Zadanie polega na zastosowaniu instrukcji NEON do obliczeń na procesorze ARM oraz oszacowanie szybkości działania programu z użyciem i bez użycia instrukcji NEON.

Uwaga: do wykonania zadania wymagane będą funkcje napisane w poprzednim zadaniu, służące do obliczania liczby cykli procesora.

Część A)

1. Znaleźć w internecie dokument zatytułowany "ARM NEON support in the ARM compiler" i zapoznać się z nim.
2. Napisać prosty program, w którym występuje funkcja suma, która dodaje do siebie wszystkie elementy podanej tabeli. Elementami tablicy powinny być liczby typu int. Rozmiar tablicy można przyjąć na 10000 elementów. Odpowiedni plik Makefile zostanie podany przez prowadzącego zajęcia.
3. Obliczyć ilość cykli potrzebnych do wykonania funkcji suma (należy użyć do tego celu funkcji z poprzedniego zadania).
4. Przeanalizować wpływ następujących trzech opcji kompilatora gcc na szybkość wykonania funkcji:
    a. Kompilacja bez optymalizacji (-O0)
    b. Kompilacja z optymalizacją, bez wektoryzacji (-O3 –fno-tree-vectorize)
    c. Kompilacja z optymalizacją i wektoryzacją (-O3 -mfpu=neon -ftree-vectorize)
5. Skompilować program z trzema powyższymi opcjami do plików assemblera i porównać te trzy pliki, w szczególności zobaczyć jak wygląda główna pętla dodająca elementy tablicy.
6. Napisać taką samą funkcję dodającą elementy tablicy, ale za pomocą tzw. instrukcji „ARM intrinsics" (należy skorzystać z przykładu podanego na stronie 7 w dokumencie "ARM NEON support in the ARM compiler".
7. Skompilować program z optymalizacją i wektoryzacją, uruchomić go i porównać szybkość wykonania sumy z wynikami otrzymanymi w punkcie 4.

Część B)

1. Powtórzyć punkty 2-7 z części A dla filtru FIR (na podstawie informacji podanych na stronach 13-17 w dokumencie "ARM NEON support in the ARM compiler").