EXPERIENCE WITH A MAP 300 ARRAY PROCESSOR

by

P. Nesfield
SACLANT ASW Research Centre
La Spezia, Italy

ABSTRACT The paper is a summary of the experience collected during almost two years of programming a CSPI MAP 300 array processor in signal processing applications. After a short overview of the most interesting features of SNAP 2, the MAP's signal processing language, the paper illustrates their value to the programmer by describing their use in the implementation of a typical signal processing algorithm. The paper also points out the most frequent sources of problems in MAP programming.

INTRODUCTION

This paper is a collection of thoughts concerning the MAP 300 and how it has performed over the last 18 months. First it must be said that any complex digital system must have the confidence of its users, a trust must be built-up such, that when some algorithm fails to perform as expected, or fails completely, the person debugging the code does not suspect everything, starting with the hardware. The problem also occurs when a small change is required, when lacking in trust too often a programmer will not dare touch the code, or breaks-out in a sweat as he imagines the side effects a seemingly harmless addition can have. One early installation of a MAP 300 in a government laboratory had a large logo saying "DANGER MAP 300's can damage your health".

In our opinion the MAP 300 has quickly accummulated that quantum of confidence that is required to pinpoint problems without recourse to a fortuneteller. The hardware is extremely reliable, during the last three sea trials we had more problems with the host CPU and disc (which is an HP and generally reckoned to be very reliable). The software, which is the user's point of contact with the device is extremely ambitious and works very well. Problems mainly concern the executive error handling, which is unfortunate since new users are apt to make simple errors that have a disastrous effect on the system. Error handling has been greatly improved with release 3.5 of the Executive.

We have never had any intermittent problems on our MAPs, every crash is repeatable and the reason entirely logical. I suspect that the government laboratory has now taken down their logo.

1. HARDWARE MAP 300

The array processor [Fig. 1(a) and 1(b)] consists of:

(a) A CSPU, which is a 16-bit fixed-point mini-computer with vector interrupt stack and block move instructions. It can address to the 16-bit (half-word level) on all three memory buses using 18-bit addresses, 512 Kbytes per bus. The CSPU has no input/output facilities as such.

(b) A floating-point arithmetic processor APU (two for a MAP 300) with an internal program memory of 128 words which is loaded by the CSPU through pseudo-memory locations. The APU can access data from its registers and from input/output FIFO's. The cycle time is 70 nsec and an operation may take up to 6 cycles for a multiply. Instructions are started sequentially, and hardware tests for resource availability freezing the APU if the current operation uses a register involved in a prior operation. In this way adds and moves can be hidden behind multiplies.

(c) An APS arithmetic processor sequencer fills and empties the read and write data FIFO's of the APU. It has a 128 x 25 bit program memory which can be loaded by the CSPU. The program of the APS must be written in conjunction with the APU to generate the input and output data sequences required for the particular algorithm. The APS thus removes the burden of main memory access from the APU which can then concentrate on arithmetic. The two processors cycle asynchronously and independently.

(d) MOS 500 and 160 nanosecond 32-bit wide memory modules addressable to the half-word level. The memory buses operate independently and so processors may access memory simultaneously on different buses. The controllers have eleven ports and a "polite priority" access scheme to arbitrate simultaneous accesses on the same bus.

(e) Input/output scrolls are programmable processors to transfer data between MAP memory and external devices. Program memory size and transfer capabilities vary between models, models 2 and 3 which we have at the Centre have a nominal transfer rate of 1.4 mHz, 16/32-bit words, with enough programming power to demultiplex, and make data-dependent decisions. Scrolls can interrupt the CSPU. The host computer is connected to a scroll model 3 called a Host Interface Scroll, which has the added ability to load its program memory from the host computer and convert to and from the different formats of the host and MAP.

The floating point format is IBM hexadecimal signed magnitude. Data is normalized when the most significant hex digit is non-zero.

2. SOFTWARE

CSPI supplies four software modules:-

Host Operating System Driver.

Host Resident Library of SNAP 2 Interfaces.

MAP Executive, Array Functions and I/O Modules.

MAP Cross Assembler and Simulator.

Using these modules the user can now (September 1979) get a MAP system running on a HP 21MX RTE-IV Host very easily and, without lavishing praise on CSPI's resident HP programmers, the documentation is superb. Six test programs are supplied, plus interface diagnostics to check the installation.

Due to the peculiar overlay structure of the HP, the assembler and the simulator cannot be loaded on an HP without some adaptation. This is promised by CSPI in the very near future. However, they may be loaded on Digital Equipment and Data General minicomputers and on any large main frame; at the Centre we use the assembler on the UNIVAC 1106.

2.1 Programming the MAP in Assembler

CSPI publish a document called "It's Simple to Program MAP" which sounds like sarcasm; in fact, the publication should be re-named "Simple MAP Programming" for, in truth, complexity of programming is proportional to the complexity of the program. All processors in the MAP are programmed in Assembler, and as assemblers or instructions sets go, the MAP is much like a PDP-11, with more instructions — 150 CSPU, 40 APU, 15 APS. The multiple programs for multiple processors neatly partition the problem.

2.2 Programming the MAP using SNAP 2

SNAP 2 has some special features:

(a) Buffers or arrays may be defined as non-contiguous in memory or any length with any spacing between samples both positive and negative. Four formats are permissible 8-bit bytes, 16-bit words, integers or 16 and 32-bit real numbers.

(b) Data may be loaded into and out of memory independant of and without interference to the arithmetic functions.

(c) Array functions have a 3 (or more) address format

$$z = f(x,y)$$

In most cases the array can be overlayed.

(d) Function Lists: Groups of functions can be grouped in lists and stored in bus 1 memory. These lists can be executed by one command from the host, and the application programmer's goal is to code all the SNAP 2 functions in lists controlling them only with data availability flags. This is a realistic goal with functions as powerful and flexible as in SNAP 2, and runs the MAP at full speed.

3. TYPICAL APPLICATION

A requirement of Mr Ph. de Heering of the Applied Sonar Group was to generate spreading functions, the operations for which are:

(1) Acquire 2K of data.

(2) Match filter (correlation process).

(3) Select data window (128 points).

(4) Hilbert transform to obtain an estimate of imaginary part of data.

(5) Repeat steps 1 to 4 for 128 arrivals.

(6) Transpose matrix.

(7) Calculate power spectrum.

(8) Display and store result [Figs 2 and 3 show typical spreading function].

The initial version worked in real-time, and with one second between acquisitions the MAP was not stretched. Now the data is read from digital tape (which is a slow 37.5 ips unit) and works at three times real-time.

The specific features of the MAP used for this application are:-

(a) Short floating point format (16 bit) adopted to enable two 16K data area in 64 Kbytes of memory.

(b) Matrix transposition by re-definition of data area; no data is moved.

(c) Independent I/O processing using two 16K data areas; area A being filled with data while area B is having spectra calculated and output.

(d) MAP processing entirely controlled by function lists; no host control required, the host manages I/O only and displays.

3.1 Causes of Problems

The simplest problem causing the most disastrous consequences to user programs written in SNAP 2 is buffer definitions. The confusion is caused by:-

(a) Overlaying two buffers such that the executive resource allocation fails to guard against multiple access.

(b) Incorrect definition of buffer type (using a complex buffer for a real data function, etc.).

(c) Signal processing algorithm design errors, difficult to correct because the final result is the only indication of the problem; the process cannot be stepped.

(d) Buffer synchronization between the host computer and the MAP, the two processors obviously have to co-operate otherwise the system will hang-up. The data traffic is the weak-link in the array processor concept.

(e) Then there is the age-old problem of speed; despite it being one of the fastest commercial array processors on the market, most tasks could use more speed.

To improve on (a), (b), (c) and (d), special software called AP-M and STROBE have been developed at SACLANTCEN.

CONCLUSIONS

The MAP 300 has been a valuable addition to the resources of SACLANTCEN. Sea trials involving the MAP as a signal processor have shown that the MAP is not just a delicate laboratory ornament but a hardy reliable tool.

DISCUSSION

J.M. Griffin Is integer arithmetic available in the MAP.

P. Nesfield Yes, with care but it is not faster and the MAP is a floating point unit.

J.M. Griffin Is there any time advantage to the 16-bit floating mode.

P. Nesfield No advantage.

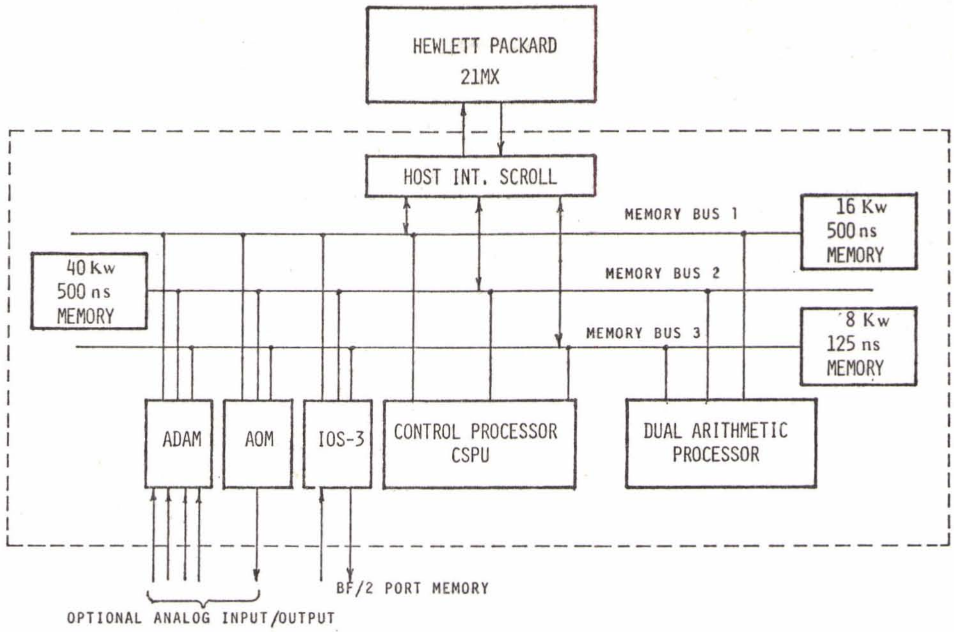

FIG. 1a

SIMPLIFIED DIAGRAM OF MAP300 ARITHMETIC UNIT

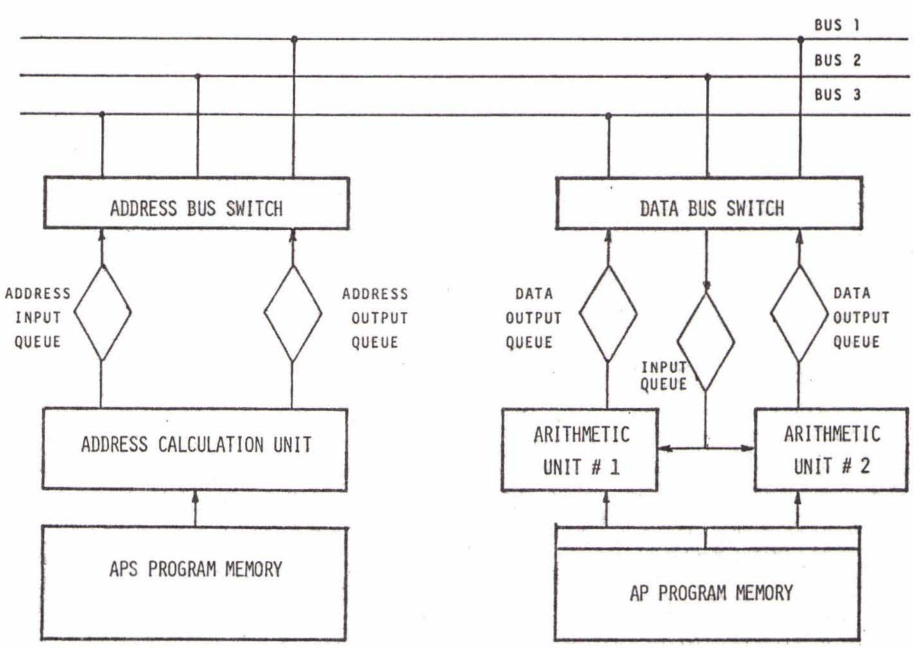

FIG. 1b

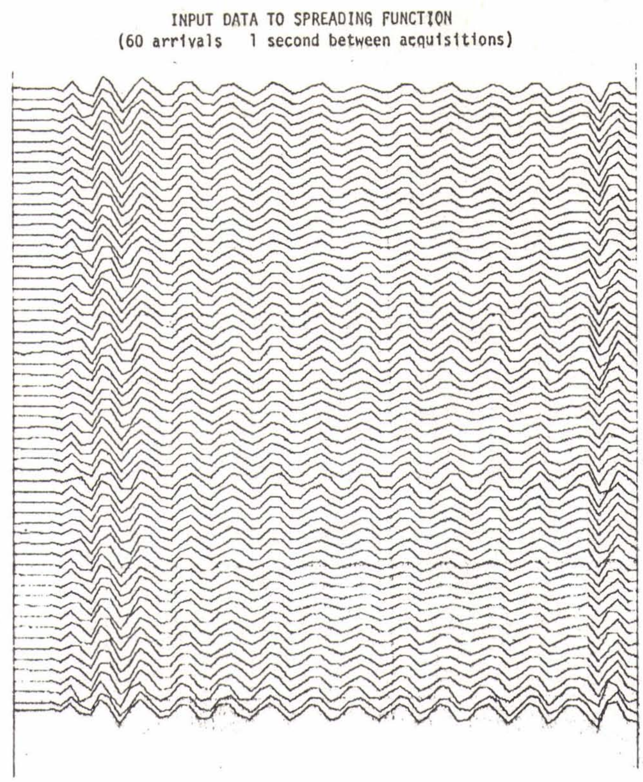
INPUT DATA TO SPREADING FUNCTION
(60 arrivals 1 second between acquisitions)

FIG. 2

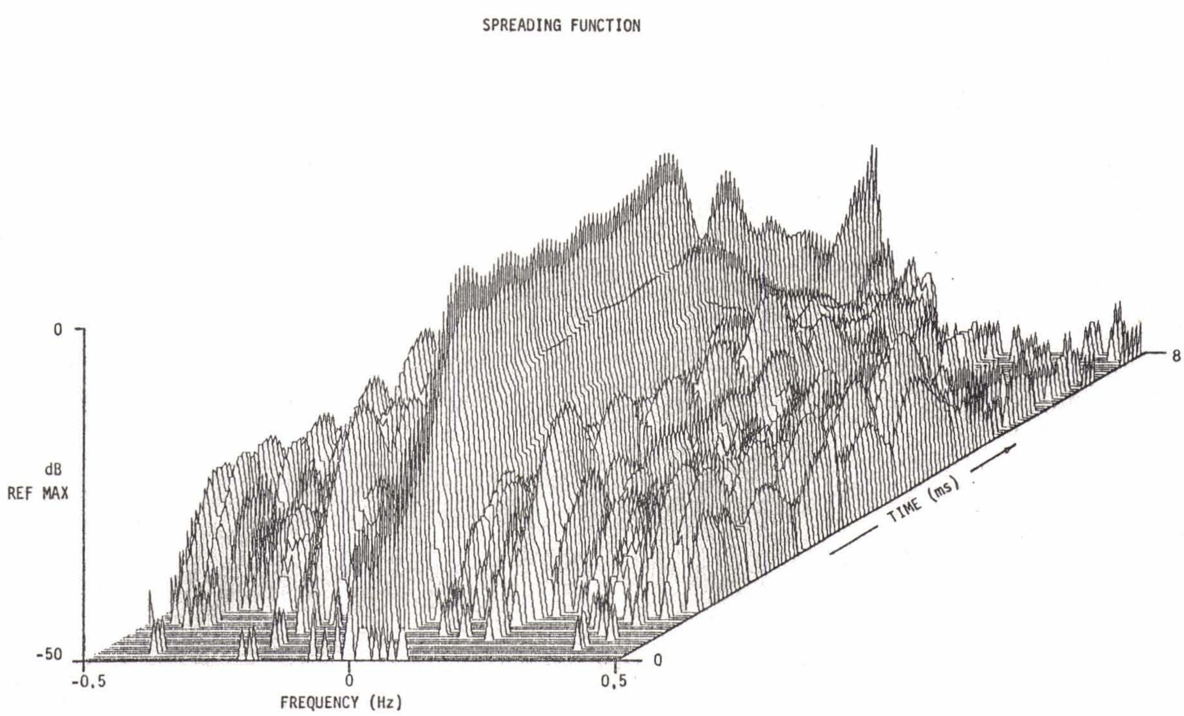
SPREADING FUNCTION
0
dB
REF MAX
-50
-0.5
0
0.5
FREQUENCY (Hz)
0
TIME (ms)
8

FIG. 3