

Politechnika Warszawska

W Y D Z I A Ł E L E K T R Y C Z N Y



Instytut Elektrotechniki Teoretycznej i Systemów Informacyjno-Pomiarowych

Praca dyplomowa magisterska

na kierunku Informatyka Stosowana
w specjalności Inżynieria Oprogramowania

Nowe możliwości Bluetooth 5 w aplikacjach Internetu Rzeczy

inż. Krzysztof Paweł Kruk
numer albumu 301140

promotor
dr inż. Łukasz Makowski

WARSZAWA 2022

Nowe możliwości Bluetooth 5 w aplikacjach Internetu Rzeczy

Streszczenie

To jest streszczenie. To jest trochę za krótkie, jako że powinno zająć całą stronę.

 Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetur id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

 Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

 Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

 Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetur adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Słowa kluczowe: Bluetooth, Bluetooth Low Energy, Bluetooth Mesh, Packet Error Rate

New Bluetooth 5 features in Internet of Things applications

Abstract

I am going to fill this abstract with some English text. I promise :)

 Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetur id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

 Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

 Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

 Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetur adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Keywords: Bluetooth, Bluetooth Low Energy, Bluetooth Mesh, Packet Error Rate

Spis treści

1 Wstęp	9
2 Komunikacja radiowa bliskiego zasięgu	11
2.1 ZigBee	11
2.2 OpenThread	11
2.3 Bluetooth Low Energy	11
2.3.1 Porównanie przedstawionych standardów	11
3 Założenia teoretyczne	13
3.1 Zużycie energii	13
3.2 Packet Error Rate	13
3.2.1 Definicja pakietu	14
3.2.2 Wpływ tła radiowego na model	15
4 Część doświadczalna	17
4.1 Przygotowanie eksperymentu	18
4.1.1 Sprzęt i oprzyrządowanie	18
4.1.2 Narzędzia i elementy dodatkowe	20
4.1.3 Oprogramowanie mikrokontrolera	23
4.1.4 Oprogramowanie PC	31
4.2 Badanie zużycia energii	35
4.2.1 Metodologia badania	36
4.2.2 BT Low Energy - Usługa Heart Rate	38
4.2.3 BLE Mesh - Model Generic OnOff	41
4.3 Badanie Packet Error Rate	44
4.3.1 Metodologia badania	44
4.3.2 Zależność PER względem częstości zapytań	48
4.3.3 Zależność PER względem odległości między węzłami	51
5 Podsumowanie	55

Bibliografia	57
Wykaz skrótów i symboli	59
Spis rysunków	61
Spis tablic	63
Spis załączników	65

Rozdział 1

Wstęp

Wstęp TBD

Rozdział 2

Komunikacja radiowa bliskiego zasięgu

2.1 ZigBee

2.2 OpenThread

2.3 Bluetooth Low Energy

2.3.1 Porównanie przedstawionych standardów

Rozdział 3

Założenia teoretyczne

3.1 Zużycie energii

[16] jako podstawa dla części teoretycznej. Spróbować symulacji zużycia energii z użyciem PCC w Cube'ie. Dodatkowo, oprzeć się na literaturze i artykułach naukowych, które poruszają tę kwestię - przynajmniej jeden.

Opisać tryby działania zarówno te przewidywane przez standard jak i te wprowadzone przez ST (niestety, jedyną dokumentacją jest kod).

3.2 Packet Error Rate

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

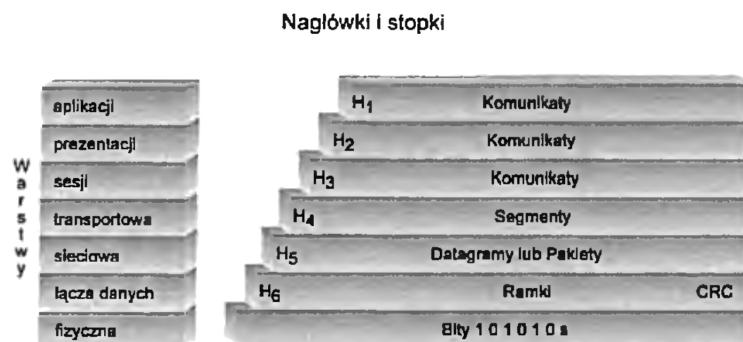
Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing

semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

3.2.1 Definicja pakietu

Przed przystąpieniem do badań należy precyzyjnie zdefiniować wykorzystywaną terminologię.

Pakietem nazywamy pojedynczą porcję danych przetwarzaną na poziomie warstwy sieciowej modelu ISO OSI [4]. Warstwa ta umożliwia routing, adresowanie logiczne oraz przetwarzanie i dostarczanie pakietów.



Rysunek 1. Model ISO OSI wraz i odpowiadające mu nazwy porcji danych. Źródło: [4]

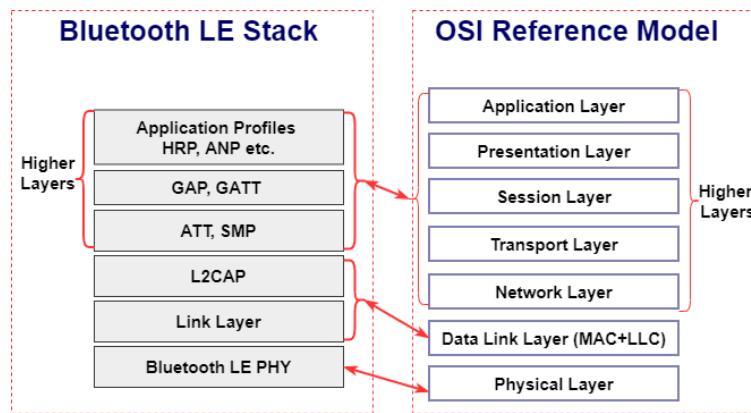
Bluetooth Low Energy (BLE) wprowadza własną nomenklaturę dla poszczególnych warstw sieciowych. Jest to o tyle istotne, iż standard ten nie zapewnia odpowiadających modelowi ISO OSI warstw jeden-do-jednego. Część z tych warstw jest agregowanych w zespół protokołów wyższych warstw - Rysunek 2.

Bazując na definicji modelu OSI oraz stosie BLE, pakietem można nazwać wiadomości będące możliwie blisko warstwy *Link Layer (LL)*. Podobną definicję prezentuje dokumentacja ST: „Pakiet to pojedyncza oznaczona wiadomość wysłana przez jedno i odebrana przez co najmniej jedno urządzenie.”¹ [15]

BLE Mesh dodatkowo wprowadza własne dodatkowe warstwy komunikacji, biorąc za podstawę stos BLE [2]. Każda z wymienionych warstw jest hermetyzowana za pośrednictwem dostarczanego przez producenta Application Programming Interface (API). Uwzględniając zamknięcie middleware'u i dwu-procesorową architekturę mikrokontrolera STM32WB55, oznacza to brak możliwości bezpośredniego nasłuchiwanego pakietów w warstwie LL.

Uwzględniając powyższe czynniki, *pakietem* dla BLE Mesh nazywana będzie wiadomość najbliższa warstwie LL. W przypadku stworzonego oprogramowania, oznacza to odbiór komunikatu odebranego

¹Tłumaczenie własne



Rysunek 2. Zestawienie stosu BLE i modelu ISO OSI. Źródło: [1]

jako zdarzenie zarejestrowane przez koprocesor Cortex-M0, będący integralną częścią mikrokontrolera STM32WB55 odpowiadający za obsługę radia.

Definicja Packet Error Rate

Posiadając definicję pakietu, Packet Error Rate (PER) możliwe staje się zdefiniowanie wzoru, a zarazem znaczenia głównego celu badań.

PER jest miarą ilości błędnych pakietów w proporcji do wszystkich wysłanych pakietów, zgodnie ze wzorem:

$$PER = \frac{s - r}{s} \cdot 100\% \quad (1)$$

gdzie:

s is ilość wysłanych pakietów

r is ilość odebranych pakietów

s-r - ilość niepoprawnych/błędnych pakietów

Powyższy wzór stanowi podstawę eksperymentu pozwalającego wyznaczyć jakość łącza w zależności od wybranych parametrów zmiennych.

3.2.2 Wpływ tła radiowego na model

* prezentacja hipotez * przegląd literatury - jeśli jakaś istnieje... * strefa Fresnela * wpływ środowiska

Rozdział 4

Część doświadczalna

Treść tego rozdziału przedstawia wszelkie kroki podjęte celem realizacji dwóch postawionych zadań: badania zużycia energii oraz PER. Wprowadzając czytelnika w aspekty przygotowawcze eksperymentu i technikalia z nimi związane, przechodzi się do prezentacji wyników i powiązanych analiz opracowanych wykresów.

Pierwszy podrozdział wprowadza w aspekty czysto techniczne i sprzętowe. Uzasadnia wybór platformy jak i oprogramowania w tym Integrated Development Environment - Zintegrowane środowisko programistyczne (IDE). Przedstawia się problem zasilania zestawów uruchomieniowych podczas badań terenowych, jednocześnie prezentując praktyczne, proste i niskokosztowe rozwiązanie problemów. Ze względu na kruchość elementów elektronicznych, wprowadza się projekt a następnie omawia fizyczne wykonanie autorskiej obudowy. Jednakże, głównym elementem tego rozdziału jest skoncentrowana na oprogramowaniu. Zarówno oprogramowaniu mikrokontrolerów jak i PC. Przedstawia się wprowadzone funkcjonalności oraz definiuje ich sposób użycia. Opisuje się kwestię kalibracji uzasadniając potrzebą i potwierdzając jej skuteczność obliczeniami arytmetycznymi bazując na równaniu uzyskanym drogą inżynierii wstępnej. Ostatnim aspektem poruszonym w tym punkcie jest aplikacja PC i jej interfejs użytkownika.

Drugi podrozdział, prezentuje badanie zużycie energii jako całość. Wprowadzi się metodologię według której dokonano akwizycji danych. Następnie, prezentuje się wykresy podając opis do każdego z nich. Badania oparto o dwa główne aspekty – konsumpcja energii przez przez usługę BLE Heart Rate (HRT) oraz model BLE Mesh. Ostatecznie, zestawia się zebrane w ten sposób dane.

Ostatni podrozdział dotyczy eksperymentu PER. Analogicznie do wyżej wymienionego doświadczenia, wprowadza metodologię badań oraz określa kategorie zbieranych zmiennych i definiuje się parametry stałej transmisji. Zebrane dane w próbach terenowych następnie są dzielone na dwie kategorie: PER względem częstości zapytań oraz PER względem odległości między węzłami. Wyniki, w postaci szeregu wykresów, są następnie omawiane.

4.1 Przygotowanie eksperymentu

Eksperymentalna część pracy bezsprzecznie wymaga przygotowania odpowiedniego zestawu sprzętu i narzędzi. Konieczne jest wybranie odpowiedniej platformy sprzętowej wspierającej komunikację bezprzewodową Bluetooth Low Energy w wersji minimum 5.0. Niezbędne jest również oprzyrządowanie pomiarowe, które umożliwia pomiar natężeń prądu poniżej 1mA, ze względu na energooszczędność urządzeń BLE.

Badania zużycia energii wymagają aparatury, która w pełni zarejestruje minima i maksima poboru prądu przy niskich błędach pomiarowych. Na podstawie otrzymanych danych wykonane zostaną właściwe obliczenia ukazujące zużycie energii przez urządzenie dla wielu trybów działania.

Eksperyment PER wymaga przygotowania wielu zestawów uruchomieniowych obsługujących komunikację BLE w konfiguracji Mesh. Dodatkowo, wymagane jest stworzenie dedykowanego oprogramowania na mikrokontroler jak i komputer osobisty. Jest to niezbędne w celu kontroli przepływu doświadczenia jak i akwizycji danych.

Niniejszy rozdział omawia zakres poniesionych przygotowań do przeprowadzenia właściwych eksperymentów.

4.1.1 Sprzęt i oprzyrządowanie

Próby doświadczalne oparto o produkty firmy STMicroelectronics (krócej: ST). Jest to europejska firma obejmująca swym portfolio projektowanie i produkcję elementów elektronicznych. Jedną z ich linii produktów są 32-bitowe mikrokontrolery architektury ARM, znane również jako STM32. Są to rozwiązania cenne na rynku.

Ze względu na powszechność stosowanych rozwiązań, jednak nie ograniczając się wyłącznie do tego kryterium, wybrano mikrokontroler z rodziny STM32WB. Gotowe, komercyjnie dostępne zestawy ewaluacyjne oraz zintegrowany ekosystem stanowią doskonałą podstawę dla badań Bluetooth Low Energy.

P-NUCLEO-WB55

Badania Bluetooth Low Energy wymagały wyboru platformy, która umożliwia eksperymentalną weryfikację wybranych celów badawczych. Ostatecznie, zdecydowano się na wykorzystanie mikrokontrolera *STM32WB55RG*. W celu zapewnienia powtarzalności eksperymentu jak i ze względu na ograniczenia czasowe, docelową platformą badawczą stał się zestaw uruchomieniowy *P-NUCLEO-WB55* [3].

Zestaw ten zgodny jest ze specyfikacją Bluetooth Low Energy v5.0. Dodatkowo, wspiera on inne standardy komunikacji, m.in. Zigbee [12]. Ten fakt może zostać wykorzystany w celu bezpośredniego porównania różnych stow komunikacji bezprzewodowej. Nie jest to jednak celem niniejszej pracy, a stanowi możliwość jej dalszego rozwinięcia.



Rysunek 3. Zestaw uruchomieniowy P-NUCLEO-WB55

X-NUCLEO-LPM01A

Płytkę rozszerzeń X-NUCLEO-LPM01A spełnia wszelkie oczekiwania dotyczące możliwości pomiarowych stawianych przed projektem. Wg oficjalnej dokumentacji [17], układ oferuje:

- Programowalne źródło napięciowe 1,8V do 3,3V
- Dynamiczne próbkowanie w zakresie od 100nA 50mA przy maksymalnej częstotliwości 100kHz z 2%-ową dokładnością pomiarów
- Pomiar statyczny natężenia prądu do 200mA
- Integracja z aplikacją do dedykowaną aplikacją akwizycji danych [9]

Moduł ten nie ogranicza się tylko do wykorzystywania autorskich złącz firmy ST. Schemat wyprowadzeń pinów umożliwia wykorzystanie popularnych platform takich jak Arduino¹. Co istotniejsze, wybrana płytka umożliwia zasilanie dowolnego układu dzięki wyprowadzeniu źródła napięciowego za pośrednictwem pinów (za dokumentacją: złącze CN14). Ten fakt został wykorzystany w badaniach, pozwalając uniknąć kłopotliwej konfiguracji P-NUCLEO-WB55 wymagającej tworzenia dodatkowych ścieżek lutowanych.

Dodatkową cechą tego modułu jest akwizycja danych z użyciem komputera osobistego. Dzięki dedykowanej aplikacji, dostarczanej wraz z modułem, możliwa jest akwizycja danych w czasie rzeczywistym przy wybranych częstotliwościach próbkowania. Zebrane w ten sposób dane służą dalszym analizom, co zostało wykorzystane w niniejszej pracy.

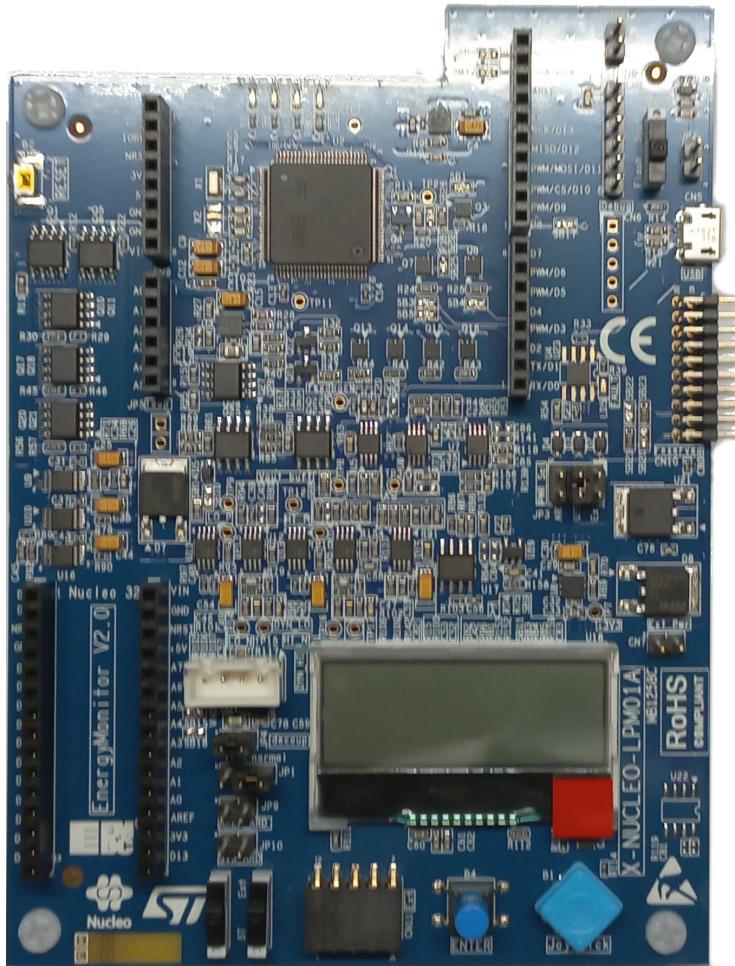
Narzędzia i firmware firmy ST

Firma ST wraz z zestawem uruchomieniowym udostępnia pełne zintegrowane środowisko programistyczne oraz niezbędne biblioteki i certyfikowany firmware:

- **STM32CubeIDE** [8] – multiplatformowe zintegrowane środowisko programistyczne dostarczane przez ST bazujące na otwartoźródłowym środowisku Eclipse².
- **STM32CubeProgrammer** [10] – narzędzie umożliwiające wykonywanie operacji odczytu, zapisu i weryfikacji skompilowanego oprogramowania produktów STM32.

¹<https://www.arduino.cc/>

²<https://www.eclipse.org/ide/>



Rysunek 4. Zestaw pomiarowy X-NUCLEO-LPM01A

- **STM32CubeMonitor-Power** [9] – oprogramowanie służące akwizycji danych pod postacią chwilowego poboru prądu elektrycznego m.in. w zestawie X-NUCLEO-LPM01A Rysunek: 4.
- **Firmware STM32CubeWB** [11] – zestaw bibliotek, narzędzi oraz przykładów przeznaczonych dla mikrokontrolerów rodziny STM32WB. W skład tego repozytorium wchodzą zależności takie jak: skompilowany, zamknięty firmware ko-processora dla różnych stosów połączeń bezprzewodowych; przykłady programów wykorzystujące biblioteki HAL jak i również bezpośrednio rejestrów; przykłady BLE; przykłady BLE Mesh.

4.1.2 Narzędzia i elementy dodatkowe

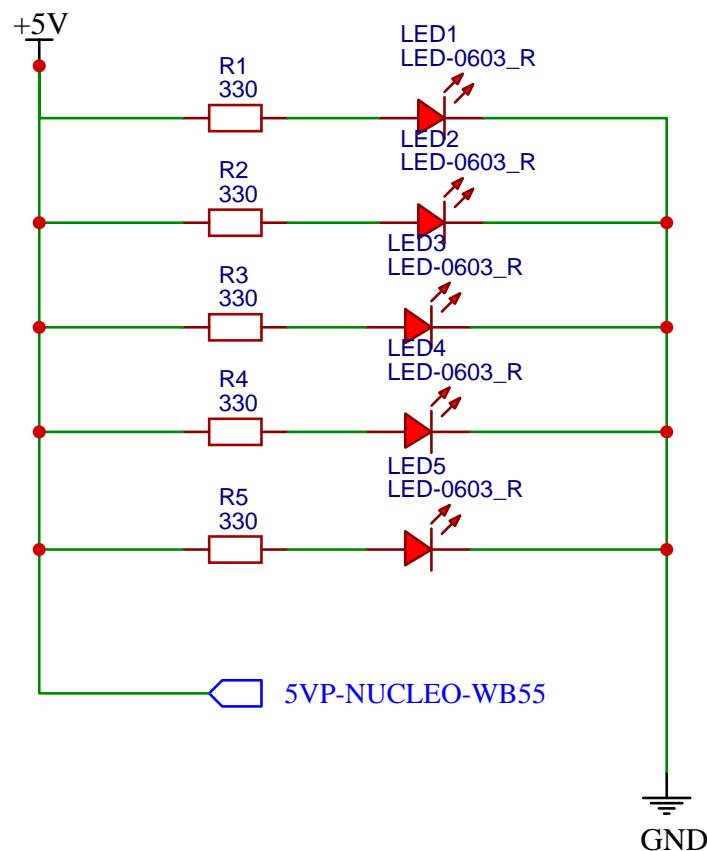
Zasilanie

Próby terenowe wymagają odrębnego, właściwego zasilania. Wybrane zestawy uruchomieniowe wykorzystują ustandaryzowane złącze komunikacyjne USB. Złącze to oprócz komunikacji oferuje zasilanie linią +5V. Przeprowadzając właściwe eksperymenty wykorzystano komercyjnie dostępne banki energii

tzw. powerbanki. Są to akumulatory litowo-jonowe lub litowo-polimerowe z wbudowanym systemem zarządzania baterią³ i właściwymi przetwornicami impulsowymi w celu zapewnienia odpowiedniego napięcia zasilania.

Elektronika wbudowana w magazyny energii automatycznie wyłącza zasilanie w przypadku braku podłączonego urządzenia lub gdy dane urządzenie pobiera marginalnie niskie wartości prądu. Służy to ochronie cennego i delikatnego akumulatora. Jest to cecha korzystna z punktu widzenia konsumenta kupującego powerbank w celu szybkiego ładowania urządzeń elektronicznych. Niestety, zaleta ta staje się wadą w przypadku przeprowadzanych doświadczeń. Podłączając moduł P-NUCLEO-WB55, urządzenie po pewnym okresie działania wyłącza się na skutek uruchomionych zabezpieczeń w źródle energii.

W celu przeciwdziałania temu zjawisku, skonstruowano trywialne urządzenie, które włutowane równolegle w linię zasilania 5V, konsumuje ok. 100mA prądu: $0,5W = 5V \cdot 0,02A \cdot 5[\text{diod}]$. Doświadczalnie stwierdzono, że tak włączony odbiornik energii umożliwia stabilne działanie włączonego do sieci modułu STM32. Schemat zaprezentowano na Rysunku 5



Rysunek 5. Odbiornik energii

³Battery Management System (BMS) – ang. Battery Management System

Obudowa – druk 3D

Kolejnym elementem niezbędnym w celu przeprowadzenia eksperymentów terenowych jest zapewnienie ochrony mechanicznej wybranych zestawów uruchomieniowych. Podczas prób, elementy mogą zostać fizycznie uszkodzone poprzez m.in. wyłamanie fragmentu Printed Circuit Board (PCB) (w tym anteny), uszkodzenie pinów etc.

Wykorzystując techniki projektowania wspomaganego komputerowo⁴, zaprojektowano autorską obudowę. Głównym celem projektowym było zapewnienie minimalnej ochrony mechanicznej, jednocześnie redukując możliwy wpływ materiału na tłumienie sygnału. Elementy interfejsu: przyciski i antena, zostały celowo wyeksponowane. Rzut izometryczny przedstawiony został na Rysunku 6.



Rysunek 6. Model obudowy dla zestawu P-NUCLEO-WB55

Obudowa wykonana została z materiału Polylactic Acid (PLA)⁵ techniką druku 3D Fused Deposition Modeling (FDM)⁶/Fused Filament Fabrication (FFF)⁷. Jednobryłowa konstrukcja podyktowana została chęcią uproszczenia wydruku, kosztem zwiększonej trudności doboru tolerancji dla umieszczanych wewnętrz płytek PCB. Obudowa przewiduje wsuwanie modułu Nucleo do wewnętrz, zapewniając jednoczesne pewne jego mocowanie.

Dobór koloru wydruku również nie był przypadkowy. Przewidując potencjalne lokalizacje przeprowadzanych doświadczeń, wybrano kolor możliwie jaskrawy. Podstawowym założeniem było ułatwienie dostrzeżenia obudowy, a wraz z obudową również zestawu uruchomieniowego, pośród flory leśnej czy terenów zurbanizowanych. Ostateczne wykonanie obudów zaprezentowano na Rysunku 7.

⁴Computer-aided Design (CAD) – ang. *Computer-aided Design*

⁵PLA – ang. *Polylactic acid*, polilaktyd

⁶FDM – ang. *Fused Deposition Modeling*

⁷FFF – *Fused Filament Fabrication*



Rysunek 7. Obudowa zestawu uruchomieniowego Nucleo wykonana w technologii druku 3D – realizacja

4.1.3 Oprogramowanie mikrokontrolera

Oczywistym faktem jest, iż zdefiniowane wcześniej rodzaje eksperymentów wymagają odpowiedniego oprogramowania. Zarówno oprogramowania dla mikrokontrolera jak i komputera PC pozwalającego wykonać i nadzorować wybrane doświadczenia. Podrozdział ten prezentuje kroki podjęte w celu stworzenia odpowiednich programów dla badania poboru zużycia energii i utraty pakietów podczas transmisji danych.

Pierwszym krokiem jest wybór odpowiedniego zestawu narzędzi i frameworka, na którym oparte będą wszelkie dalsze pracy. Rozpatrywane były dwa odrębne stosy technologiczne: Zephyr OS⁸ i natywny dostarczany przez ST. Zephyr OS, będący systemem operacyjnym czasu rzeczywistego (RTOS), jest kompletnym zestawem narzędzi i metodologii wytwarzania oprogramowania dla mikrokontrolerów różnych producentów. Możliwość przeniesienia kodu na inną platformę z minimalną ilością modyfikacji było opcją nadzwyczaj interesującą. Jednakże, pomimo oficjalnego wsparcia mikrokontrolera STM32WB55, nie udało się poprawnie zainstalować dostarczanych przykładów na urządzeniu w sposób umożliwiający ich działanie. Stąd zaniechano dalszych prób wykorzystania tego systemu operacyjnego w projekcie.

Drugim oczywistym kandydatem jest stos oprogramowania dostarczony przez ST. Jest to niejako domyślny wybór. Wybrano więc stos oprogramowania dla mikrokontrolera w wersji v1.13.2, jako najnowszy dostępny w chwili rozpoczęcia prac. Zestaw bibliotek i narzędzi dostarczanych przez producenta został wcześniej przez niego przetestowany dając niejako gwarancję spełnienia minimalnych standardów umożliwiających m.in. certyfikację urządzenia jako wspierającego BLE i BLE Mesh.

Badanie zużycia energii (4.2), wykorzystuje dostarczaną wraz ze stosem przykładową aplikację opracowaną przez producenta mikrokontrolera. Owa aplikacja, BLE HRT, wykorzystuje zdefiniowany przez Bluetooth SIG profil *HeartRate* publikując regularnie, co sekundę odpowiednią wiadomość odbieraną przez odbiornik – telefon/tablet. Do badań konsumpcji wykorzystano niezmodyfikowany przykład.

Doświadczenie PER wymaga bardziej wyrafinowanego podejścia, zdefiniowanego z wymaganiami metodologii w punkcie 4.3.1. Jako podstawę dla kolekcji trzech aplikacji wybrano przykład

⁸<https://www.zephyrproject.org/>

BLE_MeshLightingPRFNode. Przykład ten wprowadza bezpośrednio w pełny stos BLE Mesh umożliwiając jednocześnie wdrożenie pożądanych modeli Mesh jak i przypisanie węzłom konkretnej funkcji w sieci: Proxy, Relay, Friend [7].

Badając właściwości sieci, wprowadza się następujące nazewnictwo węzłów celem łatwiejszej identyfikacji:

- węzeł bliższy (ang. *proximal node*) – węzeł będący połączony bezpośrednio ze stacją akwizycji danych i kontroli przepływu eksperymentu. Dodatkowo, węzeł ten udostępnia tryb *Proxy*⁹
- węzeł środkowy (ang. *intermedial node*) – węzeł działający w trybie przekaźnika (terminologia Mesh: *Relay*). Węzeł ten nie uczestniczy bezpośrednio w badaniach tj. nie są z niego odczytywane jakiekolwiek dane.
- węzeł dalszy (ang. *distal node*) – węzeł zliczający ilość odebranych danych r , udostępniający jednocześnie usługę umożliwiającą odczyt tych wartości.

Posiadając trzy węzły, wymagane jest również dostarczenie trzech zestawów oprogramowania dla każdego z nich, w zależności od pełnionej funkcji.

Węzeł bliższy

Pierwszy węzeł, węzeł bliższy, odpowiada bezpośrednio za przeprowadzany eksperyment. Jego celem jest wysyłanie właściwych pakietów do węzła dalszego w określonych interwałach czasowych. Dodatkowo, służy jako bramka do odbioru liczby mówiącej o całkowitej ilości odebranych pakietów. Trzecim zadaniem tego modułu jest zresetowanie licznika z każdą nową iteracją badania PER.

```
1 // plik: appli_test.c
2 MOBLE_RESULT Test_ApplicationTest_Set05_GenericOnOff(MOBLE_ADDRESS src ,
3 MOBLE_ADDRESS dst)
4 {
5     MOBLE_RESULT result = MOBLE_RESULT_SUCCESS;
6     // [...]
7     meshTest.name = OP_NAME_SET05;
8     meshTest.counter = 0;
9     result = test_set05_generic_initialize(src, dst);
10    if (!result)
11    {
12        run_timer(OP_NAME_SET05, test_generic_subscription, src, dst,
13                  test_set05_generic);
14        result = MOBLE_RESULT_SUCCESS;
15    }
16    else
17    {
18        TRACE_I(TF_VENDOR_M, "%s Could not initialize the test due to error
19         code=%d \r\n", OP_NAME_SET05, result);
```

⁹z ang. pośrednik – tryb umożliwiający dostęp do sieci Mesh urządzeniom poprzez odpowiadającą usługę BLE. Umożliwia on m.in. dostęp do sieci i zarządzania nią z poziomu urządzenia nie posiadającego ani nie wspierającego bezpośrednio stos BLE Mesh

```

17     result = MOBLE_RESULT_FAIL;
18 } // [...]
19
20 return result;
21 }

```

Listing 1. Kod uruchamiający sekwencję badawczą PER

Listing 1 przedstawia kod uruchamiający sekwencję badawczą PER. Linie 4-6 inicjalizują niezbędne parametry m.in. identyfikujące aktualnie uruchomione doświadczenie. Przypisywana jest wartość zero do licznika inkrementującego ilość wysłanych pakietów. Jest to o tyle konieczne, iż pozwala na bezpośrednie porównanie ilości wysłanych komunikatów do ilości odebranych w innym węźle. Linia 8 odpowiada ze wyzerowaniem licznika po stronie węzła dalszego. Wysyłany jest odpowiedni komunikat wykorzystujący model Vendor'a z id APPLI_TEST_CMD. W treści wiadomości wysyłana jest odpowiednia wartość numeryczna interpretowana przez węzeł dalszy polecenie zresetowania węzła: APPLI_TEST_PACKET_ERROR_RATE_COUNTER=0x09U, jak pokazano na listingu 2. W przypadku, w którym zresetowanie licznika się powiedzie, dalsza część funkcji przystępuje do uruchomienia timer'a w linii 11.

```

1 // plik: appli_test.c
2 MOBLE_RESULT test_set05_generic_initialize(MOBLE_ADDRESS src ,
3                                             MOBLE_ADDRESS dst)
4 {
5     MOBLE_RESULT result = MOBLE_RESULT_SUCCESS;
6     AppliBuffer[0] = APPLI_TEST_PACKET_ERROR_RATE_COUNTER;
7
8     result = BLEMesh_SetRemotePublication(VENDORMODEL_STMICRO_ID1, src,
9                                         APPLI_TEST_CMD,
10                                        AppliBuffer, sizeof(AppliBuffer),
11                                         MOBLE_TRUE, MOBLE_TRUE);
12
13     if (result)
14     {
15         TRACE_I(TF_VENDOR_M, "%s Could not initialize the test due to error
16         code=%d \r\n", OP_NAME_SET05, result);
17     }
18     return result;
19 }

```

Listing 2. Kod resetujący wartość licznika w węźle dystalnym

Dodatkową uwagę zwrócić na nazewnictwo samych funkcji. Wskazuję one numer polecenia (Set05, Set06). Numer ten wykorzystywany jest przez *Mesh Serial Gateway* i interpretowany w sposób umożliwiający uruchomienie opisywanej funkcji. Nie jest to ścisłe wymaganie a raczej konwencja, kontrakt wprowadzony przez ST dla danego pliku. Niemniej jednak, pozwala ona w jasny i czytelny sposób zidentyfikować polecenie i odpowiadające polecenie Zbiór poleceń Hayes'a (znany

jako zbiór poleceń AT) (AT) (zbliżone do poleceń AT). Opis działania interfejsu PC-STM32WB55 znajduje się poniżej 4.1.3.

Timer

Timer odpowiada za regularne wykonywanie zdefiniowanej funkcji. Odpowiada on za główny przepływ badania interpretując zebrane parametry doświadczenia i je wykonując. Kod oparto o interfejs programistyczny dostarczony przez ST [14]¹⁰.

```
1 // plik: appli_test.c; Usunięto linie logujące przepływu TRACE_I
2 static void run_timer(char const *setName, MeshTest_t subscriptionType,
3   MOBLE_ADDRESS src, MOBLE_ADDRESS dst, void (*callback)(void) ) {
4   MOBLEUINT16 triggerInterval;
5   MOBLEUINT32 killAfterTimeout;
6
7   triggerInterval = timerTriggerInterval;
8   killAfterTimeout = Totaltest;
9
10  HW_TS_Create(subscriptionType, &(meshTest.timer_subscription_id),
11    hw_ts_Repeated, callback);
12  HW_TS_Create(test_kill_subscription, &(meshTest.
13    timer_kill_subscription_id), hw_ts_SingleShot, kill_subscription);
14
15 }
```

Listing 3. Funkcja aktywująca timer

Listing 3 przedstawia wdrożony kod uruchamiający timer. Najistotniejszym parametrem funkcji jest wskaźnik do funkcji umożliwiający uruchomienie kodu w odpowiedzi na zdarzenie (*callback*). Linie 6-7 przypisują wartości kolejno interwału, z którym periodycznie ma być uruchamiany *callback* oraz wartość po której upłynięciu timer zostanie wyłączone. Obie te zmienne są typu logicznego *tick*. Podawane w nich są wartości ticków procesora aniżeli milisekundy. Przed przypisaniem wartości należy przekonwertować milisekundy do ticków w zależności od konfiguracji zegara RTC mikrokontrolera. Zagadnienie to stanowiło problem na etapie tworzenia procedury badawczej. Ostatecznie zdecydowano się na wprowadzenie pojęcia kalibracji, w której to wyznaczano eksperymentalnie wartości ticków względem oczekiwanej długości okresu czasu. Logika takiego procesu opisana została w punkcie dotyczącym oprogramowania PC 4.1.4.

Linie 9-10, rejestrują wybrane funkcje (poprzez wskaźniki do funkcji) do wewnętrz silnika timera dostarczanego przez ST. Linijki 12-13 kolejno uruchamiają timer. Dzięki temu wprowadzono substytut wielowątkowości wykonując wiele procesów (tutaj: funkcji) pozornie jednocześnie bądź w ustalonej

¹⁰rozdział 4.5 Timer server

sekwencji. Rezultatem działania tego kodu jest periodyczne uruchamianie funkcji *callback* oraz jej zamknięcie po ustalonym czasie.

Opierając się na inżynierii wstępnej, znaleziono współczynnik, przeliczający wartości milisekund do ticków:

```

1 // plik: stm32wbxx_hal_conf.h
2 #define LSE_VALUE ((uint32_t)32768) /*!< Value of the External
   oscillator in Hz*/
3
4 // plik app_common.h
5 #define DIVR( x, y ) (((x)+((y)/2))/(y))
6
7 // plik app_conf.h
8 #define CFG_RTCCLK_DIV (16)
9 #define CFG_TS_TICK_VAL DIVR( (CFG_RTCCLK_DIV * 1000000), LSE_VALUE )
10
11 // plik app_ble.c - przykład uzycia
12 #define INITIAL_ADV_TIMEOUT (60*1000*1000/CFG_TS_TICK_VAL) /*< 60s */

```

Listing 4. Ścieżka inżynierii wstępnej w celu znalezienia współczynnika umożliwiającego konwersję milisekund do ticków

Zgodnie z listingiem 4, ostateczna wartość współczynnika CFG_TS_TICK_VAL wynosi 488.78125. Wyliczone wartości ticków z ich eksperymentalnie wyznaczonymi odpowiednikami zaprezentowano w tabeli 1. Proces ten, zwany dalej kalibracją, oparty został o dodatkową procedurę zaprezentowaną na listingu 5.

```

1 // plik: appli_test.c
2 MOBLE_RESULT Test_ApplicationTest_Set06_CalibrateTimer(MOBLE_ADDRESS src
   ,MOBLE_ADDRESS dst) {
3   MOBLE_RESULT result = MOBLE_RESULT_SUCCESS;
4   if (TestCount != 0 && TestCount > timerTriggerInterval) {
5     meshTest.name = OP_NAME_SET06;
6     meshTest.counter = 0;
7     run_timer(OP_NAME_SET06, test_generic_subscription, src, dst,
8       test_set06_calibrate_timer);
9   }
10  TestNumber = 0; // kill command
11  command = CMD_TYPE_NONE;
12  return result;
}

```

Listing 5. Kalibracja mikrokontrolera – polecenie kalibracyjne

Przedstawione polecenie kalibracji wykorzystuje jedynie węzeł bliższy, który odpowiada za przeprowadzenie wykonanie przepływu doświadczenia. Usługa ta umożliwia wykonanie akcji kalibracyjnej `test_set06_calibrate_timer` polegającej na inkrementacji licznika. Bazując na ostatecznej

wartości licznika przy znanym określonym czasie wykonywania, można wyznaczyć arytmetycznie ostateczną wartość ticków do milisekund. Fakt ten wykorzystywany jest przez aplikację PC i tam zostaje bliżej opisany.

Węzeł środkowy

Kod węzła środkowego uległ najmniejszym zmianom względem pierwowzoru. Dokonano jedynie zmiany w konfiguracji przykładu uruchamiając tryb przekaźnikowy *Relay*, co ukazano na listingu 6.

```
1  /*
2  *   Different features supported by BLE-Mesh. Uncomment according to
3  *   application.
4  *       Low power feature enabled node do not support other features.
5  *       Do not define any other feature if Low Power feature is defined
6  */
7 #define ENABLE_RELAY_FEATURE
8 //##define ENABLE_PROXY_FEATURE
9 //##define ENABLE_FRIEND_FEATURE
10 //##define ENABLE_LOW_POWER_FEATURE
11 //##define ENABLE_PROVISIONER_FEATURE
12 //##define DYNAMIC_PROVISIONER
```

Listing 6. Konfiguracja węzła środkowego

Węzeł dalszy

Oprogramowanie węzła dalszego stworzono komplementarnie do węzła bliższego. Zapewnia ono dwie główne funkcjonalności: zliczanie odebranych pakietów na poziomie Generic OnOff oraz możliwość odczytu i resetu tegoż licznika. Podstawowy interfejs do powyższych funkcjonalności zdefiniowano w pliku *appli_generic_counter.h* - listing 7 wraz z opisem przepływu wiadomości z perspektywy plików i fizycznych węzłów.

Linia 19 definiuje interfejs inicjalizacyjny doświadczenia PER. Zgodnie z wcześniejszym opisem węzła bliższego, ta funkcja aktywowana jest w odpowiedzi na żądanie wyzerowania licznika. Linia 20 reprezentuje funkcję, która pobiera wartość licznika ze struktury zdefinowanej w liniach 15-17, którą zrealizowano już wewnętrz pliku odpowiedzialnego za obsługę modelu *Generic OnOff*: *appli_generic_client.h*.

```
1 #define ENABLE_LED_BLINKING
2
3 /*
4 * The purpose of this header is to provide an API to cover Packet Error
5 * Rate experiment.
6 * The overall experiment is designed as follows:
7 *   * appli_test (proximal node): Run periodically Generic OnOff client
8 *     model (SET-05)
```

```

7 *   * appli_test: Count the number of sent Generic OnOff requests
8 *   * appli_generic_counter (remote node): Count the number of received
9 *   messages
10 *  * appli_vendor (remote node): Publish the number of received messages
11 *  * appli_test: Collect the results. Make sure the nodes are close
12 *  enough to get the results
13 *
14 *
15 typedef struct {
16 MOBLEUINT32 counter;
17 } GenericOnOffCounter_t;
18
19 void generic_onoff_counter_initialize();
20 MOBLEUINT32 generic_onoff_counter();

```

Listing 7. Interfejs eksperymentu PER dla węzła dalszego

Listing 8 prezentuje wycinek kodu zliczającego ilość odebranych komunikatów. `Appli_Generic_OnOff_Set` jest funkcją odpowiadającą na zdarzenie zarejestrowaną w stosie technologicznym BLE Mesh producenta. Uruchamiana jest ona za każdym razem gdy otrzymywane jest polecenie zgodne z modelem zdefiniowanym przez Bluetooth SIG. Najistotniejszą linią jest linia 20, która bezpośrednio odpowiada za logowanie i inkrementację licznika. Istotna jest również instrukcja warunkowa otaczająca opisywane wywołanie funkcji. Uniemożliwia inkrementację licznika na skutek odbioru wiadomości rozgłoszeniowej. Od użytkownika oczekuje się podanie rzeczywistego adresu węzła zarejestrowanego wewnętrz sieci Mesh.

Linie 24-33 wykonują polecenie włączania i wyłączania Light Emitting Diode (dioda emitująca światło) (LED). Funkcjonalność ta otoczona jest dyrektywami sprawdzającymi istnienie zdefinowanej nazwy. Służy to celu włączeniu i wyłączeniu opcji aktywacji diody w celu ograniczenia zużycia energii. Fakt ten wykorzystywany jest w badaniu zużycia energii 4.2.

```

1 /**
2 * @brief Appli_Generic_OnOff_Set: This function is callback for
3 * Application
4 *           when Generic OnOff message is received
5 * @param pGeneric_OnOffParam: Pointer to the parameters received for
6 * message
7 * @param OptionalValid: Flag to inform about the validity of optional
8 * parameters
9 * @param dstPeer: destination send by peer for this node. It can be a
10 *                  unicast or group address
11 * @param elementIndex: index of the element received from peer for this
12 * node which
13 *                  is elementNumber -1
14 * @retval MOBLE_RESULT

```

```
11 */  
12 MOBLE_RESULT Appli_Generic_OnOff_Set(Generic_OnOffStatus_t*  
13     pGeneric_OnOffParam,  
14     MOBLEUINT8 OptionalValid,  
15     MOBLEUINT16 dstPeer,  
16     MOBLEUINT8 elementIndex) {  
17     // [...]  
18     if (AppliOnOffSet[elementIndex].Present_OnOffValue  
19         == AppliOnOffSet[elementIndex].TargetValue)  
20     {  
21         if (dstPeer != 0xc000) {  
22             increment_generic_onoff_counter_packet_error_rate_experiment(  
23                 AppliOnOffSet[elementIndex].Present_OnOffValue, dstPeer);  
24         }  
25 #ifdef ENABLE_LED_BLINKING  
26         if (AppliOnOffSet[elementIndex].Present_OnOffValue > 0)  
27         {  
28             BSP_LED_On(LED_BLUE);  
29         }  
30         else  
31         {  
32             BSP_LED_Off(LED_BLUE);  
33         }  
34 #endif /* ENABLE_LED_BLINKING */  
35     }  
36     // [...]
```

Listing 8. Inkrementacja licznika realizowana jest poprzez dostosowanie funkcji odpowiadającej na zdarzenie

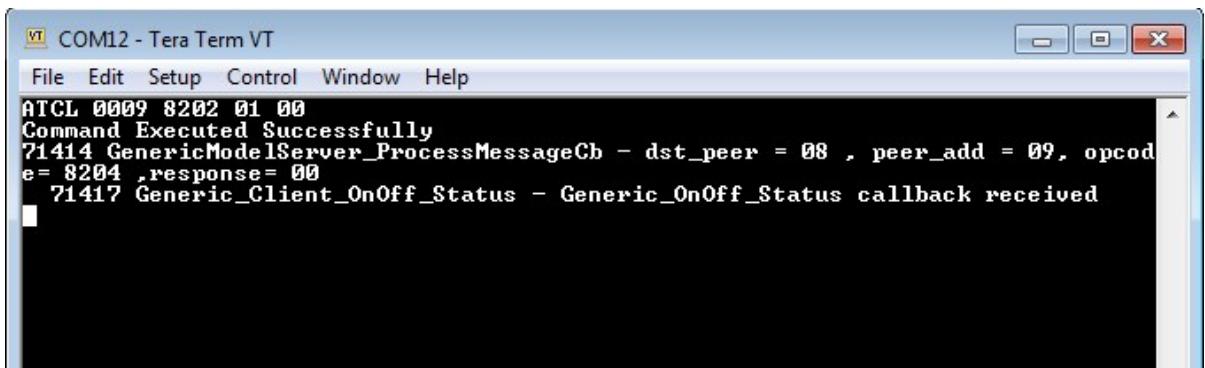
Mesh Serial Gateway

Komunikację na linii węzła bliższego zrealizowano z pomocą autorskiego rozwiązania ST: Mesh Serial Gateway [7]. Rozwiązanie to wykorzystuje komunikację szeregową Universal asynchronous receiver-transmitter (UART) do wysyłania i odbierania komunikatów sterujących. Zgodnie z dokumentacją, polecenie takie może przyjąć postać jak przedstawioną na rysunku 8, wysyłając do węzła 0009 polecenie aktywujące funkcję `Appli_Generic_OnOff_Set` i ustawiając wartość na 1. Innymi słowy, dioda zaświeci się, o ile wcześniej była nieaktywna.

W analogiczny sposób steruje się poleceniami niezbędnymi do eksperymentu, jak to ukazano na listingu 9.

```
1 ATAP SET-nn iiiiTTTTTTT src dst  
2  
3 przykład:  
4 ATAP SET-05 0a6400009c40 03 05
```

Listing 9. Przykładowe polecenie Mesh Serial Gateway



Rysunek 8. Wykorzystanie interfejsu szeregowego do wysyłania poleceń. Źródło: [7]

ATAP - polecenie testowe

SET-nn - polecenie typu SET dla funkcji numeru nn, np. SET-05

iiii - interwał, częstotliwość zapytań wyrażony w tickach; Wartość 16-bitowa

TTTTTTTT - czas po którym należy przerwać badanie wybrane w tickach; Wartość 32-bitowa

src - adres źródłowego węzła (najczęściej węzła bliższego)

dst - adres węzła docelowego (najczęściej węzła dalszego)

4.1.4 Oprogramowanie PC

Głównym zadaniem oprogramowania PC jest udostępnienie prostego interfejsu użytkownika umożliwiającego swobodne przeprowadzenie badań, w szczególności PER. Stworzenie takiego programu umożliwia nie tylko szybsze wykonanie doświadczeń, co również redukuje błędy możliwe do popełnienia podczas ręcznego wprowadzania komend AT.

Aplikacja realizuje trzy główne cele:

- Ustanawia połączenie z mikrokontrolerem przy użyciu portu szeregowego
- Umożliwia kalibrację podstawy mikrokontrolera czasu względem ticknięć
- Umożliwia wykonanie poleceń uruchamiających cykl badań PER i odbiór wartości zliczeń z węzła dystalnego

Oprogramowanie wykonano z wykorzystaniem framework'a Qt¹¹. Umożliwia on tworzenie kompletnych, wieloplatformowych aplikacji graficznych. Qt nie ogranicza się jedynie do zapewnienia bibliotek graficznych. Framework zapewnia dostęp z poziomu API do urządzeń i peryferiów takich jak drukarki, gamepady, gniazda TCP/IP, stoseu Bluetooth (w tym BLE) jak i również, co jest istotne z punktu widzenia niniejszej pracy, portu szeregowego.

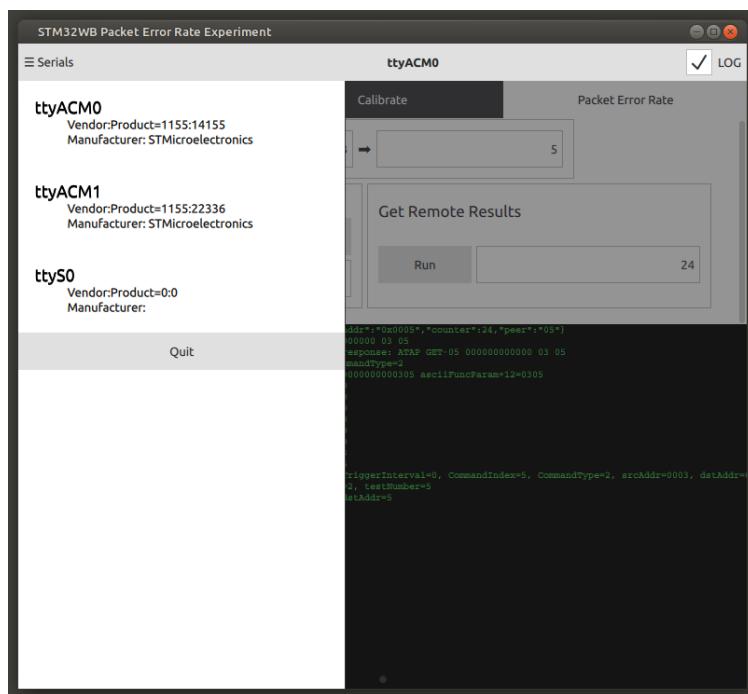
Qt zapewnia kilka możliwości tworzenia GUI wykorzystując tylko kod w języku C++ lub rozdzielając część widzialną od części biznesowej z pomocą generatora interfejsu Qt Designer

¹¹<https://www.qt.io/>

produkując odpowiedni plik Extensible Markup Language (XML) bądź *QML*. *QML* jest językiem deklaratywnym opisujący pożądany efekt graficzny. Do silnika tego narzędzia należy interpretacja i wygenerowanie właściwego efektu. Język ten rozszerza możliwości klasycznego wydania frameworka umożliwiając tworzenie dynamicznych, nowoczesnych interfejsów użytkownika działających niezależnie od platformy docelowej.

Port szeregowy

Aplikację oparto o połączenie części biznesowej tworzonej w C++ oraz interfejsie użytkownika stworzonego w QML. Pierwszym, niezbędnym elementem jest zapewnienie stabilnej komunikacji z użyciem portu szeregowego¹². Aplikacja wpierw wyszukuje listę portów w danym systemie operacyjnym i prezentuje ją w przystępny dla użytkownika sposób – Rysunek 9.



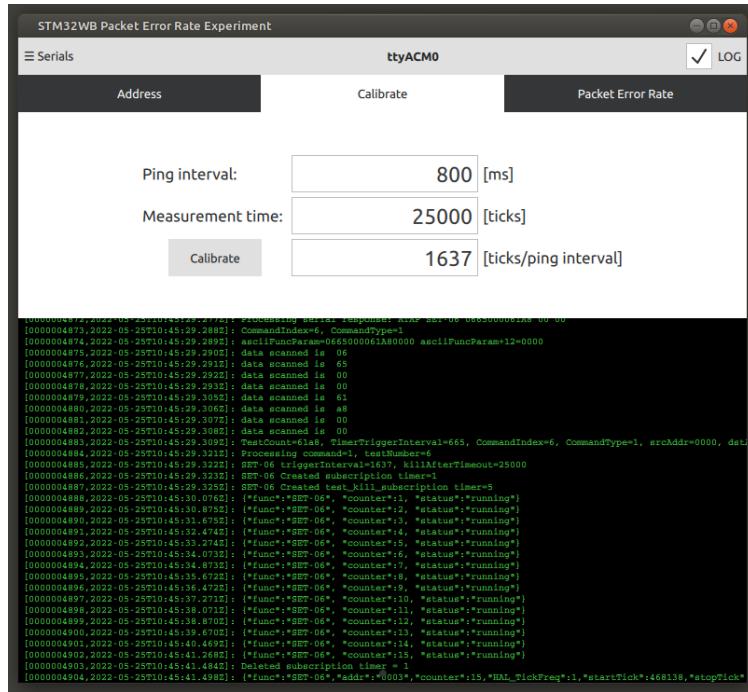
Rysunek 9. Selektor portów szeregowych w aplikacji służącej eksperymentowi PER.

Kalibracja

Kolejnym elementem, wymaganym przez chęć przeprowadzenia badania, jest zapewnienie właściwej podstawy czasu. Wykorzystywany timer (jak został opisany w punkcie 4.1.3) wymaga sprecyzowania wartości podanej w tick'ach. Zauważając potencjalny problem z matematycznym wyznaczeniem odpowiedniego mapowania z milisekund do ticków zdecydowano się na stworzenie empirycznego sposobu na wyznaczenie tej wartości.

¹²<https://doc.qt.io/qt-5/qserialport.html>

Kalibracja urządzenia polega na odnalezieniu wartości ticków przypadającą na pożądany okres wyrażony w milisekundach. Usługa obsługująca taką operację przedstawiona została na listing'u 5. Zadaniem aplikacji PC jest jej wielokrotne wykorzystanie celem wyznaczenia ostatecznej wartości.



Rysunek 10. Selektor portów szeregowych w aplikacji służącej eksperymentowi PER.

Algorytm działania jest następujący. Dla oczekiwanej wartości wyrażonej w milisekundach, wykonuje się pierwsze polecenie kalibracyjne zakładając fałszywie, iż milisekundy równe są ilości ticków. Aplikacja, nasłuchuje przychodzące komunikaty w formacie JSON poszukując słowa kluczowego „status”. Do każdego zliczonego komunikatu, aplikacja przechowuje dodatkowo znacznik czasu. Takie wiadomości akumulowane są w pamięci do momentu otrzymania innego słowa kluczowego: „not_running”. W tym momencie, aplikacja wylicza różnicę między przyległymi znacznikami czasowymi (efektywnie je różniczkując w liniach 4-9 listingu 10). Następnie, wyliczana jest średnia różnica pomiędzy poszczególnymi okresami, w których otrzymano odpowiedź z mikrokontrolera – linia 15.

```

1 // plik calibratecommand.cpp
2 std::pair<uint16_t, double> CalibrateCommand::computeNewInterval() {
3     QVector<qint64> timestampDiffs;
4     std::transform(std::cbegin(timestamps), std::cend(timestamps),
5                  std::back_inserter(timestampDiffs),
6                  [] (const QDateTime &timestamp) -> long { return timestamp.
7                      toMSecsSinceEpoch(); });
8
9     std::adjacent_difference(std::begin(timestampDiffs), std::end(
10        timestampDiffs), std::begin(timestampDiffs));

```

```
9    timestampDiffs.removeFirst();
10
11    qint64 sum = 0;
12    for (qint64 elem : qAsConst(timestampDiffs)) {
13        sum += elem;
14    }
15    double meanTimestampDiff = sum / static_cast<double>(timestampDiffs.
16        size());
16    double accuracy = abs((meanTimestampDiff - expectedIntervalMs) /
17        static_cast<double>(expectedIntervalMs));
17    double interval = currentIntervalTicks + currentIntervalTicks *
18        accuracy; // this isn't stable if we overshoot the correct value...
19
20    return std::make_pair(interval, abs(accuracy));
}
```

Listing 10. Algorytm wyznaczania kolejnej wartości dla funkcjonalności kalibracji

Kolejnym krokiem jest wyliczenie procentowej różnicy pomiędzy obecnie wyliczoną wartością a oczekiwanaą wartością wyrażoną w milisekundach – linia 16. Ostatecznie, wylicza się nową, estymowaną wartość kolejnej iteracji zgodnie z zależnością liniową podaną w linii 17.

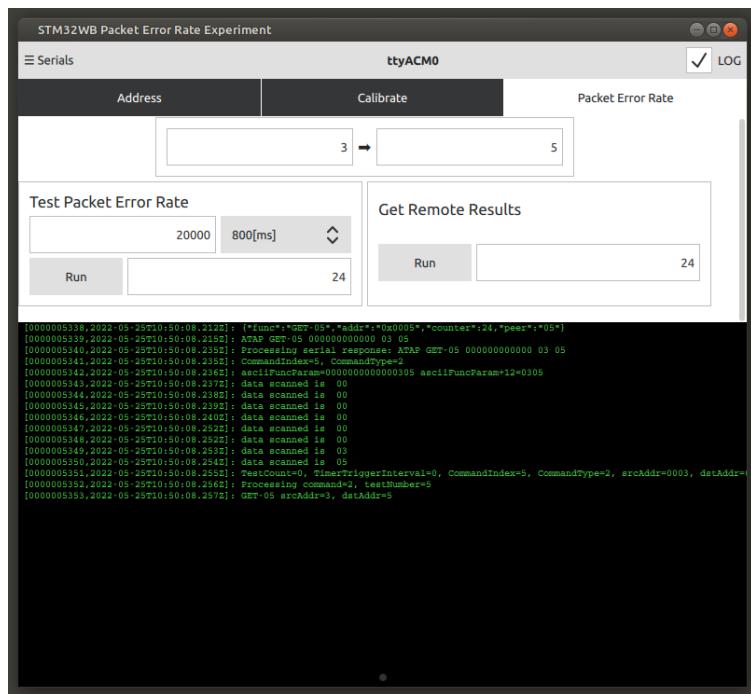
Kalibracja zatrzyma się automatycznie, gdy osiągnięta zostanie zbieżność na poziomie poniżej jednego procenta różnicy. Algorytm jest wrażliwy na „przestrzelanie” kalibrowanej wartości. Jego jakość jest jednak na tyle wystarczająca, iż ostatecznie umożliwił wyznaczenie wartości, które są zbieżne z obliczeniami teoretycznymi, co pokazuje tabela 1. Możliwym usprawnieniem byłoby zastosowanie metody wyznaczania parametrów kalibracyjnych w oparciu o analogię do metody bisekcji.

Ping [ms]	Skalibrowane [tick]	Obliczone [tick]	Różnica [%]
100	204	204.59	0.29
500	1022	1022.95	0.09
800	1637	1636.72	0.02
1300	2661	2659.68	0.05
2100	4299	4296.40	0.06

Tablica 1. Zebrane empirycznie współczynniki kalibracyjne w porównaniu z wartościami obliczonymi

Badanie PER i odczyt wyników

Główym zadaniem aplikacji jest oczywiście przeprowadzenie doświadczenia PER. W tym celu stworzono wygodny interfejs do m.in. funkcji opisywanej w punkcie 4.1.3. Aplikacja wymaga od użytkownika podania adresu (od lewej): węzła bliższego oraz węzła dalszego. W rzędzie poniżej znajdują kontenery z dwoma funkcjami w danej zakładce: właściwy test PER oraz odbiór wartości z węzła zewnętrznego – Rysunek 11.



Rysunek 11. Właściwy interfejs użytkownika do przeprowadzenia doświadczenia PER

Eksperymentator, w celu przeprowadzenia doświadczenia, zobowiązany jest wyznaczyć okres, po którym zakończy się badanie w milisekundach. Czas badania powinien być dostatecznie długi, by błędy inicjalizacji, czy konwersji milisekund do ticków nie miały znacznego wpływu na otrzymany rezultat. Drugi parametr to interwał odpytywania. Aplikacja przechowuje wyniki kalibracyjne¹³, by je wykorzystać właśnie w tym miejscu. Przyśpiesza to procedurę badawczą, a przede wszystkim zmniejsza ryzyko błędu. Ostatnim elementem jest przycisk „Run”, który uruchamia procedurę badawczą. Ekran aplikacji zostanie zablokowany na czas badania.

Analogicznie należy postąpić w przypadku odczytywania licznika z węzła dalszego. Użytkownik zobowiązany jest kliknąć przycisk „Run” w kontenerze „Get Remote Results”. Po krótkiej chwili, odpowiadające pole powinno zostać zaktualizowane o oczekiwana wartość.

4.2 Badanie zużycia energii

Celem niniejszego podrozdziału jest omówienie empirycznej weryfikacji zużycia energii przez wybrany zestaw uruchomieniowy BLE.

Zaprezentowanie zostanie metodologia pomiaru oraz sposób połączenia układu pomiarowego. Omówione zostaną parametry próbkowania i długość trwania pojedynczej sesji badawczej. Ostatecz-

¹³wykorzystując możliwości *QSettings* dane przechowywane są trwale na dysku. W zależności od systemu albo w rejestrze (Windows), albo w pliku */.config/Warsaw University of Technology/STM32WB Packet Error Rate Experiment.conf* (*nix)

nie, przedstawia się wzór i sposób przeprowadzonych obliczeń, które docelowo zapewniają oczekiwane wyniki.

Ostatnim, aczkolwiek najistotniejszym elementem poruszonym przez podrozdział, jest prezentacja wyników. Pogrupowane są one na dwie kategorie: BLE Heart Rate i BLE Mesh. Każda z nich przedstawia zebrane empirycznie dane, wskazując na właściwe tryby działania i całkowite zużycie energii.

4.2.1 Metodologia badania

Podstawą badania zużycia energii jest pomiar wartości pobieranego prądu przez zestaw uruchomieniowy P-NUCLEO-WB55 w czasie. Posiadając dane wymienione dane, na podstawie oczywistych zależności fizycznych, wyznacza się parametry jak całkowita wykorzystana energia podczas badania, przedstawiona w przystępnej formie parametru mocy.

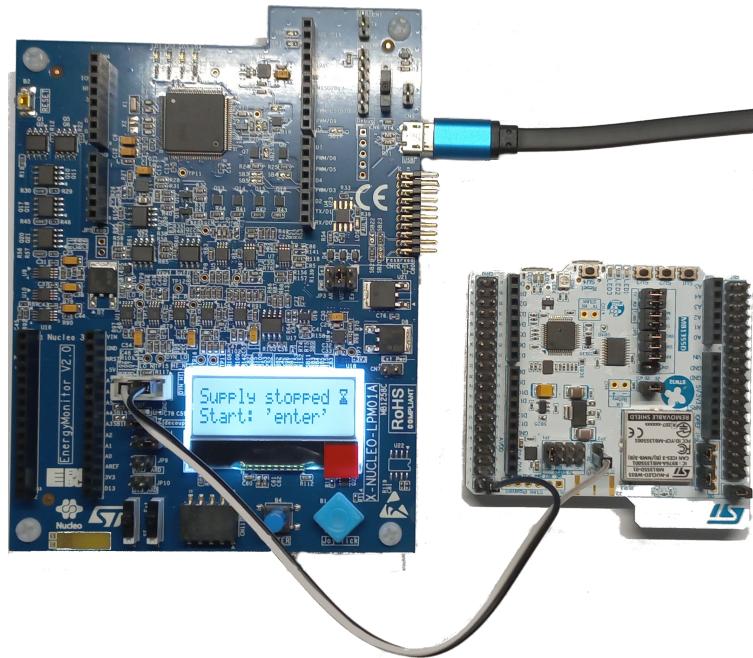
Pomiary wartości chwilowego poboru prądu oparto o moduł opisany w punkcie 4.1.1. Źródłem energii dla płytki pomiarowej jest komputer klasy PC, a precyzyjniej udostępniany port USB. Wykorzystując również ten protokół następuje transmisja danych poprzez komunikację szeregową UART, umożliwiając akwizycję danych.

Połączenie z zestawem uruchomieniowym oparto o konektor CN14 udostępniający piny: PIN1 - masa GND, PIN3 - VCC 3.3V [17]. Wyprowadzone przewody połączono następnie z P-NUCLEO-WB55 poprzez konektor JP2, wcześniej pozbawiony zawleczki. Jest to metoda rekomendowana dla pomiaru prądu opisana w dokumentacji zestawu [13]¹⁴. Porty kompatybilne z Arduino/ST Morpho mogłyby stanowić równorzędny sposóbłączenia zestawu uruchomieniowego do obwodu płytki pomiarowej. Autorska analiza dokumentacji sugeruje jednak, iż opcja ta nie jest wspierana. Wymagane byłoby lutowanie dodatkowego połączenia w (punkcie SB27), by móc wykorzystać zasilanie napięciem 3.3V.

Celem akwizycji danych, wybrano dostarczane przez producenta oprogramowanie *STM32CubeMonitor-Power*. Przykładowy zrzut ekranu zaprezentowany jest na rysunku 13. Pojedyncza sesja pomiarowa trwająca 100 sekund zapewnia niezbędne dane. Są one następnie odpowiednio przetwarzane poprzez odcięcie wartości zebranych w ostatnich sekundach. Związane to było ze sposobem działania mikrokontrolera, który po 60 sekundach domyślnie przechodzi w stan zwiększonej oszczędności energii (Low Power Advertising). Parametr ten można zmienić wg własnych upodobań, aczkolwiek zdecydowano się na domyślne nastawy, tak by umożliwić możliwie proste, ponowne przeprowadzenie doświadczenia wykorzystując przykład BLE HRT dostarczony przez ST. Stąd, by rozróżnić różne tryby pracy BLE, zdecydowano o odcięciu połowy okresu pomiarowego, zapewniając jednolite wartości względem trybów zużycia energii w 50 sekundowym oknie.

Tak zebrane dane następnie przetworzono uzyskując wartości energii i średniej mocy użytej przez układ. Wykorzystuje się w celu oczywistą zależność fizyczną zaprezentowaną wzorem 2 [6].

¹⁴Rozdział 7.12 Current measurement



Rysunek 12. Podłączony zestaw pomiarowy

$$E_{\text{całkowita}} = U \cdot \int_{t=0[s]}^{t=50[s]} di \, dt \quad (2)$$

$$P = \frac{E_{\text{całkowita}}}{t} \quad (3)$$

gdzie:

$E_{\text{całkowita}}[J]$ - wykorzystana energia podczas 50s sekundowej sesji rejestracji danych

$P[W]$ - moczużyta podczas 50s sekundowej sesji rejestracji danych

$U[V]$ - napięcie zasilania mikrokontrolera - 3.3V - stała

$di[A]$ - prąd w danej chwili

$dt[s]$ - podstawa czasowa całkowania, 0.01s/interwał (100Hz) - stała

Zebrane dane w postaci wartości chwilowych pobieranego przez mikrokontroler prądu względem czasu przetworzono z użyciem metod numerycznych. W celu wyliczenia łącznej wykorzystanej energii, a tym samym mocy, stosuje się zależność 2. Uwzględniając fakt działania w domenie dyskretnej, wykorzystano kompozytową metodę całkowania Simpsona [5]. Podstawą dla całkowania są wartości zebrane w równoodległych odstępach. Dokonując akwizycji danych, dobrano częstotliwość próbkowania jako 100Hz. Każda kolejna wartość charakteryzuje się więc 10 milisekundową różnicą w podstawie czasu. Uwzględniając ten fakt, wyliczenie wartości wykorzystanej energii jak i mocy staje się trywialne.



Rysunek 13. Przykładowa sesja pomiarowa dla BLE Mesh - sieć w trybie nasłuchującym

4.2.2 BT Low Energy - Usługa Heart Rate

Pierwszą usługą BLE badaną pod kątem zużycia energii jest Heart Rate¹⁵. Wybór ten podyktowany jest częścią przyszłego zestawienia wartości z rozwiązaniami innych producentów. Usługa zdefiniowana przez Bluetooth SIG stanowi więc wspólny międzyplatformowy język. Porównanie konkurencyjnych rozwiązań względem ST stanowi potencjalny kierunek dalszych rozważań.

Badania przeprowadzono dla trzech różnych trybów działania aplikacji BLE HRT:

- Usługa połączona (Connected)
- Usługa w trybie szybkiego ogłaszanego (Fast Advertising)
- Usługa w trybie niskomocowego ogłaszanego (Low Power Advertising)

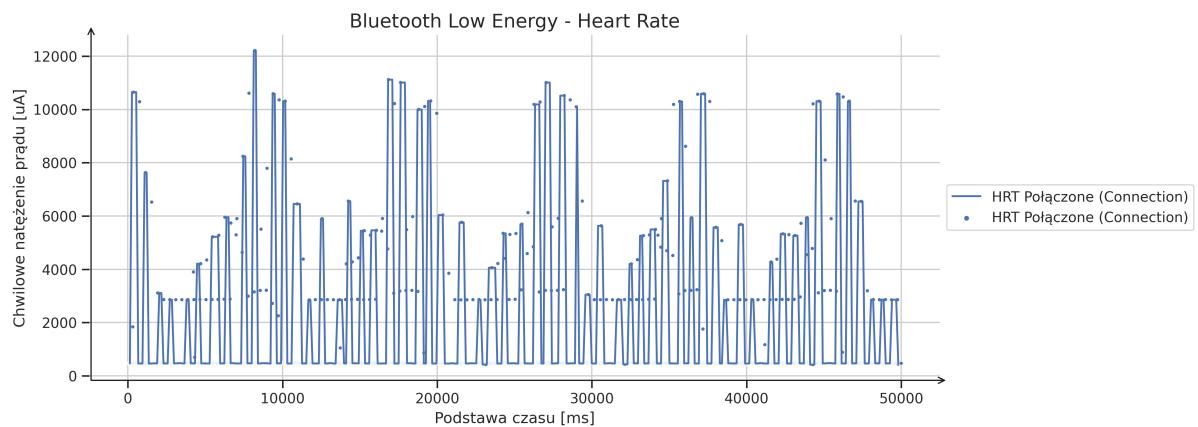
Emitowana moc ustalona została na $-0.15dBm$. Jest to wartość wspólna dla każdego pomiaru dotyczącego BLE HRT.

Rysunek 14 przedstawia charakterystykę poboru prądu w czasie po ustanowieniu połączenia urządzeniem klienckim. Mikrokontroler publikuje losowe dane z częstotliwością $1Hz$. W konsekwencji radio zestawu uruchomieniowego uruchamiane jest co najmniej raz na sekundę.

Analizując wykres dostrzega się regularne, 10-sekundowe interwały w których pobierany prąd dąży do swojego maksimum wynoszącego ok. $12mA$. Jest to prawdopodobnie związane z ustawionymi parametrami transmisji danych w mikrokontrolerze. Niniejsza praca nie podejmuje jednak próby wyjaśnienia rzeczywistej przyczyny tego zjawiska. Minimalny zarejestrowany pobór prądu ustalony jest na ok. $422\mu A$, czyli o dwa rzędy mniejsze wartości szczytowej.

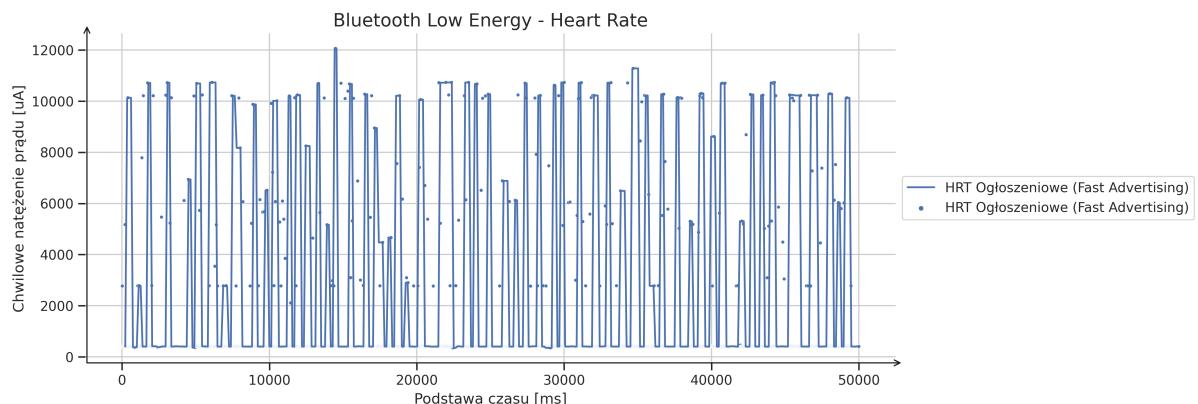
Rysunek 15 przedstawia charakterystykę poboru prądu po czasie dla szybkiego trybu ogłoszeniowego. Zgodnie za kodem źródłowym, interwały w których rozgłoszane są komunikaty powinny się zawierać w częstotliwości od 80 do 100ms. Wywołuje to konieczność uruchamiania radia co najmniej 10 razy na sekundę. Ma to swoje odzwierciedlenie na wykresie. Charakterystyka jest bardziej *burzliwa* w porównaniu z wykresem 14.

¹⁵z ang. rytm serca



Rysunek 14. Charakterystyka czasowa poboru prądu dla BLE i usługi Heart Rate - Usługa Połączona

Zauważa się większą częstotliwość szczytów osiągających wartości pomiędzy 10 – 12mA. Jest to w oczywisty sposób powiązanie ze sposobem działania trybu *Fast Advertising*. Minimalna wartość zarejestrowana wynosi jednak 350uA, co ponownie stanowi różnicę dwóch rzędów wielkości.



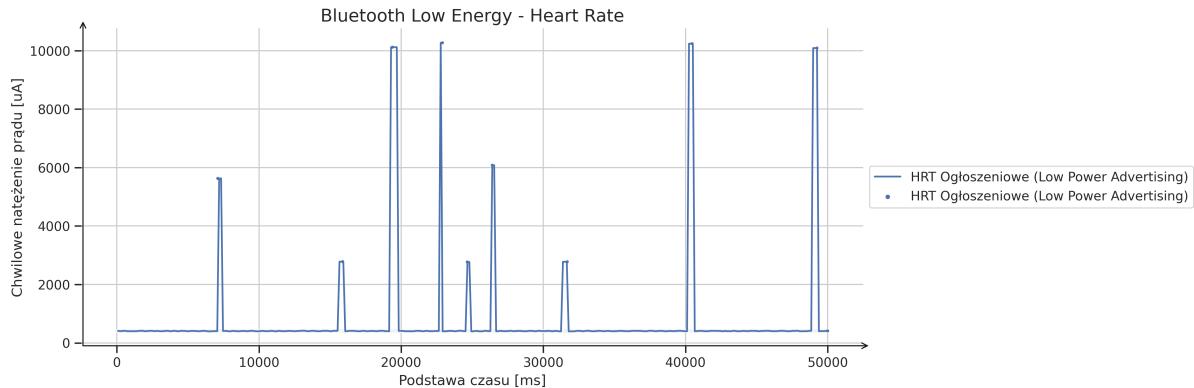
Rysunek 15. Charakterystyka czasowa poboru prądu dla BLE i usługi Heart Rate - Tryb szybkiego ogłoszenia

Ostatnim wykresem prezentującym charakterystykę poboru prądu w czasie jest rysunek 16. Przedstawia on tryb działania niskomocowego ogłoszenia. Różni się on względem trybu *Fast Advertising* częstotliwością z jaką wysyłane są pakiety ogłoszeniowe. W tym przypadku, oprogramowanie wysyła wiadomości z interwałem znajdującym się w przedziale od jednej do dwóch i pół sekundy (parametry: *CFG_LP_CONN_ADV_INTERVAL_MIN* i *CFG_LP_CONN_ADV_INTERVAL_MAX*). Oba wymienione tryby są autorskim rozwiązaniem ST. Przegląd literatury nie wskazuje, jakoby takie zachowanie wymagane jest przez sam standard.

Rysunek 16 wskazuje na podobne zależności. Obserwuje się periodyczne, aczkolwiek stosunkowo rzadkie szczyty w poborze energii rozumianej jako przepływ prądu elektrycznego. Wpisują się jednak w wyżej wymienione parametry transmisji. Brak całkowitego dopasowania względem oczekiwanych

Rozdział 4. Część doświadczalna

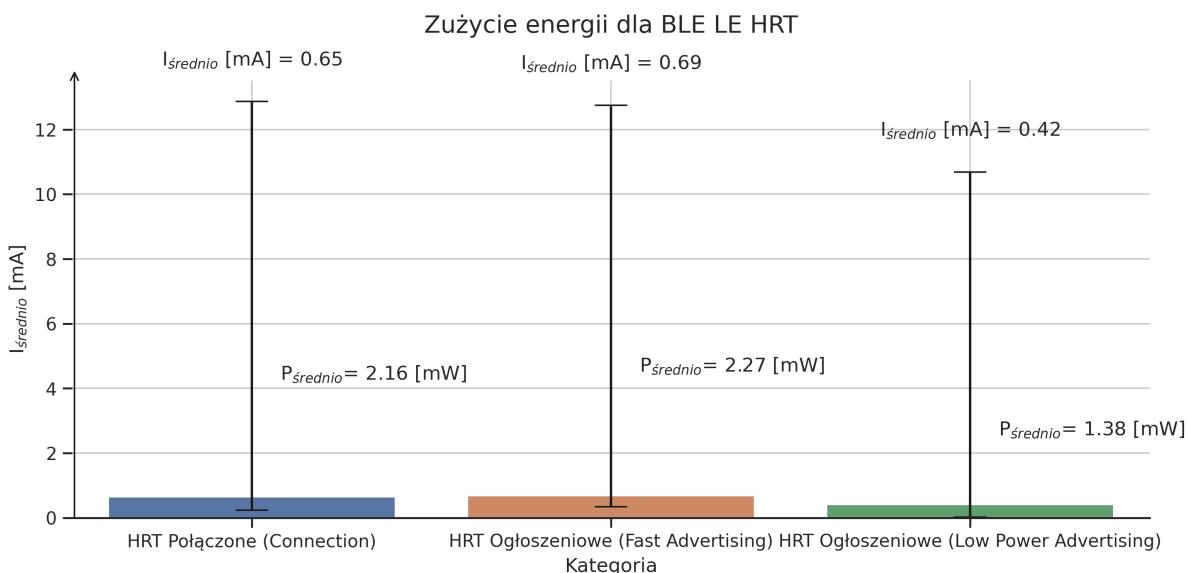
parametrów może być spowodowany niedostateczną częstotliwością próbkowania. Z pewnością jest to zagadnienie, które warto przeanalizować w kolejnych iteracjach prób doświadczalnych.



Rysunek 16. Charakterystyka czasowa poboru prądu dla BLE i usługi Heart Rate - Tryb niskomocowego ogłoszania

Ostatecznie, zestawia się wszystkie zebrane jak dotąd wartości pomiarowe na rysunku 17. Wykres słupkowy przedstawia średni pobór prądu wyrażony miliamperach. Porównując, zauważa się znaczącą, blisko dwukrotną różnicę, co do zużytnej mocy, pomiędzy trybem ogłoszeniowym niskomocowym a pozostałymi kategoriami.

Porównując wymienione tryby ogłoszeniowe, należy zwrócić uwagę na konfigurację częstotliwości, z którą wysyłane są pakiety danych. Częstotliwość, z którą aktywowane jest radio, może być nawet 30 krotnie wyższa dla przypadku szybkiego ogłoszania. Uwidocznia się również różnica w maksymalnie osiągniętym szczycie poboru mocy, wynosząca ok. 2mA.



Rysunek 17. Zestawienie zużycia prądu dla usługi Heart Rate w zależności od trybu działania

Zużycie energii w zestawieniu trybu ogłoszeniowego szybkiego i ustanowionego już połączenia jest zbliżone. Według zadanego nastawień, *Fast Advertising* aktywuje radio około 10 razy częściej aniżeli po ustanowieniu łączności z aplikacją kliencką. Natomiast, przypadek połączonej już usługi przenosi dodatkowe informacje takie jak puls czy estymowana spalona energia w wyników procesów metabolicznych. Być może, zużycie energii również zależy od ilości przenoszonych danych, a co za tym idzie, wydłużenia czasu działania radia niezbędnego do wysłania takiej wiadomości. Jest to pytanie otwarte.

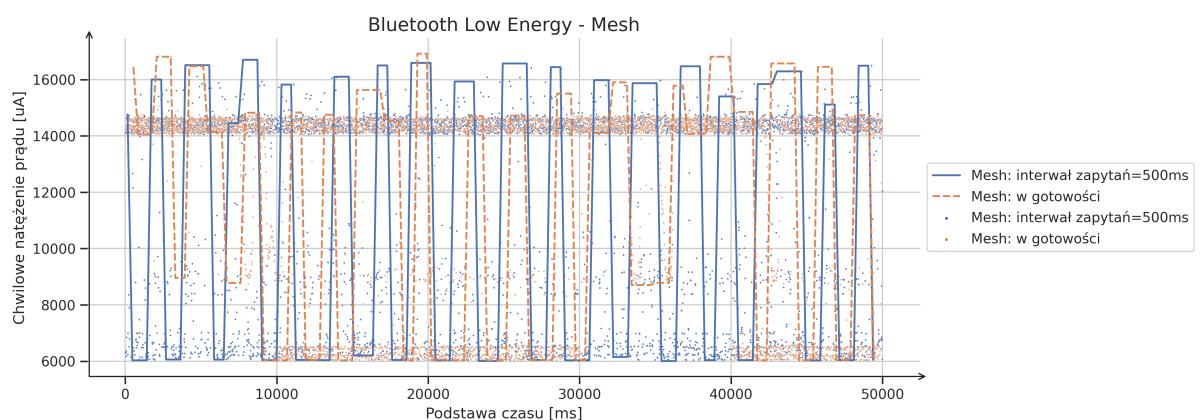
4.2.3 BLE Mesh - Model Generic OnOff

Zużycie energii dla BLE Mesh przeprowadzono dla dwóch wariantów:

- sieć w trybie gotowości (*standby*)
- sieć w której wysyłany jest komunikat z modelu *Generic OnOff*

Poprzez sieć w trybie gotowości należy rozumieć, iż wszystkie zarejestrowane¹⁶ węzły są uruchomione. Zgodnie ze standardem BLE Mesh, każdy z takich węzłów musi mieć zaimplementowane dwa modele: *Configuration Server* i *Health Server* [18]. Odpowiadają one za niezbędną konfigurację jak i również nadzorowanie stanu zdrowia sieci, poprzez wysyłanie komunikatów *keep-alive*¹⁷.

Rozpatrywany przypadek *Generic OnOff* wykorzystuje tenże model do wysyłania komunikatów wewnętrz sieci. Jeden z węzłów wysyła pakiet danych a drugi, odbiorczy, przetwarza wykonując operację zdefiniowaną przez model akcję. W przypadku poniższych badań operacją tą jest kolejne zliczanie otrzymanych pakietów. W tym celu wykorzystano oprogramowanie opracowane na potrzeby kolejnego opisywanego eksperymentu *Packet Error Rate*. Przez cały okres akwizycji wysyłano komunikaty z częstotliwością 2Hz. Zasadne jest wspomnieć, iż na czas trwania doświadczenia wszelkie diody emitujące światło zostały wyłączone celem eliminacji błędów systemowych wpływających na ostateczne zużycie energii przez urządzenie.

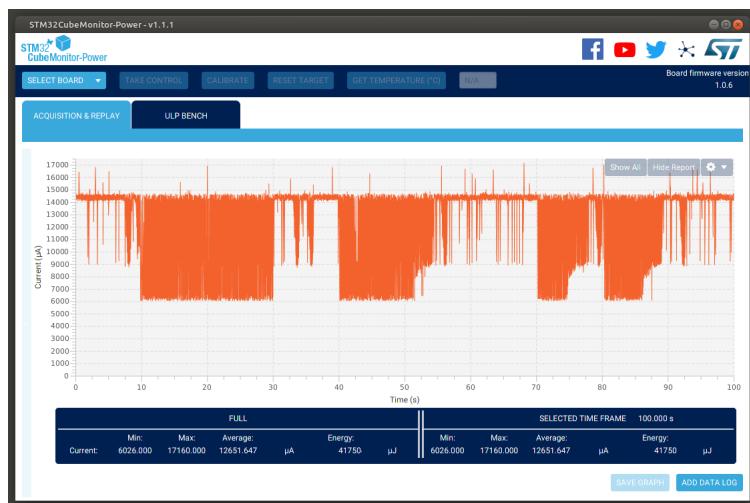


Rysunek 18. Charakterystyka czasowa poboru prądu dla BLE Mesh i modelu Generic OnOff

¹⁶proces w anglojęzycznej literaturze i dokumentacji znany jest jako *provisioning*

¹⁷standard definiuje takie komunikaty jako *heartbeats*[18][2]

Rysunek 18 przedstawia charakterystyki prądu dla wymienionych wcześniej badanych wariantów. Wartości są aproksymowane celem wskazania ogólnej zmiennej tendencji poboru prądu elektrycznego. Jej zmienność reprezentuje wykres punktowy, zlewający się w trzy rozpoznawalne klastry oscylujące wokół wartości 14mA, 9mA i 6mA. Rzeczywista charakterystyka wygenerowana przez aplikację *STM32CubeMonitor-Power* zaprezentowana jest na rysunku 19.

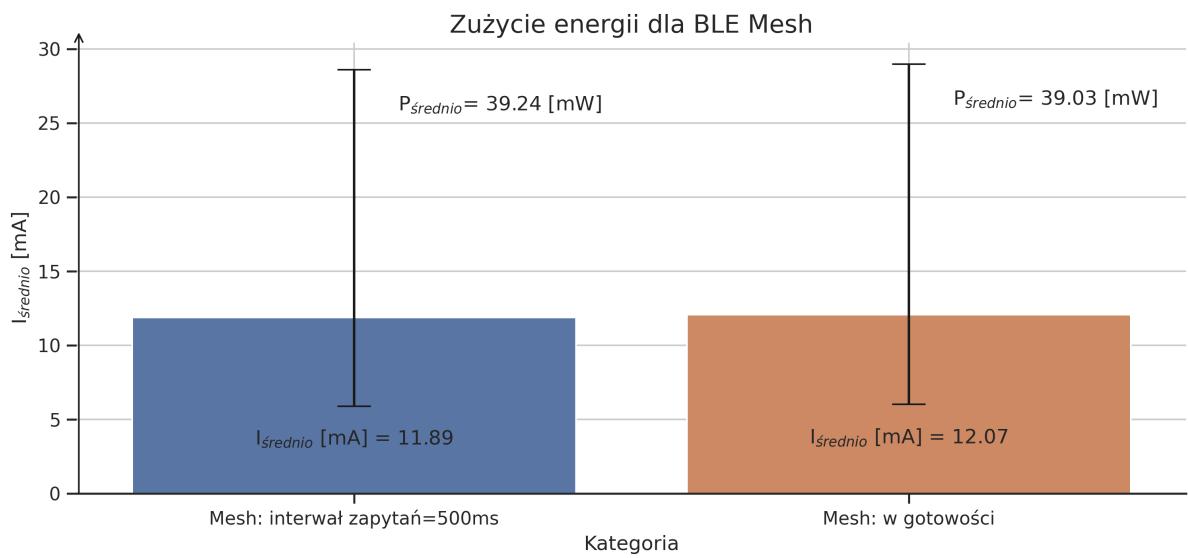


Rysunek 19. Rzeczywista charakterystyka poboru energii dla BLE Mesh działającego w trybie gotowości

Rysunek 19 wygenerowany przez aplikację produkcji ST odpowiada chmurze punktów z rysunku 18 dla wariantu regularnych zapytań co 500ms. Analogiczny wykres dla trybu nasłuchującego wygenerowany przez *STM32CubeMonitor-Power* reprezentowany jest na rysunku 13. Swoiste zagęszczenie punktów pomiarowych i częste zmiany amplitudy mogą wskazywać na złożoność protokołu Mesh, który wymaga częstych operacji w tym radiowych. Operacje te bezpośrednio oddziałują na zużycie energii, co w oczywisty sposób widoczne jest na wykresach.

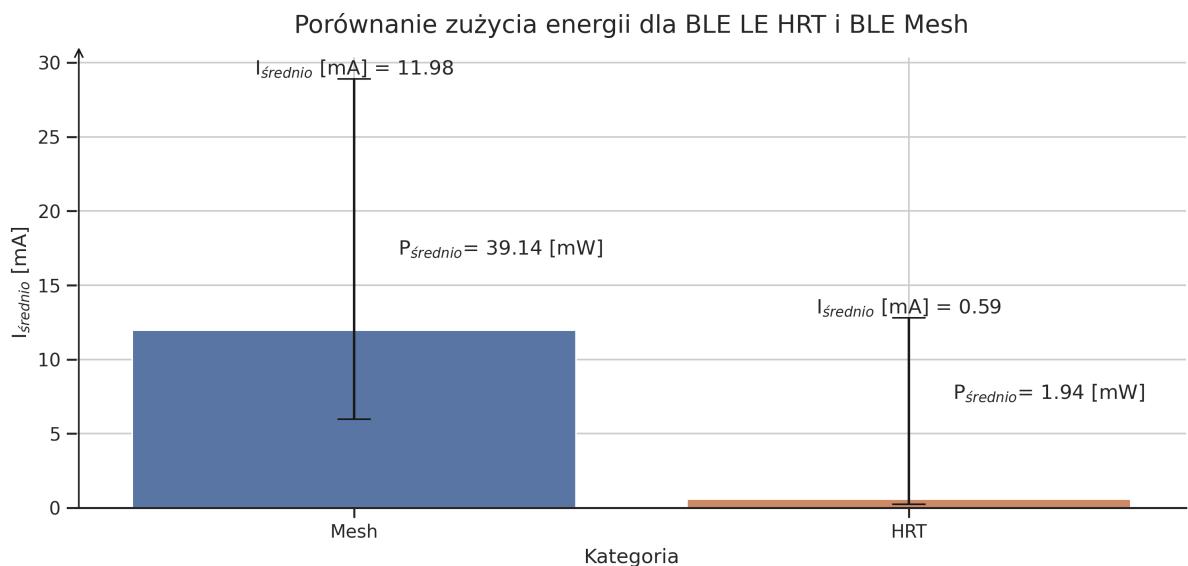
Parametry zużycia energii przedstawia rysunek 20. Porównywane są dwa zdefiniowane warianty działania aplikacji. Uwidacznia się niewielka różnica konsumowanej mocy. Węzeł odbierający komunikaty *Generic OnOff* wymaga średnio 39,2mW mocy do funkcjonowania. Konkurencyjny wariant potrzebuje jej nieznacznie mniej. Mesh będący w gotowości potrzebuje średnio 39.0mW mocy. Minimalne i maksymalne natężenia chwilowo pobieranego prądu są porównywalne.

Interesującym aspektem jest porównanie BLE Mesh ze standardową usługą BLE pod względem energetycznym. Rysunek 21 wskazuje na różnice pomiędzy dwoma standardami transmisji. Porównując średnie zużycie energii pomiędzy BLE HRT a Mesh, zauważa się ogromne dysproporcje. Mesh wymaga średnio 39.1mW mocy, by funkcjonować. Jest to rzad wielkości więcej aniżeli usługa BLE HRT, wymagająca jedynie 1.9mW. Co naturalne, również różnice pomiędzy skrajnymi wartościami pobieranego przez urządzenie prądu są znaczące. Wykorzystując ten sam sprzęt, średni pobierany prąd przez skonfigurowany węzeł Mesh wynosi 11.98mA ze szczytem bliskim 30mA. Dla porównania, usługa BLE HRT konsumuje średnio 0.59mA prądu osiągając w *peak'u* ok. 14mA.



Rysunek 20. Zestawienie zużycia prądu dla BLE Mesh w zależności od trybu działania

Różnica przede wszystkim wynika z różnego zastosowania poszczególnych elementów. Zgodnie za dokumentacją producenta i standardu BLE Mesh, energooszczędnym, głównie zasilanym baterijnie węzłem jest węzeł typu *Low Power Node - LPN* [7][18]. Węzeł tego typu oszczędza energię poprzez rzadkie uruchamianie swojego radia wykorzystując przyjacielski węzeł¹⁸ do przechowywania jego danych w buforze, tak by były dostępne dla pozostałych elementów w sieci.



Rysunek 21. Porównanie średniego zużycia energii pomiędzy BT Low Energy HRT i BLE Mesh

¹⁸z ang. *Friend Node* - węzeł pośredniczący, najczęściej zasilany ze stałego źródła energii służący nasłuchiwaniu, odbieraniu i przechowywaniu danych pochodzących z węzła LPN

W wykonanym doświadczeniu wykorzystano zwykły, prosty węzeł. Zapewniając niskie opóźnienia w transmisji danych w sieci, węzeł ten spełnia inne zadania niż urządzenie oparte o zwyczajne usługi BLE bądź węzeł LPN. Usługę BLE HRT zaprojektowano już na poziomie standardu jako energooszczędne domyślnie. Urządzenie wykorzystujące takie usługi, będące najczęściej zasilanie baterijnie, może działać miesiące a nawet lata na pojedynczej baterii (np. typu pastylki CR2032). Dostrzeżona różnica jest więc oczekiwana, naturalna, wynikająca z różnych przeznaczeń wykorzystywanych konfiguracji.

4.3 Badanie Packet Error Rate

Celem niniejszego podrozdziału jest omówienie przeprowadzonego eksperymentu PER. Przedstawiona zostanie metodologia badań oraz zbierane parametry wraz z ich opisem i uzasadnieniem.

Wychodząc z definicji, badany problem zdefiniowany jest przedstawionym wcześniej równaniem 1. Stanowi ono podstawę dla eksperymentu. Jego zrozumienie pozwoliło zaprojektowanie właściwego doświadczenia jak i przygotowanie kompletnego stosu technologicznego niezbędnego do jego przeprowadzenia.

Ostatecznym efektem przeprowadzonego eksperymentu jest przedstawienie zebranych danych w postaci wykresów. Prezentują one badane cechy zmienne, zaprezentowane opisane w sekcji opisu metodologii. Końcowym krokiem jest wyciągnięcie wniosków z zebranych danych.

4.3.1 Metodologia badania

Procedurę badawczą skonstruowano bazując na wzorze 1. Niezbędnym mechanizmem, o które oparte jest doświadczenie, to zliczanie ilości pakietów. Zliczanie dotyczy zarówno węzła bliższego jak i również węzła dalszego odbierającego wysyłane komunikaty. W przypadku tego drugiego elementu opracowano mechanizm odczytywania licznika zmian badanej wartości, patrz: 4.1.3.

Całość doświadczenia przeprowadzono z wykorzystaniem oprogramowania PC – 4.1.4. Oprogramowanie zapewnia dwie główne funkcjonalności: wysyłanie komunikatów przy określonej częstotliwości przez zadaną ilość czasu; odczytywanie wartości licznika węzła dalszego.

Wysyłanym komunikatem jest polecenie zmiany stanu modelu *Generic OnOff*. Wybrano ten standardowy element stosu BLE Mesh ze względu na jego uniwersalność. Nie ogranicza się on jedynie do wybranej platformy czy właściwościowego modelu. Model ten jest zdefiniowany przez Bluetooth SIG, przez co jest niezależny od producenta urządzeń. Czyni to eksperiment powtarzalny, niezależnie od mikrokontrolera.

Odczytywanie stanu licznika wymagało jednakże wykorzystania autorskiego rozwiązania oparte o właściwość modelu Mesh ST. Nie wpływa to jednak na ostateczny rezultat badań, gdyż stworzone polecenia wykorzystywane jest tylko do odczytu i wysłania wartości licznika węzła dalszego do węzła bliższego i komputera osobistego kontrolującego przepływ doświadczenia. Z każdą kolejną próbą

badania PER ten licznik jest automatycznie zerowany, przez co sesja zliczeń zawsze rozpoczyna się od zera.

Eksperyment wyznaczający PER oparto o następujące czynniki zmienne:

- środowisko: teren leśny, teren zurbanizowany
- interwał zapytań
- dystans pomiędzy węzłami
- ilość węzłów składających się na sieć Mesh



Rysunek 22. Fotografia miejsca badań w Kampinoskim Parku Narodowym

Wyznaczanie PER odbyło się w dwóch różnych środowiskach. Jednym z głównych hipotez jest znaczący wpływ środowiska na jakość transmisji danych. Czynniki takie jak temperatura, wilgotność, rodzaj gleby czy tło radiowe może mieć wpływ na komunikację pomiędzy węzłami. Założono, iż tło radiowe może mieć największy wpływ ja transmisję danych. Stąd dobrano możliwie skrajne miejsca do badań oceniając to jako najistotniejszy czynnik:

- Kampinoski Park Narodowy (lokalizacja: parking Roztoka) - jako teren leśny oddalony od ośrodka miejskiego ze względnie niewielkim tłem radiowym. Pogoda: pochmurnie, wilgotno, temperatura poniżej 20°C.

Rozdział 4. Część doświadczalna

- Park Pola Mokotowskie - jako teren zurbanizowany charakteryzujący się bogatym tłem radiowym działającym w pasmach Industrial, Scientific, Medical (ISM). Pogoda: słonecznie, temperatura ok. 20°C.



Rysunek 23. Fotografia miejsca badań na Polach Mokotowskich

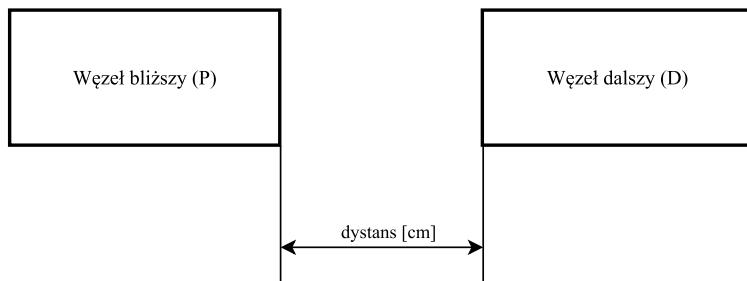
Kolejnym badanym czynnikiem jest interwał zapytań. Parametr ten został wybrany ze względu na obserwowane problemy z komunikacją podczas etapu tworzenia oprogramowania. Parametry dobrano w takim stopniu, by ów problem ukazać. Obrano następujące interwały:

- 100ms
- 500ms
- 800ms
- 1300ms
- 2100ms

Interesującym parametrem dla bezprzewodowej transmisji danych jest zasięg. Stąd też, jednym z badanych czynników jest określenie jakości PER w zależności od odległości - Rysunek 24.

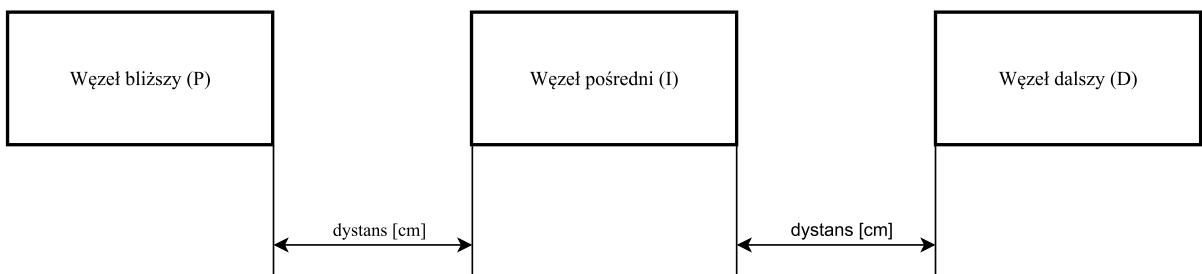
- 1,5m
- 3,0m
- 5,0m
- 8,0m
- 13,0m
- 16,0m
- 21,0m

Wyżej wymienione odległości stosowano również w przypadku kolejnego badanego parametru, tj. ilości węzłów składających się na sieć BLE. W celu łatwej identyfikacji węzłów, wprowadza się właściwe nazewnictwo opisane w punkcie 4.1.3.



Rysunek 24. Dystans pomiędzy węzłami dla sieci dwóch mikrokontrolerów

Postanowiono o równoodległym rozstawieniu węzłów - Rysunek 25. Dla sieci 3 węzłów, maksymalna odległość dzieląca węzeł bliższy od węzła dalszego to 42m. Protokół badawczy zawiera informację tylko o odległości w rozumieniu równoodległego rozstawienia węzłów. Odległość pomiędzy elementami jest oczywistą operacją arytmetyczną.



Rysunek 25. Dystans pomiędzy węzłami dla sieci trzech mikrokontrolerów

Odległość pomiędzy węzłami mierzona jest z użyciem taśmy mierniczej z podziałką 1mm. Tolerancję pomiarów należy przyjąć jako najdłuższy wymiar zestawu uruchomieniowego P-NUCLEO-WB55 - 70mm [13]. Pomiar odbywał się na płasko, mierząc odległość pomiędzy leżącymi na glebie węzłami. Błędy pomiarowe wynikłe z ukształtowania terenu są prawdopodobne i wynikają z poza laboratoryjnego charakteru badań.

W celu zminimalizowania ryzyka pojawienia się losowych błędów o nieznanym pochodzeniu badanie powtarzano pięciokrotnie, w sposób następujący: dla wybranego środowiska, ilości węzłów i zadanej odległości pomiędzy węzłami, wykonaj 5 powtórzeń pomiarowych dla każdego z zadanych interwałów pomiarowych.

$$\exists e, \exists n, \exists d : PER_i = f(e, n, d), i = 1, \dots, 5 \quad (4)$$

gdzie:

e - środowisko

n - ilość węzłów

d - równoodległy dystans pomiędzy węzłami

i - ilość powtórzeń

PER - Packet Error Rate dla wybranego powtórzenia przy stałych parametrach e, n, d

Wykorzystując tak zebrane dane, wyrysowano je na szeregu wykresów wyznaczając jednocześnie linie aproksymacyjne z użyciem modelu liniowego wyznaczonego metodą najmniejszych kwadratów. Odchylenia standardowe zaprezentowane są z użyciem ograniczonego tła otaczającego linię w tym samym kolorze o zmienionym parametrze nieprzezroczystości.

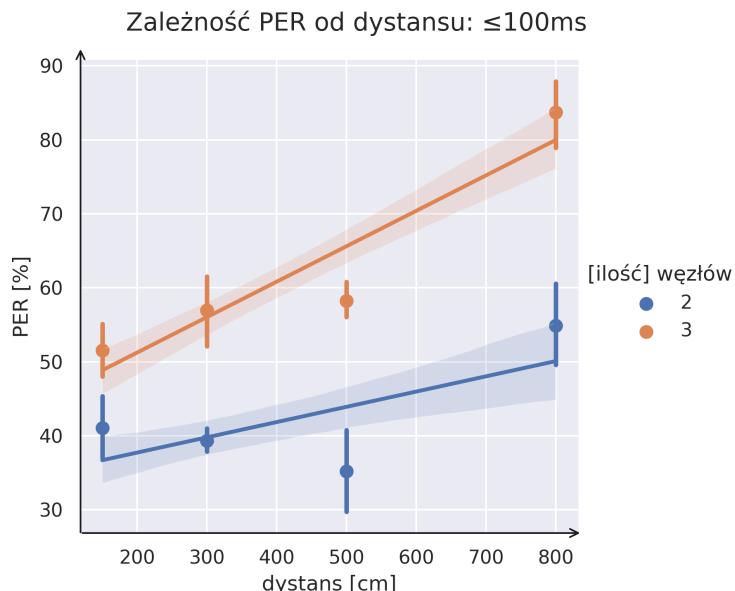
Parametry transmisji danych nie ulegały zmianie podczas przeprowadzanych doświadczeń. Poniższe wartości należy przyjąć za stałe:

- Szybkość transmisji: 2Mbps
- Moc transmisji danych: 0dBm

4.3.2 Zależność PER względem częstości zapytań

Pierwszą sprawdzaną hipotezą jest empiryczna weryfikacja czy częstotliwość zapytań wpływa na PER. W tym celu wykreślono szereg wykresów dla wybranych interwałów czasowych. Pozostałe parametry traktowane są jako stałe.

Rysunek 26 przedstawia zależność PER dla interwału 100 milisekund w zależności od odległości dla sieci złożonej z dwóch i trzech węzłów.



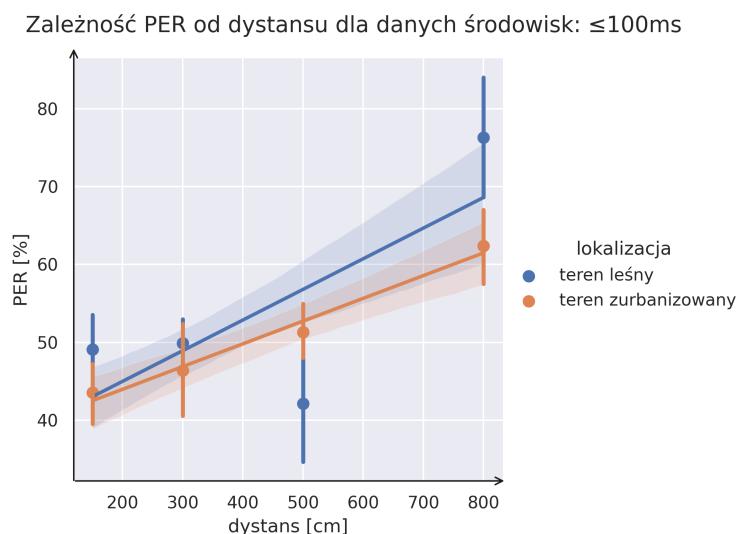
Rysunek 26. Zależność PER od dystansu dla zapytań o częstotliwości $\leq 100\text{ms}$ dla różnej liczby węzłów

Niewątpliwą cechą ukazanych danych jest wysoka wartość PER już przy tak niewielkiej odległości jak 150 cm pomiędzy węzłami. Utrata 40-50% wysyłanych pakietów danych już na pierwszym dystansie pomiarowym może być spowodowana wieloma czynnikami. Przypuszczalnie, wpływ na

taki rezultat wywodzi się z czynników środowiskowych lub ze sposobu wykonywania eksperymentu. Niewykluczone są również ograniczenia sprzętowe mikrokontrolera bądź stosu BLE Mesh.

Wraz ze wzrostem odległości pomiędzy węzłami PER wzrasta pomimo wysokiej wartości początkowej. Jest to oczekiwana zależność i zgodna z wiedzą techniczną.

Prowadząc dalszą analizę zależności PER od częstości zapytań, sprawdzono wpływ środowiska na jakość transmisji danych. Rysunek 27 przedstawia wybraną zależność. Rozróżnienie na ilość węzłów nie zostało tutaj uwzględnione. Pomiary przeprowadzone w najbliższym dobranym dystansie ponownie wskazują na 40-50% utratę pakietów w węźle dalszym. Analogiczne rezultaty obserwowane są na pozostałych dystansach z oczekiwana tendencją wzrostową. Wykres dodatkowo przedstawia pewną różnicę w jakości transmisji danych w zależności od środowiska. Różnica ta jednak mieści się w odchyleniu standardowym, posiadając część wspólną dla zadanych parametrów. Nie jest to wystarczające do stwierdzenia jednoznacznego wpływu środowiska.



Rysunek 27. Zależność PER od dystansu dla zapytań o częstości $\leq 100\text{ms}$ w wybranych środowiskach bez rozróżnienia na liczbę węzłów

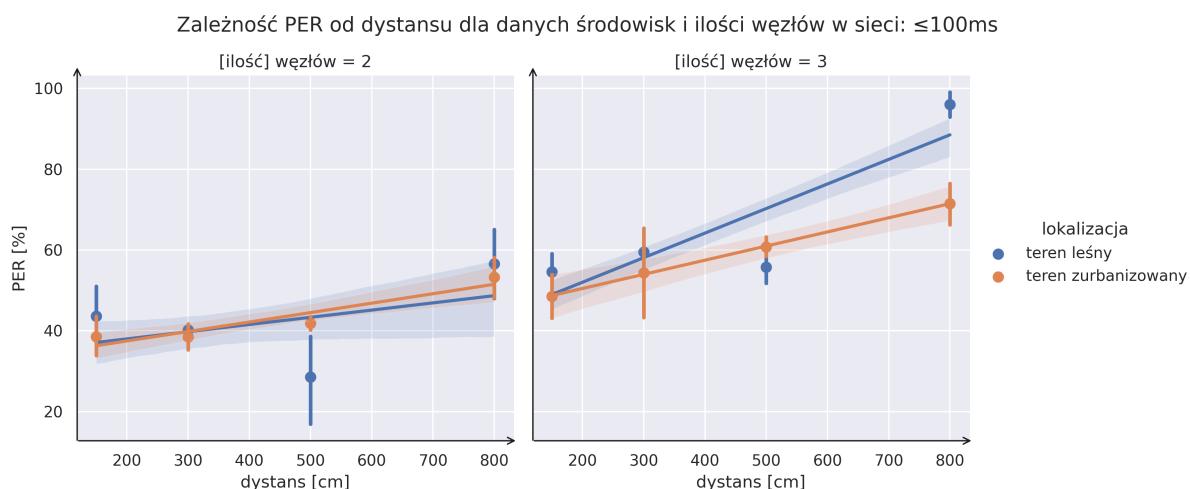
Zestawiając ze sobą wymienione wcześniej czynniki, obserwuje się interesujące zależności. Rysunek 28 wskazuje na zależność dystansu, ilości węzłów i rodzaju środowiska dla wybranego interwału zapytań 100 milisekund. Po raz kolejny obserwowany jest PER wynoszący 40-50%, niezależnie od środowiska czy ilości węzłów. Sugeruje to wpływ samej testowanej platformy na ostateczny rezultat. Prawdopodobną hipotezą jest niewystarczająca wielkość zaalokowanych buforów obsługujących transmisję danych. Mikrokontroler, nie będąc w stanie obsłużyć tak częstej transmisji, może doświadczyć awarii, co zaobserwowano podczas badań. Awaria objawiała się brakiem reakcji węzła bliższego na jakiekolwiek komendy AT, co wymagało ponownego uruchomienia urządzenia. Weryfikacja tego zagadnienia wymagałaby zaangażowania zaawansowanych narzędzi programistycznych ingerujących m.in. w pamięć urządzenia. Niniejsza praca nie podejmuje

Rozdział 4. Część doświadczalna

się wyjaśnienia przyczyn obserwowanych anomalii w działaniu mikrokontrolera, udostępniając jednocześnie możliwy punkt dla dalszych prac badawczych z zakresu BLE Mesh.

Wpływ samego stosu łączności na PER przy zadanej częstotliwości zapytań zdaje się potwierdzać wykres uwzględniający jakość transmisji dla dwóch węzłów. Niemalże pozioma linia aproksymacji sugeruje niewielki wpływ dystansu na PER. Dotyczy to zarówno terenu zurbanizowanego jak i terenu leśnego. Zbliżone rezultaty zdają się wykluczać czynniki zewnętrzne.

Interesującą zależnością jest nachylenie wykresu względem osi odciętych. Dla przypadku dwóch węzłów sieci, połączenie bezpośrednio między węzłami, PER jest niemal stałe na wybranych odległościach, porównywalne z przypadkiem terenu leśnego. W przypadku trzech węzłów, uwidacznia się potencjalny wpływ węzła środkowego, przekazującego pakiety z punktu bliższego do dalszego. Wraz ze wzrostem dystansu, rośnie wartość PER sięgając nawet 90% w terenie leśnym. Różne nachylenie dla wybranych środowisk również sugeruje znaczący wpływ środowiska na PER.

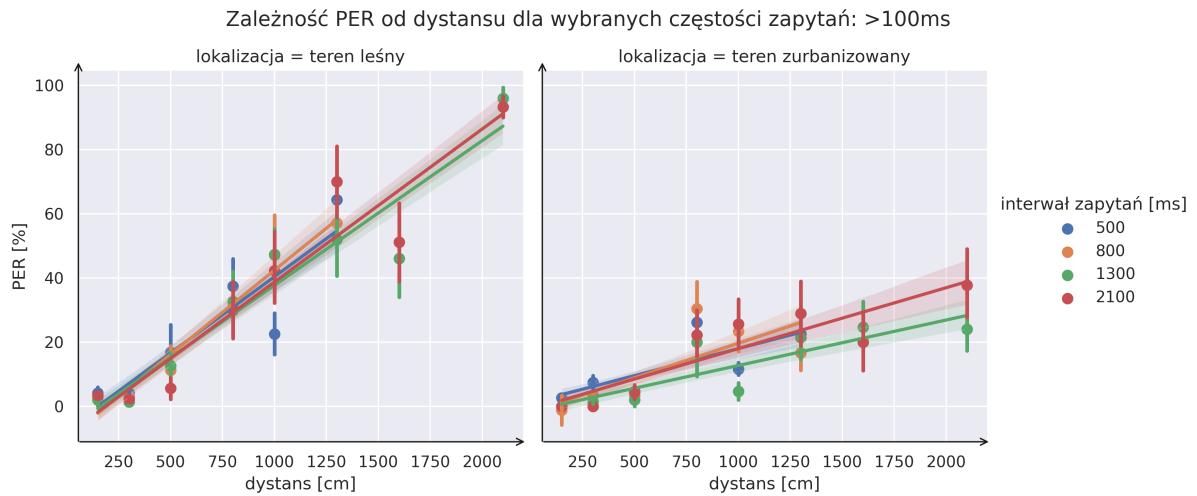


Rysunek 28. Zależność PER od dystansu dla zapytań o częstotliwość $\leq 100\text{ms}$ w wybranych środowiskach i liczbę badanych węzłów

Znając charakterystykę łączności dla okresów poniżej 100 milisekund, weryfikuje się pozostałe wybrane częstotliwości, zgodnie z podanymi wartościami 4.3.1. Rysunek 29 prezentuje pozostałe interwały w zależności od dystansu i wybranych środowisk bez rozróżnienia na liczbę węzłów w sieci.

Przeprowadzone pomiary w terenie leśnym wskazują charakterystykę połączenia zgodną z intuicyjnymi przewidywaniami. Na początkowym dystansie 150cm nie obserwuje się problemów z łącznością. Wszystkie wysłane pakiety zostały odebrane przez węzeł dalszy. Kolejny dystans wskazuje już na pewną utratę pakietów poniżej 20%. Prawdopodobnym czynnikiem jest pogoda lub ukształtowanie terenu leśnego. Co istotne, pomiary dla różnych interwałów odpłytywań są zbliżone. Sugerowałoby to brak wpływu częstotliwości na PER. Na pozostałych dystansach pomiarowych, wartości utraty pakietów są do siebie wzajemnie zbliżone osiągając swoje maksimum na dystansie 21m - blisko 100% zaginionych pakietów.

Charakterystyka łączności w terenie zurbanizowanym prezentuje się podobnie. Nie obserwuje się znaczącej utraty pakietów na względzie bliskich dystansach (150, 300 i 500cm). Wartość PER rośnie wraz ze wzrostem odległości pomiędzy węzłami osiągając w swoim szczytce wartość ok. 40%. Co istotne, linie aproksymacyjne są do siebie zbliżone, ponownie sugerując brak korelacji pomiędzy częstością wysyłania komunikatów a wartością PER.



Rysunek 29. Zależność PER od dystansu dla zapytań o częstości $>100\text{ms}$ w wybranych środowiskach

Zależność pomiędzy PER a częstością odpytywań została zbadana wykorzystując narzędzia statystyczne. Traktując model liniowy, jak sugerują zaprezentowane wykresy, tworzonych zgodnie z metodą najmniejszych kwadratów wylicza się współczynnik determinacji dla następujących zmiennych niezależnych: dystans oraz PER. W kolejnym kroku uwzględnia się fakt wykorzystywania korelacji wielu zmiennych, dostosowując odpowiednio wartość.

Środowisko	R^2	$R^2_{dostosowany}$
Teren leśny	0.04958	0.04272
Teren zurbanizowany	0.04887	0.04200

Tablica 2. Współczynnik determinacji dla zależności interwału zapytań od dystansu i PER w wybranych środowiskach

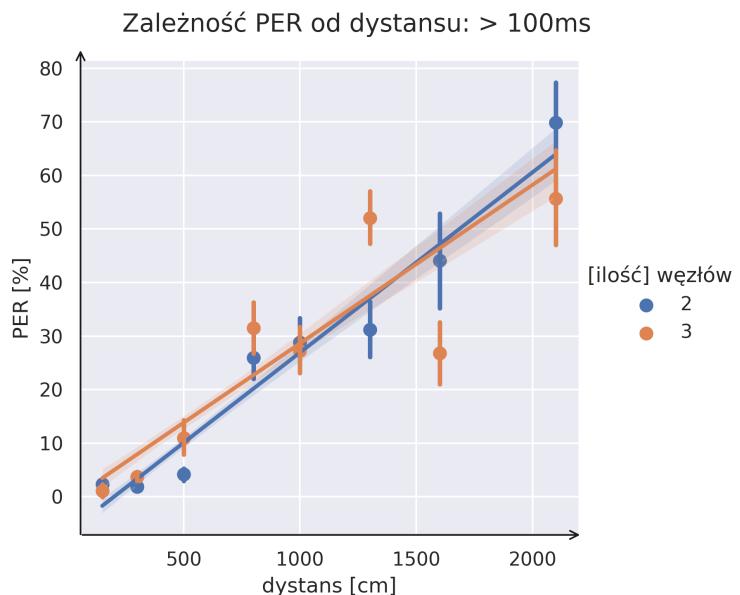
Ostatecznie, otrzymany współczynnik determinacji przyjmujący wartość $R^2 = 0,04 - 0,05$. Istnieje 4%-owy wpływ interwału zapytań o częstościach większych niż 100 milisekund na ostateczny rezultat PER. Pozwala to wykluczyć ten czynnik wykluczyć z dalszych rozważań uznając go za marginalny.

4.3.3 Zależność PER względem odległości między węzłami

Ustaliwszy brak (pomijalnie mały) wpływu częstości zapytań (dla częstości wartości $>100\text{ms}$), praca podejmuje dalszą prezentację danych pod postacią zależności odległości na PER.

Rysunek 30 przedstawia zebrane dane, zestawiając odległość i ilość węzłów składających się na sieć Mesh. Na początkowych dystansach PER ma wartość bliską bądź równą zeru. Wraz ze

wzrostem odległości pomiędzy węzłami, badany współczynnik rośnie, co jest zgodne z oczekiwaniami. Na dystansie 16m obserwuje się przełamanie linii aproksymacji. Dodatkowy węzeł środkowy, prawdopodobnie, znacząco i pozytywnie wpływa na PER przy kolejnych odległościach umożliwiając przekazywanie informacji w sieci. PER w najdalszym punkcie pomiarowym osiąga wartości od ok. 60% do 70%. Należy jednak mieć na uwadze, iż zestawienie nie rozróżnia środowiska w którym następowało zliczanie pakietów.

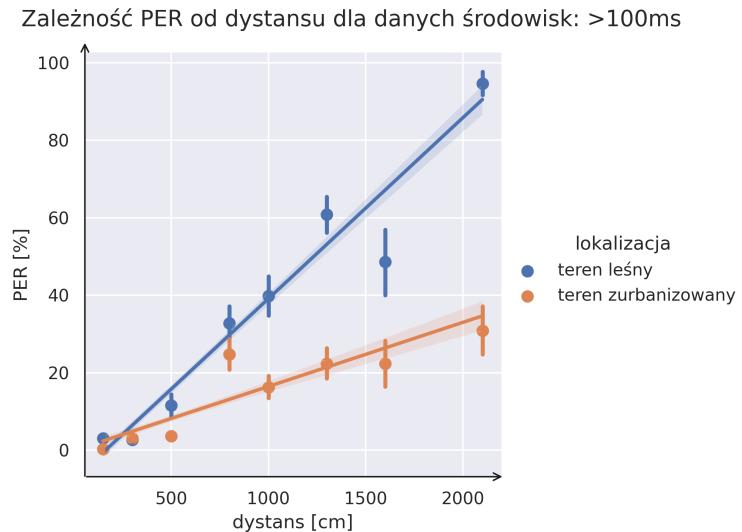


Rysunek 30. Zależność PER od dystansu dla zapytań o częstotliwość $>100\text{ms}$ dla różnej liczby węzłów

Rysunek 31 uwzględnia wpływ środowiska na PER. Wykres zdecydowanie wskazuje na różnicę pomiędzy terenem zurbanizowanym a terenem leśnym. Na początkowych dystansach PER jest zbliżone nieszależnie od odległości międzywęzłowych. Znaczące różnice w pomiarach, a dzięki temu również względem linii aproksymacyjnej, występują już na dystansie 5m. PER w przypadku miejskim jest bliskie zera, gdzie analogiczne pomiary w środowisku leśnym sugerują piętnastoprocentowy poziom zgubionych pakietów. Wraz ze wzrostem dystansu, wartość PER wzrasta. Niemniej jednak nachylenie linii wskazuje na zdecydowanie wolniejsze narastanie utraty danych podczas transmisji bezprzewodowej dla przypadku miejskiego. Na maksymalnym dystansie międzywęzłowym wynoszącym 21m, PER przybiera wartość ok. 30%. W analogicznym przypadku dla środowiska leśnego następuje niemal całkowita utrata transmisji.

Ostatnia prezentowana zależność ukazana jest na Rysunku 32. Przedstawia on zależność PER od dystansu dla wybranych środowisk i liczby węzłów. Uwidaczniają się wyżej wymienione cechy transmisji opisane w poprzednich akapitach.

Dla przypadku dwóch węzłów, PER jest bliskie zeru na początkowych dystansach do 5m nieszależnie od środowiska. W odległości 8m od stacji bazowej obserwuje się miejscową anomalię polegającą na blisko 40%-ej utraty pakietów dla terenu zurbanizowanego. Zarówno jak w poprzednich jak i kilku

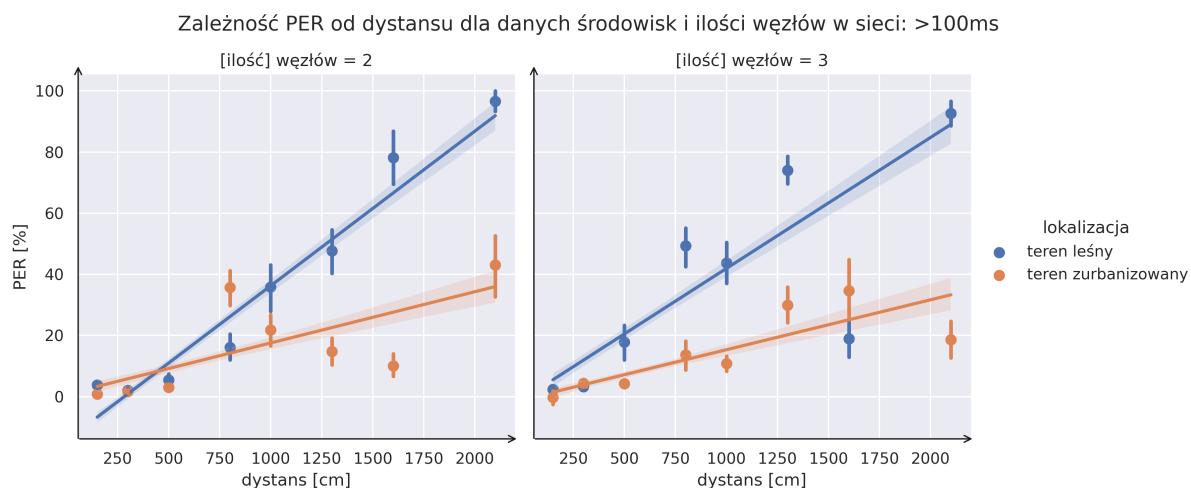


Rysunek 31. Zależność PER od dystansu dla zapytań o częstości $>100\text{ms}$ w wybranych środowiskach bez rozróżnienia na liczbę węzłów

następujących po sobie punktach pomiarowych, PER nie przybiera takich wartości. Anomalia może być wyjaśniona cechami środowiska - np. węzeł dalszy ułożony został w zagłębieniu, co utrudniło odbiór propagowanej fali radiowej. W pozostałych punktach pomiarowych w terenie zurbanizowanym obserwuje się spadek wartości PER wraz ze zwiększonym dystansem, co znów sugeruje wpływ ukształtowania terenu. Utrata pakietów wynosząca niewiele powyżej 40% zaobserwowana jest na odległości 21m. Jest to o tyle zaskakujące, o tyle dla analogicznego dystansu pomiarowego w terenie leśnym PER wynosi niemal 100%.

Charakterystyka transmisji dla badanego przypadku trzech węzłów przybiera podobną postać. Wartości PER w terenie zurbanizowanym są zbliżone w początkowych dystansach pomiarowych. Dystans 8m (węzeł dalszy na odległość 16m) wskazuje na analogiczną anomalię jak w opisywanym przypadku dwóch węzłów. Obserwuje się maksimum lokalne PER poniżej 20% w otoczeniu najbliższych punktów pomiarowych. Kolejne pomiary wskazują na stały przyrost PER względem równoodległych węzłów. Utrata pakietów na maksymalnym całkowitym dystansie 42m wynosi ok. 20%.

Transmisja danych w środowisku leśnym charakteryzuje się większym przyrostem PER względem odległości w porównaniu do warunków miejskich. Przypadek dwuwęzłowej sieci wskazuje zbliżone wartości utraty pakietów na odległość do 5m włącznie. Na każdym kolejnym dystansie pomiarowym PER wzrasta zbliżając się do 100% w miejscu oddalonym o 21m od węzła bliższego. Oznacza to niemal całkowitą utratę łączności. Charakterystyka dla sieci zbudowanej z trzech węzłów jest analogiczna. Wyjątkiem jest zbiór pomiarów zebranych na dystansie pomiędzy węzłami wynoszącym 16m. PER w tym przypadku spada do ok. 20%. Jest to o tyle niespodziewane iż, poprzedzające i następujące pomiary i wynikłe w konsekwencji PER są blisko 4-krotnie wyższe. Taką konsekwencję należy przypisać ukształtowaniu terenu aniżeli nieznanej właściwości transmisji radiowej.



Rysunek 32. Zależność PER od dystansu dla zapytań o częstotliwość >100ms w wybranych środowiskach i liczbę badanych węzłów

Przedstawione dane prezentują wpływ dystansu na PER w różnych środowiskach. Określona kategoria badanych środowisk nie jest jedynym parametrem charakteryzującym przeprowadzone badania. Dużą odpowiedzialność w ostatecznych zebranych wartościach należy położyć na czynniki takie jak ukształtowanie terenu i pogoda. Pomimo organoleptycznego doboru terenu o możliwie nieurozmaiconej charakterystyce, badane węzły mogły zostać ułożone w lokalnym zagłębiu wynikłym z gleby bądź nawet zagęszczeniu trawy w danym miejscu. Różnica w wilgotności również mogła mieć znaczenie w zebranych danych. Niepewności te są konsekwencją metodologii przeprowadzonych badań w warunkach nielaboratoryjnych (terenowych). Czynniki te, nie będące częścią formalnie zebranych danych, najprawdopodobniej miały wpływ na ilość odebranych pakietów. Ukształtowanie terenu jak i wilgotność mają bezpośredni wpływ jakość transmisji stanowiąc przeszkody dla strefy propagacji sygnału radiowego (strefy Fresnela).

Rozdział 5

Podsumowanie

- Teoria: wiemy jak ma działać, ale jednak nie działa.
- Praktyka: działa, ale nie wiemy – dlaczego?
- Łączymy teorię z praktyką: nic nie działa i nie wiemy, dlaczego.

Więcej informacji na temat \LaTeX a:

- <<https://www.overleaf.com/learn>> – przystępny tutorial na stronie Overleaf,
- <<https://www.latex-project.org/>> – strona domowa projektu,
- <<https://www.tug.org/begin.html>> – dobry zbiór odnośników do innych materiałów.

Powodzenia!

Bibliografia

- [1] *Bluetooth Protocol Stack - MATLAB & Simulink*. adr.: <https://www.mathworks.com/help/bluetooth/ug/bluetooth-protocol-stack.html> (term. wiz. 15.05.2022).
- [2] Mesh Working Group, *Mesh Profile Bluetooth® Specification*, sty. 2019. adr.: <https://www.bluetooth.com/specifications/specs/mesh-profile-1-0-1/>.
- [3] *P-NUCLEO-WB55 - Bluetooth 5 and 802.15.4 Nucleo Pack including USB dongle and Nucleo-64 with STM32WB55 MCUs, supports Arduino Uno V3 and ST morpho connectivity - STMicroelectronics*, en. adr.: <https://www.st.com/en/evaluation-tools/p-nucleo-wb55.html> (term. wiz. 10.05.2022).
- [4] SA, H., *TCP/IP. Szkoła programowania*, pl, ISBN: 978-83-246-0293-3. adr.: https://helion.pl/ksiazki/tcp-ip-szkola-programowania-heather-osterloh_tcpszp.htm (term. wiz. 15.05.2022).
- [5] *scipy.integrate.simpson — SciPy v1.8.1 Manual*. adr.: <https://docs.scipy.org/doc/scipy/reference/generated/scipy.integrate.simpson.html> (term. wiz. 25.05.2022).
- [6] Skoro, M., *Fizyka*. Państwowe Wydawnictwo Naukowe, 1973.
- [7] ST, *AN5292 - How to build a Bluetooth® Low Energy mesh application for STM32WB Series microcontrollers*, en, grud. 2021.
- [8] *STM32CubeIDE - Integrated Development Environment for STM32* - STMicroelectronics, 2022. adr.: <https://www.st.com/en/development-tools/stm32cubeide.html> (term. wiz. 14.05.2022).
- [9] *STM32CubeMonPwr - Graphical tool displaying on PC power data coming from X-NUCLEO-LPM01A* - STMicroelectronics, 2022. adr.: <https://www.st.com/en/development-tools/stm32cubemonpwr.html> (term. wiz. 14.05.2022).
- [10] *STM32CubeProg - STM32CubeProgrammer software for all STM32* - STMicroelectronics, 2022. adr.: <https://www.st.com/en/development-tools/stm32cubeprog.html> (term. wiz. 14.05.2022).
- [11] *STM32CubeWB MCU Firmware Package*, original-date: 2019-04-19T13:27:35Z, maj 2022. adr.: <https://github.com/STMicroelectronics/STM32CubeWB> (term. wiz. 14.05.2022).
- [12] *STM32WB - Bluetooth, Wireless Microcontrollers (MCU)* - STMicroelectronics, 2022. adr.: <https://www.st.com/en/microcontrollers-microprocessors/stm32wb-series.html> (term. wiz. 14.05.2022).

Bibliografia

- [13] STMicroelectronics, *UM2435 - Bluetooth® Low Energy and 802.15.4 Nucleo pack based on STM32WB Series microcontrollers*, 2019.
- [14] ——, *AN5289 - Building wireless applications with STM32WB Series microcontrollers*, 2021.
- [15] ——, *PM0271 - STM32WB BLE stack programming guidelines*, grud. 2021. adr.: https://www.st.com/resource/en/programming_manual/pm0271-stm32wb-ble-stack-programming-guidelines-stmicroelectronics.pdf.
- [16] *UM1718 - STM32CubeMX for STM32 configuration and initialization C code generation*, maj 2022. adr.: https://www.st.com/resource/en/user_manual/dm00104712-stm32cubemx-for-stm32-configuration-and-initialization-c-code-generation-stmicroelectronics.pdf.
- [17] *UM2243 - STM32 Nucleo expansion board for power consumption measurement*, mar. 2018. adr.: https://www.st.com/resource/en/user_manual/um2243-stm32-nucleo-expansion-board-for-power-consumption-measurement-stmicroelectronics.pdf.
- [18] Wooley, M., *Bluetooth Mesh Models - Technical Overview*, mar. 2019. adr.: https://www.bluetooth.com/wp-content/uploads/2019/04/1903_Mesh-Models-Overview_FINAL.pdf (term. wiz. 26.05.2022).

Wykaz skrótów i symboli

API Application Programming Interface 14, 32

AT Zbiór poleceń Hayes'a (znany jako zbiór polecen AT) 26, 31, 50

BLE Bluetooth Low Energy 14, 18, 36, 37, 40

BMS Battery Management System 21

CAD Computer-aided Design 21

FDM Fused Deposition Modeling 21

FFF Fused Filamnet Fabrication 21

HRT Heart Rate 17, 23, 37

IDE Integrated Development Environment - Zintegrowane środowisko programistyczne 17

ISM Industrial, Scientific, Medical 46

LED Light Emitting Diode (dioda emitująca światło) 29

LL Link Layer 14

PCB Printed Circuit Board 21

PER Packet Error Rate 15, 17, 18, 24, 44, 48–54

PLA Polylactic Acid 21

UART Universal asynchronous receiver-transmitter 30

XML Extensible Markup Language 32

Spis rysunków

1	Model ISO OSI wraz i odpowiadające mu nazwy porcji danych. Źródło: [4]	14
2	Zestawienie stosu BLE i modelu ISO OSI. Źródło: [1]	15
3	Zestaw uruchomieniowy P-NUCLEO-WB55	19
4	Zestaw pomiarowy X-NUCLEO-LPM01A	20
5	Odbiornik energii	21
6	Model obudowy dla zestawu P-NUCLEO-WB55	22
7	Obudowa zestawu uruchomieniowego Nucleo wykonana w technologii druku 3D – realizacja	23
8	Wykorzystanie interfejsu szeregowego do wysyłania poleceń. Źródło: [7]	31
9	Selektor portów szeregowych w aplikacji służącej eksperymentowi PER.	32
10	Selektor portów szeregowych w aplikacji służącej eksperymentowi PER.	33
11	Właściwy interfejs użytkownika do przeprowadzenia doświadczenia PER	35
12	Podłączony zestaw pomiarowy	37
13	Przykładowa sesja pomiarowa dla BLE Mesh - sieć w trybie nasłuchującym	38
14	Charakterystyka czasowa poboru prądu dla BLE i usługi Heart Rate - Usługa Połączona	39
15	Charakterystyka czasowa poboru prądu dla BLE i usługi Heart Rate - Tryb szybkiego ogłoszania	39
16	Charakterystyka czasowa poboru prądu dla BLE i usługi Heart Rate - Tryb niskomocowego ogłoszania	40
17	Zestawienie zużycia prądu dla usługi Heart Rate w zależności od trybu działania	40
18	Charakterystyka czasowa poboru prądu dla BLE Mesh i modelu Generic OnOff	41
19	Rzeczywista charakterystyka poboru energii dla BLE Mesh działającego w trybie gotowości	42
20	Zestawienie zużycia prądu dla BLE Mesh w zależności od trybu działania	43
21	Porównanie średniego zużycia energii pomiędzy BT Low Energy HRT i BLE Mesh	43
22	Fotografia miejsca badań w Kampinoskim Parku Narodowym	45
23	Fotografia miejsca badań na Polach Mokotowskich	46
24	Dystans pomiędzy węzłami dla sieci dwóch mikrokontrolerów	47
25	Dystans pomiędzy węzłami dla sieci trzech mikrokontrolerów	47

26	Zależność PER od dystansu dla zapytań o częstotliwość $\leq 100\text{ms}$ dla różnej liczby węzłów	48
27	Zależność PER od dystansu dla zapytań o częstotliwość $\leq 100\text{ms}$ w wybranych środowiskach bez rozróżnienia na liczbę węzłów	49
28	Zależność PER od dystansu dla zapytań o częstotliwość $\leq 100\text{ms}$ w wybranych środowiskach i liczbę badanych węzłów	50
29	Zależność PER od dystansu dla zapytań o częstotliwość $>100\text{ms}$ w wybranych środowiskach	51
30	Zależność PER od dystansu dla zapytań o częstotliwość $>100\text{ms}$ dla różnej liczby węzłów	52
31	Zależność PER od dystansu dla zapytań o częstotliwość $>100\text{ms}$ w wybranych środowiskach bez rozróżnienia na liczbę węzłów	53
32	Zależność PER od dystansu dla zapytań o częstotliwość $>100\text{ms}$ w wybranych środowiskach i liczbę badanych węzłów	54

Spis tablic

1	Zebrane empirycznie współczynniki kalibracyjne w porównaniu z wartościami obliczonymi	34
2	Współczynnik determinacji dla zależności interwału zapytań od dystansu i PER w wybranych środowiskach	51

Spis załączników

1 Kod źródłowy	67
-------------------------	----

Załącznik 1

Kod źródłowy

Wraz z pracą dyplomową załączone są wszystkie pliki związane z projektem, włączając w to kod źródłowy.

Publiczne repozytorium dostępne jest na platformie GitHub pod poniższym linkiem:

- https://github.com/krkruk/stm32wb_mesh_packet_error_rate