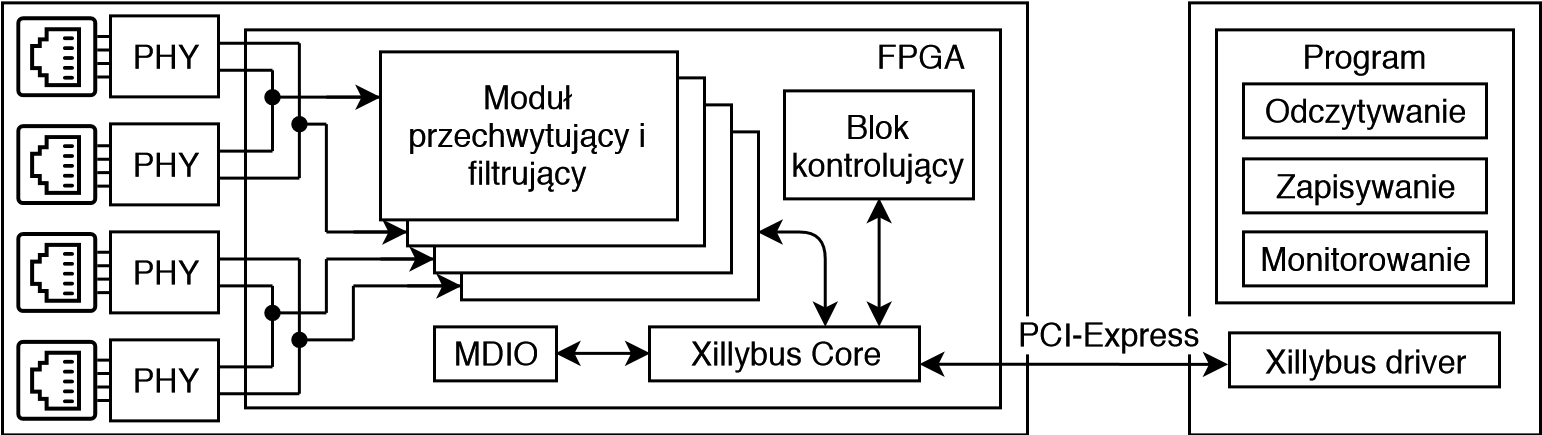
**Sprzętowy system analizy i monitorowania ruchu sieciowego**

Komunikacja pomiędzy kartą, a komputerem zrealizowana jest wykorzystując połączenie PCIExpress. Na rysunku 3.2 został przedstawiony schemat blokowy systemu.

Karta NetFPGA PC



Rysunek 3.2: Schemat blokowy systemu.

Funkcje systemu:

* przechwytywanie ruchu sieciowego,
* filtracja dla zmniejszenia ilości danych,
* zmiana nastaw filtrów w czasie pracy,
* znaczniki czasowe z rozdzielczością 4ns,
* rozłączanie tunelu w wybranym kierunku,
* obsługa pakietów jumbo do 16382 bajtów,
* brak utraty pakietów przy saturacji łącza sieciowego niezależnie od rozmiaru pakietu dla nasłuchiwania dwóch portów.

### 3.1.1 Zasada działania

System posiada dwa tunele, dzięki którym możliwe jest nasłuchiwanie dwóch osobnych połączeń sieciowych w obu kierunkach. Karta odbiera ramkę ethernetową i przekazuje ją do przeciwnego układy PHY, który wysyła ramkę dalej, tworząc wspomniany tunel. Oprócz tego, odebrana ramka jest przekazywana dalej do kolejnej części układu. Następnie do ramki dodawany jest znacznik czasowy o rozdzielczości 4ns oraz długość ramki. Format całego pakietu w tym momencie jest identyczny jak w pliku z rozszerzeniem *.pcap*, dzięki czemu z poziomu programu nie jest wymagane przetwarzanie strumienia pakietów, a wyłącznie odczytywanie i zapisywanie do pliku. Pakiet kolejno przechodzi przez blok filtrujący (dokładny opis w rozdziale 3.2.3), który posiada 4 filtry. Jeżeli nastawy przynajmniej 1 z 4 filtrów zostaną spełnione to dany pakiet zostaje przekazany do bloku Xillybus Core, który następnie z wykorzystaniem połączenia PCI-Express transmituje dane do systemu operacyjnego. Program, który jest uruchomiony na komputerze stacjonarnym w jednym wątku odbiera dane i zapisuje do bufora cyklicznego, a w drugim wątku odczytuje z bufora cyklicznego i zapisuje na dysk twardy do pliku o rozszerzeniu *.pcap*.

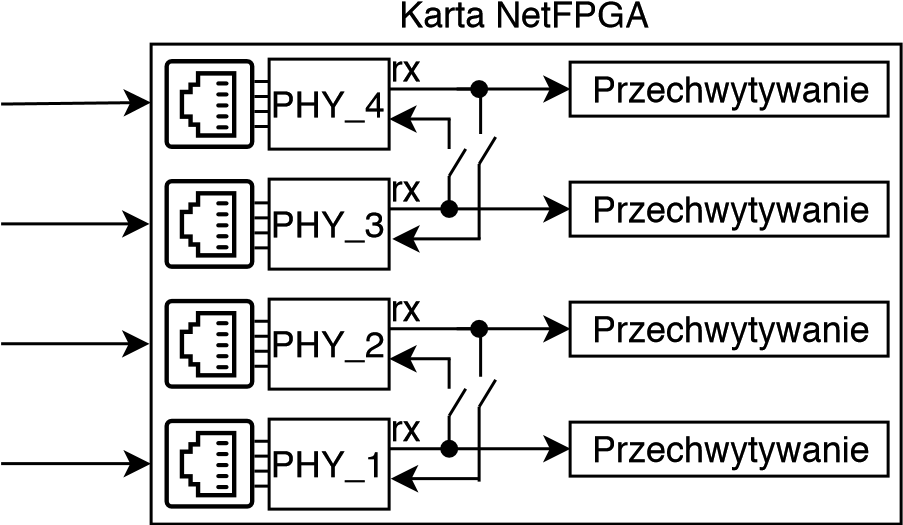
### 3.1.2 Możliwe konfiguracje

System może działać w różnych konfiguracjach. Głównym założeniem systemu jest tworzenie tunelu i monitorowanie ruchu sieciowego danego połączenia, ale system ma także inne możliwości.

#### Przechwytywanie

System po wyłączeniu tuneli jest w stanie odbierać przychodzące dane w standardach 100/1000BASE-T. Schemat konfiguracji został przedstawiony na rysunku

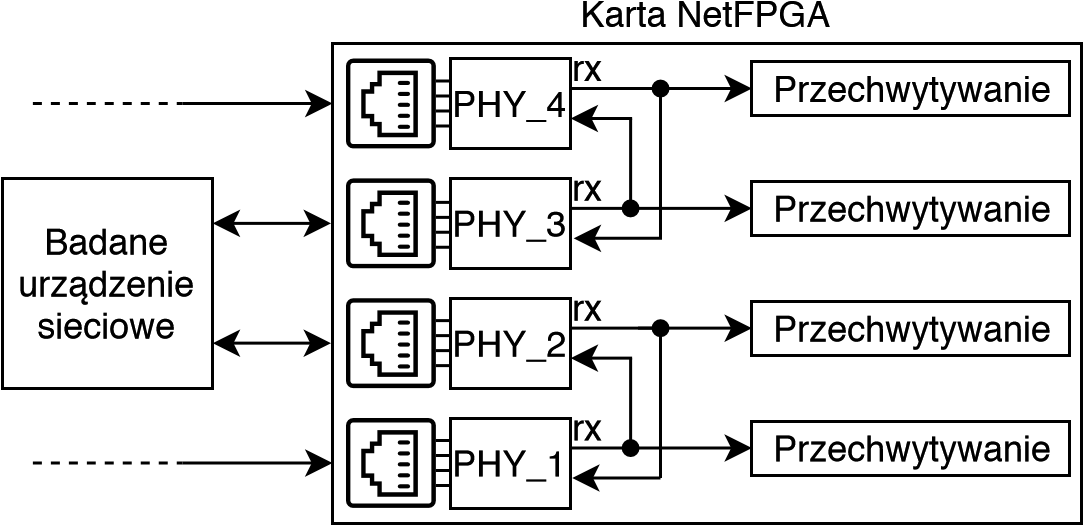
3.3.



Rysunek 3.3: Konfiguracja karty NetFPGA z wyłączonymi tunelami.

#### Pomiar czasu

Dwa tunele umożliwiają precyzyjny pomiar opóźnień wprowadzanych przez badane urządzenie np. przełącznik sieciowy, router albo sterownik przemysłowy. Dzięki precyzyjnym znacznikom czasowym w dwóch miejscach, możliwe jest obliczenie czasu przejścia ramki przez urządzenie. Omawiana konfiguracja została przedstawiona na rysunku 3.4.



Rysunek 3.4: Konfiguracja karty NetFPGA do pomiaru opóźnień wprowadzanych przez badane urządzenie.

### 3.1.3 Ograniczenia teoretyczne

Pierwotnie projekt miał przechwytywać ruch sieciowy przechodzący przez wyłącznie jedno połączenie ethernetowe, jednak warunki okazały się na tyle dobre, że udało się zduplikować rozwiązanie. Ostatecznie system umożliwia przechwytywanie

*3.1. Opis sytemu* dwóch połączeń. Jedynym ograniczeniem systemu jest przepustowość połączenia PCI-Express, która w przypadku zastosowania podstawowej wersji bloku Xillybus wynosi około 400MB*/*s. Pojedynczy strumień danych (których łącznie jest 4) jest w stanie wygenerować przy maksymalnym natężeniu ruchu sieciowego do 131MB*/*s danych (64-bajtowa pełna ramka Ethernet + 8-bajtowa preambuła + 12-bajtowy odstęp między ramkami). Zwiększona ilość danych wynika z dodatkowego narzutu na rozmiar przekazywanej ramki, która poszerzona jest o pola znacznika czasowego i rozmiaru ramki (dodatkowe 16 bajtów). Zależność została przedstawiona na rysunku 3.5.

64

256

512

1

*,*

024

1

*,*

518

126

128

130

132

Przepustowość,MB/s

Długość ramki, bajty

Rysunek 3.5: Zależność między długością ramki ethernetowej, a wymaganą przepustowością połączenia PCI-Express przy pełnym nasyceniu łącza.

### 3.3 Część programowa

Program komputerowy o nazwie *NetCon* jest nierozerwalną częścią prezentowanego systemu. Program spełnia następujące funkcje:

* odbieranie danych z karty NetFPGA i zapisywanie na dysk (rozdział 3.3.3),
* wysyłanie nastaw do bloków filtracyjnych (rozdział 3.3.4),
* sterowanie otwarciem lub zamknięciem tunelu (rozdział 3.3.5),
* obsługa interfejsu MDIO układów PHY (rozdział 3.3.6),
* możliwość wyłączenia funkcji Energy-Efficient Ethernet (rozdział 3.3.7).

### 3.3.2 Komunikacja z blokiem Xillybus

Wymiana danych z blokiem Xillybus z poziomu programu polega na otwarciu pliku o specjalnej nazwie oraz zapisywania lub odczytywania do/z otwartego pliku. Przykładowo, aby odebrać dane ze strumienia to\_host\_read\_32\_1, należy otworzyć plik o nazwie \\.\xillybus\_read\_32\_1 oraz odczytać określoną ilość danych. Pliki urządzenia Xillybus mają swoje ograniczenia w porównaniu do zwykłych plików. Można wyłącznie używać dwóch operacji \_read() oraz \_write().

W programie do tego celu została wykorzystana biblioteka <io.h>. Biblioteka na systemie Windows oferuje metody nisko-poziomowe dostępu do plików, takie jak:

* \_open() - otwiera plik,
* \_read() - czyta z pliku,
* \_write() - zapisuje do pliku,
* \_close() - zamyka plik.

Każdy ze strumieni jest jednokierunkowy, oznacza to, że z pojedynczego strumienia można wyłącznie odczytywać lub zapisywać. Otwarcie strumienia z którego dane będą odczytywane odbywa się poprzez dodanie flagi \_O\_RDONLY, a strumienia do którego dane będą zapisywane flagi \_O\_WRONLY:

int devr = \_open(deviceName, \_O\_RDONLY | \_O\_BINARY)

int devw = \_open(deviceName, \_O\_WRONLY | \_O\_BINARY)

Flagi znajdują się w nagłówku <fcntl.h>. Dodanie flagi \_O\_BINARY jest potrzebne, aby zapobiec interpretacji wartości bajtów przez system (np. wartości 0x1A jako koniec pliku EOF).

Kolejno po otwarciu pliku dane można odczytywać lub zapisywać za pomocą metod:

int rc = \_read(devr, buf, ilosc\_bajtow)   
int wc = \_write(devw, buf, ilosc\_bajtow)

### 3.3.3 Przechwytywanie - *capture*

Funkcja *capture* rozpoczyna odbieranie i zapisywanie przechwytywanego ruchu sieciowego z danego portu do pliku. Wywołanie funkcji odbywa się z wiersza poleceń z następującymi argumentami:

NetCon.exe capture N F S

Gdzie,

* N - numer portu/modułu,
* F - nazwa pliku lub ścieżka (opcjonalna),
* S - rozmiar bufora cyklicznego w megabajtach (opcjonalna).

Przykład minimalnego wywołania:

NetCon.exe capture 1

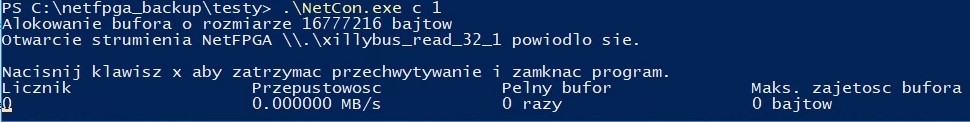
#### Implementacja

Program po inicjalizacji bufora cyklicznego, utworzenia pliku z nagłówkiem *PCAP* i otwarcia strumienia danych z karty, tworzy cztery wątki:

* wątek odczytujący dane ze strumienia (pliku) \\.\xillybus\_read\_32\_N,
* wątek zapisujący do pliku *PCAP*,
* wątek monitorujący,
* wątek czekający na wciśnięcie klawisza.

Po utworzeniu powyższych wątków następuje wysłanie rejestru otwierającego przechwytywanie do bloku kontrolującego (opisanego w rozdziale 3.2.5). Taki mechanizm zapewnia synchronizację i kompletność odbieranych danych.

Wątek odczytujący bezpośrednio odczytuje dane z otwartego strumienia \\.\xillybus\_read\_32\_N do bufora cyklicznego. Wątek zapisujący bezpośrednio zapisuje dane z bufora cyklicznego do pliku. Wątek monitorujący zlicza liczbę bajtów w buforze cyklicznym i wyświetla podstawowe informacje na temat ilości zapisanych danych, przepustowości, przepełnieniach bufora cyklicznego oraz maksymalnym zużyciu pamięci bufora cyklicznego, co zostało pokazane na rysunku 3.24.



Rysunek 3.24: Program w trybie monitoringu ruchu sieciowego z zapisem do pliku.

Ostatni wątek czeka na naciśniecie klawisza “x”. W momencie naciśnięcia klawisza, następuje zmiana wartości zmiennej globalnej, która informuje wątki o rozpoczęciu procesu zamykania programu. Zamknięcie programu rozpoczyna się od wysłania do bloku kontrolującego rejestru, który zatrzymuje przechwytywanie ramek sieciowych na karcie NetFPGA. Kolejno program czeka 100ms na odebranie pozostałych danych, które znajdują się w buforach nadawczych/odbiorczych oraz kończy działanie programu.

### 3.3.4 Nastawy filtrów - *set*

Funkcja *set* służy do wysłania nastaw filtrów dla danego bloku filtracyjnego.

Wywołanie funkcji odbywa się z wiersza poleceń z następującymi argumentami:

**NetCon.exe set N M1 F1 M2 F2 M3 F3 M4 F4**

Gdzie,

* N - numer portu/bloku filtrującego,
* M1 - M4 - minimalna długość ramki (wliczając preambułę) dla 1 - 4 filtru,
* F1 - F4 - nastawy filtra 1 - 4.

Przykład wysłania nastaw filtrów na porcie 1 z przepuszczaniem wszystkich ramek do użytkownika:

NetCon.exe set 1 0

Format nastaw dla pojedynczego filtra wygląda następująco:

"NB\_1 A\_1 B\_1;NB\_2 C\_2;NB\_3 A\_3 B\_3;..."

Gdzie,

* NB - numer bajtu ramki (wliczając preambułę),
* A - liczba od której porównywany bajt ma być większy lub równy,
* B - liczba od której porównywany bajt ma być mniejszy lub równy,
* C - liczba, której porównywany bajt ma być równy.

Przykład wywołania:

**NetCon.exe set 1 80 "9 192 193;10 25;11 120 150;"**

Powyższe argumenty i nastawy filtra oznaczają, że:

* nastawy wysłane mają być do bloku filtracyjnego 1 portu,
* minimalna długość ramki ma wynosić 80 bajtów,
* wartość bajtu ramki na pozycji 9 ma być w przedziale <192,193>,
* wartość bajtu na pozycji 10 ma być równa wartości 25,
* wartość bajtu na pozycji 11 ma znajdować się w przedziale <120;150>.

Wartości można zapisywać w formacie heksadecymalnym (z prefiksem 0x np. 0xF9), decymalnym i oktagonalnym (z prefiksem 0, np. 012). W przypadku nastaw, należy pamiętać, że system przechwytuje całą ramkę włącznie z preambułą. Rzeczywista ramka zaczyna się w takim przypadku po 8 bajtach. Oznacza to, że pierwszy bajt ramki ethernetowej zaczyna się na indeksie 8.

### 3.3.5 Sterowanie tunelami - *bridge*

Funkcja *bridge* oferuje możliwość wyłączania oraz załączania wybranego tunelu.

Wywołanie funkcji odbywa się z wiersza poleceń z następującymi argumentami:

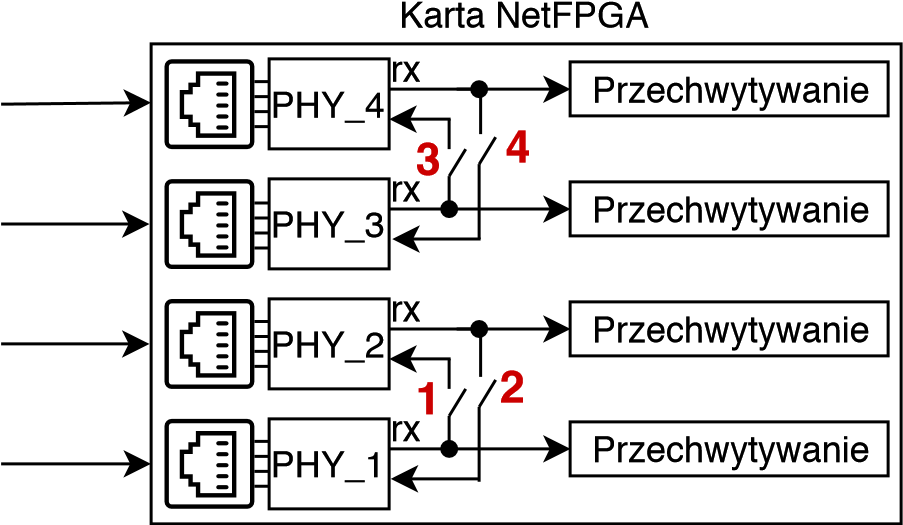
NetCon.exe bridge\_on N

NetCon.exe bridge\_off N NetCon.exe bridge\_all

Gdzie,

*•* N - numer tunelu.

Funkcja bridge\_on załącza określony tunel, funkcja bridge\_off wyłącza okreslony tunel, a funkcja **bridge\_all załącza wszystkie tunele.** Na rysunku 3.3.5 została przedstawiona numeracja tuneli.



Rysunek 3.25: Schemat przedstawiający numerację tuneli.

Przykłady wywołania:

NetCon.exe bridge\_on 1

NetCon.exe bridge\_off 2

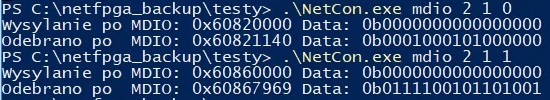
NetCon .exe bridge\_all

### 3.3.6 Rejestry MDIO - *mdio*

Funkcja *mdio* umożliwia zapis oraz odczyt rejestrów układów PHY. Dokładny opis poszczególnych wartości interfejsu znajduje się w rozdziale 2.1.3. Wywołanie funkcji odbywa się z wiersza poleceń z następującymi argumentami:

NetCon.exe mdio op\_code phy\_addr reg\_addr data

Po wywołaniu programu z określonymi argumentami, wyświetlana jest wysłana oraz zwrócona wartość pola danych. Przykładowy odczyt (op\_code = 2) z układu PHY o numerze 1 (phy\_addr = 1), rejestrów 0 i 1 (reg\_addr = 0 i 1) przedstawiono na rysunku 3.26.



Rysunek 3.26: Zrzut ekranu programu po wywołaniu funkcji mdio.

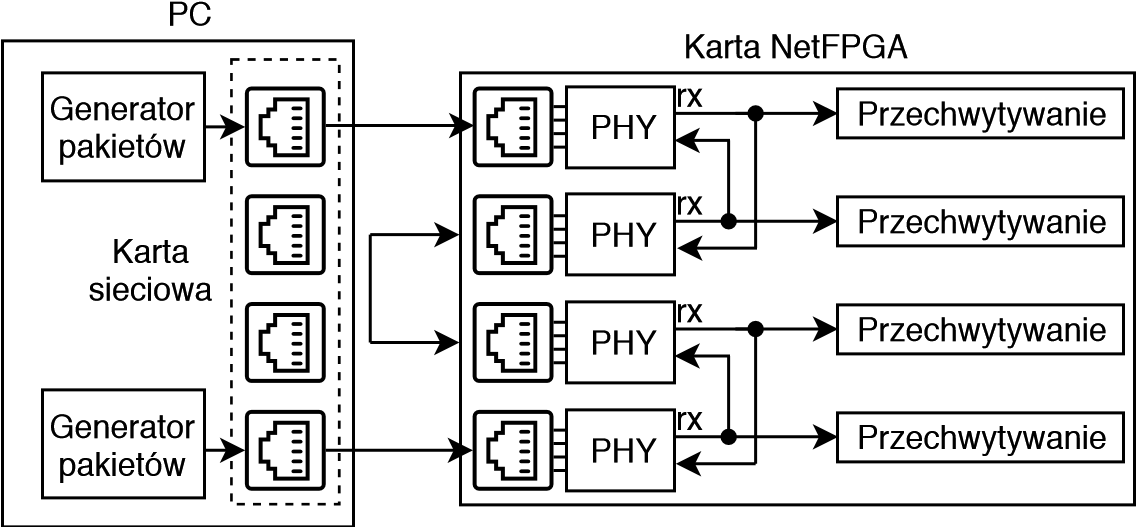
### 3.3.7 Opcja EEE - *eee\_off*

W programie została zaimplementowana funkcja *eee\_off*, która wyłącza opcję *Energy-Efficient Ethernet* na wszystkich układach PHY poprzez interfejs MDIO. Wyłączenie opcji polega na wysłaniu odpowiednich rejestrów dla każdego z układu PHY. Wywołanie funkcji odbywa się z wiersza poleceń z następującym argumentem:

NetCon.exe eee\_off

### 3.4.1 Testowanie

Kolejnym krokiem w przetestowaniu urządzenia jest podłączenie przewodów ethernetowych do odpowiednich portów karty NetFPGA i karty sieciowej. W tym rozdziale zostanie opisana konfiguracja wykorzystująca wszystkie porty karty NetFPGA. Przedstawiona konfiguracja została wykorzystana w rozdziale 4.2 do pomiaru opóźnień wprowadzanych przez układy PHY. Na rysunku 3.31 znajduje się schemat połączeniowy, a na rysunku 3.32 rzeczywiste zdjęcie połączeń.



Rysunek 3.31: Konfiguracja pozwalająca na pomiar opóźnień tunelu.



Rysunek 3.32: Zdjęcie przedstawiające podłączenie w konfiguracji umożliwiającej pomiar opóźnień tunelu. (Na dole karta NetFPGA, nad nią karta sieciowa).

Następnie należy otworzyć cztery okna konsolowe oraz uruchomić program z następującymi argumentami:

NetCon.exe capture 1

NetCon.exe capture 2

NetCon.exe capture 3

NetCon.exe capture 4

Otwarcie strumienia z powyższą komendą powoduje utworzenie się pliku o rozszerzeniu *.pcap* z datą i godziną uruchomienia programu oraz numerem portu. Struktura nazyw plików została pokazana na rysunku 3.33.

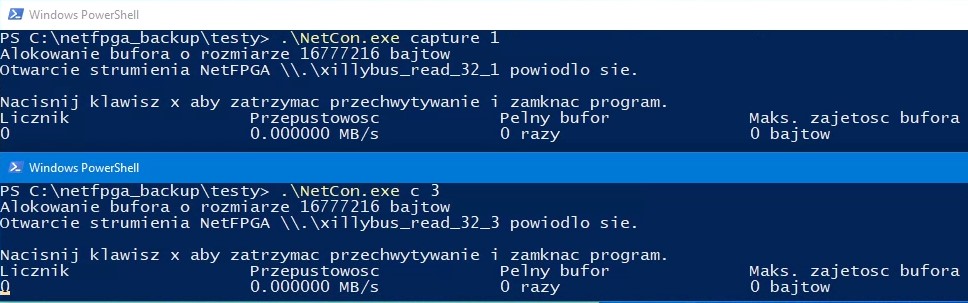


Rysunek 3.33: Struktura nazewnictwa zapisywanych plików.

Istnieje także możliwość zdefiniowania własnej nazwy pliku poprzez dodanie kolejnego argumentu do wywołania programu:

NetCon.exe capture 1 "nazwa\_pliku.pcap"

Zrzut ekranu z programów uruchomionych w trybie przechwytywania został przedstawiony na rysunku 3.34.



Rysunek 3.34: Zrzut ekranu z programów w trybie przechwytywania i zapisywania ruchu sieciowego do pliku.

Nastawy filtrów w bloku filtracyjnym na karcie NetFPGA początkowo są ustawione w trybie blokowania wszystkich ramek. Aby ustawić wszystkie filtry na wszystkich portach w trybie przekazywania całego ruchu sieciowego do użytkownika, należy uruchomić program z następującymi argumentami:

|  |  |  |
| --- | --- | --- |
| NetCon.exe | set | 1 0 |
| NetCon.exe | set | 2 0 |
| NetCon.exe | set | 3 0 |
| NetCon.exe | set | 4 0 |

Dodatkowo wyłączone są wszystkie tunele, oznacza to, że domyślnie odbierane ramki na układach PHY nie są przekazywane do przeciwnego układu PHY. Aby załączyć wszystkie ścieżki, należy uruchomić program z następującym argumentem:

NetCon.exe bridge\_all

Ostatnim krokiem jest wygenerowanie ramek na jednym z portów karty sieciowej. Można to zrobić za pomocą pythonowego skryptu z wykorzystaniem biblioteki Scapy:

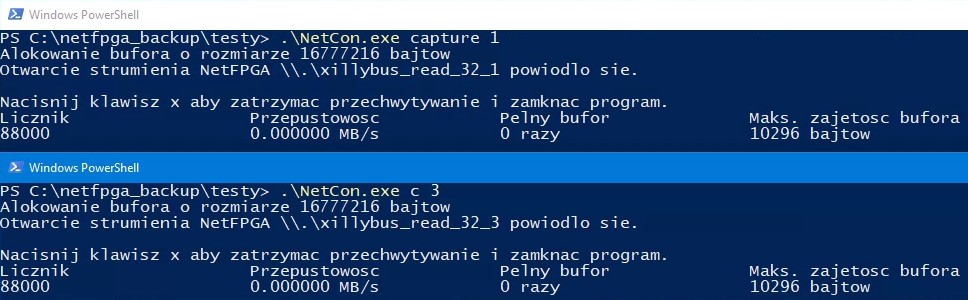
from scapy.all import \* sendp("abcdefghijklmnopqrstuvwxyz", iface=ifaces.

*,→* dev\_from\_pcapname(’\\Device\\NPF\_{DB07413B-64B8

*,→* -4005-A7B3-9F13EAB5210E}’), count=1000)

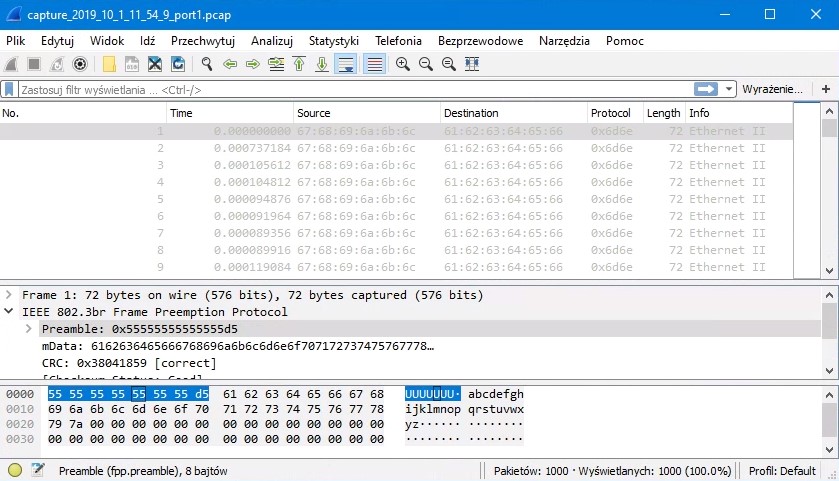
Na rysunku 3.35 zauwazyc mozna, ze wszystkie ramki zostały przechwycone.

Licznik okresla liczbe bajtów odebranych z danego strumienia.



Rysunek 3.35: Zrzut ekranu z programu po wygenerowaniu ramek.

Kolejno plik można otworzyć programem Wireshark oraz zobaczyć, że wszystkie dane są poprawne oraz liczba przechwyconych ramek zgadza się z liczbą wysłanych. Zrzut ekranu z programu został przedstawiony na rysunku 3.36.

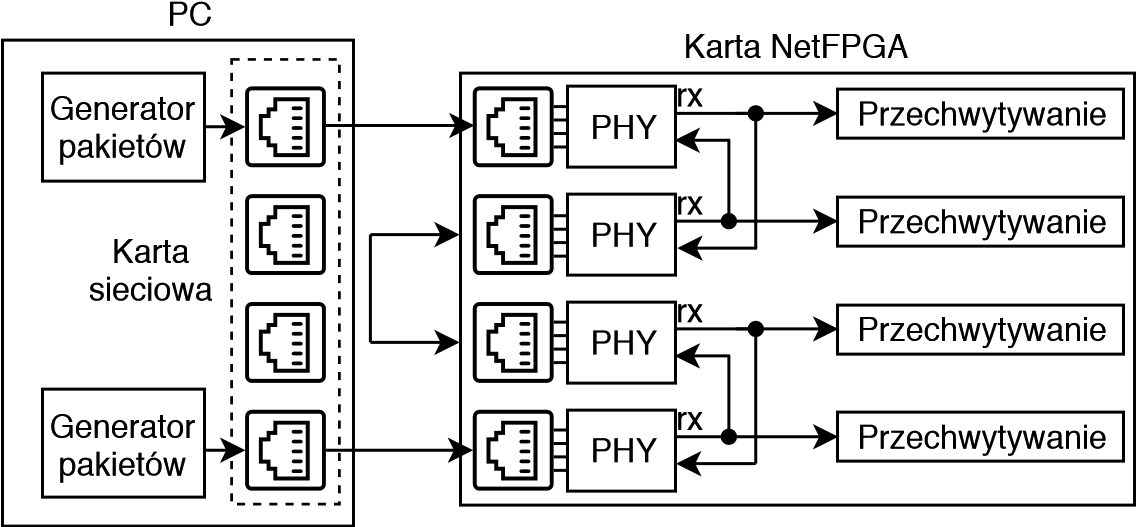


Rysunek 3.36: Zrzut ekranu po otwarciu pliku przechwytywania w programu Wireshark.

### 3.4.2 Wpływ trybu Energy-Efficient Ethernet na opóźnienia

W trakcie pierwszych pomiarów w konfiguracji przedstawionej na rysunku 3.37 okazało się, że układy PHY, które znajdują się na karcie NetFPGA mają domyślnie załączoną opcję Energy-Efficient Ethernet. Powoduje to usypianie się układów PHY, które wprowadza dodatkowe opóźnienia w tor transmisji. Maksymalne zmierzone opóźnienie wynosiło 18*.*9µs, a minimalne 0*.*8µs. Specyfikacja EnergetycznieEfektywnego Ethernetu [10] mówi o wartości 16*.*5µs w przypadku 1000BASE-T, jako najgorszym możliwym czasie wybudzenia układu PHY. Zmierzone opóźnienie jest większe o 2*.*4µs. Może to być spowodowane wewnętrznymi buforami układów. Oczywiście istnieje możliwość wyłączenia opcji ustawiając odpowiedni rejestr w układach PHY przez interfejs MDIO. Po wyłączeniu ustawienia oraz miękkim restarcie modułów PHY wartość opóźnień spadła do 0*.*7µs.

Działanie portu w tym trybie można wykryć poprzez inspekcję wskaźników



Rysunek 3.37: Konfiguracja pozwalająca na pomiar opóźnień.

LED przy portach karty NetFPGA. Jeżeli układ wynegocjował wspomniany tryb, a połączenie zostało ustalone, to wskaźnik LED przy danym porcie będzie migać naprzemienne, będąc 2s w stanie wyłączonym oraz 0*.*4s w stanie załączonym. Przy pomiarach należy zwrócić na to uwagę, ponieważ mogą zaburzyć dokładność wyników.