PROCEEDINGS

OF THE

1980 INTERNATIONAL CONFERENCE

ON

PARALLEL PROCESSING

Cosponsored by the

Ohio State University
Columbus, Ohio

and the

IEEE Computer Society

IEEE Catalog No. 80CH1569–3

VASTOR: A Microprocessor Based Associative Vector Processor for Small Scale Applications*

W.M. Loucks, W.M. Snelgrove and S.G. Zaky

Dept. of Electrical Engineering,
University of Toronto

ABSTRACT

A word-parallel, bit-serial associative processor built around an array of 1-bit wide microprocessors is introduced. It is intended as a low-cost auxiliary processor in small scale computer systems. Data are organized in an array of fixed number of elements, variable word-length vectors. Processing proceeds in parallel on all elements of a vector. Information about the location and word-length of these vectors is stored in a small general-purpose computer which is used to control the storage and processing array.

I. INTRODUCTION

The parallel processing capabilities of an associative processor are highly attractive in many non-numeric applications. Operations such as searching and sorting are inherently parallel in nature, since they may be regarded as a sequence of basic operations such as compare, shift, and mark performed in parallel on a large number of operands. Many organizations have been proposed for associative processors [8, 10]. Of these, the word-parallel, bit-serial, or vertical [9], organization has received considerable attention. This is due to the fact that the bit-serial organization leads to a considerable simplification of the hardware in comparison with fully parallel schemes.

Because of the hardware intensive nature of associative processors, they tend to be economically viable only in large, high capital cost systems. The purpose of this paper is to introduce an associative processor that is meant for relatively small applications. It is based on an array of commercially available 1-bit wide microprocessors. Machine organization is word-parallel, bit-serial. Data is stored and processed in the form of vectors consisting of a fixed number of elements. The machine has been dubbed VASTOR for Vector Associative Store TO-Ronto.

VASTOR is intended as a special purpose processor to be attached to a conventional mini-computer system. In what follows, the minicomputer will be referred to as the host. In such a system, VASTOR would handle those parts of the work load that can benefit from its associative and vector capabilities. Use of associative processors in this manner has been suggested by many authors, e.g. [5]. Also many potential applications have been studied [3]. The main feature of VASTOR is that it represents an associative structure and its implementation that are economically viable in a minicomputer system environment. A prototype processor has been constructed and tested.

The main constraints in the design of VASTOR were low cost and modularity. This required that readily available components be used, that internal communication and control be kept simple, and that VASTOR should not overload the computer to which it is attached. Modularity also meant that backplane interconnections between modules should be kept simple and easily expandable.

The VASTOR processor, figure 1, consists of two main components, namely the processing array and the controller. The processing array contains all the storage and processing elements of VASTOR. The controller translates high level commands received from a scalar machine -the host- into sequences of control signals for the processing array. This paper presents a practical implementation of the array and its controller, and describes input/output transfers between the array and the host computer. Algorithms that may be implemented on vector oriented machines such as VASTOR are readily found in the literature [2, 3 and 7].

II. MACHINE STRUCTURE

The organization of the VASTOR array is illustrated in figures 2 and 3. The storage section in the array is an n-word memory, with a word length of several kilobits. Operations are performed on vectors of data elements, figure 2, when the elements of a given vector occupy the same bit positions in all words. While the number of bits per element is the same for all elements of a given vector, it may vary from one vector to another. A 1-bit wide processing element PE is a part of every word. Shift-register SH provides the main mechanism for data transfer

---

* This work was partially supported by the Natural Sciences and Engineering Research Council of Canada under research grant #A8994