PALO ALTO SCIENTIFIC CENTER

IBM
Data Processing Division

ZZ20-6426

May 1974

AN ARCHITECTURAL AND DESIGN OVERVIEW OF SCAMP

Patrick Smith and Kitty Price

IBM CONFIDENTIAL

IBM PALO ALTO SCIENTIFIC CENTER TECHNICAL REPORT NO.

May 1974

AN ARCHITECTURAL AND DESIGN OVERVIEW OF SCAMP

Patrick Smith

and

Kitty Price

IBM CORPORATION

PALO ALTO SCIENTIFIC CENTER

2670 HANOVER STREET

PALO ALTO, CALIFORNIA 94304

TABLE OF CONTENTS

TABLE OF CONTENTS

ABSTRACT:

This report describes architectural considerations for the prototype design of an APL machine called SCAMP. Also covered are key software design decisions plus some possible I/O extensions to the existing SCAMP architecture.

Index Terms for the IBM Subject Index

Emulation	03 - Communications
Microprogramming	06 - Computer Peripheral
APL	07 - Computers
	21 - Programming

I. INTRODUCTION

The purpose of this report is to present a summary of the architectural and design considerations for SCAMP, a small standalone APL (1) machine. Specifically this report is concerned with the SCAMP prototype model.

The SCAMP architecture evolved as the result of an investigation to determine the feasibility of developing a small programmable super-calculator subject to four major requirements. The requirements were: (1) It should be portable; (2) The interface presented to the customer should be APL- like; (3) Pricing should be competitive with other companies selling similar machines; and (4) Availability of SCAMP to the customer should occur approximately before 1975. As a result of the investigation, a prototype model was built and satisfactorily demonstrated in August of 1973.

II. DERIVATION OF THE SCAMP SYSTEM

A. Historical Perspective

SCAMP is a standalone portable terminal with significant stored program capability, an interactive high level language user interface, and the ability to communicate with a larger host system. SCAMP is a logical extension to the limited capability of programmable calculators presently available. The SCAMP design represents the coming together of several of the following current computing systems trends.

-Large Scale Integration (LSI)

LSI is a technological revolution which has permitted the placing of large portions of computer logic on small silicon chips at significantly reduced costs.

-Minicomputers/Microprocessors

As the price for computer logic has decreased, new uses have appeared that were previously not economically feasible. One of the first uses of inexpensive hardware logic was in the area of small standalone computers called minicomputers. The minicomputer represents an end product which can be programmed to accomplish a desired application. As hardware costs continued to decrease the 'processor on a board/chip' or microprocessor made its appearance. The microprocessor is normally microprogrammed to perform some function that represents part of a bigger whole. Unlike the minicomputer, the microprocessor is not normally the end product, but only a part of the end product or function to be achieved.

-Interactive Computing

One of the strongest trends in computing today is the increased use of interactive terminal systems in which the user interacts directly with the computer through a terminal. Progress on application programs can often be made in hours instead of days by the normal batch turnaround method.

-Data Based Systems
Because of the increased emphasis on terminal oriented systems, there has been a corresponding trend toward using large host resident common data bases such as insurance records and credit references.

-Computer Networks
Computer Networks are special types of communication-based systems in which the terminals are also computers. The most successful trend is toward the use of small computers which are connected to the network only occasionally as intelligent terminals.

-Intelligent Terminals
Intelligent Terminals represent the coming together of Microprocessors, Computing Networks, and Data Based Systems. Terminals are provided processing capability which allows them to do significant amounts of work on a standalone basis. If there is a need to access a large data base, or to achieve a higher performance level than that which is possible on the terminal, a large central processor is accessed via a teleprocessing link. See references 3 and 4 for extended discussion of the Intelligent Terminal concept.

-High Level Languages
There is a continuing trend toward supplying the user with more powerful programming languages such as APL to make his programming task easier. APL permits the user to perform complex calculations with little required knowledge of the computer system.

-More Capable Man/Machine Interface
There is an ongoing effort to provide the user with a better way of interacting with the computer, and several of the above mentioned trends reflect this. The increased use of teleprocessing has allowed the user to interact with the computer through remotely located terminals. Displays free the user from waiting for typewriter printer output, plus they provide a better environment for the use of graphics.

B. Initial SCAMP System Design

The SCAMP prototype system evolved from an initial system design which is shown in figure 1. This section describes the major factors which influenced the initial systems design.

1) Traditional APL System Requirements

APL is a programming language which allows the user to interact directly with a computer through a terminal, and manipulate functions and data contained in a user area called a workspace. APL traditionally requires the following system resources:

a) An input device for entering APL functions and data into the system;

b) An output device to reflect user input plus the results of any calculations which take place;

c) A processing unit with attached high speed memory buffer (i.e. Main memory) containing portions of the APL systems code and portions of the active workspace;

d) A somewhat slower bulk storage which contains all the APL systems code and various user workspaces.

In the APL/360 implementation, the input device is a terminal keyboard while the output device is a terminal printer; disks are used for bulk storage.

2) Incorporation Of Modern Systems Capability

a) Display Orientation

The use of a multiline display provides SCAMP with a modern man/machine interface. The multiline display greatly enhances the user/system interaction, increasing both speed and information transfer. The need for hardcopy output still exists of course, and

the SCAMP design includes an optional printer for this purpose.

b) Intelligent Terminal Capability

SCAMP contains a teleprocessing adapter which allows it to communicate with a host computer and thus be used as an intelligent terminal. This adapter also allows SCAMP to be used as a non-intelligent terminal (e.g. 2740, 2741, Etc.) with the appropriate emulator code.

C. SCAMP Prototype System Design

The prototype system design shown in Figure 2 resulted from several hardware and software modifications to the initial system design. The development of an operating SCAMP prototype system was undertaken to demonstrate the concept and feasibility of a potential SCAMP product line. To achieve a working prototype as quickly as possible, maximum use was made of existing hardware and software modules.

1) SCAMP Prototype Hardware

a) Portable Bulk Storage Device

Although many disk technologies were investigated (e.g. IGAR, RUBY, ARIES, OEM), none were found which satisfied the SCAMP portability requirement.

The need for offline storage and retrieval was satisfied by the use of the same inexpensive cassette recorder which is used on the System 7.

The use of tape rather than disk technology for bulk storage made it impractical to use paging techniques to increase the apparent capacity of main memory. As a result the entire APL interpreter and user active workspace had to be stored in main memory.

b) Processing Unit

Although other IBM microprocessors were evaluated, the GSD Palm processor was selected for the following reasons: ~~the UC 0, or UC.5, its choice took advantage of the following:~~ (1) A PALM processor was available immediately for development purposes whereas other potential processors were not; (2) It is the least expensive processor; (3) Adapters exist for essential pieces of I/O (especially a video display); (4) Palm is considerably smaller than other processors because it reflects the latest technology. Thus PALM met the time, size, and low cost requirements.

c) I/O Units

PALM has a display adapter which supports a conventional raster driven CRT display of 16 rows by 64 characters. The display is refreshed by cycle stealing character strings from main memory and creating the required dot patterns on the fly via a seperate fixed Read Only Storage (ROS) character generator.

Palm has a BAHIA printer adapter. The BAHIA is a low cost matrix printer which can be program controlled to print any pattern desired by sweeping the print mechanism across the paper from either left to right or right to left. The printer is also capable of both forward and reverse line feed under program control. This type of printer capability and freedom makes possible graphic applications. The other mini-processors had more expensive printers without BAHIA's graphic capability.

A Calico keyboard was chosen having a normal APL keyboard layout plus twelve additional keys to be used for any special functions which might be required.

A 134.5 bits per second stop/start teleprocessing adapter was built to support the emulation of a 2741. Although any appropriate line protocol could be supported, that of a 2741 was chosen to provide convenient interaction with existing host computer systems such as APL/360 or TSO.

2) Scamp APL Software

Normally a new APL interpreter would be written to fit the PALM architecture and reflect the latest APL language developments. There were two major objections to this approach. First, the programming effort required for a new APL interpreter varies depending upon the architecture of the target machine. It was estimated that the development effort would require at least 2 years, plus a period of time for adequately

testing the result. Because a working prototype was desired in much less time, a new APL interpreter effort was impractical. Secondly, the power of the PALM instruction set is such that an unreasonable amount of code is required to write an APL interpreter using the PALM instruction set directly. It was therefore decided to use an existing APL Interpreter in its original form, and use the PALM microprocessor to emulate the machine language for the APL interpreter chosen.

The subject of using an emulated intermediate architecture is important enough that it requires further explanation. Figure 3 illustrates how an emulated intermediate architecture can be used to significantly reduce the amount of storage required to accomplish some significantly complex function such as an APL system. In the figure the slopes of the lines reflect qualitatively the power of the instruction sets involved. Since the instruction set of the intermediate architecture is more powerful (i.e. More function per average instruction can be accomplished), its slope is flatter. It should be noted that the intermediate architecture instruction set line has a storage offset equal to the storage used in emulating the intermediate architecture. If the desired function to be accomplished is relatively trivial (e.g. Matrix multiply), it may well be more expensive in terms of storage to utilize an intermediate architecture because of the emulator storage overhead. For completeness it should be noted that if too powerful an intermediate architecture is chosen, the amount of storage required in its emulation may be so great that little room is left to accomplish the desired function. The penalty for using an intermediate architecture is some degradation in system performance due to the extra level of interpretation introduced by an emulator.

A study of the different available APL interpreters indicated the type 3 1130 APL interpreter (2) was the only one which fit in PALM memory.

The PALM processor has the capability of addressing up to 65K bytes of main memory. Figure 4 shows how memory is allocated. The prototype has all read/write memory so that normal development software changes could be made. To reduce product cost, read only storage can be used for the systems software in the released product. In the prototype, the cassette recorder is used not only for saving and restoring APL workspaces, but also for loading the systems microcode when the IPL button is pushed. Input to the APL interpreter is through the keyboard, output is to the display and/or the printer.

D. Future Scamp Systems

The design of SCAMP I is based on the optimum use of existing technologies. Future SCAMP systems will be able to employ much newer technologies and thus provide better price/performance characteristics. This section discusses possible short and long term strategies for achieving this.

1) SCAMP II

As a short term strategy, SCAMP II would reflect the initial SCAMP system design described earlier and would utilize a much faster processor. It would use a disk for holding APL systems code and user workspaces, thus much less main memory would be required. The type III 1130 APL software would not be used; instead another existing APL interpreter with more capability could be selected, or a new interpreter could be written. Furthermore, a full 1130 emulator could be developed to support the execution of a considerable 1130 program library.

2) SCAMP III

The SCAMP III system design would be similar to that of SCAMP II but long term technology improvements would be taken advantage of to reduce both size and cost.

A new intermediate architecture would be defined for SCAMP III and an APL Interpreter written for this architecture. The architecture would be designed to facilitate coding of an APL interpreter, thus leading to a faster and more cost effective machine. To further enhance performance, a significant portion of the intermediate architecture would be directly implemented in new processor hardware.