Configurable Hardware: Two Case Studies of Micro-Grain Computation

TOM KEAN AND JOHN GRAY

Algotronix Ltd., TTC, Kings Buildings, Mayfield Road, Edinburgh EH9 3JL, Scotland

Received March 26, 1990, Revised May 31, 1990.

Abstract. This paper describes a new VLSI architecture—Configurable Array Logic (CAL) which, at its lowest level, can be programmed electrically to implement any circuit composed of logic gates. At higher levels the technology provides a medium for the direct implementation of algorithms. It particularly addresses systolic and cellular automaton algorithms where the basic computational elements perform computations unsuited to conventional processors.

1. Introduction

In their seminal paper [1] Foster and Kung argued for a computer architecture based around the classic von Neumann processor and memory but with a number of special purpose VLSI chips added to the bus. These chips would implement systolic algorithms to provide high-performance computation of important functions such as pattern matching, fast Fourier transform and sorting. Foster and Kung describe a methodology for the implementation of such chips based on careful algorithm design and simplified and formalized layout techniques. Despite the considerable potential performance advantages of this architecture it has not been successfully adopted in any common computer design to date. Commercial designers have not been able to justify the large design cost of designing many special purpose chips, all of which would require different support from applications programs in order to function effectively.

In this paper we will present an alternative approach to the implementation of systolic algorithms within a conventional computer system: instead of a number of special purpose systolic chips a single configurable computing surface is provided. This architecture is termed configurable logic because it is best viewed as a reconfigurable hardware implementation style: that is, algorithms are programmed by specifying connections between active logic elements (via a programmable switching structure) rather than as instruction data to be interpreted by processing units. The advantage of this is that the programmable structure can implement bit level systolic algorithms unsuited to arrays of conventional processors.

This approach is illustrated by direct conversion of two well known systolic algorithms previously implemented in special purpose silicon to the programmable structure. The ease with which this transformation can be done suggests that the programmable structure is suitable for a very general purpose systolic coprocessor.

2. Architecture

The basic configurable logic structure is a rectangular array of identical cells with the same orientation and nearest neighbor connections, see figure 1. In reality, 1024 cell CAL chips are interconnected as in figure 2. Each cell has a simple function unit and a permutation network. Nonlocal connections must be routed through intermediate cells. A range of such architectures is

Fig. 1. Basic structure. Reproduced from "Systolic Array Processors" by McCanny et al. with kind permission of Prentice Hall.
possible according to the complexity of the function unit and routing network within each cell. The regularity of these structures makes them very suitable for implementation in VLSI. They have the important properties of having a single resource (cells) and infinite expandability since large arrays can be built up by connecting multiple chips at the board level.

2.1. Design Considerations

A member of the class of cellular systems described above can be characterized completely by the design of the basic cell. This is composed of three resources: a function block or kernel which provides the computation and two kinds of routing. These are inter-cell routing which routes inputs from other cells or the function block output to adjacent cells and intra-cell routing which routes inputs from other cells to the function unit inputs. Each of these resources has associated with it a control store. A good measure of a cell's complexity is the logarithm (to base 2) of the number of functionally distinct cell configurations. This is the lower bound on the number of bits of RAM required in the cells control store. We will call this the complexity index (CI).

Early work in this area [2] focused on low-generality arrays capable of synthesizing arbitrary combinational logic functions of a set of input variables. Given that inputs and outputs occur on fixed sides of the array only very simple routing functions are required. Concentration on this problem led to the employment of metrics such as Shoup's Cell Overhead Factor (COF) [3], defined as the ratio of the total number of components in the cell to the number in the cell function unit as measures of efficiency, and to cell designs, such as the cutpoint array [2], in which nearly all the area is devoted to the function unit. The cutpoint array cell has a CI of 3 (it needs 6 distinct logic functions and has fixed routing). The design presented in this paper has a CI of 17; about 40% of cell area is used by inter-cell routing, 20% by intra-cell routing and 40% by cell function. This change in the allocation of cell area to resources is motivated by several considerations.

**Improvements in processing technology.** These have changed the cost analysis in two ways: firstly because memory cells have scaled better than logic higher generality arrays are now more practical, secondly the number of cells that can be placed on a single chip is now so great that it makes sense to implement whole systems rather than just single combinational blocks within the array. This implies that the ease of implementation of routing areas between logic blocks, the ability to rotate and mirror subunits to get a squarer floorplan and flexibility of aspect ratio and port placement in logic blocks are all important factors in obtaining efficient cellular systems.

**Experience with VLSI design.** This has led to an appreciation that routing area within logic blocks is not an overhead but a computational resource. As an illustration the PLA and standard cell implementations of many systems require approximately the same space. In the PLA all the area is used by logic devices whereas in the standard cell implementation there are many fewer logic gates and half the area may be taken up by wiring channels. The extra wires have allowed a more general network of gates resulting in less recomputation of sub-results.

**Ease of Use.** It was decided to support arbitrary interconnection of function units to allow the processes of logic synthesis and placement and routing to be separated (without a general routing network the logic synthesis step must take into account routing considerations to ensure its design can be realized). This separation can provide a major simplification of the design process whether it is done manually or automatically.

2.2. Function Unit Design

The simplest function units capable of general computation when connected together will have two one-bit wide