Alexandr ŢARIOV, Galina ŢARIOVA
POLITECHNIKA SZCZECIŃSKA, WYDZIAŁ INFORMATYKI

Struktura algorytmiczna jednostki procesorowej do realizacji operacji splotu liniowego

Dr hab. inż. Alexandr ŢARIOV

Ukończył studia na Wydziale Automatyki i Urządzeń Obliczeniowych Uniwersytetu Miernictwa w Sewastopolu, obronił pracę doktorską w 1984 r., habilitacyjną - w 2001 r. Jest profesorem w Instytucie Architektury Komputerów i Telekomunikacji na Wydziale Informatyki Politechniki Szczecińskiej. Jego zainteresowania naukowe to algorytmy cyfrowego przetwarzania oraz transmisji sygnałów, sprzętowe wspomaganie oraz zrównoleglenie obliczeń.

e-mail: atariov@wi.ps.pl

Dr Galina ŢARIOVA

Ukończyła studia na Wydziale Matematyki i Cybernetyki Mołdawskiego Uniwersytetu Państwowego w Kiszyniowie w 1978 r. Obroniła pracę doktorską w 2007 r. Jej zainteresowania naukowe są związane z różnymi aspektami aplikacyjnymi matematyki oraz informatyki teoretycznej i stosowanej: analizą falkową, metodami numerycznymi, matematyką dyskretną, algorytmami cyfrowego przetwarzania sygnałów.

e-mail: gtariova@wi.ps.pl

Streszczenie

W pracy została przedstawiona koncepcja organizacji struktury jednostki obliczeniowej dla realizacji operacji splotu liniowego ze zredukowaną liczbą mnożeń (lub układów mnożących w przypadku implementacji sprzętowej). Pozwala to zmniejszyć nakłady obliczeniowe, zapotrzebowanie na zasoby sprzętowe oraz stworzyć dogodne warunki do efektywnej realizacji operacji splotu liniowego w układzie reprogramowalnym.

Słowa kluczowe: procesory DSP, splot liniowy, szybkie algorytmy wyznaczania splotu liniowego.

Algorithmic structure of processing unit for linear convolution operation implementation

Abstract

In work the approach to the rational organization of algorithmic structure of the processor unit for realization of basic operation of linear convolution with the reduced number of multiplication (or multipliers – in hardware implementation case) is presented. This approach allows to lower hardware expenses and creates favorable conditions for effective convolution realization in the reprogrammable platform.

Keywords: DSP processors, linear conwolution, fast linear convolution algorithms.

1. Wstęp

Splot liniowy (Linear Convolution – LC) jest operacją dość często wykorzystywaną w algorytmach cyfrowego przetwarzania sygnałów, najczęściej tam, gdzie mamy do czynienia z filtracją FIR [1-4].

W przypadku operacji bazowej splotu liniowego mamy do czynienia z dwoma wektorami danych (sygnałami cyfrowymi) $\mathbf{X}_{N\times 1}=[x_0,x_1,...,x_{N-1}]^{\mathrm{T}}$ i $\mathbf{H}_{N\times 1}=[h_0,h_1,...,x_{N-1}]^{\mathrm{T}}$ oba o rozmiarze N elementów (próbek), pry czym operacja ta w postaci macierzowej jest opisana w sposób następujący:

$$\mathbf{Y}_{(2N-1)\times 1}=\mathbf{H}_{(2N-1)\times N}\cdot\mathbf{X}_{N\times 1}, \tag{1}$$

gdzie $\mathbf{Y}_{(2N-1)\times 1}=[y_0,y_1,...,y_{2N-2}]^{\mathrm{T}}$ jest wektorem wyników, zaś macierz $\mathbf{H}_{(2N-1)\times N}=\blacksquare_{i=0}^{N-1}[\mathbf{I}_{(2N-1)}^{(i\rightarrow)}(\mathbf{H}_{N\times 1}\blacksquare\mathbf{0}_{(N+1)\times 1})]$ - macierzą, której komponentami są w odpowiedni sposób ulokowane elementy wektora $\mathbf{H}_{N\times 1}$.

Znaczenie wykorzystanych tu i w dalszej części artykułu symboli przedstawiono poniżej:

$\mathbf{I}_N^{(\alpha\rightarrow)}$ - macierz jednostkowa o wymiarze określonym za pomocą dolnego indeksu, natomiast indeks górny, jeżeli jest, wskazuje liczbę pozycji cyklicznego przesunięcia wierszy macierzy jednostkowej w kierunku wskaźnika [10]; $\oplus$ - symbol sumy prostej (tensorowej) dwóch macierzy [6]; $\blacksquare$, $\blacksquare\blacksquare$ - symbole odpowiednio pionowej oraz poziomej konkatenacji dwóch lub więcej macierzy [11]; $\mathbf{0}_{M\times N}$ - macierz zerowa o rozmiarze zdefiniowanym przez indeks dolny.

Realizacja procedury (1) wymaga N^2 operacji mnożenia oraz $(N-1)^2$ operacji dodawania. Wykładniczo rosnąca liczba mnożeń została czynnikiem stymulującym do poszukiwań efektywniejszych rozwiązań algorytmicznych. Powstało wiele algorytmów, których celem jest minimalizacja tych operacji [1]. Najbardziej znanym podejściem do efektywnego wyznaczania splotu liniowego dwóch N-elementowych wektorów jest sprowadzenie tego zadania do wyznaczenia (2N-1)-elementowego splotu cyklicznego (kołowego). Zazwyczaj taka operacja może być efektywnie zrealizowana za pomocą szybkiej transformaty Fouriera (Fast Fourier Transform - FFT), transformat liczbowych Fermata (Fermat number transform -FNT) lub Mersene'a (Mersene number transform - MNT) oraz innych dobrze znanych oraz powszechnie stosowanych „szybkich" algorytmów liczenia splotu kołowego. Natomiast do liczenia bezpośrednie splotu liniowego opracowano znacznie mniej algorytmów. Do rozwiązań algorytmicznych tego typu należą m.in.: algorytm Winograda, który został opracowany na podstawie wykorzystania chińskiego twierdzenia o resztach, algorytm Tooma-Cooka, który bazuje na interpolacji Lagrange'a, algorytm Agarwala-Cooleya [2-9]. Dla wyznaczania splotów długich ciągów danych opracowane są metody „Overlap-save" oraz „Overlap-add", które jednak, też wykorzystują algorytm FFT. Okazuje się jednak, iż wymienione rozwiązania nie wyczerpują wszystkich możliwości, które mogą zostać zrealizowane na drodze minimalizacji liczby operacji arytmetycznych lub bloków mnożących w przypadku realizacji sprzętowej. Właśnie w myśl o dalszym usprawnieniu procesu obliczeniowego wyznaczania splotu liniowego zostało opracowano rozwiązanie, które przedstawione w niniejszej pracy.

2. Synteza struktury algorytmicznej jednostki procesorowej do realizacji bazowej operacji splotu liniowego ze zredukowaną liczbą bloków mnożących

W celu syntezy struktury algorytmicznej specjolizowanej jednostki procesorowej realizującej operację bazową splotu liniowego wprowadźmy kilku konstrukcji macierzowych:

- macierz rozszerzenia wektora danych wejściowych

$$\mathbf{P}_{(2N-1)\times N} = (\mathbf{I}_{\frac{N-1}{2}} \| \mathbf{0}_{\frac{N-1}{2}+1}) \overline{\overline{\mathbf{I}}}_N \overline{\overline{(\mathbf{0}_{\frac{N-1}{2}+1} \| \mathbf{I}_{\frac{N-1}{2}})}};$$

- blokowo diagonalną macierz współczynników wagowych, zbudowaną na podstawie elementów wektora $\mathbf{H}_{L\times 1}$

$$\mathbf{H}_{2N-1} = \mathbf{H}^{(2)}_{\frac{N-1}{2}} \oplus \mathbf{H}^{(3)}_{N} \oplus \mathbf{H}^{(1)}_{\frac{N-1}{2}},$$

gdzie podmacierze $\mathbf{H}^{(1)}_{\frac{N-1}{2}}$, $\mathbf{H}^{(2)}_{\frac{N-1}{2}}$ są odpowiednio dolno- i górno-triangularną macierzami Toeplitza

$$\mathbf{H}^{(1)}_{\frac{N-1}{2}} = \begin{bmatrix} h_0 & 0 & \cdots & 0 \\ h_1 & h_0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ h_{\frac{N-1}{2}-1} & h_{\frac{N-1}{2}-2} & \cdots & h_0 \end{bmatrix},$$

$$\mathbf{H}^{(2)}_{\frac{N-1}{2}} = \begin{bmatrix} h_{N-1} & h_{N-2} & \cdots & h_{\frac{N-1}{2}+1} \\ 0 & h_{N-1} & \cdots & h_{\frac{N-1}{2}+2} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & h_{N-1} \end{bmatrix},$$

zaś $\mathbf{H}^{(3)}_N = \mathbf{I}_N^{(\overset{N+1}{\frac{2}{}}\rightarrow)} \mathbf{H}_N$ jest macierzą cyrkulantem z przetasowanymi wierszami, natomiast macierz

$$\mathbf{H}_N = \begin{bmatrix} h_0 & h_{N-1} & \cdots & h_1 \\ h_1 & h_0 & \cdots & h_2 \\ \vdots & \vdots & \vdots & \vdots \\ h_{N-1} & h_{N-2} & \cdots & h_0 \end{bmatrix}$$

jest typową (nie zmodyfikowaną) macierzą cirkulantem, mnożenie przez którą można zrealizować za pomocą dowolnego z algorytmów szybkiego splotu kołowego;

- macierz tasowania danych

$$\widetilde{\mathbf{P}}_{2N-1} = \mathbf{I}_{\frac{N-1}{2}} \oplus \mathbf{I}_N^{(\overset{N-1}{\leftarrow\frac{2}{}})} \oplus \mathbf{I}_{\frac{N-1}{2}} \text{ oraz}$$

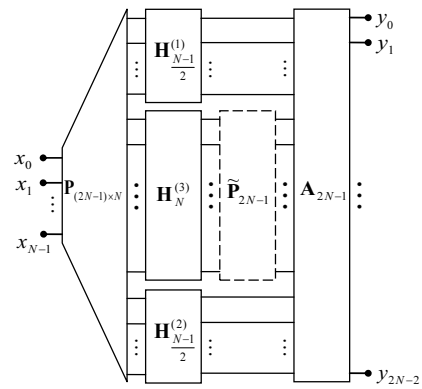
- macierz sumowania algebraicznego

$$\mathbf{A}_{2N-1} = (\mathbf{I}_{\frac{N-1}{2}} \| \mathbf{0}_{3\frac{N-1}{2}}) \overline{\overline{\left\{ [\mathbf{0}_{\frac{N+1}{2}} \overline{\overline{(-\mathbf{I}_{\frac{N-1}{2}})}}] \| \mathbf{I}_N \| [(-\mathbf{I}_{\frac{N-1}{2}}) \overline{\overline{\mathbf{0}}}_{\frac{N+1}{2}}] \right\}}}.$$

Uwzględniając wprowadzone konstrukcje wektorowo-macierzowe, algorytmiczną strukturę procesu realizacji bazowej operacji splotu liniowego można przedstawić następująco:

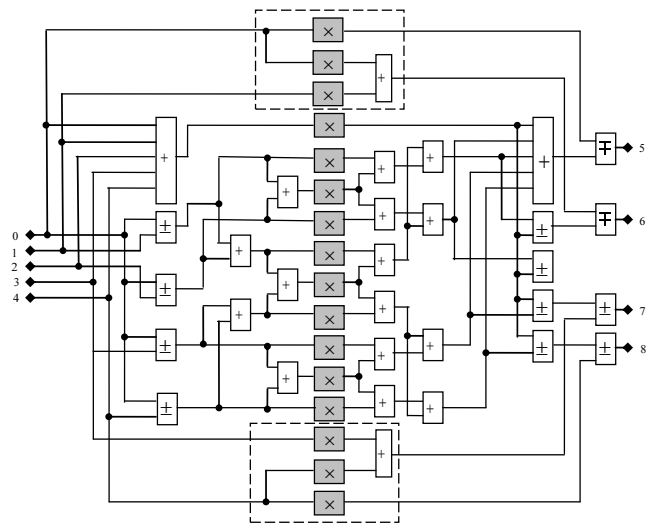
$$\mathbf{Y}_{(2N-1)\times 1} = \mathbf{A}_{2N-1} \widetilde{\mathbf{P}}_{2N-1} \mathbf{H}_{2N-1} \mathbf{P}_{(2N-1)\times N} \mathbf{X}_{N\times 1} \tag{3}$$

Na rysunku 1 została pokazana struktura jednostki procesorowej, realizującej operację bazową splotu liniowego zgodnie z opracowaną procedurą. Implementacja sprzętowa tej struktury będzie zawierać trzy moduły mnożenia macierzy stałych przez wektor (dwa - mnożenia triangularnych macierzy Toeplitza przez wektor oraz jeden moduł mnożenia cyrkulanta przez wektor), moduł tasowania danych (jest narysowany za pomocą linii przerwanej) oraz moduł dodawania algebraicznego. Należy dodać, iż moduł tasowania danych może zostać pominięty, jeśli z góry uwzględnić konieczność tasowania wierszy macierzy-cyrkulanta, modyfikując odpowiednio tradycyjny algorytm szybkiego splotu kołowego. Dlatego został właśnie pokazany za pomocą linii przerwanej.



Rys. 1. Struktura jednostki procesorowej realizującej bazową operację splotu liniowego według procedury (3)
Fig. 1. The processor unit structure for the implementation of the linear convolution basic operation corresponding to (3)

Na rysunku 2 jako przykład został pokazany wynik syntezy części operacyjnej jednostki procesorowej realizującej operację bazową splotu liniowego według proponowanej metody dla N=5.



Rys. 2. Struktura jednostki procesorowej realizującej bazową operację splotu liniowego według procedury (3) dla przykładu N=5
Fig. 2. The processor unit structure for the implementation of the linear convolution basic operation corresponding to (3) for N=5

Będzie ona składać się z szesnastu 2-wejściowych sumatorów, dwunastu 2-wejściowych sumatorów-substraktorów, dwóch 5-wejsciowych sumatorów oraz szesnastu układów mnożenia przez liczbę stałą. Jak widać liczba układów mnożących w tej strukturze została zredukowana z 25 do 16 względem „naiwnego" podejścia do sprzętowej realizacji obliczeń, to znaczy realizacji iloczynu macierzowo-wektorowego zgodnie z wyrażeniem (1). Liniami przerwanymi tutaj zaznaczone są bloki odpowiadające mnożeniu przez odpowiednie triangularne macierzy Toeplitza. Reszta układów w tej strukturze odpowiada części realizującej mnożenie macierzy-cyrkulanta przez odpowiedni wektor oraz blokowi dodawania algebraicznego (patrz też strukturę na rysunku 1).

Przy czym część dotycząca mnożenia macierzy cyrkulanta przez wektor została zrealizowana za pomocą odwzorowania w strukturze sprzętowej algorytmu szybkiego splotu dla N=5. Podkreślimy również, iż wyprowadzenia na rysunku 2, które ponumerowane są od 1 do 5 oznaczają wejścia, natomiast wyprowadzenia 6,7 – oznaczają wyjścia układu.

Jak widać z rozpatrywanego przykładu, przedstawione w tej pracy struktury mogą zostać z powodzeniem zaimplementowane w takich układach reprogramowalnych, jak Spartan-3, Stranix, Virtex –IV, które zawierają w swojej strukturze bloki mnożące.

Jednak z uwagi na to, iż opisywane w pracy struktury tak naprawdę wymagają bloków mnożenia przez wartość stałą, tzn. uproszczonych bloków mnożenia lub po prostu koderów (zostały na rysunku 2 zacienione), to przy ich implementacji mogą też być wykorzystane pospolite układy reprogramowalne, które nie zawierają wbudowanych przez producenta mnożarek. Takie bloki mogą być dość efektywnie zrealizowane w FPGA z tradycyjną strukturą bramkową.

3. Ocena zasobów sprzętowych

Jak widać, zaproponowane w artykule struktury algorytmiczne pozwalają zredukować łączną liczbę bloków mnożących względem metody „naiwnej". Dla rozważonego przykładu mamy 16 bloków mnożących zamiast 25. Co prawda liczba bloków dodawania nieco wzrasta, ale ze względu na znacznie większą złożoność bloków mnożących w stosunku do sumatorów mamy, jednak, ekonomię zasobów sprzętowych. W ogólnym przypadku zawsze będziemy mieli tyle bloków mnożenia przez stałą oraz tyle sumatorów ile zostanie określono poprzez implementację algorytmu szybkiego splotu kołowego, dwóch iloczynów wektorowo-macierzowych przez odpowiednie triangularne macierze Toeplitza oraz sumowania tych wyników zgodnie z procedurą (3).

4. Podsumowanie

W artykule opisano koncepcję specjolizowanego procesora do wyznaczenia bazowej operacji splotu liniowego. Założeniem artykułu jest, że liczba elementów współczynników odpowiedzi impulsowej filtru FIR jest nieparzysta, aczkolwiek nie stanowi żadnego problemu drobna modyfikacja procedury (3) oraz odpowiednich komponentów macierzowych w celu dostosowania algorytmu do sytuacji, gdy mamy do czynienia z parzystą liczbę współczynników. Rozpatrzono przykład syntezy szybkiego algorytmu oraz strukturę jednostki procesorowej realizującej tą operację dla przykładu N=5. Oczywiste jest, że w podobny sposób mogą być skonstruowane efektywne (posiadające mniej operacji mnożenia i dodawania) algorytmy oraz struktury jednostek procesorowych (ze zredukowaną liczbą bloków mnożących) do realizacji bazowych operacji splotu liniowego dla dowolnych długości filtrów.

5. Literatura

[1] R.E. Blahut, Fast algorithms for digital signal processing, Addison-Wesley Publishing company, Inc. 1985.
[2] R.C. Agarwal, J.W. Cooley, New algorithms for digital convolution. IEEE Transactions on Acoustics, speech, and Signal processing, vol ASSP-vol.AS25 (no.5), pp. 392-410, October 1977.
[3] H.J. Nussbaumer., Fast Fourier Transform and Convolution Algorithms, Springer-Verlag, 1982.
[4] C.S.Burrus, T.W. Parks, J.F. Potts, DFT/FFT and Convolution Algorithms and Implementation, John Willey & Sons, 1985.
[5] W. Selesnik, C.S. Burrus, Extending Winograd's Small Convolution Algorithm to Longer Lengths, In Proceedings of the IEEE International Symposium on Circuits and Systems, June 1994, pp. 2.449-2.452.
[6] R. Tolimieri M. An, C. Lu, Algotithms for Discrete Fourier Transform and Convolution, Springer-Verlag, 1989.
[7] K.K. Parhi, VLSI Digital Signal Processing Systems: Design and Implementation, John Willey & Sons, 1999.
[8] J.H. McClellan, C.M. Rader, Number Theory in Digital Signal Processing, Prentice-Hall, Inc. Englewood Cliffs, New Jersey, 1979.
[9] R. Stasiński, Extending sizes of effective convolution algorithms, Electronic letters, 1990, vol. 26, no 19, pp.1602-1604.
[10] E.E. Dagman., G.A. Kukharev. Szybkie dyskretne transformaty ortogonalne, Wydawnictwo Nauka, 1983.
[11] A. Țariov, Modele algorytmiczne i struktury wysokowydajnych procesorów cyfrowej obróbki sygnałów, Szczecin, Informa, 2001.

Artykuł recenzowany