LBL-8157
UC-32
CONF-780779

NRCC NATIONAL RESOURCE FOR COMPUTATION IN CHEMISTRY

THE MINICOMPUTER AND COMPUTATIONS IN CHEMISTRY

Report
on the Workshop
July 19-21, 1978

LAWRENCE BERKELEY LABORATORY
UNIVERSITY OF CALIFORNIA

Prepared for the U.S. Department of Energy under Contract W-7405-ENG-48 and for the National Science Foundation under Interagency Agreement CHE-7721305

Printed in the United States of America
Available from
National Technical Information Service
U.S. Department of Commerce
5285 Port Royal Road
Springfield, VA 22161
Price Code: A04

PROCEEDINGS

THE MINICOMPUTER AND COMPUTATIONS IN CHEMISTRY

Sponsored by the

NATIONAL RESOURCE FOR COMPUTATION IN CHEMISTRY

Lawrence Berkeley Laboratory

July 19-21, 1978

FOREWORD

The National Resource for Computation in Chemistry (NRCC) was established as a division of Lawrence Berkeley Laboratory (LBL) in October 1977. The functions of the NRCC may be broadly categorized as follows: (1) to make information on existing and developing computational methodologies available to all segments of the chemistry community, (2) to make state-of-the-art computational facilities (hardware and software) accessible to the chemistry community, and (3) to foster research and development of new computational methods for application to chemical problems.

Workshops are planned as an integral part of the NRCC's program. As one of its initial efforts in this direction, the NRCC sponsored the titled workshop to address many of the questions now being raised about minicomputers, their potential impact in chemistry, and the role of the NRCC in this developing area. One outgrowth of this workshop is that the NRCC has initiated the necessary steps to purchase a minicomputer. The reasons behind this planned acquisition are to assist the development of software for minicomputers per se and of portable software for both large and small hardware systems.

The NRCC is indebted to Professor Phillip R. Certain, University of Wisconsin-Madison for his considerable time and effort in organizing this workshop.

The National Resource for Computation in Chemistry is funded jointly by the Department of Energy and the National Science Foundation.

- William A. Lester, Jr.
  Director, NRCC

CONTENTS

INTRODUCTION

Phillip R. Certain
Department of Chemisty
University of Wisconsin-Madison

The minicomputer has long played a valuable role in experimental chemistry, serving to automate data acquisition and experiment control. Until recently, however, the limited word lengths of these computers have generally precluded extensive use for data reduction and theoretical modeling. The word length limits the accuracy with which numbers can be represented, the maximum program size before overlaying is necessary, and the maximum size of data arrays that can be directly addressed.

The introduction of multiple-precision hardware and longer word lengths has now given the minicomputer a much more general potential for chemistry applications. It was the purpose of this workshop to address this potential, particularly as it is related to computations. The workshop brought together persons with minicomputer experience and those who are considering how the minicomputer might enhance their research activities. Participants, who are listed on page, 47, come from university, industrial, and government research laboratories. Both chemists and computer scientists were present, sometimes in the same person.

The workshop sessions were arranged in sequence to address the following questions:

Is the general purpose minicomputer an appropriate tool to meet the computational requirements of a chemistry research laboratory?

What are the procedures for wisely designing a minicomputer configuration?

What special-purpose hardware is available to enhance the speed of a minicomputer?

How does one select the appropriate minicomputer and ensure that it can accomplish the tasks for which it was designed?

How can one network minicomputers for more efficient and flexible operation?

Can one do really large-scale computations on a minicomputer and what modifications are necessary to convert existing programs and algorithms?

How can the minicomputer be used to access the maxicomputers at the NRCC?

How are computers likely to evolve in the future?

What should be the role of the NRCC in relation to minicomputers?

The opening session of the workshop was an overview of the working sessions which considered the above questions. This report of the workshop consists mainly of edited transcripts of the introductory remarks made at the first session. The purpose of the editing was simply to give clarity to the written form of the oral presentations, and it is hoped that the informal character of the originals has been preserved. Since in general the content of the working sessions is adequately represented in available publications, the introductory remarks have been augmented by relevant bibliographies as an alternative to transcription of the entire workshop. Furthermore, the sessions on remote access and the role of attending the NRCC did not lend themselves to transcription, although information on remote access is separately available from the NRCC. In addition to attending the working sessions, the participants had the opportunity to tour the LBL computer center, and to see the real-time systems group at LBL, and the minicomputer belonging to the theoretical research groups of Professors Miller and Schaefer at the University of California-Berkeley.

There was no attempt in the workshop to give final answers to the questions that were raised, since the answers are determined in large part by each particular minicomputer environment. Thus, the session leaders simply raised the issues involved. In particular, in the final session no consensus on the role of the NRCC was attempted. It was generally felt, however, that the NRCC had a role to play in bringing some of the advantages of scale to the operation of minicomputers in chemistry research laboratories. It was suggested, for example, that the NRCC develop benchmark programs for chemistry, run them on the available minicomputers, and distribute the results. It was also suggested that the NRCC perform a tutorial role in transmitting to the chemistry community the expertise on minicomputers which has developed in other communities. While the workshop participants were perhaps not entirely representative of the entire chemistry community, we can conclude that the minicomputer will continue to make an impact on computations in chemistry. With sufficient personnel and hardware, the NRCC can greatly enhance minicomputer effectiveness.

THE MINICOMPUTER AND COMPUTATIONS IN CHEMISTRY

National Resource for Computation in Chemistry
Lawrence Berkeley Laboratory
July 19-21, 1978

PROGRAM

Building 50 Auditorium

Wednesday, 19 June

8:00: Registration

Session 1a. Orientation. 8:30-10:00 A.M.

8:30: Introduction to the Workshop
Phillip R. Certain
University of Wisconsin-Madison

Welcome to the NRCC
William A. Lester
Director, National Resource for Computation in Chemistry

8:45: Advantages of Scale vs. Advantages of Specialization
Harry A. Eick
Michigan State University

9:00: System Design and Specification
Jack W. Frazer
Lawrence Livermore Laboratory

9:20: Array Processors and Multiprocessors
Neil S. Ostlund
University of Arkansas

9:40: Benchmarking and Assessment
Chris T. Corcoran
University of Wisconsin-Madison

Coffee Break 10:00-10:15 A.M.

Session 1b. 10:15-11:30 A.M.

10:15: Distributed Data Processing and Networking
William J. Lennon
Lawrence Livermore Laboratory

10:40: Theoretical Chemistry
Clifford E. Dykstra
University of Illinois-Urbana

11:05: Remote Access of NRCC
Eric R. Beals
Lawrence Berkeley Laboratory

Session 2. Advantages of Scale vs. Advantages of Specialization
1:30-3:15 P.M.
Harry A. Eick, Michigan State University

Break: 3:15-3:30 P.M.

Session 3. System Design and Specification: 3:30-5:30 P.M.
Jack W. Frazer, Lawrence Livermore Laboratory
Sam Perone, Purdue University

Thursday, 20 July

Session 4. Array Processors and Multiprocessors: 8:30-10:15 A.M.
Neil S. Ostlund, University of Arkansas
Alec Grimison, Cornell University
Peter Hibbard, Carnegie-Mellon University

Coffee Break: 10:50-10:30 A.M.

Session 5. Benchmarking and Assessment: 10:30-11:30 A.M.
Chris T. Corcoran, University of Wisconsin-Madison

Session 5. Benchmarking and Assessment (continued) 1:30-2:30 P.M.
Robert Waters, Hanscom Air Force Base

Session X. Laboratory Tours: 2:30 P.M. - 4:30 P.M.
Computer Center and Real Times Systems Group

Friday, 21 July

Session 6. Distributed Data Processing and Networking: 8:30-10:15 A.M.
William J. Lennon, Lawrence Livermore Laboratory
Jerome V. V. Kasper, University of California at Los Angeles
Mark Abramson, Emory University

Coffee Break: 10:15-10:30 A.M.

Session 7. Theoretical Chemistry: 10:15-12:30 P.M.
Clifford E. Dykstra, University of Illinois
Steven Binkley, Carnegie-Mellon University
Thomas A. Weber, Bell Telephone Laboratories
Klaus Wenzel, Ruhr-Universitat Bochum
Brian Ford, Numerical Algorithms Group

Session 8. Remote Access of NRCC: 1:30-2:15 P.M.
Eric R. Beals, Lawrence Berkeley Laboratory

Session 9. Trends in Computer Design: 2:30-4:00 P.M.
Peter Lykos, Illinois Institute of Technology
Ronald Hochsprung, National Semiconductor
Stanley Cohen, Speakeasy Computing Corporation
Gene Small, The Harris Corporation

Session 10. The Role of the NRCC: 4:15-4:45 P.M.
William A. Lester, NRCC

ADVANTAGES OF SCALE VS. ADVANTAGES OF SPECIALIZATION

Harry A. Eick
Department of Chemistry
Michigan State University

Let me first give you an idea of where I come from and where I hope we are going to go. My background is primarily that of a central site administrator in the sense that I have over the years argued for central computing. However, I do think that there is a role for the minicomputer. We have made a number of investigations at Michigan State into minicomputers, and I hope to describe to you what we have learned.

If you look at the literature, you will find that the topic of minicomputers, namely specialization vs. centralization, was fairly extensively studied about 1975-76. The literature indicates that since that time people generally have accepted the midi-, mini-type of system in the computing hierarchy. Thus, I would like to address briefly the advantages of scale and of specialization, and then give my opinion, based on an analysis of several sites, of what is necessary to make midi-operations successful at a given level, such as in a department. Finally, I will discuss some inevitable problems which, since I am an administrator, will be more from the point of view of an institution than an individual department.

First of all, let me give some examples of types of specialization in which the minicomputer can be used effectively. I exclude from consideration real-time operations, since minicomputers are known to work well there. In my opinion, successful specialization requires a coherent group of users who utilize the computer in a similar way. One example is a chemistry department or, more likely, a subset thereof. At Michigan State, I interviewed a large number of users to determine where minicomputers could be efficiently utilized. I found that high-energy physicists would be an ideal group--but they were not interested. Meteorologists who simply collect data from around the state could use a minicomputer in addition to a central site, as could a dairy science operation that keeps track of the milk production from cows throughout the state. Our satellite data reduction system could use a minicomputer. The operation that keeps track of profits and losses for farms around the state has no reason to use a central site. Finally, I am not sure that the freshman chemistry operation would really profit from its own minicomputer, which it would really like to have.

In addition to coherency of utilization, there are language specializations. For instance, languages such as APL, Basic, or SPSS for the social sciences, can often be handled by a subset without tying up the entire system. A different type specialization is functionality. For instance, the pest management system at Michigan State, comprised of engineers, entomologists, biologists, and others, is not coherent in the sense that a department is, but they do have a coherency in the sense of goals. The minicomputer works very well in this environment.

Having given examples of where a minicomputer can work, let me look at some of the motivating factors for considering a minicomputer system. There are apparent economic advantages, such as lower CPU and memory costs.

Also, from the institutional point of view, having many computers may eliminate the need for excess capital outlay at a central site. Speed in adapting to technological advances is probably the most important economic advantage. When technology changes, the smaller computers reap the benefits more rapidly and, consequently, costs go down.

In addition to economic advantages, the minicomputer offers a different philosophy than that of the central site. Often the central site, either through policies of charging or through its operational procedures, is not user oriented. For example, some users like to work at 6:00 in the morning. They like to have a minicomputer because a central system is often down at 6 a.m. for maintenance three days out of the week. Thus, when they want to work, the central system is never up--or, at least, that is the way it seems to them.

Take another example of the impact of central site philosophy on the user. It is desirable and convenient to have some way of processing digitized information from the laboratory directly. If the central site agrees to provide a hookup for a minicomputer, but says it will take about four years, the motivation to go another way becomes very strong. Large systems may be responsive to a group but, typically, they are unresponsive to an individual user. Having control of and access to your own system is certainly a very strong motivating factor.

Finally, I feel that an intangible motivating factor is the political climate. In any institution, the political climate has a great effect on what you do at any time and, having gone through many battles with the upper administration, I am particularly tuned to the fact that you have to keep this in mind.

Let us turn now to some advantages of scale. Clearly management is a predominant factor. One organization runs the systems programming, development, and operations. They can be very good people, who can do the job, and the responsibility is well defined. An operations staff can manage the computer, and manage it very well. There is only one maintenance problem, instead of many problems. I think that, in deciding which way to go, people tend to forget that maintenance is a terrible problem for any computer system that gets appreciable in size. Security problems are minimized at one large site although, if you break the computer up into many sites, you do not have to worry about security.

Professional growth possibilities for the staff are also very important. I am firmly convinced that if you want to run a fairly sophisticated computer operation at a unit level, you must have a professional staff of at least one person. A central site can provide a ladder of professional growth and the opportunity to interact with colleagues in the same field. At a remote site, or at a unit level, keeping the necessary qualified staff is a problem. Some will argue that you do not have to have a staff, but my firm belief is that if you get in the business, you will find that you want a staff.

Smoothing of demand is another argument for a central site; namely, the large number of users do not all use the machine at one time. Disks can be maximized because disk storage is expensive relative to the CPU. The

use of plotters and, with computer output on microfilm, the use of printers can be maximized. The counter argument is that the central site must staff and stock for peak demand, which increases the cost. More powerful hardware (Grosh's law states that computing power is proportional to the square of the cost) is an advantage for the central site, as is the larger word size usually available. The central site can train those who need to be educated in the use of the computer (at least it should). There is usually better general software at the central site; the counter argument is that some minis do a better job at a specialized task. There is good cost accounting at the central site--you really know what things cost.

For an institution, what are the economic advantages of a distributed specialized system relative to those of a central system? It has been estimated that half the money spent at a central site supports the staff, a small amount goes for supplies, and the rest goes for hardware and communications. While costs for hardware and communications are decreasing, costs for personnel will probably remain as they are. Thus, if an institution cannot make savings in hardware, it probably is not going to gain from a distributed system. I have analyzed the costs of a central site at some Big Ten institutions. While it is difficult to compare data, some inferences can be made. At the University of Wisconsin, where we did a major in-depth study, salaries and wages are 53% of the central site budget; at Michigan State, they are 62% (or 58% if I include amortization of the central equipment); at other institutions they are 51% and 42%. These facts argue that, particularly at schools like mine where hardware is a very low percentage of our cost, the money is locked up in salaries, and we are not going to gain a great deal by going to a mini-midi-system. The number of jobs will not decrease if we split off the work, although this point can be argued. This personnel commitment is an important consideration favoring the central site.

The list of the advantages of specialization is long and impressive. I have already mentioned cost economy. CPU and memory costs are lower, but mass storage costs are the same for the central as for a remote or a distributed site. Software development, maintenance, and, of course, communications cost less, because the computer is where the user is. Nevertheless, mainly because of technology, hardware costs are going down so fast that it appears that you better get into the minicomputer business. It has been estimated that hardware costs are going to decrease at 15% per year, while salaries will probably increase at 12% per year. If you can take advantage of the new technology, your net computing costs will go down, which is a very strong argument for getting away from the central site.

Another advantage of the minicomputer is, as I have discussed, user control. If you have a system in your own house, you are in control, the machine is continuously available, and you have a flexible operational procedure that is oriented toward the user. We have found in our interviews that users who may complain about lack of access at the central site are very considerate when it is their colleagues who are on the system. With minicomputers, there is a different level of operation and of sharing than at the central site. You have a simpler operating system because the computer is smaller so that you get more of the cycles that are there. You have greater flexibility. It costs less to increment the system--

you do not have to spend $2 million for a new mainframe at the central site. The cost of using the minicomputer for the year can be predicted precisely, whereas you cannot predict costs at the central site, because that is a billing scheme. Minicomputer hardware is generally more reliable: 'the smaller the system, the fewer the problems' is an old adage I think we all know well. You can tailor the software and the hardware to your job, and if you need particular functions, you can make the hardware do them. For instance, I talked to people who, although they have developed extensive software over the past years, find that it is now cheaper to tailor hardware (i.e., use minicomputers) than to get a new large mainframe and redo the software. I think we will see more and more of this style of operation in the future. With minicomputers, interactive response is good because it is a shared environment--if your colleagues are on, you can ask them to get off, and they can come back in a few minutes. And, of course, the smaller software packages encountered on a minicomputer are usually less expensive then larger packages.

At the central site, I have seen exceedingly expensive fiascos on a large scale. For example, we developed a package that was supposed to handle interactive applications. It was a very good package--the only problem was that it took one entire CPU to run. It was an expensive software project that never did its job. Problems such as these are avoided with a mini system.

Now let me emphasize the factors I feel are necessary for specialization success. First, a common interest area assures good internal communication. I think a small user base is absolutely necessary. There would be little real-time effort mixed with interactive and batch use on the successful operation. You must have poor or costly central service as a motivating factor. One person must serve as a leader; I am convinced of this because I have seen failures where the leader was absent. You must have an operator-software-maintenance expert on hand. There must be adequate hardware to do the job. There should be little or no software development; you should use the software as the vendor provides it. You should have a reliable operating system, few tapes, and little RJE activity (i.e., do not use the minicomputer to access a central site 24 hours a day). These to me seem to be necessary factors for success.

From the point of view of the institution, cost accounting at remote sites is a problem. I do not think that with remote sites we really know exactly what the costs are. We neglect faculty salaries--and we use faculty to do the job that the central site does with technicians. Since total cost is often unknown, we cannot allocate expenses among the various users. Thus, we do not know the return we are getting for the cost of the CPU cycles. The accounting scheme usually does not allocate the costs and, indeed, from a user point of view, you may not want them allocated. It may be to your advantage to avoid allocation because you are burying a job that you think is important but with which the administration would disagree. Another problem is the escalation of the mini's capabilities and applications. Once the mini is there, pretty soon it needs a printer, a better disk, and so on. You are duplicating the central site equipment and, as each system in turn grows, you are duplicating the cost. As the system grows, managerial problems and incumbent costs go up. I do not see a way around this problem. A university must maintain a large central

site because everybody cannot get a minisystem. The alternative is a network type of service, which I find to be unacceptable to our users.

A minicomputer purchased today will be obsolete in five years. As a matter of fact, by the time you try to get the 3rd or 4th enhancement, your system is probably obsolete. At that point, you may have to pay to convert to another system, because the original company may be out of business.

Another problem with minicomputers is that users are not mini-oriented. There are problems of word size, indirect addressing, and limited software support. Many vendors do not provide the software support that the vendors of large systems do. IBM, for instance, will not let you fail if you are running a central site, and seek their help. Mini vendors do not have the dedication to the user base that IBM does. Finally, the peripheral costs are just as high for minisystems as they are for a maxisystem.

In summary, I am convinced that, despite the pitfalls and problems, there is a place for small computer systems in big computing environments. The problem comes, as I view it, in finding that place and identifying and educating the users. The gains that are obvious in principle can then be made in practice.

Bibliography

1. Ira W. Cotton, Microeconomics and the market for computer services, Computing Surveys, 7, 95 (1975).

   Discusses topic "Economies of Scale" and references primary literature sources.

2. David A. Hodges, Trends in computer hardware technology, Computer Design, 15, 77 (1976).

   Makes predictions on future hardware improvements and points out theoretical limitations on improvements.

3. Richard C. Canning, Making use of remote computing services, EDP Analyzer, 15, No. 9, 3 (1977).

   Discusses a Stanford Research Institute in-house study on the development of mini-computer use.

4. Richard C. Canning, "Personal" computers for business. EDP Analyzer, 16, No. 5, 2 (1978).

   Discusses application of Terah 8510A micro processors in University of California at San Diego computer science teaching laboratory.

5. Richard C. Canning, Distributed data systems, EDP Analyzer, 14, No. 6, 3 (1974).

   Discusses reasons for considering distributed delivery systems.

6. George E. Mueller, Whither the large computer, Mini-Micro Systems, 10, No. 1, 67 (1977).

   Presents arguments on why large systems must change.

7. Edward K. Yasaki, The Mini: a growing alternative, Datamation, 22, No. 5, 139 (1976).

   Discusses AIIE conference on mini computers.

8. Elizabeth F. Severino, When minicomputers are on your mind, Computer Decisions, 1, 40 (Nov. 1977).

   Discusses mini myths and argues that prospective users must know both capabilities and limitations of minis before committing to use them.

9. Robert L. Patrick, Decentralizing hardware and dispersing responsibility, 22, Datamation, No. 5, 79 (1976).

Argues for decentralization while pointing out some caveats of mini systems.

10. James C. Emery, The coming challenge of campus computation, Educom Bulletin, 13, No. 1, 20 (1978).

Argues that computing services must be obtained from a number of different sources.

11. Frank V. Wagner, Is decentralization inevitable? Datamation, 22, No. 11, 86 (1976).

Traces cost of computing components since 1960 and projects them to 1985. Argues that if a group has < 30 people, it should have its exclusive computer provided:
a) the computer is big enough to do the job, and
b) it will be loaded to >10% of capacity.

12. Frederic G. Withington, Future computer technology, Data Base, 7, No. 4, 7 (1976).

Lists qualities of computer systems which will "entice" most users, and consequent software and hardware design principles. Argues that future systems will include all classes of computers - networked.

13. Richard C. Canning, In you future: distributed systems? EDP Analyzer, 11, No. 8, 8 (1973).

Discusses advantages and disadvantages of mini systems and large systems.

14. Peter Lycos, Ed., "Minicomputers and large scale computations. American Chemical Society, Washington, D.C. (1977).

Discusses numerous specific applications of mini computers.

SYSTEM DESIGN AND SPECIFICATION

Jack W. Frazer
Lawrence Livermore Laboratory

Introduction

I am in both administration and research, and therefore come with an entirely different persuasion than the previous speaker. I believe there is no longer such a thing as a minicomputer; all so-called mini's are getting to be fairly large with respect to computational, control, and memory capabilities. In the future, all progressive laboratories will use many micro- and mini-computers in distributed and hierarchical networks to support instrumental analysis, experimentation, and distribution of information. A number of such systems are now in various stages of design and construction. My interest is in the type of system that can give me all the time-response and bandwidth characteristics required for advanced instrumentation, experimentation, and process control, as well as perform computations generally restricted to batch processing on medium and large computers.

Four or five years ago I thought that participation in a workshop such as this was something that would not be required in 1978. At that time we were working hard on development of procedures for system specification and design of complex computer systems.[1-6] We hoped that by developing more standard procedures and by documenting working examples we could provide the necessary general guidance for the design and construction of new systems.[7-24] Obviously we were wrong.

Many years ago, in the mid-60's, we at LLL started with an empty computer and built a real-time, time-shared system for instrument control, data acquisition, and data reduction. Out of that experience I became convinced that to build a good laboratory system that meets the real-time requirements, has the desired bandwidth characteristics, and supports imaginative experimentation, you had to begin by carefully defining and specifying the desired system before the selection of specific hardware! Otherwise, it is very difficult, if not impossible, to cost-effectively implement the desired system. In the late 60's, I investigated a number of failures in laboratory and process automation, including million dollar projects, and found that in every case of failure the system had not been properly specified nor designed before the computer hardware was acquired. The scientists and engineers had spent the bulk of their time and effort on hardware selection, without the benefit of system specifications and designs.

System Specifications

One definition of specifications is: the listing of those myriad details necessary to direct the uninformed in the construction, installation, and testing of a complex project. For automation of chemical instrumentation and experimentation, some of the elements of the system specifications are: the transfer function of the instrument, instrument control function, digital signal processing algorithms, environmental conditions (temperature, electrical noise, and humidity), and any chemical

procedures relevant to the automation. In short, when developing specifications and designs "think systems."

In addition to the above items, management practices are very important if you are building a distributed or a time-shared system of any kind. As one simple example, sample identifications and reporting procedures in very large systems can be one of the more difficult aspects of the automation process. If you do not carefully analyze and characterize such procedures, design and implementation of the software is extremely difficult.

There are many elements of the system to be automated that should receive careful attention before selection of specific hardware. However, because automation is so complex and requires the expertise usually found in several diverse scientific and engineering disciplines, it is very difficult to readily generate a set of specifications. Why not, then, treat automation like other complex problems, i.e., separate the variables into manageable domains? There are many ways of separating the variables so the designer can readily develop specifications and designs. One method that we found useful and which has been generally adopted by ASTM Committee E-31, is to begin by first partitioning the system into three domains, i.e., inputs to the system, outputs from the system, and the transfer functions that interconnect these two.

Inputs. Inputs to an automated system are any stimuli that can or in fact do evoke a response from the system. Inputs include signals from instruments, transducers, terminals, environmental noise (electromagnetic radiation, and light sources, when photosensitive transducers are used), and transient noise on power lines. Environmental conditions such as humidity, temperature, and rate of temperature change can also act as inputs, often stimulating the system so as to invalidate data or in worst cases result in total system failure.

When developing system specifications these various inputs should be characterized so the designer can assure system immunity to unwanted signals and design for complete recovery of the desired information from relevant signals. For example, complete characterization of the analog signals from instruments and transducers must be included in the specifications. This includes the time response, bandwidth, and noise characteristics. Time response for a required action is the time that elapses between the need for the service and the time the service is complete. As an example, for many simple data acquisition tasks the time response is the time elapsed between the clock-initiated CPU interrupt and the time the analog-to-digital conversion is complete. In other cases, where the information from an instrument or transducer is being used for control purposes, the time response might include the above time plus the time required to execute the control algorithm, initiate a change in control, and the time required for the system to reach the new (or safe) condition. In short, time response is often a crucial characteristic that must be carefully specified if very accurate information or time-precision control is required.

Signal bandwidth specifications define the frequency components of the analog waveforms that contain the useful information. Given these specifications, appropriate data rates can be established for accurate signal reconstruction and analysis of noise-free signals. Because signals

from instruments always contain noise, signal bandwidth characterization must include analysis of the noise frequency and intensity. Given good specifications of the instrument noise and the signal containing chemical (or other) information, the designer can include in the design the appropriate analog and/or digital filtering required to remove the noise without distorting the desired waveform. If this cannot be accomplished, a new or redesigned instrument must be used.

Outputs. The outputs are the system responses to inputs (stimuli). They require the same kind of considerations discussed for inputs. The output specifications should be organized so as to support design procedures, a stepwise consideration of types of outputs would include such things as whether they are digital or analog, their time response characteristics, required data rates or bandwidths, human engineering considerations, environmental conditions, and the grounding requirements.

Transfer functions. A transfer function can be considered as an algorithm that describes how an output is obtained from one or more inputs. One method of describing transfer functions is by flow charts and timing diagrams. These various functions will be executed by means of both hardware and software. The exact trade-offs between hardware and software implementation techniques principally depend on the data rate requirements, time-response characteristics, and execution times of software algorithms; i.e., computer speed and algorithm design.

Functional designs. Functional designs are graphic representations of the data paths connecting the inputs, transfer functions, and outputs. They correspond to the architect's blueprints. On many pathways it is desirable to include the required data rates and time response characteristics.

Procedures

The above gives a brief overview of the extent of specifications required for the automation of complex systems. Developing specifications is by far the most difficult task one must perform before any system is operational. Therefore, the most cost-effective procedure is to complete these before selecting specific hardware. The following is an operational procedure we use and that has been adopted by ASTM.

- Definition
- System specifications
- Functional design
- Implementation design (hardware and software selection)
- Implementation
- Test and evaluation
- Documentation

We recommend that one begin by writing a tutorial definition of what is to be accomplished by the automation. This document should include the main objectives and goals of the effort, such as improved through-put and/or accuracy, improved experimental capabilities, better analysis and associated reports, etc. It should also include a cost-benefit analysis, which can only be included after completion of the System Specifications and Functional Designs and completion of at least preliminary implementation designs.

For complex automation projects the development of specifications and design followed by implementation and testing is an iterative process as shown in Fig. 1. If one follows the procedures briefly discussed above and in the referenced literature, one finds that while complex automation is difficult, it is not only possible but manageable and generally cost-effective.

Summary

The foregoing was a very brief discussion of a procedure many have found helpful in the design and implementation of complex automation projects. The many references cited discuss the philosophy and evolutionary stages that accompanied the development of more standard procedures. In addition, case histories in the form of complete sets of specifications etc. are referenced.

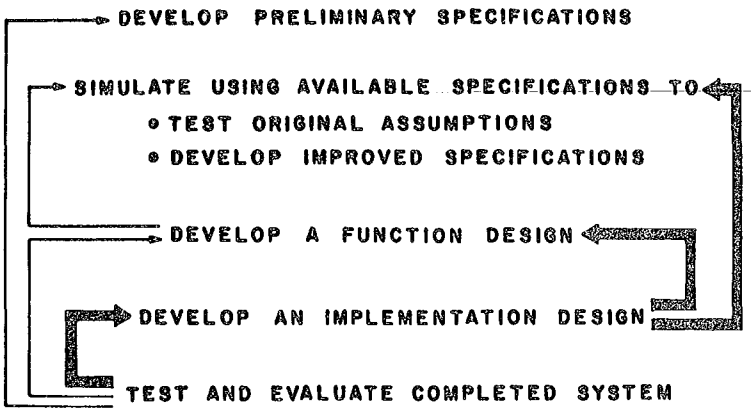

Fig. 1. Procedures for development of complex chemical measurement systems requiring multiple instruments.

REFERENCES

1. Jack W. Frazer, Laboratory Computerization: Problems and Possible Solutions, in Proc. American Society for Testing Materials, Philadelphia, Pennsylvania, UCRL-72470, May 12, 1970.

2. Jack W. Frazer, A Systems Approach for the Laboratory Computer Is Necessary, Materials Research Standards, Vol. 12, No. 2, 8-12, 1972.

3. Jack W. Frazer, Design Procedures for Chemical Automation, Amer. Lab., Vol 5, No. 2, 21-35, 1973.

4. Jack W. Frazer, Automation Trends, ASTM Special Technical Publication 578, in Proc. American Society for Testing Materials, Cleveland, Ohio, UCRL-75320, March, 1974.

5. Jack W. Frazer, Computerized Laboratory Systems, in Proc. ISA Conference, New York, New York, October, 1974.

6. J. W. Frazer, and F. W. Kunz, editor, "Computerized Laboratory Systems", American Society for Testing and Materials, Cleveland, Ohio, 1974.

7. J. W. Frazer, and G. W. Barton, et al, "A Feasibility Study and Functional Design for the Computerized Automation of the Central Research Laboratory, EPA Region V, Chicago, IL," in ASTM Computerized Laboratory Systems, STP 578, 152-256, 1975.

8. H. S. Ames, G. W. Barton, Jr., R. I. Bystroff, R. W. Crawford, A. M. Kray, and M. D. Maples, A Feasibility Study for the Computerized Automation of the Annapolis Field Office of EPA Region III, UCRL-52052, August, 1976.

9. F. B. Stephens, et al, Feasibility Study for Computerized Automation of EPA Region II Technical Support Branch, UCRL-52426, Feb., 1978.

10. W. G. Boyle, Jr., G. W. Barton, Jr., and L. Taber, A Feasibility Study for the Computerized Automation of the Laboratory Services Branch of EPA Region IV, UCRL-52416, Mar., 1978.

11. E. S. Peck, G. W. Barton, Jr., and E. Fisher, A Feasibility Study for Automating the Kansas City Surveillance and Analysis Division Laboratory, Environmental Protection Agency, Region VII, UCRL-52486, August, 1978.

12. W. F. Morris, E. R. Fisher, and G. W. Barton, Jr., Feasibility Study for Automating the Analytical Laboratories of the Chemistry Branch National Enforcement Investigation Center, Environmental Protection Agency, UCRL-52483, June, 1978.

13. W. F. Morris, et al, Feasibility Study for Automation of the Central Laboratories, Water Resources Division, U.S. Geological Survey, UCRL-52001 Jan., 1976.

14. R. I. Bystroff and W. G. Boyle, Jr., Atomic Absorption Instrument Functional Description, UCRL-52065, April, 1976.

15. W. G. Boyle, Jr., Computer Programs in BASIC Language for Atomic Absorption Flame Spectroscopy Part 1. Operating Instructions, UCRL-52401, Part 1, Jan, 1978.

16. W. G. Boyle, Jr., Computer Programs in BASIC Language for Atomic Absorption Flame Spectroscopy Part 2, Documentation, UCRL-52401, Part 2 January, 1978.

17. R. W. Crawford and L. P. Rigdon, Total Organic Carbon Analyzer: Functional Description, UCRL-52045, April, 1976.

18. L. P. Rigdon, et al, Computer-Automated Total Organic Carbon Analyzer: Operating Instructions and Computer Documentation, UCRL-52407, Feb., 1978.

19. R. W. Crawford and G. W. Barton, Jr., Technicon AutoAnalyzers; Functional Description, UCRL-52046, April, 1976.

20. L. Taber, Electronic Balance Operating Instructions, UCRL-52475, July, 1978.

21. W. F. Morris, E. R. Fisher and L. Taber, Automation of the Jarrell-Ash Model 70-314 Emission Spectrometer, UCRL-52474, April, 1978.

22. H. S. Ames, Laboratory Data Management for the Environmental Protection Agency, UCRL-52054, April, 1976.

23. R. W. Crawford and H. S. Ames, Sample File Controller: Functional Description, UCRL-52047, April, 1976.

24. ASTM Standards E622 through #627 in the 1978 Annual Book of ASTM Standards, Part 42(American Society for Testing and Materials, 1916 Race Street, Philadelphia, PA, 1978), 494-531.

Work performed under the auspices of the Department of Energy by Lawrence Livermore Laboratory under Contract W-7405-Eng-48.

ARRAY PROCESSORS AND MULTIPROCESSORS

Neil S. Ostlund
Department of Chemistry
University of Arkansas

I feel very much a novice at this topic so let me say a little about how we got into the area of multiprocessors. Until a year and a half ago I knew nothing about microprocessors, whatsoever. Like many quantum chemists, we had been using a central university facility and had ignored, for the most part, questions of hardware, electrical engineering, and a lot of the details of computer science that we really did not care too much about. I have changed my mind and I now think it is important, with what is happening in computer hardware, that theoreticians like myself get involved in some of these questions. It is a very exciting business. Our involvement is a result of one of the undergraduates leaving a copy of BYTE, the personal computing magazine, lying around. We started reading it and got very interested. We decided to participate in the microprocessor revolution to a small extent, and to get started we built a Z-80 microcomputer. After you are used to using CPU's that cost $19.95, on sale, it is clear that what you would really like to do is put a whole bunch of these things together. As computational chemists, we have a great demand for processing power, and I think we can get a lot of this power very cost-effectively by parallel computations on multi-microprocessor architectures. A graduate student of mine, James Neece, and I are in the process of designing a prototype multiprocessor, based on the Z-80 CPU, which we eventually hope to build. It is rather a different thing for quantum chemists to worry about cold solder joints, defective integrated circuits, and so forth. Nevertheless, I think it is a worthwhile thing. If we are really going to take advantage of the revolution occurring in microelectronics, we have to become much more involved with hardware questions and the interface between hardware and software than we have been. Particularly, if we really want a lot of processing power, we will have to do things in parallel. Eventually, the speed of light and finite gate delay times will get us, and some form of parallelism will be necessary for the processing power we demand.

I will be spending the next year in the computer science department of Carnegie-Mellon University which has been involved with multiprocessors and multiprocessing for a number of years. What I intend to do is to try and implement quantum chemistry calculations in parallel on the Carnegie-Mellon machines. They have two large multiprocessors; one is C.mmp (16 PDP-11's) and the other is Cm* (50 LSI-11's). I will particularly be trying to develop quantum chemical algorithms for Cm*. The calculations we do now, like solving the Roothaan equations, all involve serial algorithms. It is a very different problem doing such calculations in parallel. Optimum algorithms for solving the electronic Schroedinger equation may be totally different in parallel rather than serial form. In the future one is definitely going to do things in parallel and one needs a lot of development and a lot of thinking about how to organize such calculations.

To put various forms of parallelism in perspective, let me define a few terms. A pipelined processor is a lot like an automobile production

line where an automobile exits the "pipeline" every few minutes rather than the weeks or months it would take a single person to assemble it from parts. A Vector machine like the Cray-1 or an array machine like the Illiac IV are single instruction, multiple data (SIMD) machines which have multiple arithmetic and logic units (ALU's) so that, for example, a vector of numbers can be added to another vector of numbers in one simultaneous operation (Fig 1). Commonly, individual operations like floating point multiplication are pipelined also.

The complexity of integrated circuits is going up exponentially (Moore's law) and it seems clear that sometime in the 1980's a 370 or a VAX will be available on a single chip. One of the ways of taking advantage of such very large scale integration (VLSI) is to connect individual computers or CPU's (rather than ALU's) together to form a multiple instruction, multiple data (MIMD) machine. A network (Fig. 2) is a very loosely coupled set of computers communicating by message transfer. The classical multiprocessor (Fig. 3), on the other hand, is tightly coupled with each central processor (Pc) having uniform access to all of primary memory (Mp). C.mmp, with 16 PDP-11's connected to 16 memory modules through a central switch, is an example of such a machine.

Cm* is a multiprocessor with a somewhat different hierarchical structure. The basic unit is a computer module (Cm) consisting of an LSI-11 processor, local memory, various I/O devices, and a local switch (Slocal) which provides a simple interface between the Cm and the rest of the system. The primary memory of the system consists exclusively of the local memory of the Cm's. Up to 14 computer modules can be connected by a MAP BUS to form a cluster, under the control of a mapping controller (the Kmap). Communication between clusters occurs via the intercluster buses. All of primary memory is accessible to any processor, but the access time is non-uniform; references to memory outside a cluster take longer than references to memory in another module within a cluster, which take longer than references to local memory inside a module. Relative times are approximately 9:3:1. A simple 3 cluster system is shown in Fig. 4. For the last year and a half Cm* has been operating with 10 processors but it is now being expanded to 50 processors, 10 in each of 5 clusters. The basic structure is extensible and one could envision hundreds of individual processors in the system.

One knows very little yet about how to program these machines. The hardware is getting cheaper but a great deal of effort, by computational chemists, is needed to develop parallel algorithms appropriate to these newer architectures.

Tomorrow Alex Grimison from Cornell will describe the array processors such as that produced by Floating Point Systems. These are not really array processors in the sense of Illiac IV, but add-on very fast pipelined floating point units with a small additional degree of parallelism in their operations. I believe Alex has some impressive results on the cost-effectiveness of adding an array processor to a mainframe such as the IBM 370. Then, Peter Hibbard from Carnegie-Mellon will talk about multiprocessors, specifically C.mmp and Cm*. Peter is much more capable than I am at describing these machines, and I am sure you will find them very interesting. Multiprocessors, unlike the SIMD machines, need not operate in a lock-step synchronous fashion and Peter will also talk about

the advantages of asynchronous operation for iterative problems, on a machine like CM*. Finally, I will suggest a few of the possibilities for applying multiprocessors and parallel algorithms to problems in quantum chemistry.

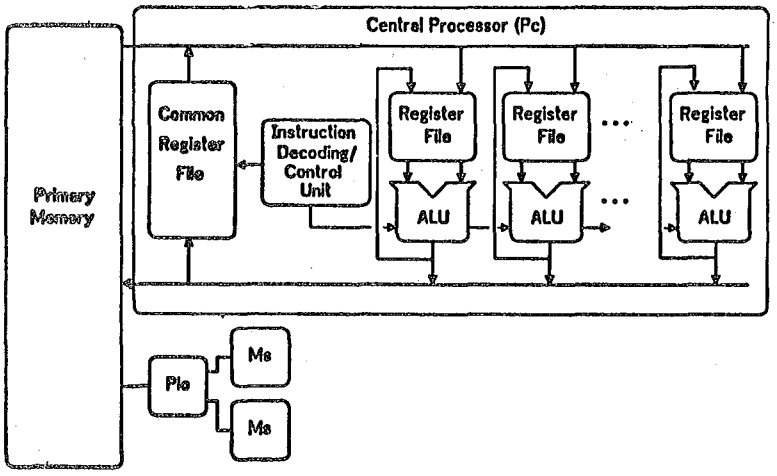

Fig. 1. Multi-ALU processor.

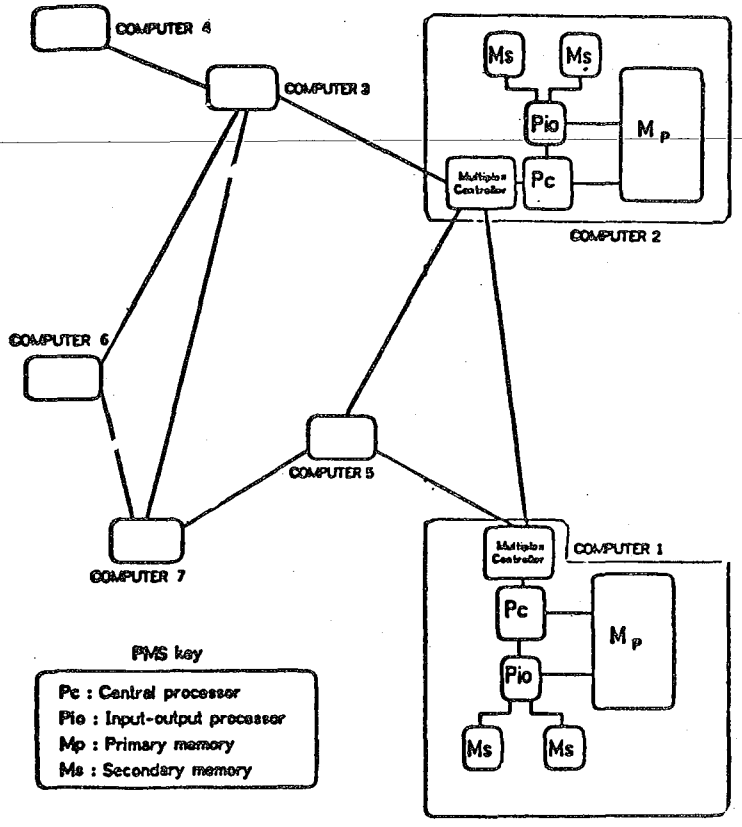

Fig. 2. A network of computers.

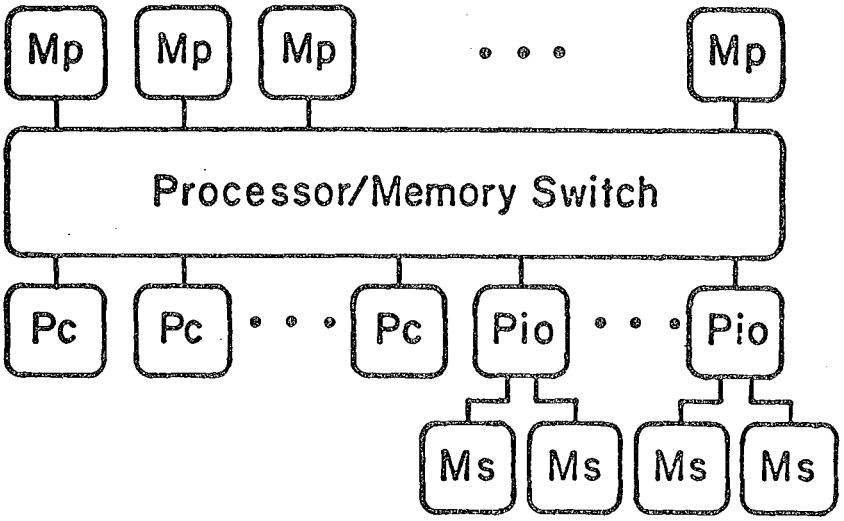

Fig. 3. The basic structure of a multiprocessor.

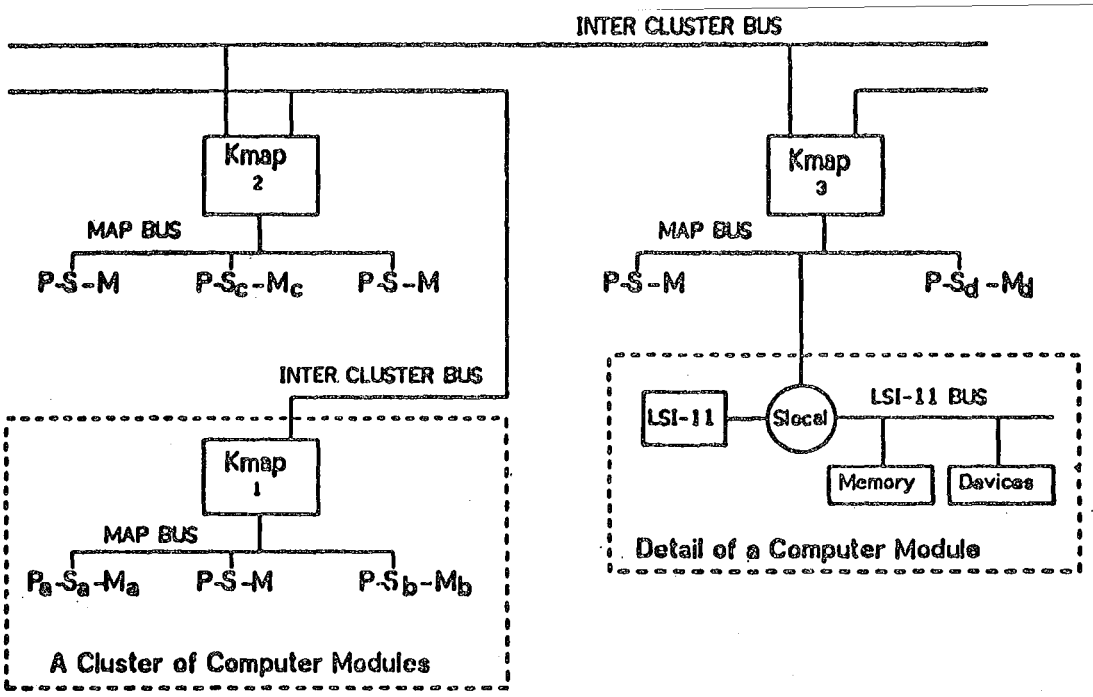

Fig. 4. A simple three-cluster CM* system.

BIBLIOGRAPHY

SURVEY PAPERS

1. P.H. Enslow (ed.),
Multiprocessors and Parallel Processing,
Wiley, 1974.

2. P.H. Enslow,
Multiprocessor organization - a survey,
Computing Surveys, 9, 103 (1977).

3. S.H. Fuller, A.K. Jones and I. Durham (eds.),
Cm* Review,
Carnegie-Mellon University Computer Science Dept., June, 1977.

4. L.C. Hobbs and D.J. Theis,
Survey of parallel processor approaches and techniques,
in L.C. Hobbs et al. (eds.), Parallel Processor Systems, Technologies and Applications,
Spartan Books, 1970.

5. D.J. Kuck,
A survey of parallel machine organization and programming,
Computing Surveys, 9, 29 (1977).

6. H.S. Stone,
Parallel Computers,
in H.S. Stone (ed.), Introduction to Computer Architecture,
SRA, 1975.

7. Special Issue on Computer Architecture,
Communications of the ACM, Vol. 21, #1 (January, 1978).

8. Special Issue on Computer Systems Architecture,
Computing Surveys, Vol. 7, #4 (December, 1975).

9. C.V. Ramamoorthy and H.F. Li,
Pipeline Architecture,
Computing Surveys, 9, 61 (1977).

10. H. Lorin,
Parallelism in Hardware and Software,
Prentice-Hall, 1972.

ARRAY PROCESSORS

For many of the Array Processors which have been produced in limited numbers, the main source of information is the manufacturers' literature. Manufacturers of Array Processors include:

1. ESL Inc., 495 Java Drive, Sunnyvale, Cal. 94086

ESL produces an Advanced Scientific Array Processor (ASAP), and also a Peripheral Array Processor System (PAPS) consisting of an ASAP and a Telefile TCP-16 minicomputer, permitting connection to large IBM, CDC, or Univac Systems.

2. IBM Corporation.

   IBM produces a very large Array Processor, the IBM 3838 which contains an IBM 370/148 as a controller and can support up to 7 concurrent users.

3. Datawest Corp., 7333 E. Helm Drive, Scottsdale, Ariz. 85260.

   Datawest produces a MATP 400 Array Processor system with 4 processors.

4. CSP inc., 209 Middlesex Turnpike, Burlington, MA 01803.

   A product line of Modern Array Processors (MAP). It is stated that over 40 manuals documenting MAP hardware and software are available.

5. Data General Inc., 290 Elwood Davis Road, Liverpool, NY 13088.

   Data General have recently announced an Array Processor, which is a CPU extension to the ECLIPSE S/130, and adds 42 special purpose instructions (including a Fourier Transform instruction) to the standard ECLIPSE instruction set.

There is a certain amount of literature available on the Systems AP-120B and 190-L Array Processors, produced by Floating Point Systems, P.O. Box 23489, Portland, OR 97223.

6. Floating Point Systems AP-120B Processor Handbook.

   A description of the architecture of the AP-120B and AP-190L hardware and instruction set, with some discussion of sample host interfaces.

7. Floating Point Systems AP-120B Software Development Package Manual.

   Describes the use of the AP Assembly Language (APAL), the AP Simulator (APSIM), and the AP Linkage Editor (APLINK) provided by FPS for the AP-120B and 190L.

8. Floating Point Systems AP-120B Math Library Part I and II.

   A description of the over 200 hand-coded APAL routines available from FPS for the 120B and 190L, covering a variety of signal processing and general vector and matrix operations.

9. "Floating Point Systems Array Processor Software and Simulator" Cornell Computer Services TN-94.

   A (draft) Technical Memorandum available from Computer Services, G-24 Uris Hall, Cornell University, Ithaca, NY 14850 describing the use of the FPS software on the Cornell 370/168, with sample sessions, etc.

10. "Floating Point Systems Array Processor Hardware" Cornell Computer Services TN-95.

    A (draft) Technical Memorandum available from Computer Services, G-24 Uris Hall, Cornell University, Ithaca, NY 14850 describing the use of the AP-190L Array Processor attached to the Cornell 370/168.

11. "Array Processor FORTRAN Compiler (APTRAN)" Cornell Computer Services TN-97.

A (draft) Technical Memorandum available from Computer Services, G-24 Uris Hall, Cornell University, Ithaca, NY 14850 describing the use of the FORTRAN compiler for the AP-120B and 190L, developed at Cornell.

MULTIPROCESSOR HARDWARE AND PERFORMANCE

1. S.H. Fuller, J. Ousterhout, L. Raskin, P. Rubinfeld, P. Sindhu and R. Swan, Multi-microprocessors: an overview and a working example, Proc. IEEE Trans. Computers, 66, 216 (1978).

2. F.E. Heart, S.M. Ornstein, W.R. Crowther and W.B. Barker, A new minicomputer/multiprocessor for the ARPA network, AFIPS Conf. Proc., 42, 529 (1973).

3. R.J. Swan, The switching structure and addressing architecture of an extensible multiprocessor: Cm*, Carnegie-Mellon University Computer Science Department, July, 1978.

4. R.J. Swan, A. Bechtolsheim, K. Lai and J. Ousterhout, The implementation of the Cm* multi-microprocessor, AFIPS Conf. Proc., 46, 645 (1977).

5. R.J. Swan, S.H. Fuller and D.P. Siewiorek, Cm* - a modular multi-microprocessor, AFIPS Conf. Proc., 46, 637 (1977).

6. Tandem Computers, Tandem 16 System introduction, Tandem Computers, Cupertino, CA, 1977.

7. W.A. Wulf and C.G. Bell, C.mmp - a multi-miniprocessor, AFIPS Conf. Proc., 41, 765 (1972).

8. Levy Raskin, Performance Evaluation of Multiple Processor Systems, Carnegie-Mellon University Computer Science Department, August, 1978.

9. S.H. Fuller, Price/performance comparison of C.mmp and the PDP10, IEEE 3rd Ann. Symp. Comp. Arch., 1976,195.

10. W.A. Wulf, Minicomputer Complexes: Progress and Prospects, in J.M. Ortega (ed.), Computer Science and Scientific Computing, Academic Press, 1976.

MULTIPROCESSOR PROGRAMMING ISSUES

1. S.H. Fuller, A.K. Jones and I. Durham (eds.),
Cm* Review,
Carnegie-Mellon University Computer Science Department, June, 1977.

2. P.G. Hibbard,
Parallel processing facilities,
New Directions in Algorithmic Languages,
IRIA, 1976,1.

3. P.G. Hibbard, A. Hisgen and T. Rodeheffer,
A language implementation scheme for a multiprocessor computer system,
IEEE 5th Ann. Symp. Comp. Arch., 1978, 66.

4. P.G. Hibbard, P. Knueven and B. Leverett,
Efficient implementation of multiprocessing in Algol68,
Proc. 5th Int. Symp. Algor. Lang., 1977, 202.

5. A.K. Jones, R. Chansler, I. Durham, P. Feiler, D. Scelza, K. Schwans and S. Vegdahl,
Programming issues raised by a multi-microprocessor,
Proc. IEEE Trans. Computers, 66, 229 (1978).

6. A.K. Jones, R. Chansler, I. Durham, P. Feiler and K. Schwans,
Software management of Cm*,
AFIPS Conf. Proc., 46, 657 (1977).

7. A. Newell and G. Robertson,
Some issues in programming multi-miniprocessors,
Behaviour: Research methods and instrumentation, 7, 75 (1975).

8. P. Oleinick,
The implementation of parallel algorithms on a multiprocessor,
Carnegie-Mellon University Computer Science Department, July, 1978.

9. W.A. Wulf, E. Cohen, W. Corwin, A. Jones, R. Levin, C. Pierson and F. Pollack,
HYDRA: the kernel of a multiprocessor operating system,
Comm. ACM, 17, 337 (1974).

10. W.A. Wulf and S. Harbison,
Reflections in a pool of processors,
Carnegie-Mellon University Computer Science Department, February, 1978.

11. W.A. Wulf, R. Levin and C. Pierson,
Overview of the HYDRA operating system,
Proc. 5th Ann. Symp. on OS Principles, 1975, 122.

PARALLEL ALGORITHMS

1. G.M. Baudet,
Asynchronous iterative methods for multiprocessors,
Journal ACM, 25, 226 (1978).

2. P. Oleinick and S.H. Fuller,
The implementation and evaluation of a parallel algorithm on C.mmp,
Carnegie-Mellon University Computer Science Department, June, 1978.

3. G.M. Baudet,
The design and analysis of algorithms for asynchronous multiprocessors,
Carnegie-Mellon University Computer Science Department, April, 1978.

4. G.M. Baudet, R.P. Brent and H.T. Kung,
Parallel execution of a sequence of tasks on an asynchronous multiprocessor,
Carnegie-Mellon University Computer Science Department, June, 1977.

5. D. Heller,
Survey of parallel algorithms in numerical linear algebra,
Carnegie-Mellon University Computer Science Department, February, 1976.

6. H.T. Kung,
Synchronized and asynchronous parallel algorithms for multiprocessors,
in J.F. Traub (ed.), Algorithms and Complexity, New Directions and Recent Results,
Academic Press, 1976.

7. H.T. Kung,
The complexity of coordinating parallel asynchronous processes,
Proc. 15th Ann. Allerton Conf., 1977, 34.

8. W. Miranker,
Parallel Methods for solving equations,
IBM Res. Rept., RC 6545, 1977.

9. M.A. Franklin,
Parallel solution of ordinary differential equations,
IEEE Trans. Comput., Vol C-27, pp. 413-420, May, 1978.

10. W.G. Poole, Jr. and R.G. Voigt,
Computing Rev., Vol. 15, pp. 379-387, Oct., 1974.

11. A.H. Sameh,
Numerical Parallel Algorithms -- A survey,
D.J. Kuck et.al. (eds.), High Speed Computer and Algorithm Organization,
Academic Press, 1977.

12. D. Heller,
A survey of parallel algorithms in numerical linear algebra,
to appear in Siam Review.

BENCHMARKING AND ASSESSMENT

Chris T. Corcoran
Department of Chemistry
University Wisconsin-Madison

As the solid state technology continues to advance, many chemists are showing interest in getting into the world of microprocessors and minicomputers. Often the people involved in the procurement of this equipment for their department or research group have had little prior experience in the evaluation of sophisticated computer systems. They may be influenced as much by friends who have equipment similar to what they want, as by any systematic study of all the available equipment and its effectiveness for the problems which they wish to tackle. Unfortunately, if someone has taken delivery on a system and has had a chance to evaluate its performance, there is undoubtedly already something faster and cheaper on the market. Thus, the friend's comments may no longer be relevant to the situation of the prospective buyer.

Once a decision has been made to purchase a particular piece of equipment, and once delivery has been taken on that equipment, the purchaser usually finds himself in the situation of being much more intimately involved in the actual day-to-day operation of the machine than he was when he was using the central computer facility. It is important for him to develop a much clearer understanding of how the hardware and software operate, particularly since most of the minicomputers that are of interest for computational purposes currently involve some sort of a virtual memory or cache memory system. These systems are perhaps foreign to the person who has been using CDC 6600's all his life. He may find there are difficulties in changing his programs to run effectively on a minicomputer. By running some simple programs on the machine he has already purchased, he may be able to develop a greater understanding of how the system operates, and how to more effectively use the wall-clock time of the machine. It is with these points in mind that we approach not only the subject of benchmarking, but also the systematic assessment of minicomputers in general. We should also consider the potential role of the NRCC in developing benchmarks and assessing various computer systems. Obviously a single user is not going to be familiar with the wide range of programs that would be of interest to the various research groups throughout a chemistry department, whereas the NRCC has that specific role, and would be in a much better position to develop wide-ranging benchmarks. For example, in the Wisconsin chemistry department, there are a number of research groups with diverse interests using the departmental computer, and a number of questions arise as to how effective the machine is for these various groups. This sort of an analysis could possibly be handled very effectively by a central organization.

To begin our discussion of what benchmarking actually is, we can define a benchmark as a program or mix of programs which is run in several environments in order to assess those environments. A simulation program, which is frequently used to aid in benchmarking, is usually very simple with only a few adjustable parameters which control the various levels of I/O and CPU usage within the program. In many ways one can more clearly understand a particular system by running a simulation program, rather

than a very large, time consuming program, which goes through many different steps. Looking simply at the time the program finished, and seeing whether or not it got the right result is oversimplifying the role of benchmarking. Generally the simulation program will provide a much more detailed breakdown of the performance of the system.

In the development of a benchmark one must consider the reason the benchmark is required. As I have mentioned, one reason might be to compare various hardware and software systems available from different vendors. Other reasons would be to determine the cost-effectiveness of the mini-computer vs. the maxi-computer, or to assess the effectiveness of the various optimization levels of the system software. We also consider that bench-marking is vital as a quality control measure for accepting a piece of equipment. If you order equipment with certain specifications you would expect to do some sort of benchmarking to determine that the equipment meets those specifications. Another reason for running benchmarks is to determine ways to improve the efficiency of the system you may already have. This is one of the primary reasons we have been running benchmarks during the past year. Specifically we have attempted to determine the effect the addition of more memory would have on our system. We have found, while running benchmarks, a deeper understanding of the idiosyncrasies of the operating system which are not clearly explained in any manual. This has allowed us to schedule jobs in such a way as to increase the overall throughput of the system.

Finally, we expect that there will be a significant difference in the type of benchmark program used for a minicomputer as opposed to a large scale computer. In general, a minicomputer will be used by one or a few users, while the large scale computer must be simultaneously accessible to a large number of users. Also the minicomputer will frequently employ a cache or virtual memory system to complement a relatively small high-speed memory. Benchmarks for such a machine will certainly have to take such factors into account.

Quantum Chemistry Simulation Benchmark (1976)

Machine	Bits/Word	Time	Calculation (bits)
Varian V73	16	503	4 x 16 = 64
Systems 32	32	260*	2 x 32 = 64
ModComp IV/25	16	195	4 x 16 = 64
Interdata 8/32	32	146	2 x 32 = 64
Harris /4	24	140	2 x 24 = 48
PDP 11/70	16	120 (?)+	4 x 16 = 64
TR 440	48	165	1 x 48 = 48

Hardware Maintenance

Interdata 8/32	DM 52,000.-/year
TR 440	DM 1,200,000.-year

Note: Supplied by Klaus Wenzel, Ruhr-Universitat, Theoretische Chemie, Postfach 10 21 48, 4630 Bochum 1, West Germany.

*Significantly reduced since 1976.

+Benchmark program favored cache system.

THEORETICAL CHEMISTRY

Clifford E. Dykstra
Department of Chemistry
University of Illinois-Urbana

In the last six years minicomputers have established themselves as important tools for theoreticians doing both small and very large-scale calculations. About six years ago there were four dominant considerations when thinking about using minicomputers for theoretical chemistry calculations. On the advantage side was the cost effectiveness, rising primarily from the cheap hardware of the minicomputer and the fact that you normally would not provide any staff support. Another important advantage, which has been mentioned in some of the earlier talks, is the direct control of system. On the disadvantage side was the size limitation. When minicomputers were first being used for quantum chemical calculations, it was clear the minicomputers would not really have as much memory as large scale machines. In addition, you would not buy 5 or 10 disc drives--you would be happy to have one high speed disc drive, so there was less external storage capability. It seemed that the calculations that could be done on minicomputers would not be the same size as what you could do with, say, the CDC 7600. Furthermore, the software that was generally available on machines such as the Harris/4 at Berkeley, was not at all equivalent to what you would have with CDC or IBM. This lack of software would result in additional programming effort. For example FORTRAN compilers would not give you as much help in debugging your program.

One of the points about cost effectiveness is of course the computer memory, which is one of the real motivations for using minicomputers in theoretical calculations. If we look back we can see how costs have declined. In 1968, the Department of Chemistry at the University of Illinois purchased an IBM 1800. The additional cost for 16,000 16-bit words at that time was $36,000. If we assume that one would want about 48 bits to have sufficient precision for theoretical chemical calculations, this works out to about $6.75 per word. The Berkeley Harris/4 was purchased in 1973, and the cost for 8,000 24-bit words was about $7,500, or a factor of four reduction in the memory cost over a five-year period. Even more dramatic is the reduction in memory cost with the introduction of the Digital Equipment Corporation Vax 11/780. With this machine, the incremental cost for 250 kilobytes of memory is only $13,000, or roughly another factor of four reduction in the per-word memory cost over the Harris/4. While these figures are not precise, they give an idea of the overall trend. In the next year or two, it looks as if the reduction in cost will be about another factor of 4, so essentially memory is becoming very, very cheap on minicomputer systems. (If we extrapolate memory costs forward you can see that about April or May 1981, computer memory might even be free!) This means that minicomputers purchased for theoretical calculations are going to have a lot more memory, so we have eliminated one of the initial disadvantages of minicomputers.

Some of the topics discussed in this session of the workshop include adapting and writing programs on a minicomputer, including what has been learned about making programs restartable in case of system crashes and about the differences in software; selecting a minicomputer system with

regard to doing large scale theoretical chemistry calculations; benchmarking for specific types of operations that one would encounter in a quantum chemical calculation; and the cost per calculation as a function of the utilization of minicomputer machines.

As we look to the future, we see several trends in addition to declining memory costs. There is a trend towards better software in minicomputers. With better software, and with growing experience of a body of minicomputer users doing theoretical chemistry, there will be an increase in transferability of computer programs. Programs are being written so that they do not necessarily rely on special FORTRAN features, and so that they are restartable and use disc space in a fashion which can be taken from one machine to another. Putting all this together, I think the outlook is that minicomputers are going to be of continued importance for theoretical chemists. The extension of that conclusion is that if the NRCC wants to give strong support to theoretical calculations, it should certainly consider the possibility of investing in a large scale dedicated minicomputer as a way of providing cheap but efficient computing time for theoretical chemists.

A comparison of minicomputers available around 1973 has been given by Schaefer.[1] This outlines economic arguments for using minicomputers for large scale theoretical chemistry calculations. A very complete discussion of problems, advantages and accomplishments with the Harris/4 minicomputer used by the Miller-Schaefer theoretical chemistry groups is their final report to the National Science Foundation.[2]

Among the most recently developed minicomputers which may be suitable for theoretical chemistry applications are those produced by Digital Equipment (VAX), Harris, Interdata, and Hewlett-Packard. Useful and detailed information on these systems is available from the manufacturers.

---

[1] H. F. Schaefer, IEEE Annals 75, 61 (1975).

[2] W. H. Miller and H. F. Schaefer, Final Report to National Science Foundation, Grant GP-39317, 1976.

Table 1. Incremental computer memory costs of selected systems.

Year	System	Amount of memory	Total cost	Cost per word (at least 48 bits)	Cost per byte
1968	IBM 1800	16,000 16-bit words	$36,000	$6.75	$1.125
1973	Harris/4	8,000 24-bit words	$7,500	$1.875	$0.3125
1977	Harris/7	96,000 bytes	$30,000	$1.875	$0.3125
1978	DEC VAX 11/780	250,000 bytes	$13,000	$0.416	$0.052
1978	HP 3000	64,000 bytes	$4,000	$0.375	$0.0625
1979	NS System 400		(estimate)	$0.09	$0.01

Table 2. Representative monthly maintenance costs of roughly equivalent systems.

Unit	H-P (40hr/wk)	DEC(40hr/wk)	Harris
Complete system	984 (1%)	1117 (0.5%)	2115 (1%)
Extra memory	183	60	360
Disk drive	54	140	175
Tape drive	62	60	95
Line printer	147	185	195
Card reader	64	53	55

MOLECULAR MOTION SIMULATION COMPUTER

Thomas A. Weber

Bell Laboratories
Murray Hills, New Jersey 07974

Introduction

The general selection process which was used by the molecular motion simulation group at Bell Laboratories in 1976 to choose a midicomputer system has application to the selection of a system today. Although the optimal choice of a machine and various vendors' relative positions might well be different today, some underlying principles remain.

One of the most important initial requirements is to be able to define the user group and the type and range of problems which will be solved. For example, our machine was to be used by a group of only seven principal investigators, and only for molecular dynamics simulations (both Newtonian and Langevin dynamics), Monte Carlo simulations, and a small set of problems involving slowly converging differential equations. Implicit in knowing the type of problem is also knowing the type of problem which will not be solved by the computer system. For example, system requirements were not intended to duplicate the extensive graphical facilities available on the in-house computer system.

The molecular dynamics calculations are typically compute-intensive with modest core and I/O requirements. For example, one sixteen-hour calculation might require 30 to 40 K of double precision words and produce 5-10 pages of output. The Monte Carlo calculations require random array addressing capabilities and can easily tax the limits of a virtual memory machine which has a very small resident core memory. This requirement immediately eliminated some of the more popular 16-bit minicomputers. The slowly converging differential equations problems require a timesharing capability to allow convergence by successive approximation without totally utilizing the system to the exclusion of the large background jobs.

In selecting a vendor, it was clearly necessary to separate the various hardware and software features into (a) those that were absolutely necessary and (b) those that would be useful, but were secondary in importance. For example, a fast floating-point processor was essential to enhance throughput, whereas microprogramming capability was useful but not of first priority. Where possible, objective criteria, such as benchmarking were used to rank the various machines.

It is very important to have benchmarks that are representative of the typical workload of the system, to be sure that the system will be capable of handling the job requirements. Also, benchmarks are performed at a specific date: relative ranking of machines may change as manufacturers introduce newer and faster processors, and as different problem sets must need to be executed. However, objective decisions must be made on proven capabilities and not on promises of better hardware, arriving at some future date. Figure 1 shows benchmark speeds (in seconds) for a Gaussian molecular dynamics simulation in September 1976. This calculation

utilized over one-quarter of the available machine time, and so is representative of the typical workload.

Another objective criterion which may be used to rank machines would be required hardware and software capabilities. For example, a bi-directional remote job entry link (RJE) was necessary so that the data could be transmitted to our Honeywell system for graphical processing.

Subjective features, such as the quality of the text editor, the operating system, and the support libraries available must also be considered. Clearly a high powered editor or advanced Fortran IV with many extended features is not absolutely necessary, but would greatly aid in the process of software generation. When basic hardware and software capabilities are nearly the same, these qualities can tip the balance in machine choice.

Harris System 220

When all these considerations were weighed in the fall of 1976, we chose the Harris System 220 (Slash 7). Some of the hardware features of the System 220 are listed in Fig. 2. These include a central processor unit, 96 K of 24-bit words, two 40-megabyte disks, a 600 CPM card reader, a 600 LPM line printer, a 9 track 800 BPI magnetic tape unit, and ports capable of handling 8 terminals and an RJE interface. The system cost approximately $200,000 in 1976. Today a similar system with substantially more core is available for about the same price. The Harris Slash 7 processor can access 256 K words of physical core although the virtual addressing system is capable of accessing 1024 K words. The full core complement may be purchased for the newer Slash 8 for the same cost as the 96 K of core for the Slash 7.

The System 220 features the VULCAN operating system (see Fig. 3) which is a virtual memory system supporting timesharing, multi-stream batch, remote job entry, and real-time processing. The system includes an interactive editor and numerous language processors of which Fortran is the most important for the planned applications. Also the job control commands may be programmed to form higher level commands.

Cost Analysis

The Harris S220 compares favorably with the Honeywell in terms of speed and is considerably cheaper. Figure 4 shows a cost analysis for the Harris Slash 7, partially derived from one and one-half years of operating experience. Hardware cost was obtained by amortizing the machine over five years. The maintenance contract represents approximately 1% per month of the purchase price. The RJE costs are charges incurred for unit record transfer and equipment rental. The loading figure represents the loaded salary of one-half of a staff member who, although not necessary to maintain machine operations, allowed us to use a more sophisticated machine with expansive debugging facilities rather than a midicomputer.

The useable machine time for the first half of 1978 was 85% of the total available time. Therefore a reasonable charging rate for calculations on the Harris S220 would be $15 per hour.

To illustrate the cost effectiveness of using a high-powered midicomputer to perform large scale computation, it is helpful to look at the example of the Gaussian molecular dynamics simulation. Figure 5 shows the cost of a 2000-step run on the Harris Slash 7, the IBM 370/168 and the Honeywell 6078. The total CPU time for each run on the different machines is also listed. The runs were made at the lowest priority on each machine, necessitating a substantial turnaround delay. Between December 1976 and June 1977, 150 runs were made on the Harris at an estimated cost of $33,000. The equivalent runs on the Honeywell would have cost $180,000 and would have taken from one to one-and-one-half years. This dramatic example of the cost effectiveness of using a midicomputer is directly related to the processor speed of the Harris Slash 7 and the significant price reduction of the newer hardware.

Summary

Since December 1976 the Harris System 220 has been fully loaded and producing simulations from about the second week of installation. Over this rather long time span, better than 85% of the total available time has been used for productive user-calculations.

In part, the good experience with the system 220 is due to the minimal software difficulties encountered in the VULCAN operating system. Because of the overall maturity of the VULCAN software, the inevitable software bugs have been minimal and also the support languages are, in general, advanced enough to enable easy software generation. The advanced Fortran, which contains extended language features and which supports in-line assembly code, can be credited for making software generation easier.

Hardware problems have been minimal and, except for two periods when the machine was down for over a week, down-times have usually been less than a day. It should be noted that at our location there are three Harris systems, so that two resident field engineers are available.

User dissatisfaction has been minor, relating mostly to the lack of extensive debugging and diagnostic aides for program generation. This dissatisfaction is more a reflection of the very positive experience users have had with the Honeywell system than a condemnation of the Harris system.

Benchmark Speed (9/13/76)

IBM 370/168	19
Harris Slash 7	84
Honeywell 6078	86
Interdata 8/32	124
HP 3000	155 (sp)
Data General Eclipse	186
Modcomp IV	186
Prime 400	206
SEL 8/32	210

Fig. 1. Benchmark speeds 9/13/76). obtained by various systems for a Gaussian molecular dynamics simulation (9/13/76).

Harris System 220

- CPU — Memory Parity
  Priority Interrupt
  Multiply/Divide/Square Root
  Stall Alarm
  Demand Page / Virtual Memory
  Clock / Interval Timer
  Scientific Arithmetic Unit
  Hardware Bootstrap
- 288 KByte interleaved core memory
- Two 40 MByte Disks
- CRT Operator's Console
- 600 CPM Card Reader
- 600 LPM Line Printer
- 9 Track 800 BPI, 45 IPS Magnetic Tape Unit
- Terminal Ports and RJE Interface

Fig. 2. Some hardware features of the Harris System 220.

VULCAN Operating System

- Virtual Memory System
- Batch, Interactive, Real-time Processing
- Remote Job Entry
- Support Programs
  — Sort/Merge, Accounting, DEBUG, etc.
- Language Processors
  — Fortran IV, Macro, Basic, Snobol 4, etc.
- Programmable Macro Job Control Language

Fig. 3. Features of the tem. VULCAN operating system.

Operating Costs

Harris	$40,000
Maintenance	23,000
RJE	6,000
Loading	40,000
	$109,000

Useable hours/year

(360 × 24 × .85)	7344

Cost $15/hour

Fig. 4. Operating costs of the Harris System 220.

Cost Comparison

	Harris	IBM	Honeywell
2000 steps	$220	$273	$1200
CPU hrs/run	14.7	1.72	16.7
Turnaround (av. days)	1.4 - 4	14	7
150 runs	$33K	$41K	$180K

Fig. 5. Comparison of the cost of a 2000-step run on three different systems.

THE CARNEGIE-MELLON UNIVERSITY CHEMISTRY COMPUTER FACILITY

Stephen Binkley

Department of Chemistry
Carnegie-Mellon University
Pittsburgh, PA 15213

I. Applications

The main goal of the Carnegie-Mellon University Chemistry Computer Facility is to provide the tools necessary for solving problems in computational chemistry. To date, the equipment (described below) has been mainly used in *a priori* calculations of the electronic structure of atoms and small molecules.

II. Equipment

The principal piece of equipment is a VAX-11/780 computer system that has the following specifications:

1.	processor	VAX-11/780 CPU with optional floating point accelerator
2.	memory	512 K-bytes of metal-oxide semiconductor memory
3.	disc	RP06 disc unit, 200 mega-byte storage capacity (unformatted), 30 millisecond average access time
4.	tape	TE16 tape unit, 9 track, 800/1600 bits-per-inch, 45 inches-per-second
5.	terminal support	DZ11-A unit, supports 8 terminals
6.	miscellaneous	line printer--DEC LP11, 300 lines-per-minute card reader--DEC CR11, 300 cards-per-minute 2 Lear-Siegler ADM-30 CRT terminals console--DEC LA36 terminal
7.	installation date	mid-april 1978

III. Operational Comments

A. Commendable Features

1. Stand-alone operation: in the absence of tape- and disc-mounts, a full-time operator is not needed. The system fully recovers from failures (hardware and software crashes, and power interruptions).

2. Overall reliability of software: we have encountered no major software failures.

3. Overall hardware reliability: not only is the hardware reliable (two intermittent failures and one hard failure in the first 6 months of operation), it is easy to diagnose and repair (our 3 CPU problems required an average of 10 minutes each to fix).

B. Problem Areas

1. The Fortran compiler runs slowly. Several minutes are required to compile a 2000 card deck.

2. Fortran direct-access I/O is inefficient.

IV. Sample Benchmarks

The table lists the results of three different benchmark programs on a series of computers ranging from large mainframes to small mini's. Three benchmarks were performed:

1. JOB1--double precision arithmetic with no indexing (i.e., no dimensioned variables). This benchmark consists of iterative execution of the Fortran routine SP1111, obtained from the two-electron integral evaluation routines of the QCPE version of Gaussian 70. In each run, 5000 iterations were performed.

2. JOB2--matrix diagonalization. A 100 by 100 matrix was diagonalized using a standard Jacobi algorithm. All arrays were doubly subscripted (e.g., A(I,J)).

3. JOB3--same as JOB2, except the arrays used in the diagonalization were singly subscripted.

Note that double-precision was used on all machines except CDC where single-precision was used. The times quoted, for the Control Data machines were obtained using the FTN compiler with OPT = 2.

Due to the construction of the benchmarks, JOB1 provides a rough estimate of the floating point power of the various computers. It has been thoroughly hand optimized and there are few (if any) floating point operations that can be eliminated. JOB2 and JOB3 provide insight to compiler efficiencies in treating singly and doubly subscripted arrays. Note that the execution time increases in going from JOB2 to JOB3 on the Univac 1108, and decreases on the Harris Slash 6. Close scrutiny of the code generated by the 1108 compiler indicated that use of singly-indexed arrays had severely constrained the optimization capability of the compiler.

Benchmark Statistics

Computer	Time (in seconds)		
	JOB 1	JOB 2	JOB 3
Univac 1108	210	290	350
CDC 6600	54.4	65.2	-
CDC 7600	8.3	9.1	-
Harris Slash 4	264	702	-
Harris Slash 6	282	667	553
Harris Slash 7	278	627	-
Interdata 8/32	338	665	588
SEL 32/75	(280)	-	-
DEC 2040	462	897	-
DEC 2050	235	405	-
VAX-11/780	181	360	-

JOB 1 = Double precision arithmetic, no-indexing

JOB 2 = Jacobi diagonalization, double-indexing

JOB 3 = Jacobi diagonalization, single-indexing

TRENDS IN COMPUTER DESIGN

Peter G. Lykos
Department of Chemistry
Illinois Institute of Technology

There is a little anecdote which I thought might be appropriate. "In the late 1880's, a number of efforts were made to automate the switching of telephone calls. The first truly successful system was activized by Almond B. Strowger, an undertaker. How an undertaker came to design the world's longest lasting and most pervasive telephone switching system, is a classic story. Almond was one of two undertakers in a small town in the midwestern United States. The other undertaker's wife was a town telephone operator. When town residents suffered a death in the family they would often ask the operator for "an undertaker," and would of course be connected to the operator's husband. Strowger saw that his prospects for business were small unless he could eliminate the operator. So he devised a simple rotary switch mechanism using a Celluloid shirt collar and some common pins as a sample. He subsequently sold his idea to the Automatic Electric Company, which refined and developed his idea into the famous two motion mechanical switch that is still used in more than 25% of the telephone switching systems in the United States, and in a greater percentage of systems in various other countries. The Strowger system is called Step by Step by Bell System Companies."[1] This anecdote sets the theme for what we are going to be talking about.

You are already aware, through the popular or technical press, or papers presented earlier in this workshop, of the impact of very large-scale integration. We are witnessing the generation of very cheap processors and cheap memories, which means that we can make cheap copies of current hardware to run the corresponding software. That may be a very crude and blunt statement, but it has rather broad implications. Five years from now it should be possible to run the current IBM 370/195 operating system and utilities such as the FORTRAN compiler from a desk-top IBM 370/195 machine.

Then there is the multi-processor system, which can be thought of as a single, conventional, serial CPU with additional intelligence--disks, tape drives, or whatever--distributed to the peripheral devices. Instead of a single, serial machine supported by components handling some of its data,[2] a more sophisticated concurrent multi-processing system would be one in which the processors are related to each other in some non-trivial way.

Because of cheap processors and cheap memory, we no longer have to optimize hardware, and the user - machine interface can now move closer and closer to the user. Indeed, perhaps the user's needs can now be better understood by the designer, and reflected in the machine. People in advanced planning and design for the large vendors are polarized as to whether the potential of large, concurrent systems can ever be realized. Not only is the designer so much freer now, but the user also is basically and fundamentally tied to the serial machine. If we are going to realize the capability of much more complex devices with a high degree of concurrency,

then the old algorithms are probably no longer appropriate. In addition, new languages will have to be created. A major problem is that FORTRAN has become the de facto standard. It is a millstone around our neck, and there does not seem to be any way of getting out from under. According to computer scientists FORTRAN is an obsolete language, which imposes a lot of constraints, and is certainly causing a major problem in thinking about highly concurrent systems.

In this Workshop Session we will have five presentations speaking to the five points raised here, namely,

1. Moving the user-machine interface closer to the user. Stanley Cohen will illustrate this point by describing the SPEAKEASY system.

2. Using many processors in a highly concurrent system to increase computer power. David Stevenson of the Institute for Advanced Computation, home of the ILLIAC IV, will talk about the design of a successor to the ILLIAC IV, namely, the Phoenix.

3. The rapidly decreasing cost and increasing amount of circuitry and memory on a chip as a way of producing inexpensive copies of today's widely used machines in order to make more effective use of existing software--operating systems, utilities, compilers, etc. Ron Hochsprung of National Semiconductor will talk about the recently announced desk-top copy of the IBM 370.

4. Designing a system to handle particular classes of problems and abandoning the notion of the general purpose computer. As an example I will describe briefly the recently announced Culler-Harrison machine which seeks to embed in an optimum hardware/software environment the floating point array processor.

5. Finally, the fact that the quantum of computing has become so small that the choice as well as the computer is in the hands of the user. Manufacturers and retail outlets for hardware and software are already delivering personal total computers. I will describe that situation briefly as well for it underscores the fact that chemists and computers are now one to one, and those devices can now be put directly into the hands of the individual. In other words, the politics of computer acquisition is fast and deservedly becoming a lost art.

In a recently held symposium,[3] there was at least a first attempt to understand the potential of concurrent systems, and their implications in terms of new algorithms for familiar procedures. David Kuch, one of the symposium organizers, has simulated a computer (word length, number of registers, channel width, cycle times, instruction time, and so forth) in an effort to classify the problems for which FORTRAN programming is used.

A program accepts FORTRAN programs, and produces an analyzed program that can be run on the simulated machine. The machine can be fine-tuned to optimize the execution of a particular problem. The point of the invest-

gation is to see (1) whether such a simulated system can be made to operate, and (2) if FORTRAN programming is used for only a small number of classes of problems, requiring a correspondingly small number of computer designs. One reason for thinking about this is that a general purpose computer is, almost by definition, non-optimal for any specific problem. If there are distinct classes of problems, with the potential of highly concurrent systems, perhaps the time has come to define machines around classes of problems.

The Culler-Harrison machine (which has just been announced, although its existence has been known for some time) is an example of such a special-purpose machine. The objective was to design a machine that addresses two problems: (1) processing large volumes of data, and (2) massive mathematical computations. The multi-processor machine consists of six processors, and has four fast disks. Four of the processors handle I/O with the disks, and have direct memory access to the memory of the other two processors--the macro and the array processors. Here, a system was built around the power of the array processor, taking full advantage of its capability. The macro-processor, from which the system derives its designation MP-32A, serves as the host for the array processor, and handles control, I/O, bookkeeping, integer arithmetic, and the like.

FORTRAN and BASIC are available on this system, which attempted to produce not only useful hardware, but also useful software. A special math system language makes it possible to realize the full power of the system from the comfortable environment of these common, although no longer adequate, languages. The operating system supports both batch and on-line users.

In tests, the system achieved 4 million floating point operations per second (flops) for the fluid code, and 6 million flops for the particle code. Purchase price is on the order of $350 thousand plus, of course, the cost of software, maintenance, and supporting personnel. This gives you a feeling for what can be done by extending the current "super mini-computers," designing systems aimed at specific classes of problems.

Another consequence of very large-scale integration is the introduction of powerful personal computers. The growth in production and use of micro-processors is literally an explosion: There appeared recently in Time magazine a special section on the computer society. With micro-processors finding their way into consumer products there will be competition to produce a better product, better standardization, and better reliability--and this can only auger well for chemistry.

Let me give an example of what is currently available. An ordinary television set (giving visual information) can now be connected to a micro-computer that has been designed for use by a consumer who is not an electronics person. In addition, the microcomputer can also be connected to another common household item, the audio cassette (a mass storage device). Putting the whole thing together, we have a fully operational machine--a computer in the full sense of the word, with an operating system; supporting software; and input, output, and mass storage devices.

At the center of the microcomputer is the micro-processor, which usually works on an 8-bit data structure. The technology of the 8-bit systems is already so stable that the 16-bit micro-processor will be introduced into product lines only very gradually. However, the performance of some of the 16-bit machines has been compared favorably with that of the PDP 11/45. This begins to give you a feeling for the power that is available. The point is that these personal computers are engineered as total systems, not just bare, stand-alone micro-processors. They have all the buffers and interfaces needed to make them useful to most people. The system functions as a general, stand-alone machine, and also as a computer terminal.

On wire racks in the 500 personal computer retail stores around the country are plastic bags containing an audio cassette and a manual instructing the user how to put the cassette into a cassette player and load Basic into the computer. This is a very well conditioned Basic; there has been a lot of experience with it and, as far as I know, it is bug free. The instructions are very clear. So when I say a total system, I mean total in terms of hardware and software.

In addition to Basic, there are self-instruction texts available. The personal computer itself is a fully documented device, which is available as a completely assembled, stand-alone system or as a kit. The manual which supports it is reasonably complete. So we are not talking about buying bare, hardware components in the surplus shop, and then trying to figure out how to assemble them on an old piece of plywood--we are talking about a totally integrated system. The power of a such a minimal system today is fairly modest. Typically it has 16 K bytes of semiconductor memory, and can support a simple Basic. The speed of the audio cassette leaves something to be desired. One can, however, move up to a system with as much as 65 K bytes, dual floppy disks, a disk operating system, a FORTRAN compiler, an extended Basic, and so on. These systems have been around for some time, produced by stable companies. As a consequence the computer has now come to every man, if every man wants it, on a one--on-one basis. As very large scale integrated circuits continue to grow in size and complexity, and decrease in cost, and the corresponding memory sizes increase, very powerful machines indeed can become available to the individual. The manufacturing, delivery, and marketing systems are in place and operating, and the power and cost-effectiveness of these devices continue to improve with the end not yet in sight.

---

[1] John E. McNamara, "Technical Aspects of Data Communication," Maynard, Mass. (1977).

[2] Special Issue on Computer Architecture, Comm. of the ACM, January (1978).

[3] Kuch, D. J., Laurie, D. H., and Sameh, A. H., "High Speed Computer and Algorithm Organization," Academic Press (1977).

THE MINICOMPUTER AND COMPUTATIONS IN CHEMISTRY

National Resource for Computation in Chemistry
Lawrence Berkeley Laboratory
July 19-21, 1978

PARTICIPANTS

*Mark Abramson, Department of Chemistry, Emory University, Atlanta, GA 30322

Thomas L. Allen, University of California, Davis, Department of Chemistry, Davis, CA 95616

*Eric R. Beals, LBL, 50B - 2232D

Donald S. Berkowitz, Consulting, LBL Computer Center, 50A, Room 1148

Herbert J. Bernstein, Brookhaven National Laboratory, Chemistry Department, Upton, NY 11973

David L. Beveridge, Hunter College, CUNY, Chemistry Department, 695 Park Avenue, New York, NY 10021

*Steven Binkley, Department of Chemistry, Carnegie-Mellon University, Pittsburgh, PA 15213

Tom Boates, Bemidji State University, Bemidji, MN 56601

Heinz Bosshard, Washington University, Department of Computer Science, Box 1045 Lindell Skinker, St. Louis, MO 53136

David Ceperley, NRCC, LBL, 50 - 208

*Phillip R. Certain, Theoretical Chemistry Institute, University of Wisconsin, Madison, Wisconsin 53706

*Stanley Cohen, Speakeasy Computing Corporation, PO Box 5354, Chicago, IL 60680

*Chris T. Corcoran, Theoretical Chemistry Institute, University of Wisconsin, Madison, WI 53706

Raymond Cullen, Texas A&M University, Department of Chemistry, College Station, TX 77843

Paul Day, Argonne National Laboratory, 9700 South Cass Street, Argonne, IL 60439

*Indicates invited panel member

Minicomputer Workshop Participants

Jim Deer, Tektronix, Box 500 DS 58/299, Beverton, OR 97077

James P. Dix, Informatics Inc., NASA Ames Research Center, MX 230-3, Moffitt Field, CA 94035

David R. Dion, NRCC, LBL, 50 - 208

David A. Dixon, University of Minnesota, Chemistry Department, 207 Pleasant Street, S.E., Minneapolis, MN 55455

Jack J. Dongarra, Argonne National Laboratory, Applied Math Division, 9700 S. Cass Avenue, Argonne, IL 60439

*Clifford E. Dykstra, Department of Chemistry, University of Illinois, Urbana, IL 61801

Clyde Edmiston, Los Alamos Scientific Laboratory, Mail Stop 531, Los Alamos, NM 87545

*Harry A. Eick, Department of Chemistry, Michigan State University, East Lansing, MI 48824

Stephen T. Elbert, Ames Laboratory, Iowa State University, Ames, IA 50011

Thomas Farrar, National Science Foundation, Program Director, Chemical Instrumentation, Washington, D.C.

William H. Fink, Chemistry Department, University of California, Davis, Davis, CA 95616

Robert Florh, Signetics Inc., 730 Evelyn BEN 37, Sunnyvale, CA 94086

*Brian Ford, Director, Numerical Algorithms Group, Ltd., N.A.G. Central Office, 7 Banbury Road, Oxford OX2 6NN, England

*Jack W. Frazer, Lawrence Livermore Laboratory, L-326, PO Box 808, Livermore, CA 94550

Mark Freeman, Dan Oliver Corporation, International Headquarters, Stanford, CN 06904

Arthor Gaylord, Chemistry Department, University of California, Berkeley, Berkeley, CA 94720

Frederic Glaser, Pan American University, Department of Physics, Edinburg, TX 78539

***Indicates invited panel member**

Minicomputer Workshop Participants

Albert Gong, U.S. Customs Laboratory, 630 Samsome Street, Room 1508, San Francisco, CA 94111

Mark S. Gordon, North Dakota State University, Chemistry Department, Fargo, N.D. 58102

Artie Green, Hewlett-Packard, 3003 Scott Blvd., Santa Clara, CA 94050

David Grier, University of California, Berkeley, Department of Chemistry, Berkeley, CA 94720

*Alec Grimison, User Services, G24 Uris Hall, Cornell University, Ithaca, NY 14850

Carl Grose, 934 Liberty Street, #1, El Cerrito, CA 94530

William D. Gwinn, Department of Chemistry, University of California, Berkeley, Berkeley, CA 94720

Stanley Hagstrom, NRCC, LBL, 50 - 208

Kenton Hanson, Lawrence Berkeley Laboratory, MMRD, Room 180, Hearst Mining Building, University of California, Berkeley, CA 94720

Mervin P. Hanson, Chemistry Department, Humboldt State University, Arcata, CA 95521

*Peter Hibbard, Department of Computer Science, Carnegie-Mellon University, Pittsburgh, PA 15213

*Ronald Hochsprung, National Semiconductors MS-235, 2900 Semiconductor Drive, Santa Clara, CA 95051

Dick E. Hoskins, Chemistry Department, University of Wshington, BG-10, Seattle, WA 18195

James L. Jensen, Cal State, Long Beach, Chemistry Department, Long Beach, CA 90840

Harold S. Johnston, Department of Chemistry, University of California at Berkeley, Berkeley, CA 94720

M. Thomas Jones, University of Missouri, Department of Chemistry, St. Louis, MO 63121

*Jerome V. V. Kasper, Department of Chemistry, University of California at Los Angeles, Los Angeles, CA 90024

C. W. Kern, National Science Foundation, Chemistry Division, Washington, D.C. 20550

***Indicates invited panel member**

Minicomputer Workshop Participants

Bernard Kidda, 740 La Para Avenue, Palo Alto, CA 94306

Walter Levy, Exxon Research & Engineering, CRL, PO Box 45, Bldg. 14 - Room 102A, Linden, NJ 07036

Jim Lellmen, Searle Research & Development, 8220 Skokie Blvd., Skokie, IL 60077

*William J. Lennon, Lawrence Livermore Laboratory, Electrical Engineering Department, L-227, PO Box 5504, Livermore, CA 94551

Robert Linder, Chemistry Department, Stanford University, Stanford, CA 94305

Janet Lundgren, Informatics, NASA Ames Research Center, MS - 230, Moffitt Field, CA 94035

*Peter Lykos, Department of Chemistry, Illinois Institute of Technology, Chicago, IL 60616

Gilbert C. Lyle, NASA Ames Research Center, Moffitt Field, CA 94035

Gulzari Malli, Simon Frazer University, Department of Chemistry, Burnaby, B.C., Canada V5A156

Rod Manis, Computer Consulting, Suite 606, 75 Buena Vista East, San Francisco, CA 94117

Elizabeth H. Mei, Chemistry Department, University of California at Davis, Davis, CA 95616

Jelmer Medema, 2610 Hillegass Av., #201, Berkeley, CA 94704

Wilfred Meyer, Universitat Kaiserslautern, Department of Physical Chemistry, 675 Kaiserslautern, Pfaffenbergstrasse, Gebaud 14, West Germany

Alan Miller, New Mexico Tech, Socorro, NM 87801

James Morgan, Colorado State University, Department of Biochemistry, Ft. Collins, CO 80523

Bill Morgenstern, SYNTEX Research, Lab Automation, 3401 Hillview Ave., Palo Alto, CA

James T. Muckerman, Brookhaven National Laboratory, Chemistry Department, Upton, NY 11973

***Indicates invited panel member**

Minicomputer Workshop Particpants

T. N. Nararimhan, LBL, Bldg. 90, Room 1012D

Yorhimasa Nihei, Institute of Industrial Science, University of Tokyo, 7-22-1 Roppongi, Minato-ku, Tokyo 106, Japan

James Neece, Department of Chemistry, University of Arkansas, Fayetteville, AR 72701

*Neil S. Ostlund, Department of Chemistry, University of Arkansas, Fayetteville, AR 72701

Stephen L. Padwa, Brookhaven National Laboratory, Applied Mathematics Department, Upton, NY 11973

Christopher Alan Parr, University of Texas at Dallas, M.S. Be 2.6, Box 688, Richardson, TX 75080

Franklin G. Pater, Jr., E.I. DuPont de Nemours & Co., Central Research and Development Dept., Experimental Station E320, Wilmington, DE 19898

Stephen C. Peacock, Molecular Design LTD, 1122 "B" Street, Hayward, CA 94541

*Sam P. Perone, Department of Chemistry, Purdue University, West Lafayette, IN 47907

James A. Perry, E. I. DuPont de Nemours & Co., Central Research and Development Department, Experimental Station E320, Wilmington, DE 19898

William Plachy, Chemistry Department, San Francisco State University, San Francisco, CA 94132

George D. Purvis III, Battelle Columbus Laboratories, 505 King Avenue, Columbus, Ohio 43201

Leo Radom, Research School of Chemistry, Australian National University Canberra A.C.T. 2600, Australia

Richard Raffenetti, Argonne National Laboratory, Building 200, Chemistry Division, Argonne, IL 60439

David Franklin Randolph, 2322 Gardner Place, Glendale, CA 91206

M. V. Rao, Chemistry Department, Columbia University, 955 Havemeyer Hall, 116 Street and Broadway, New York, NY 10027

***Indicates invited panel member**

Minicomputer Workshop Participants

Garry L. Ritter, Schering Corp., B-8-1, Analytical Research, 60 Orange St., Bloomfield, NJ 07003

Timothy J. Rolfe, Department of Chemistry, University of Chicago, 5735 S. Ellis Avenue, Chicago, IL 60637

Klaus Ruedenberg, Ames Laboratory, Iowa State University, Ames, Iowa 50011

Bill Schriver, University of California at Berkeley, Chemistry Department, Berkeley, California 94720

Chet Seligman, PO Box 209, Pt. Reyes Station, CA 94956

Thomas Schatzki, United States Department of Agriculture, Berkeley, CA 94710

Huey-Sheng Shieh, Institute for Cancer Research, 7701 Burholme Avenue, Philadelphia, PA 19111

*Gene Small, The Harris Corporation, 10960 Wilshire Blvd., Westwood, CA

Allan L. Smith, Drexel University, Chemistry Dept., 32nd and Chestnut St., Philadelphia, PA 19104

Jack Smith, Quantum Chemistry Project, University of Florida, Gainesville, FL 32601

Dale Spangler, NRCC, LBL, 50 - 208

David Stephenson, Institute for Advance Computation, Sunnyvale, California

Allen Stock, Signetics, Inc., 730 Evelyn, BEN 37, Sunnyvale, CA 94086

Thomas Strong, LBL Computer Center, Building 40A, Room 1140C

Dan Sunday, 2325 Channing, Apt. 52, Berkeley, CA 94704

Sam B. Trickey, Department of Physics, Texas Tech University, Lubbock, TX 79409

Myrtha Truijillo, University of Puerto Rico, Cayey University College, Chemistry Division, Cayey, P.R. 00633

Isiah M. Warner, Texas A&M University, Department of Chemistry, College Station, TX 78843

*Robert Waters, Chief of the Benchmark Support Group, Air Force Office of Computer Acquisitions, Hascom Air Force Base, MA 01731

*Thomas A. Weber, Bell Telephone Laboratories, Room 1A364, Murray Hill, NJ 07974

***Indicates invited panel member**

Minicomputer Workshop Participants

Paul Weiner, University of California, School of Pharmacy, Dept. of Pharmaceutical Chemistry, San Francisco, CA 94143

*Klaus Wenzel, Ruhr-Universitat, Theoretische Chemie, Postfach 10 21 48, 4630 Bochum 1, West Germany

John E. Wilkalis, Allied Chemical Corp., Building CRL, Columbia Rd., Morristown, NJ 07960

James Work Syntex Research, 3401 Hillview Avenue, Palo Alto, CA 94303

*Indicates invited panel member

TECHNICAL INFORMATION DIVISION
LAWRENCE BERKELEY LABORATORY
UNIVERSITY OF CALIFORNIA
BERKELEY, CALIFORNIA 94720