IOSR Journal of VLSI and Signal Processing (IOSR-JVSP)
Volume 4, Issue 4, Ver. II (Jul-Aug. 2014), PP 06-11
e-ISSN: 2319 – 4200, p-ISSN No. : 2319 – 4197
www.iosrjournals.org

High Performance Systolic Architecture by Evolutionary Design

[1]Hanumanthareddy P S, [2]Shilpa K C
[1,2] *Dept. of Electronics & Communication Engineering,*
Dr.Ambedkar Institute of Technology,

Abstract: *This paper focuses on design of systolic architecture and by the application of evolutionary programming to achieve 100% HUE (Hardware Utilization Efficiency) for space representation containing delays in systolic design. Heuristic method involves EP (Evolutionary Programming) to solve the problems where the solution is in large search space. Using EP, we are able to find the number of different solutions for designing the systolic architecture for regular iterative algorithms. From which the user can choose any design of his choice according to his application.*
Keywords: *Systolic array architecture, Processing Element, Hardware Utilization Efficiency*

I. Introduction

The essential goal of developing new computer architectures and efficient use of existing modern systems is to run larger and more complicated applications faster over time. The continued demand for increased computing power led in the late. 1980's to the development of high parallel scalable multiprocessing systems. Parallelism is an intuitive and appealing concept. Parallel computing is a form of computation which many calculations are carried out simultaneously ,operating on the principle that large problems can often be divided into smaller ones, which are then solved concurrently ("in parallel").There are basically two ways to improve the computer performance in terms of computational speed. One way is to use faster devices (VLSI chips). Although faster and faster VLSI components have contributed a great deal on the improvement of computation speed, the breakthroughs in increasing switching speed and circuit densities of VLSI devices will be difficult and costly in future. The other way is to use parallel processing architectures, which employ multiple processors to perform a computation task. When multiple processors working together, an appropriate architecture is very important to achieve the maximum performance in a cost- effective manner. Systolic arrays are ideally qualified for computationally intensive applications with inherent massive parallelism because they capitalize on regular, modular, rhythmic, synchronous, concurrent processes that require intensive, repetitive computation.

1.1 Systolic Array:

A set of simple processing elements with regular and local connections, which takes external, inputs and processes them in a predetermined manner in a pipelined fashion. Features of systolic array: synchrony, Modularity, Regularity, Spatial locality, temporal, locality & pipelinability.

1.2 Evolutionary Programming:

Evolutionary Programming (EP) is presented by L. J.Fogel. He initially studied this method to develop the artificial intelligence and succeeded in evolving a mathematical automaton that predicts a binary time series. EP form a subset of evolutionary computation in that they generally only involve techniques implementing mechanisms inspired by biological evolution, such as reproduction, mutation, recombination, natural selection and survival of fittest. Candidate solutions to the optimization problem play the role of individuals in a population, and the cost function determines the environment within which the solutions "live". Evolution of the population then takes place after the repeated application of the above operators. In this process, there are two main forces that form the basis of evolutionary systems: Recombination and mutation create the necessary diversity and thereby facilitate novelty, while selection acts as a force increasing quality.

II. Systolic Architecture Design

Systolic Architecture, a general methodology for mapping high-level computations into hardware structures. This can be rectangular, triangular or hexagonal to make use of higher degrees of parallelism. Moreover to implement a variety of computations, data flow in a systolic system may be at multiple speeds in multiple directions-both inputs and (partial) results flow, whereas only results flow in classical pipelined systems. Systolic architecture results in cost-effective, high performance special purpose system for a wide range of problems. The cycle for developing special purpose systems can be divided into 3 phases –task definition, design and implementation.

2.1 Systolic architectures: Basic principle

Systolic architectures, an architectural concept originally proposed for VLSI implementation of some matrix operations. A systolic system consists of a set of interconnected cells each capable of performing some simple operation. as simple, regular communication and controlled structures have substantial advantages over complicated ones in design and implementation, cells in a systolic system are typically interconnected to form a systolic array or systolic tree. Information in systolic systems flows between cells in a pipelined fashion and communication with the outside world occurs only at the "boundary cells". The basic principle of a systolic architecture, a systolic array in particular is shown in fig.1

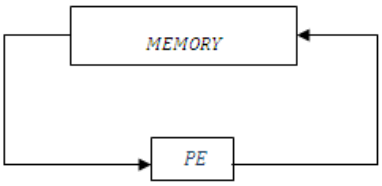


Fig. 1 Basic Principle of Systolic Array with single PE

By replacing the single processing element with an array of PE's as shown in fig.2. a higher computation throughput can be achieved without increasing memory bandwidth. Being able to use each input data item a number of times (and thus achieving high computation throughput with only modest memory bandwidth) is one of the many advantages of systolic approach.

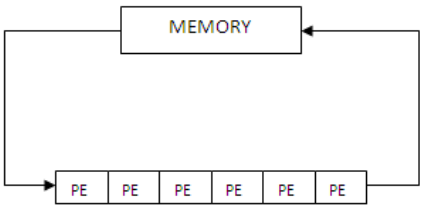


Fig 2 Basic Principle of Systolic Array

III. Systolic Array Design Methodology:

Design 1

To design a systolic array, when the three vectors namely the projection vector, processor vector and scheduling vectors are given, is shown in fig.3

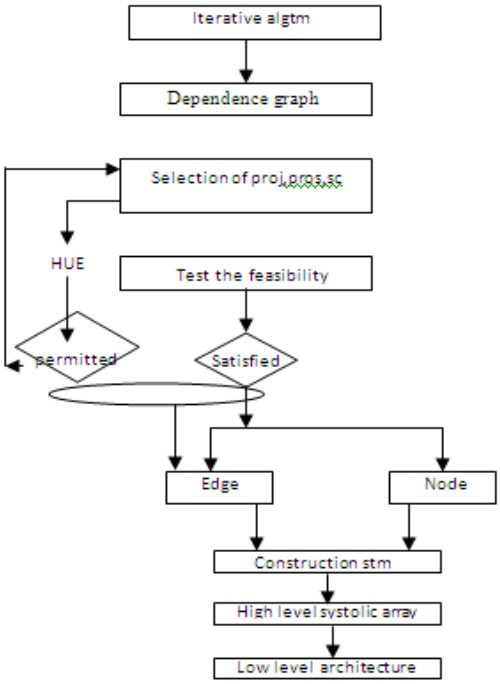


Fig 3 Design1 methodology

Systolic array architectures are designed by using linear mapping techniques on regular dependence graphs (DG). The DG corresponds to space representation, where no time instance is assigned to any computation (t=0). A DG is said to be regular, if a presence of an edge in a certain direction at any node in the DG represents presence of an edge in the same direction at all nodes in the DG. Systolic architectures have space-time representation where each node is mapped to a certain PE and is scheduled at a particular time instance. Systolic design methodology maps an N-dimensional DG to a lower dimensional systolic architecture. Mapping of N-dimensional DG to (N-1) dimensional systolic array is considered.

Basic vectors involved in the systolic array design are:
1. Projection vector: (also called as iteration vector)

$d = \begin{bmatrix} d1 \\ d2 \end{bmatrix}$ d is the projection vector.

Two nodes that are displaced by 'd 'or multiples of 'd 'are executed by same processor.

2. Processor space vector

$P^T = (p_1, p_2)$

Any node with index $I^T = (i, j)$ would be executed by processor in space time representation by: $P^T I = (p_1, p_2)$

3. Scheduling Vector $S^T = (s_1, s_2)$

Any node with index I would be executed by: $S^T I = (s_1, s_2) \begin{bmatrix} i \\ j \end{bmatrix}$

4. Hardware Utilization Efficiency

$HUE = 1/|S^T d|$

This is because two tasks executed by the same processor are spaced $|S^T d|$ time units apart.

Much systolic architecture can be designed for a given problem by selecting different projection, processor space and scheduling vectors. These vectors must satisfy the feasibility constraints.

The feasibility constraints are:

1.If points A and B differ by the projection vector, i.e. $(I_A - I_B)$ is same as d, then they must be executed by the same processor. in other words:

$P^T I_A = P^T I_B$

$P^T (I_A - I_B) = 0$

$P^T d = 0$

2. If A and B are mapped to the same processor, then they cannot be executed at the same time. i.e. $S^T I_A \neq S^T I_B$

$S^T (I_A - I_B) \neq 0$

$S^T d \neq 0$

Once the above two constraints are satisfied, edge mapping is done.

Edge mapping:

If an edge E exists in a space representation or DG then an edge $P^T e$ is introduced in a systolic array with $S^T e$ delays. Once edge mapping is done, then we construct low-level implementation of the particular design. Now the edges can be mapped using table 1

e^T	$P^T e$	$S^T e$
wt (1 0)	0	1
i/p (0 1)	1	1
Result(1 −1)	-1	0

Table 1 Edge Mapping for Design

Design 2

When the three vectors are not given, then we go for RIA description. From which we are able to draw RDG. So that for the FIR problem, the standard output RIA description is as follows

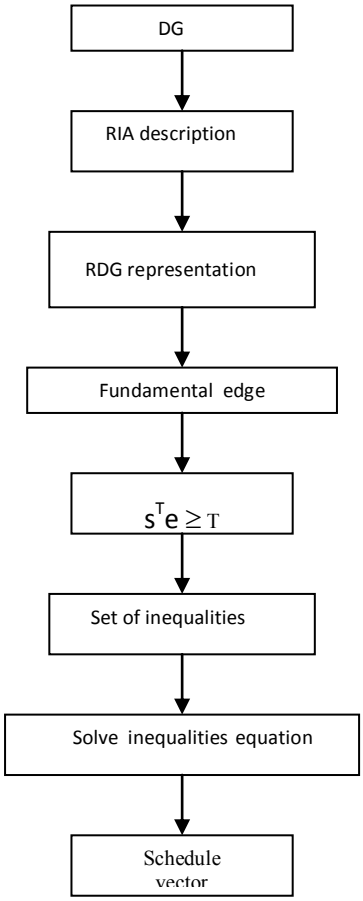


Fig 4 Design 2 methodology

IV. Test & Results

Design of SA for 3-TAP FIR Filter using EP

Node mapping and edge mapping for the given projection vector, processor vector and scheduling vector.

Design 1.

If the projection vector $d = \begin{pmatrix} 1 \\ -1 \end{pmatrix}$ Processor vector $p^T = (1\ 1)$

Scheduling vector $s^T = (1\ 0)$

Probability of systolic array

$$p^T d = [1\ 1] \begin{bmatrix} 1 \\ -1 \end{bmatrix} = 0$$

$$s^T d = [1\ 0] \begin{bmatrix} 1 \\ -1 \end{bmatrix} = 1$$

HUE= $1/|s^T d| = 1$

Design 2

When the three vectors are not given, then we go for RIA description. From which we are able to draw RDG. So that for the FIR problem, The standard output RIA description is as follows.

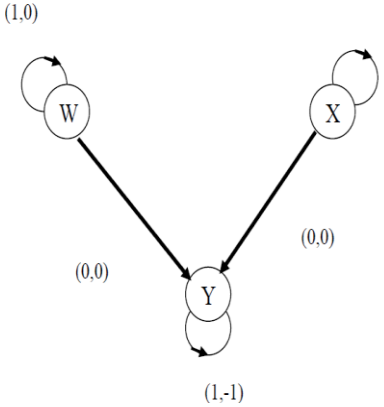
$W(i,j) = W(i-1,j)$

$X(i,j) = X(i,j-1)$

$Y(i,j) = y(i-1,j+1) + W(i,j)X(i,j)$

Using these, RDG of fig 5 can be obtained.

Fig.5 Reduced dependence graph of the FIR filtering example.



Assume the time to perform multiplication, addition and communication are as follows.
Tmult=5 Tadd=2,Tcom=1
Scheduling inequality for an edge in a DG is given by:
$S^T + \gamma_y - \gamma_x \geq T_x$

Where $s = \begin{bmatrix} s1 \\ s2 \end{bmatrix}$

There are 5 edges in the RDG shown in Fig.5

$W \rightarrow Y : e = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$, $\gamma_y - \gamma_x \geq 0$

$X \rightarrow X : e = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$, $s_2 + \gamma_x - \gamma_x \geq 1$

$W \rightarrow W : e = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$, $s1 + \gamma_w - \gamma_w \geq 1$

$X \rightarrow Y : e = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$, $\gamma_y - \gamma_x \geq 0$

$Y \rightarrow Y : e = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$, $s_1 - s_2 + \gamma_y - \gamma_y \geq 5+2+1$

V. Conclusion

In this proposed system using Evolutionary Programming, we are able to find the number of different solutions for designing the systolic architecture for regular iterative algorithms. From which the user can choose any design of his choice according to his application. When the projection vector, processor vector and scheduling vector are given, then we are able to find the node mapping and edge mapping. When the vectors are not known we are using EP to find different design patterns, where in the user can choose the design depending upon his application.

.

VI. Future Work

Once the systolic array is designed, then further we have to go for retiming, folding, unfolding and scheduling where:

Retiming: Is the technique of moving the structural location of latches or registers in a digital circuit to improve its performance, area, and/or power characteristics in such a way that preserves its functional behavior at its outputs.

Folding and unfolding: We show how new connections in a systolic array can be established by folding the array along a line; and how computations in one-dimensional arrays unroll to two-dimensional structures resemble.

Scheduling: Is to implement an algorithm this way, we need a mapping procedure to map a set of equations, which describe the algorithm, to the systolic array. This mapping consists of scheduling (i.e. time mapping, mapping of each DG node to a particular time instant) and space mapping (mapping of each DG node to a systolic array cell).

References

[1]. G. S. Almasi and A. Gottlieb. Highly Parallel Computing Benjamin/Cummings, second edition, 1994
[2]. P.Banerjee. "Parallel Algorithmsfor VLSI Computer-Aided Design" Prentice-Hall, 1994.
[3]. R. Duncan. "Parallel computer architectures". In Advances in Computers, volume 34, pages 113-152. Academic Press, 1992.
[4]. VLSI Digital Signalling Processing Systolic Architecture Design-Lan- Da Van Phd. Dept.of Computer Science. National Chiao Tung University. Taiwan.R.O.C.Spring, 2007.
[5]. Evolutionary Programming:"Evolutionary and adaptive computing in Engineeering Design".-Ian.C.Parmee