A cellular computer to implement the Kalman filter algorithm
by Lynn Elliot Cannon

A thesis submitted to the Graduate Faculty in partial fulfillment of the requirements for the degree of
DOCTOR OF PHILOSOPHY in Electrical Engineering
Montana State University

Abstract:
The subject of this thesis is the development of the design for a specially-organized, general-purpose computer which performs matrix operations efficiently.

The content of the thesis is summarized as follows: First, a review of the relevant work which has been done with microcellular and macrocellular techniques is made, Second, the discrete Kalman filter is described as an example of the type of problem for which this computer is efficient. Third, a detailed design for a cellular, array-structured computer is presented. Fourth, a computer program which simulates the cellular computer is described. Fifth, the recommendation is made that one cell and the associated control circuits be constructed to determine the feasibility of producing a hardware realization of the entire computer.

A CELLULAR COMPUTER TO IMPLEMENT THE KALMAN FILTER ALGORITHM

by

LYNN ELLIOT CANNON

A thesis submitted to the Graduate Faculty in partial fulfillment of the requirements for the degree

of

DOCTOR OF PHILOSOPHY

in

Electrical Engineering

Approved:

Head, Major Department

Chairman, Examining Committee

Graduate Dean

MONTANA STATE UNIVERSITY
Bozeman, Montana

August, 1969

ACKNOWLEDGMENT

The author wishes to thank Professor R. C. Minnick for his encouragement and helpful suggestions during the course of this graduate work and thesis research.

The financial support of the graduate work provided by the Electrical Engineering Department and by a National Science Foundation Traineeship is gratefully acknowledged, as is the thesis support furnished by the Office of Naval Research Contract No. N00014-67-C-0477.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

The subject of this thesis is the development of the design for a specially-organized, general-purpose computer which performs matrix operations efficiently.

The content of the thesis is summarized as follows: First, a review of the relevant work which has been done with microcellular and macrocellular techniques is made. Second, the discrete Kalman filter is described as an example of the type of problem for which this computer is efficient. Third, a detailed design for a cellular, array-structured computer is presented. Fourth, a computer program which simulates the cellular computer is described. Fifth, the recommendation is made that one cell and the associated control circuits be constructed to determine the feasibility of producing a hardware realization of the entire computer.

Chapter 1

INTRODUCTION AND REVIEW OF CELLULAR RESEARCH

1.1 Introduction

Designers of computers are often confronted with the development of a computer for a particular application such as vehicle navigation, signal processing or the like. Many of these applications require that computations be performed as rapidly as possible and that the computer be as small as possible. In order to meet the requirements for speed, cost and physical size, the designer will sometimes resort to the development of a special purpose computer. The disadvantages of this approach are the large initial design costs and the possible necessity of a complete redesign to account for a change in the computation algorithm. On the other hand, general purpose computers, while they do not have to be redesigned to account for algorithm changes, do have disadvantages. Those general purpose computers which are fast enough for some applications are usually not small enough; and those which are small enough are usually not fast enough or are too expensive. These arguments suggest that some thought should be given to the design of general purpose computers which are capable of performing certain types of operations efficiently. Although efficiency may be gained by using faster components, this method is usually expensive and leads to other complications, such as poor reliability. Another method of obtaining greater efficiency for certain operations is to build a computer which takes advantage of the structure inherent in these operations. This type of machine can be called a specially organized, general purpose computer. A computer of this type could be highly efficient for some

types of operations as is a special purpose computer, but would be easy to adapt to changes in computation algorithms as is a general purpose computer.

One class of problems whose inherent structure may be incorporated into the design of a computer is that class which deals with operations on matrices and vectors. An example of problems of this type is the discrete Kalman filter.[3] For certain applications of the Kalman filter many operations must be performed on matrices as rapidly as possible. Thus, the basis for this research is to develop a design for a small specially organized, general purpose computer which is particularly efficient for performing matrix operations.

Incorporating matrix operations into the structure of a computer suggests the use of arrays of logic elements. Therefore, it is natural to consider some of the work which has been done with arrays of such elements.

Arrays of similar or identical logic circuits together with some interconnection structure constitute cellular arrays. If the cells have a low complexity, the arrays of these cells are referred to as microcellular. If not, they are referred to as macrocellular. Low-complexity cells contain no more than a few gates. In either case, the logic circuits, or cells as they are called, are usually arranged in some rectangular fashion and the interconnections between the cells usually forms a uniform pattern.

In the next section some of the work which has been done on micro-

cellular arrays will be discussed, and in the following section some of the pertinent work on macrocellular arrays will be considered.

1.2 Survey of Microcellular Work

One of the earliest arrays which used identical cells and a uniform interconnection was originated in 1961 by Brooking[4,8] at the Air Force Cambridge Research Laboratories (AFCRL). The cells in this array are eight-input NOR gates with each input of a cell being connected to the output of one of the eight neighboring cells in the array. The array is known as an eight-neighbor array. Each cell in the array has inputs from outside the array which are used to provide an electrical disconnect for any of the eight inputs. Several variations of the eight-neighbor arrays were produced at AFCRL by Giusti.

Methods for the logical design of eight-neighbor cellular arrays of NAND cells were developed by Spandorfer and Murphy[21] in 1963. These workers also developed methods of avoiding faulty cells in an array.

Much work has been done on arrays whose cells may perform any one of several different functions. In 1962, Maitra[11] considered one-dimensional arrays in which each cell can be any of the 16 functions of two variables. Such an array is called a Maitra cascade. Additional work on Maitra cascades eventually led to Minnick's cutpoint array[16] in 1964. This array contains cells which can produce any of eight combinational switching functions or an S-R flip-flop. Four parameters

specify the function to be produced for each cell. In a later array by Minnick[15], called the cobweb array, cell parameters specify the interconnection as well as the cell function. Techniques have been developed for the logical design of both cutpoint and cobweb arrays[14-16].

For a much more complete survey of the development of microcellular work the reader is referred to reference (10).

1.3 Survey of Macrocellular Work

Arrays of large or complex cells are known as macrocellular arrays. Most computers which have been designed with an array structure use some form of a macrocellular array. In this section, some of the work which has been done in the design of array structured computers is described.

1.3.1 Unger's Machine

In 1958 Unger[23] described a stored-program computer which was specially organized for pattern detection. The computer consists of a master control and a rectangular array of logical modules as shown in Figure 1.1. The master control contains a clock, decoding circuits and a random access memory. It accesses instructions from memory, decodes them and issues commands which go simultaneously to all of the modules.

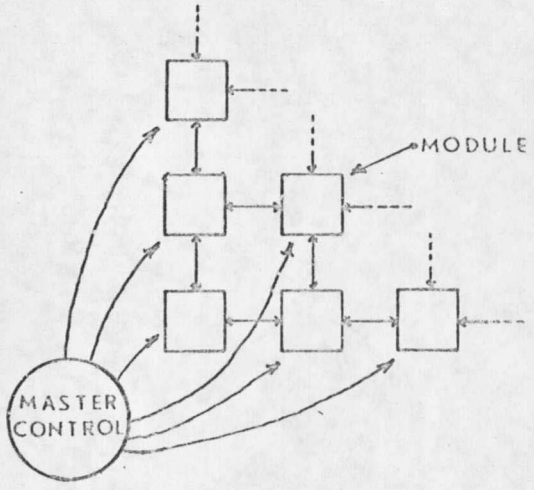

Figure 1.1. Unger's Machine

Each module (or macro-cell in the nomenclature of Section 1.1) contains a one-bit accumulator, six one-bit words of random access memory and associated logic. Each has inputs from the master control and from the accumulators of its four immediate neighbors. Each accumulator also has an input from outside the machine.

A logical OR circuit with inputs from each of the accumulators informs the master control if all of the accumulators contain zeros. This information is used for a transfer-on-zero command which is the only decision-dependent command used.

Other commands for Unger's machine are shown in Table 1.1. These commands are described in detail in reference (23).

Table 1.1. Command Table for Unger's Machine

Command	Meaning
tr x	Transfer to instruction x.
trz x	Transfer on zero to instruction x.
in	Logical invert (NOT).
add	Logical add (OR).
mpy	Logical multiply (AND).
adm	Logical add to memory (OR).
mpm	Logical multiply to memory (AND).
st	Store.
wr	Write (actually a "LOAD" command).
sL(sR,sU,sD)	Shift left (right, up, down).
Add ref	Add reference.
Link	Link.
exp	Expand.
sA	Shift around.

Inputs to the array can be made in parallel by associating an input device such as a photocell with each module or by using the shift around command.

A typical program for Unger's machine might locate lower-left corners of a pattern. Lower-left corners are located by placing a 1 in all cells (here a cell refers to the accumulator of a module)

which satisfy the conditions: (a) there is a 1 in the cell, (b) there are 1's in the cells immediately above and immediately to the right of the cell, and (c) there are 0's in the cells immediately to the left, immediately below and diagonally adjacent to the lower left. The shaded cell of Figure 1.2 satisfies these conditions.

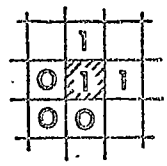

Figure 1.2. A Lower-Left Corner.

In order to illustrate Unger's machine, a program to find lower-left corners is given below. The first command of this program loads the original pattern into the a-register of all cells. In the following steps, the operands R and U refer to the a-register of adjacent cells to the right and above the cell in question, respectively. In Figure 1.3 an example set of data are shown in order to make the execution of the program more clear. From an examination of Figure 1.3(a) it is clear that cells (5,2) and (4,5) are the only two lower-left corners.