Politechnika Śląska
Wydział Automatyki, Elektroniki
i Informatyki

ANALIZA METOD IMPLEMENTACJI SIECI PROGRAMOWALNYCH W KOMPUTEROWYCH SYSTEMACH PRZEMYSŁOWYCH WYKORZYSTUJĄCYCH PRZEMYSŁOWY INTERNET RZECZY

MGR INŻ. IRENEUSZ SMOŁKA

Rozprawa doktorska napisana pod kierunkiem
Prof. dr. hab. inż. Andrzeja Kwietnia

Gliwice 2023

Chciałbym podziękować rodzicom za wsparcie przy podejmowaniu ważnych decyzji oraz pomoc w realizacji zaplanowanych celów.
A także Ani za wiele cennych rad i motywację do działania.

Dziękuję!

Spis treści

1 Wstęp 5

1.1 Cel pracy 6
1.2 Tezy pracy 7
1.3 Układ pracy 7

2 Komputerowe systemy przemysłowe 8

2.1 Modele referencyjne systemów przemysłowych 8
2.2 Systemy czasu rzeczywistego 12

2.2.1 Sterownik PLC i zasada jego działania 15

2.3 Systemy zdecentralizowane 16
2.4 Sieci przemysłowe 17
2.5 Komunikacja pomiędzy urządzeniami sieci przemysłowej 17
2.6 Sieć przemysłowa – topologia 17
2.7 Sieci przemysłowe - protokoły 26

3 Przemysłowy Internet Rzeczy 36
4 Sieci definiowane programowo 38

4.1 Zasada działania sieci SDN 38
4.2 Protokoły i narzędzia SDN 40

4.2.1 Protokół OpenFlow 40
4.2.2 Język programowania P4 43
4.2.3 Wirtualny przełącznik sieciowy Open vSwitch 46

4.3 Kontroler sieci SDN – porównanie 50

4.3.1 Kontroler OpenDaylight 50
4.3.2 Kontroler ONOS 53
4.3.3 Kontroler RYU 55
4.3.4 Porównanie kontrolerów 55

5 Analiza możliwości wykorzystania sieci SDN w systemach czasu rzeczywistego 58

5.1 Rozwiązania stosowane obecnie 58
6 Symulacja systemów SDN 59
6.1.1 Symulacja sieci w narzędziu Mininet 60
6.1.2 Symulator sieci komputerowej - GNS3 66
6.2 Opis metodologii badań 68
7 Model badawczy 69
8 Koncepcja badań 75
8.1 Urządzenie wbudowanie a praca w systemach czasu rzeczywistego 75
8.2 Zastosowanie SDN w systemach czasu rzeczywistego 81
8.3 Pozyskiwanie danych przy użyciu SDN 82
8.4 Programowanie logicznej topologii sieciowej 86
8.5 Uruchomienie Wireless EtherCAT 89
8.6 Inne rozwiązania - OPC UA (OPC Unified Architecture) 91
8.7 Przemysłowy Internet Rzeczy 94
9 Prezentacja wyników badań 97
9.1 Wykorzystanie urządzeń wbudowanych w systemach czasu rzeczywistego 97
9.1.1 Urządzenie wbudowane Raspberry Pi 97
9.1.2 Beagle Bone Black 103
9.2 Koncepcja SDN 109
9.2.1 Analiza ruchu sieciowego przez urządzenie NetFPGA 109
9.2.2 Analiza ruchu sieciowego SDN przez urządzenie NetFPGA 110
9.3 SDN topologia logiczna 112
9.3.1 Topologia sieciowa bez używania przełączników sieciowych SDN 113
9.3.2 Topologia sieciowa z użyciem przełączników sieciowych SDN 114
9.4 Komunikacja bezprzewodowa w sieciach SDN 118
9.5 Integracja sieci przemysłowej z OPC UA 122
10 Podsumowanie 125

11 Spis rysunków 128
12 Bibliografia 133
13 Uzupełnienie 139

1 Wstęp

W roku 2015, po raz pierwszy pojawiło się określenie „Czwarta rewolucja przemysłowa”, które jest następstwem wprowadzania w Europie z początkiem drugiego dziesięciolecia XXI wieku strategii „Przemysłu 4.0”. Wspomniana strategia polegając na integracji współpracy człowieka z maszynami oraz systemami informatycznymi pozwalającymi na wymianę informacji poprzez Internet.

Zgodnie z ideą „Przemysłu 4.0” [1], [2] komunikacja pomiędzy poszczególnymi elementami systemu może występować w dwóch kierunkach: poziomym oraz pionowym. Kierunek pionowy określa przesyłanie informacji począwszy od części zarządczej przedsiębiorstwa poprzez systemy informatyczne, operatora maszyn, aż po końcowe elementy wykonawcze i czujniki. Ten sam kierunek określa przesyłanie informacji np. z czujników do działów odpowiedzialnych za przetwarzanie pobranych danych. Komunikację pomiędzy kolejnymi urządzeniami lub systemami informatycznymi można opisać jako kierunek poziomy, gdzie przesyłane informacje są dostarczane do elementów systemu będącymi na tym samym poziomie w hierarchii.

Ściśle powiązanymi terminami z „Przemysłem 4.0” są „Internet Rzeczy” oraz „Przemysłowy Internet Rzeczy”. Określają one system urządzeń elektronicznych wzajemnie komunikujących się ze sobą i wymieniających dane wymagane do stabilnej pracy systemu. Aktualnie do grupy urządzeń wykorzystujących rozwiązania „Internetu Rzeczy” zaliczamy zarówno urządzenia domowe jak pralki, lodówki, telewizory, ale i również urządzenia kontroli ruchu drogowego, monitorowania stanu zdrowia czy urządzenia wykorzystywane do produkcji.

W związku ze stale rosnącą liczbą urządzeń wymagających nie tylko szybkiego przesyłania danych, ale również odpowiedniego zarządzania nimi w sieci, zaproponowano koncepcję sieci definiowanych programowo lub używając innego określenia sieci swobodnie programowalnych. Koncepcja ta pozwala na oddzielenie warstwy transportowej od warstwy zarządzającej ruchem w sieci przez

co możliwe staje się scentralizowane zarządzanie wszystkimi urządzeniami przesyłającymi. Dzięki takiemu sterowaniu możliwe staje się filtrowanie ruchu sieciowego w oparciu nie tylko o konkretne adresy fizyczne urządzeń, ale również na podstawie rodzaju przesyłanych danych i wykorzystywanych protokołów (pole Ethertype w ramce sieciowej).

Rozprawa przedstawia możliwość wykorzystania urządzenia wbudowanego do zarządzania siecią urządzeń „Internetu Rzeczy" poprzez implementację kontrolera sieci SDN oraz dynamicznego przydzielania odpowiednich parametrów połączenia nawiązywanego pomiędzy kolejnymi elementami składowymi systemu. W pracy przedstawiono możliwość zastosowania zarówno fizycznego urządzenia wbudowanego jak i jego zwirtualizowaną wersję uruchomioną na komputerze typu PC.

1.1 Cel pracy

Celem pracy jest opracowanie i przedstawienie ***metod*** działania w przypadku zastosowania sieci definiowanych programowo SDN w infrastrukturze sieci systemu przemysłowego. Wprowadzając sieci SDN do systemu sterowania wzięto pod uwagę wymagania determinizmu czasowego stawiane tego typu systemom. Częściowo odrzucono jednak koncepcję, że czas pracy urządzenia sieciowego musi być jak najmniejszy i porównywalny co do rzędu wartości z czasem cyklu działania sterownika.

W pracy zaproponowano model sieci przemysłowej z wykorzystaniem urządzeń i rozwiązań dostępnych na rynku, których czas przetwarzania danych nie pozwalałby na wykorzystanie w systemach, gdzie wymagany jest bardzo krótki czas reakcji, ale możliwe jest jego wykorzystanie przy wprowadzeniu do badań założenia, mówiącego, że czas pracy samych urządzeń jest na tyle duży, że wprowadzenie dodatkowego urządzenia sieciowego nie będzie mieć wpływu na pracę całości systemu.

Planując i prowadząc badania starano się wybierać rozwiązania mające znaczenie w obecnie projektowanych systemach sterowania, a także te, które były możliwe do przetestowania w praktyce. Nie ograniczano się także do zmiany

wykorzystywanego protokołu sieciowego, ale także do wprowadzania zmian w topologii sieciowej. Dzięki zastosowaniu programowalnych przełączników sieciowych, spróbowano wykonać eksperymenty polegające na programowej zmianie fizycznych połączeń w przełączniku.

Mając na względzie ciągły rozwój i możliwość zastosowania zaproponowanych rozwiązań w praktyce do badań wykorzystano urządzenia instalowane w rzeczywistych systemach sterowania tj. sterowniki PLC, zdalne wyspy wejść/wyjść czy bramy pozwalające na integrację systemów przemysłowego Internetu Rzeczy. Wyjątek stanowią tutaj tylko urządzenia sieciowe, które w dużej mierze nie są oferowane w wersji "przemysłowej".

1.2 Tezy pracy

Tezy pracy zostały zdefiniowane, tak aby krok po kroku przeanalizować możliwości stosowania rozwiązań SDN w infrastrukturze sieci przemysłowych. Potwierdzenie zaproponowanych tez pozwoli również na zaplanowanie dalszych prac ukierunkowanych na wirtualizację proponowanych rozwiązań.

Teza 1: Możliwe jest zastosowanie urządzeń wbudowanych wspierających system operacyjny Linux i dostępnych na rynku jako kontrolerów sieci definiowanych programowo.

Teza 2: Możliwe jest wykorzystanie sieci definiowanych programowo w projektach przemysłowych w celu zarządzania urządzeniami przemysłowego Internetu rzeczy (ang. *Industrial Internet of Things*)

Teza 3: Możliwe jest wykorzystanie sieci definiowanych programowo w projektach przemysłowych jako integratora różnych systemów stosowanych obecnie i w przyszłości.

1.3 Układ pracy

Rozprawa doktorska posiada następujący układ. Rozdział 1 jest swoistym nakreśleniem celu i problematyki przedstawionej w rozprawie i sprawdzonej w części eksperymentalnej. Rozdziały od drugiego do czwartego skupiają się na części teoretycznej pozwalającej przedstawienie problemów i zagrożeń, które

wynikają z zastosowanych technologii i rozwiązań, a które zostały odniesione do aktualnie stosowanych rozwiązań w rozdziale 5. W rozdziale 6 zaprezentowano zaproponowany model badawczy pokazujący wpływ poszczególnych elementów na działanie całej sieci. Kolejne rozdziały siódmy i ósmy przedstawiają część symulacyjną i eksperymentalną. Ostatnią część jest podsumowanie całej pracy i przedstawienie wniosków końcowych.

2 Komputerowe systemy przemysłowe

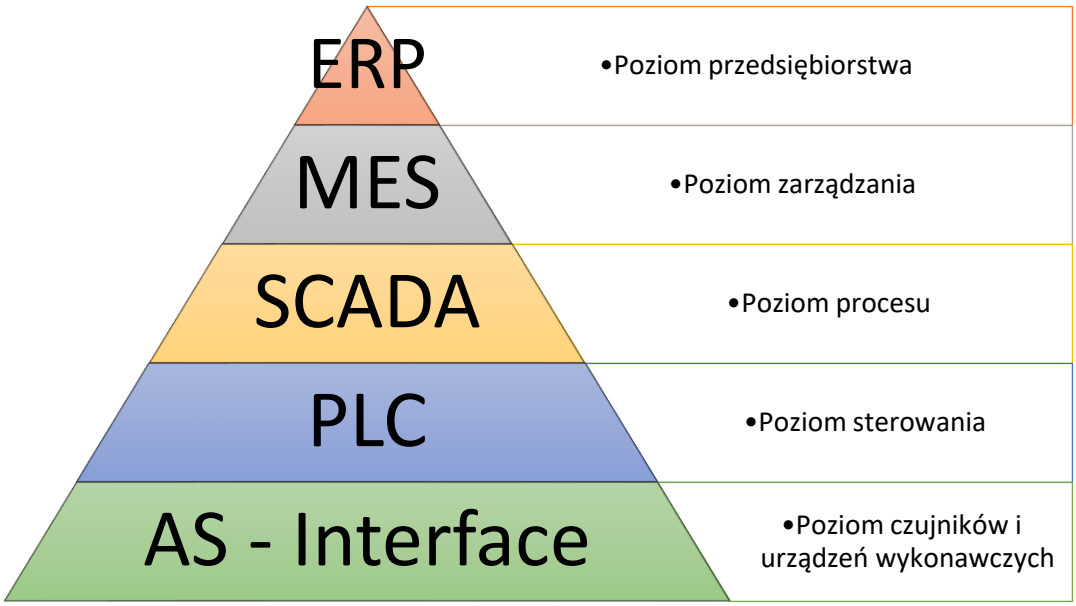
Systemy przemysłowe to pojęcie określające nie tylko połączenie kilku urządzeń sterujących w jeden system, ale również cała infrastruktura sieciowa wraz z systemami nadrzędnymi (systemami zarządzania). Do poprawnego wyjaśnienia zasadności przeprowadzonych badań i w ogóle rozpoczęcia prac nad pracą doktorską należy wspomnieć o kilku modelach referencyjnych stosowanych do opisu systemów przemysłowych i określić, w którym miejscu znajdzie zastosowanie zaprezentowane rozwiązanie. Dodatkowo będzie można podać możliwy wpływ na całość działania systemu.

2.1 Modele referencyjne systemów przemysłowych

Pierwszym z omawianych modeli referencyjnych jest piramida narzędzi i technologii automatyki. Na *Rysunku 3.1* widzimy, że podstawą takiego modelu są elementy wykonawcze oraz urządzenia dokonujące pomiarów. Ta część jest odpowiedzialna za interakcję z częścią fizyczną obiektu. To również te dane muszą zostać przesłane w odpowiednio krótkim, zależnym od ich rodzaju czasie. Kolejną warstwą jest część sterowania, w której znajdują się urządzenia sterujące np. sterowniki PLC. Zarówno warstwa AS jak i PLC mają ogromne znaczenie dla wyboru rozwiązań w części SCADA. To w tej części dane są prezentowane i realizowana jest szeroko rozumiana komunikacja. To również w tej części swoje miejsce znajduje proponowana technologia przemysłowych sieci definiowanych programowo. Dzięki możliwościom zaproponowanym w dalszej części rozprawy możliwe staje pozyskiwanie użytecznych danych bez zmiany „schematu” sieci

przemysłowej. Również możliwości integracji kilku różnych systemów zostałyby umieszczone w tym miejscu.

Ze względu na możliwości programowania całego ruchu sieciowego każdy kolejny etap również będzie mógł stać się częścią pozyskującą dane bez szkodliwego wpływu na działanie systemu sterowania, który będzie niejako odseparowany od innych elementów. Przygotowane reguły dla kontrolera całej sieci pozwolą również na przefiltrowanie całości ruchu jedynie pod kątem danych najbardziej użytecznych, czy to na potrzeby sterowania, czy na potrzeby dalszej analizy.



Rysunek 2.1 Piramida automatyzacji

Drugim modelem referencyjnym do którego odniesiono się w pracy jest model IoT.[3], [4] U podstaw tego modelu podobnie jak w poprzednim przypadku stoją rzeczywiste urządzenia, a poziom wyżej mamy informację o komunikacji. W tym modelu mamy jednak uwzględnione elementy, które wprowadza pojęcie Przemysłu 4.0. Z tego względu w modelu tym znajdziemy dodatkowo połączenie z chmurą danych, archiwizowanie danych, a także aplikacje końcowe zapewniające dostęp do „wypracowanych" informacji. Uwzględniając wszystkie 8 poziomów modelu IoT (*Rysunek 3.2*), wprost zastosowanie sieci definiowanych programowo możemy określić w na poziomach 2 i 8. W poziomie 2 urządzenia sieciowe tak jak poprzednio są odpowiedzialne za poprawne przesyłanie danych

pomiędzy urządzeniami. Poziom 8 natomiast określą kwestię bezpieczeństwa, czyli wykorzystanie sieci SDN do zapewniania połączenia tylko określonym użytkownikom sieci lub filtrowanie szkodliwego dla systemu przemysłowego ruchu.

Odnosząc się do tematyki zawartej w temacie samej pracy jak i przedstawionych tezach, urządzenia znajdujące się w najniższej warstwie mogą być reprezentowane przez urządzenia wbudowane. Przez urządzenia wbudowane w tym kontekście możemy rozumieć zarówno już gotowe płytki uruchomieniowe, jak i specjalnie projektowane urządzenia do specyficznych zastosowań. Rozwiązania dostępne na rynku zostały sprawdzone pod kątem wydajności i możliwości stosowania w tego typu systemach.

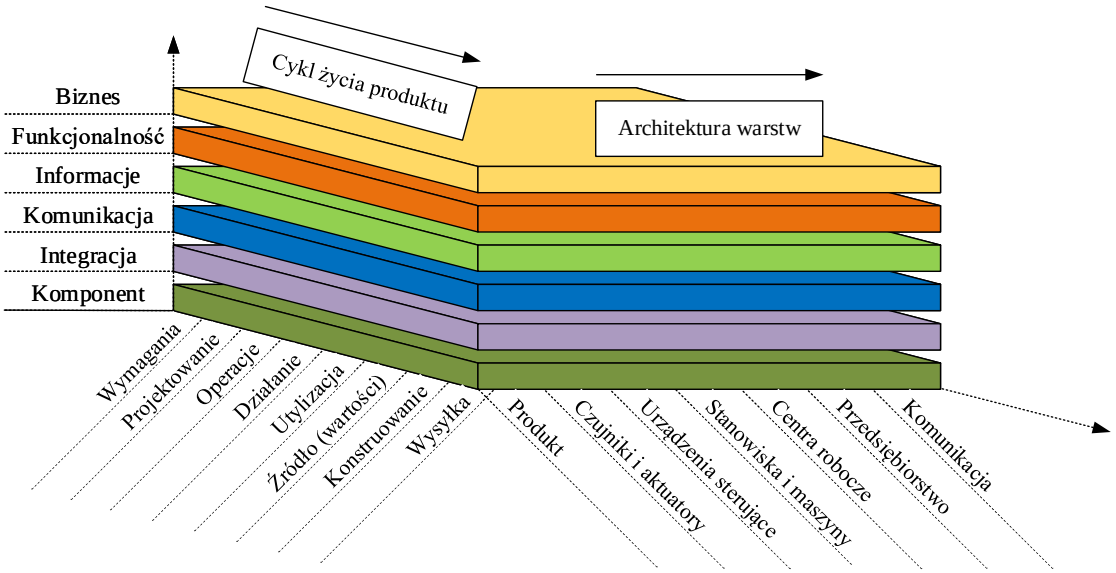
Poziom 8: Bezpieczeństwo
Poziom 7: Współpraca z użytkownikiem
Poziom 6: Aplikacja końcowa
Poziom 5: Dostęp do danych
Poziom 4: Przechowywanie danych
Poziom 3: Analiza danych (Edge computing)
Poziom 2: Komunikacja
Poziom 1: Fizyczne urządzenia

Rysunek 2.2 Model ISO komunikacji sieciowej

Na *Rysunku 3.3* przedstawiono strukturę modelu referencyjnego RAMI 4.0.[5] Model ten pozwala na szczegółowe opisanie architektury systemu przemysłowego oraz komunikowania się pomiędzy poszczególnymi częściami przedsiębiorstwa. W modelu RAMI zależności pomiędzy poszczególnymi częściami możliwa jest do opisania jako hierarchia zamodelowana jako

trójwymiarowa struktura. W podstawowej strukturze możliwe jest wyznaczenie trzech osi:

- Oś cyklu życia i tworzenia wartości,
- Oś architektury warstw,
- Oś poziomu hierarchii.



Rysunek 2.3Model referencyjny RAMI 4.0 systemu przemysłowego

W osi poziomów hierarchii możliwe jest zamodelowanie przepływu surowców wykorzystywanych do stworzenia produktu. Także w tym miejscu opisane są wszelkiego rodzaju czujniki i urządzenia wykonawcze. Dzięki zastosowaniu modelu RAMI dużych rozmiarów system jest możliwy do opisania prostszymi zależnościami.

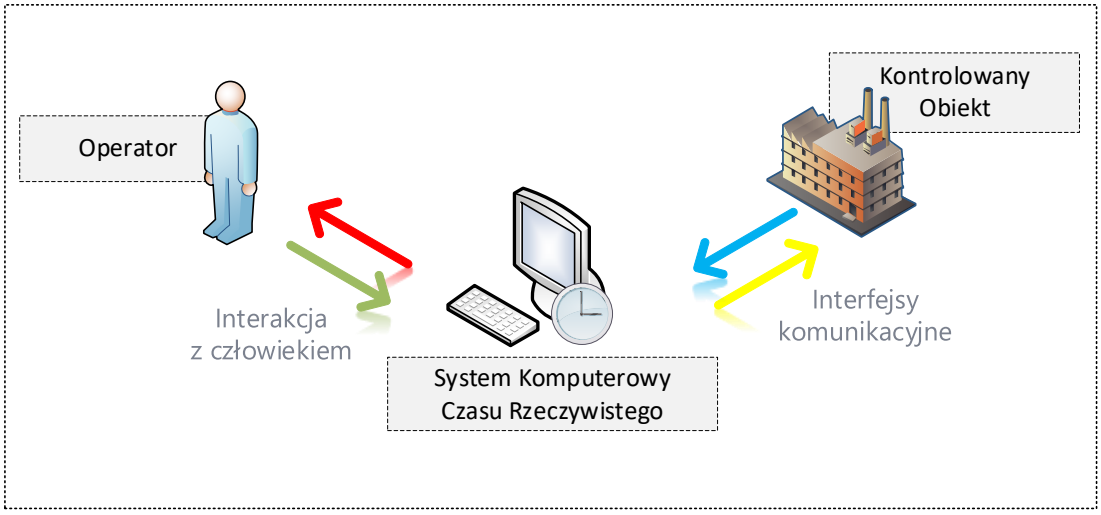
Oś cyklu życia i tworzenia wartości pozwala w zaprezentowaniu tworzenia produktu i wszystko co związane z cyklem życia np. urządzenia. W tej części określa się takie procesy tworzenia produktu takie jak: projektowanie czy utylizację. W całej piramidzie można uwzględnić również klienta końcowego wraz z jego wymaganiami, a całość produkcji może być spersonalizowana.

Siedmiowarstwowy model ISO/OSI stał się wzorcem do prezentacji w architekturze warstw i służy do obrazowania takich elementów jak: komponent, integracja, komunikacja, przetworzone dane, funkcjonalności oraz biznes.

Model Rami 4.0 pozwala w łatwy sposób rozłożyć system przemysłowy na prostsze w analizie składniki. Umiejscawiając sieci SDN w strukturze modelu RAMI 4.0 możemy umieścić je na osi hierarchii, gdzie zostaną opisane wykorzystywane standardy i możliwe połączenia pomiędzy elementami systemu.

2.2 Systemy czasu rzeczywistego

Mitch Albom w swojej książce “Zaklinacz czasu” używa sformułowania “Nigdy nie jest za późno ani za wcześnie. Jest dokładnie wtedy, kiedy trzeba”. Odnosząc ten fragment do informatyki i komputerowych systemów czasu rzeczywistego można by powiedzieć, że opisuje on idealny system przemysłowy *(Rysunek 3.4)*. Wszystkie zadania realizowane przez urządzenia są wykonywane poprawnie, zadania komunikacyjne spełniają rygorystyczne zasady determinizmu czasowego. Niestety w realnym świecie takie stwierdzenie nie zawsze jest prawdziwe, a same systemy przemysłowe możemy podzielić na kilka grup i podać zależności, od których zależy poprawna praca systemu. Poprawna tzn. realizująca wszystkie zaprogramowane funkcje w z góry określonym czasie.[6]–[8]

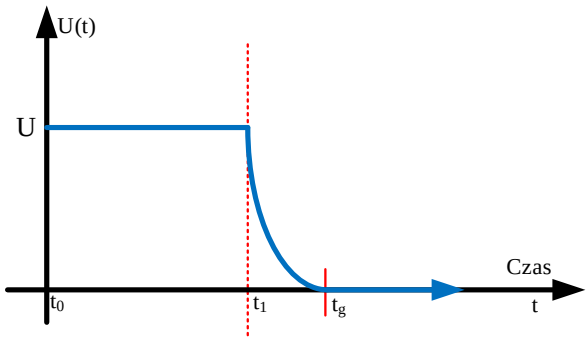


Rysunek 2.4 Podstawowe elementy składowe systemu przemysłowego

Systemy czasu rzeczywistego możemy podzielić ze względu na ograniczenia i skutki spowodowane niedotrzymanie terminu realizacji zadania. W zależności od rodzaju systemu przemysłowego możliwe są przekroczenia założonego czasu na wypracowanie odpowiedzi bądź takie przekroczenie, które spowoduje problem z działaniem systemu. W odniesieniu do tematyki poruszanej w pracy, czyli

komunikacji sieciowej przy użyciu sieci SDN, możemy określić czy wprowadzone opóźnienie czasowe w realizacji komunikacji będzie miało wpływ na jakość wygenerowanej informacji zwrotnej. Podział systemów ze względu na ograniczenia zaprezentowano na *Rysunkach 3.5. do 3.7.*

- Łagodne ograniczenia czasowe (*ang. Soft Real-Time*) – w tego typu systemach wypracowanie odpowiedzi zwrotnej nie spowoduje błędów w działaniu samego procesu lub skutki jego przekroczenia będą pomijalne. Podobnie bezpieczeństwo działania sterowania jest niezagrożone. W tego typu systemach działanie zaproponowanych urządzeń SDN nie będzie miało negatywnego wpływu na całość.

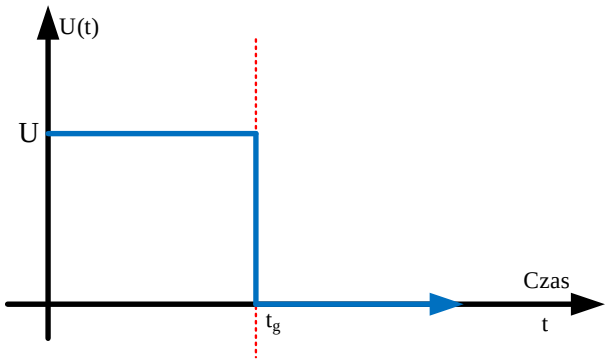


$$U(t) = \begin{cases} U & t_0 < t < t_1 \\ U * \frac{t_g - t}{t_g - t_1} & t_1 < t < t_g \\ 0 & t > t_g \end{cases}$$

Łagodne ograniczenia czasowe

Rysunek 2.5 Wykres przedstawiający działanie systemów czasu rzeczywistego z łagodnymi ograniczeniami czasowymi

- Mocne ograniczenia czasowe (*ang. Hard Real-Time*) – zagrożenie wprowadzane przez przekroczenie czasu wypracowania odpowiedzi są uzależnione od długości trwania tego przekroczenia. Im jest on dłuższy tym skutki dla systemu i osób pracujących z nim są bardziej niebezpieczne. Zastosowanie przełączników SDN w tego typu systemach jest jak najbardziej możliwe pod warunkiem ich dopasowania do typu sytemu i określenia możliwego maksymalnego czasu wprowadzonego opóźnienia.

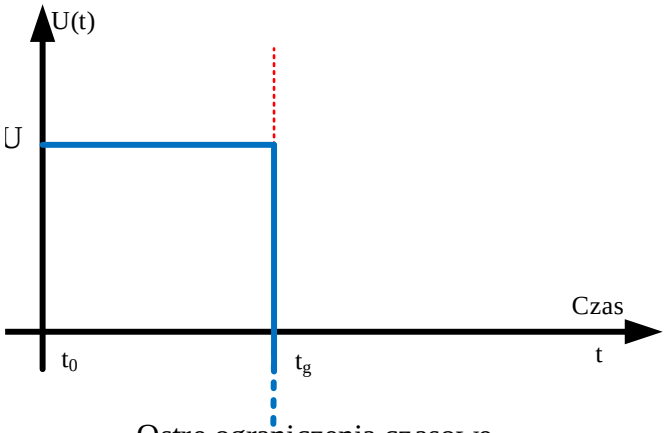


$$U(t) = \begin{cases} U & t_0 < t < t_g \\ 0 & t > t_g \end{cases}$$

Mocne ograniczenia czasowe

Rysunek 2.6 Wykres przedstawiający działanie systemów czasu rzeczywistego z mocnymi ograniczeniami czasowymi

- Ostre ograniczenia czasowe (*ang. Firm Real-Time*) – wypracowanie informacji zwrotnej w tego typu systemach po czasie dłuższym niż określony jako maksymalny powinno spowodować natychmiastowe zatrzymanie procesu sterowania. Wszystkie przekroczenia mogą skutkować zagrożeniem dla zdrowia i życia. Z tego powodu już na tym etapie można stwierdzić, że wprowadzane urządzenia SDN mogą nie mieć praktycznego zastosowania w tego typu systemach, wada ta jednak może zostać wyeliminowana wraz z rozwojem technologii i wprowadzeniem na rynek szybszych urządzeń (np. przełączników czy kontrolerów)



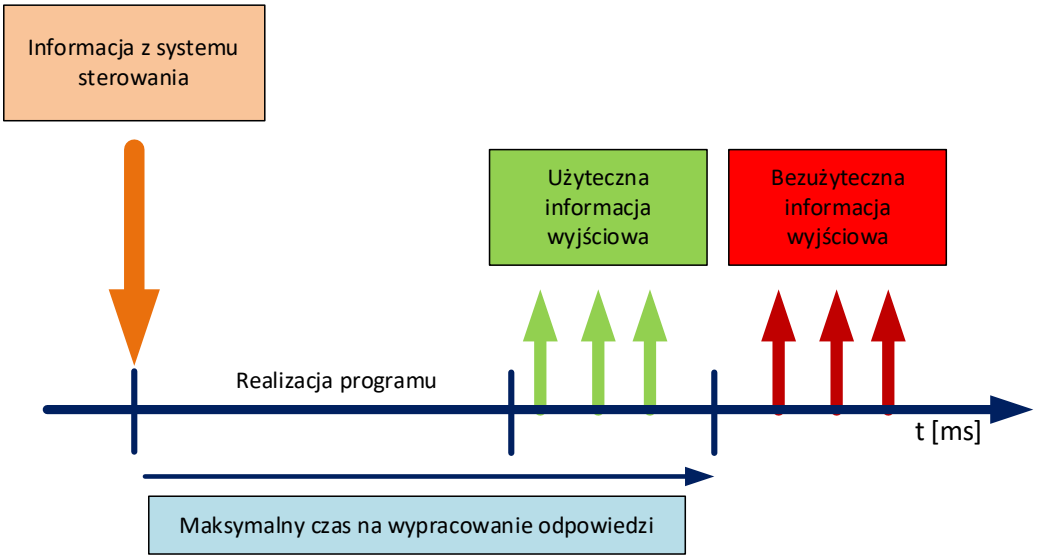
$$U(t) = \begin{cases} U & t_0 < t < t_g \\ -\infty & t > t_g \end{cases}$$

Ostre ograniczenia czasowe

Rysunek 2.7 Wykres przedstawiający działanie systemów czasu rzeczywistego z ostrymi ograniczeniami czasowymi

System przemysłowy (lub tak jak w przypadku opisanym na *Rysunku 3.8* sterownik PLC) otrzymuje informację ze środowiska, która jest przetwarzana

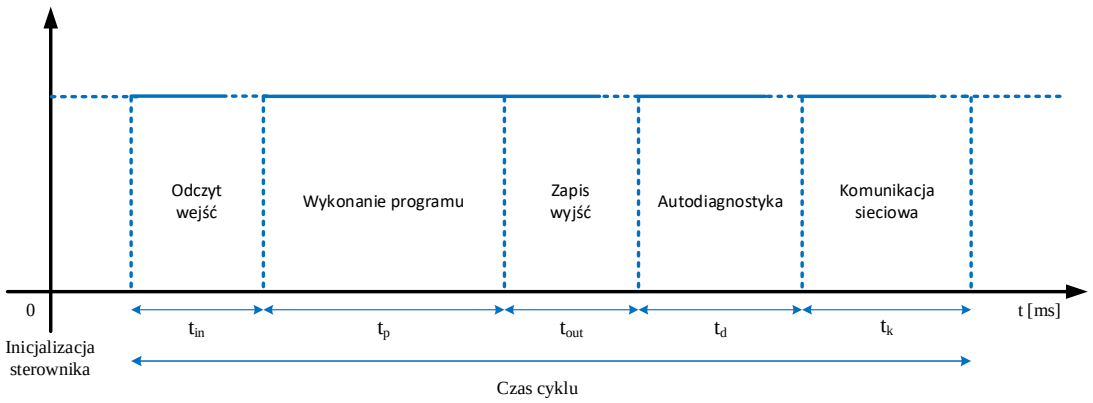
przez odpowiednie algorytmy i na jej podstawie wypracowana jest informacja zwrotna. Informacją zwrotną może być zarówno nowa wartość określonego parametru, jak również sygnał cyfrowy uruchamiający bądź zatrzymujący urządzenie.



Rysunek 2.8 Przebieg czasowy generowania odpowiedzi przez system sterowania

2.2.1 Sterownik PLC i zasada jego działania

Pracę sterownika swobodnie programowalnego (*ang. Programmable Logic Controller*) [9], [10] opisujemy jako cykl pracy *(Rysunek 3.9)*. Na początkowym etapie odczytujemy wejścia i tworzymy obraz wszystkich sygnałów. Na bazie odebranych sygnałów wejściowych realizowany jest program i przygotowywany jest obraz sygnałów wyjściowych, które zostaną ustawione w kolejnym kroku. Ostatnimi dwoma etapami jest autodiagnostyka urządzenia oraz realizacja procesów komunikacyjnych. To ten ostatni etap będzie miał znaczenie na działanie sterownika w sytuacji, gdy zostanie zastosowana infrastruktura sieciowa inna niż dedykowana. [11]



Rysunek 2.9 Cykl pracy sterownika PLC

2.3 Systemy zdecentralizowane

Duża liczba przedsiębiorców posiada swoje oddziały w różnych miejscach kraju, a nawet świata. Pozwala to na sprawniejsze zarządzanie i łatwiejszy dostęp do klienta na danym terenie. Podobnie jest z systemami przemysłowymi, jeśli wszystkie jego elementy umieścimy w jednym miejscu, dostarczane informacje niekoniecznie będą odpowiadać rzeczywistym wartościom. Z tego powodu budowane są systemy zdecentralizowane, czyli takie, gdzie poszczególne elementy systemu sterowania nie są zlokalizowane w jednym miejscu tylko rozproszone po całym obiekcie. Za wymianę informacji pomiędzy składowymi odpowiada z reguły sieć przemysłowa.

2.4 Sieci przemysłowe

Poprawne działanie systemu przemysłowego wymaga ciągłej wymiany informacji pomiędzy urządzeniami pracującymi w danej sieci. Wymiana informacji może następować zarówno na poziomie odczytu informacji z czujników czy odpowiedniego ustawienia wejścia, ale również na wykorzystaniu sieci komputerowej. Rozważając ten drugi przypadek należy wziąć pod uwagę specyficzny rodzaj sieci komputerowych jakimi są sieci przemysłowe.

2.5 Komunikacja pomiędzy urządzeniami sieci przemysłowej

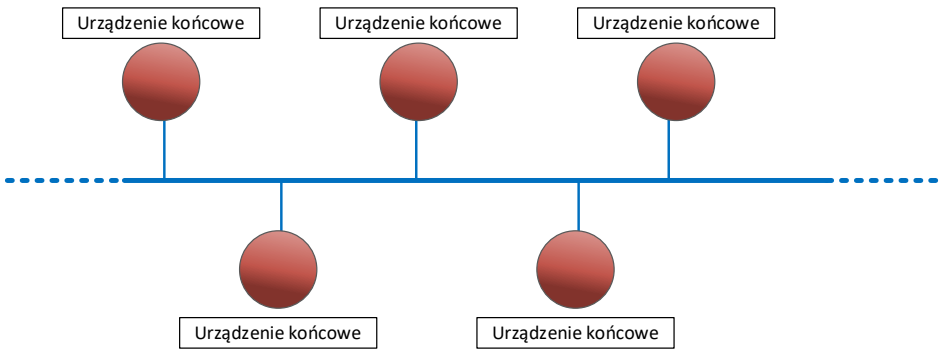
Wraz z biegiem czasu zmieniały się sposoby dostępu urządzeń do danych. Początkowe systemy zbierając dane jedynie w swoim najbliższym otoczeniu nie wymagały połączeń komunikacyjnych pomiędzy dwoma jednostkami sterowania. Rozwój systemów sterowania wymusił na producentach i działających w branży firmach opracowanie rozwiązań pozwalających na komunikowanie się dwóch różnych urządzeń za pomocą sieci komunikacyjnej, a nie modułów wejść/wyjść i sygnałów cyfrowo-analogowych. Pierwszymi wprowadzonymi sieciami były sieci polowe tj. CAN, Profibus, Modbus, które w późniejszym czasie coraz częściej były wypierane przez sieci oparte o rozwiązanie Ethernet. Aktualnie prócz coraz to bardziej zaawansowanych sieci budowanych na bazie klasycznego Ethernet'u w nowoczesnych systemach przemysłowych możemy spotkać również komunikację bazująca na sieciach bezprzewodowych, a nawet sieciach krótkiego zasięgu tj. Bluetooth, ZigBee.[12]–[16]

2.6 Sieć przemysłowa – topologia

Opisując komunikację sieciową nie można pominąć tak ważnego określenia jakim jest „topologia"[17]–[19]. Pojęcie to opisuje jak wygląda projektowana bądź użytkowana przez nas sieć. W kontekście sieci komputerowych możemy dokonać podziału na dwa rodzaje topologię logiczną i fizyczną. Topologia fizyczna opisuje rodzaj zastosowanych urządzeń, mediów transmisyjnych, a także ilość hostów.

Także tutaj (w opisie topologii fizycznej) znajdziemy informacje w jaki sposób komputer jest w stanie uzyskać dostęp do łącza. Najpopularniejsze topologie fizyczne zostały opisane poniżej. Wprowadzenie pojęcia topologii sieciowej w tym miejscu wymaga nawiązania do tematyki poruszonej w pracy. Ze względu na fakt, że w większości nowoczesnych przypadków topologia sieciowa straciła na znaczeniu, zaprezentowane topologie odniesiono do możliwości jej zaimplementowania z wykorzystaniem sieci definiowanych programowo (SDN).

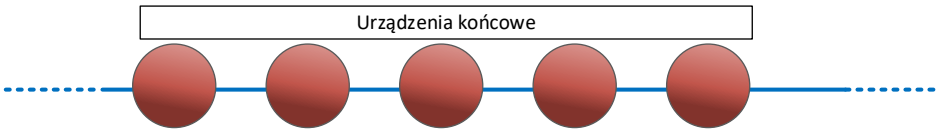
Topologia magistralowa – elementy sieci podłączone są do jednego przewodu (magistrali). Najczęściej wykorzystywanym medium transmisyjnym jest kabel koncentryczny, z tego powodu należy zastosować terminator na każdym końcu magistrali. Najczęściej takim terminatorem jest opornik o rezystancji dopasowanej do impedancji falowej kabla użytego do połączenia. Brak takiego zabezpieczenia, może spowodować „odbicie" sygnału i zajęcie całej linii komunikacyjnej poprzez przesyłanie sygnału w przeciwnym kierunku. Przykładem sieci pracującej w tej topologii jest sieć np. NMEA 2000, wykorzystywana głównie do połączenia urządzeń elektronicznych w jednostkach pływających. *(Rysunek 3.10)*



Rysunek 2.10 Schemat przedstawiający połączenie w topologii magistrali

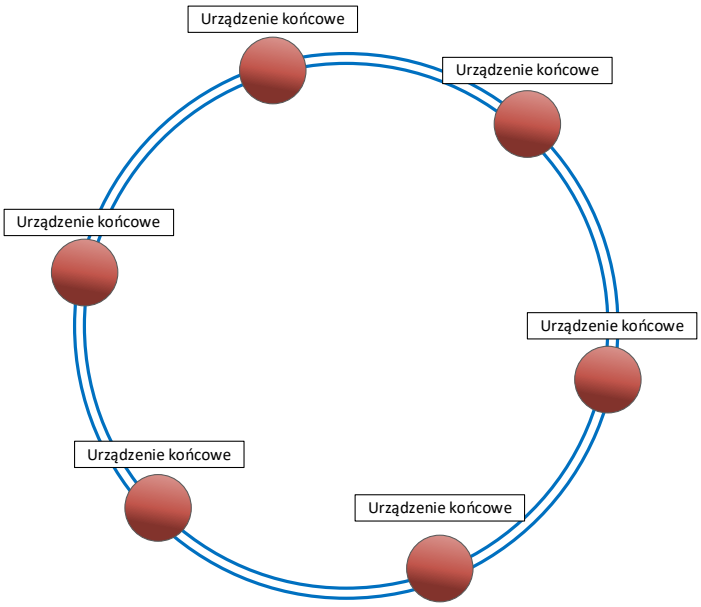
Topologia liniowa – jedna z najprostszych w realizacji topologii. Polega na połączeniu każdego kolejnego urządzenia w sieci do urządzenia poprzedzającego i urządzenia następnego. Topologia ta wymaga, aby urządzenia

były wyposażone w dwa interfejsy sieciowe. Do wad takiego rozwiązania można zaliczyć wymóg ciągłej pracy urządzeń oraz to, że przerwanie łącza pomiędzy dwoma urządzeniami spowoduje brak komunikacji ze wszystkimi urządzeniami umiejscowionymi za miejscem uszkodzenia. Przykładem sieci w topologii liniowej może być połączenie sterownika PLC z wyspami sygnałowymi lub napędami. *(Rysunek 3.11)*



Rysunek 2.11 Schemat przedstawiający połączenie w topologii liniowej

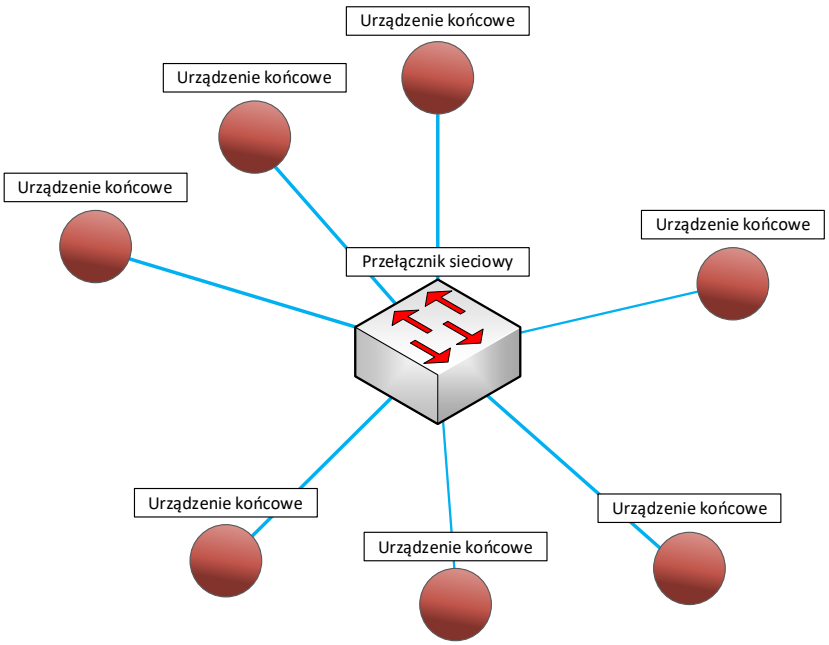
Topologia pierścieniowa – ten rodzaj połączeń można porównać do sieci pracującej w topologii liniowej, gdzie punkt początkowy i końcowy są dodatkowo połączone ze sobą. Tak jak w topologii liniowej tak samo i tutaj każdy kolejny komputer pełni rolę wzmacniacza sygnału. Sieci pierścieniowe bardzo często korzystając z dostępu do łącza za pomocy metody przekazywania „żetonu”. *(Rysunek 3.12)*



Rysunek 2.12 Schemat przedstawiający połączenie w topologii pierścieniowej

Topologia typu gwiazda – obecnie najczęściej spotykany rodzaj sieci. Wszystkie urządzenia w sieci podłączone są do urządzenia centralnego tzw. punktu dostępowego (ang. Access Point). Ze względu na zastosowanie osobnego przewodu do każdego z komputerów, sieci budowane w tej topologii charakteryzują się większą bezawaryjnością, w praktyce tylko uszkodzenie punktu dostępowego spowoduje problem z pracą całej sieci. Ze względu na coraz większą ilość wykorzystywanych urządzeń punktem dostępowym najczęściej jest przełącznik sieciowy z odpowiednią liczbą portów. Przykładem tego rodzaju sieci

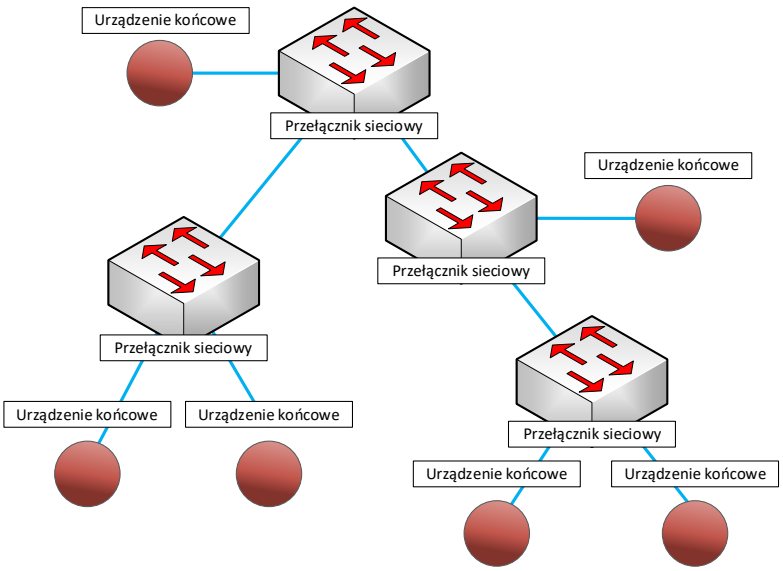
może być sieć domowa, gdzie do jednego punktu podłączone są z reguły wszystkie urządzenia. *(Rysunek 3.13)*



Rysunek 2.13 Schemat przedstawiający połączenie w topologii gwiazdy

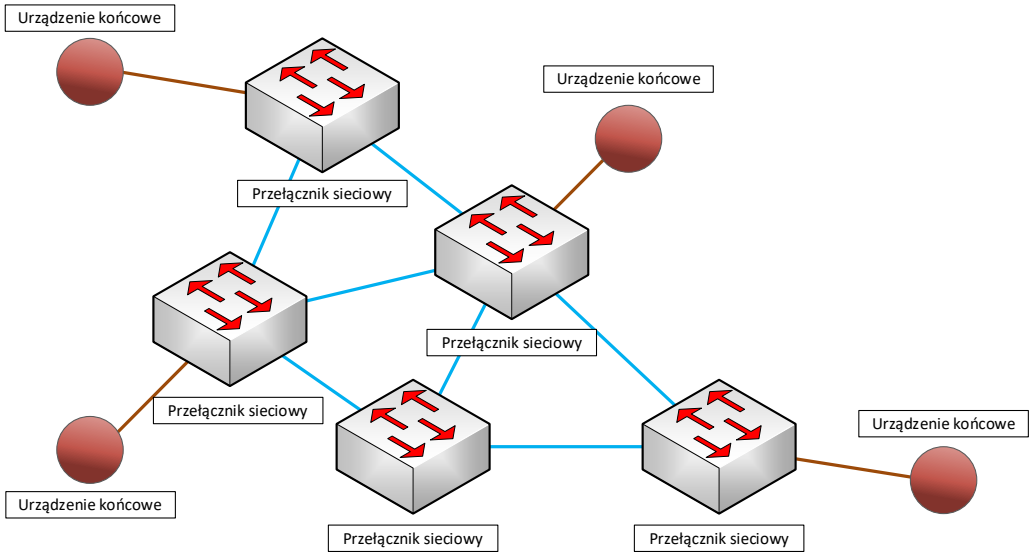
Topologia typu drzewo – topologia będąca połączeniem innych topologii: magistrali i gwiazdy. Zastosowanie takiej topologii zapewnia zdecydowanie najłatwiejszą łatwość rozbudowy. Najczęściej pojawia się w

sytuacjach rozbudowy istniejącego systemu, ze względu na brak potrzeby budowania sieci od podstaw. (*Rysunek 3.14)*.



Rysunek 2.14 Schemat przedstawiający połączenie w topologii typu drzewo

Topologia typu siatka – topologia będąca rozwinięciem topologii gwiazdy. Zastosowanie takiej topologii zapewnia zwiększone bezpieczeństwo, jeśli chodzi niezawodność połączenia. Podobnie jak w topologii gwiazdy także tutaj mamy do czynienia z łatwością rozbudowy infrastruktury sieciowej – (*Rysunek 3.15)*.



*Rysunek 2.15 Schemat przedstawiający połączenie w topologii typu siatka (*ang. MESH*)*

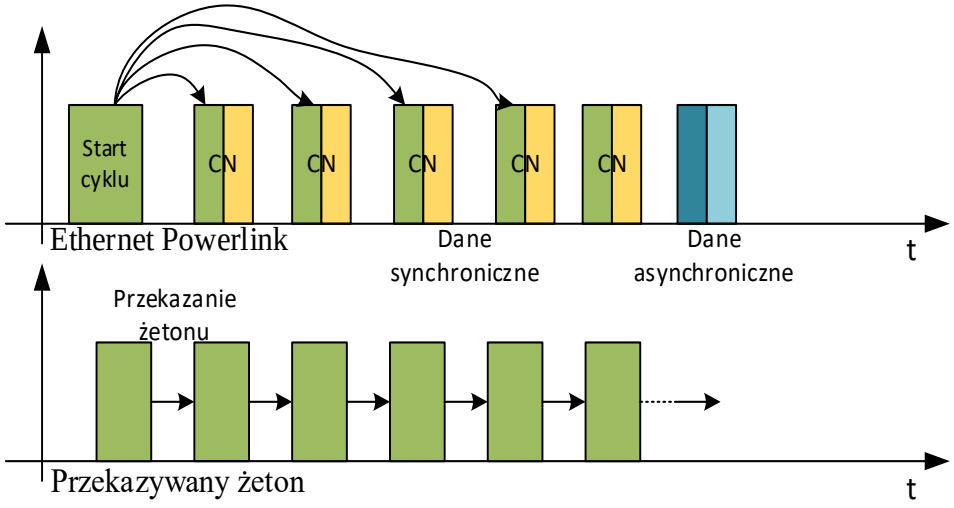
Warto zwrócić uwagę, że oficjalne opisy topologii powstawały kilkadziesiąt lat temu i wiele z ich wad udało się już wyeliminować czy to w sposób sprzętowy czy przy pomocy odpowiedniego oprogramowania.

Wykorzystując jedynie opis topologii fizycznej nie zawsze, a w praktyce nigdy nie jesteśmy w stanie poprawnie opisać działania sieci i uwzględnić wszystkich wymaganych parametrów. W tej kwestii z pomocą przychodzi opis topologii logicznej. W tej części można wyróżnić dwa rodzaje topologii logicznych:

- typu przekazywany żeton (ang. *token*)
- typu rozgłoszeniowego (ang. *broadcast*).

W topologii przekazywania żetonu w sieci krąży specjalna ramka „żeton”, a host aktualnie posiadający tę ramkę znacznik ma możliwość korzystania z łącza w danym momencie. W sytuacji, kiedy dany host posiada żeton, lecz nie ma zaplanowanych zadań komunikacyjnych zrealizowane jest przekazanie „żetonu” kolejnemu urządzeniu. Przykładem takiego działania sieci może być Ethernet Powerlink *(Rysunek 2.16)*. Mimo, że w sieciach przemysłowych nie spotkamy klasycznego „żetonu” to udostępnianie zezwolenia na transmisję można porównać

do właśnie takiego sposobu dostępu do łącza. Główny węzeł sieci MN (ang. *Manage Node*) zezwala na transmisję kolejnym węzłom CN (ang. *Controlled Node*), po czym realizowana jest transmisja asynchroniczna. [20], [21]



Rysunek 2.16 Porównanie realizacji wymian danych w sieci Powerlink oraz wymian z wykorzystaniem wymiany żetonu.

Zastosowanie sieci definiowanej programowo w takim połączeniu, może pozwolić na dynamiczne aktywowanie kolejnych węzłów sieci Powerlink. Przykładem takiej realizacji może być skonfigurowanie jednego urządzenia i modyfikowanie informacji o realizowanym połączeniu bezpośrednio w przełączniku sieciowym.

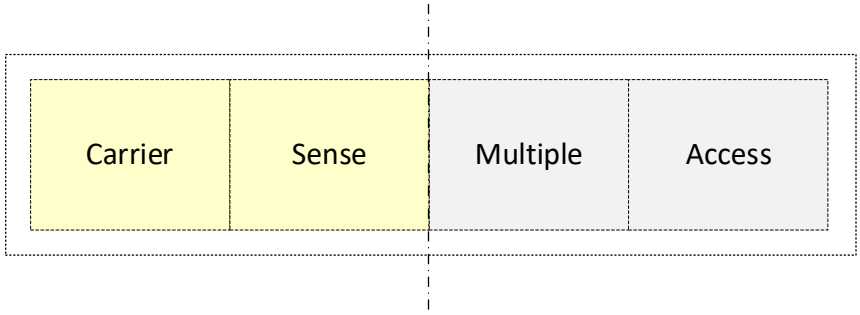
Inaczej sytuacja wygląda w logicznej topologii rozgłoszeniowej. W tym przypadku nadawca wysyła informacje do wszystkich urządzeń w sieci w dowolnym momencie, urządzenie, do którego dane były adresowane jest w stanie je odebrać. Niestety możliwość nadawania przez wszystkie urządzenia w dowolnym momencie może powodować kolizje, przez co transmisja staje się bezużyteczna.

Wraz z rozwojem technologii do sieci wprowadzanych zostało coraz to więcej urządzeń będących nadajnikami i odbiornikami. W związku z czym zaczęły pojawiać się problemy z przesłaniem informacji poprzez medium transmisyjne. Jednym z zaproponowanych rozwiązań jest sposób dostępu do łącza transmisyjnego CSMA (ang. *Carrier Sense Multiple Access*). [22] W dosłownym

tłumaczeniu oznacza to nasłuchiwanie nośnej i wielokrotny dostęp do łącza. *(Rysunek 2.17)*.

Część *„Carrier Sense”* oznacza, że nadajnik nasłuchuje ruchu w łączu, czyli sygnału przesyłanego z innego nadajnika w sięci. Dzięki takiemu działaniu możemy wykryć zajętość łącza, przez co nasz nadajnik musi oczekiwać na zezwolenie na nadawanie. Carrier Sense część ta odpowiedzialna jest za nasłuchiwanie ruchu w sieci i detekcja kolizji. Dzięki takiemu mechanizmowi możliwe jest wykrycie zajętości medium i uniknięcie kolizji w sieci.

Część *„Multiple Access”* oznacza, że z łącza/medium może korzystać kilka urządzeń, a informacja przesyłana przez jednego z nich dostępna jest do wszystkich urządzeń. Multiple Access ta część mówi użytkownikowi, że możliwy jest dostęp do łącza przez wiele urządzeń. Informacja z jednego urządzenia nie trafia tylko do jednego odbiorcy tylko może być odebrana przez wszystkie urządzenia w sieci.



Rysunek 2.17 Opis protokołu wielodostępowego CSMA

Posiadając informację, że z łącza może korzystać kilka urządzeń, a łącze jest w praktyce cały czas monitorowane zaproponowane zostały dwa protokoły pozwalające na zarządzanie współdzieleniem łącza:

CD (ang. *Collision Detection*) W tej sytuacji w medium transmisyjne jest wysyłany sygnał zakłócający pozwalający na wykrycie kolizji, które też są nasłuchiwane w sieci. W momencie „kolizji” z sygnałem z innego urządzenia transmisja nie jest kontynuowana. Typ z wykrywaniem kolizji jest stosowany w bardzo prostych sieciach typu Ethernet.

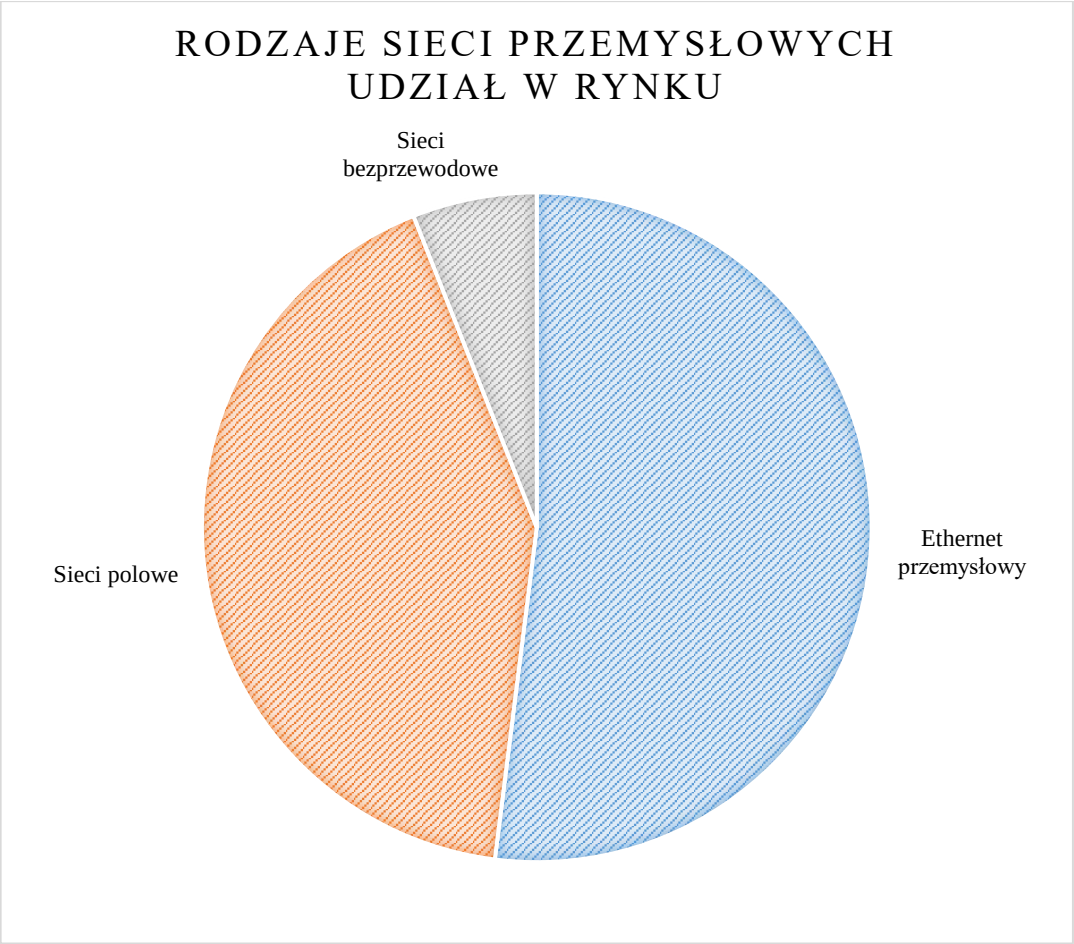
CA (ang. *Collision Avoidance*) unikanie kolizji. To rozwiązanie może powodować straty czasowe, ze względu na próbowanie możliwości dostępu do

linii. Wysłany sygnał próbny tzw. pilot sprawdza wystąpienie kolizji i jeśli takiej kolizji nie ma uzyskujemy zgodę na nadawanie. Mechanizm ten jest często wykorzystywany w komunikacji bezprzewodowej, w obsłudze punktu dostępowego.

Obecnie większość algorytmów działania sieci jest zrealizowana w przełączniku sieciowym i powyższe mechanizmy nie mają praktycznego zastosowania. Powyższa analiza ma na celu zobrazowanie problemów występujących w sieciach i możliwości ich eliminacji.

2.7 Sieci przemysłowe - protokoły

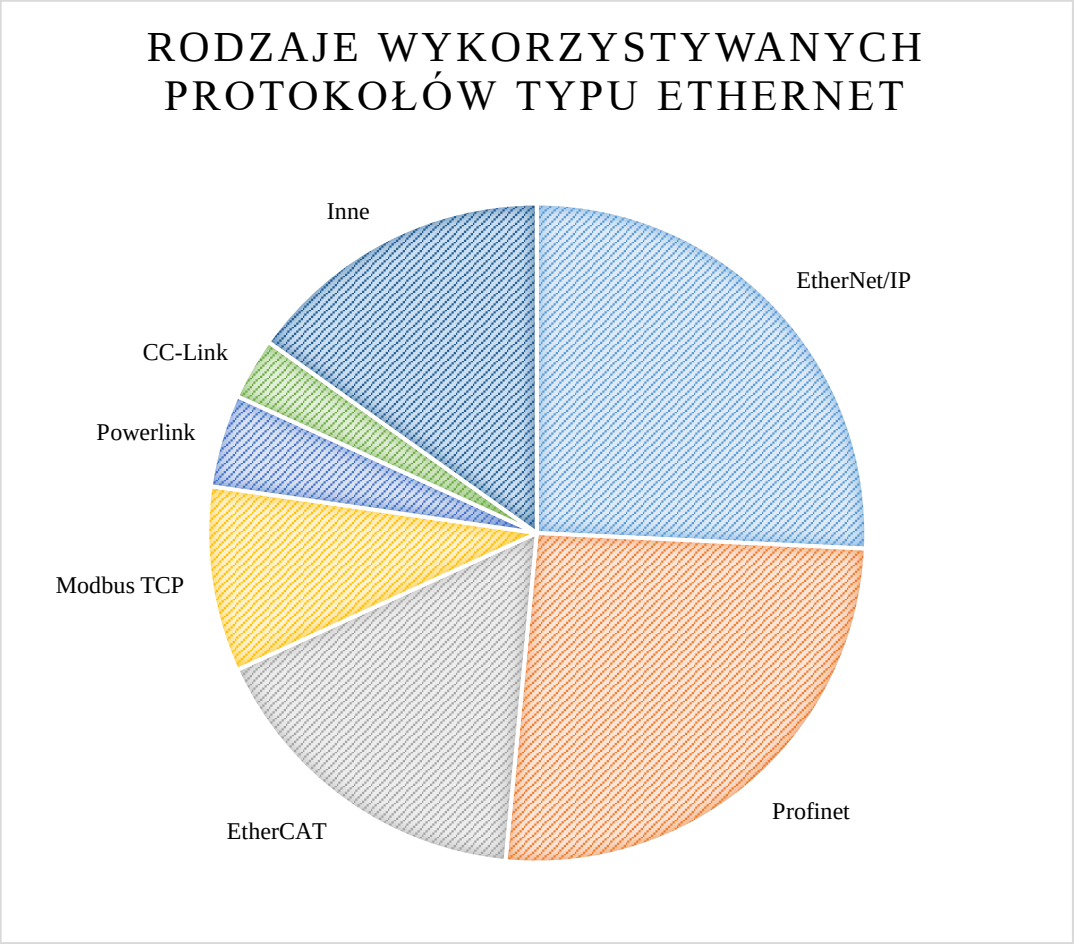
Praktycznie żadna sieć komputerowa nie będzie w stanie działać, jeśli nie określimy w jaki sposób będziemy przekazywać informacje pomiędzy urządzeniami. Opis działania konkretnej sieci zazwyczaj definiuje protokół, który wykorzystywany jest do realizacji połączeń w systemie. Wiele obecnie stosowanych rozwiązań (protokołów) stanowi migrację sieci polowych oraz zasad ich działania do warstwy ethernetowej. Powszechnie stosowane kiedyś protokoły takie jak Profibus DP czy Modbus RTU [23], [24]ustępują miejsca swoim nowocześniejszym następcom protokołom Profinet oraz Modbus TCP. Nie oznacza to jednak, że udział w rynku sieci polowych jest znikomy i praktycznie nie spotkamy ich użycia w dzisiejszych instalacjach przemysłowych. Mimo, że infrastruktura sieciowa oparta o połączenie typu Ethernet to dzisiaj ponad 50% wszystkich przypadków (co zostało przedstawione na *Rysunku 2.18*), to połączenia takie jak Profibus DP ciągle są wspierane i urządzenia kontrolujące są częścią nawet najnowszych jednostek sterujących, bądź to jako integralną część jednostki, bądź jako moduł dodatkowy.



Rysunek 2.18 Wykres prezentujący udział poszczególnych sieci w komunikacji przemysłowej

W dalszej części omówiono najpopularniejsze protokoły sieciowe, zarówno wersje oparte o sieci Ethernet, jak i ich starsze wersje polowe. Część ta obrazuje postęp, który się dokonał w komunikacji sieciowej i szanse dla rozwiązań, które niekoniecznie są projektowane bezpośrednio do komunikacji pomiędzy urządzeniami w sieci przemysłowej. Sytuacja taka miała miejsce przy sieciach typu Ethernet, które zostały zaadaptowane do środowiska przemysłowego wraz z rozwojem urządzeń sieciowych, a dziś stały się podstawowym elementem komunikacji pomiędzy urządzeniami w systemie sterowania . Na *Rysunku 2.19* przedstawiono udział w rynku poszczególnych rozwiązań. Niektóre z nich zależne są od regionu użytkowania jak np. Ethernet/IP, którego popularność jest większa w Ameryce Północnej czy Profinet zdeycodowanie popularniejszy w Europie. Konkretne protokoły sieciowe promowane są również przez poszczególnych producentów, a więc popularność rozwiązania sieciowego jest ściśle związana z

popularnością danego producenta. Wyjątkiem może być tutaj protokół Modbus TCP, który wspierany jest w praktyce przez wszystkich producentów. Niestety szybkość działania nie zawsze jest do zaakceptowania przez wymagania stawiane systemom sterowania.



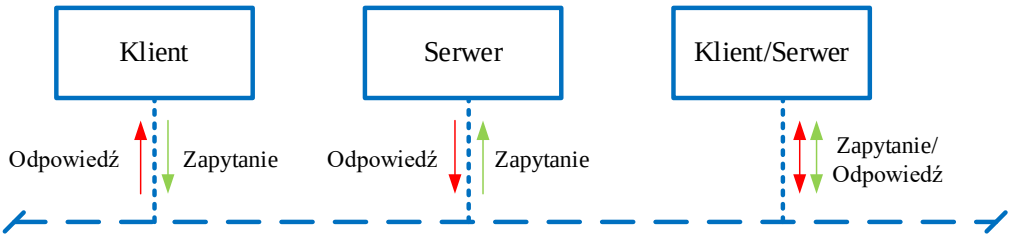
Rysunek 2.19 Wykres prezentujący udział poszczególnych protokołów w komunikacji przemysłowej z wykorzystaniem sieci Ethernet

Ze względu na uwarunkowania pracy z zachowaniem zasad determinizmu czasowego w sieciach przemysłowych wymagane jest zastosowanie odpowiednich reguł dostępu do łącza. Najpowszechniejszymi stosowanymi obecnie rozwiązaniami są:

- Sieci Master-Slave
- Sieci z przekazywanie żetonu - Token Ring

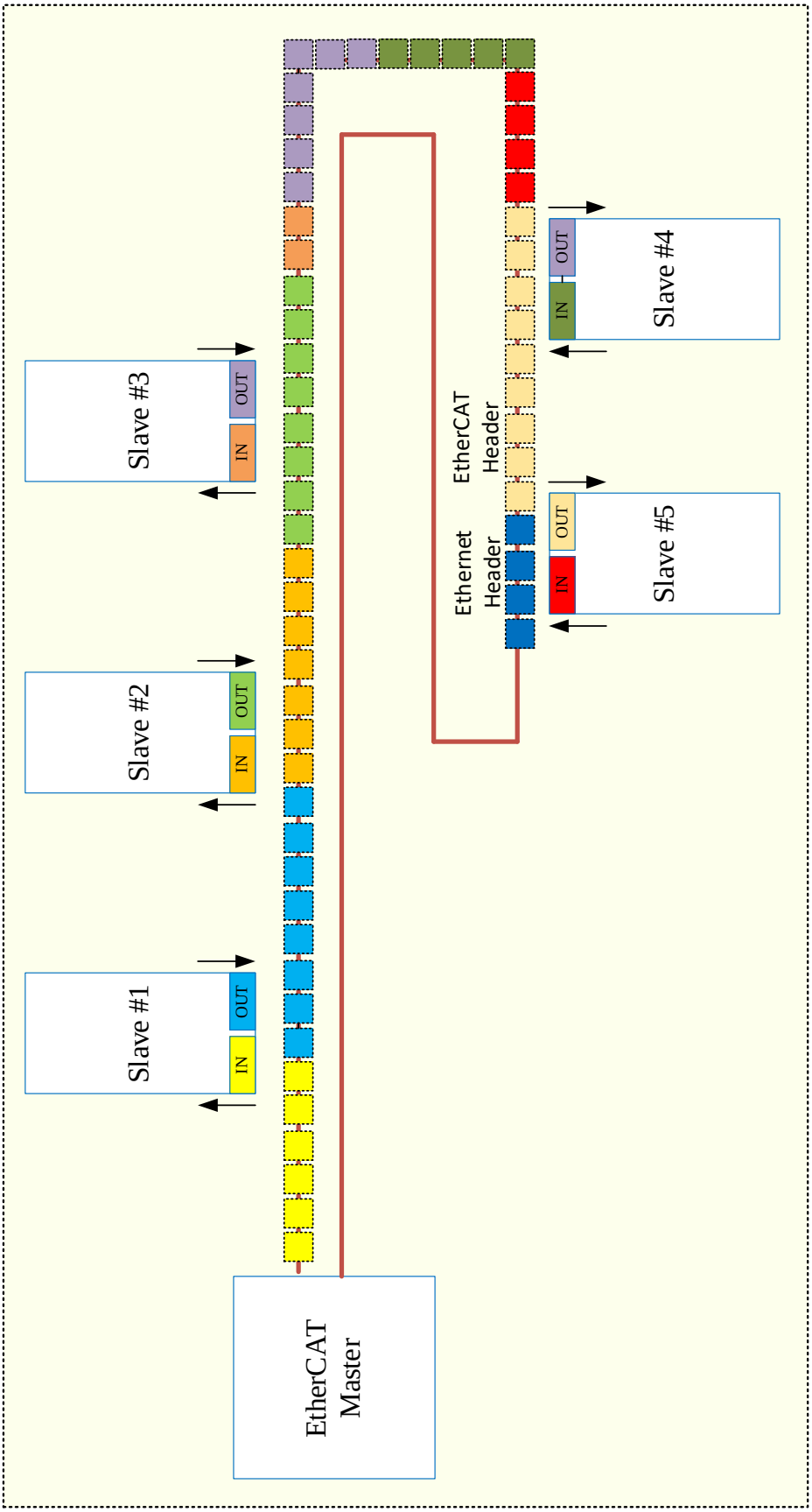
W szczególnych przypadkach można znaleźć protokoły wykorzystujące metodę CSMA/CD (ang. *Carrier Sense Multiple Access with Collision Detection*)

a) **Sieć Master-Slave** - metoda przydzielania dostępu do łącza wykorzystywana zarówno w klasycznych sieciach typu "*fieldbus*", jak i w rozwiązaniach opartych o sieć Ethernet. Rozwiązanie to wykorzystuje dwa rodzaje urządzeń: urządzenie nadzorcze Master oraz urządzenie podrzędne Slave. Najczęściej urządzenie typu Master inicjuje połączenie oraz dokonuje synchronizacji z wszystkimi urządzeniami w sieci. Urządzenia Slave nie mogą same rozpocząć nadawania, mogą jedynie odpowiadać na zapytania stacji nadrzędnej. Jednym z przykładów tego typu transmisji może być protokół przemysłowy Modbus. Niezależnie od wersji stacja nadzorcza przesyła adres docelowy, dane użyteczne i sumę kontrolną. Dane te są przesyłane do wszystkich urządzeń jednak odebrać je może jedynie adresat, czyli docelowa stacja podrzędna.



Rysunek 2.20 Zasada działania protokołu Modbus TCP

Innymi przykładem sieci typu Master-Slave może być infrastruktura oparta o rozwiązanie EtherCAT.[25] Sieć składa się z urządzenia typu Master wysyłającego ramki do wszystkich urządzeń w sieci, które dzięki możliwości przetwarzania jej zawartości w locie odbierają tylko dane dla siebie użyteczne. Zasada działania przesyłania informacji została zaprezentowana na *Rysunku 3.21*.



Rysunek 2.21 Zasada działania sieci EtherCAT – wymiana danych w urządzeniach typu Slave

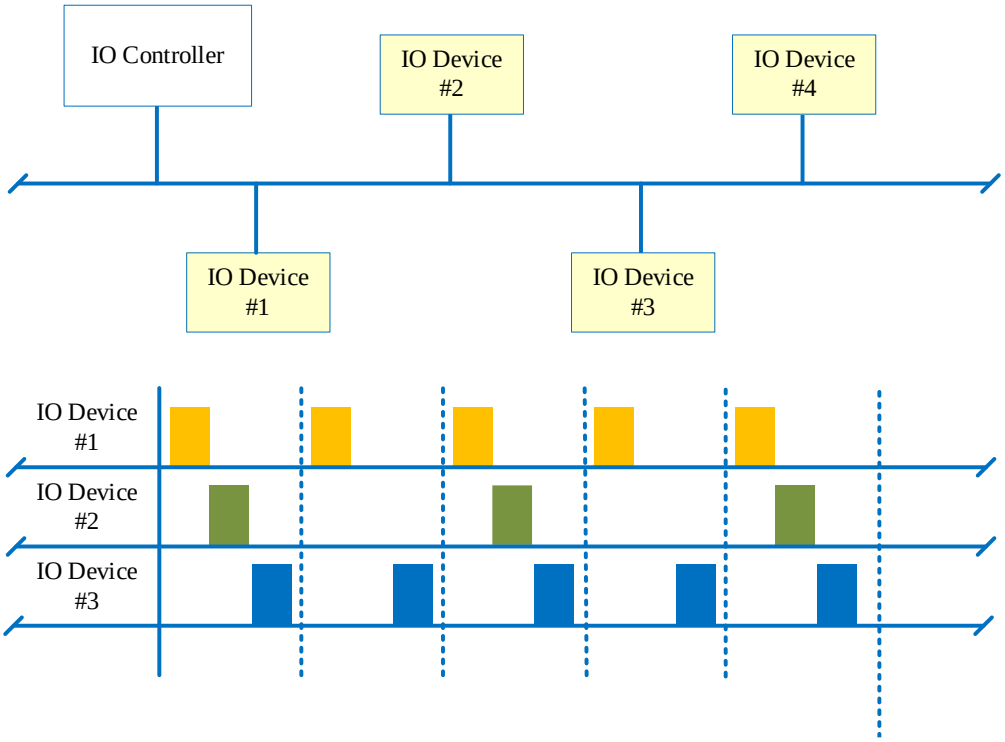
Już w tym miejscu należy zwrócić uwagę na znaczącą różnicę sieci typu EtherCAT względem typowych sieci Ethernet. Pozbawienie warstw od trzeciej do szóstej powoduje, że szybkość transmisji jest wyższa w porównaniu do tradycyjnej komunikacji Ethernet, jednak zarządzanie infrastrukturą sieciową jest trudniejsze ze względu na wymóg korzystania z dedykowanych urządzeń sieciowych. Analizując schemat przedstawiony na *Rysunku 2.22* można zauważyć podobieństwo względem klasycznego rozwiązania na poziomie warstwy drugiej – adresów MAC. W dalszej części pracy skupiono się głównie na danych typu RealTime.

<table>
<tr><th>Model ISO</th><th colspan="2">EtherCAT</th></tr>
<tr><td>Warstwa aplikacji</td><td>Dane podstawowe</td><td>Dane Real Time</td></tr>
<tr><td>Warstwa prezentacji</td><td rowspan="2">↕</td><td rowspan="4">↕</td></tr>
<tr><td>Warstwa sesji</td></tr>
<tr><td>Warstwa transportowa</td><td>TCP, UDP</td></tr>
<tr><td>Warstwa sieciowa</td><td>IP</td></tr>
<tr><td>Warstwa łącza danych</td><td colspan="2">EtherCAT MAC adres</td></tr>
<tr><td>Warstwa fizyczna</td><td colspan="2">Warstwa fizyczna EtherCAT</td></tr>
</table>

Rysunek 2.22 Porównanie warstw w klasycznym modelu ISO oraz sieci EtherCAT

Także w sieci PROFINET IO do dyspozycji mamy urządzenia Master oraz Slave, nazywane odpowiednio *IO Controller* oraz *IO Device (Rysunek 2.23)*. Urządzenia IO Controller dynamicznie zarządzają pracą sieci. Ze względu na konieczność wymieniania dużej ilości danych przez IO Controller, urządzenia te są z reguły najbardziej obciążone. Urządzenia IO Device, którymi mogą być również dedykowanie przełączniki sieciowe są zależne od urządzenia typu Master. Dzięki urządzeniu IO Controller możliwe jest realizowanie komunikacji według dedykowanych scenariuszy, z różnymi parametrami dla każdego z urządzeń. Przełączniki sieciowe działając w systemie „Przechowaj i przekaż” (ang. *Store-*

Forward) monitorują w praktyce każdą transmitowaną ramkę co pozwala na natychmiastowe wykluczanie ramek błędnych lub uszkodzonych. System działania Store-Forward polega na odebraniu wszystkich ramek i dopiero po sprawdzeniu ich poprawności przesyła je do kolejnych urządzeń. Taki system działania jest zwykle realizowany w urządzeniach sieciowych tj. routery czy przełączniki sieciowe.



Rysunek 2.23 Schemat działania sieci Profinet IO

b) **Sieć z przekazywaniem żetonu** – rozwiązanie polega na przekazywaniu żetonu pomiędzy urządzeniami i zezwolenie na nadawanie urządzeniu, które posiada w danym momencie żeton. Jeśli dane urządzenie zakończy całą swoją transmisję żeton przekazywany jest do kolejnej stacji. Często spotyka się sieci z przekazywaniem żetonu wraz z topologią pierścieniową, w której ostatnie urządzenie przekazuje żeton do urządzenia pierwszego.

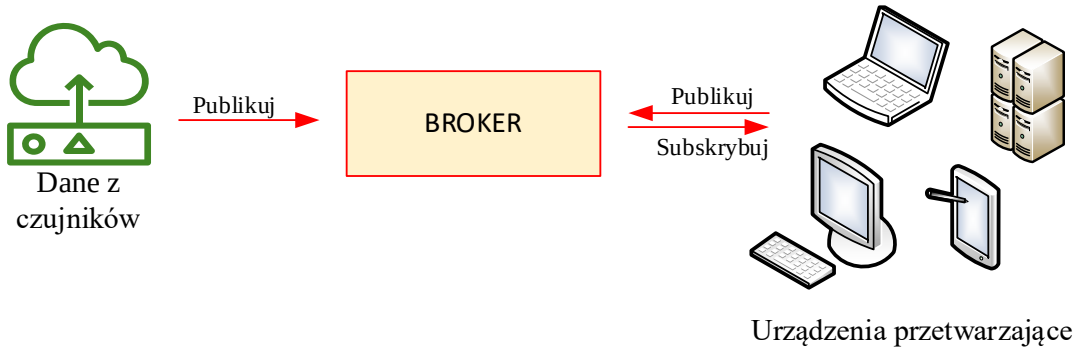
Specyficznym połączeniem sieci z przekazywaniem żetonu jest sieć Profibus DP. Podstawą działania sieci są urządzenia Master-Slave, pomiędzy, którymi przekazywany jest żeton. Żeton może być wymieniany tylko pomiędzy dwoma urządzeniami typu Master, oznacza to, że rozpoczęcie odpytywania kolejnych stacji Slave możliwe jest jedynie przez aktywną stację Master.

Posiadanie żetonu pozwala na rozpoczęcie transmisji (odpytywania stacji Slave). Żeton krążąc w sieci gwarantuje, że w danym momencie nadawać może tylko jeden z abonentów. Ze względu na liczbę nowocześniejszych rozwiązań Token Ring jest stosowany coraz rzadziej, a jego miejsce zastępują mechanizmy implementowane bezpośrednio w przełącznikach sieciowych.

Mimo, że rozwiązania tego typu są stosowane coraz rzadziej możliwe jest wykorzystanie tego typu mechanizmów do implementacji algorytmów w kontrolerze sieci definiowanej programowo.

Ze względu na sposób działania algorytmów sieciowych wiele rozwiązań zostało przeniesionych z klasycznych sieci *fieldbus* do warstwy Ethernetu, a później również do infrastruktury bezprzewodowej. Rozwój sieci bezprzewodowych pozwala na ich powolne wdrażanie i wykorzystanie w systemach czasu rzeczywistego np. w miejscach, gdzie doprowadzenie infrastruktury przewodowej jest czasochłonne, a może nawet niemożliwe.

Rozwiązania bezprzewodowe są również częściej wykorzystywane, ze względu na integrację systemów sterowania z Przemysłowym Internetem Rzeczy *(ang. Industrial Internet of Things)*. Jednym z powszechniejszych protokołów jest MQTT *(ang. MQ Telemetry Transport)*. Zasada działania opiera się na pracy w sieci brokera, do którego podłączają się klienci i została zaprezentowana na *Rysunku 2.24*. Klienci mogą w ten sposób publikować w zadanym temacie bądź subskrybować informację publikowane przez innych klientów. Protokół ten jest wykorzystywany w połączeniach bezpośrednich M2M (ang. *Machine To Machine*). Ze względu prostotę działania protokół ten wykorzystywany może być również w sterownikach PLC czy urządzeniach wbudowanych.



Rysunek 2.24 Schemat sieci pracującej z protokołem MQTT

Protokół MQTT jest protokołem transmisji danych, pracującym w modelu publikacja/subskrypcja. Protokół pracuje na górnej warstwie TCP (ang. *Transmision Control Protocol*) / IP (ang. *Internet Protocol*). Takie rozwiązanie pozwala na szybkie wymiany danych z takimi elementami jak np. czujniki temperatury.z tego powodu w łatwy sposób mogą być realizowane funkcje w aplikacjach IoT (*ang. Internet of Things*), czy M2M (*ang. Machine to Machine*)

Zaletami takiego sposobu komunikacji są:

- niskie obciążenie łącza komunikacyjnego,
- szyfrowanie i zabezpieczenie komunikacji.
- szybkość konfiguracji,
- wymiana informacji, odbywająca się w niemal czasie rzeczywistym,

Do wad możemy zaliczyć:

- na dziś istnieje stosunkowo niewiele urządzeń obsługujących MQTT
- konieczność uruchomienia brokera.

Do najważniejszych parametrów wyróżniających protokół MQTT należy zaliczyć:

- Szybkość transmisji do 1Gbps,
- Liczba urządzeń podłączonych do jednej sieci zależy przede wszystkich od brokera i może ich być ponad 1000,
- Jako medium transmisyjne wykorzystywany jest Ethernet/WiFi

- Do zabezpieczenia transmisji możliwe jest wykorzystanie certyfikatów, a z sama transmisja z brokerem jest zabezpieczona poprzez szyfrowanie połączenia SSL. Samo połączenie może być zabezpieczone wymogiem zalogowania się urządzenia do brokera poprzez nazwę użytkownika i hasło.

3 Przemysłowy Internet Rzeczy

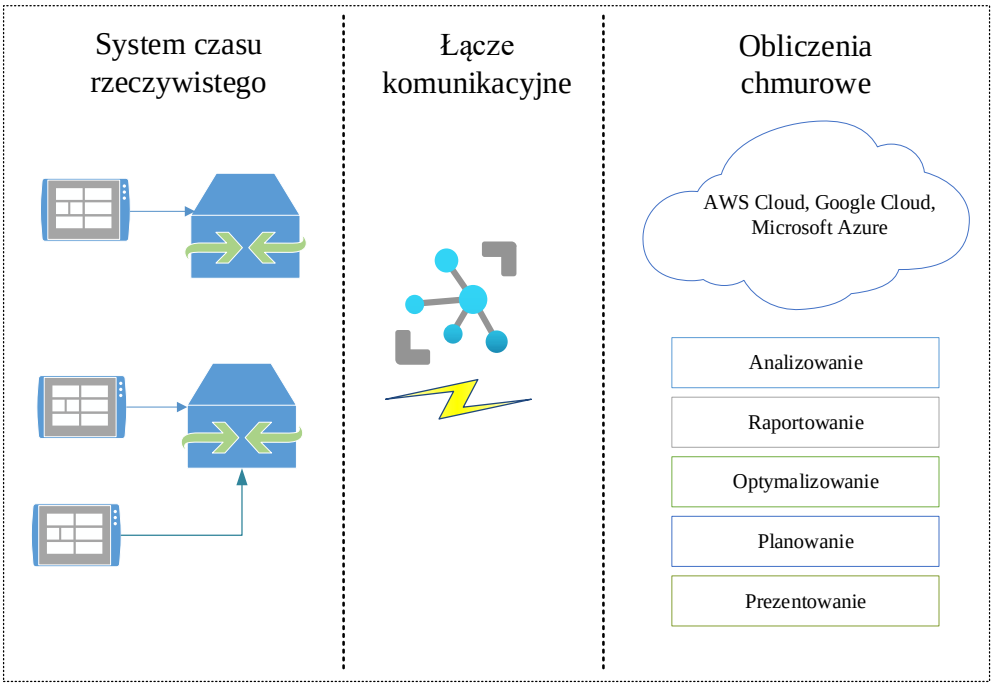
Internet Rzeczy, określenie tak często dzisiaj wykorzystywane w kontekście wszelkiego rodzaju sprzętu AGD, RTV czy urządzeń pozwalających na zdalne sterowanie urządzeniami w domu czy ogrodzie. Ciągły dostęp do sieci spowodował, że użytkownicy nie są w stanie wyobrazić sobie sytuacji, w której zostają pozbawieni możliwości komunikowania się czy zarządzania swoim domostwem.

Początki Internetu Rzeczy jako opisu połączenia sieci ze światem fizycznym za pomocą wielu czujników użył w 1999 roku Kevin Ashton, współzałożyciel Auto-ID Labs. To grupa naukowa, która wypracowała m.in. standardy dla systemu zdalnej identyfikacji radiowej (ang. RFID). Internet rzeczy to sieć połączonych ze sobą urządzeń, które samodzielnie zbierają i udostępniają dane. Podstawą działania Internetu Rzeczy są nie tylko czujniki, takimi urządzeniami mogą być również urządzenia mobilne. Przykładem takiego systemu może być sieć telefonów udostępniających swoją lokalizację na podstawie, której realizowane jest sterowania infrastrukturą miejską.

Przemysłowy Internet Rzeczy łączący technologię zbierania informacji z częścią wykonawczą, która dotyczy usieciowienia procesów i przemysłowych systemów sterowania (ang. Industrial Control System), chodzi o interfejsy człowiek-maszyna, oprogramowanie do nadzorowania produkcji czy programowalne sterowniki logiczne. Połączenie IT z OT zapewnia większą spójność systemów w zakresie automatyzacji i optymalizacji oraz lepszą dostępność danych odnoszących się do dostaw i logistyki. Przedsiębiorstwo posiadające precyzyjne dane o czasie zaopatrzenia i potencjalnych opóźnieniach, może na nie zareagować, np. przezbrajając linie. Natomiast wewnętrzną dystrybucją surowców da się zarządzać z użyciem systemów przekazujących informacje o stanie magazynów lub poszczególnych półek.

IIoT (ang. Industrial Internet of Things) jest ściśle związane z czwartą rewolucją przemysłową, w której informacja z czujników jest wykorzystywana do analizowania aktualnego stanu maszyny np. jej zużycia. Informacje z czujników

dostarczają bardzo często informacje algorytmom sztucznej inteligencji. Algorytmy te pozwalają na przetwarzanie danych w czasie rzeczywistym przez co reagowanie na ewentualne awarie jest szybsze i pozwala na ograniczenie przestojów w pracy maszyny. Posiadając dużą ilość wartościowych danych możliwe staje się generowanie systemów cyfrowego bliźniaka (ang. digital twin). Wprowadzenie do narzędzia projektowego dużej ilości parametrów modelu maszyny spowoduje, że kolejne wersje czy to oprogramowania czy samej maszyny mogą być pozbawione wad poprzednika, przy czym wykorzystywać do pracy mniejszą ilość energii.



Rysunek 3.1 Rysunek wyjaśniający zasadę działania przemysłowego Internetu rzeczy.

Systemy typu Internetu Rzeczy mogą służyć do obserwacji środowiska naturalnego z wykorzystaniem pojazdów typu AGV (ang. Automated guided vehicle) czy klasycznych dronów. W ten sposób przemieszczający się pojazd jest w stanie zbierać informacje z czujników zlokalizowanych na trasie przejazdu lub przelotu. Dodatkowo możliwe jest szybkie reagowanie na zmiany w pracy czujników np. poprzez zdalne aktualizowanie oprogramowania lub przesłanie nowych parametrów pracy.

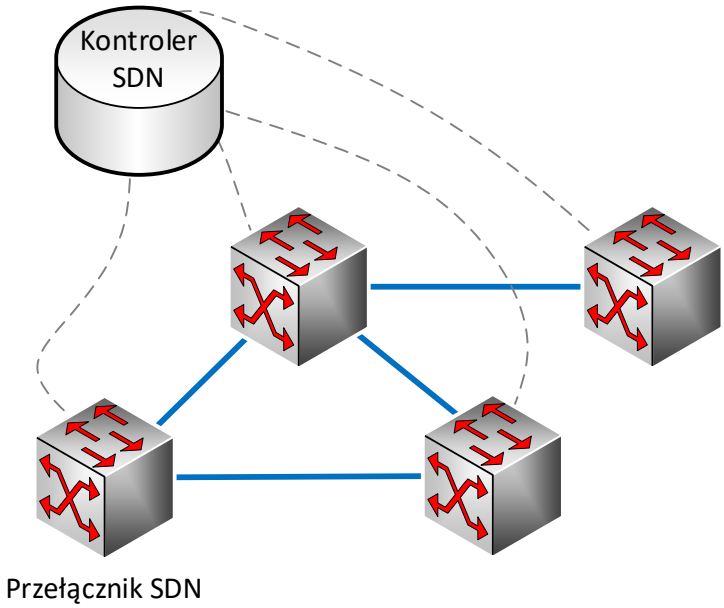
4 Sieci definiowane programowo

Sieci definiowane programowo *(ang. Software Defined Network)* w literaturze określane są również jako sieci programowalne lub sieci swobodnie programowalne. Rozwiązanie to oddziela część transportową od części programowej *(zarządzającej)*. Dzięki takiemu podejściu element zarządzający siecią może centralnie z jednego miejsca zarządzać wszystkimi urządzeniami przełączającymi/dostępowymi. Dzieje się tak dzięki zastosowaniu jednego głównego kontrolera przejmującego wszystkie funkcje zarządcze domyślnie zlokalizowanie w każdym z urządzeń osobno. Zasada działania sieci definiowanych programowo i wykorzystywane mechanizmy zostały opisane w kolejnych podrozdziałach.

4.1 Zasada działania sieci SDN

W systemie sterowania za podstawowy element możemy uznać sterownik programowalny odpowiedzialny za realizację najważniejszych funkcji: sterowanie, komunikacja, wizualizacja. Sieci SDN [26] opierają swoje działanie na protokole OpenFlow, zapewniającym komunikację pomiędzy urządzeniem sterującym, a urządzeniami transportowymi - przełącznikami sieciowymi. Dzięki wyodrębnieniu funkcji sterującej poza urządzenia przełączające możliwe jest szybkie reagowanie na aktualną sytuację w sieci poprzez dynamiczne modyfikowanie konfiguracji sieciowej i tras przesyłu danych zawartych w ruchu sieciowym.

Pojawiające się w pracy określenie „kontroler sieci definiowanej programowo”, może być rozumiany na kilka sposobów. Pierwszy rozumiany jako dodatkowe urządzenie podłączone do infrastruktury sieciowej. Tego typu urządzenia mogą opierać się np. na standardowych komputerach typu PC lub urządzeniach wbudowanych co w dalszej części prace również zostało opisane. Sposób drugi polega na uruchomieniu kontrolera sieci SDN jako usługi sieciowej na przykład jako część infrastruktury chmurowej.



Rysunek 4.1 Schemat ideowy sieci SDN

Kontroler sieci SDN dzięki braku przywiązania do konkretnego rodzaju sprzętu czy producenta jest w stanie aranżować i zarządzać siecią opartą na elementach różniących się pomiędzy sobą, jednak działających ze sobą w jednej spójnej architekturze.

Opierając kontroler sieci SDN na usłudze chmurowej możliwe jest zapewnienie szerokiego poziomu skalowalności i wydajności, a niejako przerzucając część utrzymania na dostawcę usługi chmurowej zwiększa się również dostępność kontrolera.

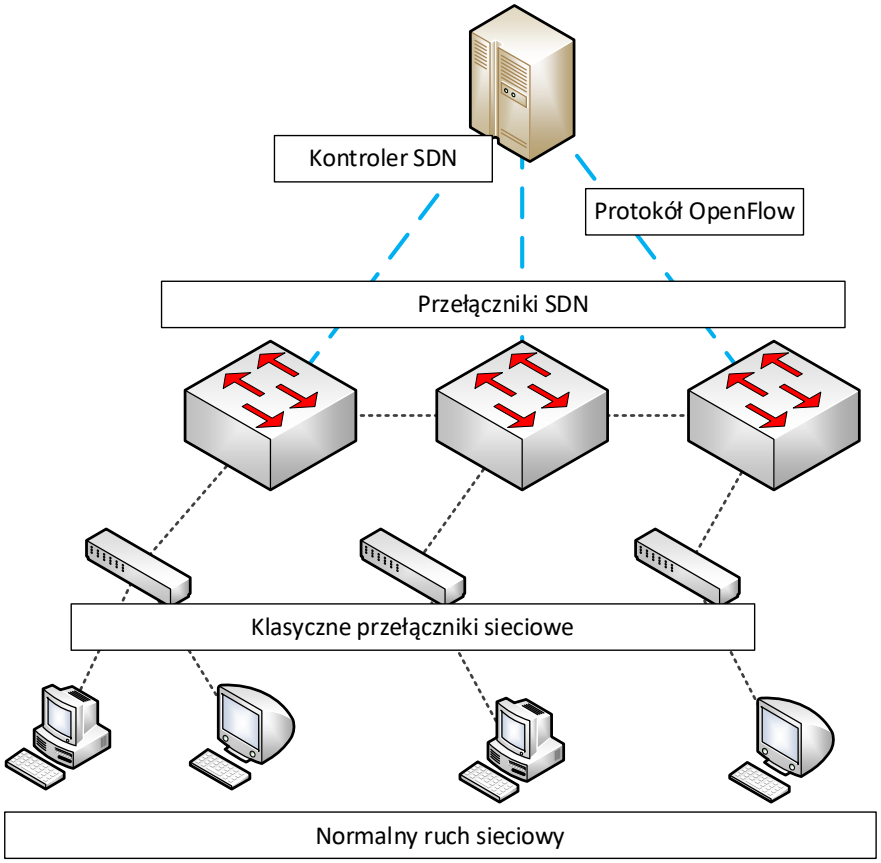
Możliwość uruchomienia części zarządzającej na maszynie wirtualnej czy w środowisku chmurowym powiązana jest z inną technologią wykorzystywaną do zapewniania dostępu do funkcji sieciowych jaką jest NFV. NFV (ang. *Network Function Virtualization*) oznacza wirtualizację funkcji sieciowych, które domyślnie są uruchamiane na urządzeniach fizycznych. Mechanizmy działania wirtualizacji funkcji sieciowych zostały przedstawione w kolejnym podrozdziale.

4.2 Protokoły i narzędzia SDN

Sieci SDN ze względu na charakter swojego działania nie zawsze są w stanie korzystać z typowych rozwiązań sieciowych np. przełączników. Do tego celu wymagane jest użycie urządzeń wspierających natywnie technologię SDN lub pozwalają na instalację oprogramowania zapewniającego komunikację z kontrolerem SDN. Za ustandaryzowanie wymagań dotyczących sieci SDN oraz ciągłe wsparcie zapewnia obecnie organizacja ONF (*Open Networking Foundation*). Organizacja ta wspierana przez największe firmy z branży IT tj. Intel, Google czy Microsoft dba o jak największą dostępność rozwiązań Open Source w kontekście sieci SDN oraz możliwość ich ciągłego rozwoju. W sekcji poniżej przedstawiono narzędzia i metody wykorzystywane do przeprowadzania badań i analiz sieci SDN w kontekście systemów czasu rzeczywistego.

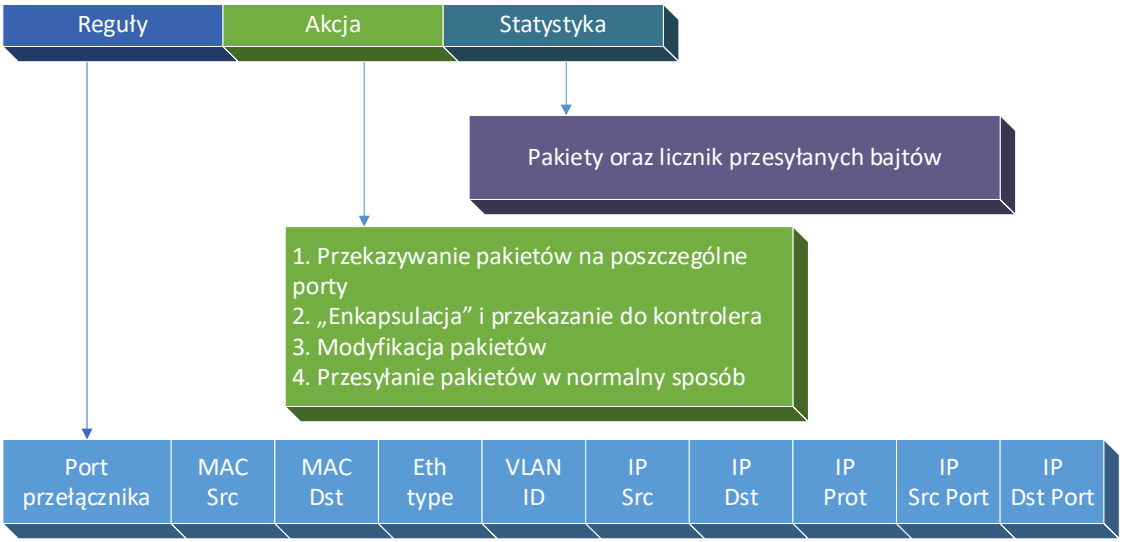
4.2.1 Protokół OpenFlow

Protokół OpenFlow w strukturze odpowiedzialny jest za połączenie i wymianę danych pomiędzy kontrolerem sieci SDN, a przełącznikami sieciowymi. Do jego głównych zadań należy zarządzanie regułami, ich modyfikacja oraz pobieranie informacji z przełącznika. Protokół OpenFlow [27], [28] jest w stanie uzyskać informacje na temat wykorzystania portów, statystyk dotyczących kontrolowanych urządzeń czy aktualnie skonfigurowanych reguł.



Rysunek 4.2 Sieć SDN wraz z urządzeniami końcowymi podłączonymi do infrastruktury

Tabela przepływów określa czynności, jakie powinien podejmować przełącznik po otrzymaniu segmentu o określonych w tabeli cechach. Każdy rekord w tabeli składa się z dziesięciu pól. Kolejność pól w danym rekordzie przedstawiono w poniższej tabeli:



Rysunek 4.3 Informacja o tabeli przepływów konfigurowanej przez protokół OpenFlow

Pierwsze osiem pól służy do identyfikacji otrzymanych segmentów, na których powinny zostać wykonane działania. Przykładowo może to być rekord, który będzie wykonywał pewne działania na wszystkich segmentach otrzymywanych przez przełącznik na porcie 1 i posiadających docelowy adres IP 10.40.48.20. Dziewiąte pole rekordu służy do określenia działań, jakie przełącznik ma wykonać na zidentyfikowanych segmentach. Może to być np. zmiana źródłowego adresu IP lub wysłanie segmentu wskazanym portem. Ostatnie pole rekordu to priorytet, będący wartością liczbową. Pole to wykorzystywane jest, gdy otrzymany segment pasuje do więcej niż jednego rekordu w tabeli przepływów. W takiej sytuacji decyzja o tym jaka akcja powinna zostać podjęta, rozstrzygana jest na podstawie priorytetu.

Tabele przepływów mogą także określać, jakie czynności powinien podjąć przełącznik w przypadku otrzymania segmentu, który nie będzie dopasowany do któregokolwiek rekordu w tabeli przepływów. Każdy przełącznik może wykorzystywać wiele tabel przepływów. W takim przypadku tabele te ustawione są w określonej przez administratora kolejności. Działania związane z poszczególnymi tabelami tworzą ciąg następujących po sobie faz przetwarzania danego segmentu. Manipulowanie zawartością tabel przepływów odbywa się za pomocą protokołu OpenFlow.

Architektura SDN rozwiązuje jeden z najpoważniejszych problemów dzisiejszych sieci komputerowych. Sieci konwencjonalne są mało podatne na zmiany i rekonfigurację, a więc niedostosowane do wymogów dzisiejszych aplikacji potrzebujących elastyczności sieci. Przykładem są w tym wypadku np. serwisy wykorzystujące geolokalizację użytkownika podczas udzielania mu dostępu do zasobów sieciowych. Jest on przekierowywany do odpowiedniego centrum danych na podstawie miejsca, w którym się znajduje. Innym przykładem może być zyskujący ostatnio na popularności trend BYOD wymagający zarówno elastyczności jak i bezpieczeństwa sieci.

Rozwojem architektury SDN zajmuje się Open Networking Foundation. Jest to organizacja skupiająca się na promowaniu, adaptacji do istniejących rozwiązań i wdrażaniu tego paradygmatu. Realizowane jest to poprzez definiowanie otwartych standardów tego typu sieci. Celem działalności tej organizacji jest sprawienie by SDN stał się w przyszłości rozwiązaniem komercyjnym, równie popularnym, co stosowane dziś rozwiązania konwencjonalne. Właśnie ta organizacja odpowiada między innymi za stworzenie i ustandaryzowanie protokołu OpenFlow. Jest to jedyny otwarty standard definiujący sposób komunikacji pomiędzy płaszczyzną kontrolną i transmisyjną sieci w architekturze SDN.

4.2.2 Język programowania P4

Rozwiązaniem mającym duży wpływ na rozwój technologii SDN było przedstawienie w publikacji „P4: Programming Protocol-Independent Packet Processors” sposobu zarządzania ruchem sieciowym za pomocą niezależnego języka programistycznego P4. Koncepcja ta zakłada zdefiniowanie odpowiednich reguł na podstawie kluczowych elementów:

- Nagłówki (*ang. Headers*) opisują strukturę i pola w niej dostępne. W tej części zdefiniowana jest też wielkość poszczególnych pól.

```
header ethernet {
        fields {
        dst_addr : 48;
```

```
        src_addr : 48;
        ethertype : 16;
        }
 }
```

- Analizatory składni (*ang. Parsers*) – element interpretujący i dopasowujący odpowiednie pola pobierane z nagłówka. W przypadku przedstawionym poniżej interpretowana jest pole *„ethertype”* pobierane ze struktury *„ethernet”*.

```
parser ethernet
{
       switch(ethertype) {
         case 0x8100: vlan;
         case 0x9100: vlan;
         case 0x800: ipv4;
       }
}
```

- Tablice (*ang. Tables*) – element opisujący, które pola mają zostać użyte do sterowania ruchem, jakie akcje mają zostać oraz z które pola mają zostać dopasowane do siebie. Typowymi tablicami są tablicę *„Match”* oraz *„Actions”*

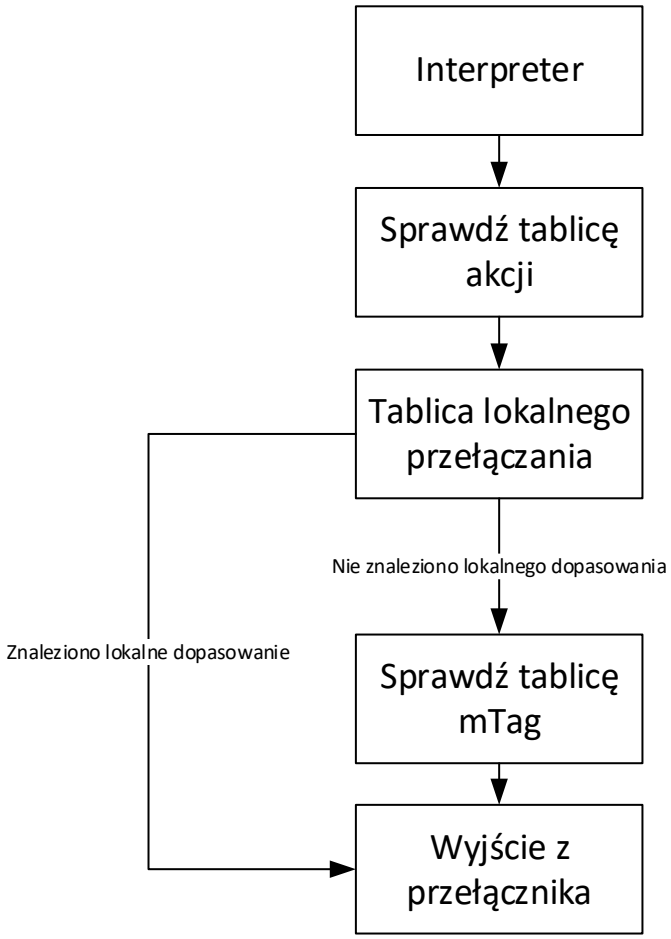
```
table mTag_table {
      reads {
       ethernet.dst_addr : exact;
       vlan.vid : exact;
}
      actions{
       add_mTag;
       }
       max_size : 20000;

}
```

- Akcji do wykonania (*ang. Actions*) – część definiująca akcje jakie mają zostać wykonane w kontrolerze z przesyłanymi ramkami. W

przykładzie poniżej do pola *„mTag”* kopiowana jest informacja o typie przesyłanej informacji, po czym pole to jest ustawiane na wartość określoną w programie sterującym.

```
      action   add_mTag(up1,   up2,   down1,   down2,
egr_spec) {
           add_header(mTag);
                        //Copy VLAN ethertype to mTag
     copy_field(mTag.ethertype, vlan.ethertype);
               //Set VLAN's ethertype to signal mTag
     set_field(vlan.ethertype, 0xaaaa);
     set_field(mTag.up1, up1);
     set_field(mTag.up2, up2);
     set_field(mTag.down1, down1);
     set_field(mTag.down2, down2);
          // Set the destination egress port as well
     set_field(metadata.egress_spec, egr_spec); }
```

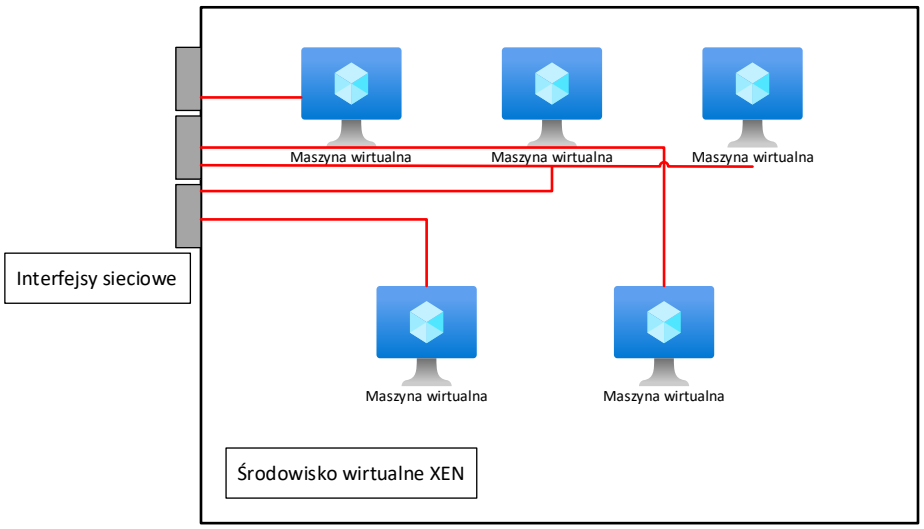
- Programu sterującego (*ang. Control Programs)* – algorytm pozwalający na sterowanie ruchem sieciowym i realizowanie odpowiednich reguł. Algorytm ten opisuje połączenie tablic dopasowania i akcji zapisanych w kontrolerze. *(Rysunek 5.4)*



Rysunek 4.4 Schemat blokowy prezentujący działanie programu sterującego w języku P4

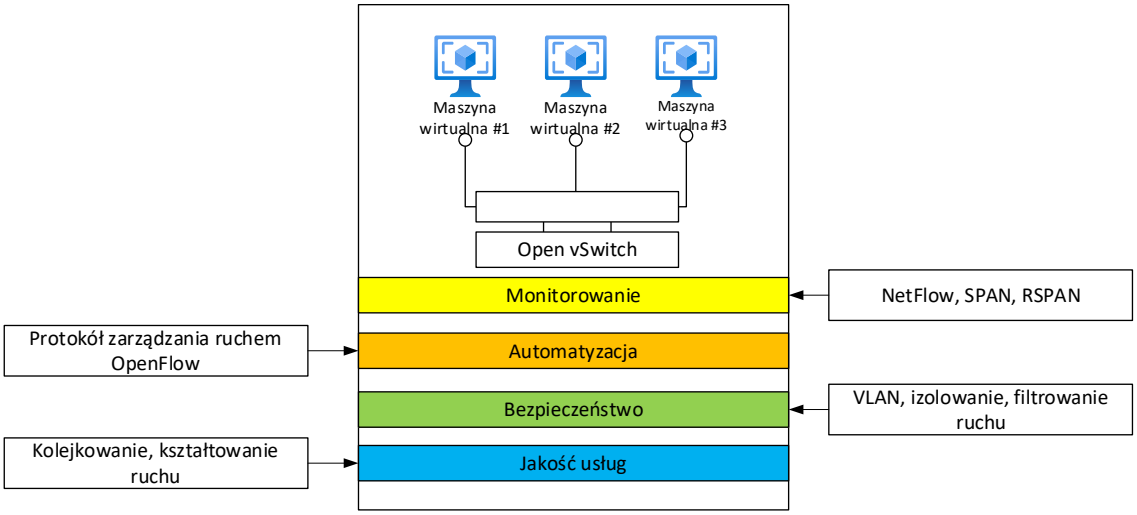
4.2.3 Wirtualny przełącznik sieciowy Open vSwitch

Open vSwitch czyli wirtualny przełącznik sieciowy, w swoim zamiarze miał być i jest wykorzystywany do komunikacji pomiędzy maszynami wirtualnymi. Jest także podstawowym elementem do komunikacji w narzędziu do wirtualizacji urządzeń w monitorze maszyn wirtualnych Xen. Zadaniem narzędzia Xen jest zarządzanie maszynami wirtualnymi uruchomionymi na platformie. W większości przypadków systemy operacyjne muszą być przystosowane do działania z mechanizmami parawirtualizacji wykorzystywanymi do uruchomienia maszyn. W narzędziu Xen najczęściej uruchamiane są systemy operacyjne Linux.



Rysunek 4.5 Wirtualizacja usług sieciowych

Ze względu na możliwość zaimplementowania wirtualnego przełącznika również w środowiskach opartych o platformę Linux i wsparcie dla protokołu OpenFlow jest on również wykorzystywany w urządzeniach pracujących w sieciach SDN. *Rysunek 5.5* przedstawia funkcjonalności jakie może zaoferować wirtualny przełącznik. Zarówno jak na *Rysunku 5.5*, także na *Rysunku 5.6* zaznaczono połączenie wirtualnego przełącznika Open vSwitch z maszynami wirtualnymi. Takie stwierdzenie jest jednak prawdziwe tylko w częściowo, ze względu na możliwość instalacji na rzeczywistych urządzeniach, użytkownik może w taki sposób skonfigurować przełącznik sieciowy, aby zarządzać ruchem przez fizyczne porty ETH.



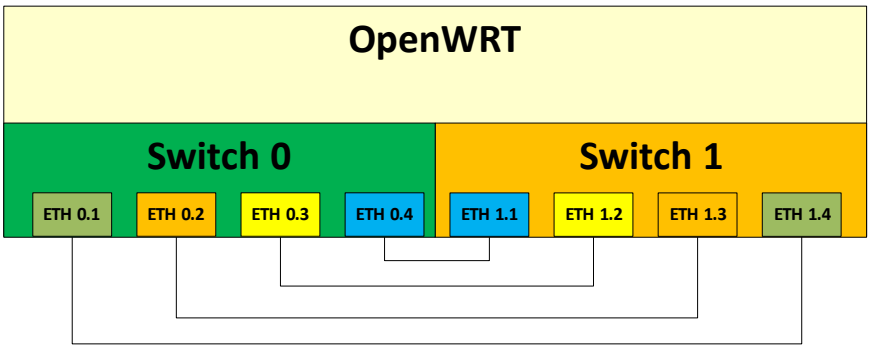
Rysunek 4.6 Schemat przedstawiający funkcjonalności związane z sieciami SDN zbudowanymi w oparciu o przełączniki Open vSwitch

W *Tabeli 1* przedstawiono konfigurację przełącznika sieciowego Mikrotik z zainstalowanym oprogramowaniem OpenWrt oraz uruchomionym wirtualnym przełącznikiem Open vSwitch[29], [30]. Zaprezentowana konfiguracja daje nam informację o kontrolerze, który jest odpowiedzialny za sterowanie ruchem, o liczbie portów możliwych do wykorzystywania w transmisji pomiędzy urządzeniami.

Tabela 1 Konfiguracja przełącznika sieciowego Open vSwitch

```
ubuntu@ubuntu_vm:~$ sudo ovs-vsctl show
    79ae-4c51-97938afb-be7b-637cd4edc535
Bridge ovs-br
    Controller "tcp:192.168.100.150:6653"
    Port "eth1"
  Interface "eth0.1"
Port "eth2"
    Interface "eth0.2"
Port "eth3"
    Interface "eth0.3"
Port "eth4"
            Interface "eth0.4"
Port "eth5"
    Interface "eth0.5"
ovs_version: "2.11.6"
```

Rysunek 5.7 przedstawia w sposób graficzny częściową konfigurację przełącznika sieciowego Open vSwitch opisanego w *Tabeli 1*.



Rysunek 4.7 Konfiguracja przełącznika sieciowego SDN opartego o urządzenie Mikrotk wraz z oprogramowaniem OpenWRT

W zależności od wybranego urządzenia rzeczywistego konfiguracja może się różnić. Wykorzystywane w dalszej części pracy przełączniki SDN, Mikrotik RB2011 pozwalają na instalację oprogramowania OpenWRT,a co za tymi idzie

skonfigurowanie wirtualnego przełącznika sieciowego. Charakterystyka sprzętowa wybranego urządzenia pozwala na zdefiniowane w strukturze 5 portów dostępowych, jednak każdy z portów składa się z 2 interfejsów. Ustawienie przepływów w taki sposób, aby komunikacja odbywała się jedynie na porcie 1 spowoduje, że komputer podłączony zarówno do interfejsu eth0.1 jak i eth1.4 będzie posiadał dostęp do sieci. Tak zbudowane urządzenie może być wykorzystywane w infrastrukturze SDN, jednak jego konfiguracja nie będzie w każdym przypadku taka sama i może się różnić w zależności od użytego urządzenia bazowego.

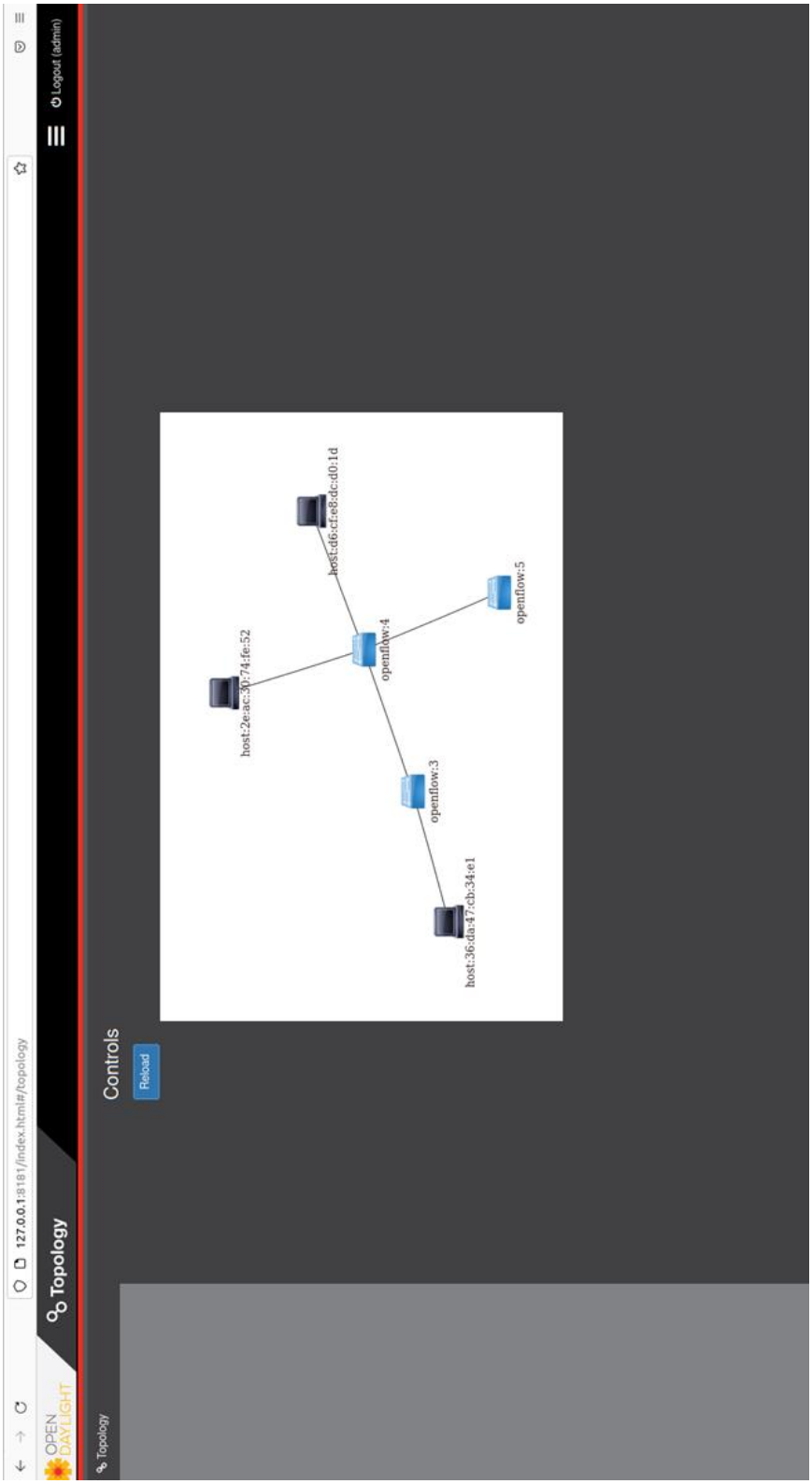
4.3 Kontroler sieci SDN – porównanie

Wybór odpowiedniego kontrolera sieci SDN w dużym stopniu zależny jest od oczekiwanych funkcjonalności systemu. Rozwiązania dostępne na rynku pozwalają na takie dostosowanie programowanej infrastruktury, aby spełniała nie tylko oczekiwania w momencie, ale również uwzględniała możliwości dalszej rozbudowy systemu i wprowadzania aktualizacji do samego kontrolera. W dalszej części pracy przedstawiono trzy rozwiązania typu OpenSource, które wykorzystano do przeprowadzenia eksperymentów. Warto w tym momencie podkreślić, że poniższy przegląd kontrolerów nie wyczerpuje do końca tej tematyki chociażby ze względu na brak porównania do rozwiązań typowo komercyjnych.

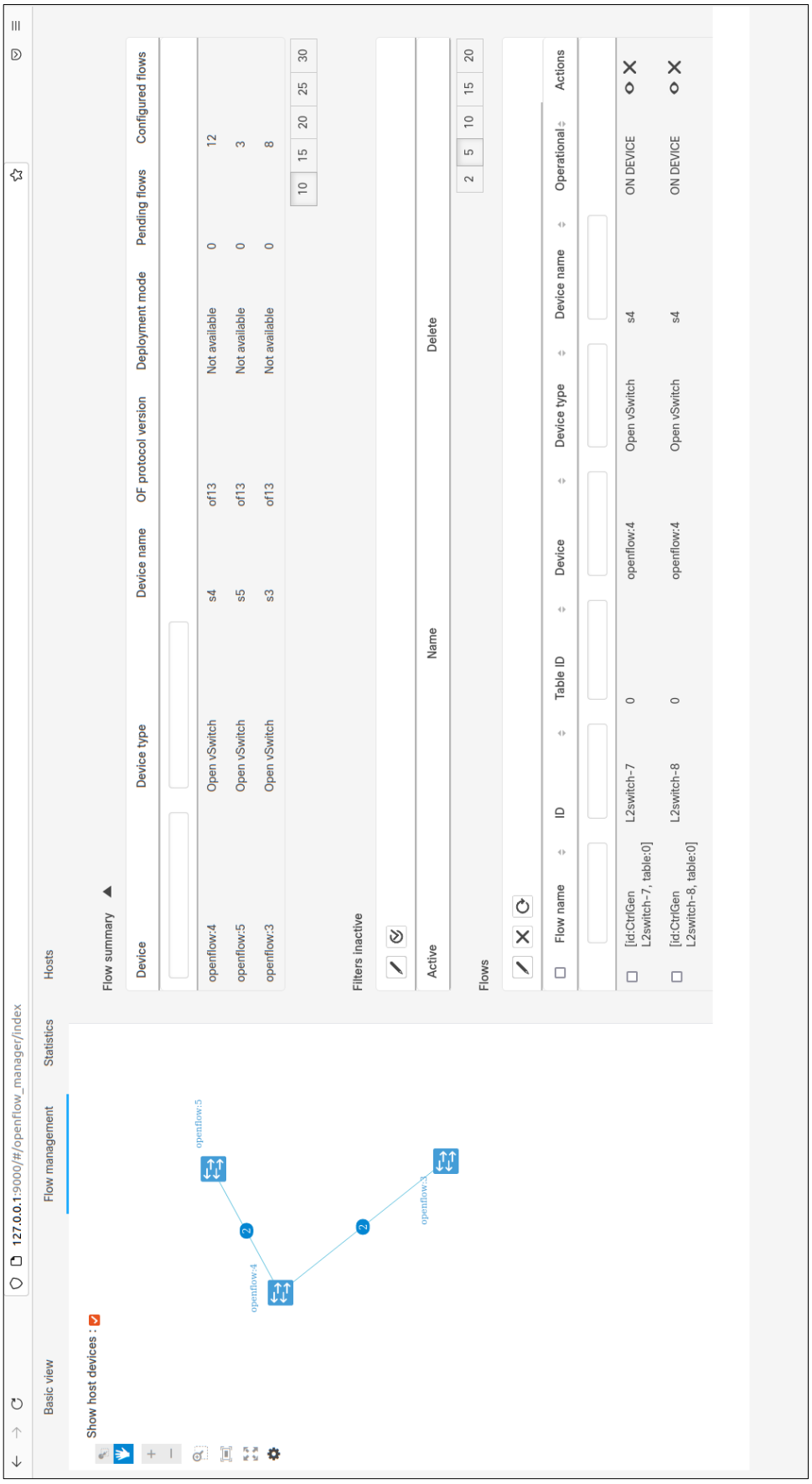
4.3.1 Kontroler OpenDaylight

Projekt typu OpenSource pozwalający na monitorowanie i centralne zarządzanie siecią komputerową. W rozwój środowiska zaangażowane są duże firmy z branży IT tj. Microsoft, Cisco, Citrix, RedHat czy Vmware. W każdym roku kalendarzowym dostępna są dwa nowe wydania. Środowisko OpenDayLight może zostać uruchomione na komputerze lub maszynie wirtualnej z zainstalowanym systemem operacyjnym Linux. Narzędzie korzysta z oprogramowania Java Development Kit. Dodatkową zaletą może być możliwość

połączenia kontrolera z narzędziem OpenFlow Manager. Całością można zarządzać poprzez przeglądarkę internetową.



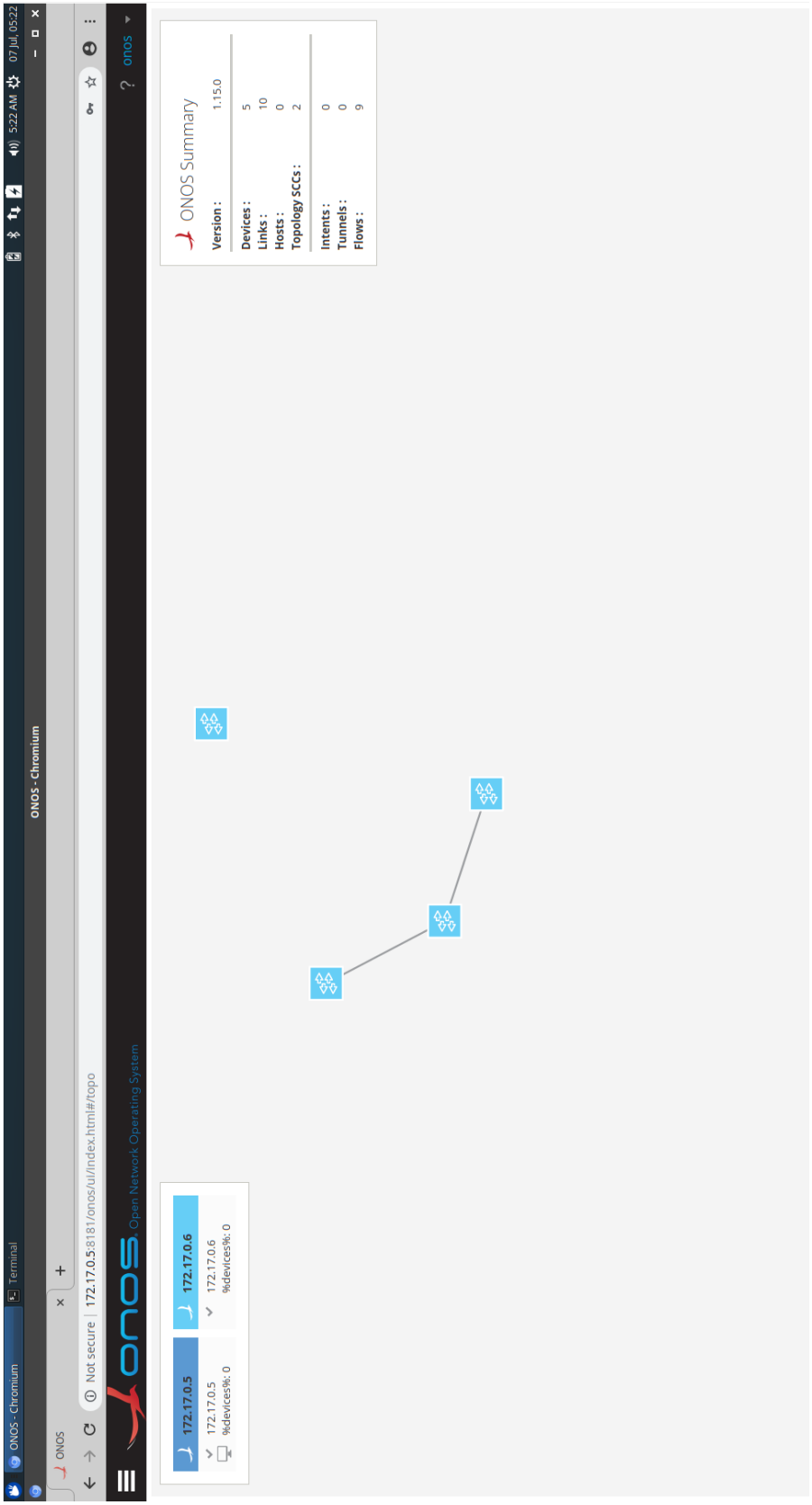
Rysunek 4.8 Kontroler OpenDayLight



Rysunek 4.9 Panel zarządzający OpenFlow Management

4.3.2 Kontroler ONOS

Podobnie jak narzędzie OpenDayLight środowisko zbudowanie w oparciu o platformę Java Development KIT. Rozwiązanie Onosa jest otwarte (typu Open Source), co daje pełną możliwość modyfikacji źródeł. Pozwala na kontrolowanie sieci definiowanych programowo oraz wirtualizację funkcji sieciowych. Jego zaletą jest to, że konfiguracja sieci oraz przesyłu pakietów może odbywać się z poziomu wybranego kontrolera, który decyduje o tym w jaki sposób pakiety mają zostać przesyłane i wysyła taką informacje do przełączników sieciowych. Komunikacja i wprowadzanie reguł do kontrolera odbywa się za pomocą REST API.



Rysunek 4.10 Kontroler ONOS

4.3.3 Kontroler RYU

Kontroler sieci SDN, którego nazwa wzięła się od japońskiej wersji słowa przepływ „ryu”. Pozwala na zarządzanie i kontrolowanie sieci programowalnych, a całość aplikacji może być stworzona w języku Python. Kontroler tego typu pozwala na dużo bardziej dokładne dopasowanie funkcjonalności do aktualnych potrzeb infrastruktury systemu, ze względu na przygotowywanie aplikacji praktycznie od samego początku pod kątem projektowanego systemu. Mimo to kontroler wspiera protokół OpenFlow maksymalnie w wersji 1.5, narzędzia NetConf oraz OF-Config.[31]

Tabela 2 Program sterujący dla kontrolera RYU (dokumentacja kontrolera)

```
from ryu.base import app_manager
from ryu.controller import ofp_event
from ryu.controller.handler import MAIN_DISPATCHER
from ryu.controller.handler import set_ev_cls
from ryu.ofproto import ofproto_v1_0

class L2Switch(app_manager.RyuApp):
    OFP_VERSIONS = [ofproto_v1_0.OFP_VERSION]

    def __init__(self, *args, **kwargs):
        super(L2Switch, self).__init__(*args, **kwargs)

    @set_ev_cls(ofp_event.EventOFPPacketIn, MAIN_DISPATCHER)
    def packet_in_handler(self, ev):
        msg = ev.msg
        dp = msg.datapath
        ofp = dp.ofproto
        ofp_parser = dp.ofproto_parser

        actions = [ofp_parser.OFPActionOutput(ofp.OFPP_FLOOD)]

        data = None
        if msg.buffer_id == ofp.OFP_NO_BUFFER:
             data = msg.data

        out = ofp_parser.OFPPacketOut(
            datapath=dp, buffer_id=msg.buffer_id, in_port=msg.in_port,
            actions=actions, data = data)
        dp.send_msg(out)
```

4.3.4 Porównanie kontrolerów

Wykorzystane w eksperymentach kontrolery sieci definiowanej programowo różnią się w zależności od wybranego rozwiązania. Różnią się szybkością działania i możliwościami dynamicznej rekonfiguracji. Część z nich

lepiej sprawdzi się w momencie uruchomienia jako niezależne urządzenia, z kolei inne zdecydowanie lepiej działają jako usługi uruchomione na maszynach wirtualnych lub jako usługi chmurowe. Przedstawione w poprzednich podrozdziałach rozwiązania zostały sprawdzone w praktyce i ze sobą porównane. Patrząc z perspektywy systemu przemysłowego, najlepszym wyborem jest kontroler ONOS, który pozwoli na szybką rekonfigurację oraz znaczące możliwości rozbudowy sieci. Z kolei spojrzenie z perspektywy działalności naukowej/laboratoryjnej uwydatni zalety kontrolera RYU pozwalającego na zmianę ruchu w sposób kontrolowany z wykorzystaniem skryptu w języku Python. Mimo, że pozwoli to na sprawną zmianę, sprawdzając działanie w systemie docelowym (użycie protokołów przemysłowych) rekonfiguracja powodowała przerwanie komunikacji. Takie zachowanie jest niewskazane, a ciągłe inicjowanie wymian danych jest wręcz niedopuszczalne. Rozwiązaniem, które pozwoliło na połączenie akceptowalnej szybkości działania, możliwości monitorowania i łatwości konfiguracji okazał się kontroler OpenDayLight z dodatkowo zainstalowanym interfejsem zarządzającym OpenFlow Manager. Dzięki takiemu połączeniu uzyskano możliwość realizacji planu badań wraz z ciągłą obserwacją działania sieci. Wybór konkretnego kontrolera sieci SDN nie ma wpływu na ocenę tezy pracy (jej potwierdzenie lub zaprzeczenie).

Tabela 3 Porównanie kontrolerów sieci definiowanej kontrolerów

	ONOS	**OpenDayLight**	**RYU Manager**
Szybkość działania	wysoka	średnia	niska
Możliwość zmiany konfiguracji (Northbound)	Restful APIs/Interfejs użytkownika	Restful APIs/Interfejs użytkownika	Skrypt w języku PYTHON
Southbound	OpenFlow, P4, NETCONF, TL1, SNMP, BGP, RESTCONF, PCEP	OpenFlow, P4, NETCONF, SNMP, BGP, RESTCONF, PCEP	OpenFlow, NETCONF, OF-Config, częściowo P4
Monitorowanie	Zapewniane przez wtyczki możliwe do instalacji	Z ograniczonymi możliwościami	brak
Rekonfiguracja dynamiczna	Na podstawie liczby węzłów, spójność sieci priorytetem	Na podstawie liczby węzłów, wysoka dostępność	brak
Przeznaczenie	Kontrolowanie dużych sieci SD-WAN	Kontrolowanie infrastruktury sieciowej mniejszych rozmiarów	Zastosowanie laboratoryjne, naukowe

Wtyczki	Dostępne	Dostępne	Brak
Platforma	Ubuntu 18.04 LTS	Ubuntu 22.04 LTS	Linux

W *Tabeli 3* porównano cechy trzech wybranych kontrolerów. Jak zostało wspomniane wcześniej każdy z kontrolerów ma swoje charakterystyczne cechy pozwalające na uruchomienie infrastruktury SDN. Każdy z kontrolerów nadaje się również do przeprowadzenia eksperymentów ze względu na wsparcie dla języka P4 czy obsługę protokołu OpenFlow. Dodatkowe funkcjonalności takie jak monitorowanie ruchu sieciowego czy dynamiczna rekonfiguracja są jednak ważnym elementem w celu analizy otrzymywanych w pomiarach wyników.

5 Analiza możliwości wykorzystania sieci SDN w systemach czasu rzeczywistego

Analizując nowe sposoby zarządzania sieciami przemysłowymi oraz nowe możliwości komunikacji pomiędzy dwoma punktami przemysłowymi należy wziąć pod uwagę w jaki sposób komunikacja sieciowa realizowana jest w danym momencie. Przykłady zarządzania i realizacji komunikacji przedstawiono na podstawie popularnych rozwiązań takich jak: Profinet IO, EtherCAT czy Powerlink. W każdym z nich sposób zarządzania infrastrukturą ma trochę inny charakter i skupia się na innych parametrach transmisji.

Ważnym elementem analizy jest analiza wymagań dotyczących urządzeń wbudowanych, które miałyby pełnić rolę kontrolera sieci definiowanej programowo, a także nad aktualnych zastosowań tego typu urządzeń.

5.1 Rozwiązania stosowane obecnie

Proponując inny sposób zarządzania siecią przemysłową należy zastanowić się w jaki sposób połączenia pomiędzy urządzeniami wykorzystywanymi w systemach przemysłowych realizowane są obecnie. A także czy wprowadzenie dodatkowych komponentów systemu przyniesie jakiekolwiek korzyści. Korzystanie z przełączników sieciowych pozwala na zarządzanie ruchem sieciowym na podstawie konfiguracji możliwej do wykonania np. poprzez interfejs dostępny przez przeglądarkę lub środowisko przeznaczone do zarządzania urządzeniami przemysłowymi. Ten rodzaj konfiguracji pozwala na dostosowanie działania sieci do początkowych wymagań systemu. Zmiana połączenia pomiędzy urządzeniami nie jest jednak możliwa w sposób dynamiczny, podobnie jak ingerencja w sprzęt sieciowy używany w instalacji przemysłowej.

Dodatkową przeszkodę w modernizacjach sprzętowych stanowi fakt, że wiele urządzeń sieciowych jest projektowanych ściśle pod kątem jednego rozwiązania przemysłowego np. Profinet czy EtherCAT. W sieciach Profinet wykorzystywane są trzy klasy przełączników sieciowych z których dwie dedykowane do konkretnego typu sieci. Sam protokół zezwoli na prace z podstawowymi usługami sieciowymi, to dopiero użycie sprzętu odpowiedniej

klasy pozwoli na uruchomienie pełnego zakresu funkcjonalności. Użyte w eksperymentach przełączniki sieciowe, także te skonfigurowane jako przełączniki SDN będą również działały poprawnie, jednak parametry czasowe cyklu sieci (jitter, okres sieci) nie zawsze będą spełniały wymagania aplikacyjne.

Systemy przemysłowe bardzo często są wykorzystywane w zamkniętej infrastrukturze, bez dostępu do urządzeń spoza sieci OT. Poszczególne urządzenia posiadają swoje własne zabezpieczenia jak informacje do logowania, certyfikaty wymagane do realizacji połączeń pomiędzy urządzeniami takimi jak sterowniki czy panele HMI. Podobnie jak w sieci IT, także w sieciach OT zastosowanie znajdują zapory ogniowe (ang. *firewall*) lub realizacja komunikacji tylko z wybranymi węzłami siec. W pracy węzeł sieci przedstawiono jako urządzenie pozwalające na zestawienie połączenia. Przykładem węzła sieciowego może być zarówno przełącznik sieciowy SDN, ale również sterownik PLC. Zarówno jedno, jak i drugie jest urządzeniem sieciowym, różnica polega na oferowanych funkcjach. Funkcja przełącznika jest realizacja połączeń sieciowych, natomiast sterownika PLC, wypracowanie informacji wyjściowych na podstawie sygnałów wejściowych. Opierając proponowane rozwiązanie na aktualnym stanie wiedzy, można dojść do wniosku, że dynamiczne zarządzanie urządzeniami aktualnie dostępnymi w sieci lub zmianą w topologii logicznej mogłoby znacząco ułatwić zarządzanie całością systemu. Co najprościej można opisać jako łatwość rekonfiguracji i wprowadzania zmian w przesyłaniu informacji np. poprzez aktywację konkretnego urządzenia lub zmianę parametrów połączenia sieciowego w celu ograniczenia ilości przesyłanych danych.

6 Symulacja systemów SDN

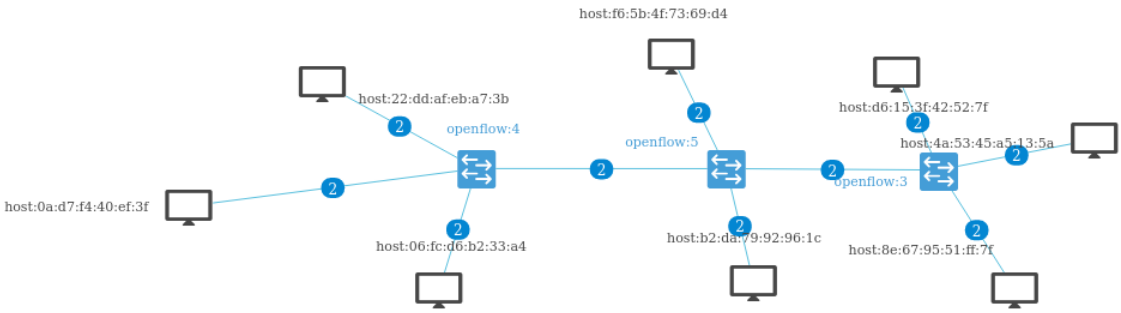
Dostęp do systemów czasu rzeczywistego realizujących konkretny proces przemysłowy jest bardzo ograniczony. Możliwość spowodowania awarii, a co za tym idzie przestojów w produkcji jest kluczowa w tego typu systemach. Z tego powodu przy projektowaniu systemów sieciowych i nie tylko sieciowych wykorzystywana jest symulacja komputerowa.[32] Korzystając z symulacji można przetestować zachowanie urządzeń w momencie wprowadzenia do

infrastruktury innych niekoniecznie certyfikowanych, nieprzeznaczonych do celów przemysłowych urządzeń. W wielu symulatorach dodano możliwość korzystania z wirtualnych obrazów rzeczywistych urządzeń. W pracy analizie poddano zachowanie urządzeń wbudowanych w kontekście pracy zarówno jako urządzenia przemysłowego (wykorzystano urządzenia Beckhoff CX2020 z systemem operacyjnym Windows 7 Embedded), ale również jako urządzenia sieciowe (kontroler SDN, Raspberry Pi z systemem operacyjnym Raspbian/Ubuntu). Analiza została przeprowadzona w dwóch środowiskach symulacyjnych: Mininet oraz GNS3. W pierwszym z nich skupiono się na działaniu kontrolera i sprawdzeniu działania reguł pozwalających na sterowanie siecią komputerowa, której szczególnym przypadkiem jest sieć przemysłowa. Natomiast narzędzie GNS3 pozwoliło na przetestowanie działania sieci bez korzystania z fizycznego sprzętu, co pozwoliło na zaproponowanie urządzeń, które pozwolą na ich wykorzystanie w systemach czasu rzeczywistego z połączeniami sieciowymi opartymi o technologię SDN. Przeprowadzone symulacje omówiono w poniższych podrozdziałach.

6.1.1 Symulacja sieci w narzędziu Mininet

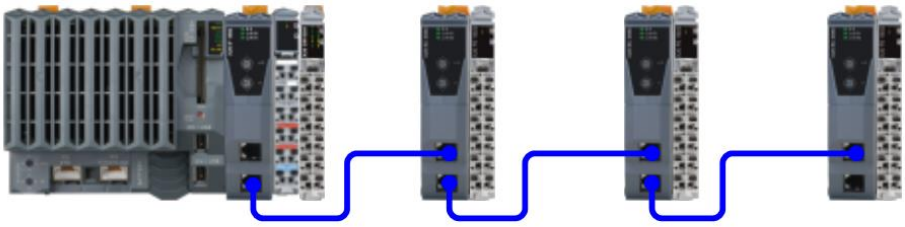
Środowisko symulacyjne Mininet pozwoliło na stworzenie modeli sieci przemysłowych powszechnie stosowanych w infrastrukturze przemysłowej. Zamodelowanie konkretnej topologii sieciowej przy użyciu fizycznych urządzeń to nierzadko trudne zadanie. W tym podejściu możliwe było odwzorowanie rzeczywistych połączeń bez wpływu na działanie sieci. Do modelowania wybrano sieci typu Profinet, EtherCAT oraz mające zastosowanie w infrastrukturze opartej o Internet Rzeczy sieci wykorzystujące protokół MQTT.

Na podstawie schematów przetestowanych fizycznie przygotowane zostały modele sieci z uwzględnieniem w topologii przełączników sieciowych SDN. Ze względu na ograniczenia symulatora urządzenia końcowe nie zostały rozróżnione na tym etapie.



Rysunek 6.1 Topologia sieciowa składająca się z trzech przełączników SDN oraz 8 urządzeń końcowych.

Prosta topologia liniowa *(Rysunek 7.1)* pozwala na połączenie urządzeń których liczba jest ograniczona liczbą portów każdego przełącznika. Ten rodzaj sieci zezwala na połączenie pomiędzy końcowymi punktami sieci jedynie w momencie, gdy wszystkie urządzenia pomiędzy są dostępne lub tak jak w analizowanym przypadku reguły przesłane z kontrolera zezwalają na taki ruch. Zastosowanie tego typu topologii jest wskazane w sieciach przemysłowych w których wykorzystywana jest topologia liniowa. Przykładem takiej sieci może być sieć typu PowerLink, gdzie urządzenia komunikują się bezpośrednio pomiędzy sobą np. sterownik PLC zostaje połączony z modułem wejść/wyjść, który z kolei zostaje połączony do kolejnych urządzeń w sieci. *(Rysunek 7.2)*



Rysunek 6.2 Schemat połączeń w sieci Powerlink

Wprowadzenie przełącznika SDN pomiędzy kolejne urządzenia umożliwia zablokowanie komunikacji i odseparowanie urządzeń z zaplanowanych wymian. Możliwym scenariuszem takich wymian może być zastosowanie kilku identycznych urządzeń w całej topologii i aktywowanie jedynie tego, które powinno brać udział w wymianach. Stworzone w tej części reguły powinny

zawierać informację o adresach MAC, które uwzględnione zostały w dokumentacji modułów wejść/wyjść sieci PowerLink *(Rysunek 7.3)*.

	0x03	InterfaceType_U8	Constant	6	6 → Ethernet CSMA/CD
	0x04	InterfaceMtu_U16	Constant	1500	
	0x05	InterfacePhysAddress_OSTR	Constant		MAC address: "xx:xx:xx:xx:xx:xx"
	0x06	InterfaceName_VSTR	Read only	"IF1"	
	0x07	InterfaceOperStatus_U8	Read only	1	0 = Down, 1 = Up
	0x08	InterfaceAdminState_U8	Read/Write	1	0 = Down, 1 = Up
	0x09	Valid_BOOL	Read/Write	TRUE	
0x1050		NMT_RelativeLatencyDiff_AU32			
	0x00	NumberOfEntries	Read only	254	
	0x01 - 0xFE	RelativeLatencyDiff	Read only	0	
0x1101		DIA_NMTTelegrCount_REC			
	0x00	NumberOfEntries	Constant	8	

Rysunek 6.3 Opis parametrów definiujących sieci PowerLink

Ustawienie reguły pozwalającej na zablokowanie ruchu sieciowego na podstawie adresu MAC w tym konkretnym przypadku powinno wyglądać następująco:

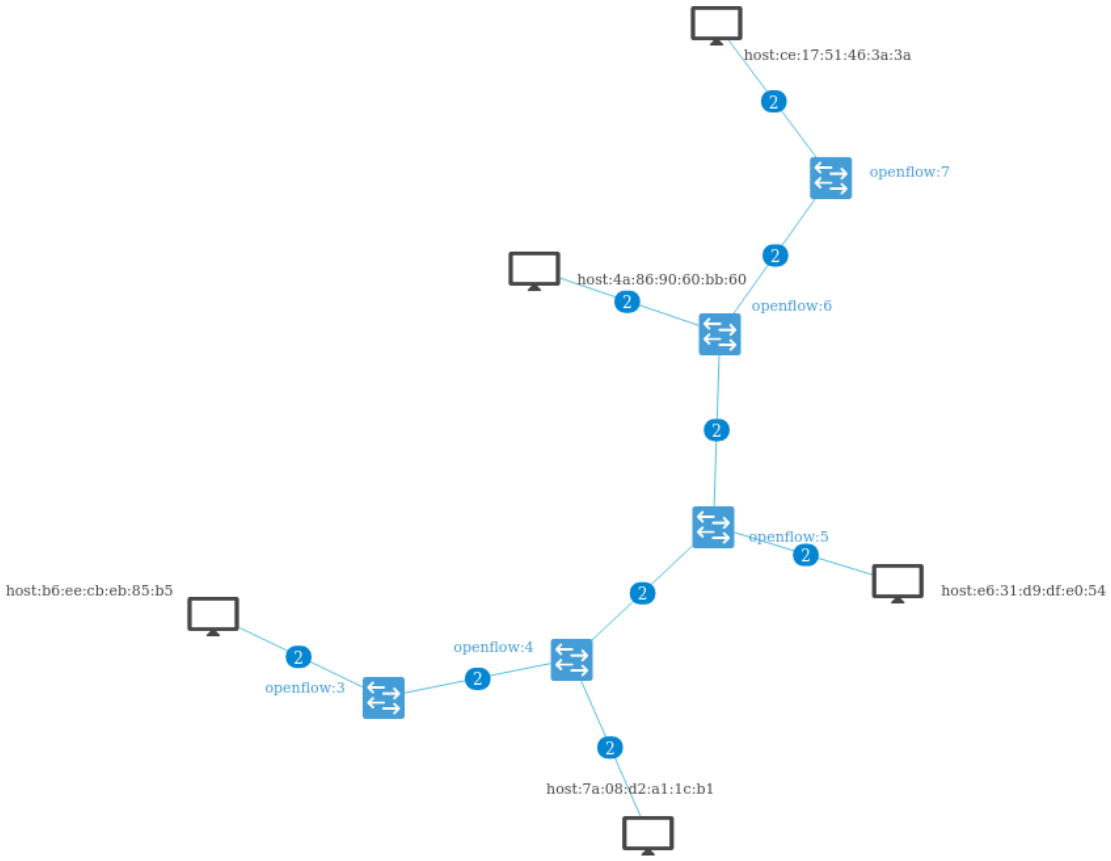
ovs-ofctl add-flow br0 "table=0, dl_dst=01:80:c2:00:00:00, actions=drop"

Przeciwnością takiej sytuacji jest możliwość przesłania ramek jak przez klasyczny przełącznik sieciowy, czyli zezwolenie na ruch sieciowy dla konkretnego adresu MAC.

ovs-ofctl add-flow br0 "table=0, dl_dst=01:80:c2:00:00:00, actions=normal"

Zamodelowanie prostych topologii sieciowych bazujących na rozwiązaniach stosowanych w systemach przemysłowych pozwoliło na przeanalizowanie możliwości kontrolowania interfejsów w przełączniku sieciowym SDN. Zaprezentowane poniżej modele opierają swoją budowę na topologiach stosowanych powszechnie w systemach przemysłowych.

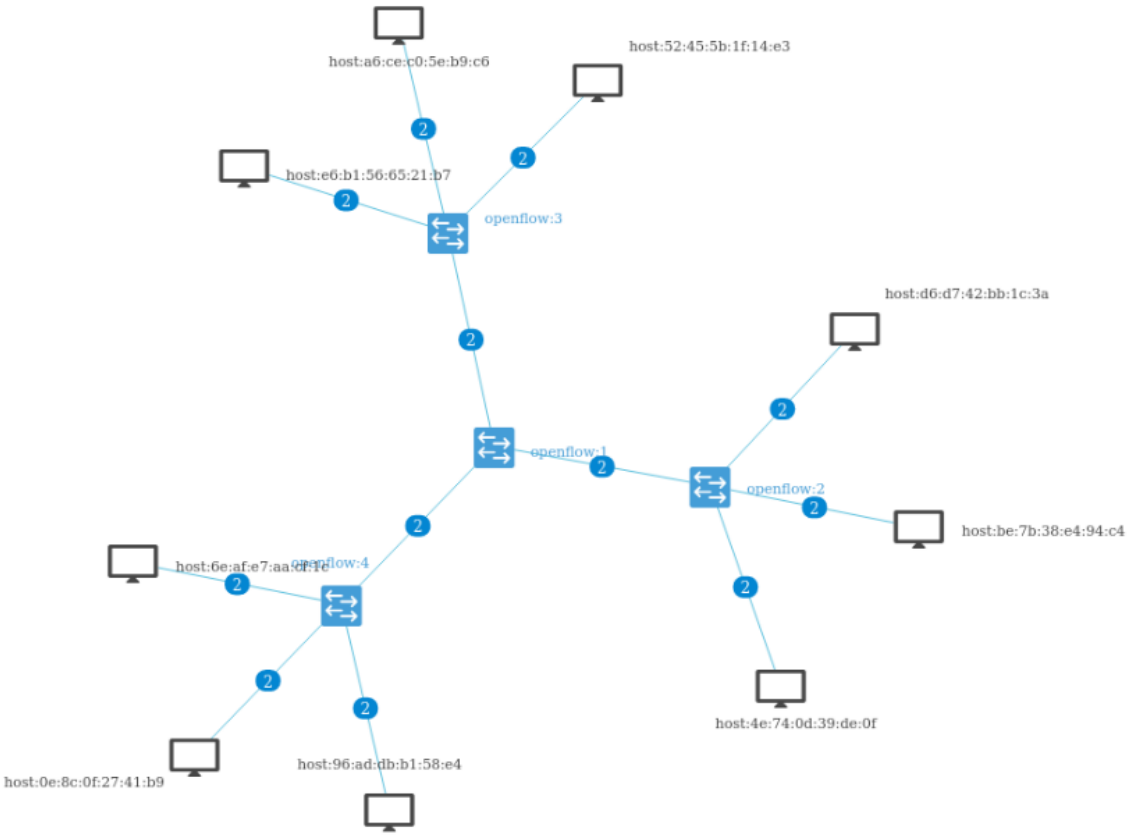
Topologia liniowa zaprezentowana na rysunku 5.4 ma odzwierciedlać połączenie możliwe do zrealizowania w sieciach typu EtherCAT. Każdy z przełączników oprócz połączeń pomiędzy przełącznikami posiada podłączone urządzenie końcowe. W symulacji nie rozróżniamy rodzaju urządzenia końcowego na sterowniki PLC lub komputery typu PC. *(Rysunek 7.4)*



Rysunek 6.4 Schemat przedstawiający topologię liniową zasymulowaną w narzędziu Mininet

Wykorzystana topologia liniowa z przełącznikami SDN odpowiada konfiguracji EtherCAT z jędnym urządzeniem typu Master i czterema urządzeniami typu Slave. Do jednego przełącznika sieciowego SDN podłączonych może zostać kilka urządzeń typu Slave, które mogą być dynamicznie przełączane i wybierane na podstawie aktualnego stanu systemu (aktualnych parametrów procesu).

Topologia typu gwiazda przedstawiona na *Rysunku 7.5* może zostać porównana do prostych sieci typu Modbus TCP lub dużo bardziej złożonych sieci typu Profinet. Sieci Profinet ze posiadają większe zaawansowanie mechanizmów kontroli czasu transmisji i konfiguracji urządzeń w stosunku do sieci opartych o protokół Modbus TCP. Ze względu jednak na możliwość rozróżnienia urządzeń na podstawie adresu IP w obydwu przypadkach do dalszej analizy w symulacji wybrano dużo prostszy w testowaniu i diagnostyce protokół Modbus TCP.



Rysunek 6.5 Schemat przedstawiający topologię typu gwiazda zasymulowana w narzędziu Mininet

Po etapie implementowania topologii sieciowej sprawdzono możliwość sterowania ruchem sieciowym w kontekście użyteczności w sieciach przemysłowych. Podchodząc do problematyki przełączania urządzeń dostępnych w sieci skupiono się na możliwości zmiany aktualnie dostępnego urządzenia końcowego. Eksperymenty polegały na zasymulowaniu kilku urządzeń końcowych tego samego typu podłączonych do infrastruktury sieciowej i dynamicznym zarządzaniu aktualnie dostępnym urządzeniem. Dzięki takiej funkcjonalności możliwe staje się przenoszenie procesu przemysłowego pomiędzy aktualnie dostępnymi urządzeniami. To z kolei pozwala na skrócenie czasu przestoju po wystąpieniu awarii bądź w trakcie modernizacji linii technologicznej. W tym etapie skupiono się na sterowaniu dostępnością urządzeń i blokowaniu całego ruchu sieciowego pomiędzy urządzeniami w sieci.

Przetestowanie działania reguł została oparte o funkcję *„pingall”*, czyli sprawdzania komunikacji pomiędzy kolejnymi węzłami. W pierszym przypadku odseparowano od sieci cały segment pośrodku topologii liniowej. Z tego powodu możliwa jest komunikacja pomiędzy urządzeniami H1 i H2, a także H3 i H5. Urządzenie H4 jest całkowicie odseparowane, a co za tym idzie również komunikacja z innymi węzłami jest niemożliwa. *(Rysunek 7.6)*

	2	3	4	5
1	OK			
2	OK			
3				OK
4				
5			OK	

```
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 X X X
h2 -> h1 X X X
h3 -> X X X h5
h4 -> X X X X
h5 -> X X h3 X
*** Results: 80% dropped (4/20 received)
```

Rysunek 6.6 Wynik testów przeprowadzonych w narzędziu Mininet polegających na aktywacji odpowiednich połączeń - 80% ruchu jest blokowana

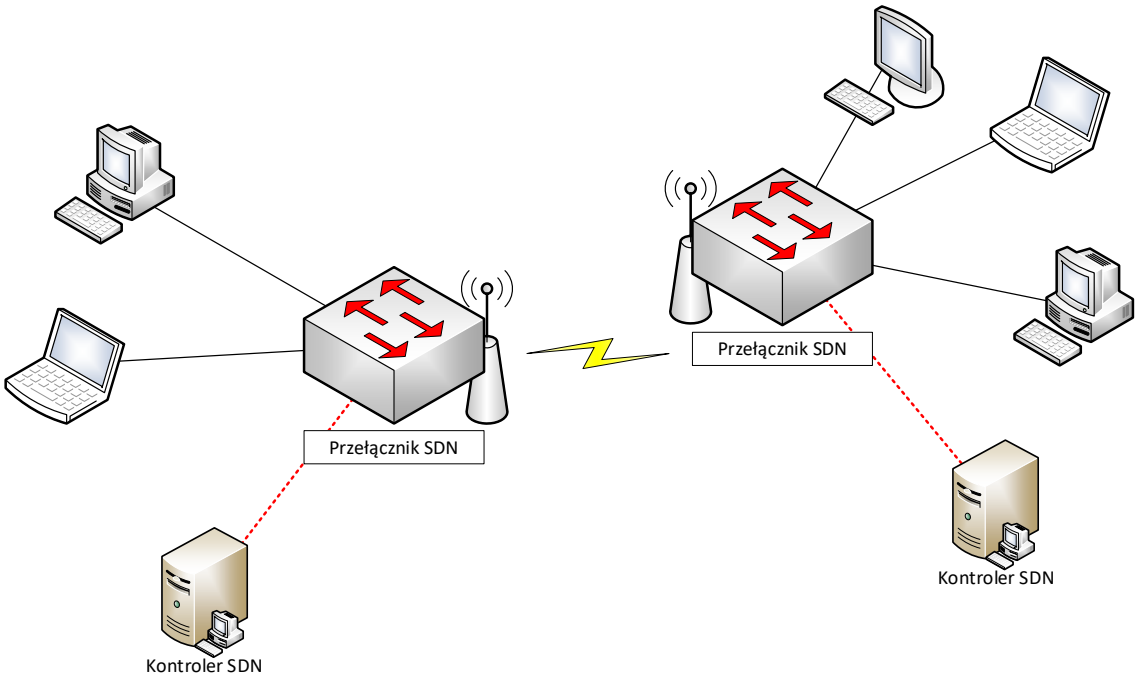
W drugim przypadku zablokowana została komunikacja z jednym urządzeniem na podstawie jego adresu IP. W ten sposób możliwe jest przełączanie urządzeń biorących udział w transmisji. Przykładem takiego zarządzania może być odbieranie danych tylko z określonego węzła, który może być źródłem danych (serwerem). *(Rysunek 7.7)*

	2	3	4	5
1	OK	OK		OK
2	OK	OK		OK
3	OK	OK		OK
4				
5	OK	OK	OK	

```
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3 X h5
h2 -> h1 h3 X h5
h3 -> h1 h2 X h5
h4 -> X X X X
h5 -> h1 h2 h3 X
*** Results: 40% dropped (12/20 received)
```

Rysunek 6.7 Wynik testów przeprowadzonych w narzędziu Mininet polegających na aktywacji odpowiednich połączeń - 40% ruchu jest blokowane

Część symulacyjna pozwoliła określić sposób połączenia urządzeń bezprzewodowych w sieci typu SDN. Nie zdecydowano się na podłączenie i zarządzanie punktem dostępowym z możliwością obsługi protokołu OpenFlow ze względu na niemożliwe „tunelowanie” protokołów przemysłowych do łącza bezprzewodowego. Połączenie zostało zaprezentowane na *Rysunku 7.8*. Zastosowanie dwóch kontrolerów nie zawsze jest wymagane, jednak brak możliwości komunikacji z usługą sieciową spowoduje problem z zastosowaniem odpowiednich reguł, a co może się wiązać z błędną transmisją danych.

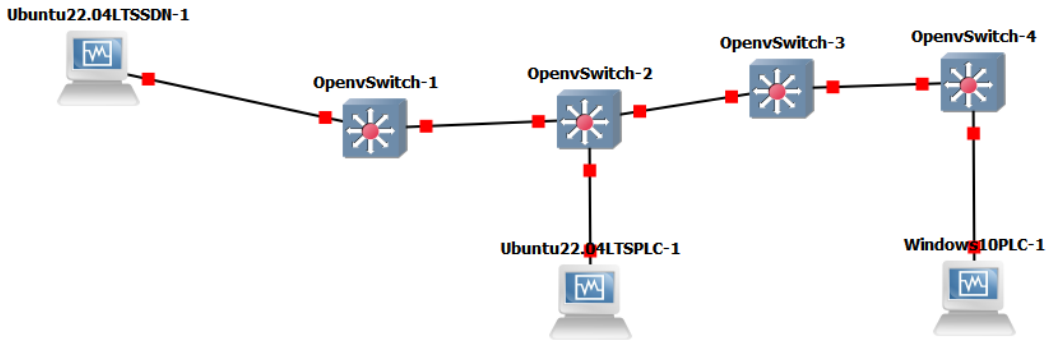


Rysunek 6.8 Model symulacji sieci bezprzewodowej w narzędziu Mininet

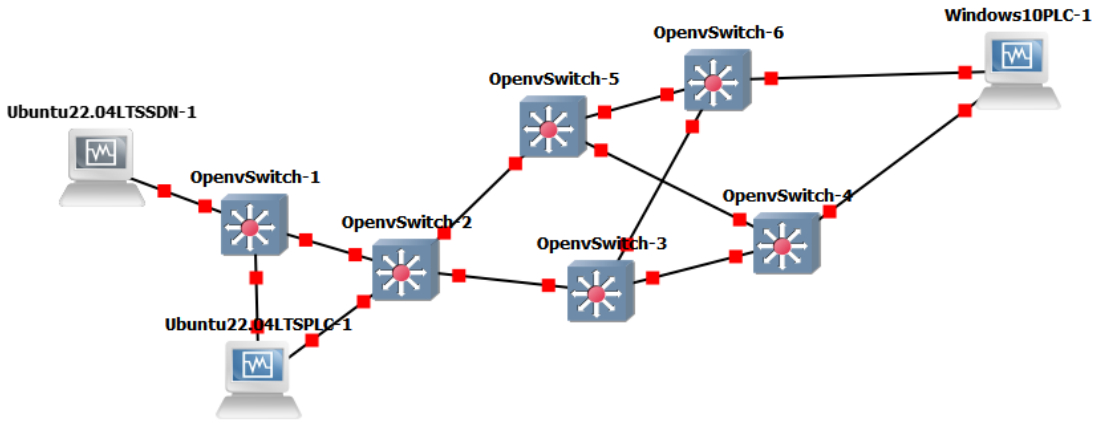
6.1.2 Symulator sieci komputerowej - GNS3

Znacząco różniącym się środowiskiem symulacyjnym jest narzędzie GNS3. Korzystając z możliwości uruchamiania maszyn wirtualnych zasymulowanie rzeczywistego obiektu jest zdecydowanie prostsze. Stworzenie i uruchomienie maszyny wirtualnej wraz „runtime'em" CodeSys [33]–[35] pozwoliło wykorzystać pełne możliwości przełączników sieciowych SDN i sprawdzić możliwości zarządzania na podstawie nie tylko takich parametrów jak adres IP czy fizyczny adres MAC. Do sterowania wprowadzono dodatkowo pola typu odpowiadające za analizę pola „Ethertype" co pomogło w zarządzaniu ruchem sieciowym bezpośrednio pod kątem wykorzystanego protokołu sieciowego. *Rysunki 7.9* do *7.11* przedstawiają zrealizowane systemy sieciowe. W samym założeniu symulacja miała pomóc w określeniu czy założenia poczynione przy planowaniu badań okazały się słuszne. Sama możliwość zrealizowania konkretnych połączeń i uruchomienia programu testowego (program wykorzystywany w części badawczej odpowiedzialnej za urządzenia wbudowane) odpowiedziała na sensowność prowadzenia dalszych badań. Brak badania parametrów czasowych w tej części nie wpływa na możliwość postawienia odpowiedzi na wprowadzone na początku rozprawy tezy. Uruchomienie kolejnych

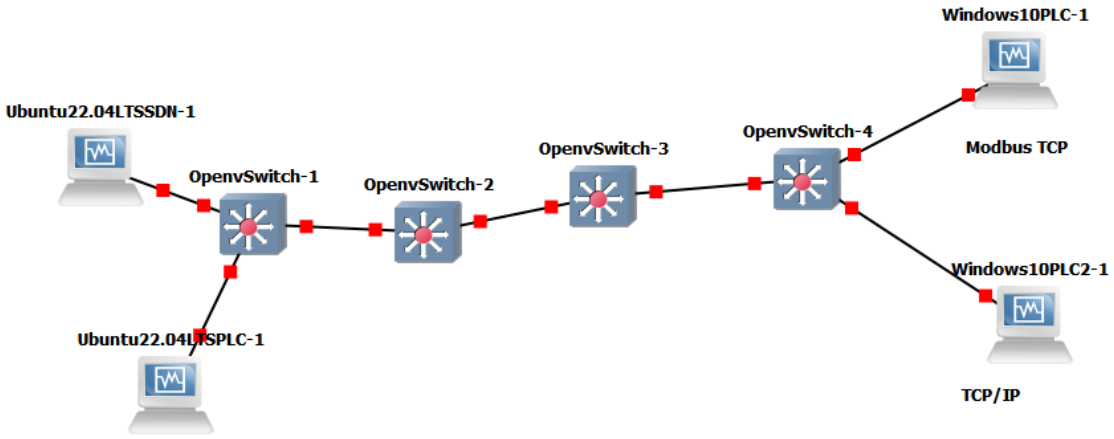
węzłów końcowych jako maszyn wirtualnych pozwoliło jednak na zbudowanie węzłów symulujących rzeczywiste urządzenia.



Rysunek 6.9 Topologia liniowa (w odniesieniu do połączenia pomiędzy przełącznikami) zrealizowana w środowisku testowym GNS3



Rysunek 6.10 Topologia typu siatka (MESH) zrealizowana w środowisku testowym GNS3



Rysunek 6.11 Topologia liniowa (w odniesieniu do połączenia pomiędzy przełącznikami) zrealizowana w środowisku testowym GNS3 z urządzeniami na końcach sieci

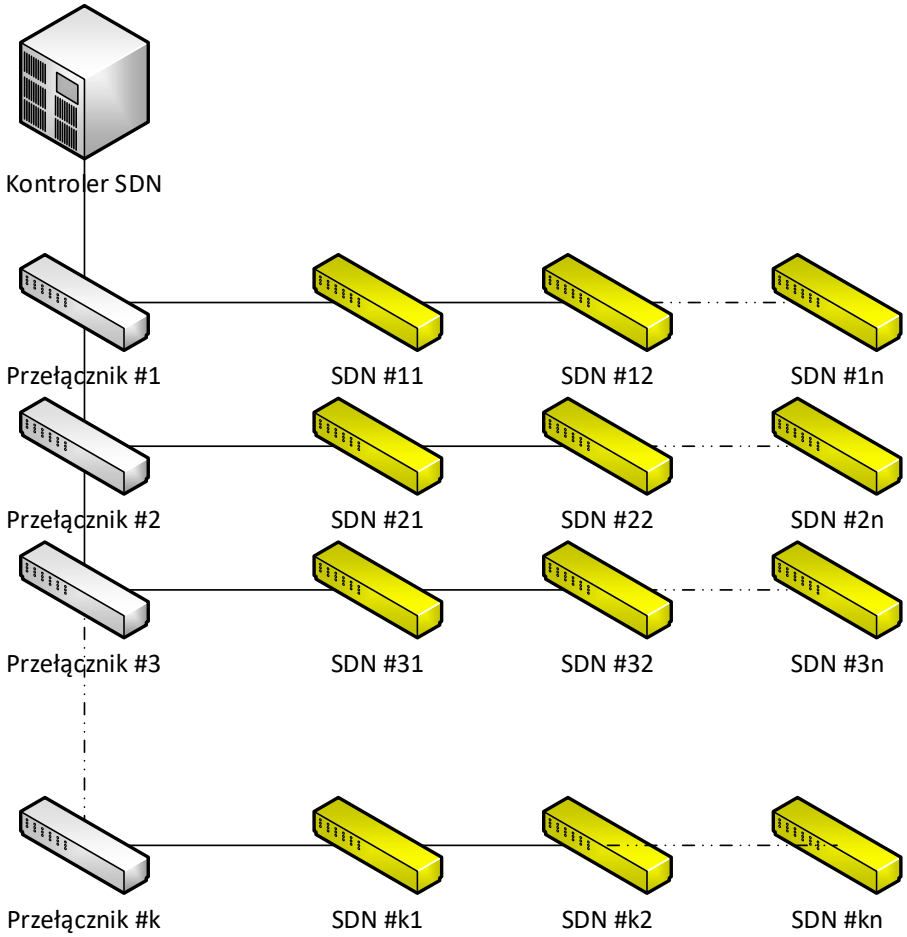
6.2 Opis metodologii badań

W celu wybrania odpowiednich metod i technik przeprowadzenia badań wykonano symulację rzeczywistego obiektu przemysłowego. Zdecydowano o sprawdzeniu urządzeń pod względem wydajności i rzeczywistego wpływu na działanie wszystkich urządzeń w szczególności tych wymagających spełnienia wymagań pracy w czasie rzeczywistym. Eksperymenty zostały przeprowadzone w kilku scenariuszach mających zwrócić uwagę na możliwe problemy oraz wskazać możliwe obszary zastosowania badanych technologii. Omawiane scenariusze zawierają się w przedstawionym planie badań:

- Zastosowanie urządzenia wbudowanego w systemach czasu rzeczywistego – możliwość spełnienia wymagań czasowych,
- Zastosowanie wirtualnego przełącznika sieciowego zbudowanego na platformie urządzenia wbudowanego – wpływ na działanie sieci przemysłowej,
- Zarządzanie ruchem sieciowym w oparciu o rodzaj przesyłanych – wpływ kontrolera sieci SDN na pracę systemu przemysłowego,
- Zestawienie połączenia bezprzewodowego w sieci przemysłowej z wykorzystaniem urządzeń SDN – wpływa opóźnienia wprowadzonego przez połączenie bezprzewodowej.

7 Model badawczy

Przed rozpoczęciem prac badawczych opracowano model pozwalający na opisanie działania sieci przemysłowej i porównanie otrzymywanych wyników. Opierając swoje rozważania na publikacji określającej koszt realizacji sieci definiowanej programowo pod kątem opłacalności finansowej stworzono model opisujący wpływ takiej sieci na obiekt przemysłowy [36]. Określono także wymagania dotyczące przebiegu badań, wymaganych urządzeń oraz wartości parametrów/współczynników, którymi można się posłużyć do sprawdzenia realizowalności przedsięwzięcia. *(Rysunek 8.1)*



Rysunek 7.1 Infrastruktura sieciowa wybrana do opracowania modelu badawczego

W modelu sieci możemy określić:

- Liczbę przełączników sieciowych SDN dostępnych w infrastrukturze sieciowej

$$S_p = \{s_{p1}, s_{p2}, s_{p3} \dots s_{pn}\} \quad (1)$$

- Liczbę klasycznych przełączników sieciowych dostępnych w infrastrukturze sieciowej

$$S_k = \{s_{k1}, s_{k2}, s_{k3} \dots s_{kk}\} \quad (2)$$

Każdy z przełączników sieciowych zarówno klasycznych jak i programowalnych (SDN) można opisać dodatkowo parametrami technicznymi:

- Przepustowość interfejsu komunikacyjnego - określa ilość możliwych do przesłania danych wybranym portem komunikacyjnym w jednostce czasu.
- Liczba dostępnych portów komunikacyjnych – przełącznik z dwoma portami pozwala na podłączenie maksymalnie dwóch urządzeń. Ilość portów może wpływać na szybkość działania urządzenia, a co z kolei będzie miało wpływ na wprowadzone opóźnienie.
- Opóźnienie wprowadzone do sieci przez przełącznik – określa ilość czasu potrzebnego na obsługę ramki i przesłanie jej dalej do kolejnego urządzenia. W rozważaniach przyjęto opóźnienie mierzone w milisekundach *„ms”*.
- Czas rekonfiguracji przełącznika – określa ilość czasu potrzebną na zmianę przepływów (zmiany zasad ruchu sieciowego tj. blokowanie określonych ramek lub odpowiednio zdefiniowanych urządzeń). Podobnie jak poprzednio uwzględniono czas mierzony w *„ms”*.

- Liczbę kontrolerów SDN dostępnych w infrastrukturze - od liczby kontrolerów, zależny będzie czas rekonfiguracji sieci. Urządzenia w sieci mogą być działać jako kontrolery redundantne lub jako urządzenia odpowiedzialne za sterowanie innymi częściami sieci.

$$C_{SDN} = \{c_1, c_2, c_3 \dots c_n\} \quad (3)$$

- Liczbę połączeń pomiędzy węzłami – określa ilość wymaganych połączeń kablowych lub bezprzewodowych pomiędzy urządzeniami sieciowymi dostępnymi w systemie. W zależności od zastosowanego łącza wprowadzone opóźnienie może się znacząco różnić.

$$L = \{l_1, l_2, l_3 \dots l_n\} \quad (4)$$

Dodatkowym elementem modelu jest liczba lokalizacji, w której zainstalowana zostanie infrastruktura SDN. System przemysłowy może składać się z kilku lokalizacji (instalacje procesowe mogą być zlokalizowane w różnych budynkach). Każda z tych lokalizacji może charakteryzować się inną specyfiką ruchu sieciowego. Z tego powodu ten element może zostać uwzględniony w składowych opisujących ruch sieciowy.

Korzystając z opisanych parametrów można zdefiniować funkcję kosztu transmisji z punktu A do punktu B $-\ C$, a także dodatkowy czas wymagany na rekonfigurację przełączników. Jako punkt *„A”* przyjęto przełącznik numer #1, a jako urządzenie *„B”* przełącznik SDN #32. Do opracowania modelu badawczego przyjęto dwa przypadki.

***Przypadek 1*:** Przełącznik SDN jest nieskonfigurowany i wymagane jest odczytanie informacji z kontrolera, a dopiero później zrealizowanie połączenia.

$$C = 2 * (\sum_{i=1}^{k} \delta_i + \sum_{j=1}^{n} \Delta_j) + n * \varepsilon \quad (5)$$

$\delta_i - czas\ połączenia\ pomiędzy\ klasycznymi\ przełącznikami$

$\Delta_j - czas\ połączenia\ pomiędzy\ przełącznikami\ SDN$

$\varepsilon - czas\ realizacji\ zadań\ kontrolera$

$n - liczba\ kontrolerów\ w\ sieci$

Przypadek 2: Przełącznik SDN jest skonfigurowany i połączenie może być zrealizowane natychmiastowo.

$$C = (\sum_{i=1}^{k} \delta_i + \sum_{j=1}^{n} \Delta_j) \tag{6}$$

Wprowadzono również równania zawierające elementy potrzebne do określenia czasu potrzebnego do połączeń pomiędzy przełącznikami. Do analizy przygotowano tabelę ze współczynnikami uwzględnionymi w równaniach. Zaproponowane równania pozwalają na wstawienie wartości oczekiwanych lub odczytanych z kart katalogowych urządzeń i określenie wymagań dotyczących projektowanej sieci.

$$\delta_i = x * (1 + b + p) \tag{7}$$

$x - domyślny\ czas\ osbługi\ zadań\ w\ przełączniku$

$b - przepustowość$ łącza

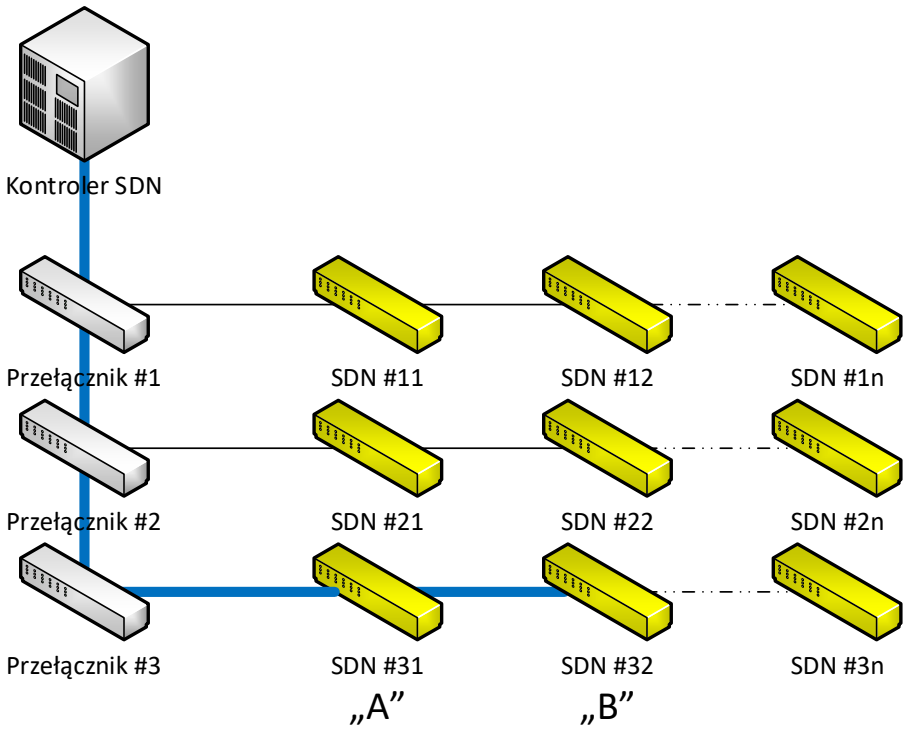
$p - liczba\ \ interfejsów\ komunikacyjnych$

Tabela 4 Tabela prezentująca współczynniki uwzględnione w równaniach

Przepustowość interfejsu Mbit/s	10	100	1000	10000
Współczynnik	0.75	0.5	0.25	0.1
Ilość interfejsów	8	16	24	48
Współczynnik	0.08	0.16	0.24	0.48
Ilość kontrolerów w sieci	1	2	5	10
Współczynnik	0.05	0.1	0.25	0.5

$$\Delta_j = x * (1 + b + q\ + s) \tag{8}$$

$q - czas\ obsługi\ reguł\ z\ kontrolera$



Rysunek 7.2 Schemat sieci ze zdefiniowanym połączeniem pomiędzy urządzeniami „A” oraz „B”

Dla połączenia *„AB”* W celu dalszej analizy parametry zostały określone jako pesymistyczne (najgorszego przypadku, ang. *worst case*). Przyjęto przepustowość interfejsu równą 10Mbit/s, 48 interfejsów oraz 10 kontrolerów sieci definiowanej programowo. Do zobrazowania pracy sieci przyjęto domyślny czas obsługi zadań w przełączniku równych 5ms.

$$\Delta_j = 5ms * (1 + 0.75 + 0.48 + 0.5) = 13.65ms$$

$$\delta_i = 5ms * (1 + 0.75 + 0.5) = 11.25ms$$

$$k = 3 \quad n = 2$$

$$\varepsilon = 2.2ms$$

Przypadek 1:

$$C = 2 * (\sum_{i=1}^{3} 13.65ms + \sum_{j=1}^{2} 11.25ms) + 10 * 2.2ms = 148.9ms$$

Przypadek 2:

$$C = \sum_{i=1}^{k} \delta_i + \sum_{j=1}^{n} \Delta_j) = (\sum_{i=1}^{3} 13.65ms + \sum_{j=1}^{2} 11.25ms) = 63.45ms$$

Takie założenia zostały użyte ze względu na praktyczny charakter rozprawy, gdzie odpowiednie wielkości zostały sprawdzone w eksperymentach. Odnosząc model badawczy do dalszych eksperymentów można stwierdzić, że na jego podstawie można wyznaczyć parametry projektowanej sieci oraz wymagania dotyczące użytych urządzeń. Jako urządzenia należy rozumieć w tym miejscu zarówno przełączniki sieciowe, jak kontrolery sieci SDN czy same przełączniki SDN. W przypadku pierwszym komunikacja zostałaby zrealizowana w czasie 148.9ms co oznacza, że cała wymiana danych, łącznie z odczytaniem i interpretacją reguł kontrolera trwałaby prawie dwa razy dłużej niż w momencie już zdefiniowanych reguł, gdzie czas trwania wymiany określono na 63.45ms. Z obliczeń może wynikać, że wprowadzenie sieci SDN jest nieopłacalne (zarówno finansowo jak i pod względem charakterystyk czasowych). Uwzględniając jednak tylko jedną wydłużoną wymianę danych lub wprowadzenie urządzeń o innych parametrach może znacząco pomóc w działaniu samej sieci oraz w zarządzaniu nią z punktu widzenia administratora.

8 Koncepcja badań

Realizując prace analityczne wymagane było również poparcie pewnych założeń w sposób eksperymentalny. Z tego względu zaproponowane rozwiązania zostały przetestowane w kilku częściowych badaniach i eksperymentach. Starano się, aby całość przedstawiała ciąg sprawdzający kolejne istotne parametry możliwe do zmierzenia w systemach przemysłowych.

- Wykorzystanie urządzeń wbudowanych w systemach czasu rzeczywistego na podstawie Raspberry Pi i realizacji funkcji sterownika PLC
- Wprowadzenie do systemu dodatkowych urządzeń sieciowych: wpływ na komunikację sieciową
- Realizacja nowych funkcjonalności przez infrastrukturę sieci SDN
    - Dublowanie ruchu sieciowego: dynamiczna funkcja mirroringu,
    - Dynamiczne modyfikowanie topologii sieciowej,
    - Zarządzanie bezprzewodową infrastrukturą sieciową: protokoły przemysłowe w wersji bezprzewodowej.

8.1 Urządzenie wbudowanie a praca w systemach czasu rzeczywistego

Na rynku dostępnych jest wiele rozwiązań pozwalających na realizację programów sterujących procesem przemysłowym. Począwszy od najbardziej popularnych sterowników PLC, przez urządzenia wbudowane z systemem operacyjnym i uruchamianą specjalną aplikacją, a skończywszy na zdalnych programowalnych stacjach wejść/wyjść. Najtańsze, ale też najprostsze urządzenia bardzo często posiadają ograniczenia w możliwości realizacji komunikacji sieciowej, ilość przetwarzanych sygnałów wejść/wyjść, a także instalacji dodatkowych modułów. Przykładami bardzo prostych sterowników mogą być urządzenia firmy Siemens S7-1200. Nieznacznie różniące się jednostki główne nie posiadają takich funkcjonalności jak chociażby brak edytora języka Graph, a

komunikacja sieciowa nie zostanie zrealizowana w trybie izochronicznym zapewniającym odpowiednią synchronizację w przesyłaniu danych. Tego typu rozróżnienie urządzeń występuje w praktyce u wszystkich producentów. Kolejnym przykładem, na którym niejako bazowało podejście badawcze jest firma Beckhoff. Produkowanie przez tę firmę urządzenia są bardziej minikomputerami o różnym stopniu zaawansowania niż klasycznymi sterownikami PLC. Podstawowe urządzenia oparte o system operacyjny Windows CE zapewniają możliwość realizacji programów sterujących oraz prostej wizualizacji procesu. Wraz z rozwojem sprzętu i wykorzystywaniem coraz to nowszych technologii wprowadzone zostały nowe możliwości programowania, komunikacji sieciowej oraz tworzenia wizualizacji. Porównując moc obliczeniową dostępnych na rynku sterowników PLC i podobnych im urządzeń postanowiono sprawdzić możliwość realizacji tych samych zadań przez urządzenia wbudowane. Poniżej przedstawiono porównanie parametrów urządzeń wbudowanych do sterowników PLC.

Tabela 5 Porównanie parametrów profesjonalnych urządzeń wbudowanych do zastosowań przemysłowych oraz typowych urządzeń wbudowanych dostępnych na rynku.

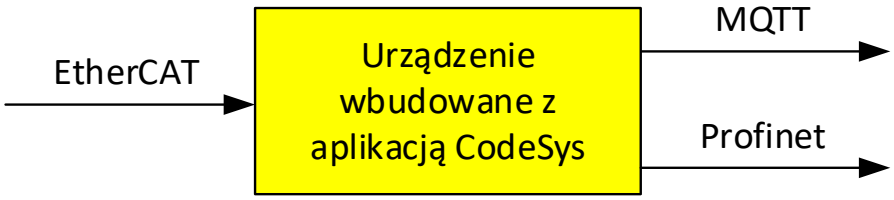
	Beckhoff CX2020	***Beckhoff CX8200***	***Raspberry Pi 4***	***Beagle Bone Black***
System operacyjny	Windows 7 Embedded	----	Raspabian (CodeSys runtime)	(CodeSys runtime)
Procesor	Jednordzeniowy Intel® Celeron® 827E 1.4 GHz	Dwurdzeniowy 1.2GHz ARM Cortex-A53	Czterordzeniowy 1.5GHz Quad core Cortex-A72	Jednordzeniowy 1GHz TI Sitara AM335x
Pamięć RAM	2GB	1GB	8GB	512MB
Zasilanie	24V DC	24V DC	5V DC	5V DC
RealTime	Tak	Tak	Brak	Programowalna jednostka czasu rzeczywistego

Porównując parametry urządzeń można dojść do wniosku, że różnice w specyfikacji sprzętowej nie powinny mieć znaczenia w poprawnym działaniu, zarówno jeśli sterowanie maszyną jest zlokalizowane w obrębie maszyny, jak również w sytuacji, gdy urządzenie pracuje w systemie rozproszonym. Różnicą w porównaniu do profesjonalnych rozwiązań stanowi brak jednostek czasu rzeczywistego oraz projektowanie urządzeń, nie w kontekście pracy w warunkach przemysłowych. Kompatybilność elektromagnetyczna [37], [38],[39], [40] w takim przypadku może mieć ogromne znaczenie, jeśli chodzi o stabilną pracę w miejscach, gdzie systemy sterujące są silnie narażone na działanie wpływu z innego rodzaju maszyn np. praca w sąsiedztwie silników zasilanych wysokim napięciem. Z tego samego względu zastosowanie samego urządzenia w profesjonalnych instalacjach wymaga przeprowadzenia szeregu testów i eksperymentów pozwalających określić przydatność konkretnej wersji urządzenia w celach profesjonalnych. Ze względu na różne możliwości oferowane przez urządzenia wbudowane zdecydowano się na przebadanie i porównanie otrzymanych wyników dwóch urządzeń: Raspberry Pi oraz Beagle Bone Black.[41], [42] W oby dwóch przypadkach urządzenia pracowały pod kontrolą systemu operacyjnego Linux z uruchomionym programem wykonywalnym przygotowanym w narzędziu CodeSys. Zaletą takiego rozwiązania jest wykorzystanie tego samego narzędzia programistycznego dla każdego przypadku. Z tego powodu do porównania wybrano platformę Beckhoff. Urządzenia te również posiadają schemat działania oparty o system operacyjny oraz uruchomiony program realizujący proces sterowania. Dodatkowo wspólnym elementem jest środowisko programistyczne oparte o narzędzie CodeSys.

Środowisko CodeSys pozwala na przygotowanie działania urządzenia w kilku trybach. Każdy z nich pozwala w inny sposób wyzwalać realizację programu co mimo braku wsparcia przez urządzenia wbudowane dla sprzętowej jednostki „Real Time” pozwalało na takie skonfigurowanie pracy urządzenia, aby możliwe było wykrycie ewentualnych problemów z realizacją samego programu. Tryby pracy sterowników PLC na bazie oprogramowanie CodeSYS zostały przedstawione poniżej.

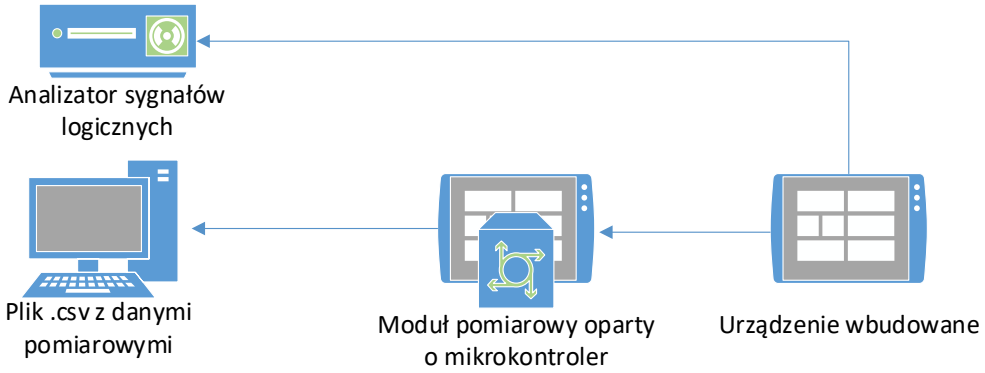
- ***Cyclic CODESYS*** – przetwarzanie odbywa się cyklicznie. Czas cyklu zazwyczaj jest stały, dobrany do funkcji realizowanej przez program. Najczęściej definiowany w konfiguracji urządzenia.
- ***Event CODESYS*** - rozpoczęcie przetwarzania danych w momencie wykrycia zbocza narastającego na wejściu zdefiniowanym w konfiguracji i przypisanym do konkretnej zmiennej.
- ***Freewheeling CODESYS*** – cykl pracy sterownika nie jest ściśle opisany. Rozpoczęcie procesu wykonywania programu następuję niezwłocznie po zakończeniu realizacji poprzedniego wywołania.
- ***Status CODESYS*** - program realizowany jest tak długo, jak długo wartość zmiennej użytej do sterowania cyklem jest w stanie "PRAWDA".
- ***External CODESYS*** – realizacja programu następuję po wykryciu wystąpienia "zdarzenia" zdefiniowanego w konfiguracji. W zależności od jednostki głównej rodzaje zdarzeń do obserwacji mogą się różnić.

Do prowadzenia eksperymentów starano się wybierać konfiguracje jak najbardziej przypominające rozwiązania mające praktyczne zastosowanie. Jednak pewne elementy pozwalały na zaproponowanie wykorzystania konkretnych funkcjonalności do realizacji zadań nie objętych programem badań. Takimi przypadkami są w tej części tryby pracy urządzenia wykorzystujące bądź to sygnał zewnętrzny, bądź stan zmiennej. A zaproponowanym rozwiązaniem jest użycie zbudowanego „sterownika PLC" jako bramy wieloprotokołowej, czyli urządzenia pozwalającego na przesyłanie danych pomiędzy częściami systemu wykorzystującymi różne protokoły sieciowe.



Rysunek 8.1 Zasada działania bramy wieloprotokołowej opartej na urządzeniu wbudowanym

W celu sprawdzenia możliwości pracy urządzeń wbudowanych w warunkach czasu rzeczywistego zbudowano stanowisko laboratoryjne składające się z oprócz badanego układu z analizatora sygnałów wyjściowych oraz mikrokontrolera zbierającego dane i pozwalającego na wygenerowanie pliku .csv.

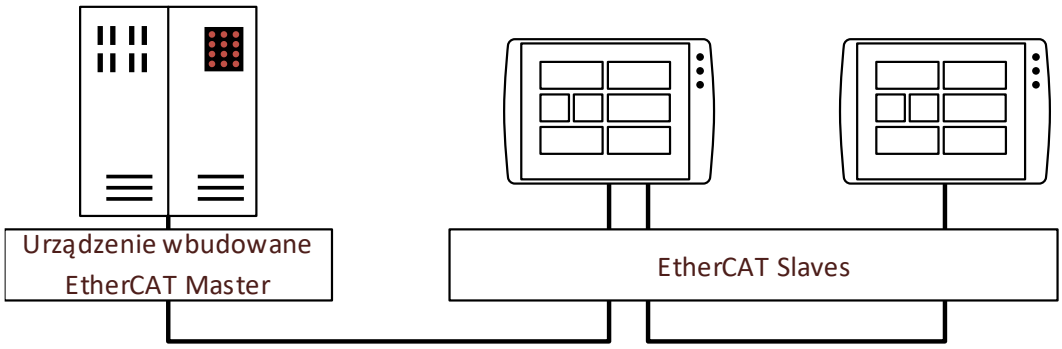


Rysunek 8.2 Stanowisko laboratoryjne do analizy szybkości działania urządzeń wbudowanych z programem sterującym typowym dla sterownika PLC.

Przeprowadzona w ten sposób analiza była w stanie odpowiedzieć na pytanie jak zachowuje się układ wbudowany w pracy, gdzie wymogiem jest spełnienie założeń co do czasu realizacji programu sterującego w czasie rzeczywistym. W badaniach wykorzystano dwa rodzaje urządzeń w celu potwierdzenia poprawności zbieranych danych. Układ oparty o mikrokontroler przesyłał dane do anlizy komputerowej dodatkowo analizator sygnałów pozwalał na bieżąco w czasie trwania eksperymentów

obserwować poprawność generowanego sygnału wyjściowego. Zdecydowano o podwójnym potwierdzaniu sygnału wyjściowego ze względu na możliwe przekłamania na linii Atmega, a Raspberry Pi/BeagleBoneBlack. Możliwość działania tego typu urządzeń potwierdzono w dwóch konfiguracjach: BeagleBone Black oraz Raspberry Pi. [43]–[45]

Po dokonaniu sprawdzenia możliwości działania urządzenia wbudowanego jako sterownika PLC, postanowiono zrealizować model sieci przemysłowej oparty o urządzenie wbudowane pełniące rolę zarządzającą cała infrastrukturą. Do realizacji wybrano sieć typu EtherCAT z dwoma urządzeniami typu Slave.



Rysunek 8.3 Schemat sieci EtherCAT wykorzystywanej do testów z urządzeniami wbudowanymi.

8.2 Zastosowanie SDN w systemach czasu rzeczywistego

Zastosowanie technologii SDN [46] w systemach przemysłowych nie do końca wpisuje się w działanie zarówno jednego rozwiązania, jak i drugiego. Technologia SDN stworzona głównie do zarządzania dużymi centrami danych w zdecydowanie mniejszych instalacjach nie powinna mieć racji bytu. Podobnie w bardzo hermetycznych systemach przemysłowych wprowadzanie rozwiązania niededykowanego do konkretnego rodzaju instalacji może zaburzyć pracę urządzeń, a w ostateczności całości systemu. Istnieją jednak sytuacje, gdzie zastosowanie koncepcji SDN pozwoliłoby w znaczący sposób pomóc w uzyskiwaniu danych użytecznych zarówno do diagnostyki, jak i analizy i doboru odpowiednich parametrów.

Tabela 6 Konfiguracja symulatora Mininet

mininet> ovs-ofctl add-flow s1 in_port=1, actions=output:3 tp_dst=502, nw_src=’192.168.2.232/24’ mininet> ovs-ofctl add-flow s1 in_port=2, actions=output:3 tp_src=502, nw_dst=’192.168.2.232/24’ mininet> ovs-ofctl add-flow s1 in_port=2, actions=output:3 dl_type=0x88A4 mininet> ovs-ofctl add-flow s1 in_port=1, actions=output:3 dl_type=0x88A4

```
mininet> ovs-ofctl add-flow s1 in_port=2, actions=output:3 dl_type=0x8892
mininet> ovs-ofctl add-flow s1 in_port=1, actions=output:3 dl_type=0x8892
```

W *Tabeli 6* zaprezentowano możliwość konfiguracji symulatora Mininet pozwalające na zdefiniowane konkretnego portu wejściowego (in_port), akcję jaka ma być wykonana z odebranymi danymi (output:…; port wyjściowy) oraz rodzaj przesyłanych ramek Ethernet (dl_type), czyli możliwość filtrowania ruchu sieciowego. Innymi przypadkami są filtrowanie według numeru portu (tp_src) czy przesyłania danych pod konkretny adres IP (nw_dst).

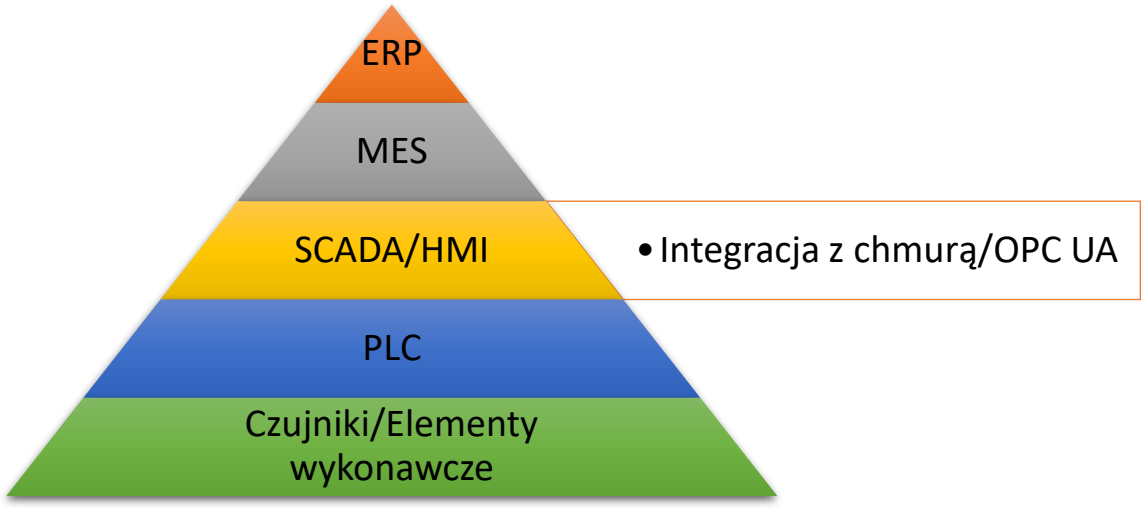
8.3 Pozyskiwanie danych przy użyciu SDN

Wszechobecne rozwiązania chmurowe w połączeniu z technologią sztucznej inteligencji pozwalają na coraz to dokładniejsze przewidywanie zachowań systemów przemysłowych w kontekście możliwości wystąpienia awarii. Jak wspomniano we wstępie większość systemów przemysłowych jest hermetyczna i wprowadzanie dodatkowych urządzeń jest niemożliwe. Integracja wielu systemów przemysłowych staje się nie lada wyzwaniem. Duża ilość nie tylko samych urządzeń, ale również wykorzystywanych rozwiązań do zestawienia komunikacji to w praktyce poważny problem. Problem staje się jeszcze poważniejszy w momencie rozbudowy istniejącego systemu. Brak możliwości zmiany konfiguracji sprzętowej czy wprowadzanie zmian w aplikacji sterującej wymaga stosowania dodatkowych elementów takich jak zduplikowane czujniki czy informacje o aktualnym stanie przekaźników wyjściowych. Uproszczeniem w takiej sytuacji może być stosowanie w systemach protokołów pozwalających na integrację wielu protokołów.

Jednym z takich rozwiązań jest protokół OPC UA (ang. OPC Unified Architecture), dzięki wspieraniu najpopularniejszych protokołów sieciowych możliwa staje się komunikacja z elementami systemów opartych o wytyczne związane z koncepcją Przemysłu 4.0.

A dzięki wykorzystaniu modelu Producent/Konsument nie jest wymagane ciągłe utrzymywanie połączenia. Dane rozprowadzane w takim schemacie

działania mogą być dostarczane w sposób optymalny ze względu rozgłaszanie wszystkich informacji w sieci, jednak odbierane są tylko dane użyteczne dla danego węzła.



Rysunek 8.4 Piramida prezentująca miejsce zbieranie danych przez przełącznik SDN

Wśród wielu zalet takiego rozwiązania mamy ciągle poważny problem do rozwiązania w postaci dostępu do danych w już istniejących systemach. Dostępne urządzenia sieciowe w znakomitej większości pozwalają na tzw. mirroring, czyli przesyłanie danych na dwa porty jednocześnie. Takie rozwiązanie sprawdza się w sieciach, gdzie urządzenia nie weryfikują integracji sieci czy źródła pochodzenia danych. Wykorzystanie mirroringu pozwala na skopiowanie danych z portu lub kilku portów przełącznika sieciowego na inny port. Taka funkcjonalność w zupełności jest wystarczająca, jeśli potrzebujemy użyć danych do analizy i/lub diagnostyku działania systemu. Korzystanie z funkcjonalności kopiowania danych na zasadzie „jeden do jeden" pozwala również na wykorzystanie otrzymanych danych do niepożądanych działań np. ataki DDoS z powodu możliwości uruchomienia komunikacji zdublowanej w dwóch kierunkach.

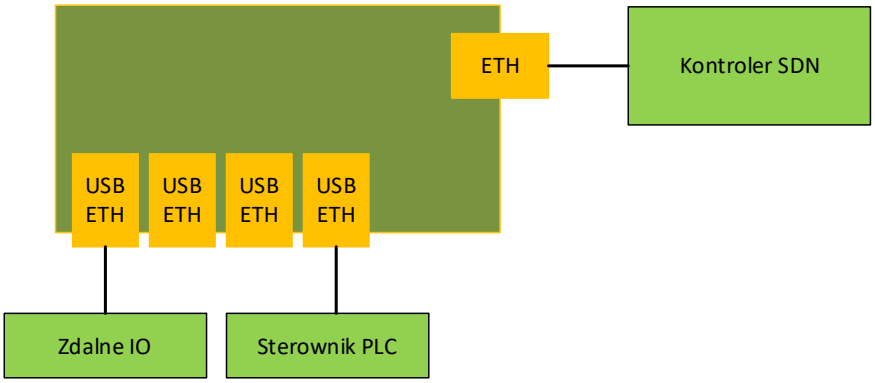
Między innymi z tego powodu analizie poddano możliwość użycia infrastruktury programowalnej SDN do pozyskiwania konkretnych danych użytecznych i przesyłania ich do urządzeń diagnostycznych lub innych urządzeń wykonawczych. Podstawowymi założeniami wykonanych eksperymentów była możliwość zarówno zdublowania ruchu sieciowego, ale również odpowiednie

przefiltrowanie i uzyskanie ruchu sieciowego użytecznego w innym częściach systemu. [47], [48]

Ze względu na ograniczenia czasowe sprzętu wykorzystywanego w systemach przemysłowych niemożliwe stało się użycie np. przełączników sieciowych dedykowanych do pracy w tego typu sieciach.

Po przeanalizowaniu możliwości wykorzystania urządzeń wbudowanych jako sterowników PLC i pracy w systemach czasu rzeczywistego zdecydowano do eksperymentów wykorzystać minikomputery z systemem operacyjnym Linux (Raspbian).

Do zbudowania przełącznika sieciowego skorzystano z minikomputera Raspberry Pi wyposażonego w dodatkowe karty sieciowe Ethernet podłączone za pomocą interfejsów USB 3.0 *(Rysunek 9.5)*. Zastosowanie kart sieciowych w specyfikacji USB 3.0 pozwala na transmisje danych z prędkością około 500Mb/s, co w połączeniu z urządzeniami sieciowymi skonfigurowanymi do pracy w trybie Fast Ethernet 100Mb/s. Korzystając z takiego rozwiązania umożliwiono ograniczenie wpływu połączenia USB Ethernet do minimum.



Rysunek 8.5 Schemat przełącznika sieciowego zbudowanego na bazie urządzenia wbudowanego Raspberry Pi

Wprowadzenie do systemu dodatkowych odseparowanych interfejsów pozwoliło na uniknięcie problemu konfiguracji połączeń typu VLAN. Konfigurację sprzętową przedstawiono w *Tabeli 7.*

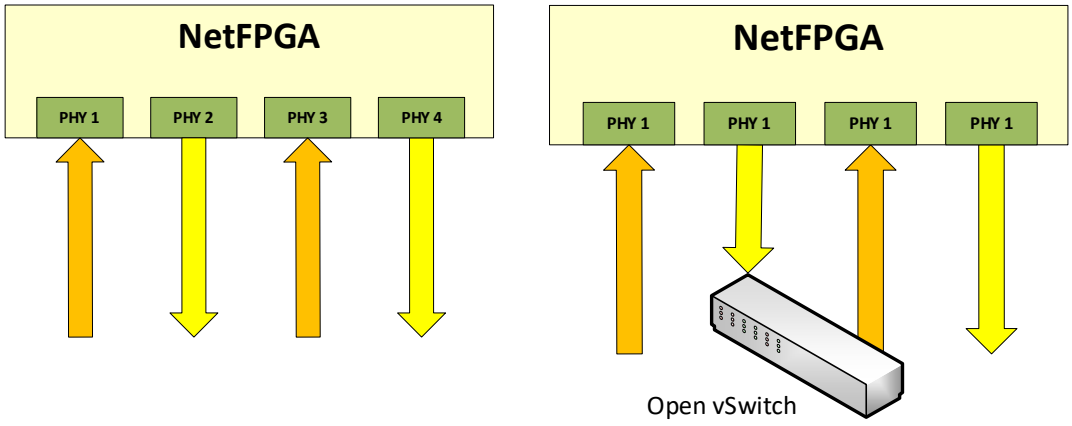
Tabela 7 Konfiguracja interfejsu sieciowego pracującego w sieci SDN

enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500 inet 192.168.0.134 netmask 255.255.255.0 broadcast

```
192.168.0.255  inet6  fe80::977c:b416:e26f:96df  prefixlen
64  scopeid 0x20<link>
ether 08:00:27:c9:c8:22  txqueuelen 1000  (Ethernet)
```

Każdy z interfejsów został dodany do wirtualnego przełącznika Open vSwitch, który zarządzany jest przez kontroler SDN.

W badaniach wykorzystano płytę programowalną NetFPGA w celu zarejestrowania ruchu sieciowego. Do dalszej analizy zaproponowano wykorzystanie oprogramowania Wireshark w celu sprawdzenia wprowadzonego opóźnienia przez zbudowany przełącznik sieciowy. Sposób połączenia urządzeń został przedstawiony na *Rysunku 9.6*.



Rysunek 8.6 Schemat podłączenia przełącznika SDN w infrastrukturze pozwalającej na dublowanie ruchu sieciowego

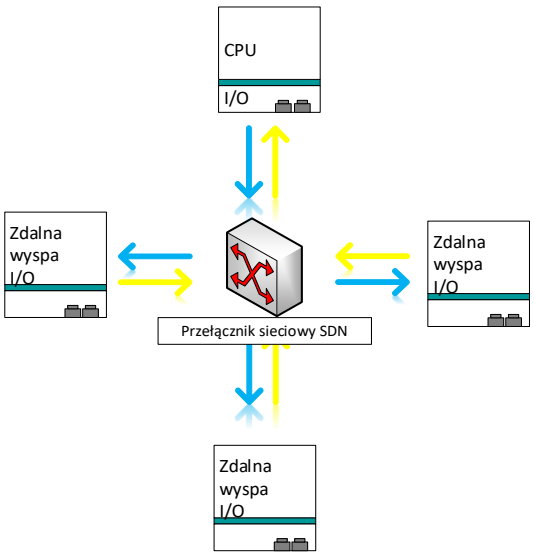
W przypadku pierwszym zrealizowano bezpośrednie połączenie pomiędzy interfejsami PHY2 oraz PHY3. Rolę medium w tym przypadku pełnił możliwie krótki przewód typu patchcord. Zapisywanie ruchu sieciowego wraz ze stemplami czasowymi pozwoliło na określenie opóźnienia wprowadzanego przez platformę NetFPGA. Przypadek ten nie uwzględniał żadnego z elementów infrastruktury sieciowej SDN.

W części drugiej w miejscu połączenia kablowego wprowadzono przełącznik sieciowy SDN. Porównując wyniki uzyskane w oby dwu przypadkach możliwe stało się określenie wpływu dodatkowego czasu wprowadzonego do systemu przez przełącznik SDN. Ważną częścią projektowanej konfiguracji jest

możliwość kopiowania danych pomiędzy portami i filtrowania niepotrzebnych w danym momencie danych.

8.4 Programowanie logicznej topologii sieciowej

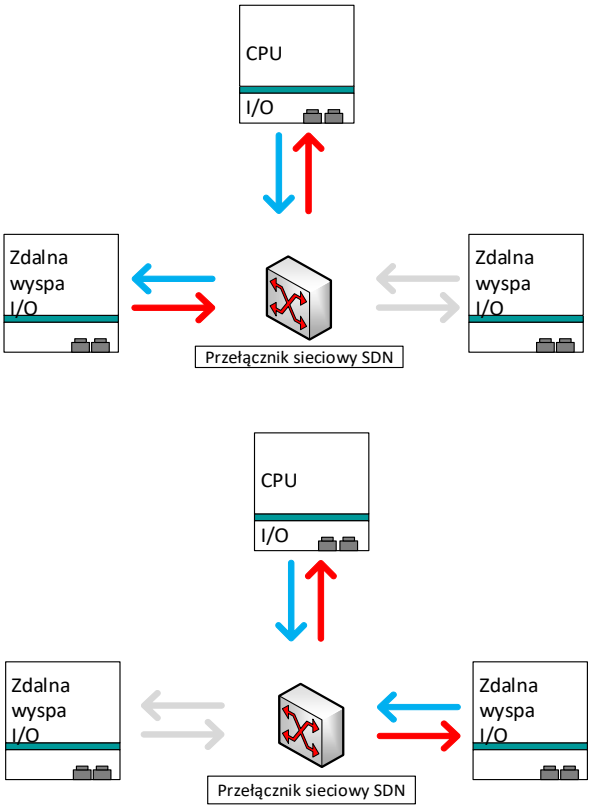
Wiele sieci przemysłowych wykorzystuje zdefiniowaną topologię sieciową. Jeżeli nie jest to topologia wymuszona sprzętowo, to jest ona dopasowywana logicznie przy pomocy odpowiednich algorytmów i narzędzi. Ze względu na ograniczenia nie zawsze istnieje możliwość instalacji urządzeń z dedykowaną topologią.[49]–[51] W takim przypadku wymagane jest przeanalizowanie dostępnych możliwości, co w rezultacie może zakończyć się wyborem zupełnie nowej platformy sprzętowej. Zaproponowane podejście pozwala na zmodyfikowanie topologii sieciowej dynamicznie w przełączniku sieciowym SDN w taki sposób, aby możliwe było uruchomienie wszystkich urządzeń bez fizycznej modyfikacji topologii sieciowej. *(Rysunek 9.7)* Zaletą takiego rozwiązania może być łatwość w przełączaniu sterowania pomiędzy maszynami, a także łatwiejsze kierowanie ruchu sieciowego do konkretnego urządzenia podłączonego bezpośrednio do przełącznika sieciowego.



Rysunek 8.7 Schemat proponowanej topologii sieciowej

Zastosowanie programowalnej topologii sieciowej pozwala nie tylko na łatwą zmianę instalacji urządzeń, ale również na modyfikację drogi przesyłania

informacji w momencie wykrycia awarii w sieci. Eksperymenty związane z modyfikacjami topologii sieciowej zostały przeprowadzone w sieciach typu EtherCAT. Do badań użyto profesjonalnego sterownika PLC (Master) wraz z urządzeniami typu slave oraz przełączniki sieciowe Open vSwitch. Jak wspomniano we wcześniejszych rozdziałach, sieci EtherCAT posiadają element wspólny z klasycznymi sieciami jakim jest adres MAC. Na tej podstawie zostały przygotowane reguły kontrolera sieci SDN pozwalające na odpowiednią zmianę urządzenia docelowego przesyłanych danych. Korzystając z interfejsu webowego skonfigurowano połączenie początkowo tylko z dwoma urządzeniami, a w kolejnym kroku zrealizowano modyfikację topologii z liniowej w gwiazdę.



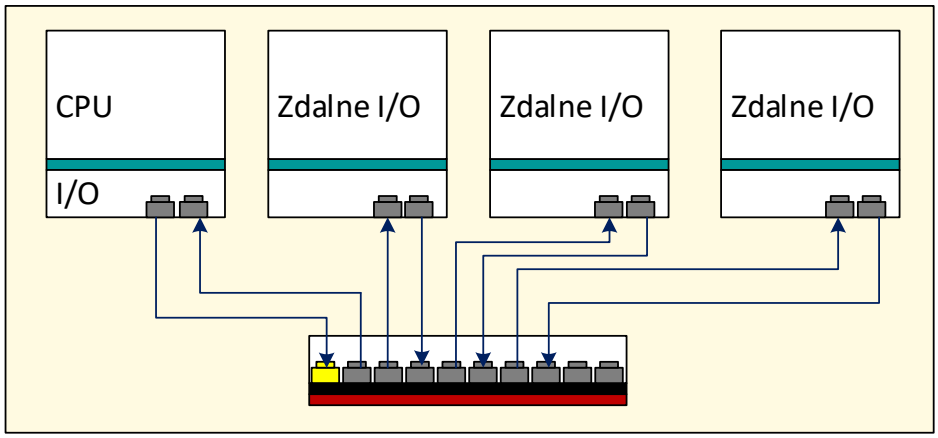
Rysunek 8.8 Logiczna zmiana topologii sieciowej przy użyciu przełączników SDN

Rysunek 9.8. przedstawia sposób połączenia i aktywowania kolejnych urządzeń. Przedstawione poniżej wpisy kontrolera pozwalają na zmianę aktywnego urządzenia i przesyłanie informacji do innych wysp wejść/wyjść.

ovs-ofctl add-flow s1 in_port=2, actions=output:3

ovs-ofctl add-flow s1 in_port=2, actions=output:4

Na kolejnym etapie do infrastruktury sieciowej wprowadzono dodatkowe urządzenia i zmodyfikowano topologię z klasycznej liniowej na topologię typu gwiazda. Wszystkie urządzenia podłączone zostały do przełącznika sieciowego za pomocą dwóch przewodów.



Rysunek 8.9 Prezentacja połączeń w przełączniku SDN wraz z kierunkiem przesyłu informacji

Taki sposób podłączenia został wybrany ze względu na realizację topologii liniowej w przełączniku i sposób przekierowywania ruchu w jednym punkcie sieci – przełączniku sieciowym SDN. Sposób realizacji zmiany topologii został przedstawiony w krokach przedstawionych poniżej.

1. Master EtherCAT rozpoczyna komunikację i przesyła ramkę do przełącznika SDN, a przełącznik przekazują ją do urządzenia typu Slave.

   ovs-ofctl add-flow s1 in_port=1, actions=output:3

2. Ramka przesyłana jest ponownie do przełącznika sieciowego SDN, który przekazuję ramkę do kolejnych urządzeń typu Slave.

   ovs-ofctl add-flow s1 in_port=4, actions=output:5

3. Po obsłużeniu wszystkich urządzeń w analogiczny sposób ramka ponownie przekazywana jest do urządzenia typu Master.

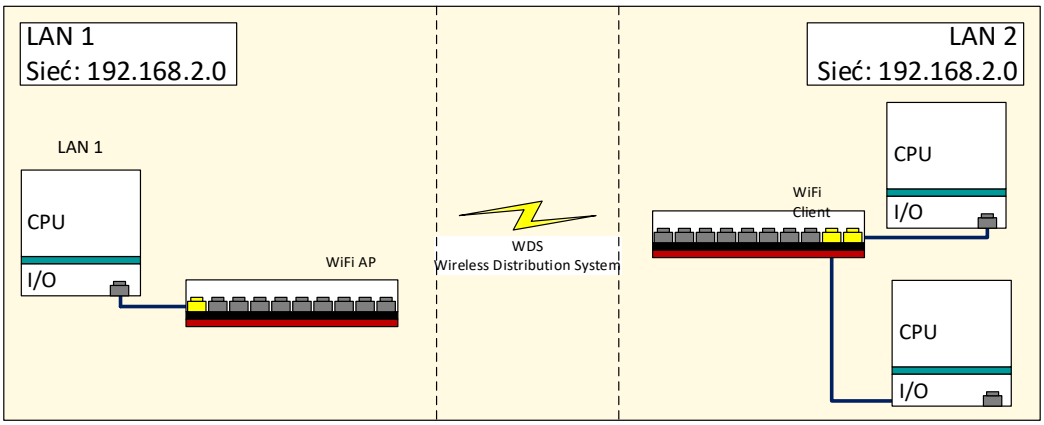
   ovs-ofctl add-flow s1 in_port=8, actions=output:2

Takie rozwiązanie generuje problem badawczy w postaci analizy czasów komunikacji i obsługi całej sieci. Zebrane dane oraz wyniki zostały przedstawione w dalszej części skupiającej się na ich przedstawieniu i dalszej analizie.

Wyniki badań zostały omówione i przedstawione również w publikacji „……”, w której dokładniej poruszono zagadnienia związane z samą problematyką działania sieci EtherCAT i możliwościami dalszych badań tj. przedstawiona w kolejnych rozdziałach komunikacja bezprzewodowa.

8.5 Uruchomienie bezprzewodowej komunikacji EtherCAT

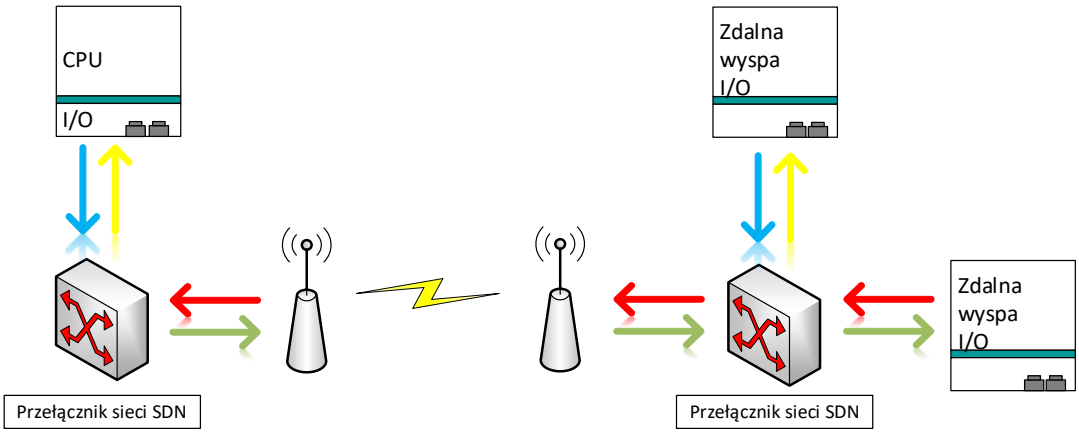
Rozproszenie systemów przemysłowych powoduje, że komunikacja przewodowa jest znacznie utrudniona. [52]–[54], [55] W takich przypadkach możliwe byłoby zastosowanie łączności bezprzewodowej, której szybkość przy zastosowaniu sieci WiFi jest minimum na poziomie powszechnie stosowanych interfejsów 100Mb/s. Problemem w większości instalacji są jednak generowane przez środowisko pracy zakłócenia oraz zasada działania protokołów sieciowych. Realizowane połączenie powinno pozwalać na przesyłanie informacji w sposób transparentny tzn. bez modyfikacji zawartości ramki. Taką możliwość zapewnia tryb pracy WDS (Wireless Distribution System), czyli mostu bezprzewodowego łączącego ze sobą dwie części tej samej sieci. Zdarza się, że tryb ten posiada ograniczenia co do liczby realizowanych połączeń w tym samym momencie, najczęściej dostępna jest możliwość podłączenia tylko 6 urządzeń klienckich. Sposób działania został przedstawiony na *Rysunku 9.10*.



Rysunek 8.10 Schemat połączenia bezprzewodowego pomiędzy urządzeniami w sieci EtherCAT

Brak transparentności realizowanych połączeń powoduje, że rozwiązania sieciowe tj. EtherCAT czy oparte o protokół Profinet nie są w stanie rozpocząć komunikacji z powodu problemów związanych z przesyłaniem danych konfiguracyjnych. Problem taki nie występuję w momencie użycia protokołów pracujących na warstwie aplikacji (warstwa 3)

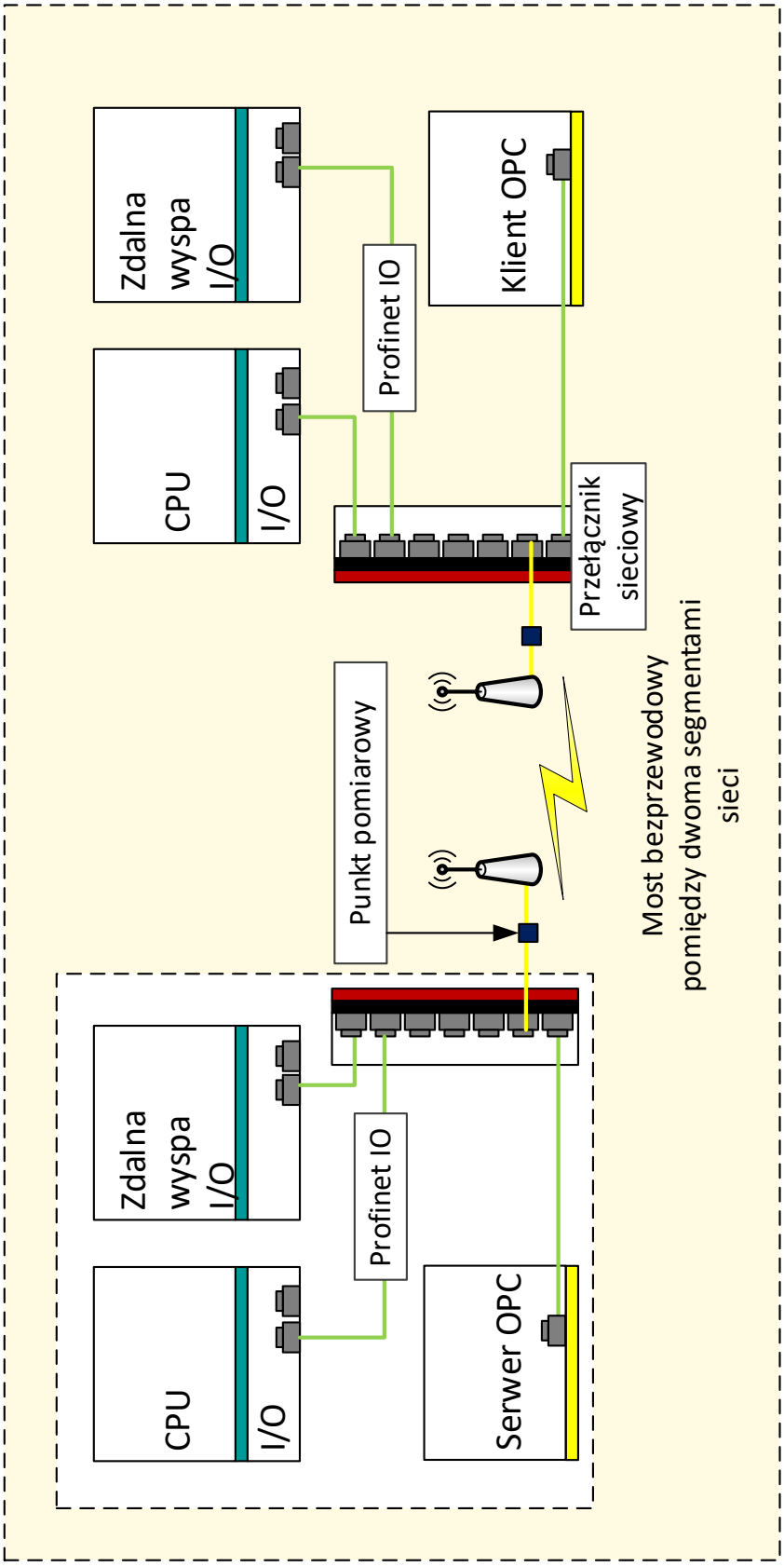
Drugą możliwością wykorzystaną w badaniach jest zastosowanie przełączników sieciowych pozwalających na dynamiczne zmiany konfiguracji. W tym przypadku zastosowane zostały przełączniki, pozwalające na zmianę adresów MAC w transmitowanej ramce przez co po każdej stronie sieci mam



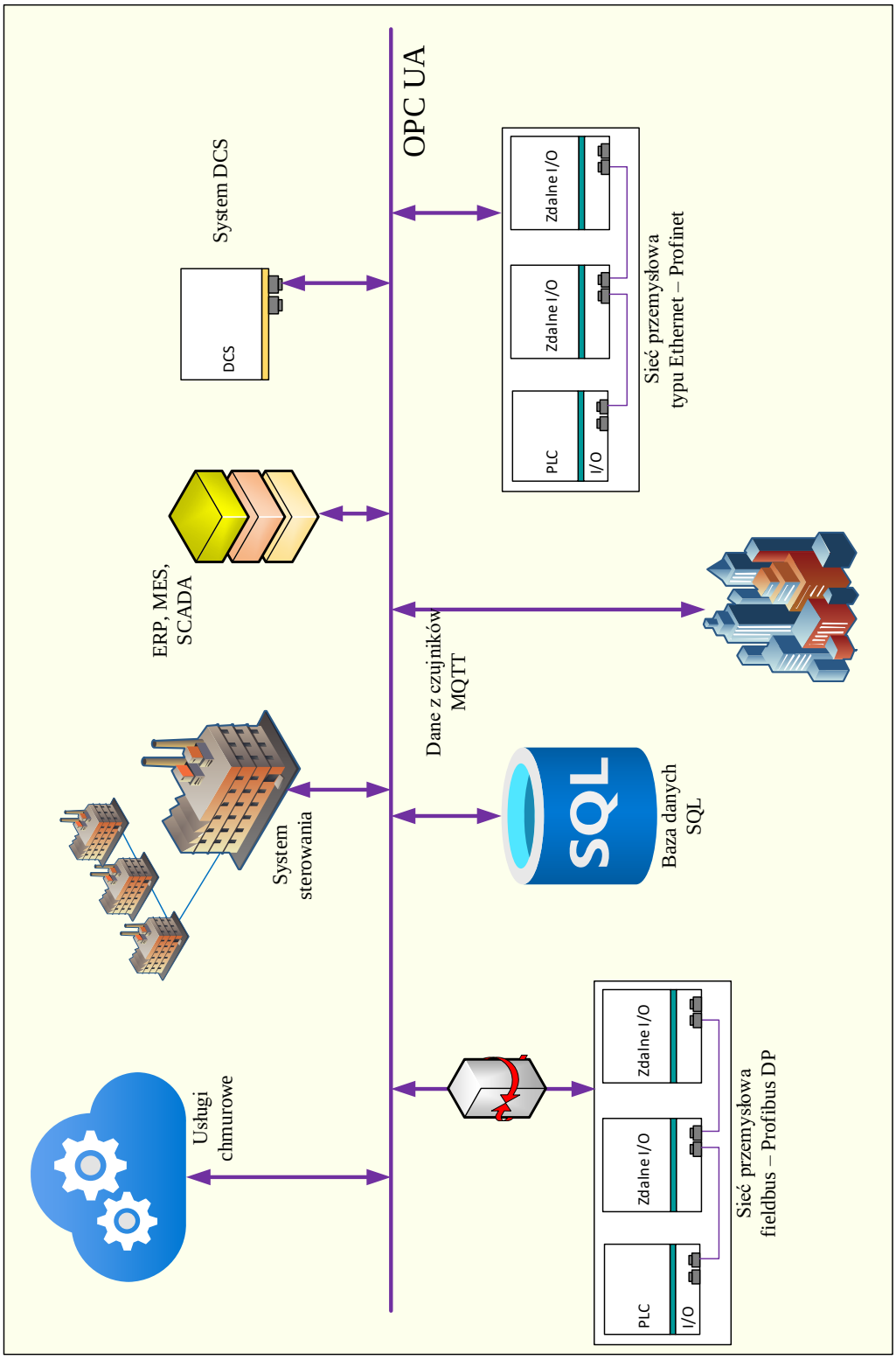
Rysunek 8.11 Komunikacja bezprzewodowa pomiędzy dwoma segmentami sieci przemysłowej

8.6 Inne rozwiązania - OPC UA (OPC Unified Architecture)

Zaproponowane rozwiązania wykorzystujące sieci SDN oraz urządzenia wbudowane pozwalające na integrację systemów przemysłowych można spróbować porównać do technologii, które już są powszechnie dostępne na rynku. Przykładem takiego systemu jest rozwiązanie OPC UA.[56]–[59] W systemie OPC UA możliwe jest zintegrowanie kilku systemów i w zależności od zastosowanych protokołów komunikacyjnych czy urządzeń sterujących możliwe jest pobieranie danych bezpośrednio z nich.



Rysunek 8.12 Schemat sieci wykorzystywanej do połączenia różnych standardów przesyłania danych przy użyciu tej samej bezprzewodowej infrastruktury sieciowej



Rysunek 8.13 Schemat prezentujący elementy integrowane przez rozwiązanie OPC UA

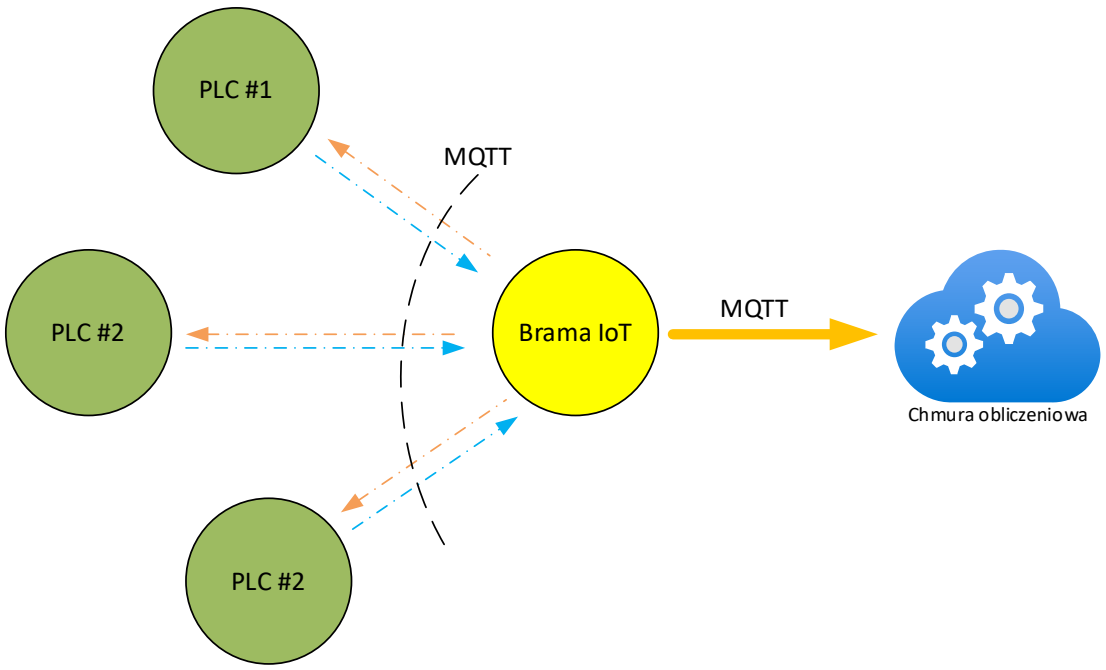
Komunikacja w systemach OPC UA odbywa się na zasadzie połączenia „maszyna do maszyny”, a dzięki bazowaniu na protokołach TCP/IP jest rozwiązaniem uniwersalnym działającym niezależnie od zastosowanej platformy. Możliwość komunikacji z wykorzystaniem OPC UA implementowana jest w większości platform wykorzystywanych do sterowania procesem przemysłowym (np. sterownikach PLC, urządzeniach brzegowych), robotach przemysłowych czy rozwiązaniach chmurowych odpowiedzialnych za przetwarzanie danych. Połączenia klientów z serwerem mogą być dodatkowo zabezpieczane poprzez używanie zapory ogniowej oraz stosowanie różnych poziomów weryfikacji użytkownika np. certyfikatów. Zastosowanie OPC UA nie wyeliminuje jednak problemów związanych z narzutem czasowym powstałym przy dodaniu dodatkowego elementu w postaci serwera OPC.

8.7 Przemysłowy Internet Rzeczy

Przemysłowy Internet Rzeczy i technologie związane z Przemysłem 4.0 niosą ze sobą wiele zagrożeń związanych przede wszystkim z bezpieczeństwem działania systemów. Wprowadzanie nowych technologii zawsze jednak wiąże się z zagrożeniami, jednak ich wyeliminowanie bądź chociaż zminimalizowanie mimo wszystko jest w stanie przynieść wiele korzyści. Każdy system przemysłowy jest stanie dostarczyć ogromnych ilości danych, które w klasycznych instalacjach nie są w żaden sposób analizowane. Wykorzystanie informacji o czasie pracy maszyny, ilości zużytej energii czy ewentualnych uszkodzeniach np. wibracjach łożyska, jest w stanie poprawić proces produkcji części zamiennych czy dobierania nowych parametrów pracy układu. W systemach Internetu Rzeczy stosowanych w domach wykorzystuje się najczęściej protokół MQTT[60], [61], stosowany również w przemysłowym zastosowaniu tej technologii. Dodatkowo urządzenia pozwalające na przekazywanie danych do systemów chmurowych wykorzystują również takie rozwiązania jak OPC UA czy REST. Niektórzy producenci zapewniają wsparcie dla protokołów przemysłowych wspieranych przez inne swoje urządzenia tj. Profinet, Modbus czy EtherCAT. W tej części poddano analizie możliwości integracji systemów przemysłowych z

rozwiązaniami chmurowymi i sposoby dystrybucji danych oraz analizy chmurowej. Zaproponowano podejście łączące funkcjonalności dostępnego na rynku sprzętu z możliwościami technologii SDN i urządzeń wbudowanych poruszonymi w poprzednich częściach.

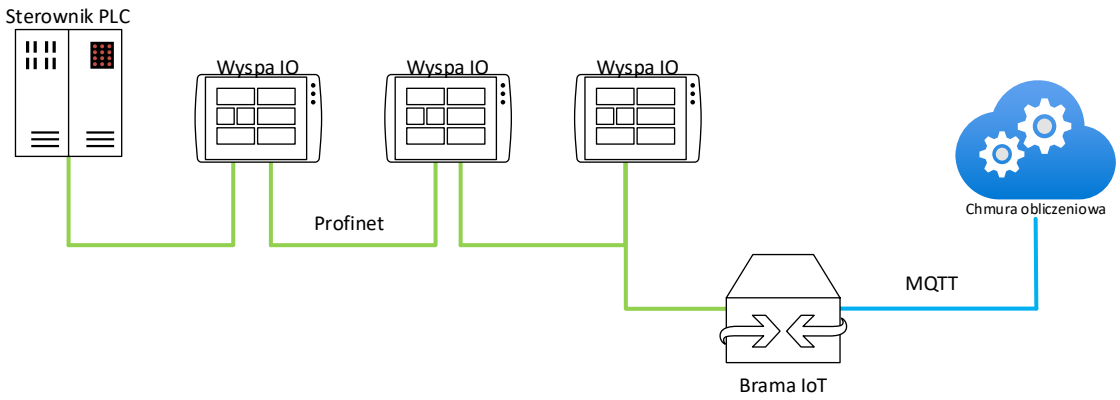
Do przeprowadzenia badań użyto sprzętu pozwalającego na zbieranie danych z systemów przemysłowych i przekazywanie ich do chmury obliczeniowej. Sprawdzono kilka konfiguracji różniących się stopniem zaawansowania oraz możliwościami personalizacji komunikacji. W podejściu pierwszym skorzystano z bramy przemysłowej pobierającej dane z systemu przemysłowego przy użyciu modułów IO dedykowanych dla sieci EtherCAT. W tej części przesyłano dane binarne przy użyciu bramy IoT firmy Beckhoff EK9160. Mimo, że sama wyspa nie zapewnia połączenia z typowymi protokołami sieciowymi. Dostępność bloczków programowalnych do zrealizowania komunikacji MQTT w sterowniku PLC pozwala natomiast na wyodrębnienie użytecznych danych i zestawienie połączenia pomiędzy sterownikiem, a bramą IoT.



Rysunek 8.14 Zasada działania bramy Przemysłowego Internetu Rzeczy

Taki rodzaj połączenia zapewnia możliwość komunikacji z chmurą i nieobciążanie dodatkowymi zadaniami sterownika PLC. Czym niewątpliwie byłoby połączenie z chmurą obliczeniową będące „otwarciem systemu na świat" i możliwością przeprowadzenia ataku sieciowego.

Podejście drugie uwzględniało bezpośrednio wprowadzenie do systemu przemysłowego dodatkowego urządzenia, będącego bramą do połączenia z chmurą i jednocześnie integratorem całego systemu. Dzięki zastosowaniu urządzenia z systemem operacyjnym Linux, możliwe było wykorzystanie programu wykonywalnego CodeSys. Dzięki takiemu podejściu dane są zbierane bezpośrednio z systemu i przekazywane do usługi chmurowej. Zastosowanie programu CodeSys rozszerza niejako możliwości urządzenia, pozwalając na skonfigurowanie i zestawienie połączeń przy użyciu najpopularniejszych protokołów tj. Profinet, EtherCAT, EthernetIP, Modbus TCP. *(Rysunek 9.15)*



Rysunek 8.15 Zasada działania bramy Przemysłowego Internetu Rzeczy

Zaproponowano rozwiązanie problemów związanych z atakami sieciowymi przy użyciu elementów sieci SDN zaproponowanych we wcześniejszych częściach pracy. Takimi funkcjonalnościami jest blokowanie połączeń, aktywowanie portu tylko na określony czas czy wykorzystywanie zdublowanego ruchu sieciowego zamiast rzeczywistego połączenia. Badania przeprowadzane w tej części zawierały się w innych częściach badawczych.

9 Prezentacja wyników badań

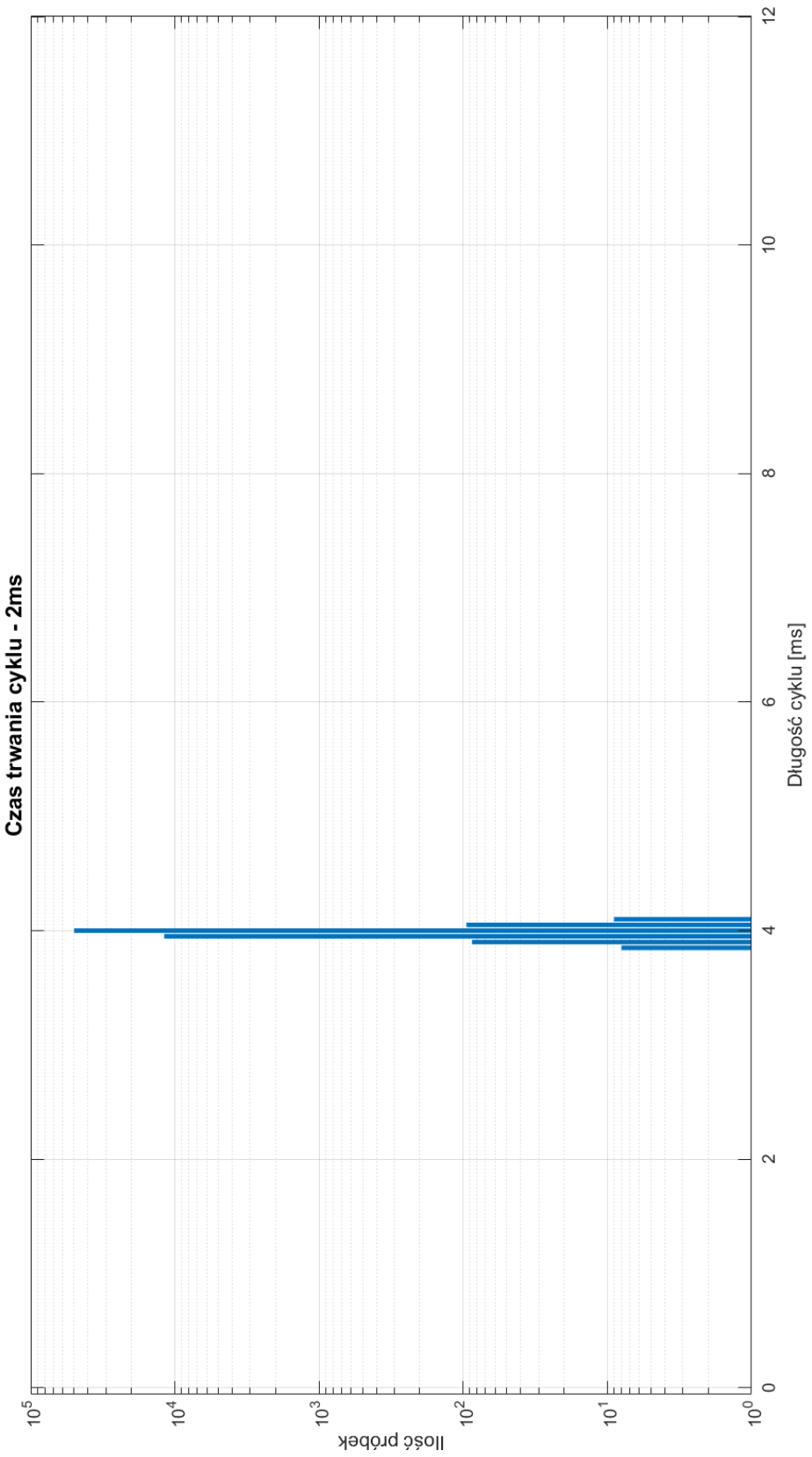
Podobnie jak w poprzednim rozdziale także tę część podzielono ze względu na rodzaj testowanych parametrów. W tej części skupiono się na omówieniu uzyskanych wyników badań i ich odniesieniu do oczekiwanych wartości mających zastosowanie w systemie przemysłowym.

9.1 Wykorzystanie urządzeń wbudowanych w systemach czasu rzeczywistego

Sprawdzenie działania urządzeń wbudowanych pracujących jako sterownik PLC w systemach czasu rzeczywistego pozwoliło określić działanie urządzeń w rygorze czasowym. Sprawdzone zostało działanie urządzenia w momencie realizacji programu sterującego w cyklach o krótkim czasie trwania. Ze względu na realizację badań na wczesnym etapie, czas trwania cyklu nie jest ściśle powiązany z szybkością działania połączeń sieciowych.

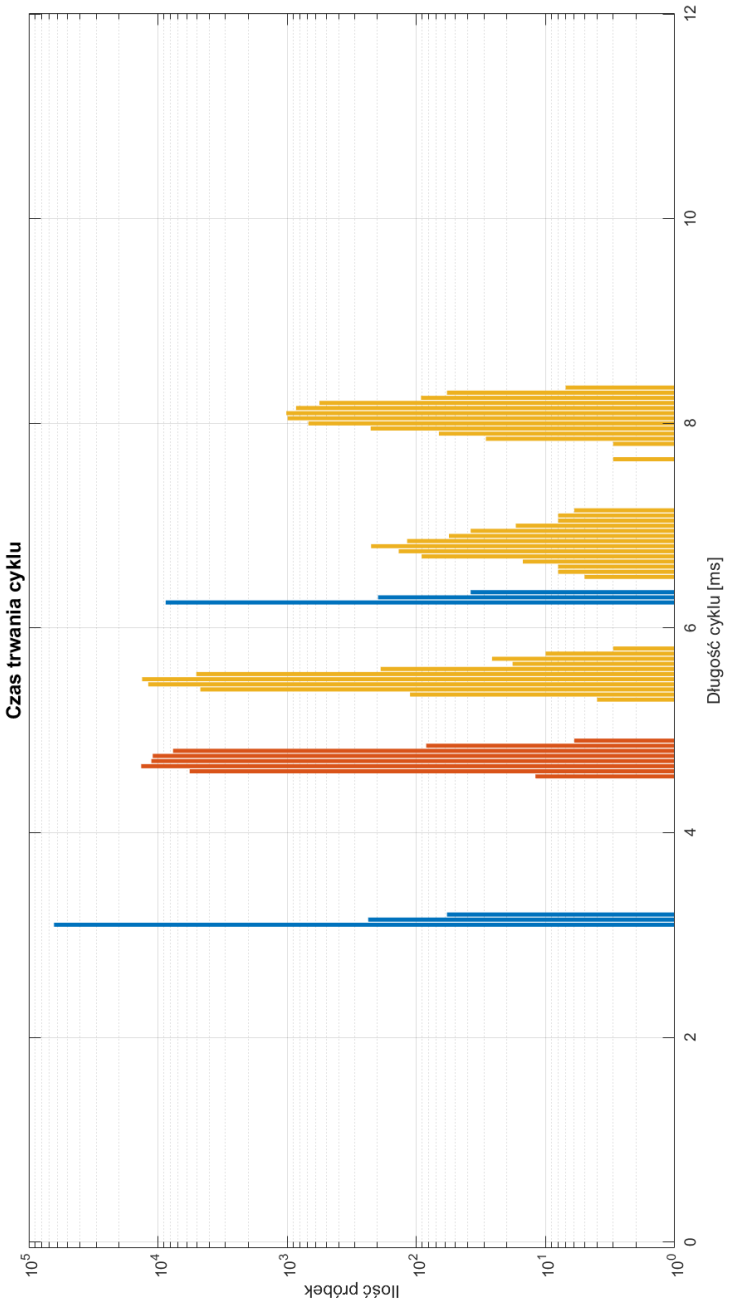
9.1.1 Urządzenie wbudowane Raspberry Pi

Skonfigurowany 2ms cykl pracy sterownika nie został osiągnięty w przebadanej konfiguracji. Wszystkie pomiary zostały zarejestrowane przy dwukrotności ustawionego cyklu – 4ms. Pomiary te zostały zarejestrowane dla konfiguracji pracy bez obciążenia, czyli bez realizacji dodatkowych pętli programu. Mimo, że ustawiony cykl pracy sterownika PLC na wartość 2ms wydaje się bardzo krótki, obrazuje on już w tym momencie możliwe opóźnienia występujące w momencie realizacji funkcji sterujących siecią SDN (urządzenie wbudowane jako kontroler sieci)



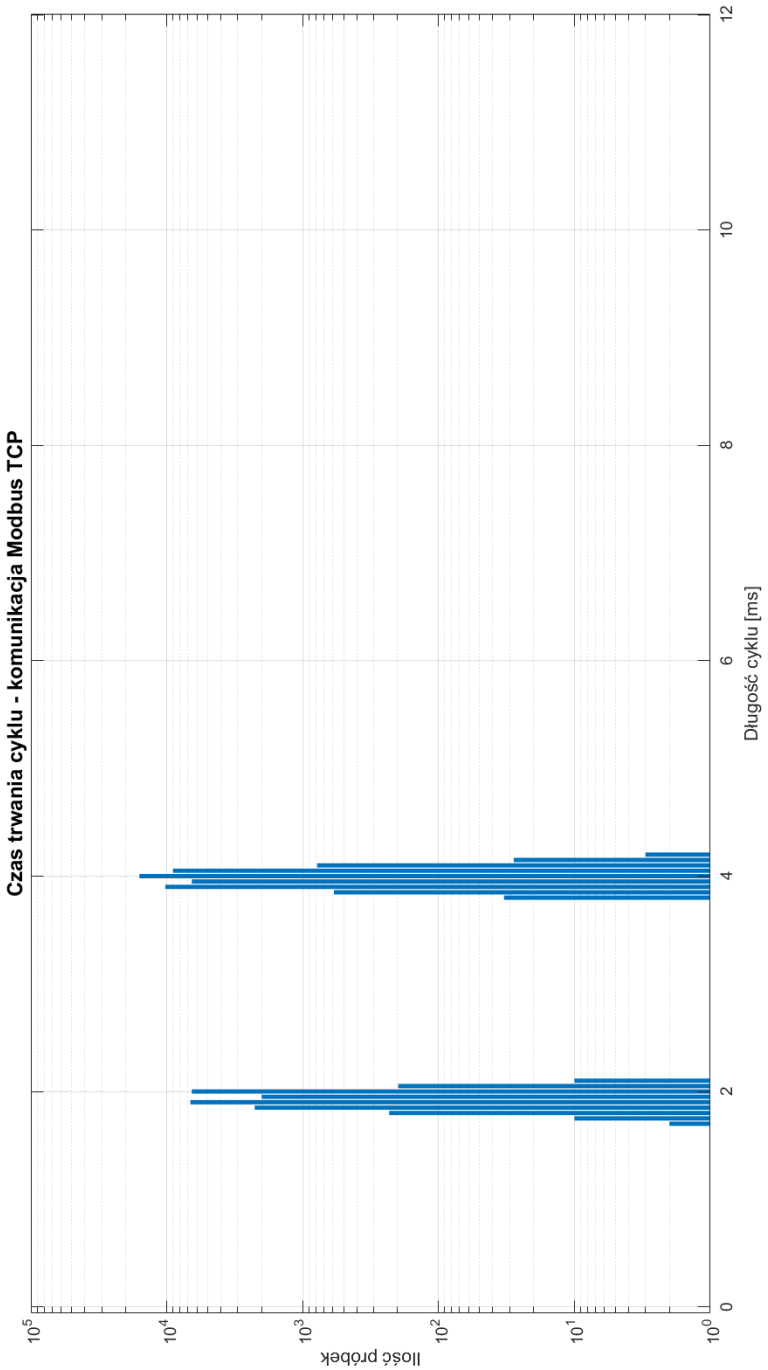
Rysunek 9.1 Wykres prezentujący pracę Raspberry Pi – bez obciążenia cyclic 2ms

Kolejnym etapem prac badawczych było wprowadzenie obciążenia, czyli kolejnych iteracji wykonywanego wcześniej programu sterującego. Początkowe pomiary pozwoliły na wyznaczenie minimum czasu potrzebnego do zrealizowania całości programu. Niestety również tutaj możliwe jest zaobserwowanie braku odpowiedniej wydajności przez co realizacja kolejnego programu wydłużona jest do dwukrotności podstawowego (wyliczonego przez narzędzie) cyklu. W tym przypadku jest to wartość 6ms. *(Rysunek 10.1)*

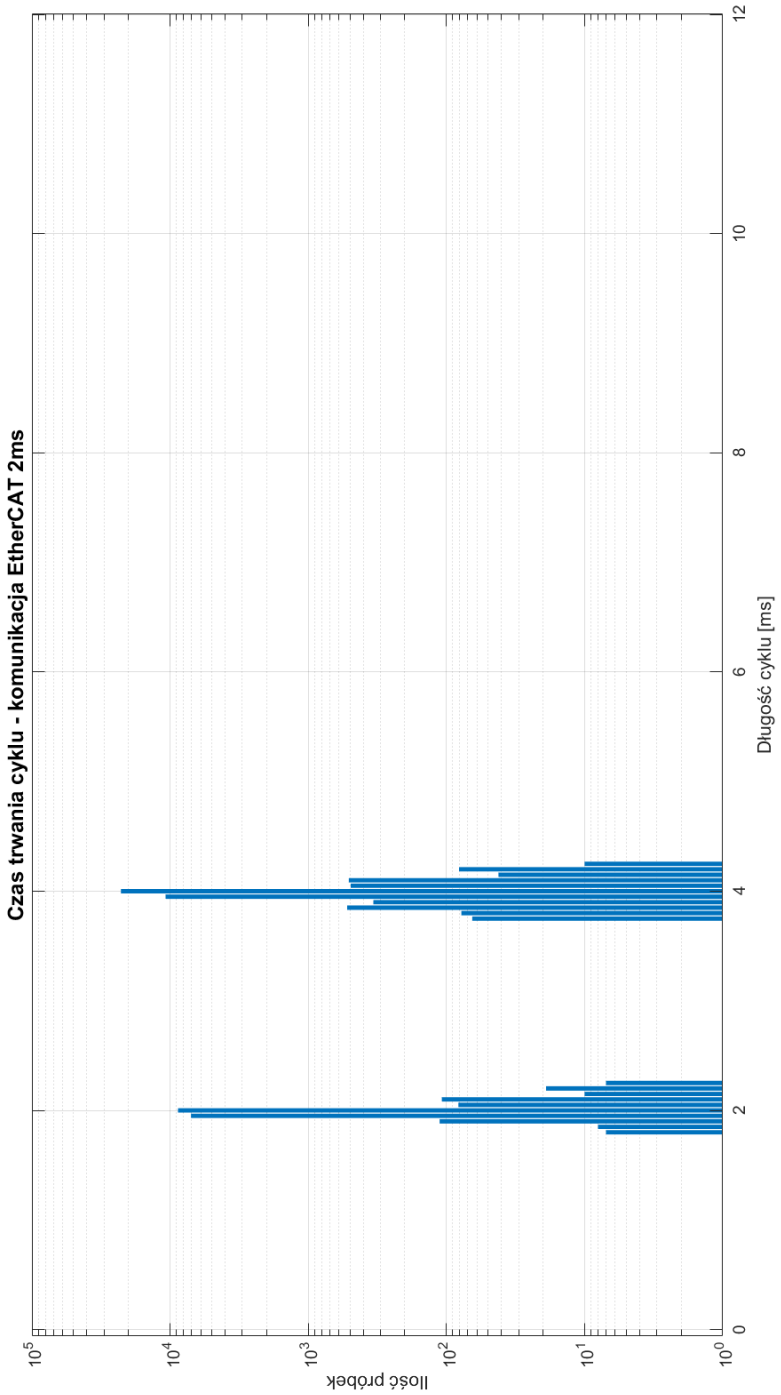


Rysunek 9.2 Wykres prezentujący pracę Raspberry Pi z dodatkowym obciążeniem freewhelling

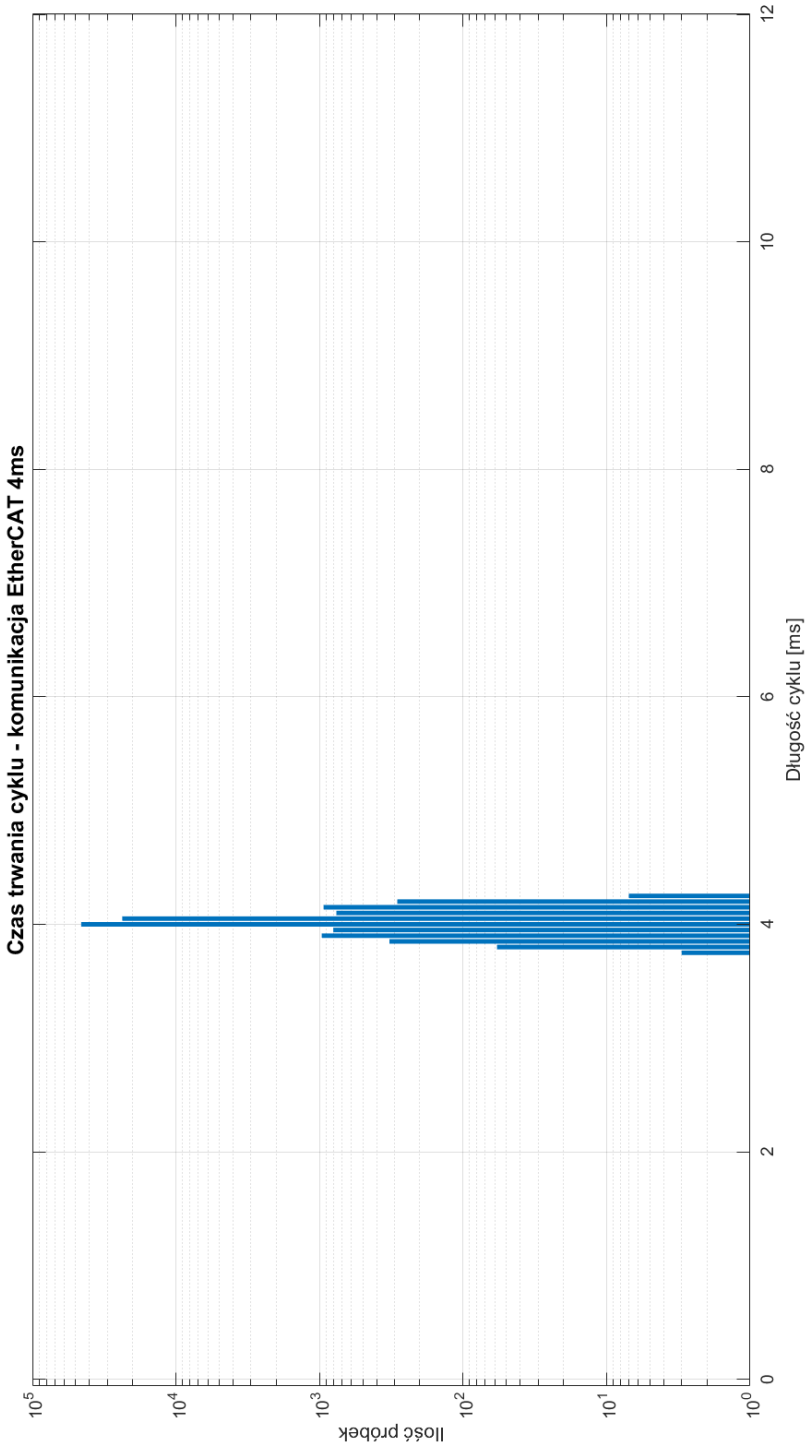
Wprowadzenie do konfiguracji komunikacji sieciowej Modbus TCP spowodowało pojawienie się cykli o długości 2ms względem tych samych eksperymentów przeprowadzonych dla pracy bez komunikacji sieciowej. Podobne wyniki zostały uzyskane w momencie wprowadzenia komunikacji EtherCAT.



Rysunek 9.3 Wykres prezentujący pracę Raspberry Pi Modbus TCP – bez obciążenia cyclic 2ms



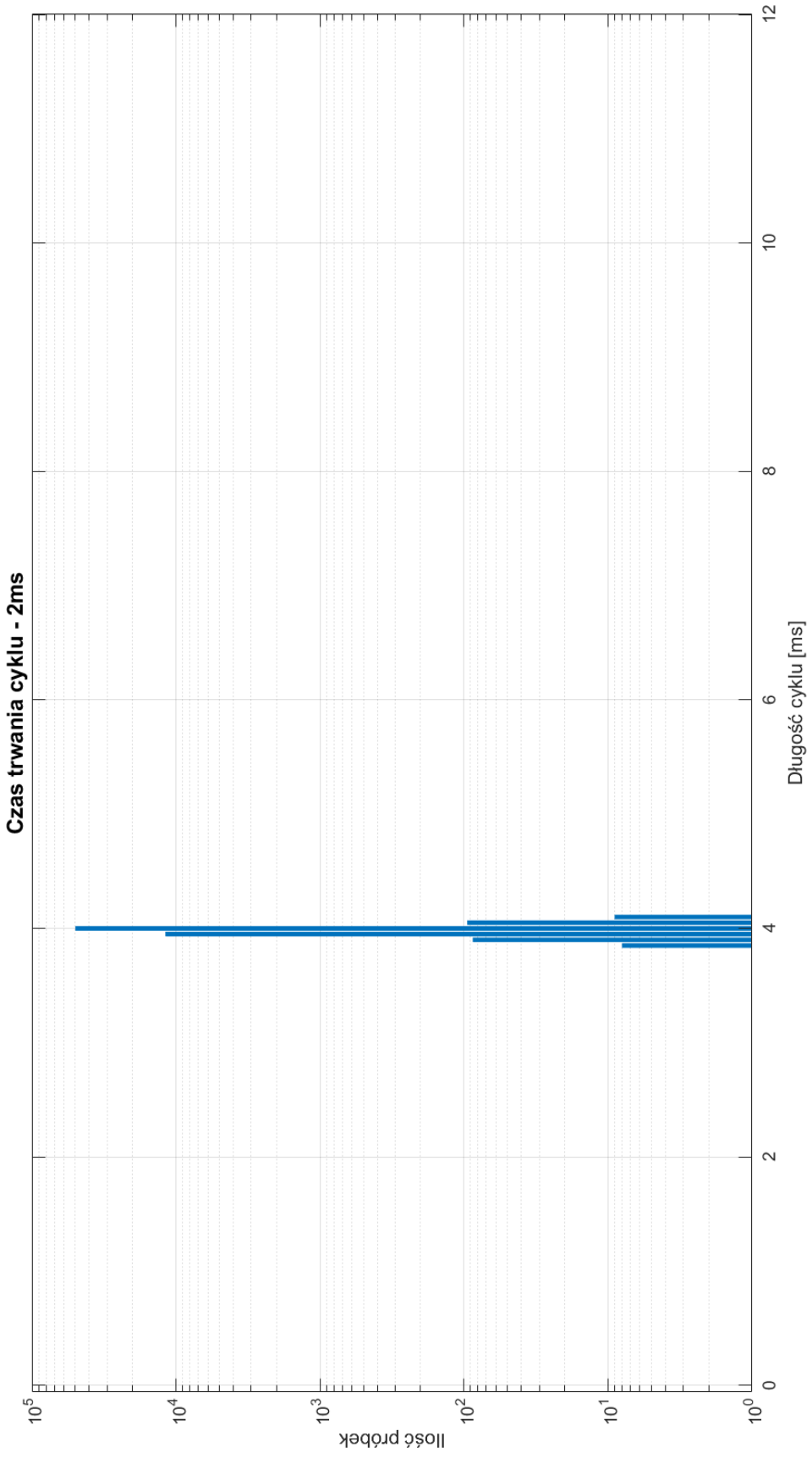
Rysunek 9.4 Wykres prezentujący pracę Raspberry Pi EtherCAT – bez obciążenia cyclic 2ms



Rysunek 9.5 Wykres prezentujący pracę Raspberry Pi EtherCAT – bez obciążenia cyclic 4ms

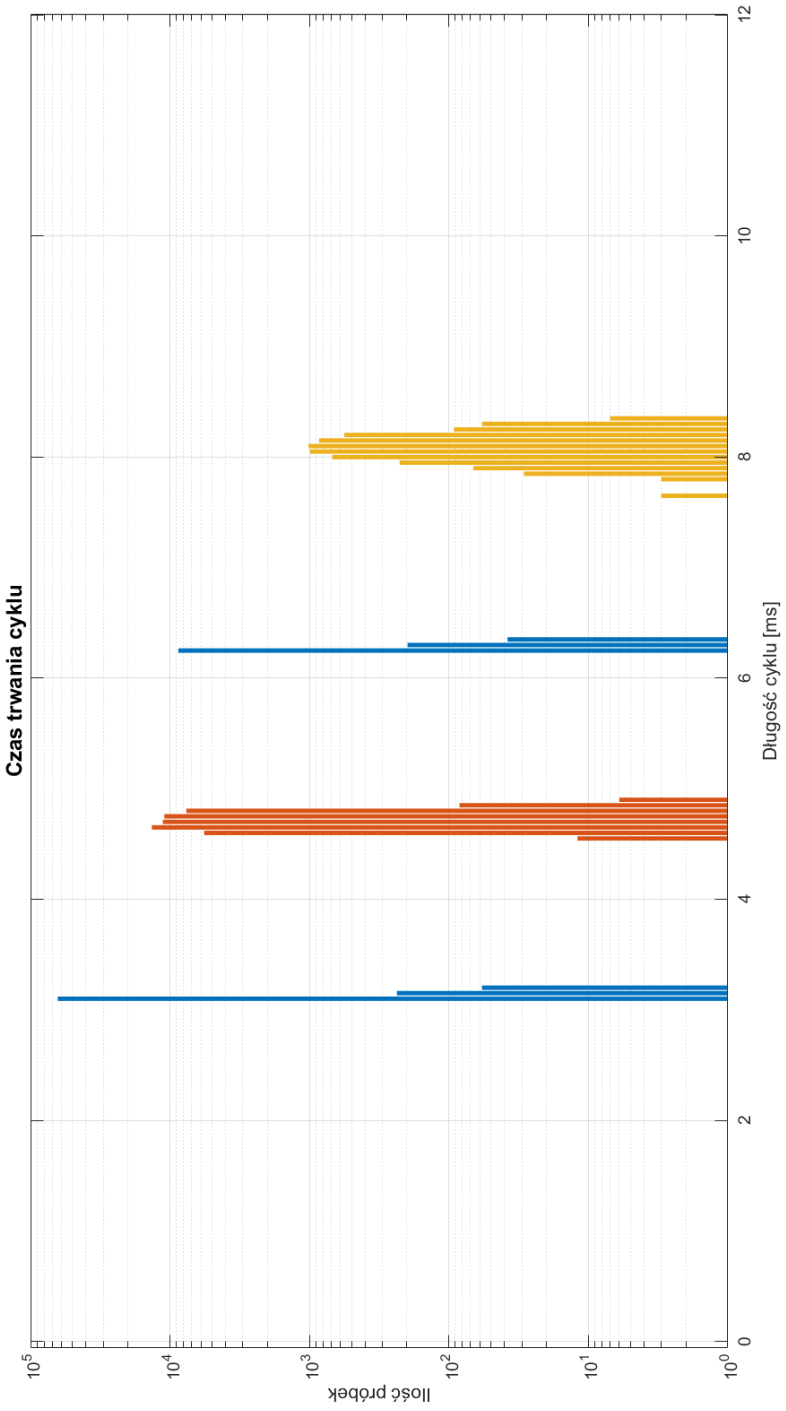
9.1.2 Urządzenie wbudowane Beagle Bone Black

Podobnie jak w poprzedniej części skonfigurowany 2ms cykl pracy sterownika nie został osiągnięty. Osiągnięte wartości znajdowały się w obrębię wartości 4ms. Także tutaj pomiary te zostały zarejestrowane dla konfiguracji pracy bez obciążenia, czyli bez realizacji dodatkowych pętli programu. Zarówno w poprzedniej serii dla urządzenia Raspberry Pi, także dla Beagle Bone Black przyjęto podejście polegające na ustawieniu krótkiego cyklu i obserwacji ewentualnego przekroczenia czasu wraz z wartością przekroczenia. Przekroczenia czasu realizacji cyklu sterownika zdarzają się również w sterownikach PLC dostępnych na rynku. Ta część badań pozwalała jednak określić w jaki sposób zadziałają urządzenia wbudowane w momencie realizacji programu wymagającego determinizmu czasowego.



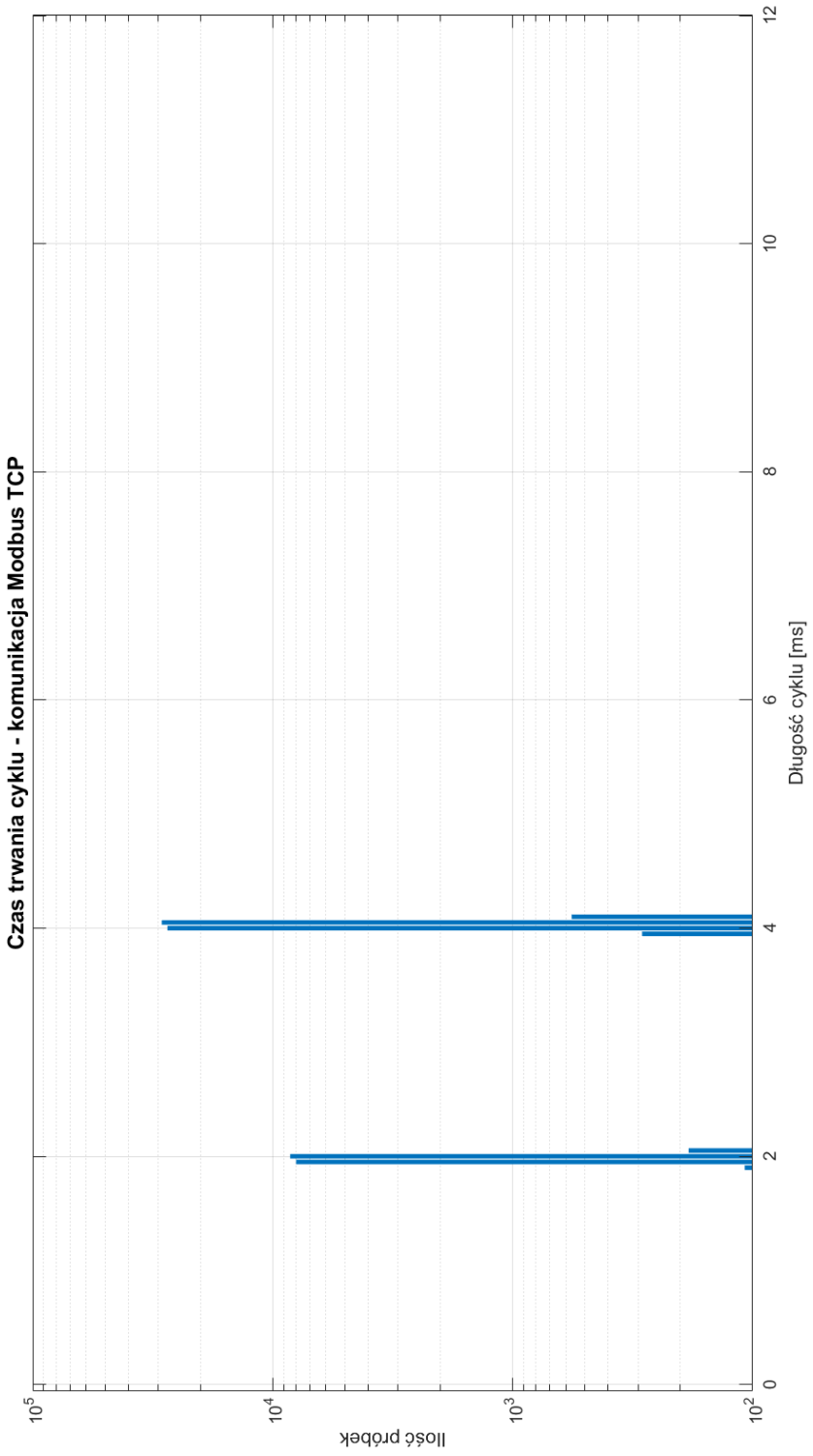
Rysunek 9.6 Wykres prezentujący pracę BeagleBone Black – bez obciążenia cyclic 2ms

Kolejnym etapem prac badawczych było wprowadzenie obciążenia, czyli kolejnych iteracji wykonywanego wcześniej programu sterującego. Początkowe pomiary pozwoliły na wyznaczenie minimum czasu potrzebnego do zrealizowania całości programu. Niestety również tutaj możliwe jest zaobserwowanie braku odpowiedniej wydajności przez co realizacja kolejnego programu wydłużona jest do dwukrotności podstawowego cyklu. W tym przypadku jest to wartość 6ms.

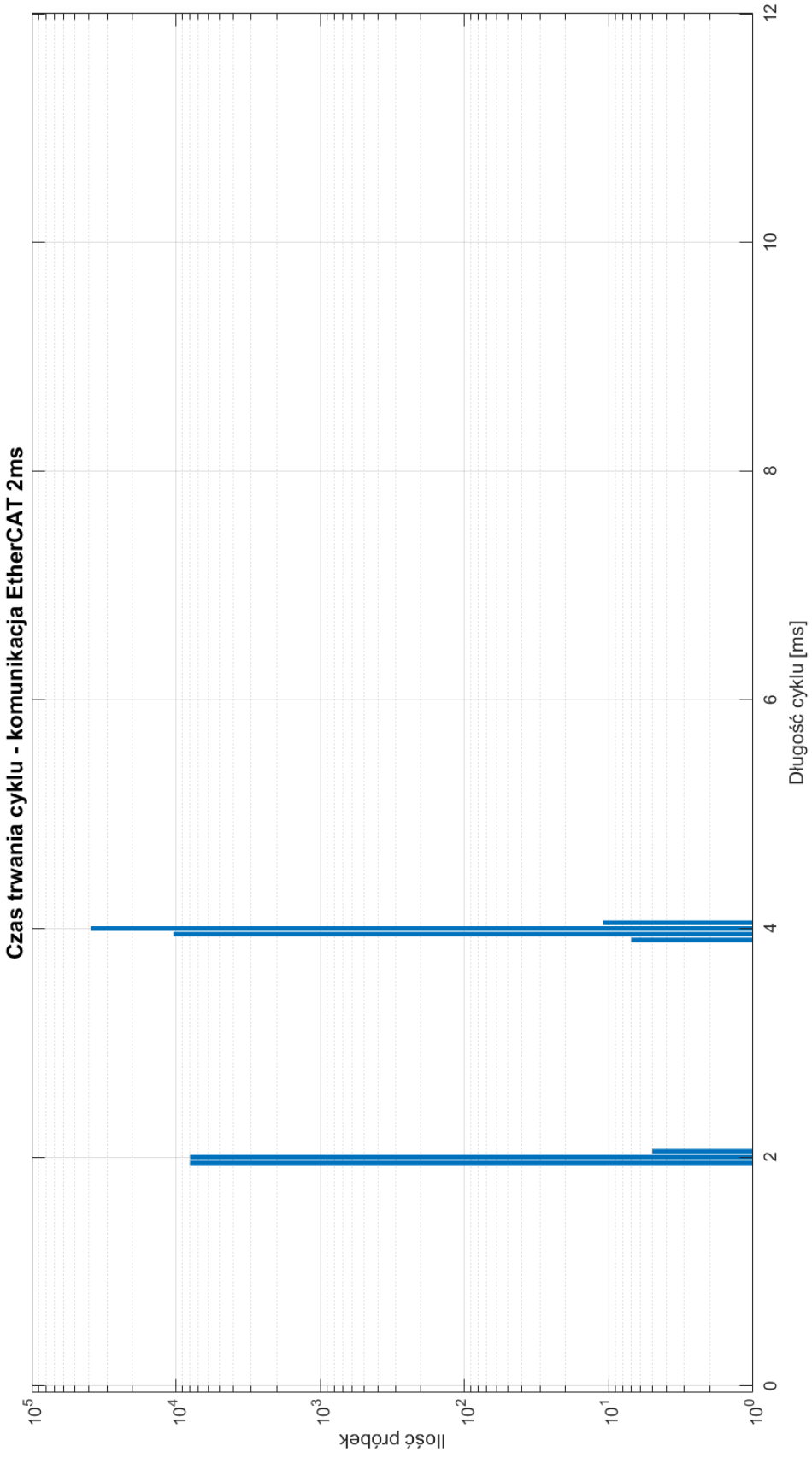


Rysunek 9.7 Wykres prezentujący pracę BeagleBone Black – z obciążeniem

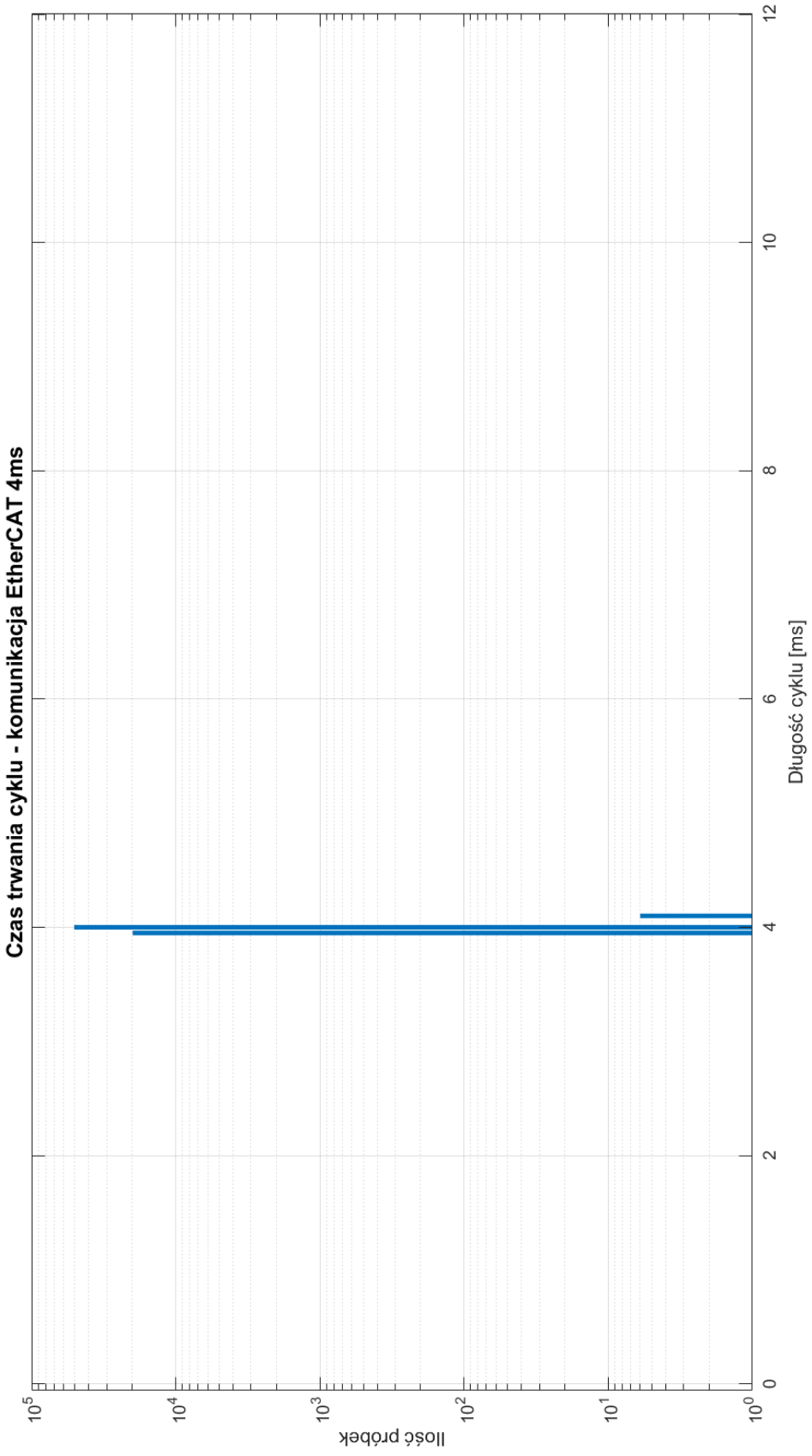
Wprowadzenie do konfiguracji komunikacji sieciowej Modbus TCP spowodowało pojawienie się cykli o długości 2ms względem tych samych eksperymentów przeprowadzonych dla pracy bez komunikacji sieciowej. Podobne wyniki zostały uzyskane w momencie wprowadzenia komunikacji EtherCAT.



Rysunek 9.8 Wykres prezentujący pracę BeagleBone Black – Modbus TCP



Rysunek 9.9 Wykres prezentujący pracę BBB – EtherCAT 2ms



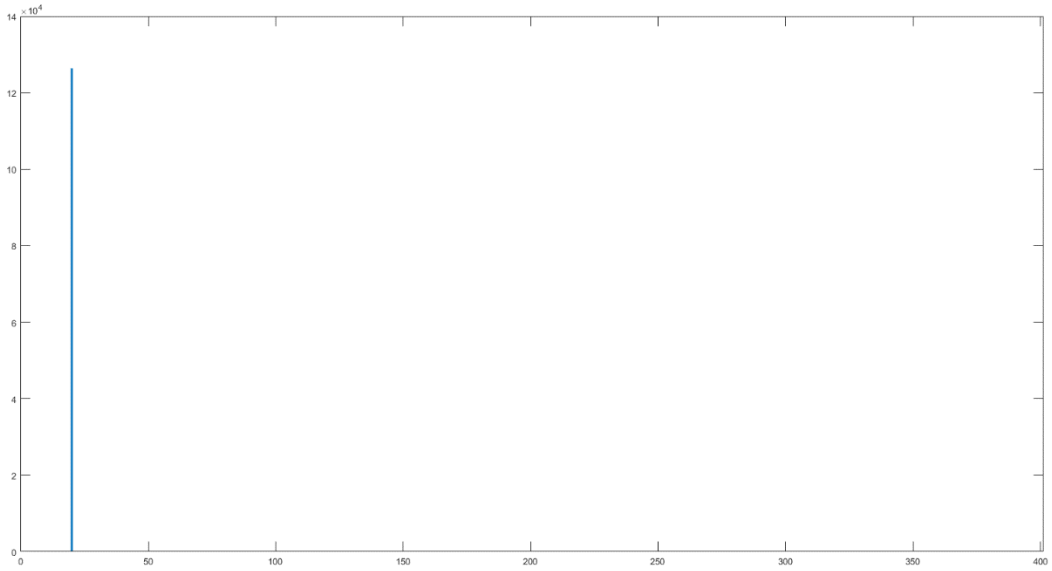
Rysunek 9.10 Wykres prezentujący pracę BBB – EtherCAT 4ms

9.2 Koncepcja SDN

Ta część badań pozwoliła na określenie działania sieci definiowanej programowo, z wykorzystaniem urządzenia wbudowanego w infrastrukturze. W badaniach użyto sieci RealTime Ethernet z cyklem na poziomie 20ms. Ze względu na niski poziom zaawansowania części badawczej cykl został skonfigurowany na relatywnie niskim poziomie. Jak zostało wspomniane we wcześniejszych rozdziałach do realizacji eksperymentów wykorzystano kartę NetFPGA[62]–[65] i przełącznik oparty o Raspberry Pi, a także kontroler sieci SDN uruchomiony na urządzeniu wbudowanym. Część badawcza uwzględniała sprawdzenie działania kontrolera sieci SDN uruchomionego na urządzeniu wbudowanym oraz na maszynie wirtualnej jako niezależny proces.

9.2.1 Analiza ruchu sieciowego przez urządzenie NetFPGA

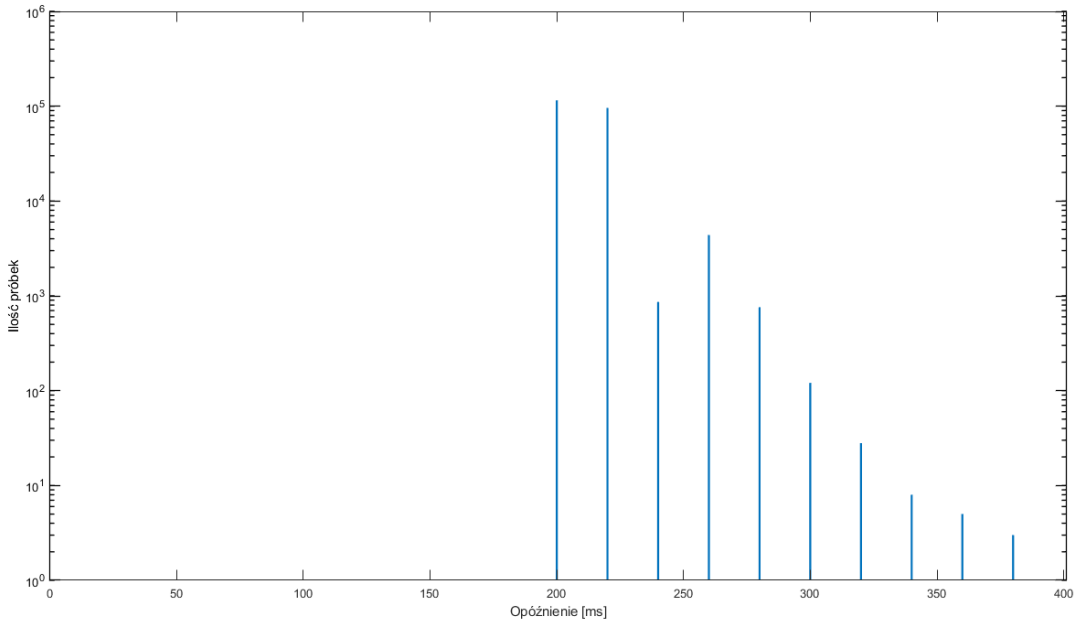
Otrzymane wyniki na tym etapie badań potwierdziły utrzymywanie długości trwania cyklu przez urządzenia wybrane do analizy. Do przełącznika podłączone zostały wbudowane urządzenie sterujące (Urządzenie wbudowane realizujące funkcje sterownika PLC. Klasę urządzeń można porównać do jakości oferowanej przez klasyczne sterowniki PLC. Wykorzystane urządzenia pracowały pod kontrolą systemu operacyjnego Windows 7 Embedded.). Skonfigurowany cykl na poziomie 20ms jest utrzymywany przy każdej wymianie. Otrzymane wyniki są podstawą do dalszej analizy i punktem odniesienia do porównania wyników po podłączeniu urządzeń przez przełącznik sieci definiowanej programowo SDN. Na *Rysunku 10.17* przedstawiono otrzymane w tej części wyniki, gdzie wszystkie pomiary zostały zarejestrowane na wartości 20ms.



Rysunek 9.11 Wyniki prezentujące pracę sieci RealTime Ethernet bez przełącznika SDN

9.2.2 Analiza ruchu sieciowego SDN przez urządzenie NetFPGA

W porównaniu do pomiarów zrealizowanych bez dodatkowej infrastruktury sieciowej minimalny czas obsługi ruchu sieciowego wynosi 200ms. Wartość ta jest dziesięciokrotnie większa do zadanej wartości cyklu sieci RealTime Ethernet. Na taki rezultat wpływ mają przede wszystkim budowa samego przełącznika sieciowego zrealizowana na bazie kart USB zainstalowanych w urządzeniu wbudowanym Raspberry Pi. Taka architektura wprowadza dodatkowy czas wymagany na obsługę połączenia USB i przetworzenie danych przez procesor i sterownik odpowiedzialny za obsługę karty sieciowej.



Rysunek 9.12 Wyniki prezentujące pracę sieci z przełącznikiem sieciowym SDN zbudowanym na bazie urządzenia wbudowanego Raspberry Pi.

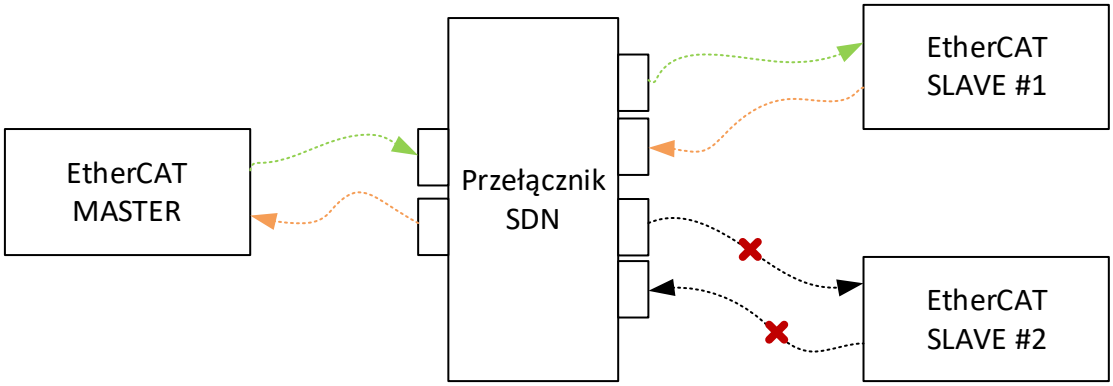
Próbując odnieść otrzymane wyniki do rzeczywistego systemu przemysłowego, można stwierdzić, że dziesięciokrotne przekroczenie długości trwania cyklu sieci, będzie skutkowało zatrzymaniem procesu sterowania ze względu na brakujące dane pochodzące z urządzeń sieciowych tj. czujniki sieciowe lub dane z innych sterowników pracujących w systemie. Nie oznacza to jednak, że wykorzystanie przełącznika sieciowego i kontrolera SDN opartego o urządzenie wbudowane nie jest możliwe. Jeśli tylko parametry czasowe są możliwe do zaakceptowania, możliwe jest zastosowanie takiej konfiguracji. Przykładem takich wymian może być pomiar wielkości fizycznych, których zmiana w czasie jest wolna np. zmiana temperatury lub pomiar zużycia elementów tocznych (np. łożyska).

Wykorzystanie karty sieciowej NetFPGA mimo, że pozwalało na analizę ruchu sieciowego, generowało problemy w postaci zatrzymywania analizy w losowym momencie. Takie zachowanie było obserwowane w czasie prób uruchomienia komunikacji z interfejsami o szybkości działania większej niż 100Mb/s. Statyczne ustawienie pracy karty sieciowej na tym poziomie pozwoliło zdiagnozowanie problemów w czasie realizacji pomiarów i przeprowadzenie odpowiednio długich serii pomiarowych. Wyeliminowanie takiego zachowanie

karty sieciowej NetFPGA wymaga przeprojektowania oprogramowania układowego. Z tego powodu do dalszej analizy zdecydowano się wykorzystać urządzenie dedykowane do analizy ruchu sieciowego na poziomie 1Gb/s.

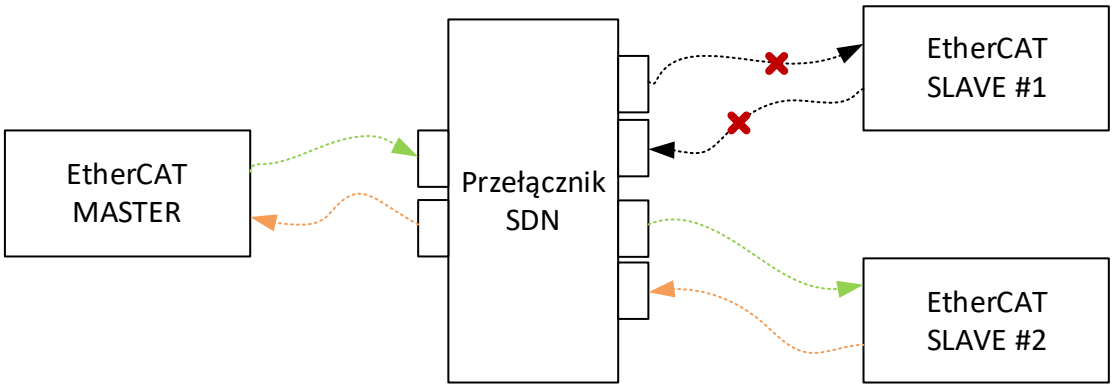
9.3 SDN topologia logiczna

Jak zostało wspomniane wcześniej kontroler umieszczony w infrastrukturze sieci SDN umożliwia sterowanie wybieranie odpowiednich urządzeń biorących udział w wymianie danych. Modyfikacja adresów MAC pozwala na przesyłanie danych do określonych urządzeń. Wykorzystane w badaniach urządzenia EtherCAT wymagały dopasowania adresów MAC do konfiguracji zapisanej w sterowniku. Przygotowanie w konfiguracji kontrolera SDN odpowiednich adresów wpisów dotyczących zmiany adresów MAC. Uruchomienie odpowiedniej reguły aktywowało transmisję pomiędzy odpowiednimi interfejsami w przełączniku sieciowym SDN.



Rysunek 9.13 Aktywna reguła przesyłająca dane do urządzenia "Slave" numer jeden

Działanie ręcznej zmiany konfiguracji potwierdziła możliwość zarządzania aktywnymi urządzeniami. Jeśli taka możliwość istnieje w ręcznym zarządzaniu topologią dokładnie ten sam mechanizm można zastosować w dynamicznym zarządzaniu.

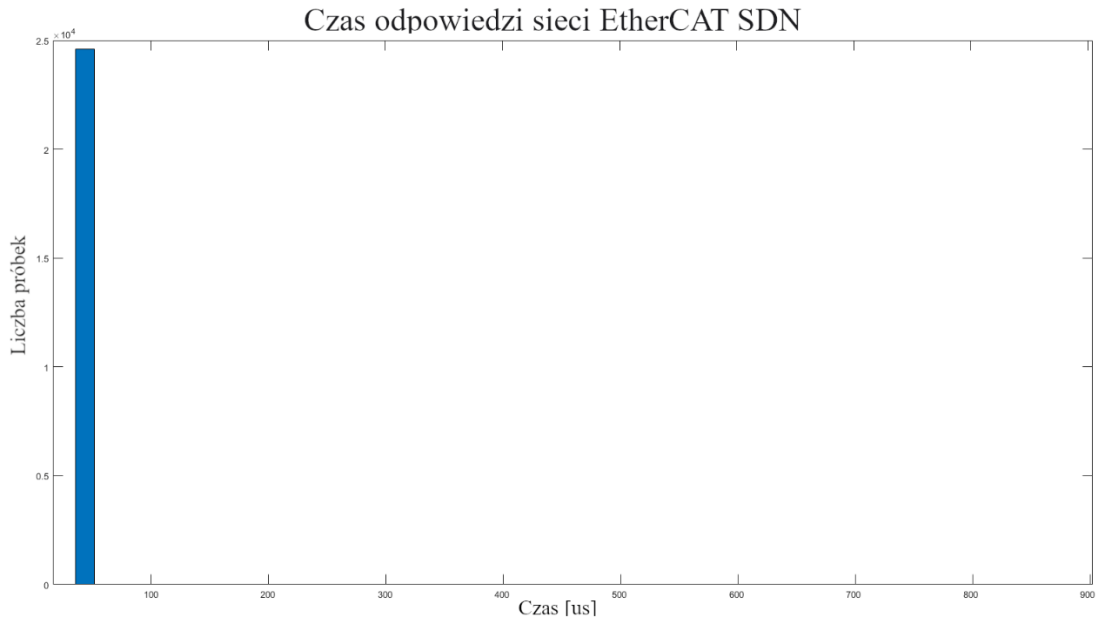


Rysunek 9.14 Aktywna reguła przesyłająca dane do urządzenia "Slave" numer dwa

W kolejnym etapie sprawdzono parametry czasowe początkowo bez wykorzystania przełączników SDN, a w kolejnym kroku z zainstalowanym przełącznikiem sieciowym SDN.

9.3.1 Topologia sieciowa bez używania przełączników sieciowych SDN

W analizowanym przypadku potwierdzono działanie zaproponowanej sieci EtherCAT. Do komunikacji wykorzystano jedynie urządzenia dedykowane komunikacji EtherCAT. Czas oczekiwania na informację zwrotną pokrywa się z dodatkowym czasem wprowadzanym do sieci przez urządzenia i moduły biorące udział w komunikacji. Można przyjąć, że czas oczekiwania kształtuje się na poziomie ok. 450us. Wartość ta jest akceptowalna i sieć EtherCAT działa poprawnie. Dodatkowo wartość ta będzie punktem odniesienia do porównania wyników pomiarów po uwzględnieniu przełączników sieciowych SDN. Na wykresie zaprezentowanym na *Rysunku 8.13* przedstawiono, jak kształtuje się czas przejścia ramki przez sieć typu EtherCAT bez rozróżniania kolejnych elementów. Podział na poszczególne elementy został zaprezentowany w dalszej części i zostało porównane z czasem wprowadzanym przez przełączniki sieci SDN.

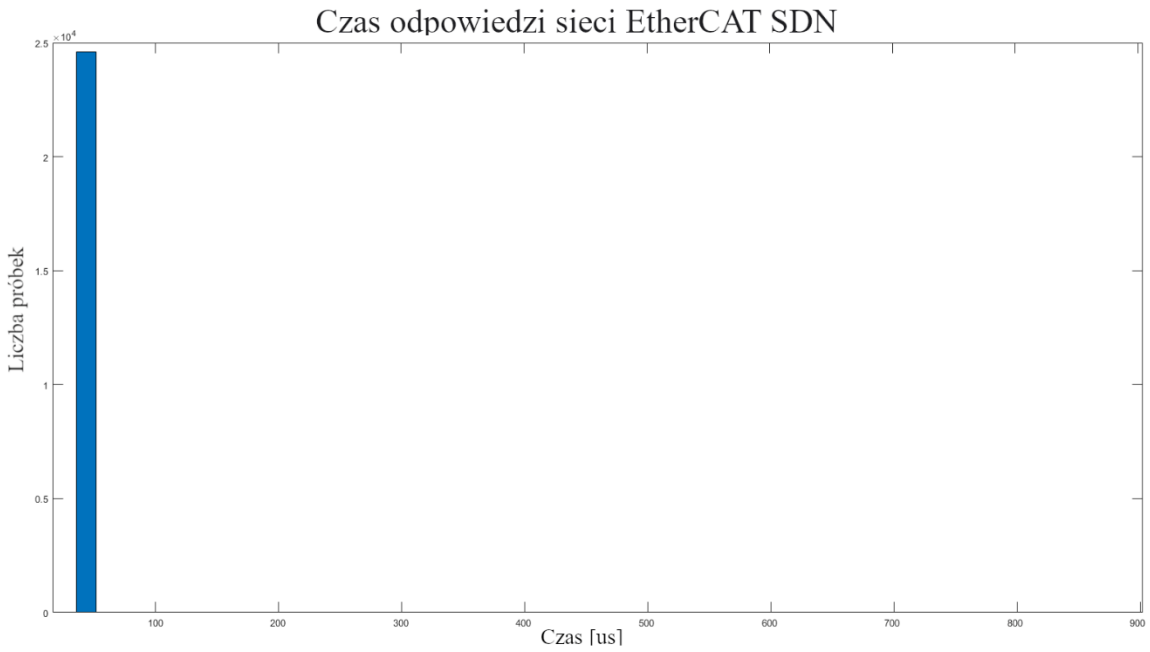


Rysunek 9.15 Rozkład czasu odpowiedzi w sieci EtherCAT bez przełączników SDN

9.3.2 Topologia sieciowa z użyciem przełączników sieciowych SDN

Otrzymane wyniki prezentują przejście przez poszczególne składowe systemu przemysłowego. W tej części eksperymentów potwierdzono opóźnienie wprowadzane przez przełącznik SDN. Do analizy użyto urządzenia analizującego ruch sieciowy Beckhoff ET2000.[66] Przy jednoczesnym monitorowaniu 8 kanałów przeanalizowano transmisję nie tylko na z jednym urządzeniem Master i jednym urządzeniem Slave, ale w rzeczywistej sieci z kilkoma urządzeniami. W analizowanym przypadku była to jedna wyspa zdalnych wejść/wyjść oraz dwóch kontrolerów napędów.

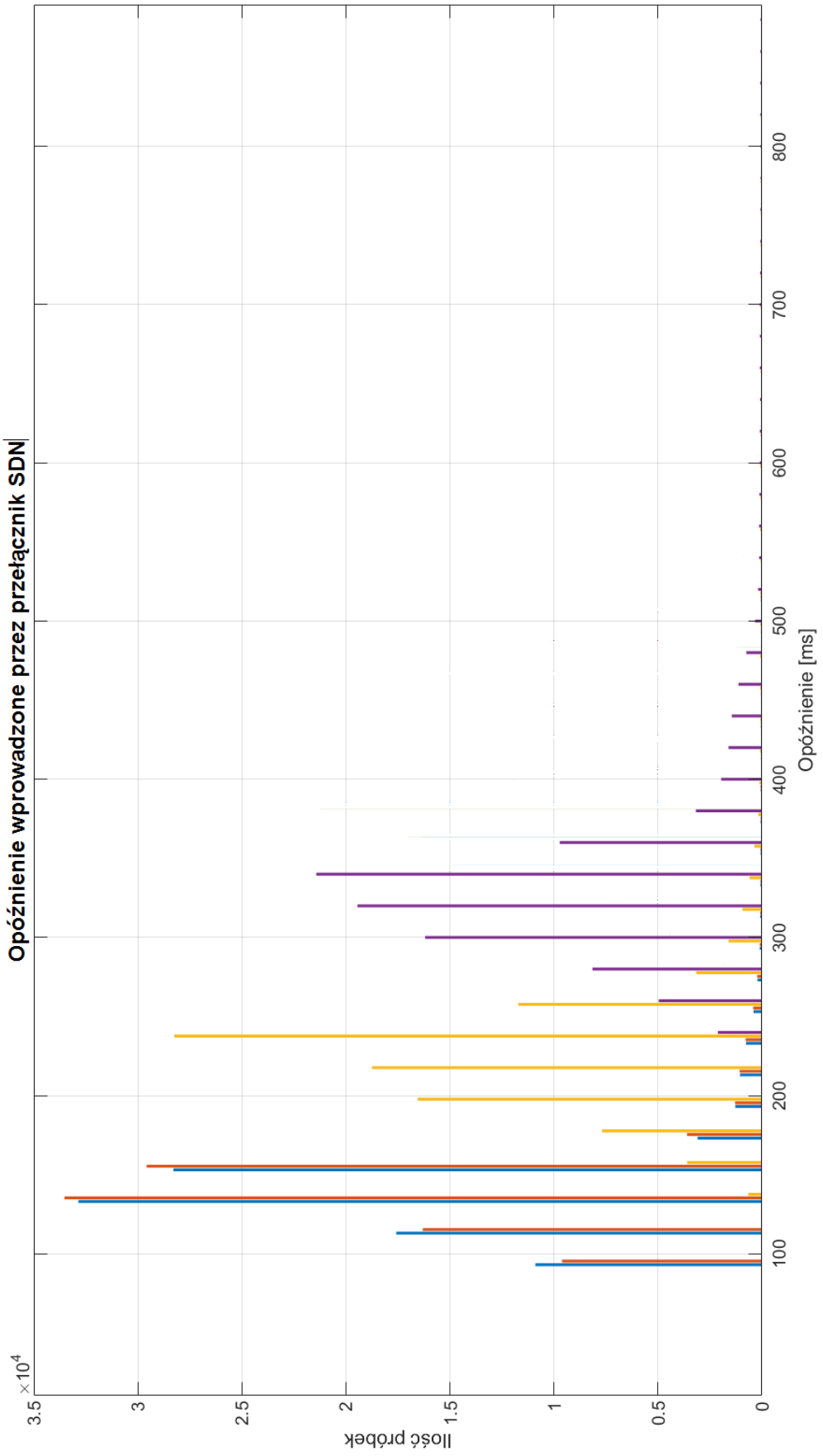
Rysunek 10.14 przedstawia czas odpowiedzi uwzględniając przejście ramki transmisyjnej przez wszystkie urządzenia. Pomiar ten jest realizowany w sposób identyczny jak w sieci bez przełączników SDN. W stosunku do poprzedniej serii pomiarowej czas odpowiedzi wydłużył się o dodatkowe 200us.



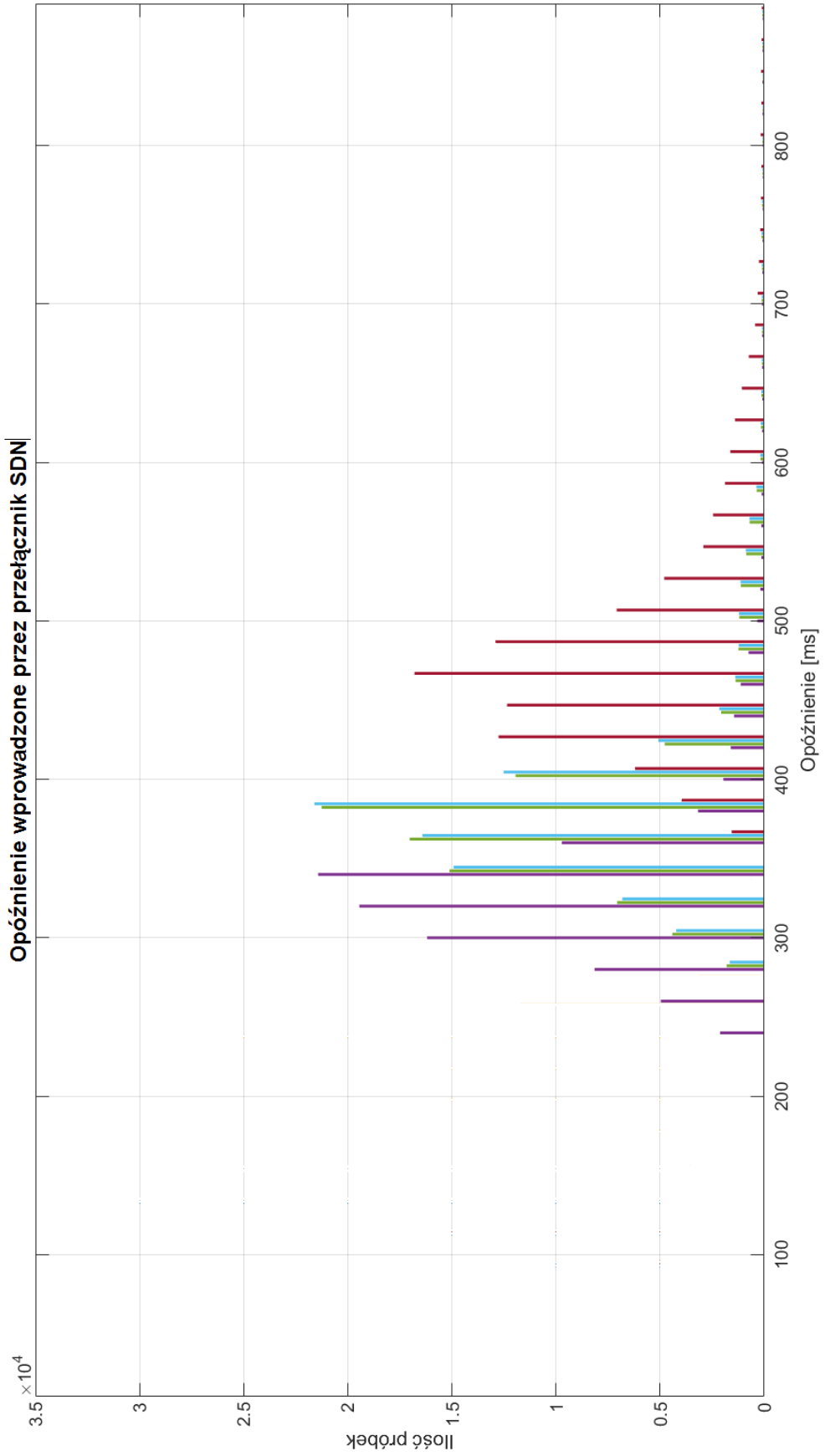
Rysunek 9.16 Rozkład czasu odpowiedzi w sieci EtherCAT z urządzeniami SDN

Na *Rysunkach 10.17* oraz *10.18* przedstawiono opóźnienie wprowadzono przez kolejne urządzenie biorące udział w komunikacji. W tej części na dodatkowy czas w głównej mierze wpływ mają czasy przejścia przez przełączniki SDN, a dodatkowe składowe wprowadzane jako urządzenia możemy traktować jako stałą wartość, taką samą dla wszystkich pomiarów.

Rysunki 10.17 oraz *10.18* prezentują *rozkład* wprowadzanych do systemu poszczególnych opóźnień. Kolejne kolory pokazują przejście przez kolejny moduł. Największy wpływ na działanie sieci mają przełączniki SDN, najmniejszy elementy infrastruktury sieci EtherCAT. Wraz ze wzrostem czasu oznaczonego na osi „X", ilość próbek z poprzednich części maleje, a wzrasta część pomiarów z dłuższym czasem obsługi. Całość wykresu prezentuje przejście ramki przez całą sieć przemysłową wraz z przełącznikami SDN. Podsumowując tę część badań można stwierdzić, że używanie przełączników SDN nie będzie miało wpływu na działanie sieci pod pewnymi warunkami. Dopasowanie długości cyklu sieci do wymagań systemu przemysłowego.



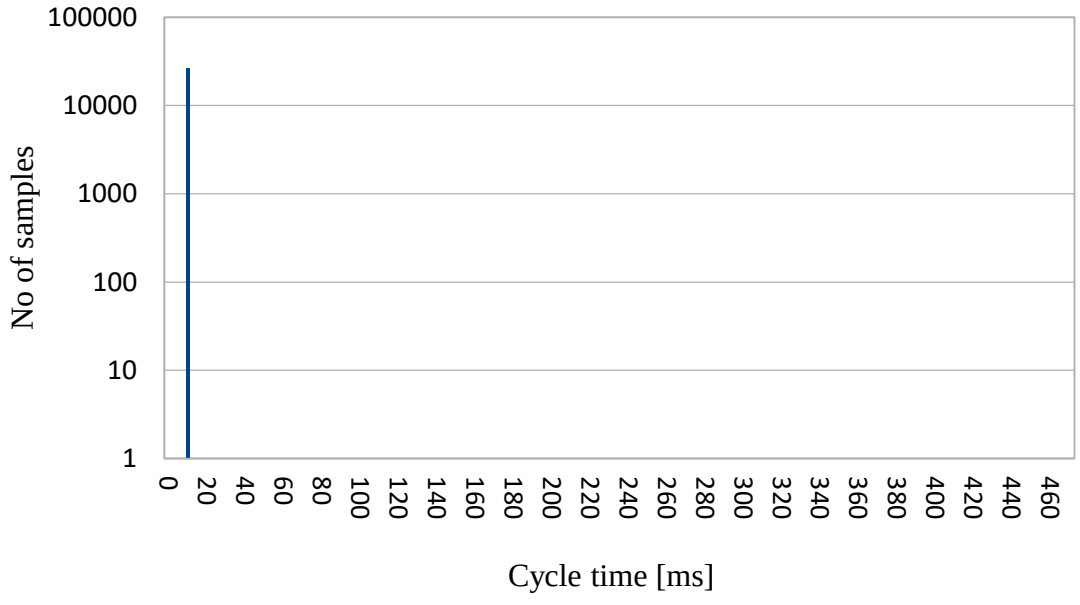
Rysunek 9.17 Opóźnienia wprowadzane przez kolejne urządzenia w sieci EtherCAT



Rysunek 9.18 Opóźnienia wprowadzane przez kolejne urządzenia w sieci EtherCAT

9.4 Komunikacja bezprzewodowa w sieciach SDN

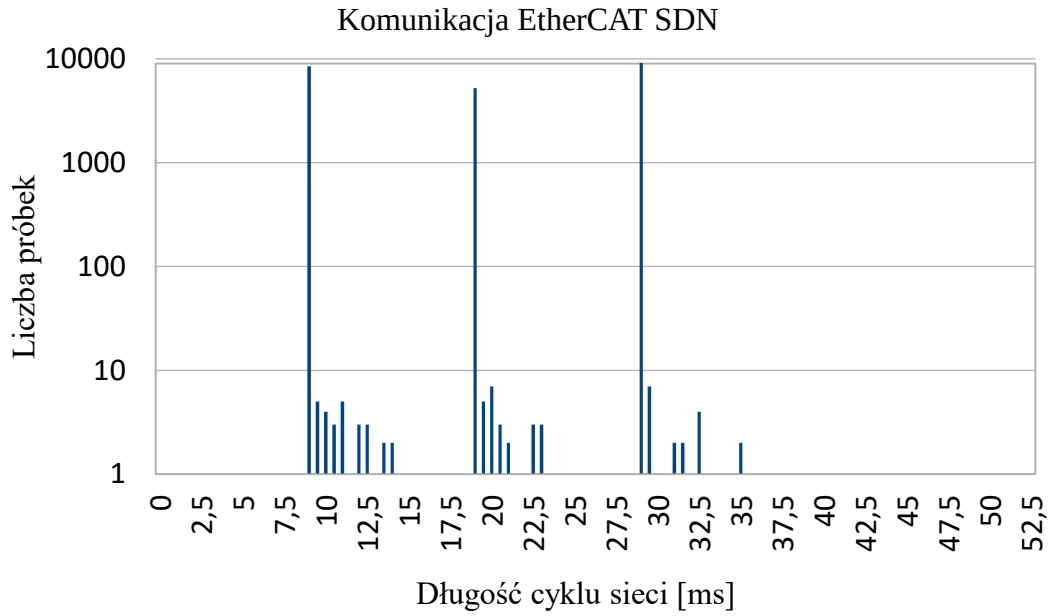
Wprowadzając termin „komunikacji bezprzewodowej” do systemów przemysłowych należy liczyć się z problemami z osiąganymi parametrami takimi jak opóźnienie lub „jitter”. Wyniki po przeprowadzonych eksperymentach obrazują, że jest to znaczący problem, co wydaje się udowodniono zarówno w części praktycznej jak i dalszej analizie otrzymanych wyników. Na wykresach zaprezentowano wyniki za równo dla pracy w domyślnej konfiguracji jak i po wprowadzeniu dodatkowych elementów tj. przełączniki SDN czy mostu bezprzewodowego. Dla każdej z prób zebrano ponad 20 tysięcy próbek, co było wartością pozwalającą na porównanie działania sieci. W pierwszym teście zrealizowano proste połączenie z wykorzystaniem jednej stacji typu Master (sterownik PLC) oraz jednej stacji Slave (zdalna stacja wejść/wyjść). W tym przypadku skonfigurowanie wymian cyklicznych na poziomie 10ms skutkowało odzwierciedleniem tej wartości w pomiarach i obserwacją wymian co 10ms, co zostało przedstawione na *Rysunku 10.19.*



Rysunek 9.19 Komunikacja w sieci EtherCAT z cyklem 10ms

W kolejnym etapie wprowadzono do infrastruktury przełączniki sieciowe SDN pozwalające na przekierowanie ruchu pomiędzy urządzeniami wraz z odpowiednią modyfikacją adresów MAC kart sieciowych w taki sposób, aby

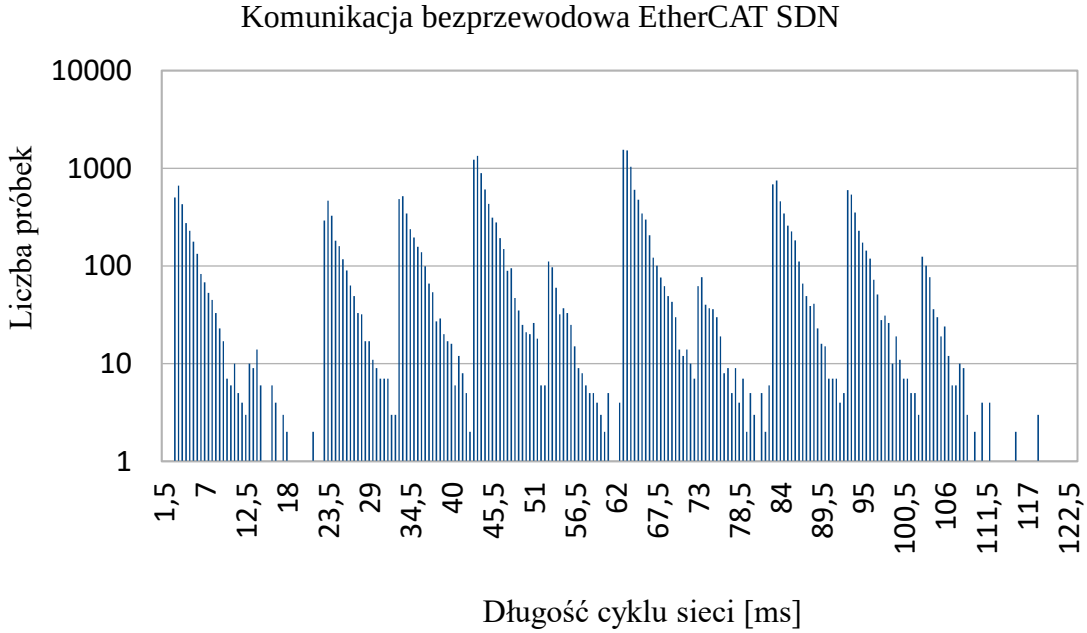
urządzenie EtherCAT nie były w stanie wykryć obecności dodatkowych przełączników. Uzyskane wyniki nie są już tak idealne jak w poprzedniej części, czego należało się spodziewać. Przełączniki sieciowe SDN pomimo na stałe wprowadzonej reguły potrzebują więcej czasu na realizację zadań co powoduje zmianę „jittera" z wartości w praktyce zbliżonej do 0ms, na wartości równe nawet połowie długości skonfigurowanej długości cyklu, co w tym przypadku jest równe 5ms. Część zadań nie jest także realizowalna w czasie jednego pełnego cyklu, co powoduje wydłużenie czasu realizacji wymiany do poziomu 2-krotności i 3-krotności trwania cyklu (odpowiednio 20ms i 30 ms). W tym przypadku „jitter" również jest na poziomie maksymalnie 5ms. Otrzymane wyniki w tej serii pomiarów pokazują, że przy odpowiednich założeniach czasu cyklu sieci, możliwa jest realizacja wymian z przełącznikami sieciowymi SDN. *(Rysunek 10.20)*



Rysunek 9.20 Komunikacja w sieci EtherCAT z cyklem 10ms oraz wykorzystaniem przełączników sieciowych SDN.

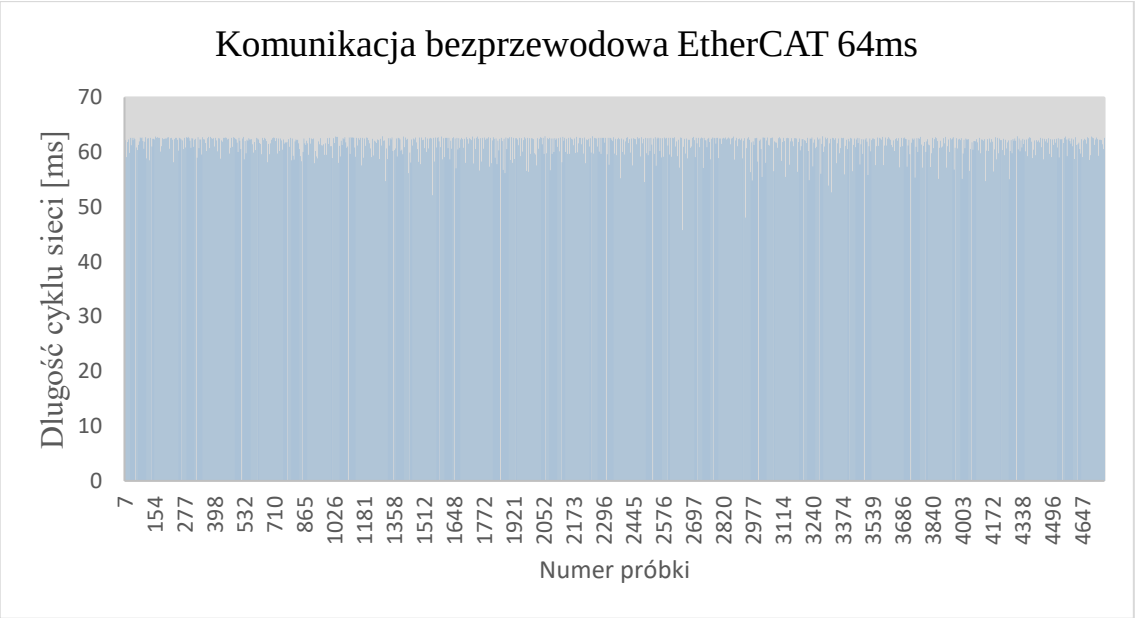
Wprowadzenie dodatkowych urządzeń bezprzewodowych spowodowało nie do końca prawidłową pracę sieci przemysłowej EtherCAT. O tyle, o ile dane były przesyłane i możliwa była ich obserwacja na zdalnych modułach wejść/wyjść, o tyle obserwując parametry czasowe zauważono pojawienie się sygnałów poniżej skonfigurowanej długości trwania cyklu (poniżej 10ms). Otrzymane wyniki pokazują, że sposób komunikacji jest realizowalny, jednak

uzyskiwane parametry czasowe są dalekie od wartości wymaganych do pracy np. z napędami. Nie wyklucza to jednak zastosowania komunikacji bezprzewodowej w systemach, gdzie dane zmieniają się wolno lub ich błędny odbiór nie będzie miał wpływu na działanie całego systemu. Przy skonfigurowaniu cyklu sieci na poziomie ok.100ms (jak wynika z pomiarów zaprezentowanych na *Rysunku 10.21*,) sieć będzie działała poprawnie. Takimi sygnałami może być np. informacja o zmianie temperatury lub o poziomie wody w zbiorniku.

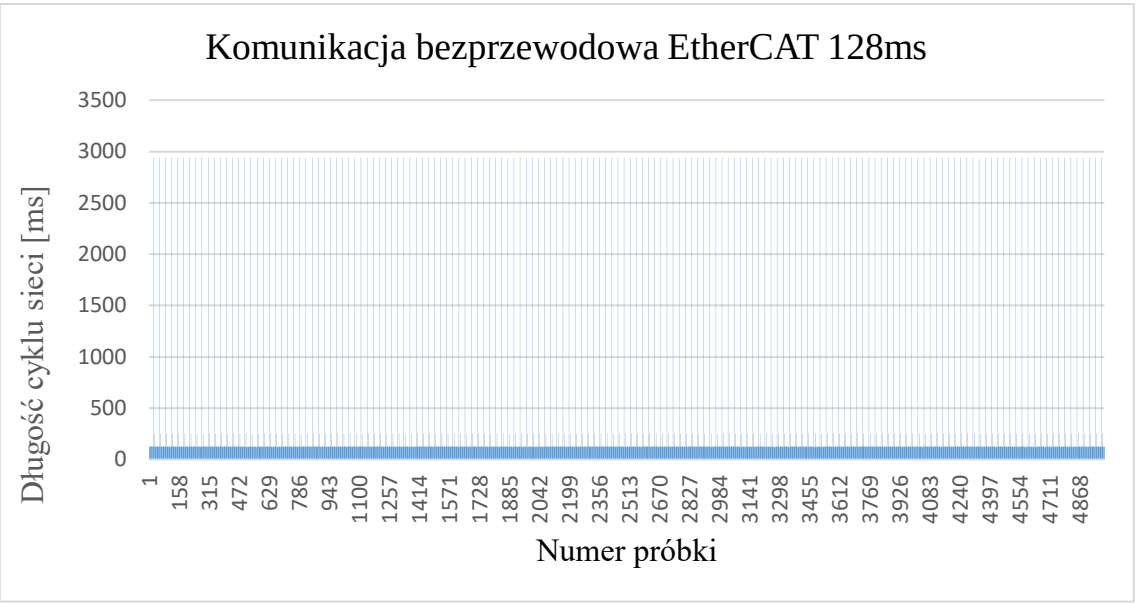


Rysunek 9.21 Komunikacja w sieci EtherCAT z cyklem 10ms, wykorzystaniem przełączników sieciowych SDN oraz komunikacji bezprzewodowej WiFi.

Aby potwierdzić wcześniejszą interpretację wyników, przeprowadzono dodatkowe pomiary, gdzie sprawdzono działanie sieci przy konfiguracji sieci z cyklem wynoszącym 64ms *(Rysunek 10.22)* oraz 128ms *(Rysunek 10.23)*. W cyklu sieci ustawionym na poziomie 64ms nie zaobserwowano znaczących przekroczeń zadeklarowanej wartości. Zadeklarowanie długości cyklu sieci na poziomie 128ms spowodowało występowanie próbek o długości większej niż kilkukrotność ustawionej wartości. Brak przełącznika SDN w testach z cyklem 64ms oraz 128ms tłumaczy brak charakterystycznego kształtu wykresu („piłokształtny”)



Rysunek 9.22 Komunikacja bezprzewodowa EtherCAT z cyklem sieci 64ms



Rysunek 9.23 Komunikacja bezprzewodowa EtherCAT z cyklem sieci 128ms

W wynikach otrzymanych w testach komunikacji bezprzewodowej, duży wpływ na wyniki pomiarów ma środowisko badawcze tzn. możliwe do zaobserwowania zakłócenia pochodzące od innych urządzeń bez przewodowych.

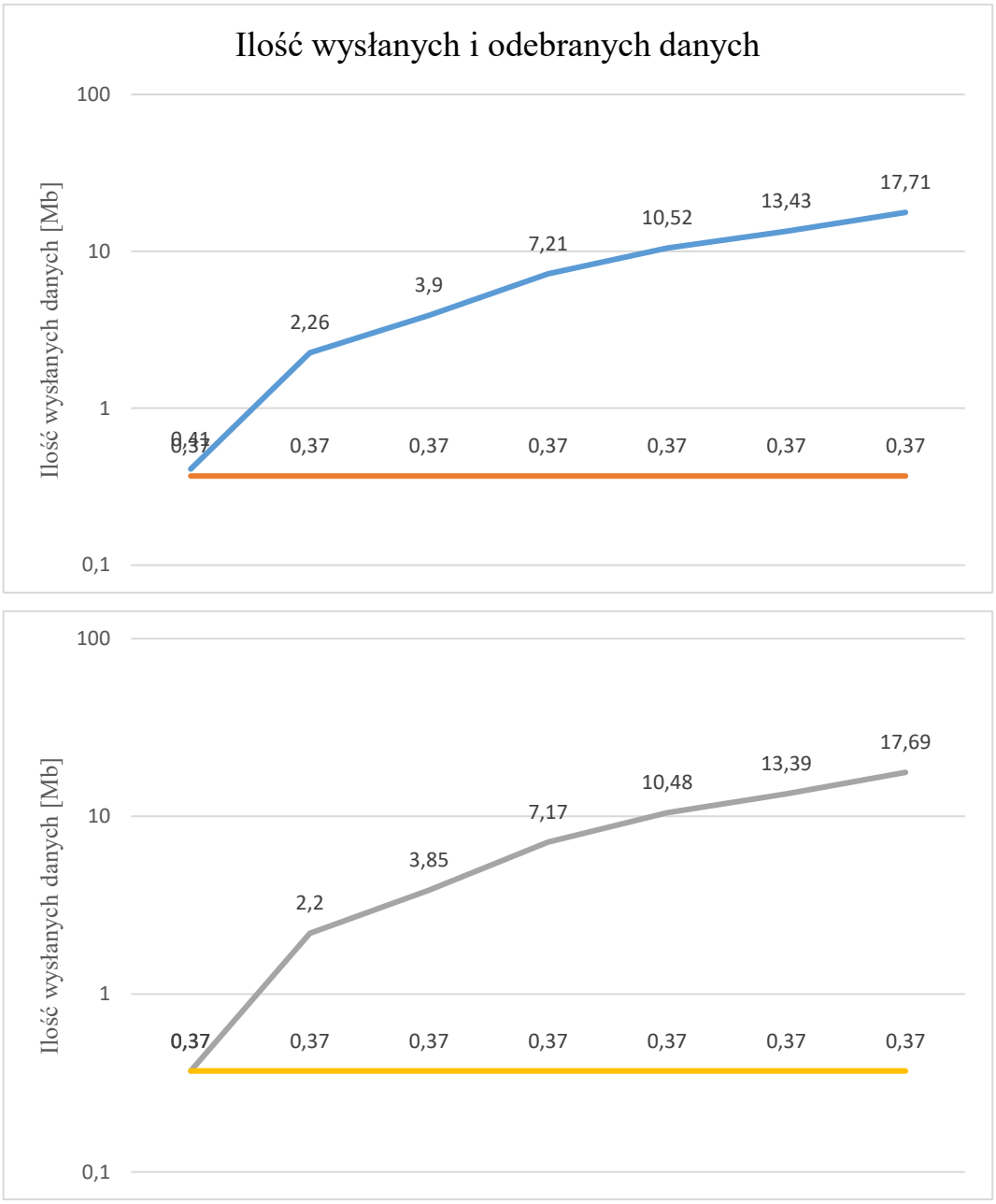
Zdecydowano się zaprezentować otrzymane wyniki, ze względu na wartość, którą wnoszą do całości pracy, a także do udowodnienia wcześniej postawionych tez i założeń.

9.5 Integracja sieci przemysłowej z OPC UA

Połączenie sieci przemysłowej wraz z systemem integrującym jest możliwe. W części badawczej zaproponowano sprawdzenie działania takiego połączenia. Wprowadzanie kolejnych klientów powodowało przesyłanie coraz to większej ilości informacji. Ilość danych przesyłanych przez sieć Profinet przez cały czas trwania eksperymentów był niezmienny, skonfigurowany na tym samym, stałym poziomie. Wyniki pomiarów zostały przedstawione w *Tabeli 6* oraz na *Rysunku 10.24*. Wraz ze wzrostem ilości przesyłanych danych przez system OPC UA (co związane jest ze wzrostem liczby klientów w sieci) dane w sieci Profinet przesyłane są bez przerw w stabilny sposób. Połączenie realizowane jest w sposób bezprzewodowy, co dodatkowo powinno mieć wpływ na realizację komunikacji.

Tabela 8 Ilość przesłanych informacji Profinet oraz OPC UA przez sieć przemysłową.

Konfiguracja pomiarowa	**Dane wysłane [Mb]**		**Dane odebrane [Mb]**	
	SUMA	**Profinet**	**SUMA**	**Profinet**
Profinet, Serwer OPC UA	0.41	0.37	0.37	0.37
Profinet, OPC UA – 1 klient	2.26	0.37	2.20	0.37
Profinet, 2 klientów, 2 urządzenia	3.90	0.37	3.85	0.37
Profinet, 4 klientów, 4 urządzenia	7.21	0.37	7.17	0.37
Profinet, 6 klientów, 6 urządzęń	10.52	0.37	10.48	0.37
Profinet, 8 klientów, 6 urządzęń	13.43	0.37	13.39	0.37
Profinet, 10 klientów, 6 urządzęń	17.71	0.37	17.69	0.37



Rysunek 9.24 Ilość przesłanych i odebranych danych

Porównując ilość przesłanych i odebranych danych w nie zaobserwowano znacznej ilości utraconych informacji. W sieci Profinet 100% wysłanych informacji trafiło do odbiorcy. W przypadku ruchu OPC UA utrata ramek początkowo osiąga poziom 1%, w kolejnych pomiarach w praktyce osiąga wartość 0% lub jest na granicy błędu pomiarowego. *(Rysunek 10.25).* Dodatkowym parametrem przenalizowanym w badaniach jest wartość parametru „jitter”. Wyniki zaprezentowano w *Tabeli 9.*

Tabela 9 Parametr "jitter" dla długości cyklu sieci 128ms

Konfiguracja pomiarowa	**2.4 Ghz**	**5 Ghz**
Brak serwera OPC	14.1%	18.8%
Profinet, Serwer OPC UA	20.4%	17.6%
Profinet, 1 klient, 1 urządzenie	39.2%	19.4%
Profinet, 2 klientów, 2 urządzenia	32.5%	17.1%
Profinet, 4 klientów, 2 urządzenia	35.7%	17.3%
Profinet, 6 klientów, 2 urządzenia	37.0%	19.8%

Dla częstotliwości działania sieci 5GHz osiągnięto wartość parametru „jitter” dwukrotnie mniejszą niż dla częstotliwości 2.4GHz. Taka wartość jest zbliżona do wartości osiąganej bez uruchomionego serwera OPC UA. W przypadku pracy na częstotliwości 2.4GHz w najgorszym przypadku sieć osiąga czas odpowiedzi dłuższy o prawie 40%. Taka wartość w wielu systemach może być nie do zaakceptowania.

Wyniki badań zaprezentowane w tej części pokazują, że nie tylko zastosowanie sieci SDN może mieć negatywny wpływ na działanie systemu sterowania. Także w przypadku skorzystania z OPC UA wymagane jest dopasowanie parametrów sieci do charakteru systemu i wymagań narzuconych przez projektanta.

10 Podsumowanie

Czy istnieje możliwość wykorzystania urządzeń wbudowanych jako kontrolerów sieci definiowanych? Czy możemy wykorzystać sieci definiowane programowo w projektach przemysłowego Internetu rzeczy? A także czy sieci tego typu mogą integrować różne systemy? Te trzy pytania odwołują się bezpośrednio do postawionych we wstępie tez i na wszystkie te pytania odpowiedź jest twierdząca. Twierdząca chociaż w każdym z przypadków, a także w każdym scenariuszu przeprowadzonych badań należy brać pod uwagę elementy zewnętrzne mające wpływ na poprawność działania całości systemu, a nie tylko konkretnego urządzenia.

Wykorzystując urządzenia wbudowane jako kontroler sieci SDN musimy określić, czy wykorzystywane urządzenie będzie w stanie realizować zadania zgodnie z narzuconym rygorem czasowym. Na pracę takiego urządzenia wpływ może mieć nie tylko realizowana aplikacja, ale również możliwości interfejsów np. szybkość interfejsu sieciowego typu Ethernet czy możliwość komunikacji wykorzystując połączenie szeregowe RS232/RS485.

Na większości urządzeń, które zostały uwzględnione w części badawczej możliwe było zainstalowanie systemu operacyjnego Linux z uruchomionymi usługami sieciowymi w postaci kontrolera SDN wraz z opcjonalnym interfejsem graficznym. Takie rozwiązanie pozwala na podłączenie kontrolera bezpośrednio do części infrastruktury sieciowej bez potrzeby uruchamiania dodatkowych serwerów usług. Dodatkowo wprowadzenie kilku kontrolerów bazujących na urządzeniach wbudowanych może poprawić „stabilność" pracy sieci poprzez zapewnienie redundancji sterowania ruchem sieciowym, a także przeniesienie części zadań (reguł sterujących) pomiędzy różnymi segmentami sieci i urządzeniami transmisyjnymi.

Wprowadzenie przełączników sieciowych bazujących na urządzeniach Mikrotik oraz wykorzystanie wirtualnego przełącznika sieciowego Open vSwitch pozwoliło w praktyce przeanalizować możliwości wprowadzenia sieci definiowanych programowo do infrastruktury sieciowej systemów sterowania.

Udowodniono także, że dzięki zastosowaniu sieci definiowanych programowo możliwe jest dynamicznie zmienianie topologii sieciowej czy filtrowanie określonego ruchu sieciowego np. na podstawie pola Ethertype czy interfejsu docelowego. Zaletą takiego rozwiązania jest możliwość łączenia kilku wpisów co pozwala na budowanie nie tylko prostej logiki przesyłania danych, lecz także skomplikowanych reguł opierających swoje działanie na bazie kilku kontrolerów.

Łącząc wszystkie elementy przebadane w pracy i potwierdzeniu przedstawionych tez, należy wziąć pod uwagę również rodzaj wykorzystywanych urządzeń. Inaczej sytuacja będzie wyglądała, jeśli w całości infrastruktury sieciowej użytkownik będzie pracował z urządzeniami dedykowanymi dla przemysłu, inaczej w momencie pracy z urządzeniami typu „Soft Real Time", a jeszcze inaczej w infrastrukturze łączonej. Każdy z etapów miał znaczący wpływ na określenie zależności, które to umożliwiają uruchomienie całej infrastruktury SDN w sieci przemysłowej. Dalszy rozwój zarówno samych urządzeń, algorytmów zmian reguł ruchu sieciowego, a także narzędzi pozwalających na dokonywanie takich zmian będzie miał znaczący wpływ na przedsięwzięcia realizowane w przyszłości. Tak jak obecnie sieci SDN są wykorzystywane do połączeń dużych centrów danych, tak w przyszłości może być możliwa integracja kilku oddziałów firmy działających w praktyce na całym świecie. Może nie do końca mówimy tutaj o samym procesie sterowania, ale analizie parametrów sterujących przesyłanych do centrów obliczeniowych czy zespołów zarządzających.

Teza: Możliwe jest zastosowanie urządzeń wbudowanych wspierających system operacyjny Linux i dostępnych na rynku jako kontrolerów sieci definiowanych programowo.

Teza: Możliwe jest wykorzystanie sieci definiowanych programowo w projektach przemysłowych w celu zarządzania urządzeniami przemysłowego Internetu rzeczy (Industrial Internet of Things.)

Teza: Możliwe jest wykorzystanie sieci definiowanych programowo w projektach przemysłowych jako integrator różnych systemów stosowanych obecnie i w przyszłości.

Przedstawione powyżej tezy zostały potwierdzone. Należy jednak mieć na uwadze pewne okoliczności, w których dane rozwiązania będą pracować. Bez dopasowania sprzętu do warunków systemu przemysłowego, działanie będzie nieprawidłowe. Jednakże, podobnie jak przy projektowaniu systemu przemysłowego dobieramy sprzęt do kontrolowanego procesu, także w przypadku stosowania nowoczesnych rozwiązań jakimi są niewątpliwie sieci definiowane programowo wymaga spełnienia określonych warunków. Dodatkowo zastosowanie urządzeń wbudowanych jako kontrolerów sieci SDN może wpłynąć pozytywnie na działanie systemu nie tylko pod kątem wydajnościowym, ale również finansowym.

11 Spis rysunków

Rysunek 3.1 Piramida automatyzacji 9
Rysunek 3.2 Model ISO komunikacji sieciowej 10
Rysunek 3.3Model referencyjny RAMI 4.0 systemu przemysłowego 11
Rysunek 3.4 Podstawowe elementy składowe systemu przemysłowego 12
Rysunek 3.5 Wykres przedstawiający działanie systemów czasu rzeczywistego z łagodnymi ograniczeniami czasowymi 13
Rysunek 3.6 Wykres przedstawiający działanie systemów czasu rzeczywistego z mocnymi ograniczeniami czasowymi 14
Rysunek 3.7 Wykres przedstawiający działanie systemów czasu rzeczywistego z ostrymi ograniczeniami czasowymi 14
Rysunek 3.8 Przebieg czasowy generowania odpowiedzi przez system sterowania 15
Rysunek 3.9 Cykl pracy sterownika PLC 16
Rysunek 3.10 Schemat przedstawiający połączenie w topologii magistrali..... 18
Rysunek 3.11 Schemat przedstawiający połączenie w topologii liniowej 19
Rysunek 3.12 Schemat przedstawiający połączenie w topologii pierścieniowej 20
Rysunek 3.13 Schemat przedstawiający połączenie w topologii gwiazdy 21
Rysunek 3.14 Schemat przedstawiający połączenie w topologii typu drzewo.... 22
Rysunek 3.15 Schemat przedstawiający połączenie w topologii typu siatka (ang. MESH) 23
Rysunek 3.16 Porównanie realizacji wymian danych w sieci Powerlink oraz wymian z wykorzystaniem wymiany żetonu. 24
Rysunek 3.17 Opis protokołu wielodostępowego CSMA 25
Rysunek 3.18 Wykres prezentujący udział poszczególnych sieci w komunikacji przemysłowej 27
Rysunek 3.19 Wykres prezentujący udział poszczególnych protokołów w komunikacji przemysłowej z wykorzystaniem sieci Ethernet 28
Rysunek 3.20 Zasada działania protokołu Modbus TCP 29

Rysunek 3.21 Zasada działania sieci EtherCAT – wymiana danych w urządzeniach typu Slave 30
Rysunek 3.22 Porównanie warstw w klasycznym modelu ISO oraz sieci EtherCAT 31
Rysunek 3.23 Schemat działania sieci Profinet IO 32
Rysunek 3.24 Schemat sieci pracującej z protokołem MQTT 34
Rysunek 5.1 Schemat ideowy sieci SDN 39
Rysunek 5.2 Sieć SDN wraz z urządzeniami końcowymi podłączonymi do infrastruktury 41
Rysunek 5.3 Informacja o tabeli przepływów konfigurowanej przez protokół OpenFlow 42
Rysunek 5.4 Schemat blokowy prezentujący działanie programu sterującego w języku P4 46
Rysunek 5.5 Wirtualizacja usług sieciowych 47
Rysunek 5.6 Schemat przedstawiający funkcjonalności związane z sieciami SDN zbudowanymi w oparciu o przełączniki Open vSwitch 48
Rysunek 5.7 Konfiguracja przełącznika sieciowego SDN opartego o urządzenie Mikrotk wraz z oprogramowaniem OpenWRT 49
Rysunek 5.8 Kontroler OpenDayLight 51
Rysunek 5.9 Panel zarządzający OpenFlow Management 52
Rysunek 5.10 Kontroler ONOS 54
Rysunek 7.1 Topologia sieciowa składająca się z trzech przełączników SDN oraz 8 urządzeń końcowych. 61
Rysunek 7.2 Schemat połączeń w sieci Powerlink 61
Rysunek 7.3 Opis parametrów definiujących sieci PowerLink 62
Rysunek 7.4 Schemat przedstawiający topologię liniową zasymulowaną w narzędziu Mininet 63
Rysunek 7.5 Schemat przedstawiający topologię typu gwiazda zasymulowana w narzędziu Mininet 64
Rysunek 7.6 Wynik testów przeprowadzonych w narzędziu Mininet polegających na aktywacji odpowiednich połączeń - 80% ruchu jest blokowana 65

Rysunek 7.7 Wynik testów przeprowadzonych w narzędziu Mininet polegających na aktywacji odpowiednich połączeń - 40% ruchu jest blokowane 65
Rysunek 7.8 Model symulacji narzędzia Mininet w sieciach bezprzewodowych SDN 66
Rysunek 7.9 Topologia liniowa (w odniesieniu do połączenia pomiędzy przełącznikami) zrealizowana w środowisku testowym GNS3 67
Rysunek 7.10 Topologia typu siatka (MESH) zrealizowana w środowisku testowym GNS3 67
Rysunek 7.11 Topologia liniowa (w odniesieniu do połączenia pomiędzy przełącznikami) zrealizowana w środowisku testowym GNS3 z urządzeniami na końcach sieci 68
Rysunek 8.1 Infrastruktura sieciowa wybrana do opracowania modelu badawczego 69
Rysunek 8.2 Schemat sieci ze zdefiniowanym połączeniem pomiędzy urządzeniami „A" oraz „B" 73
Rysunek 9.1 Zasada działania bramy wieloprotokołowej opartej na urządzeniu wbudowanym 79
Rysunek 9.2 Stanowisko laboratoryjne do analizy szybkości działania urządzeń wbudowanych z programem sterującym typowym dla sterownika PLC..... 80
Rysunek 9.3 Schemat sieci EtherCAT wykorzystywanej do testów z urządzeniami wbudowanymi. 81
Rysunek 9.4 Piramida prezentująca miejsce zbieranie danych przez przełącznik SDN 83
Rysunek 9.5 Schemat przełącznika sieciowego zbudowanego na bazie urządzenia wbudowanego Raspberry Pi 84
Rysunek 9.6 Schemat podłączenia przełącznika SDN w infrastrukturze pozwalającej na dublowanie ruchu sieciowego 85
Rysunek 9.7 Schemat proponowanej topologii sieciowej 86
Rysunek 9.8 Logiczna zmiana topologii sieciowej przy użyciu przełączników SDN 87

Rysunek 9.9 Prezentacja połączeń w przełączniku SDN wraz z kierunkiem przesyłu informacji 88
Rysunek 9.10 Schemat połączenia bezprzewodowego pomiędzy urządzeniami w sieci EtherCAT 90
Rysunek 9.11 Komunikacja bezprzewodowa pomiędzy dwoma segmentami sieci przemysłowej 90
Rysunek 9.12 Schemat sieci wykorzystywanej do połączenia różnych standardów przesyłania danych przy użyciu tej samej bezprzewodowej infrastruktury sieciowej 92
Rysunek 9.13 Schemat prezentujący elementy integrowane przez rozwiązanie OPC UA 93
Rysunek 9.14 Zasada działania bramy Przemysłowego Internetu Rzeczy 95
Rysunek 9.15 Zasada działania bramy Przemysłowego Internetu Rzeczy 96
Rysunek 10.1 Wykres prezentujący pracę Raspberry Pi – bez obciążenia cyclic 2ms 98
Rysunek 10.2 Wykres prezentujący pracę Raspberry Pi z dodatkowym obciążeniem freewhelling 99
Rysunek 10.3 Wykres prezentujący pracę Raspberry Pi Modbus TCP – bez obciążenia cyclic 2ms 100
Rysunek 10.4 Wykres prezentujący pracę Raspberry Pi EtherCAT – bez obciążenia cyclic 2ms 101
Rysunek 10.5 Wykres prezentujący pracę Raspberry Pi EtherCAT – bez obciążenia cyclic 4ms 102
Rysunek 10.6 Wykres prezentujący pracę BeagleBone Black – bez obciążenia cyclic 2ms 104
Rysunek 10.7 Wykres prezentujący pracę BeagleBone Black – z obciążeniem 105
Rysunek 10.8 Wykres prezentujący pracę BeagleBone Black – Modbus TCP. 106
Rysunek 10.9 Wykres prezentujący pracę BBB – EtherCAT 2ms 107
Rysunek 10.10 Wykres prezentujący pracę BBB – EtherCAT 4ms 108
Rysunek 10.11 Wyniki prezentujące pracę sieci RealTime Ethernet bez przełącznika SDN 110

Rysunek 10.12 Wyniki prezentujące pracę sieci z przełącznikiem sieciowym SDN zbudowanym na bazie urządzenia wbudowanego Raspberry Pi. 111
Rysunek 10.13 Aktywna reguła przesyłająca dane do urządzenia "Slave" numer jeden 112
Rysunek 10.14 Aktywna reguła przesyłająca dane do urządzenia "Slave" numer dwa 113
Rysunek 10.15 Rozkład czasu odpowiedzi w sieci EtherCAT bez przełączników SDN 114
Rysunek 10.16 Rozkład czasu odpowiedzi w sieci EtherCAT z urządzeniami SDN 115
Rysunek 10.17 Opóźnienia wprowadzane przez kolejne urządzenia w sieci EtherCAT 116
Rysunek 10.18 Opóźnienia wprowadzane przez kolejne urządzenia w sieci EtherCAT 117
Rysunek 10.19 Komunikacja w sieci EtherCAT z cyklem 10ms 118
Rysunek 10.20 Komunikacja w sieci EtherCAT z cyklem 10ms oraz wykorzystaniem przełączników sieciowych SDN. 119
Rysunek 10.21 Komunikacja w sieci EtherCAT z cyklem 10ms, wykorzystaniem przełączników sieciowych SDN oraz komunikacji bezprzewodowej WiFi. 120
Rysunek 10.22 Komunikacja bezprzewodowa EtherCAT z cyklem sieci 64ms121
Rysunek 10.23 Komunikacja bezprzewodowa EtherCAT z cyklem sieci 128ms 121
Rysunek 10.24 Ilość przesłanych i odebranych danych 123

12 Bibliografia

[1] M. N. Hassan Reza, C. Agamudai Nambi Malarvizhi, S. Jayashree, i M. Mohiuddin, „Industry 4.0–Technological Revolution and Sustainable Firm Performance”, w *2021 Emerging Trends in Industry 4.0 (ETI 4.0)*, Raigarh, India: IEEE, maj 2021, s. 1–6. doi: 10.1109/ETI4.051663.2021.9619363.

[2] A.-L. Kampen, M. Fojcik, R. Cupek, i J. Stoj, „Low-Level Wireless and Sensor Networks for Industry 4.0 Communication – Presentation”, w *Advances in Computational Collective Intelligence*, K. Wojtkiewicz, J. Treur, E. Pimenidis, i M. Maleszka, Red., w Communications in Computer and Information Science, vol. 1463. Cham: Springer International Publishing, 2021, s. 474–484. doi: 10.1007/978-3-030-88113-9_38.

[3] E. Kadiyala, S. Meda, R. Basani, i S. Muthulakshmi, „Global industrial process monitoring through IoT using Raspberry pi”, w *2017 International Conference on Nextgen Electronic Technologies: Silicon to Software (ICNETS2)*, mar. 2017, s. 260–262. doi: 10.1109/ICNETS2.2017.8067944.

[4] S. Jain, A. Vaibhav, i L. Goyal, „Raspberry Pi based interactive home automation system through E-mail”, w *2014 International Conference on Reliability Optimization and Information Technology (ICROIT)*, Faridabad, Haryana, India: IEEE, luty 2014, s. 277–280. doi: 10.1109/ICROIT.2014.6798330.

[5] P. F. S. de Melo i E. P. Godoy, „Controller Interface for Industry 4.0 based on RAMI 4.0 and OPC UA”, w *2019 II Workshop on Metrology for Industry 4.0 and IoT (MetroInd4.0&IoT)*, cze. 2019, s. 229–234. doi: 10.1109/METROI4.2019.8792837.

[6] T. Akima i K. Shibata, „Development of real-time ethernet based I/O network”, w *2008 SICE Annual Conference*, Chofu: IEEE, sie. 2008, s. 83–86. doi: 10.1109/SICE.2008.4654627.

[7] J. Stój, „State Machine of a Redundant Computing Unit Operating as a Cyber-Physical System Control Node with Hot-Standby Redundancy”, w *Information Systems Architecture and Technology: Proceedings of 40th Anniversary International Conference on Information Systems Architecture and Technology – ISAT 2019*, J. Świątek, L. Borzemski, i Z. Wilimowska, Red., w Advances in Intelligent Systems and Computing, vol. 1051. Cham: Springer International Publishing, 2020, s. 74–85. doi: 10.1007/978-3-030-30604-5_7.

[8] J. Stoj, „Cost-Effective Hot-Standby Redundancy With Synchronization Using EtherCAT and Real-Time Ethernet Protocols”, *IEEE Trans. Automat. Sci. Eng.*, t. 18, nr 4, s. 2035–2047, paź. 2021, doi: 10.1109/TASE.2020.3031128.

[9] F. Basile, P. Chiacchio, i D. Gerbasio, „On the Implementation of Industrial Automation Systems Based on PLC”, *IEEE Transactions on Automation Science and Engineering*, t. 10, nr 4, s. 990–1003, paź. 2013, doi: 10.1109/TASE.2012.2226578.

[10] „IEC 61131-3:2013 | IEC Webstore | water automation, water management, smart city”. https://webstore.iec.ch/publication/4552 (dostęp 7 lipiec 2023).

[11] J. Masino, M. Frey, F. Gauterin, i R. Sharma, *Development of a highly accurate and low cost measurement device for Field Operational Tests*. 2016. doi: 10.1109/ISISS.2016.7435548.
[12] W. Rząsa i D. Rzonca, „Event-Driven Approach to Modeling and Performance Estimation of a Distributed Control System", w *Computer Networks*, P. Gaj, A. Kwiecień, i P. Stera, Red., w Communications in Computer and Information Science. Cham: Springer International Publishing, 2016, s. 168–179. doi: 10.1007/978-3-319-39207-3_15.
[13] M. Jamro i D. Rzonca, „Impact of Communication Timeouts on Meeting Functional Requirements for IEC 61131–3 Distributed Control Systems", *Automatika*, t. 56, nr 4, s. 499–507, sty. 2015, doi: 10.1080/00051144.2015.11828663.
[14] T. Muller i H. Dermot Doran, „Protecting PROFINET cyclic real-time traffic: A performance evaluation and verification platform", w *2018 14th IEEE International Workshop on Factory Communication Systems (WFCS)*, Imperia: IEEE, cze. 2018, s. 1–4. doi: 10.1109/WFCS.2018.8402379.
[15] X. Wu, L. Xie, i F. Lim, „Network delay analysis of EtherCAT and PROFINET IRT protocols", *IECON 2014 - 40th Annual Conference of the IEEE Industrial Electronics Society*, s. 2597–2603, paź. 2014, doi: 10.1109/IECON.2014.7048872.
[16] S.-M. Park, H.-W. Kim, H.-J. Kim, i J.-Y. Choi, „Accuracy Improvement of Master–Slave Synchronization in EtherCAT Networks", *IEEE Access*, t. 8, s. 58620–58628, 2020, doi: 10.1109/ACCESS.2020.2982704.
[17] Q. Liu i Q. Liu, „A Study on Topology in Computer Network", w *2014 7th International Conference on Intelligent Computation Technology and Automation*, paź. 2014, s. 45–48. doi: 10.1109/ICICTA.2014.18.
[18] Y. Zhang i J. Sun, „Topology analysis of wireless mesh network based on 802.11a", w *The 27th Chinese Control and Decision Conference (2015 CCDC)*, maj 2015, s. 5978–5980. doi: 10.1109/CCDC.2015.7161881.
[19] „IEEE Standards for Local and Metropolitan Area Networks: Specification for 802.3 Full Duplex Operation", *IEEE Std 802.3x-1997 and IEEE Std 802.3y-1997 (Supplement to ISO/IEC 8802-3: 1996/ANSI/IEEE Std 802.3, 1996 Edition)*, s. 1–324, lis. 1997.
[20] A. Romanov i E. Slepynina, „Real-time Ethernet POWERLINK Communication for ROS. Part I. General Concept", w *2020 Ural Smart Energy Conference (USEC)*, Ekaterinburg, Russia: IEEE, lis. 2020, s. 159–162. doi: 10.1109/USEC50097.2020.9281198.
[21] „POWERLINK - Basics". [Online]. Dostępne na: https://www.ethernet-powerlink.org/uploads/media/POWERLINKBasics_brochure_e.pdf
[22] W. Xiaofan, P. H. J. Chong, i L. W. Yie, „Performance comparison of CSMA/CD, CSMA/CA, CSMA/RI, CSMA/PRI and CSMA/PR with BEB", w *2010 5th IEEE Conference on Industrial Electronics and Applications*, cze. 2010, s. 1843–1848. doi: 10.1109/ICIEA.2010.5515401.
[23] S. Tamboli, M. Rawale, R. Thoraiet, i S. Agashe, „Implementation of Modbus RTU and Modbus TCP communication using Siemens S7-1200 PLC for batch process", w *2015 International Conference on Smart Technologies*

and Management for Computing, Communication, Controls, Energy and Materials (ICSTM), maj 2015, s. 258–263. doi: 10.1109/ICSTM.2015.7225424.
[24] D. Zaheri i M. H. Refan, „Design and Implementation of Modbus RTU/TCP to Profibus Gateway Using Raspberry Pi", w *2023 15th International Conference on Computer and Automation Engineering (ICCAE)*, mar. 2023, s. 109–113. doi: 10.1109/ICCAE56788.2023.10111395.
[25] V. Q. Nguyen i J. W. Jeon, „EtherCAT network latency analysis", w *2016 International Conference on Computing, Communication and Automation (ICCCA)*, Greater Noida, India: IEEE, kwi. 2016, s. 432–436. doi: 10.1109/CCAA.2016.7813815.
[26] K. Benzekki, A. El Fergougui, i A. Elbelrhiti Elalaoui, „Software-defined networking (SDN): a survey: Software-defined networking: a survey", *Security Comm. Networks*, t. 9, nr 18, s. 5803–5833, grudz. 2016, doi: 10.1002/sec.1737.
[27] N. McKeown *i in.*, „OpenFlow: enabling innovation in campus networks", *SIGCOMM Comput. Commun. Rev.*, t. 38, nr 2, s. 69–74, mar. 2008, doi: 10.1145/1355734.1355746.
[28] R. Wazirali, R. Ahmad, i S. Alhiyari, „SDN-OpenFlow Topology Discovery: An Overview of Performance Issues", *Applied Sciences*, t. 11, nr 15, s. 6999, lip. 2021, doi: 10.3390/app11156999.
[29] D. T. Bui i K. Aberkane, „A generic interface for Open vSwitch", w *2016 IEEE NetSoft Conference and Workshops (NetSoft)*, cze. 2016, s. 53–57. doi: 10.1109/NETSOFT.2016.7502442.
[30] P. Gorja i R. Kurapati, „Extending open vSwitch to L4-L7 service aware OpenFlow switch", w *2014 IEEE International Advance Computing Conference (IACC)*, luty 2014, s. 343–347. doi: 10.1109/IAdCC.2014.6779346.
[31] „The First Application — Ryu 4.34 documentation". https://ryu.readthedocs.io/en/latest/writing_ryu_app.html (dostęp 7 lipiec 2023).
[32] A. B. D. Kinabo, J. B. Mwangama, i A. A. Lysko, „Towards Wi-Fi-based Time Sensitive Networking Using OMNeT++/NeSTiNg Simulation Models", w *2021 International Conference on Electrical, Computer and Energy Technologies (ICECET)*, Cape Town, South Africa: IEEE, grudz. 2021, s. 1–6. doi: 10.1109/ICECET52533.2021.9698580.
[33] G. Pujolle, *Software Networks: Virtualization, SDN, 5G and Security*, 1. wyd. Wiley, 2020. doi: 10.1002/9781119694748.
[34] P. Gaj, M. Skrzewski, J. Stoj, i J. Flak, „Virtualization as a Way to Distribute PC-Based Functionalities", *IEEE Trans. Ind. Inf.*, t. 11, nr 3, s. 763–770, cze. 2015, doi: 10.1109/TII.2014.2360499.
[35] G. Ndonda i R. Sadre, *A low-delay SDN-based countermeasure to eavesdropping attacks in industrial control systems*. 2017, s. 7. doi: 10.1109/NFV-SDN.2017.8169840.

[36] A. Sallahi i M. St-Hilaire, „Optimal Model for the Controller Placement Problem in Software Defined Networks”, *IEEE Commun. Lett.*, t. 19, nr 1, s. 30–33, sty. 2015, doi: 10.1109/LCOMM.2014.2371014.
[37] I. Frieslaar i B. Irwin, „Investigating the electromagnetic side channel leakage from a Raspberry Pi”, w *2017 Information Security for South Africa (ISSA)*, sie. 2017, s. 24–31. doi: 10.1109/ISSA.2017.8251771.
[38] P. Gaj i M. Maćkowski, „Electromagnetic compatibility issues in hybrid wired and wireless industrial networks”, *PLoS ONE*, t. 15, nr 5, s. e0232405, maj 2020, doi: 10.1371/journal.pone.0232405.
[39] M. Kreitlow, F. Sabath, i H. Garbe, *Influence of software effects on the susceptibility of Ethernet connections*, t. 2014. 2014. doi: 10.1109/ISEMC.2014.6899031.
[40] S. Smys, J. Thara Prakash, i J. S. Raj, „Conducted Emission Reduction by Frequency Hopping Spread Spectrum Techniques”, *Natl. Acad. Sci. Lett.*, t. 38, nr 3, s. 197–201, cze. 2015, doi: 10.1007/s40009-014-0324-6.
[41] D. Molloy, *Exploring BeagleBone: tools and techniques for building with embedded Linux*. Indianapoli, IN: John Wiley & Sons, 2015.
[42] S. Sahitya, H. Lokesha, i L. K. Sudha, „Real time application of Raspberry Pi in compression of images”, *2016 IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT)*, s. 1047–1050, maj 2016, doi: 10.1109/RTEICT.2016.7807990.
[43] I. Smołka, „Embedded Systems Based on Raspberry Pi Programmed with CODESYS”, *Studia Informatica*, t. Vol 39, s. 77-87 Pages, maj 2018, doi: 10.21936/SI2018_V39.N1.839.
[44] J. Stój, I. Smołka, i M. Maćkowski, „Determining the Usability of Embedded Devices Based on Raspberry Pi and Programmed with CODESYS as Nodes in Networked Control Systems”, w *Computer Networks*, P. Gaj, M. Sawicki, G. Suchacka, i A. Kwiecień, Red., w Communications in Computer and Information Science, vol. 860. Cham: Springer International Publishing, 2018, s. 193–205. doi: 10.1007/978-3-319-92459-5_16.
[45] J. Stój i I. Smołka, „Temporal Characteristics of CodeSys Programmed Raspberry Pi and Beaglebone Black Embedded Devices”, w *Computer Networks*, P. Gaj, M. Sawicki, i A. Kwiecień, Red., w Communications in Computer and Information Science, vol. 1039. Cham: Springer International Publishing, 2019, s. 156–167. doi: 10.1007/978-3-030-21952-9_12.
[46] R. Amin, M. Reisslein, i N. Shah, „Hybrid SDN Networks: A Survey of Existing Approaches”, *IEEE Commun. Surv. Tutorials*, t. 20, nr 4, s. 3259–3306, 2018, doi: 10.1109/COMST.2018.2837161.
[47] J. Stój, A. Ziębiński, i R. Cupek, „FPGA based Industrial Ethernet Network Analyser for Real-time Systems Providing Openness for Industry 4.0”, *Enterprise Information Systems*, t. 16, nr 10–11, s. 1711–1731, paź. 2022, doi: 10.1080/17517575.2021.1948613.
[48] I. Smołka, J. Stój, i M. Fojcik, „Application of Software Defined Networks for Collection of Process Data in Industrial Real-Time Systems”, w *Advances in Computational Collective Intelligence*, C. Bădică, J. Treur, D. Benslimane, B. Hnatkowska, i M. Krótkiewicz, Red., w Communications in Computer and

Information Science, vol. 1653. Cham: Springer International Publishing, 2022, s. 446–458. doi: 10.1007/978-3-031-16210-7_37.
[49] I. Smołka i J. Stój, „Utilization of SDN Technology for Flexible EtherCAT Networks Applications”, *Sensors*, t. 22, nr 5, s. 1944, mar. 2022, doi: 10.3390/s22051944.
[50] M. Bolanowski, D. Mazur, M. Oleksy, G. Budzik, i A. Paszkiewicz, „Analysis of possible SDN use in the rapid prototyping process as part of the Industry 4.0”, *Bulletin of the Polish Academy of Sciences: Technical Sciences; 2019; 67; No. 1; 21-30*, 2019, Dostęp: 7 lipiec 2023. [Online]. Dostępne na: https://journals.pan.pl/dlibra/publication/127334/edition/111099
[51] D. Henneke, L. Wisniewski, i J. Jasperneite, „Analysis of realizing a future industrial network by means of Software-Defined Networking (SDN)”, *2016 IEEE World Conference on Factory Communication Systems (WFCS)*, s. 1–4, maj 2016, doi: 10.1109/WFCS.2016.7496525.
[52] I. Smolka, J. Stoj, P. Gaj, i M. Fojcik, „Communication between AGV and standalone station via EtherCAT using WiFi – proof of concept”, w *2022 IEEE International Conference on Big Data (Big Data)*, Osaka, Japan: IEEE, grudz. 2022, s. 6337–6346. doi: 10.1109/BigData55660.2022.10020996.
[53] X. Wu i L. Xie, „On the Wireless Extension of EtherCAT Networks”, w *2017 IEEE 42nd Conference on Local Computer Networks (LCN)*, Singapore: IEEE, paź. 2017, s. 235–238. doi: 10.1109/LCN.2017.15.
[54] A.-L. Kampen, M. Fojcik, R. Cupek, i J. Stoj, „The requirements for using wireless networks with AGV communication in an industry environment”, w *2021 17th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, Bologna, Italy: IEEE, paź. 2021, s. 212–218. doi: 10.1109/WiMob52687.2021.9606399.
[55] A. Morato, S. Vitturi, A. Cenedese, G. Fadel, i F. Tramarin, „The Fail Safe over EtherCAT (FSoE) protocol implemented on the IEEE 802.11 WLAN”, w *2019 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, Zaragoza, Spain: IEEE, wrz. 2019, s. 1163–1170. doi: 10.1109/ETFA.2019.8869503.
[56] D. Bruckner *i in.*, „An Introduction to OPC UA TSN for Industrial Communication Systems”, *Proceedings of the IEEE*, t. PP, s. 1–11, sty. 2019, doi: 10.1109/JPROC.2018.2888703.
[57] J. Stój, A.-L. Kampen, R. Cupek, I. Smołka, i M. Drewniak, „Industrial Shared Wireless Communication Systems—Use Case of Autonomous Guided Vehicles with Collaborative Robot”, *Sensors*, t. 23, nr 1, s. 158, grudz. 2022, doi: 10.3390/s23010158.
[58] R. Cupek, A. Ziebinski, i M. Drewniak, „An OPC UA server as a gateway that shares CAN network data and engineering knowledge”, w *2017 IEEE International Conference on Industrial Technology (ICIT)*, Toronto, ON: IEEE, mar. 2017, s. 1424–1429. doi: 10.1109/ICIT.2017.7915574.
[59] V. Gavriluţ i P. Pop, „Traffic-type Assignment for TSN-based Mixed-criticality Cyber-physical Systems”, *ACM Trans. Cyber-Phys. Syst.*, t. 4, nr 2, s. 1–27, kwi. 2020, doi: 10.1145/3371708.

[60] Y. Shi *i in.*, „Using Machine Learning to Provide Reliable Differentiated Services for IoT in SDN-Like Publish/Subscribe Middleware”, *Sensors*, t. 19, nr 6, s. 1449, mar. 2019, doi: 10.3390/s19061449.
[61] B. Mishra i A. Kertesz, „The Use of MQTT in M2M and IoT Systems: A Survey”, *IEEE Access*, t. 8, s. 201071–201086, 2020, doi: 10.1109/ACCESS.2020.3035849.
[62] Y. Xiaohua i H. Canhui, „Design and Implementation of OpenDayLight Manager Application”, w *2020 13th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*, paź. 2020, s. 977–982. doi: 10.1109/CISP-BMEI51763.2020.9263553.
[63] N. V. Ha, D. D. Quan, i T. T. T. Nguyen, „Graphical User Interface for RYU Software Defined Network Controller”, w *2022 IEEE 8th Information Technology International Seminar (ITIS)*, paź. 2022, s. 312–317. doi: 10.1109/ITIS57155.2022.10010215.
[64] W. Kim, J. Li, J. W.-K. Hong, i Y.-J. Suh, „OFMon: OpenFlow monitoring system in ONOS controllers”, w *2016 IEEE NetSoft Conference and Workshops (NetSoft)*, cze. 2016, s. 397–402. doi: 10.1109/NETSOFT.2016.7502474.
[65] C. Metter, V. Burger, Z. Hu, K. Pei, i F. Wamser, „Evaluation of the Detection Capabilities of the ONOS SDN Controller”, w *2018 IEEE Seventh International Conference on Communications and Electronics (ICCE)*, lip. 2018, s. 96–101. doi: 10.1109/CCE.2018.8465741.
[66] B. A. G. & C. K. Germany Hülshorstweg 20, 33415 Verl, „Industrial Ethernet multi-channel probe”, *Beckhoff Automation*. https://www.beckhoff.com/en-en/products/i-o/ethercat-development-products/elxxxx-etxxxx-fbxxxx-hardware/et2000.html (dostęp 7 lipiec 2023).

13 Uzupełnienie

1. Udział w projektach badawczych:
   a. Pojazd autonomiczny zintegrowany z robotem współpracującym *„Automated Guided Vehicles integrated with Collaborative Robots for Smart Industry Perspective'', acronym: CoBotAGV, registration number: NOR/POLNOR/CoBotAGV/0027/2019–00 obtained funding as part of the framework POLNOR 2019 Polish-Norwegian research projects financed under EEA and Norway Grants for the years 2014-2021*
   b. SDN
      „Technologia do bezpiecznego i niezawodnego dostarczania profesjonalnych przekazów kontrybucyjnych audio/wideo na żywo przy zachowaniu minimalnego możliwego opóźnienia", nr umowy POIR.01.01.01-00-1896/20-00
2. Współpraca z przemysłem:
   a. Bombardier/Alstom – opieka nad laboratorium specjalizowanym, zaprojektowanym i uruchomionym wspólnie z firmą Bombardier/Alstom
   b. Proloc sp. z o. o. – realizacja projektów na rzeczywistych obiektach przemysłowych
      - Fortum Zabrze – przygotowanie komunikacji sieciowej dla obiektu przemysłowego Modbus TCP oraz uruchomienie obiektu stacji uzdatniania wody,
      - Żwirownia Dębina – uruchomienie komunikacji sieciowej Profibus wraz z analizą ruchu sieciowego oraz przygotowanie programu sterującego taśmociągami do transportu materiału,
      - Coal Control – analiza komunikacji sieciowej (Profinet, Ethernet IP, Modbus TCP) w sterownikach PLC Turck oraz dobranie czujników odległości po przeprowadzeniu analizy czasowej działania,

- Wągrowiec – przygotowanie komunikacji sieciowej polegającej na informowaniu operatora o wystąpieniu awarii (wysyłanie wiadomości SMS).

3. Popularyzacja nauki
    a. Organizacja prelekcji w tematyce informatyki przemysłowej dla Technikum nr 1 im. Stanisława Staszica w Rybniku,
    b. Współorganizacja Olimpiady Informatycznej w Gliwicach
    c. Wystąpienia dla szkół powiązane z Wszechnicą Informatyki dla Zespołu Szkół Budowlanych w Rybniku oraz gliwickich szkół średnich.