

Implementing Algorithms on Multilayered Reconfigurable Arrays

Toomas P. Plaks
SCISM, South Bank University
103 Borough Road, London SE1 0AA, UK
E-mail: plakst@sbu.ac.uk

1 Introduction

The emerging reconfigurable computing platform [11, 5, 1] provides new possibilities for developing flexible application-specific processors with low cost and high performance. The advance in reconfigurable computing necessitates the search for new algorithmic and architectural solutions together with developing high-level design tools for automatic mapping of algorithms into reconfigurable platform. The design systems still lag seriously behind by their capability and functionality because the underlying synthesis theory of adequate hardware/software structures is not implemented satisfactorily in design tools, or the gaps in synthesis theory itself have remained to be researched.

Regular or systolic arrays have been a widely used approach for efficient implementation of algorithms on reconfigurable computers [10, 6, 4]. The theory of regular arrays was established in the light of VLSIs and, thus, the main aim has been the optimizing of design, i.e. minimizing the time or the number of processors required for solving a given problem. Now, in the light of emerging reconfigurable computing, the main emphasis must be lifted to the reconfiguring of algorithms to fit the particular hardware structure (the available area and configuration) together with time parameters.

This paper deals with the extending of regular array theory: improving architectural and algorithmic properties of classical regular arrays, and presents a variety of different applications from signal and image processing. The respective Iso-plane synthesis method [9, 8] is briefly described in this paper.

2 Architecture and Algorithms

This paper deals with 3-D multilayered arrays that consists of 2-D subarrays connected with each other only by few edges (Fig. 1(a)). Multilayered arrays include, for example, a motherboard with FPGA daughter-boards [3] as layers, or a multiboard reconfigurable FPGA computer [2] where each board can serve as a layer.

Classical regular arrays are 1 or 2-D. The array size and the degree of parallelism depend on the size of problem. Mapping algorithms onto higher-dimensional (3-D) arrays makes it possible to improve the degree of parallelism and relaxes the size dependence. We use similar algorithms as on hypercubes. Not all connections, involved by a hypercube, are used in implementation of a particular algorithm. For example, efficient tree-like algorithm for reduction operator on

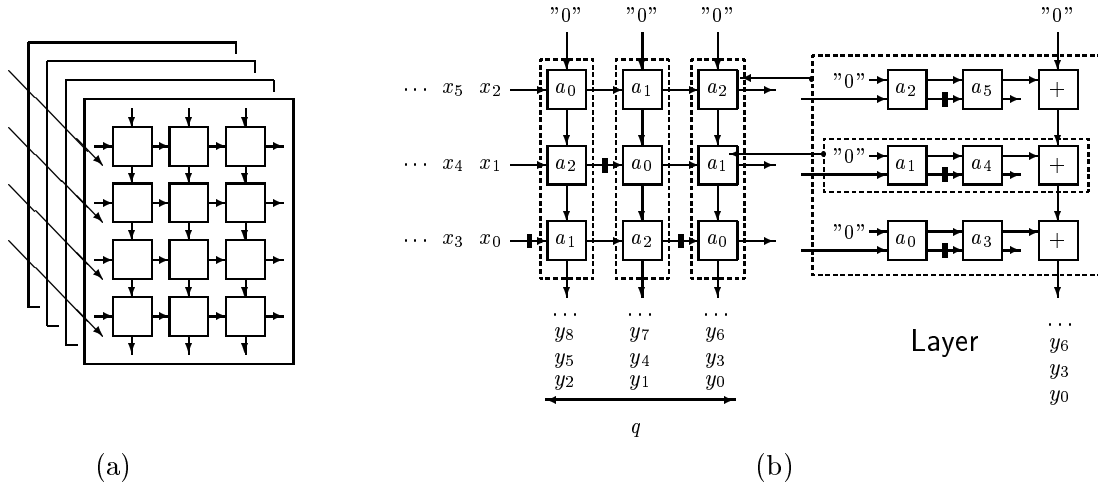


Figure 1: (a) Multilayered architecture. (b) Multilayered structure for 1-D convolution.

a hypercube uses only few connections. Reconfigurable computing on FPGAs allows us to customize the connection structure to the particular application. This makes it possible to produce higher-dimensional arrays that can be split into layered structures providing, thus, easily implementable structures in hardware.

For example, the conventional 1-D systolic array for r -point FIR-filter has the size $\mathcal{O}(r)$ and requires $\mathcal{O}(N)$ time units, where N is the length of the input stream. FIR-filter can be implemented on a 3-D multilayered array of size $\mathcal{O}(qr)$, where $q, 0 \leq q < N$, is the number of input/outputs. This array computes in $\mathcal{O}(N/q)$ time and has the latency time $\mathcal{O}(n^{1/2})$ (Fig. 1(b)). We present multilayered array structures for efficient implementation of algorithms from signal and image processing: so-called 3-D algorithms (matrix multiplication, back-substitution, etc.), FIR-filter, string matching, template matching [7] and discrete transformations (DCT, DFT).

3 Synthesis

In regular array theory the dependence graph of a perfectly nested for-loop algorithm is embedded into Euclidean space — into an r -D space for r nested for-loops. Integer points in the Euclidean space represent the loop body computations and vectors denote the dependencies between computations [10, 6, 4]. Such a model is called a polytope model of algorithm. The problem of mapping algorithms into hardware — i.e. into a physical space and time — is now to find a new coordinate system where one axis can be treated as time and the others as space.

Efficient reconfiguring of algorithms requires the rewriting of algorithms using algebraic transformations. Each algebraic transformation of a problem specification provides, in general, a different dependence graph. In this paper we use the Iso-plane method [9, 8] for directly manipulating the polytope model of algorithm for doing specific algebraic transformations. This method increases the dimensionality of the problem representation and uses specific techniques for moving data around the array. Note, that a high-level problem specification does not specify the exact order of computations. The Iso-plane technique allows us to change the interconnection structure between elementary processors and to achieve a higher performance and a better layout — multilayered structure.

4 Conclusions

Multilayered structures fit the physical construction of processors and offer better performance in comparison to classical regular arrays. Iso-plane method extends the classical space-time mapping technique in regular array theory and allows the automatic synthesis of multilayered arrays. Thus, the high-level design tools can be built basing on this approach.

The iso-plane technique produces a large variety of new scalable regular array solutions where the performance (i.e., the number of processors and I/O channels) and array structure are largely parameterized and can be chosen according to the particular needs of applications. This method allows us to develop efficient hardware solutions not known in the literature and not obtainable by using existing methods.

References

- [1] T. M. Conte, P. K. Dubey, M. D. Jennings, et al. Challenges to combining general-purpose and multimedia processors. *Computer*, 30(12):33–37, December 1997.
- [2] B. K. Gunther. SPACE 2 as a reconfigurable stream processor. In *4th Ann. Australian Conf. on Parallel and Real-Time Systems*, page , 1997.
- [3] S. D. Haynes, J. Stone, P. Y. Cheung, and W. Luk. Video image processing with the Sonic architecture. *Computer*, 33(4):50–57, April 2000.
- [4] C. Lengauer. Loop parallelization in the polytope model. In E. Best, editor, *CONCUR'93*, Lecture Notes in Computer Science 715, pages 398–416. Springer-Verlag, 1993.
- [5] W. H. Mangione-Smith, B. Hutchings, D. Andrews, et al. Seeking solution in configurable computing. *Computer*, 30(12):38–43, December 1997.
- [6] G. M. Megson. *An Introduction to Systolic Algorithm Design*. Clarendon Press, Oxford, UK, 1992.
- [7] T. P. Plaks. Mesh of linear arrays for template matching. *Real-Time Imaging Journal, special issue on Special Purpose Architectures for Real Time Imaging*, 6(2):373–382, December 1996.
- [8] T. P. Plaks. Efficient mapping reductions using iso-planes on the polytope model. *Parallel Algorithms and Applications*, 13(4):321–343, 1999.
- [9] T. P. Plaks. *Piecewise Regular Arrays: Application-Specific Computations*, volume 1 of *Parallel Processing Series*. Gordon and Breach Science Publishers, 1999.
- [10] P. Quinton and Y. Robert. *Systolic Algorithms and Architectures*. Prentice Hall, Masson, UK, 1991.
- [11] J. Villasenor and W. H. Mangione-Smith. Configurable computing. *Scientific American*, pages 66–71, June 1997.