Autorzy: Mikołaj Chmielecki 252723
Piotr Rodziewicz 252769

Prowadzący: dr inż. Dominik Żelazny

Termin zajęć: piątek 17:05 TP

Niezawodność i diagnostyka układów cyfrowych Etap 1

Temat: System FEC (Forward Error Correction)

1. Założenia:

Napisanie pierwszej wersji symulacji systemu składającej się z:

- generatora ciągów bitów o zadanej długości,
- kodera potrajającego każdy bit naszego sygnału,
- kanału transmisyjnego z losowo pojawiającymi się przekłamaniami bitów,
- dekodera który odczytuje zakodowany sygnał.

Przeprowadzenie analizy systemu z wykorzystaniem przygotowanego przez nas symulatora oraz wykonanie odpowiednich eksperymentów w zależności od długości danego ciągu bitowego i prawdopodobieństwa przekłamania p.

2. Wykonanie:

Program symulacji został wykonany w środowisku Python wykorzystując zasady programowania obiektowego. Każda z funkcji programu wraz z jej metodami i polami została napisana w oddzielnej klasie. W celu łatwego przedstawienia wyników naszych obserwacji użyliśmy biblioteki *matplotlib,* która pozwoliła nam na utworzenie wykresów w naszym programie.

3. Funkcjonalność:

Symulacje możemy wykonać na dwa sposoby poprzez funkcję plot lub funkcję *print_results*.

Funkcja plot wyświetla wykres zależności liczby przekłamanych bitów od prawdopodobieństwa. Wykonuje ona symulację dla podanej długości sygnału dla prawdopodobieństwa w przedziale od 0 do 1. Skoki prawdopodobieństwa dla pojedynczej symulacji wynosi 0.05.

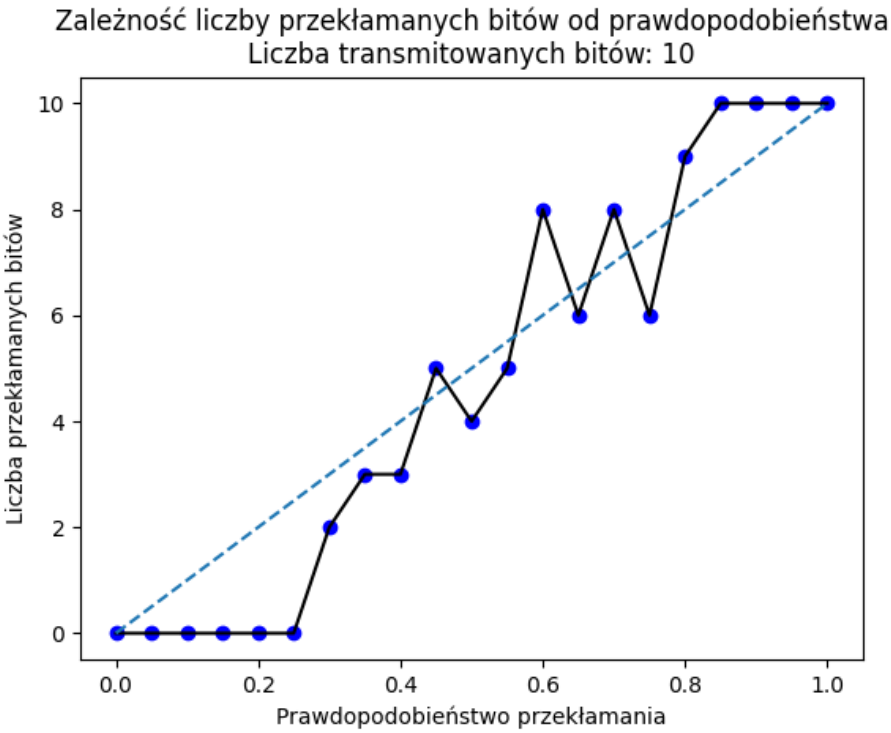
Funkcja print_results wykonuje pojedynczą symulację oraz wyświetla nasz sygnał na każdym kroku naszego symulowanego pasma transmisyjnego: po generowaniu, po kodowaniu, po przejściu przez pasmo transmisyjne oraz po zdekodowaniu. Pod koniec otrzymujemy także informację, ile bitów zostało przekłamanych po przejściu przez nasz kanał transmisyjny.

W tej funkcji podajemy długość sygnału oraz prawdopodobieństwo przekłamania.

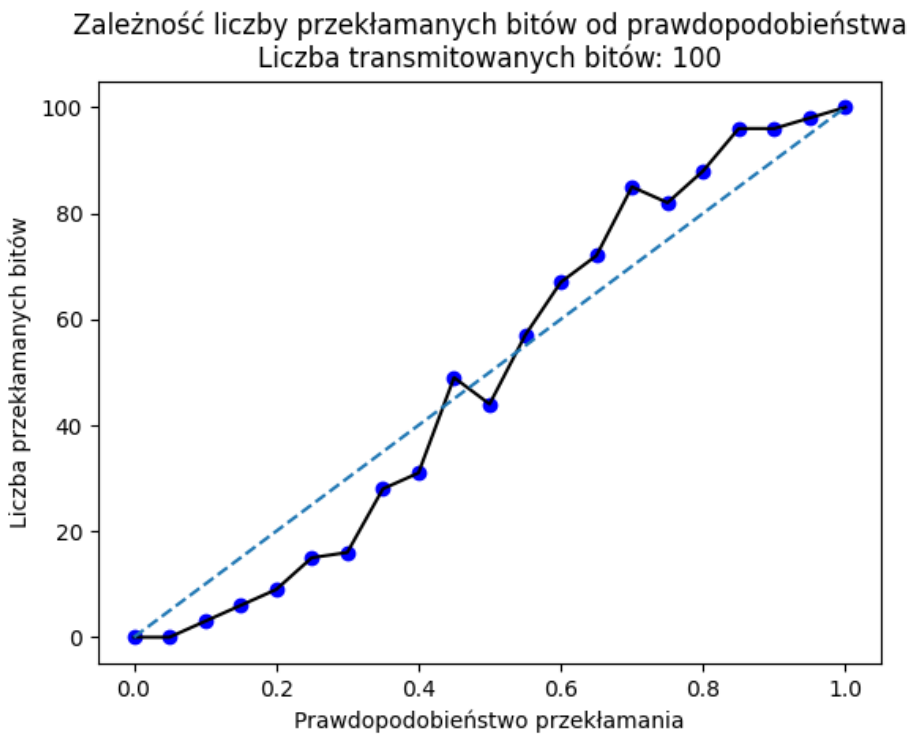
4. Wyniki symulacji

Symulację przeprowadziliśmy dla 6 różnych liczb m przesyłanych bitów – 10, 100, 1000, 10000, 12208 (maksymalna długość ramki Ethernet) oraz 100000 bitów. Dla każdej długości ciągu bitów testowaliśmy różne poziomy prawdopodobieństwa przekłamania. Wyniki przedstawiliśmy na wykresach zależności przekłamanych bitów od prawdopodobieństwa przekłamania, które należy do przedziału [0, 1], ze skokiem o wartości 0,05. Przez liczbę przekłamanych bitów rozumiemy liczbę bitów różniących się w ciągach bitów przez kodowaniem oraz po dekodowaniu.

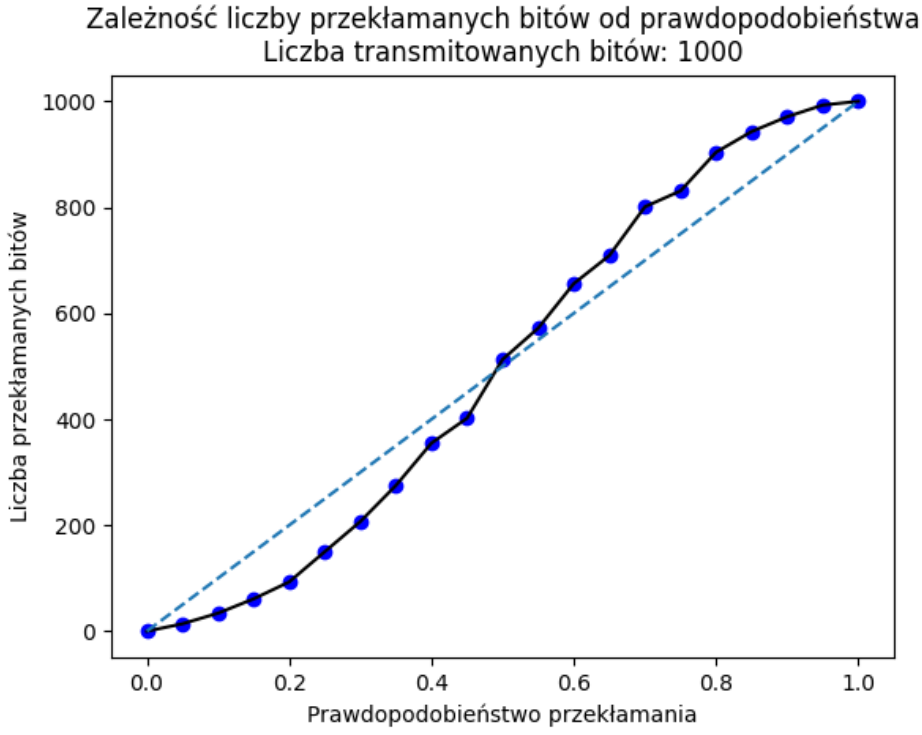
Linia przerywana koloru niebieskiego na wykresach oznacza, teoretyczną liczbę bitów przekłamanych, gdybyśmy przez kanał transmisyjny wysłali niezakodowany sygnał.



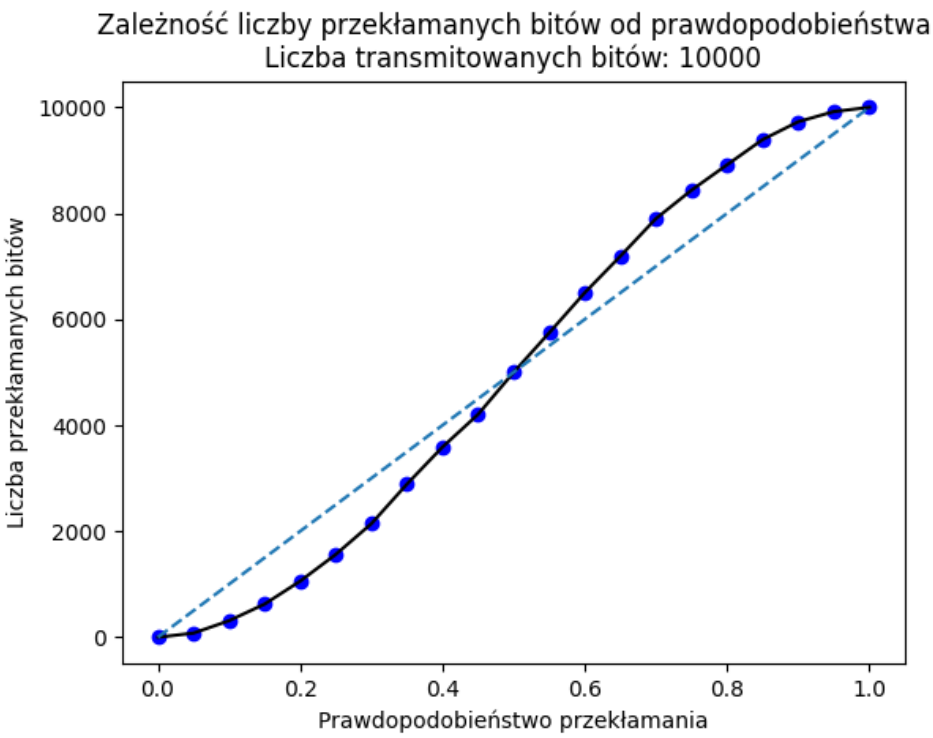
Rysunek 1 Wykres zależności przekłamanych bitów od prawdopodobieństwa przekłamania dla m = 10



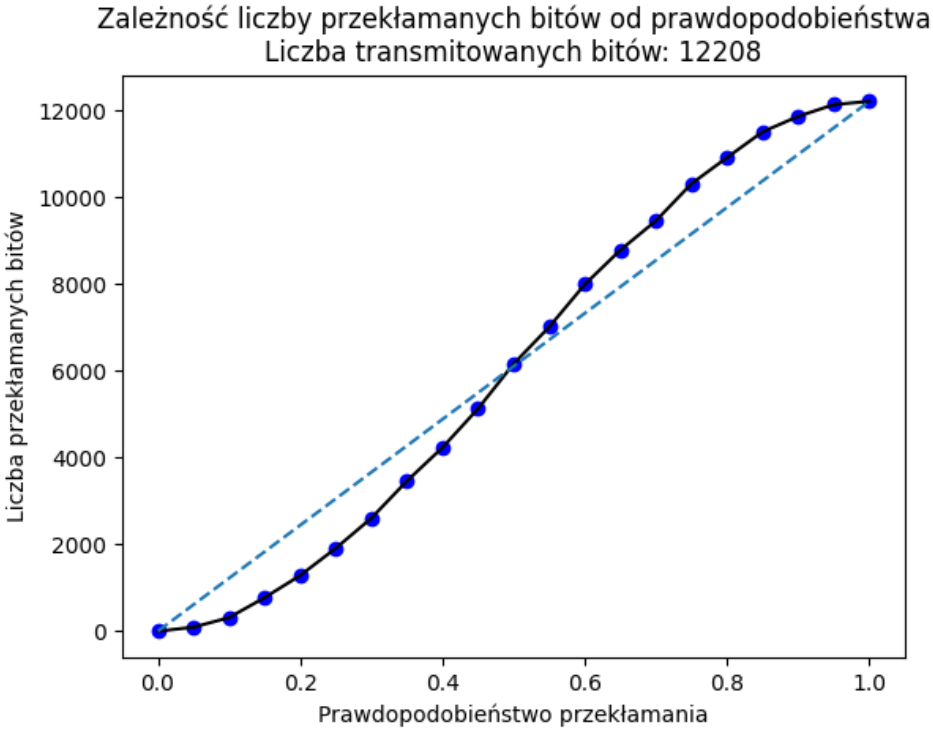
Rysunek 2 Wykres zależności przekłamanych bitów od prawdopodobieństwa przekłamania dla m = 100



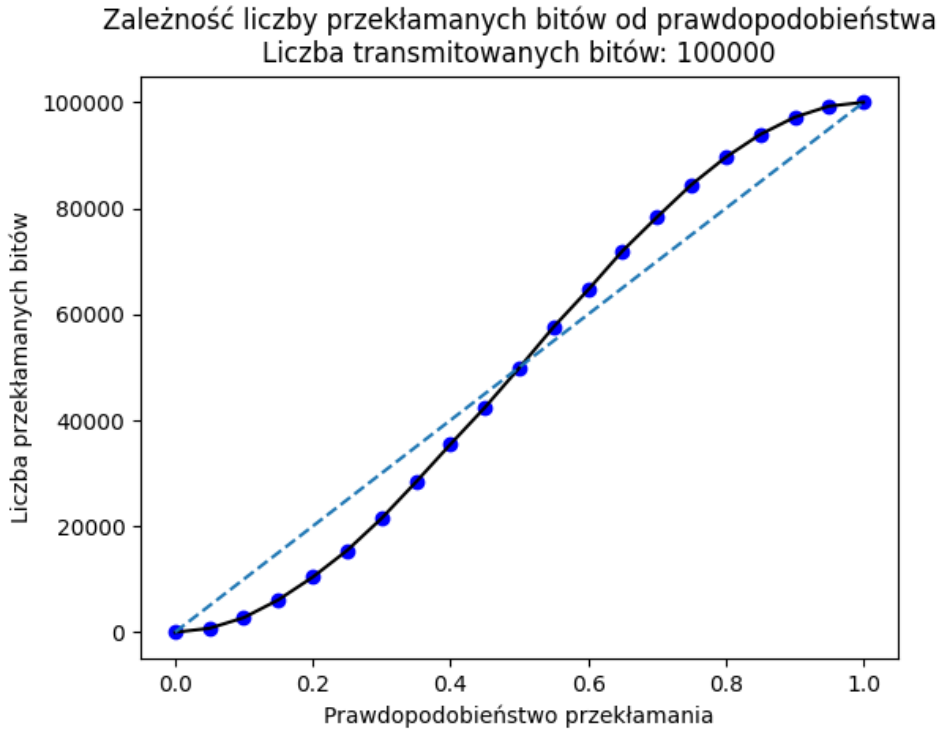
Rysunek 3 Wykres zależności przekłamanych bitów od prawdopodobieństwa przekłamania dla m = 1000



Rysunek 4 Wykres zależności przekłamanych bitów od prawdopodobieństwa przekłamania dla m = 10000



Rysunek 5 Wykres zależności przekłamanych bitów od prawdopodobieństwa przekłamania dla m = 12208



Rysunek 6 Wykres zależności przekłamanych bitów od prawdopodobieństwa przekłamania dla m = 100000

5. Wnioski

Należy zauważyć, że stabilne wyniki otrzymujemy dla $m \geq 10000$, ponieważ wtedy wykresy mają „łagodny" przebieg (bardziej formalnie: możemy drugą pochodną tej zależności uznać za funkcję ciągłą). Dla $m = 10$ losowość odgrywa bardzo dużą rolę i jedyny wniosek jaki można w tym miejscu sformułować to, że dla większego prawdopodobieństwa przekłamania zazwyczaj więcej bitów jest przekłamywanych, co w zasadzie jest oczywiste. Dla coraz większych wartości m, można zauważyć charakterystyczny kształt zależności, który przypomina funkcję $sin()$ w przedziale $[-\pi/2, \pi/2]$. Ponadto, kiedy zwrócimy uwagę na linię przerywaną, która wyznacza teoretyczną wartość przekłamanych bitów w przypadku transmitowania niezakodowanego sygnały, zauważymy jeszcze bardziej interesującą zależność. Otóż, zwracając uwagę na wykresy dla $m \geq 10000$, zauważamy, że dla $p \in (0, 0.5)$ liczba bitów przekłamanych po zastosowaniu kodowania potrajającego jest faktycznie mniejsza niż teoretyczna liczba przekłamywanych bitów bez kodowania. Jest to efekt, który był celem zastosowania kodowania i przesyłania informacji nadmiarowej przez kanał transmisyjny. Dzięki zastosowaniu kodowania dla $p \in (0, 0.5)$ byliśmy w stanie uniknąć części błędów powstających w kanale transmisyjnym. Natomiast dla $p \in (0.5, 1)$, można zauważyć, że ilość błędów po zastosowaniu kodowania jest większa niż teoretyczna liczba błędów bez zastosowania kodowaniu. Innymi słowy stosując kodowanie uzyskaliśmy efekt negatywny, tzn. jeszcze gorszy niż bez stosowania kodowania. Podsumowując, kodowanie sygnału przez potrajanie bitów dla $p \in (0, 0.5)$ poprawia jakość przesyłanego sygnału,. Jednak bardzo duża nadmiarowość, którą to kodowanie wprowadza, i mała

skuteczność sprawia, że należy je uznać za rozwiązanie nieoptymalne. W dalszej części realizacji projektu, skupimy się na kodowaniach mniej trywialnych i jednocześnie znacznie bardziej skutecznych, czyli m. in. na kodowaniu BCH oraz Hamminga.