Computer Science Journal of Moldova, vol.**8**, no.3(24), 2000

The Parallel Algorithms for a Correlation Function Computing [*]

A. Tsaryov

Correlation function along with convolution and discrete Fourier transformation is the basic operation of the majority of tasks of digital signal processing [1]. The necessity of fulfilling of numerous time-taking multiplication operations stimulated elaboration of economical "fast" algorithms, aimed at minimization of total quantity of these operations[2,3]. The effectiveness of such algorithms manifests itself, mainly by single-processor computing. On transitting to parallel super technology, the complexity of organization of computing processes when realizing "fast" algorithms grew, and the increase of data transfer operations share is transforming into complexity of inter-processor exchange procedures and thus increases the total time of computation. This fact can bring to naught the advantage in speed obtained due to the reduction of quantity of arithmetic operations necessary for computation of correlation function. The indicated circumstance forces to refuse complicated more than necessary, though effective from the computing point of view, algorithms of correlation function computation and stimulates elaboration of new ones, specially oriented towards parallel processing. Thus, for example, parallel algorithms, based on systolic principle of paralleling of computations [4] had been elaborated. However, this approach requires organization of special regime of data supply to the inputs of systolic correlator, but it can be not always easily realized. Another approach to the synthesis of parallel algorithms of digital signal processing is shown in the papers [5,6]. Within this approach algorithms proposed in the present article were obtained.

Let

$$\mathbf{X} = [x_0, x_1, \ldots, x_{N-1}]^T, \quad \text{be a dimension } N \times 1 \text{ vector}$$

[*] The work was partially supported by grant 16–085–0143/17–99–00.

$\mathbf{H} = [h_0, h_1, \ldots, h_{M-1}]^T$, be a dimension $M \times 1$ vector $M < N$;
A correlation function is defined as follows:

$$y_l = \sum_{m=0}^{M-1} h_{l+m} \cdot x_m; \qquad l = \overline{0, N-M} \tag{1}$$

The expression obtained completely defines the list of mathematical operations, which are to be used in order to implement a correlation function computing. Yet, the peculiarities of data processing organization being not quite evident, the investigation of these problems is of interest in order to find out some efficient methods of computation paralleling for solving this problem.

A generalized computational procedure describing the vectorized computational process that implements a correlation function computing, may be presented as follows:

$$\mathbf{Y} = \mathbf{A}^{(j)}_{(N-M+1)\times M(N-M+1)} \cdot \mathbf{H}^{(j)}_{M(N-M+1)} \cdot \mathbf{P}_{M(N-M+1)\times N} \cdot \mathbf{X}, \tag{2}$$

where

$\mathbf{H}^{(j)}_{M(N-M+1)}$ – is a diagonal matrix of the $M(N-M+1)$ order formed out of vector $\mathbf{H}$ elements;

$\mathbf{A}^{(j)}_{(N-M+1)\times M(N-M+1)}$ – is a summation matrix , whose dimension is defined by the lower index;

$\mathbf{P}^{(j)}_{M(N-M+1)\times N}$ – is an input data multiplexing matrix;

j – is an index of algorithm modification;

$y = [y_0, y_1, \ldots, y_n]^T$ – is a $(N-M+1)$ dimension vector describing the result of the given operation implementation.

For one of the most natural methods of computation vectoring these matrices are formed in the following way:

$$\mathbf{P}^{(l)}_{M(N-M+1)\times N} = \underset{l=0}{\overset{N-M}{\square}} \mathbf{P}^{(l)}_{M\times N};$$

$\mathbf{P}^{(l)}_{M\times N} = \mathbf{V}_{M\times N} \cdot \mathbf{I}^{(l\rightarrow)}_{N}$;

$\mathbf{V}_{M\times N} = \mathbf{I}_M \square|\square\ \mathbf{0}_{N-M}$,

where $\mathbf{I}$ here and further on is a unitary matrix the order of which is defined by the lower index whereas the upper index (if there is one) shows the number of positions to which the columns of this matrix are to be shifted in the direction of the arrow [7].

$\mathbf{H}^{(l)}_{M(N-M+1)} = \mathbf{I}_{N-M+1} \otimes \mathbf{H}_M$, where $\mathbf{H}_M = diag(h_0, h_1, \ldots, h_{M-1})$

$$\mathbf{A}_{(N-M+1)\times M(N-M+1)} = \mathbf{I}_{N-M+1} \otimes \mathbf{1}^T_M$$

Here [6]:
$\mathbf{1}^T_M$ – is a matrix all the elements of which are equal to 1 while its dimension is defined by the lower index;

$\otimes$ – Kronecker product sign for two matrices;

$\underset{i=0}{\overset{N-1}{\otimes}}$ – Kronecker product sign for N matrices;

$\oplus$ – tensor sum sign for two matrices;

$\underset{i=0}{\overset{N-1}{\oplus}}$ – tensor sum sign for N matrices ;

$\frac{\square}{\square}$ – vertical sum sign for two matrices;

$\underset{i=0}{\overset{N-1}{\frac{\square}{\square}}}$ – vertical sum sign for N matrices;

$\square|\square$ – horizontal sum sign for two matrices;

$\underset{i=0}{\overset{N-1}{\square|\square}}$ – horizontal sum for N matrices.

Fig.1 presents a graph-structural model of implementing vector $\mathbf{Y}$ parallel computation corresponding to the first method for $N = 7$,

$M = 3$. Data transfer operations are shown by straight lines; dots mark summation operations and circles show the operations of multiplication by a scalar written inside each of these circles.

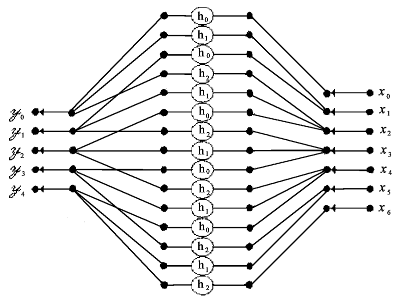

Figure 1.

As seen from the figure, data processing using the method considered can be carried out simultaneously over $N - M + 1$ channels, irrespective of the data input speed, to the input of the device which implements this procedure.

For the second method the respective matrices will be formed as follows:

$$\mathbf{P}^{(2)}_{M(N-M+1)\times N} = \overset{M-1}{\underset{\alpha=1}{\overset{\square}{\square}}} \mathbf{P}^{(\alpha)}_{(N-M+1)\times N},$$

$$\mathbf{P}^{(\alpha)}_{(N-M+1)\times N} = \mathbf{V}_{(N-M+1)\times N} \cdot \mathbf{I}^{(\alpha\to)}_{N};$$

$$\mathbf{V}_{(N-M+1)\times N} = \mathbf{I}_{N-M+1}\square|\square\; \mathbf{0}_{(N-M+1)\times(M-1)};$$

$$\mathbf{H}_{M(N-M+1)} = \overset{M-1}{\underset{\alpha=0}{\oplus}} \mathbf{H}^{\alpha}_{N-M+1}; \ \mathbf{H}^{(\alpha)}_{N-M+1} = \mathbf{I}_{N-M+1} \cdot h_{\alpha};$$

$$\mathbf{A}^{(2)}_{(N-M+1)\times M(N-M+1)} = \mathbf{1}^{T}_{M} \otimes \mathbf{I}_{N-M+1}.$$

Fig.2 shows a graph-structural model of a correlation function paralleling computation corresponding to the second method for $N = 7$, $M = 3$. This method is different from the previous one in that it supposes that there are M parallel operating channels in the device which implements the procedure, $N-M+1$ elements of the initial data vector being to be processed in each channel.

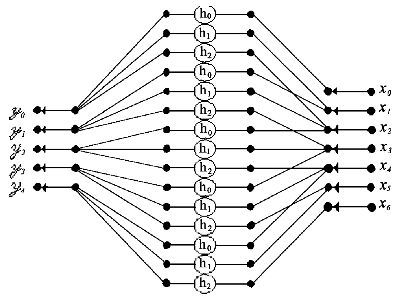

Figure 2.

The third method of a correlation function computing paralleling presupposes data processing in N channels at a time, and the respective matrices can be formed as follows:

$$\mathbf{P}^{(3)}_{M(N+M-1)\times N} = (\overset{M-1}{\underset{m=1}{\oplus}} \mathbf{1}_{m}) \oplus (\mathbf{I}_{M} \otimes \mathbf{1}_{M}) \oplus (\overset{M-1}{\underset{m=1}{\oplus}} \mathbf{1}_{M-m});$$

$$\mathbf{H}^{(3)}_{M(N-M+1)} = (\bigoplus_{m=1}^{M-1} \mathbf{D}^{(m)}) \oplus (\mathbf{I}_M \otimes (\bigoplus_{k=1}^{M} h_{M-k})) \oplus (\bigoplus_{m=1}^{M-1} \mathbf{B}^{(m)}),$$

where $\mathbf{D}^{(m)} = \bigoplus_{\alpha=1}^{m} h_{m-\alpha}$, and $\mathbf{B}^{(m)} = \bigoplus_{\alpha=1}^{M-m} h_{M-\alpha}$ – diagonal matrices of $(k+1)\times(k+1)$ and $(M-l-1)\times(M-1-1)$ dimensions, respectively, formed out of vector $\mathbf{H}$ elements:

$$\mathbf{A}^{(3)}_{(N-M+1)\times M(N-M+1)} =$$

$$= (\square|\square_{m=1}^{M-1} \mathbf{V}^{(m)}_{(N-M+1)\times m}) \square|\square (\square|\square_{l=0}^{N-M} \mathbf{W}^{(l)}_{(N-M+1)\times M}) \square|\square$$

$$(\square|\square_{m=1}^{M-1} \mathbf{\Lambda}_{(N-M+1)\times(M-m)}),$$

$$\mathbf{V}^{(m)}_{(N-M+1)\times m} = \mathbf{I}_m \frac{\square}{\square} \mathbf{0}_{((N-M+1)-m)\times m};$$

$$\mathbf{W}^{(l)}_{(N-M+1)\times M} = \mathbf{I}^{(l\rightarrow)}_{N-M+1} \cdot \mathbf{W}_{(N-M+1)\times M};$$

$$\mathbf{W}_{(N-M+1)\times M} = \mathbf{I}_M \frac{\square}{\square} \mathbf{0}_{M\times(N-M)};$$

$$\mathbf{\Lambda}_{(N-M+1)\times(M-m)} = \mathbf{0}_{((N-M+1)-(M-m))\times(M-m)} \frac{\square}{\square} \mathbf{I}_{M-m};$$

$\mathbf{0}$ – is a zero matrix of the dimensionality indicated.

A graph-structural model of computation arrangement corresponding to the third paralleling method for $N = 7$, $M = 3$ is given in Fig.3.

At maximum computation paralleling, when the simultaneously processed data vector dimension is equal to $M(N-M+1)$, i.e., is maximal, all the three methods are equivalent from the point of view of computation delay. Such implementation requires considerable hardware expenditure of $M(N-M+1)$ multipliers and $N-M+1$ of M-input adders. Time delay for a correlation function implementing is minimal $\tau = \tau_\times + \tau_+$, where $\tau_\times$, τ_+ are the duration of multiplication and M-input addition operations. It is obvious that the application of

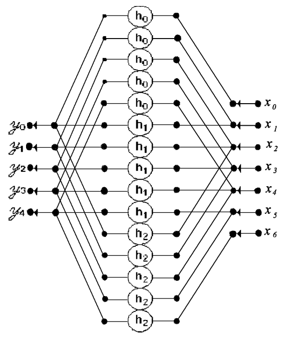

Figure 3.

maximum computation paralleling is permitted either when the total data input time to the device inputs is extremely short (compared to the processing time in each channel) or when all the data are coming simultaneously and minimum delay time of the result readiness is required. In practical work such fast acting is not, as a rule, required. In some cases it is enough to implement just parallel- successive processing. Within the methods described there exist several variants of parallel-successive data processing. In case the designers have at their disposal no more than M multipliers and only one M-input adder, it would be advisable to use the parallel- successive implementation of the first method, and the time required for data processing will be defined by the value of $\tau = \tau_\times + \tau_+ \times (N - M + 1)$. In another case, when there are $N-M+1$ multipliers and the same number of accumulating adders, it would be more convenient to make use of the parallel-successive implementation of the second method. This time the calculation result delay will be defined as follows:

$\tau = (\tau_\times + \tau_+)M$, where τ_+ is an accumulating type summarizer processing time.

In still another case, when the data are coming to the device input with some delay, it is possible to syntheses the efficient implementation of the third method which permits to rationally combine the data input and processing. In this case due to some complication of computation process control (when additional multiplexing is introduced) both time and hardware expenditures are optimized. For instance, if the input data are coming to the input of the device which implements the operation under consideration after a $\Delta t \geq \tau_+^M + \tau_\times$ time interval, it is sufficient to have no more than M multipliers and M accumulating adders the inputs of which are to be multiplexed. In this case the data at the device output will appear after a $\tau_M \approx (\tau_+^M + \tau_\times) \cdot M$ time interval, whereas the last result value y_{N-M+1} will appear after a $\tau_N \approx (\tau_+^M + \tau_\times) \cdot N$ time interval.

Thus, proceeding from the requirements of a particular problem to data processing time, from the peculiarities of this problem, from the number and the nomenclature of standard units the designer has at his disposal, also from weight and overall dimensions restrictions, the algorithms designed enable to select one or another variant of implementing one of the above described computation paralleling methods and to find a reasonable compromise between the device fast acting required and the hardware expenditure. It is quite natural that it is impossible in the article to consider all the varieties of parallelism implementation variants within the conceptions listed above, as well as the peculiarities of hardware solutions. However, the principal aspects of computation process paralleling of correlation function computing, which permit to evaluate the possibilities of raising the efficiency of devices for implementing the given operation, have been reflected on. As to the problems of these devices practical implementation, they deserve a special discussion and are going to be considered in the author's next publication.

References

[1] J.S. Bendat, A.G. Piersol. *Engineering application of correlation and spectral analysis.* John Wiley & Sons, New York, 1980.

[2] L.R. Rabiner and B. Gold. *Theory and application of digital signal processing.* Prentice-Hall, Inc., Englewood Cliffs, N.J., 1975

[3] A. V. Openheim, R. V. Shafer. *Digital signal processing.* Prentice-Hall, Inc., Englewood Cliffs, N.J., 1988.

[4] S.Y. Kung. *VLSI array processors.* Prentice-Hall, Inc., Englewood Cliffs,N.J., 1975

[5] A. P. Tsaryov. *Parallel algorithm adaptation for accelerated Fourier Discrete Transformation computation to be realized on vector computers.* Proceedings of Higher Schools of the USSR, Series: Radioelectronics, 1985, No.11, pp. 98-100.

[6] A. P. Tsaryov. *Algorithmic principles of vectorizing computations over digital date arrays.* Acta ACademia, International Informatization Academi, Chishinev. Evrica, 1997, pp.67-99.

[7] E. E. Dagman and G. A. Kukharev. *Fast discrete orthogonal transforms.* Novosibirsk, Nauka, 1983.

A.Tsaryov,
Szczecin, Poland
e-mail: *aleksander.carev@wl.ps.pl*

Received August 15, 2000