Natalija Maslennikowa
Computer Engineering Department,
Politechnika Koszalińska
E–mail: olegmas@wp.pl
Anatolij Sergiyenko
Computer Engineering Department,
National Technical University of Ukraine„KPI”
E–mail: aser@comsys.kpi.ua

Scheduling of synchronous dataflow graphs for datapath synthesis

Keywords: data flow graphs (DFG), synchronous dataflow graphs (SDFs), matrix, pipelined datapath design.

Introduction

In personal computers, mobile phones, DSP systems the computational processes are performed, which are repeated with a period coinciding with the input data arrival. These algorithms are represented by the data flow graphs (DFG). DFG is a directed graph, whose nodes represent the operators, and arcs, or dataflows represent the means for the data transmission. Among DFGs the synchronous dataflow graphs (SDFs) are very popular. In SDF each actor generates and consumes a number of variables, which is unchanged from cycle to cycle [1,2].

The homogeneous SDFs, and multirate SDFs are distinguished. In the multirate SDF the number of variables, that are consumed, or generated by a node in a single cycle can be more than one. To simplify the analysis of the multirate SDF it is usually converted into equivalent uniform SDF [2]. The article deals only with homogeneous SDFs. To get the optimal structural solution it is necessary to find the schedule of the algorithm execution in SDF. The article deals with a new scheduling method based on spatial SDF, which provides the fast search for both effective schedule and structure of the datapath.

1. Scheduling for SDF

Consider an example of a test algorithm for computing the second order digital IIR filter, which implements the frequency response function [3]:

$$H(z)=\frac{1+bz^{-1}+dz^{-1}}{1+az^{-1}+cz^{-1}}$$

It is computed due to the following equations

$$u_i = x_i - a\,u_{i-1} - c\,u_{i-2}; \quad y_i = u_i + b\,u_{i-1} + d\,u_{i-2}, \tag{1}$$

where x_i, y_i are the input and output data samples. These equations are computed by SDF, shown in Fig.1. The nodes of multiplication, addition, input-output, delayed variable are marked by "+", "×", empty circle, and bold point, respectively. Dotted arrow means the interiteration dependency, and is loaded by a delay, marked by the thick line.

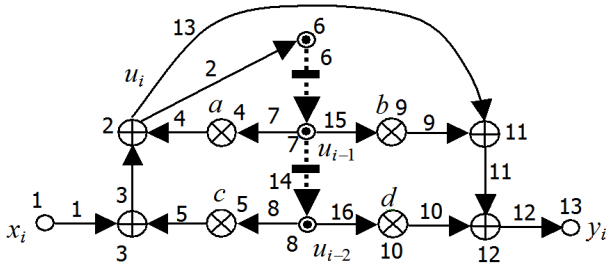

Fig. 1. SDF of the second order IIR filter

The SDF scheduling is one of the steps of the datapath synthesis. The other steps are resource selection, resource assignment, structure finding, control unit synthesis. When SDF is mapped into the structure by the one-to-one rule, then a single algorithm iteration is implemented for $L = 1$ clock cycle. This schedule is determined by the behavior of SDF, and the delays of the logic circuits, in which the graph nodes are mapped. But because of the great length of the critical path this schedule is non-rational. In this example, the length of the critical path is $T_C = t_M + 2t_A$, where t_M, t_A are multiplier, and adder delay, respectively.

The most popular scheduling methods for limited resources, and execution time consider the acyclic SDF subgraph. These methods are list scheduling, force directed scheduling [4]. The register allocation is effectively implemented by the Tseng heuristic, and by the left edge scheduling. The use of the cyclic interval graph takes into account the cyclic nature of the SDF algorithm [5]. The retiming methods, and the graph folding methods simplify the SDF mapping [3,6].

Each step of the datapath synthesis is usually performed independently, so as the most of the methods reduces the possibility of the global optimization. For example,

the hardware cost optimization during the resource selection is in contradiction with the cycle time minimizing during the scheduling. Below a method of the datapath synthesis is proposed in which the steps of resource selection, operator scheduling, and resource allocation are implemented in a single step providing the more effective optimization.

2. Spatial SDF and its schedule

In [7] the method of the datapath synthesis is described, in which SDF is presented in the three-dimensional space in the form of a triple $K_G = (K, D, A)$, where K is the matrix of vectors-nodes $\boldsymbol{K}_i$, which mean the operators, D is matrix of vectors-edges $\boldsymbol{D}_j$, performing the links between operators, A is the incidence matrix of SDF. In the vector $\boldsymbol{K}_i = (k_i, s_i, t_i)^T$ the coordinates k_i, s_i, t_i correspond to the type of operator, the processor unit (PU) number, and the clock cycle. The SDF graph in such a representation is called as spatial SDF.

Spatial SDF is splitted into the spatial configuration $K_{GS} = (K_S, D_S, A)$, and event configuration $K_{GT} = (K_T, D_T, A)$, which correspond to the datapath structure, and its schedule. By this splitting the vectors $\boldsymbol{K}_i = (k_i, s_i, t_i)^{\mathrm{T}}$ are decomposed into vectors $\boldsymbol{K}_{Si} = (k_i, s_i)^{\mathrm{T}}$, corresponding to the PU coordinates, and vectors $\boldsymbol{K}_{Ti} = t_i$, which mean the execution time of the relevant operators in PU $\boldsymbol{K}_{Si}$. Then the temporal component $\boldsymbol{D}_{\mathrm{Ti}} = t_i$ of the vector $\boldsymbol{D}_{\mathrm{i}}$ is equal to the delay of transfer, or processing of the relevant variable.

We can assume that the matrix K encodes some acceptable solution, since the matrix D is calculated by the equation

$$D = KA. \tag{2}$$

The structural optimization consists in finding such a matrix K, which minimizes a given quality criterion. It is possible to specify a matrix D_{O} which provide the minimum value of T_{C}. Then the vectors $\boldsymbol{K}_i$ are found from a relationship

$$K = D_{\mathrm{O}} A_{\mathrm{O}}^{-1}, \tag{3}$$

where D_{O} is the matrix of vectors-nodes, A_{O} is the incidence matrix of the maximum spanning tree for SDF. When looking for effective structural solution, the following relations have to be considered. Spatial SDF is valid, if the matrix K has no two identical vectors, i.e.

$$\forall\, \mathbf{K}_{\mathrm{i}}, \mathbf{K}_{\mathrm{j}}\ (\mathbf{K}_{\mathrm{i}} \neq \mathbf{K}_{\mathrm{j}}, \mathrm{i} \neq \mathrm{j}\ ). \tag{4}$$

The schedule with the period of L clock cycles is correct if the operators, which are mapped into the same PU, are performed in different cycles, i.e.

$$\forall\, \mathbf{K}_{\mathrm{i}}, \mathbf{K}_{\mathrm{j}} (\mathrm{k_i} = \mathrm{k_j},\ \mathrm{s_i} = \mathrm{s_j}) \Rightarrow \mathrm{t_i} \not\equiv \mathrm{t_j} \bmod \mathrm{L}\ . \tag{5}$$

Moreover, the next operator is executed no earlier than the previous one, i.e.

$$\forall \boldsymbol{D}_j \neq \boldsymbol{D}_{Dj}\ (t_i \geq 0). \tag{6}$$

where $\boldsymbol{D}_{Dj}$ is the vector of the interiterational dependence $\boldsymbol{D}_{Dj} = (k_j, s_j, -wL)^T$, which means a delay in w algorithm iterations. The operators of the same type should be mapped into PU of the same type, i.e.

$$\mathbf{K}_i, \mathbf{K}_j \in K_{p,q}(k_i = k_j = p, s_i = s_j = q), |K_{p,q}| \le L, \quad (7)$$

where $K_{p,q}$ is a set of p-type vectors-operators, which are mapped in the q-th PU of p-th type ($q = 1,2, \ldots q^p_{\max}$). Consider the vectors-edges $\boldsymbol{D}_j$, including the interiterational dependence vectors $\boldsymbol{D}_{Dj}$, which belong to the i-th graph cycle. Then for it the following equation must be satisfied

$$\sum_j b_{i,j} \boldsymbol{D}_j = (0,0,0)^T, \quad (8)$$

where $b_{i,j}$ is an element of the i-th row of the cyclomatic matrix of SDF. In the simplest case, each operator is executed in a single cycle. This is a natural situation in designing at the RTL level. Complex operators can be calculated for a few cycles. However, when operating in the pipelined mode, the corresponding PU contains stages, each of them has a single cycle delay.

Then the search for the schedule consists in the following. The vectors $\boldsymbol{D}_i \in D_O$ are assigned the coordinate $t_i = 1$, i.e. the respective operators have the delays of a single clock cycle. The matrix K_T is found from the equation (3). The remaining elements of the matrix D_T are found from the equations (2), and (8). If for some of vectors the inequality (6) is not satisfied, then the coordinate t_i is increased for certain vectors $\boldsymbol{D}_i \in D_O$, and the schedule search is repeated. The rest of $\boldsymbol{K}_i$ coordinates are found from the conditions (4) – (8). In such a way the fastest schedule is built, as each statement is executed in a single clock cycle without unnecessary delays. To minimize the searching for coordinates t_i in the vectors $\boldsymbol{D}_i$, in [8] the perfect maximum spanning tree of the SDF graph is proposed for the synthesis process.

The resulting spatial SDF can be described by the VHDL language, so the pipelined datapath description can be translated into the gate level description by the proper compiler-synthesizer [9].

3. Schedule search example

To estimate the method effectiveness consider the schedule searching for SDF in Fig.1. The perfect maximum spanning tree of the SDF is shown in Fig. 2. The additional vector $\boldsymbol{D}_B$ connects the coordinate system origin with an arbitrary node in order to provide the equation (3) consistency.

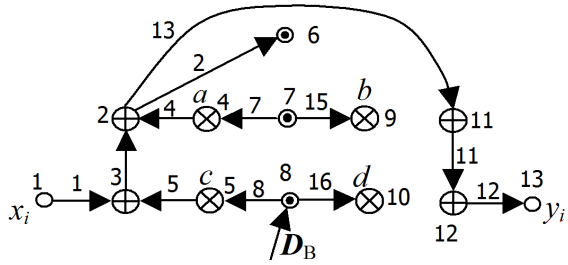

Fig. 2. Perfect maximum spanning tree for SDF in Fig. 1

In this example the adder, and pipelined multiplier have one, and two cycle delays, respectively. The following delay matrix is arranged using these delays:

$$D_{OT} = \overset{1\ 2\ 3\ 4\ 5\ 7\ \ 8\ \ 11\ 12\ 13\ 15\ 16\ B}{(1\ 1\ 1\ 2\ 2\ x_1\ x_2\ 1\ \ 1\ \ 1\ \ x_3\ x_4\ 0)}.$$

where $x_1,\ldots,x_4$ are unknown values, that depend on other vectors in equations (8), whose number is equal to the number of edges minus the number of nodes in SDF. That is, every graph cycle must have at least one edge with the variable delay. So we get a system of four equations (8):

$$\begin{cases} D_2 + D_4 + D_6 + D_7 = 0, \\ D_3 + D_5 + D_8 + D_{14} - D_4 - D_7 = 0, \\ D_2 + D_6 + D_{15} + D_9 - D_{13} = 0, \\ D_{10} + D_{14} + D_{16} - D_9 - D_{11} - D_{15} = 0. \end{cases}$$

The solving of this system of equations gives the solution $x_1 = 1$; $x_2 = 4$; $x_3 = 2$; $x_4 = 7$. The algorithm period of $L = 4$ clock cycles is selected to get a single multiplier unit. Due to (2) the matrix K_T is found:

$$K_T = A_O\ \overset{1\ 2\ 3\ 4\ 5\ 7\ 8\ 11\ 12\ 13\ 15\ 16\ B}{(1\ 1\ 1\ 2\ 2\ 1\ 4\ 1\ \ 1\ \ 1\ \ 2\ \ 7\ \ 0)} = \overset{1\ 2\ 3\ 4\ 5\ 6\ 7\ 8\ 9\ 10\ 11\ 12\ 13}{(5\ 7\ 6\ 5\ 4\ 8\ 4\ 0\ 6\ 7\ \ 8\ \ 9\ \ 10)}.$$

The rest of the matrix K elements is obtained according to the equation (4). One should take into account that the number of nodes, standing in one line, which is parallel to the axis *ox*, approaches to L:

$$K = \begin{pmatrix} 1 & 2 & 2 & 3 & 3 & 4 & 4 & 4 & 3 & 3 & 2 & 2 & 4 \\ 0 & 1 & 1 & 2 & 2 & 3 & 3 & 3 & 2 & 2 & 1 & 1 & 4 \\ 5 & 7 & 6 & 5 & 4 & 8 & 4 & 0 & 6 & 7 & 8 & 9 & 10 \end{pmatrix}.$$

A graphic representation of the constructed space SDF is shown in Fig.3. Fig. 4. illustrates the structure of the corresponding digital filter. The small figures in it mark the clock cycles, in which the respective multiplexor inputs, and registers accept the data.

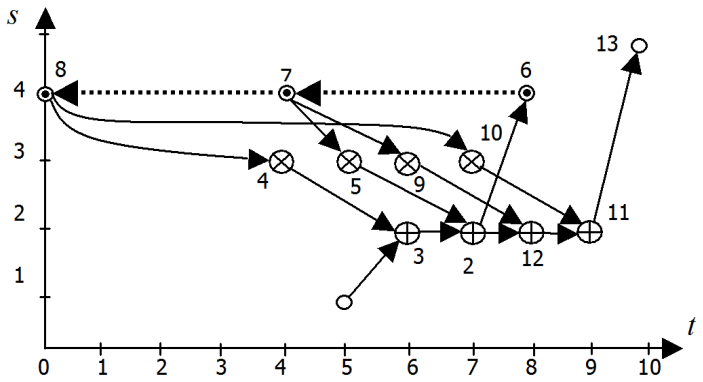

Fig. 3. Space SDF, which calculates equations (1)

One can see that this structure contains in four registers, and five multiplexor inputs less than the filter structure derived in [3], providing the same data throughput. The critical path is minimized to $T_C = \max(t_M/2, t_A)$. Besides, it is derived using a set of formal rules.

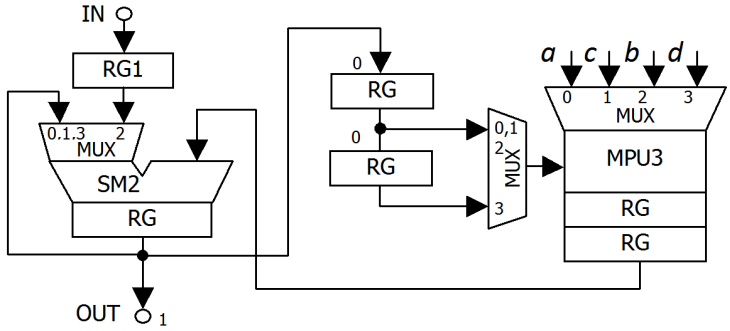

Fig. 4. Structure of the IIR filter

4. Conclusions

A method of the SDF scheduling is proposed, which provides the formal design of pipelined structures with high throughput and minimized hardware volume. The example of the IIR filter structure synthesis shows that for small SDFs the exact solution is possible. The experience of design of more complex projects like FFT processors, DCT processors, multistaged IIR filters [10] showed that the method use must be provided by a set of combinatorial optimization steps. But the number of these steps is rather small because the method has a set of limitations to the vectors of the spatial SDF. The method can be used not only for the pipelined datapath design, but for programming the parallel computing systems.

References

1. Edwards, S., Lavagno, L., Lee, E.A., Sangiovanny-Vincentelli, A. Design of Embedded Systems: Formal Models, Validation, and Synthesis, Proc. IEEE, Vol. 85, No. 3, p. 366–390, 1997.
2. Lee, E. A., Messerschmitt D. G. Static scheduling of synchronous data flow programs for digital signal processing. IEEE Trans. on Computers, Vol. 36, No. 1, p. 24–35, 1987.
3. Keshab, K., Parhi, K.K., Chen, Y. Signal Flow Graphs and Data Flow Graphs. In Handbook of Signal Processing Systems, Bhattacharyya, S.S., Deprettere, E.F., Leupers, R. – Ed-s, Springer, p. 791-816, 2010.
4. Paulin, P.G., Knight, J.P. Force – Directed Sheduling for the Behavioral Synthesis of ASICs, IEEE Trans. CAD, –Vol. 7, – No. 3, p. 356–370, 1988.
5. The Systhesis Approach to Digital System Design, Editors P. Micheli, U. Lauther, P. Duzy, Kluwer Academic Pub, 415 p., 1992.
6. Introduction to Scheduling, Y. Robert, and F. Vivien – Ed-s, CRC Press, Taylor and Francis Group, 310 p, 2010.
7. Sergiyenko, A.M., Maslennikow, O., Vinogradow Y. Tensor approach to the application specific processor design, Proc.10th Int. Conf. "The Experience of Designing and Application of CAD Systems in Microelectronics", CADSM'2009, 24-28 Feb. 2009, IEEE Library, –p. 146–149, 2009.
8. Sergiyenko, A.M. Doskonaly kistyak grafa algorithmu, Vistnyk NTUU–KPI: Informatics, Operation, and Computer Science, No. 46, p. 62–67, 2007. (in Ukrainian)
9. Sergiyenko, A.M. VHDL for Design of Computers, Kiev: DiaSoft, 210 p, 2003. (in Russian)
10. Sergiyenko, A., Uzenkov, O. Low-Pass IIR Filter, 2010. Available at http://opencores.org/project,lp_iir_filter

Summary

A method of the schedule searching is proposed, which is based on the properties of the spatial SDF. The method is based on the SDF representation in the multidimensional space. The dimensions of this space are spatial coordinate of the processing unit, time moment of the operator calculation, and operator type. During the synthesis, the nodes are placed in the space according to a set of rules, providing the minimum hardware volume for the given number of clock cycles in the algorithm period. The resulting spatial SDF is described by VHDL language and is modeled and compiled using proper CAD tools. The method is successfully proven by the synthesis of a set of FFT processors, IIR filters, and other pipelined datapaths for FPGA.

Słowa kluczowe: graf zależności informacyjnych (DFG), synchroniczny graf zależności informacyjnych(SDFs), macierz, projektowanie potokowej jednostki przetwarzającej.

Streszczenie

Zaproponowany sposób poszukiwania opiera się na właściwościach przestrzennych SDF. Metoda ta bazuje na SDF prezentacji w przestrzeni wielowymiarowej. Wymiarami danej przestrzeni są współrzędne jednostki przetwarzającej, czas momentu obliczeniowego operatora oraz typ operatora. Podczas syntezy, węzły są umieszczone w przestrzeni zgodnie z zestawem reguł, dając minimalną częstotliwość pracy zegara systemowego podczas wykonywania algorytmu. Powstały przestrzenny SDF jest opisany przez język VHDL i jest modelowany i skompilowany przy użyciu odpowiednich narzędzi CAD. Ta metoda jest pomyślnie sprawdzona przez syntezę zestawu procesorów FFT, filtrów IIR, oraz innych potokowych jednostek przetwarzających płytki FPGA.