

MARIA KOSICKA
MICHAŁ MORAWSKI

METODY IMPLEMENTACJI SPLOTU NA TABLICY SYSTOLICZNEJ KOMPUTERA RÓWNOLEGŁEGO SYSTOLA 1024^{*)}

STRESZCZENIE *W artykule przedstawiono i porównano dwie metody implementacji dyskretnego spłotu na tablicy systolicznej komputera równoległego SYSTOLA 1024: przy wykorzystaniu liniowej tablicy systolicznej (jednego wiersza tablicy ortogonalnej) oraz przy wykorzystaniu kwadratowej tablicy ortogonalnej. Stwierdzono, że obie metody są równie dokładne. Ponadto porównano wyniki obliczeń z otrzymanymi za pomocą programu Matlab.*

1. HISTORIA I POJĘCIE SPLOTU

Wiek XIX w matematyce to czas badań, porządkowania i krystalizowania się pojęć i metod matematyki współczesnej tak, jak rozumiemy je dzisiaj [1, 10]. W owym czasie pojawiło się również pojęcie *spłotu* będące operacją całkową

^{*)} Artykuł jest rozszerzoną i uzupełnioną postacią referatu prezentowanego na XXIV Konferencji IC-SPETO'2001 Gliwice-Ustroń 23-26.05.2001.

mgr inż. MARIA KOSICKA, mgr inż. MICHAŁ MORAWSKI
Zakład Badań Podstawowych Elektrotechniki
Instytut Elektrotechniki

wykonywaną na dwóch funkcjach. Pojęcie to znane było już na początku XIX wieku, ale zbadanie jego właściwości jest głównie zasługą włoskiego matematyka Vita Volterry (1860...1940), który zajmował się w sposób ogólny badaniem złożenia dwóch funkcji dwóch zmiennych, natomiast podstawowe znaczenie *splotu* zrozumiał dopiero matematyk niemiecki Hermann Weyl (1855...1955).

Splot (ang. *convolution*) jest matematyczną operacją całkową wykonywaną na dwóch funkcjach tej samej zmiennej, np.: $x(t)$ i $h(t)$:

$$y(t) = \int_{-\infty}^{\infty} x(\tau) h(t-\tau) d\tau = h(t) * x(t), \quad (1)$$

gdzie:

$y(t)$ – funkcja będąca wynikiem *splotu*,

τ – wielkość pomocnicza, znak $*$ oznacza *splot*.

Operacja *splotu* ma swój odpowiednik dyskretny zapisywany w postaci:

$$y(n) = \sum_{k=-\infty}^{\infty} x(k) h(n-k) \quad (2)$$

gdzie:

n – czas dyskretny,

k – wielkość pomocnicza.

Splot [9] ma ciekawą właściwość wykorzystywaną przy analizie układów automatycznej regulacji oraz przy przetwarzaniu sygnałów; transformata (Fouriera lub Laplace'a) *splotu* jest równa iloczynowi transformat splatanych funkcji, przy czym zależność ta jest prawdziwa zarówno dla sygnałów ciągłych jak i dyskretnych, np. dla transformaty Fouriera funkcji ciągłych:

$$F[x(t) * h(t)] = F[x(t)] \cdot F[h(t)] \quad (3)$$

Transformata Fouriera funkcji czasu jest funkcją częstotliwości. Można zatem powiedzieć, że – zgodnie ze wzorem (3) – *splot w dziedzinie czasu jest równoważny mnożeniu w dziedzinie częstotliwości* lub też ogólnie: *splot w jednej dziedzinie jest równoważny mnożeniu w drugiej dziedzinie* [9].

Operacja *splotu* znajduje zastosowanie przy dokonywaniu filtracji sygnału poprzez splatanie sygnału oryginalnego będącego pewnym ciągiem liczb z drugim ciągiem liczb zwanym współczynnikami filtru, wagami lub odpowiedzią impulsową. W praktyce mamy do czynienia z dwoma skończonymi ciągami określonymi w dziedzinie czasu. Wzór (2) ulega zatem modyfikacji i przybiera postać:

$$y(n) = \sum_{k=0}^{Q-1} x(k) h(n-k), \quad (4)$$

gdzie:

$\{h_n\} = h(n)$, $n = 0 \dots P-1$ – ciąg współczynników filtru, przy czym $h(n)$,
dla $\forall_{n \notin \{0..P\}} h(n) = 0$,

$\{x_n\} = x(n)$, $n = 0 \dots Q-1$ – filtrowany ciąg wejściowy,

$\{y_n\} = y(n)$, $n = 0 \dots N-1$ – ciąg wyjściowy.

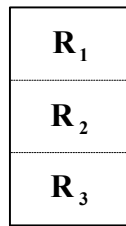
W ogólnym przypadku $N = P + Q - 1$ [9], gdzie P , Q , N – długości odpowiednio: ciągu współczynników filtru, filtrowanego ciągu wejściowego, ciągu wyjściowego.

2. OBLICZANIE SPLOTU ZA POMOCĄ TABLICY SYSTOLICZNEJ

W pracach [4, 7] opisano budowę, sposób działania i zastosowanie komputera równoległego *SYSTOLA 1024*. Jako jedno z zastosowań tablicy systolicznej wymieniono obliczanie *splotu*. Poniżej pokazano dwa sposoby wykonania omawianej operacji.

2.1. Obliczanie splotu przy wykorzystaniu liniowej tablicy systolicznej (jednego wiersza tablicy ortogonalnej)

W opisywanej metodzie korzysta się wprost ze wzoru (4) będącego sumą iloczynów. W procesie obliczania bierze udział tylko pierwszy rząd procesorów tablicy systolicznej, a w każdym procesorze 6 rejestrów (dla liczb stałoprzecin-

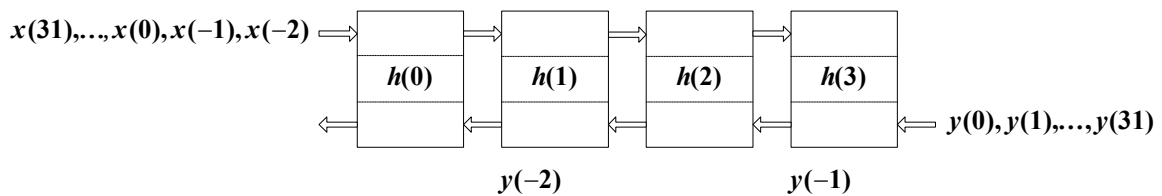


Rys.1. Umowne ustawienie rejestrów w procesorze.

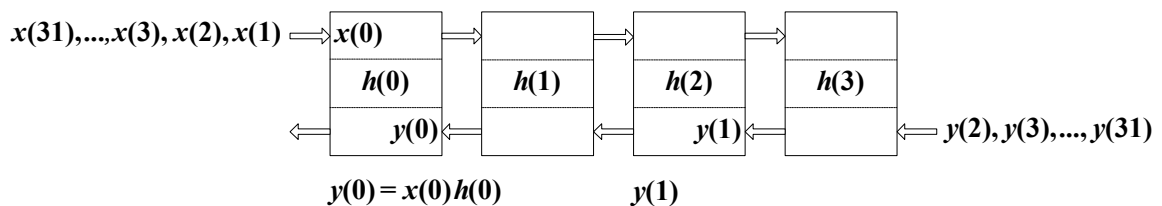
kowych) lub 3 (dla liczb całkowitych). Dla uproszczenia przy omawianiu metody uwzględniono tylko 3 rejestry zwane umownie: R_1 , R_2 i R_3 (rys.1). W demonstrowanym przykładzie założono $P = 4$, $Q = 32$, ograniczono się do $N = 32$.

W poniższym opisie stosowane są „geograficzne” opisy kierunków typowe dla opisu prostokątnej tablicy systolicznej zgodne z typową orientacją mapy.

W pierwszym etapie obliczeń wprowadza się do tablicy systolicznej od strony *zachodniej* elementy ciągu $\{h_n\}$ rozpoczynając od ostatniego (rys.2). Dane te są zapisywane w rejestrach R_2 pierwszych P procesorów i pozostają tam do końca obliczeń.



Rys.2. Wprowadzanie elementów ciągu $\{x_n\}$ po wprowadzeniu elementów ciągu $\{h_n\}$. Wartości $x(-2) = x(-1) = 0$. Elementy $y(-2)$ i $y(-1)$ są odrzucane.



Rys.3. Pierwszy takt – wprowadzenie pierwszego elementu ciągu wejściowego $\{x_n\}$: $x(0)$, obliczanie pierwszego elementu ciągu wyjściowego $\{y_n\}$: $y(0)$ w pierwszym procesorze.

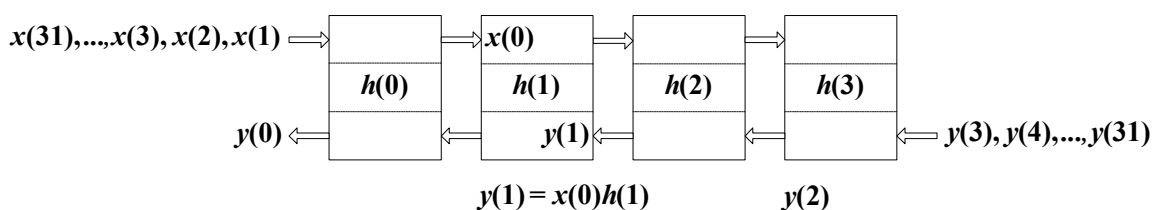
Aby obliczyć pierwszy element ciągu $\{y_n\}$ (rys.3, 4) należy:

- wprowadzić z *zachodu* wartość $x(0)$ do rejestru R_1 pierwszego procesora,

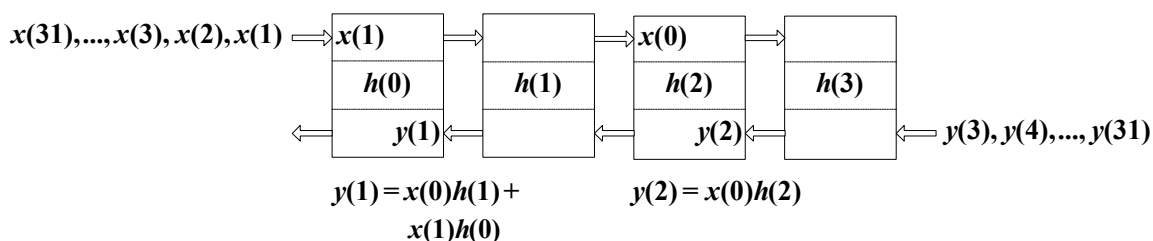
- załadować zawartość rejestru R_1 pierwszego procesora jako pierwszy czynnik mnożenia, pomnożyć przez zawartość rejestru R_2 i zmagazynować w rejestrze R_3 pierwszego procesora,
- wyprowadzić wynik $y(0)$ *na zachód*.

Obliczanie drugiego elementu ciągu $\{y_n\}$ (rys.4, 5, 6) przebiega następująco:

- wprowadzić $x(1)$ z zachodu do rejestru R_1 pierwszego procesora, a wartość $x(0)$ przesunąć *na wschód* z rejestru R_1 drugiego procesora do rejestru R_1 trzeciego procesora.
- załadować zawartość rejestru R_1 pierwszego procesora jako pierwszy czynnik mnożenia, pomnożyć przez zawartość rejestru R_2 tego samego procesora i dodać do zawartości rejestru R_3 ,
- wyprowadzić wynik $y(1)$ *na zachód*, jednocześnie w procesorze trzecim rozpoczyna się „tworzenie” trzeciego elementu ciągu $\{y_n\}$: $y(2)$.

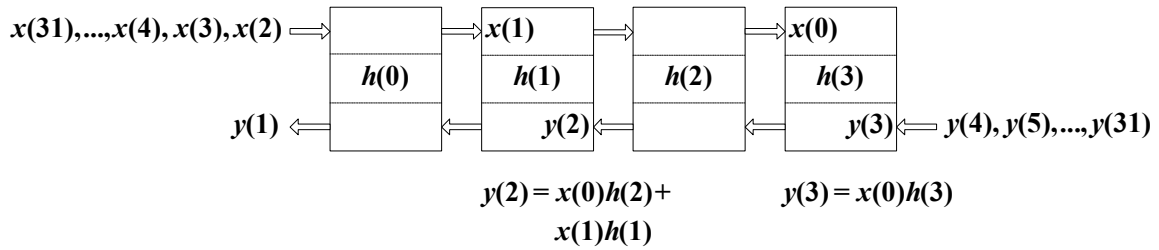


Rys.4 Drugi takt – przesunięcie $x(0)$ na wschód, wyprowadzenie $y(0)$ *na zachód*, początek obliczania drugiego elementu ciągu $\{y_n\}$: $y(1)$ w drugim procesorze.

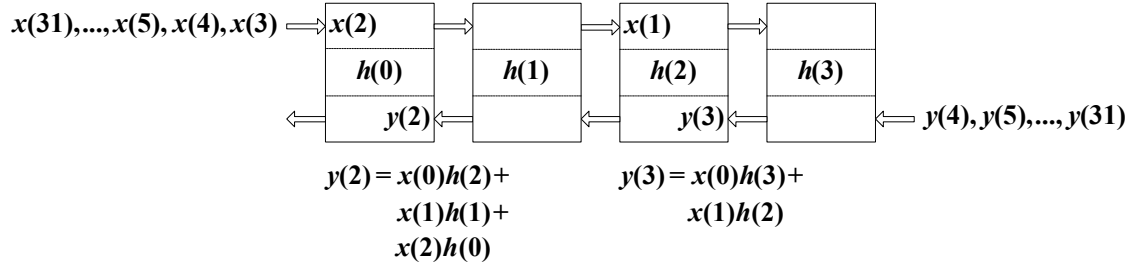


Rys.5. Trzeci takt – przesunięcie $x(0)$ *na wschód*, wprowadzenie $x(1)$ do pierwszego procesora, przesunięcie $y(1)$ *na zachód*, sumowanie w pierwszym procesorze, początek obliczania trzeciego elementu ciągu $\{y_n\}$: $y(2)$ w trzecim procesorze.

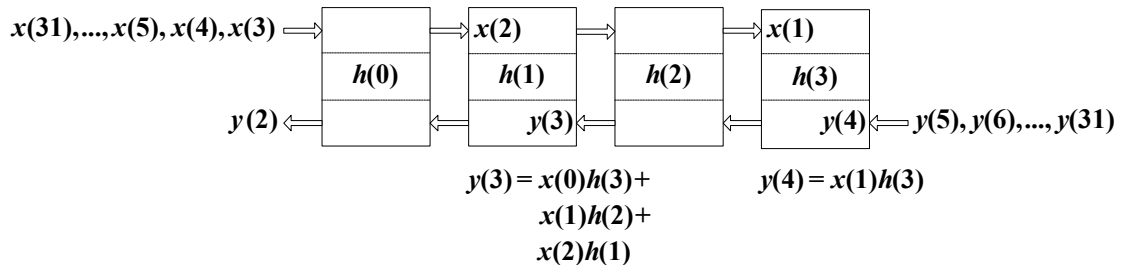
Każdy kolejny element ciągu wyjściowego $\{y_n\}$ jest obliczany w ten sam sposób (rys. 6, 7, 8). Rośnie jedynie liczba operacji. Na rysunku 2...8 uwidoczono cztery z 32 procesorów pierwszego rzędu tablicy systolicznej. Należy zauważyć, że podczas wykonywania obliczeń zarówno elementy ciągu $\{x_n\}$, jak i elementy ciągu $\{y_n\}$ znajdują się w co drugim procesorze.



Rys.6. Czwarty takt – przesunięcie $x(0)$ i $x(1)$ na wschód, wyprowadzenie $y(1)$, przesunięcie $y(2)$ na zachód, sumowanie w drugim procesorze, początek obliczania czwartego elementu ciągu $\{y_n\}$: $y(3)$ w czwartym procesorze.



Rys.7. Piąty takt – przesunięcie $x(1)$ na wschód, wprowadzenie $x(2)$ do pierwszego procesora, przesunięcie $y(2)$ i $y(3)$ na zachód, sumowanie w pierwszym i trzecim procesorze.



Rys.8. Szósty takt – wyprowadzenie $y(2)$, przesunięcie $x(1)$ i $x(2)$ na wschód, przesunięcie $y(3)$ i $y(4)$ na zachód, sumowanie w drugim procesorze, początek obliczania piątego elementu ciągu $\{y_n\}$: $y(4)$ w czwartym procesorze.

2.2. Obliczanie splotu przy wykorzystaniu kwadratowej, ortogonalnej tablicy systolicznej

Splot jest operacją symetryczną, zatem wzór (4) można przedstawić w równorzędnej postaci:

$$y(n) = \sum_{k=0}^{P-1} h(k) x(n-k) \quad (5)$$

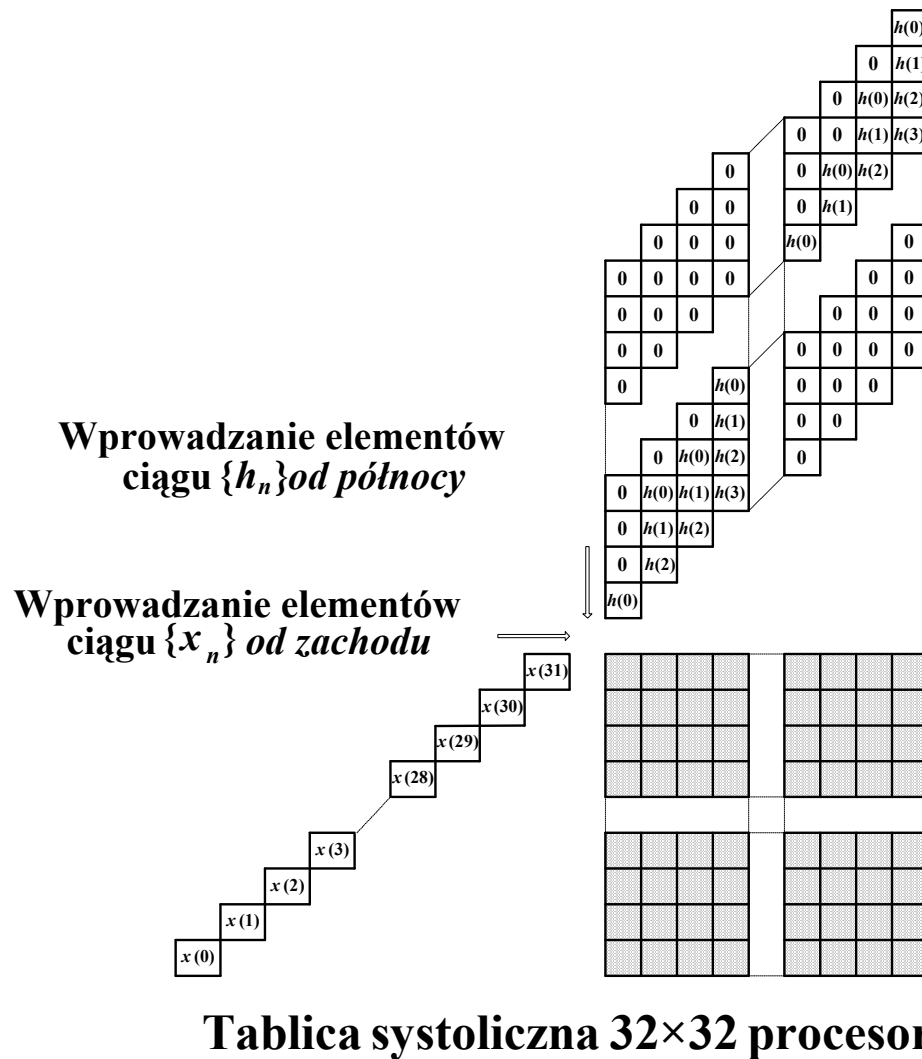
i sprowadzić obliczenia do mnożenia trójkątnej macierzy Toeplitza przez wektor [8]. Operacja mnożenia macierzy przez wektor została dokładnie opisana w pracy [5].

Dla prezentowanego przykładu wzór (5) przyjmie postać:

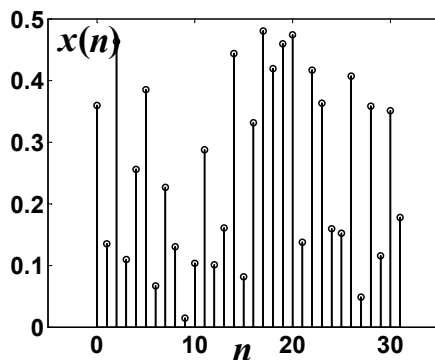
$$\begin{bmatrix} y(0) \\ y(1) \\ y(2) \\ y(3) \\ y(4) \\ \vdots \\ y(31) \end{bmatrix} = \begin{bmatrix} h(0) & 0 & 0 & 0 & 0 & \dots & 0 & 0 & 0 \\ h(1) & h(0) & 0 & 0 & 0 & \dots & 0 & 0 & 0 \\ h(2) & h(1) & h(0) & 0 & 0 & \dots & 0 & 0 & 0 \\ h(3) & h(2) & h(1) & h(0) & 0 & \dots & 0 & 0 & 0 \\ 0 & h(3) & h(2) & h(1) & h(0) & \dots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & h(3) & h(2) & h(1) & h(0) \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \\ x(4) \\ \vdots \\ x(31) \end{bmatrix}, \quad (6)$$

gdzie trójkątna macierz Toeplitza została utworzona z ciągu $\{h_n\}$. Oczywiście można również utworzyć macierz Toeplitza z ciągu $\{x_n\}$ i mnożyć ją przez wektor $h(n)$.

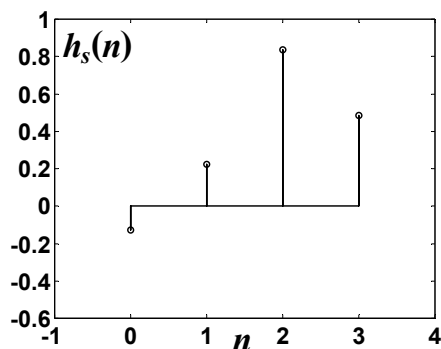
Wprowadzenie danych do tablicy nastąpi w sposób pokazany na rys.9 [4, 5].



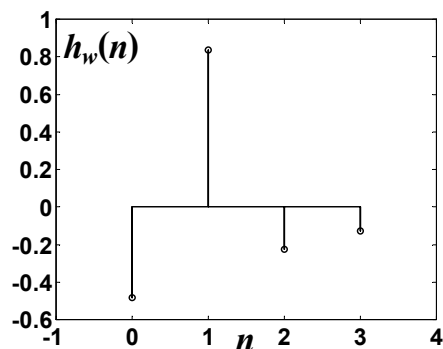
Rys. 9 Wprowadzanie danych do tablicy systolicznej przy obliczaniu *spłotu* metodą mnożenia trójkątnej macierzy Toeplitza przez wektor.



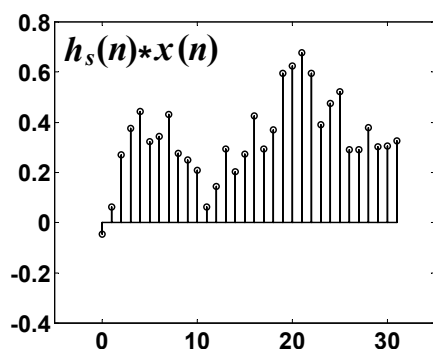
Rys. 10 Filtrowany ciąg dyskretny $\{x_n\}$ o przebiegu losowym.



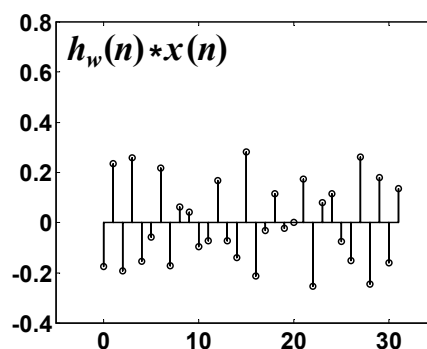
Rys.11. Czteroelementowy dolno-przepustowy filtr skalujący Daubechies $\{h_{ns}\}$ stosowany do dekompozycji sygnałów.



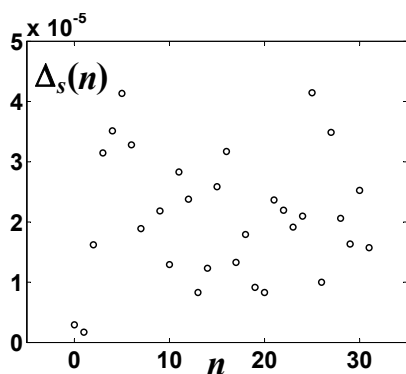
Rys.12. Czteroelementowy górno-przepustowy filtr falkowy Daubechies $\{h_{nw}\}$ stosowany do dekompozycji sygnałów.



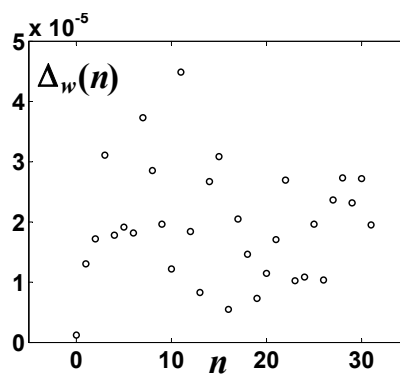
Rys.13. Splot $\{y_n\} = \{h_{ns}\} * \{x_n\}$.



Rys.14. Splot $\{y_n\} = \{h_{nw}\} * \{x_n\}$.



Rys. 15.



Rys. 16.

Różnice w wynikach obliczeń *splotu* przeprowadzonych na tablicy systolicznej oraz za pomocą programu *Matlab*.

Na rysunkach 11...16 pokazano przykładowe wyniki obliczeń *splotu* wykonane na tablicy systolicznej dla sygnału o przebiegu losowym (rys.10). Jako filtry przyjęto dwa filtry Daubechies [2, 3, 6]: filtr dolnoprzepustowy-skalujący (rys.11) i filtr górnoprzepustowy-falkowy (rys.12) o parametrach:

$$h_{ns} = [-0,1294 \ 0,2241 \ 0,8365 \ 0,4830],$$

$$h_{nw} = [-0,4830 \ 0,8365 \ -0,2241 \ -0,1294]$$

obliczonych za pomocą funkcji `wfilters` programu *Matlab/Toolbox/Wavelet*. Operacja *splotu* pozwala na znalezienie współczynników rozkładu sygnału na odpowiednie funkcje skalujące (rys.13) i falki Daubechies (rys.14) [1, 3, 6].

Jako wyniki wzorcowe przyjęto obliczenia *splotu* wykonane za pomocą funkcji `conv` programu *Matlab*. Stwierdzono, że różnice w wynikach obliczeń (rys.15, 16) wykonanych na tablicy systolicznej i za pomocą programu *Matlab* są pomijalnie małe i nie przekraczają 0,05 % tych ostatnich wartości.

3. WNIOSKI

Porównując obliczenia wykonane na tablicy systolicznej za pomocą obu metod stwierdzono, że dają one tę samą dokładność. Wydaje się jednak, że metoda druga jest bardziej przejrzysta.

Dla prezentowanego przykładu czas obliczeń pierwszej metody wynosi 1765 pulsacji, podczas gdy czas obliczeń drugiej tylko 495 pulsacji. Trzeba sobie jednak zdawać sprawę z faktu, że ze wzrostem długości ciągu $\{x_n\}$ czas obliczeń pierwszej metody rośnie liniowo, podczas gdy w przypadku drugiej metody – z kwadratem liczby elementów ciągu. Ponadto druga metoda wymaga od komputera – gospodarza utworzenia z wektora odpowiedniej macierzy.

Warto zwrócić uwagę na to, że jeżeli rozwiązywany problem wymaga obliczania wielu *splotów* jednocześnie, to pierwsza zaprezentowana w niniejszym artykule metoda pozwala, przy wykorzystaniu tablicy systolicznej *SYSTOLA 1024*, na obliczenie do 32 *splotów* jednocześnie **przy dokładnie takiej samej liczbie pulsacji**. W takim przypadku, nawet dla małej liczby elementów ciągu $\{x_n\}$ metoda pierwsza jest znacząco bardziej efektywna.

Przedstawione metody nie są jedynymi metodami obliczania *splotu* na tablicy systolicznej, jednakże wydaje się, że mają one najprostszą realizację na komputerze *SYSTOLA 1024*, tzn. są najlepiej dostosowane do architektury tego

komputera. W pracy [8] pokazano inne klasyczne metody obliczania *splotu* i każda z nich może być zaimplementowana. Jednakże implementacja metod, w których przesuwane są zarówno elementy ciągu $\{x_n\}$ jak i $\{h_n\}$, ze względu na konstrukcję tej konkretnej tablicy nie będą efektywne.

Splot rzadko obliczany jest jako samodzielna operacja, zwykle stanowi część innego problemu. Dobór konkretnego algorytmu powinien być zatem dokonywany z uwzględnieniem postaci danych w tym problemie występujących oraz ich liczby.

LITERATURA

1. Bourbaki N.: Elementy historii matematyki. PWN, Warszawa, 1980.
2. Białasiewicz J.T.: Falki i aproksymacje. WNT, Warszawa, 2000.
3. Burrus C.S., Gopinath R.A., Guo H.: Introduction to wavelets and wavelets transform. Prentice Hall, New Jersey, 1998.
4. Kosicka M.: Komputer równoległy ISATEC *SYSTOLA 1024* – budowa i zastosowanie. Prace Instytutu Elektrotechniki, z.198, 1998.
5. Kosicka M.: Obliczanie transformaty falkowej przy pomocy komputera równoległego *SYSTOLA 1024*. Prace Instytutu Elektrotechniki, z.202, 1999.
6. Kosicka M.: Falkowa dekompozycja sygnału za pomocą komputera równoległego *SYSTOLA 1024*. Prace Instytutu Elektrotechniki, z.206, 2000.
7. Lipowska-Nadolska E., Morawski M.: Tablice systoliczne. Problemy wybrane. Akademicka Oficyna Wydawnicza PLJ, Warszawa, 1999.
8. Lipowska-Nadolska E.: Ujęcie systoliczne form macierzowych w przetwarzaniu sygnałów. Prace Instytutu Elektrotechniki, z.166, 1991.
9. Lyons R.G.: Wprowadzenie do cyfrowego przetwarzania sygnałów. WKiŁ, Warszawa, 1999.
10. Sękowski T.: Człowiek i matematyka. Polska Oficyna Wydawnicza BGW, Warszawa, 1996.

Rękopis dostarczono, dnia

Opiniował: prof. dr hab. inż. Krystyn Pawluk

IMPLEMENTATION OF THE DIGITAL CONVOLUTION
USING THE SYSTOLIC PARALLEL
COMPUTER SYSTOLA 1024

M. KOSICKA, M. MORAWSKI

SUMMARY *The paper presents and compares two methods of the implementation of discrete convolution using the systolic parallel computer SYSTOLA 1024. Both linear array (one row of the orthogonal array) and orthogonal array approach is presented. Implementation of these methods precisely described. Either precision or the time of execution are discussed and compared to the results obtained when using a Matlab program.*