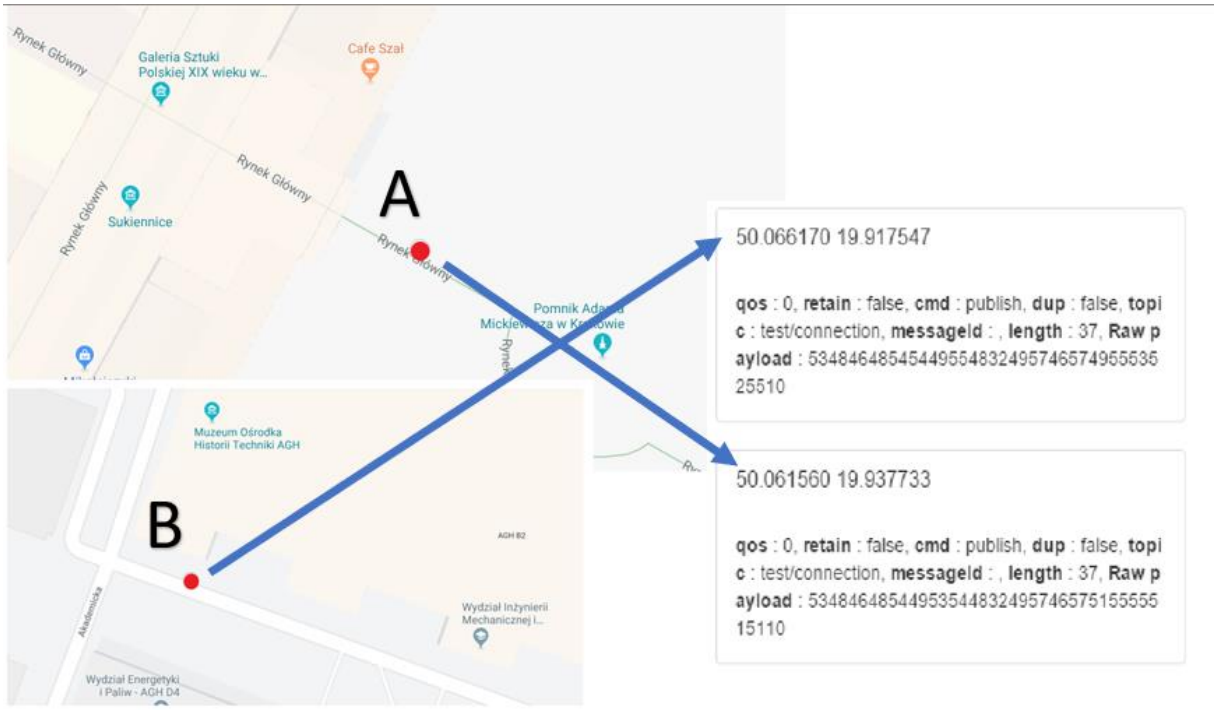
9.2. Test połączenia z brokerem MQTT

Podczas pisania oprogramowania wykonywano wiele testów mających na celu sprawdzenie czy urządzenie łączy się poprawnie z brokerem MQTT. Wysyłano testowe wiadomości a następnie sprawdzano przy pomocy MQTTBox czy docierały one do subskrybentów. Rysunek 9.4 Przedstawia przykładową odczytaną wiadomość.

Rysunek 9.4 - przykładowa wiadomość otrzymana przez klienta stworzonego w MQTTBox

Po wykonaniu prototypu sprawdzano, czy otrzymane przez klienta współrzędne wysłane po naciśnięciu przycisku alarmowego są poprawne. Testy przeprowadzano w różnych lokalizacjach, zarówno we wnętrzu budynku jak i na otwartych przestrzeniach.

Na rys. 9.5 przedstawiono dwa z przeprowadzonych testów. Test oznaczony literą A został przeprowadzony na Rynku Głównym w Krakowie. Otrzymane współrzędne dokładnie odpowiadały miejscu, w którym naciśnięto przycisk do wzywania pomocy. W przypadku B przycisk został naciśnięty wewnątrz budynku na jego najniższej kondygnacji. Z powodu braku zasięgu GPS urządzenie wysłało ostatnie zarejestrowane położenie, które znajdowało się w pobliżu wejścia do budynku.



Rysunek 9.5 - Testy poprawności lokalizacji

10. Podsumowanie

Celem niniejszej pracy było zaprojektowanie oraz zaprogramowanie opaski na rękę do wzywania pomocy. Przedstawiono oraz uzasadniono wybór użytych technologii i komponentów elektrycznych, opisano implementację oprogramowania oraz zaprezentowano konstrukcję.

Wykonane urządzenie zostało stworzone jako prototyp, który miał na celu sprawdzenie wybranych rozwiązań oraz zdiagnozowanie problemów wiążących się z tego typu technologiami. Wykonana praca umożliwi dalszy rozwój projektu a w efekcie stworzenie opaski zoptymalizowanej pod kątem zużycia prądu, niezawodności, wygody użytkowania oraz kosztu wykonania. W tym celu niezbędne będzie zastosowanie innego modułu GSM opartego na układzie SIM808, który umożliwi wykorzystanie możliwości ładowania baterii i nie będzie wymagał tak wysokiego napięcia. Dodatkowo, układ elektryczny powinien zostać wykonany w oparciu o specjalnie zaprojektowaną płytkę PCB aby całość zajmowała mniej miejsca co pozwoli zmniejszyć wymiary i wagę opaski.

11. Literatura

[1]http://www.gsmalarmsystem.com/UploadFiles/20150415/King%20Pigeon%20Wireless%20Panic%20Button%20EM-60%20EM-70%20EM-90%20EM-100.pdf

[2] https://www.comarch.pl/healthcare/produkty/zuo/bransoletka-zycia/

[3] https://cdn.instructables.com/ORIG/FAO/80RU/IXLALERK/FAO80RUIXLALERK.pdf

[4] *MQTT Version 3.1.1 Plus Errata 01*. Edited by Andrew Banks and Rahul Gupta. 10 December 2015. OASIS Standard Incorporating Approved Errata 01. http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/errata01/os/mqtt-v3.1.1-errata01-os-complete.html. Latest version: http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/mqtt-v3.1.1.html.

[5] https://tools.ietf.org/html/rfc7252

[6] S. Bandyopadhyay, A. Bhattacharyya. (2013). Lightweight Internet Protocols for Web Enablement of Sensors using Constrained Gateway Devices

[7]https://www.zdnet.com/article/google-moves-away-from-the-xmpp-open-messaging-standard/

[8]https://www.facebook.com/notes/facebook-engineering/building-facebook-messenger/10150259350998920

[9] https://www.arduino.cc/en/Main/Software

[10] https://www.cloudmqtt.com/

[11] http://workswithweb.com/mqttbox.html

[12] https://www.google.com/maps/about/

[13] https://github.com/knolleary/pubsubclient

[14] https://github.com/vshymanskyy/TinyGSM

[15] Dean, R. B., & Dixon, W. J. (1951). Simplified Statistics for Small Numbers of Observations. Analytical Chemistry, 23(4), 636–638. doi:10.1021/ac60052a025

[16] https://www.arduino.cc/en/Reference/ClientConstructor