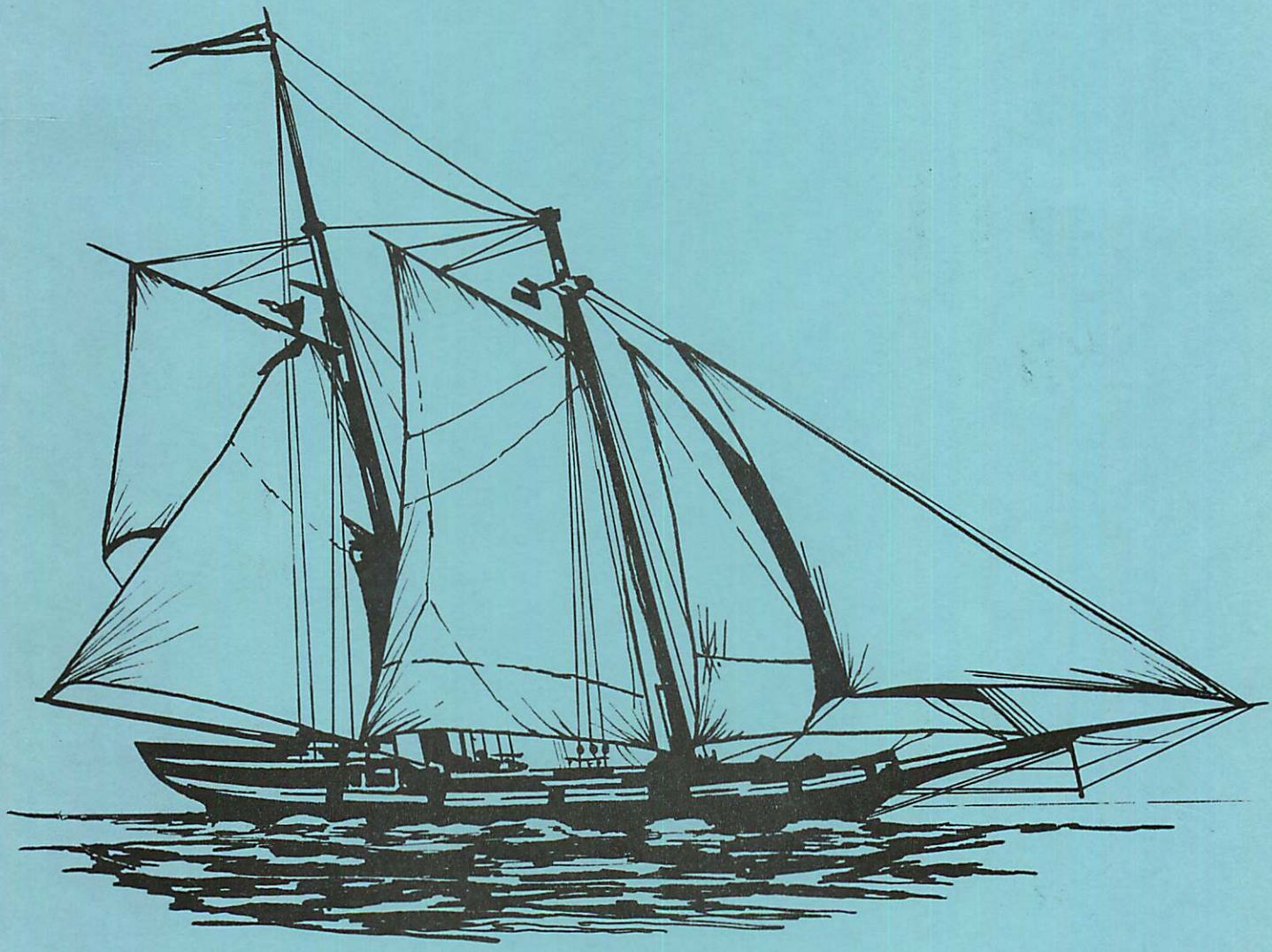
THOMAS P. COMBS

# :HP3000

# International Users Group Conference



Baltimore, Maryland
September 28 - October 1,1976

TABLE OF CONTENTS


Page

Preface..... iii

V. Chachra and R. Crockett (Library Circulation and Finding System)..... 1

Jay Denny (SE Service/Report)..... 12

Jim Elliott (HP Product Announcement)..... 21

Bill Foster (Future Directions of HP3000 Software)..... 23

Gil Drynan (Supporting and Manufacturing Systems--Material, Control, Accounting)..... 25

Jim Gunderson (IO Switching)..... 31

Tom Harbron (Installation Management)..... 33

Ed McCracken (State of General Systems Division)..... 36

Doug Mecham (Grin and Bare It)..... 39

Larry Meyers (Capability)..... 47

Lanny Norensberg (Getting Data Reports Managers Need--SPSS)..... 55

Stan Shell (HP Product Announcement)..... 64

Stan Shell (HP User Support Services)..... 66

Kenneth Van Bree (An Interactive Problem Solving Language)..... 71

Will Workman (New Product Announcements)..... 95

Robert Young (ARTHUR-JOB and Process Monitor).....101

Tom McShane (Tele Communications and The User).....105

Adelin M. Clesse (Micro Processor Communications Link for Data Collection).....128

Ross Scroggs (An Intra-Line Editor and Demonstration).....142

Charles W. Jackson (COBAL Shorthan).....173

Robert A. Goodman (Software That Works).....175

Norman Wright (How To Capture More Information Than You Really Wanted, Optical Scanners).....183

Page

Charles Van Ausdall (Transaction Logging).....195

HP3000 User's Group Questions and Answers.....210

HP3000 User's Group Membership.....219

The Library Circulation and Finding System discussed here is implemented at the Newman Library at Virginia Tech. Virginia Tech is an educational sanctuary that is located somewhere between the Blue Ridge Mountains and the Appalachian Mountains in southwest Virginia. We have a student population of 19,500 and the faculty and staff add up to 5,000 in number. So the library has approximately 25,000 patrons to serve. The library has a collection of one million volumes and is growing at the rate of approximately 10% a year, or 100,000 volumes a year. The manual circulation system was becoming totally inadequate to handle the 225,000 circulation transactions that we have each year. The obvious route in which to go was to see if some of the functions could be computerized.

In order to support circulation alone it is only necessary to use absence files. Absence files imply that a machine readable record be kept of only those items that are currently in circulation. Typically, this would be anywhere from 5 to 15% of the entire collection. Thus, a system based on absence files would require less resources than one that is predicated on having the entire collection in machine readable form. Very early in the decision making process it was agreed to have a system based on the entire collection being in machine readable form. Whereas this approach is initially more expensive to implement, it has some significant long term benefits. For instance, it allows for a computer based searching and finding capability which would not be possible otherwise. It permits the production of microfiche catalogs that can be easily distributed

to locations within and outside the University. It allows for a proper monitoring of the usage and trends in the library collection.

The design and implementation of the system was divided into two phases. Phase I of the system involved the development of the functional aspects of the system. Phase II, not yet implemented, involves the development of management analysis reports and statistics. Phase I is divided into four functional areas. They are:

1. Circulation control functions
2. Searching and finding functions
3. Data entry functions
4. Backup recovery and other utilities

The circulation control functions included checkout and checkin, holds/recalls, overdue notices and fines, bindery control, and inter-library loan. One of the primary objectives of the system was to provide a quick and efficient method for the check-in and check-out process. In order to facilitate this it was necessary to have a machine readable record associated with each item in the collection and with each patron using the Library. Accordingly, a barcoded label was (or will be) placed on each item of the entire collection. This barcoded label is machine readable and uniquely identifies the item. The ID-numbers of all patrons are also barcoded on the ID-cards. The checkout process simply consists of running a light pen (which reads the bar coded labels) across the label on the ID-card which identifies the patron to the system. This is followed by running

the light pen across the labels on the books to be checked out. This essentially completes the check-out process for the operator since the system automatically assigns the due dates for the items.

Whereas the check-out process is externally very simple, the system performs several functions before it permits the successful completion of the process. On reading the patron ID-card the system makes two checks. First, it checks to see if the patron is delinquent. If the patron is delinquent he has to first clear his record or receive special permission to check out the book. Next the system identifies the type of patron (staff, student, etc.). Associated with each type of patron is a default circulation period and it is this that the system retrieves and stores for subsequent use.

Upon reading the barcode on the item to be circulated, the system first checks to see whether or not the item is allowed to circulate. Reference materials do not normally circulate. If the item is not allowed to circulate then the system blocks the check-out process. Next, a check is made to insure that the item is not already charged to another patron. It would certainly be undesirable to have the records show that a given item is checked out to more than one patron. Thus, an item has to be checked in before it can be checked out. The system also retrieves the normal circulation period of the item. The return date for the item is determined by using the smaller of the patron circulation period and the item circulation period. A prestamped return slip is placed in the book, thus completing the check-out for that

particular item.

The check-in process is essentially similar except that the patron ID-card is not necessary for check-in. At the time of check-in, if the book was overdue then the system automatically computes the fine. If the patron is able to pay the fine, this is recorded and the transaction completed. If the patron is not able to pay the fine then a fine record is automatically created for the patron and the patron is considered delinquent.

The hold/recall operation is closely associated with the check-in process. Holds may be placed either at the call number level or at the item level. (Simplistically speaking, a call number hold means that a patron is willing to take any copy of a given book and an item number hold means that a patron wishes to have a specific edition, copy, or volume). In order to place a hold the patron gives the information to the operator and also fills out his address on a notification card. The notification cards are numbered serially and this serial number (or hold number) is also identified to the system. During check-in the system checks to see if a hold was placed on the book being checked in. If a hold was placed the operator is alerted and the hold number is displayed on the terminal. Using this hold number the operator pulls out the right notification card which is then mailed to the patron, informing him that his book is now available. The book itself is placed on the hold shelf.

When the patron comes to check out the book that was being held, the system, during the check-out process, insures that the book is indeed being checked out to the patron that placed a hold

on it. This acts as a double check and insures that the patron who requested the book first gets it first. There are many situations in which such an operating procedure can be unduly restrictive. Therefore, the system allows the operator to override any of the built-in checks if the circumstances so dictate.

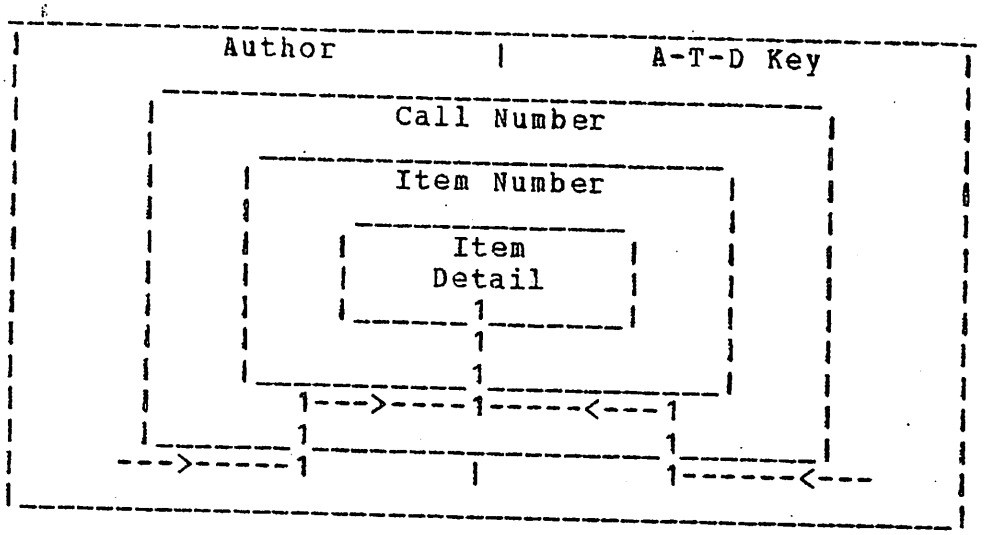
Procedurally, with one exception, the recalls are processed exactly in the same manner as the holds. In the case of the hold, the patron requesting the item waits until it is returned. However, in the case of the recall, a notice is sent to the patron informing him that the item is needed by another patron. Barring this one distinction, the two procedures are identical.

The system is designed to automatically generate overdue notices. Unlike all of the procedures described thus far, which are controlled by the use of a terminal, the overdue notices are produced by a batch program. As needed a batch program is initiated to generate overdue notices. A notice is sent to each patron having items that are overdue. The information on the notice includes the call number, title of the book, and its due date. An overdue book also makes a patron delinquent.

The bindery control function produces the necessary reports needed to determine what items have been sent to the bindery and when those items are expected to be returned from the bindery. Inter-library loans are processed as if the requesting library were a patron, with each library being assigned a unique barcode number. Thus, items are checked out as in the normal checkout process described above.

The searching and finding capabilities include an author search, a call number search, an item number search, and an author-title-date key search. Also, there are ways in which the circulation file may be searched to determine the status of an item or what items a patron may have checked out. At data entry time the system automatically generates an author-title-date key. The author-title-date key is 14 characters long and consists of three parts. The author data is used for the first part and consists of 6 characters. Five characters come from the author's last name and the sixth character is the first initial. In case of corporate authors, the first six non-blank characters are used. (In many cases a list of standard abbreviations is used). Eight characters are taken from the title. The first four characters of the first word plus two characters each from the next two words are used. The key does not need to be composed of the parts given above. It may be formulated in any fashion as long as it is fourteen characters in length. In addition, the publication date, if known, can be used to further limit the number of items displayed. In searching with the key use of partial information is permissable. Thus, if the author's initials are unknown then a '?' may be used instead. It would be perfectly acceptable to use S?,? as an author name and initial, implying thereby that the author's last name begins with S and no further information is available.

In all cases, the search is hierarchic. The following diagram explains the hierarchy.




The author search or the ATD-key search both lead to call number information. The call number search leads to item information and the item search gives the detail information concerning the item. Starting at any level of the search it is possible to go on to lower levels without rekeying any information displayed on the screen. Each request produces a response of no more than 16 lines. If a greater number of entries meet the search criteria then one screen is filled up and the word "more . . ." is displayed in the bottom right hand corner to indicate that there is more to come.

The 80/20 rule of inventory control states that approximately 80% of inventory carrying costs are due to approximately 20% of the items in inventory. A similar rule is surely true of library circulations. A rather small percentage of the total collection accounts for a large percentage of total circulation. In order to get the system running it seemed reasonable therefore to first capture all the data associated with items that circulated. Accordingly, the following data

entry and circulation procedure was established to support initial entry of high usage data:

a. Barcoded item labels were ordered in pairs.

b. For items that did not have a barcode on them the patron, at circulation time, wrote out only the call number on a slip of paper.

c. One barcoded label was placed on the item and its duplicate on the slip.

d. Circulation was now completed for the barcoded item and the slip with the item number and call number sent to data entry.

e. The data entry personnel could then enter the necessary data for the item in circulation without delaying the patron at the circulation desk.

The placing of the barcode on the book required some consideration. Perhaps the best location for it would be on the spine. However, the spine already had the call number on it and in many cases the spine was too thin to hold the label. The best alternative, therefore, was to place the barcoded label on the top left hand side of the front cover. This would be the most visible and accessible location. In case of physical inventory the label could be exposed and read by merely tilting the book on the shelf without completely removing it. The barcoded labels are coated with a protective coating to give them greater shelf life.

Two formats are available for data entry. The short format includes only the minimum data necessary for operations. This

data consists of item number, call number, LC card number, location, and circulation period. The LC card number is included because it acts as an access point to the MARC records. This will permit, at some future date, the extraction of full bibliographic records from MARC files. The long format consists of the data shown in Appendix A. Data may be entered on-line or by a combination of batch and on-line processing. Both formattable and non-formattable screens are available for data entry.

A three pronged approach is being tried for data entry. First, the circulation data is being captured as the books circulate. Initially only the short form is used for these items. Next, complete data is being entered for newly acquired items and for items processed during a shelf by shelf conversion. The third approach calls for capturing the complete data by extracting it from the MARC record using the LC card number. This last approach has yet to be implemented.

Since many of the operations for this system are on-line, it was imperative that proper backup and recovery procedures be included as an integral part of the system. In order to have the most reliable recovery procedure it was decided to log every update transaction on a system log. Further it was decided to put the log on a tape drive rather than a disc drive. Whereas tape logging is slower than disc logging it is significantly more reliable. If logging were done on disc then a disc crash would lose both the data and the transaction log causing an "unrecoverable" problem. Since no hard copy is kept on many of

the Library transactions this situation would be unacceptable.

Also in order to maintain the integrity of the data base contents various reports listing the contents of the data base are available. These reports include author list, galley proof list, and holdings list.

Summary

To summarize, the Library Circulation and Finding System provides for the following capabilities:

1. Creates a new record or modifies an existing record for each item in the library.
2. Creates a new record or modifies an existing record for a patron.
3. Provides a check-out facility which performs the following checks:
   a. identifies delinquent patrons
   b. identifies items that may not circulate
   c. automatically determines loan period
   d. allows for loan period other than normal.
4. Provides a check-in facility which performs the following:
   a. automatically clears patron record
   b. identifies items on hold or on recall
   c. keeps track of fines.
5. Provides a procedure for holds and recalls.
6. On a periodic basic prints overdue and recall notices.

7. Provides access into data using the following:

   a. Patron number

   b. Item number

   c. Call number

   d. Author

   e. Author-Title-Date key.

8. Provides for backup and recovery.

9. Collects statistics and does housekeeping chores.

Jay Denny

My name is Jay Denny and my job at Hewlett-Packard is as follows. I am responsible for all systems engineering services worldwide.

My objective is to discuss: 1) a perspective of Computer Systems Group; 2) why this organization is of benefit to you; and 3) how your local Hewlett-Packard team that we have allocated around you is coordinated and supported by Computer Systems group. The next two subjects I would like to talk about very specifically is Customer Engineering division, and the Systems Engineering Services program.

Here you see that our group is actually broken down into two basic enities. Each division has a marketing, research and development, and a manufacturing team. The various divisions are: 1) General Systems (3000); 2) Data Systems (1000); and (disc and magnetic tape), and Terminal Products Division. Divisions are Hewlett-Packard's way of managing our business. One of the newest divisions is Customer Engineering Division. The other portion of Hewlett-Packard is the Field Marketing Organization. That particular organization's job is to 1) support and assist the customer and 2) sell Hewlett-Packard products. That is made up of three basic employees. The sales representative, the system engineering personnel, and customer engineering personnel. As you can see, Paul Ely, our vice-president, keeps us operating as a single unit.

Each one of the divisions are profit centers. A general rule of thumb in Hewlett-Packard is to break down every particular portion of our business into some manageable enity. When these profit centers become up around 100 million dollars, we start looking very carefully at ways to break them apart to manage that business more effectively.

As our customer, you are surrounded by three of our representatives. My objective is to give you a brief idea of the jobs of those three people, the management team supporting them, and briefly, the functional team that is supporting them. The real benefits of our organization is not going to be visible unless it really does assist you as customers.

The sales representative. This person coordinates the team. His direct management is the district sales manager, regional sales manager, and national sales manager. The national sales managers are Jim Schnider and Bill Richion (408/257-7000).

The system engineering personnel. These are involved in technical education, technical evaluation, and technical information. His immediate responsible boss is the district sales manager and the regional sales managers. I am responsible worldwide for this organization (Jay Denny, 408/257-7000).

Customer engineering. The CE's are responsible for preventive maintenance and repair of the system. You can see that their organization structure is district CE manager, regional CE manager, customer engineering service manager (Tom Lauhan, 408/257-7000).

Every member of this team has organizations supporting that team. For example, if you ask your sales representative a question about the product (the price, the technical features, specifications) that he can't answer, a sales support organization within each division assures that he gets the answer to you on a timely basis, typically in a couple of days.

The systems engineering personnel also have support organizations. If you ask them a technical question that they cannot answer, they have an organization at the factory that they may call to get the technical answer for you.

Customer engineering. Customer engineering has a series of support mechanisms. If the customer engineer cannot resolve your problems, he has an immediate management backup team, but he also has a product specialist supporting him. If that team cannot resolve the problem, the divisions have a support mechanism as well.

Why is this important to the users group? Why is it important to you as the customer? When you are talking about having a new feature added to my fortran

compiler or a new feature added to my cobal compiler, this is a development project. The divisions are responsible for all development. The users group has set up the right channels to discuss those subjects with us. Your interface committee definitely does have a good channel in the division. If you are concerned about operational problems or you would like to give us operational hints of how we can do a better job for you, the field marketing is responsible for assisting on these subjects. If you have a question about the level of support in a particular geography, the field marketing team is the right people to interface with.

I thought I would spend most of my time now talking about two specific enities of the group. The first is Customer engineering division or Customer Engineering Service. Customer engineering is a service that our customers will always utilize. If you don't utilize these services either through service contracts or a time and material basis, most potentially your system will not function correctly for you. So by definition, all customers will have a service contract on a 3000 product now.

Recently we looked at our customer engineering organization and said we would like to put more management focus on it and get some more research dollars into doing a better job for our customer. So we formed a division. The major advantage here is, as I said, is this is Hewlett-Packard's way of focusing in on solving problems. It is the first Services division at HP. It is divided into marketing, engineering, and manufacturing. Marketing is responsible for setting the policies so HP has consistent policies, pricing, CE support. Describing the services to our cusomters so you understand them and your expectations are not raised beyond the level they should be. Additionally, logisitics are of concern to marketing. Where do you put these people? Where do you put the inventory? Where do you put the tools? When do you train them?

The engineering department of the division is the science of reliability. Development of CE tools. Customer engineering tools (software or hardware) helps HP do a more effective job for you. Manufacturing is really the field located customer engineering organization. This includes both the people and part inventory to help you locally.

I think it is important to understand that the impact of this division won't be seen by the customers for a year. Initially, in the first six months what you will find is that HP may have clearer policies. But, new customer engineering tools to help us do a better job for you is a development project; and it is like any development project, it is going to be some time before you see the benefits of this program.

When do you use customer engineering service as a customer? Call customer engineering if: 1) a peripheral does not work properly; 2) the system halts or a parity light is on; and 3) when an operator can no longer control the system. Does everyone know their local customer engineering managers? Please try to meet this individual over the next three months. How many of you know your regional Customer Engineering Manager? We have five regions: Midwest, Canada, West Coast, Southern, Eastern. If in an event you become extremely dissatisfied with Hewlett-Packard (I am not suggesting that you do this as a normal operating procedure), please tell us about it, and don't wait until it is too late. When management knows early enough about your situation we can help.

If as you are going through the year, you find you might be a little bit dissatisified with the overall service program. It isn't that your system has been down, but there are some things you can point to. For example, maybe we don't always put the bolts back together right, your lights are burned out. When the customer engineer comes around, mention it. The immediate responsibility in your area is

the Customer Engineering District Manager and he is an individual you need to get to know.

How is sales involved in my service problems? Sales sold this equipment. The philosophy of our company is that we service things because we sell them. The regional CE manager has a dual responsibility to the regional sales manager. But when you have a specific down system, the motivating enity to come up with game plan and to make sure your system is operational within a reasonable period of time is the customer engineering organization.

Supporting the customer engineer is the Products Specialist team. The product specialist's job in customer engineering is to be highly software intensive on MPE. By definition, if your system halts, it is an MPE failure. There are two types of failures. 1) Hard failure. You found one that is really a hard one. You try a warm start. It will not come back up. It could be a hardware failure at that point. It could be a very hard software failure. In that case, call in customer engineering. Regarding software failures, we immediately assign a design engineer in the lab to resolve it.

The second failure mode is when it is intermintent software failures. They are very complex to find. If you left your system down and called our customer engineering, the chances are there is not enough information left at that point to solve that problem today. Instead, please use the warm start mechanism. The objective is to take some data from that particular problem, get you back up on the air into a production environment as soon as possible. And, we try to corrolate that data at the regional levels. Now, if you can imagine, we get a number of these failures, we corrolate them, it gives us symptoms, then we can get to the specific portion of the software that is failing.

When you have a repeatable MPE error, it crashes your system, it is to our advantage that you make sure that we know about it. If it is repeatable, we can find it at the factory and fix it.

I think it is important that you understand where we are. We haven't done a 100% job. But, do not tolerate an extensive down period of time. I talked to a gentleman last week who said a portion of his system had been down for eight weeks. That is intolerable. There is absolutely no other way to put it.

System engineering. I am going to talk about the system engineer. If you recall my discussion on customer engineering, briefly, I said that basically it was a service that you would all want to buy and couldn't do without. System engineering services are different. We do not just help you to be successful by training your staff, helping to make them more productive. Our objective isn't to hide from you our system capabilities or to keep you unaware. In fact, our objective in this area is that we expect you, as a customer, to allocate one of your people to become, in effect, your personal system engineer. What that means is we need to train your people to become talented technically. The objective of System Engineering Services is to train your personnel. Requirement for this service varies from customer to customer depending on what your 1) specific application is and 2) specific talent is. So, it is what I call a discretionary service. We provide the opportunity for you to get your staff trained, but we can't tell you what services you must take. If you don't take the training, we assume basically that you are so talented that you don't require them. Planning System Engineering requirement at HP has not been effective. This is my personnel chapter. Today we roughly have in the North American continent fifty-five 3000 system engineers. We have done a lot of hiring to get to that number and they are not all well trained. But, they can provide the basic set of services typically required.

Customer training in a technical center. What is a technical center? We have reestablished the roles of the technical centers, or what you might have called previously the training center. Their roles will expand over time and provide a complete set of services. A developmental program is under way which is to help our customers learn quicker and more extensively about the system in our normal training center environment. Advanced courses you can send your staff to. They are not here today, but they are going to be here within three months. Technical centers are those places that the training is going to be available at. Another method of taking a course is you may purchase the course to be taught at your facility. Unless you have a staff of five or ten people, this may not be a cost effective way for you to get training. We are looking at alternative ways to make it easier for you to train your staff.

What is phone-in consulting? I would like to think of phone-in consulting as admittedly our training courses are not going to answer all the questions, and our documentation is not going to answer all the questions. This is a vehicle that we tried to design, to come up with, to augment training. I believe it could be a valuable tool if you use it. As an example, your staff is stumped on a problem for four hours. And, if they could pick up the telephone and get an answer within an hour, they could be productive again. Now, we would like that to happen. I believe that this service is a reasonable tool for you to utilize. What I said initially was that your going to end up with a system engineer on your staff, eventually. At some point in time you will be self-sufficient, theoretically. Eventually all of you should, if your program is successful, not have a requirement to call more than three or four times a month.

Now, I said over time you would become self-sufficient. The fact is, that is not really true. You have proven, beyond a shadow of a doubt, that you will not

be self-sufficient. You promote your people, move them to another installation, and suddenly the same training problems reoccur. It is a real problem.

System Engineering time. This is just your ability, if you prefer, to utilize our SE's onsite so they may assist your staff. For example, how to potentially design a particular portion of the data base you have decided to implement.

The training material and tools. I believe we have to provide you with access to training material and tools, like video tapes in the future, that will help you train your internal staff and help us train you.

On the subject of subsystem bugs. I think that is a point that I heard a lot of conversation about. And, sometimes it is confusing. Our experience has been that the majority of the time that our customers feel they have a subsystem bug, it really is not a subsystem bug but they really have a misunderstanding of our products. It turns out that about fifteen percent of the time there is actually a bug. Probably another ten percent of the time, a total of one-quarter, it is a documentation error that misled our customers. When that happens, a system engineer is responsible in assuring that that bug is resolved. Where are we today? I think it is important to understand that today our program is running very lean. What that means to you, it may require that when you have a subsystem bug, you have to help us in ways. The way we define the subsystem bugs, I would like to ask for your help in two categories. If it severely impacts you and your operation I want it resolved as fast as we possibly can respond as a company. To give you an example, we have resolved these problems in three to five days. That has been redesigned and back installed. However, if it is a simplier problem which does not cause you harm, please avoid using that portion of the software if possible. If you help us determine those that are critical to you, we definite have the resources to help you.

A situation that could occur is as follows: we delivered a mit tape with the objective of making a better, more reliable system. What we actually did we accidentally introduced a new error which affected all the production programs. What should you do as a customer in that situation? One thing you can do is go back to the previous mit tape. Don't be afraid to tell us that it is critical. I don't want you to do a workaround for fifty or sixty days and you are required to undo all those workarounds to utilize the patch.

If you have any inputs that will assist SE programs, I would like to have them. My address is as follows: Jay Denny, System Engineering Services Manager, Computer Systems Group, 11000 Wolfe Road, Cupertino, California 95014.

HP has now provided you a total solution for APL. Final stages of product development for the terminal entails actually putting it on the price list and making it available. This will occur in November and delivery will begin soon thereafter.

The HP 2641A APL Display Station, interfaces similarly as our other 2640 terminals to the 3000. It was developed in conjunction with HP Labs and some of the other major APL users around the country. We have the terminal in the lobby and after this we will show you some of the key features of the terminal. For those of you who are not familiar with APL terminals let me tell you a little bit about the HP 2641A. It has a 128 character APL set, a full 64 character overstrike set and 64 character Roman set which is optionally expandable to 128 character Roman. Also, it will support one optional character set. What that means, is that it will support one of our three alternate character sets. These are the line drawing character set, a mathematical symbol character set and the large character set, which was just recently introduced. With the large character set you can type characters on the screen that are about three characters high and three times the width of characters that normally you see on the screen. It operates up to 9600 baud as a standard ASCII teletypewriting terminal or as an APL terminal. Additionally it has all the features of the recently introduced HP 2645 High Performance Terminal that features multiple and standard communications protocols. Also, it features eight user definable soft keys. These eight function keys have been designed so you can execute up to an 80 character sequence from one key. You may ask what is the benefit? One of the key benefits is that you can tailor the terminal to your application. For example a whole computer log-on sequence can be stored under one of these keys. There are many other benefits of this terminal that you may want to see and I would be happy to show you after this talk.

Price? A lot of you are going to be surprised. It is only $4100 without tapes, with tapes it is $5700. It works very well with the

3000 system and with other systems that support ASCII APL terminals. Another of the key features of using the 2641A with the 3000 APL system is that when you log on to the 3000 many of the frequently made system calls such as variables, or functions will be automatically loaded under a soft key. This happens to be a very useful feature; and one that you can change at any time.

Overstike. I would like to talk a little bit about this. When overstrikes are performed we access an alternate character set and display the actual overstrike character. They are very clear and well defined.

It will also display an "out" character, when you try to perform invalid overstrikes. What this means is that with the 2641A you do not have to wait until you transmit this program to see your error. You can see and correct it on line.

To date we have tested the terminal on a number of non-HP timeshare systems and it has worked perfectly. Are there any questions? I will be happy to answer them.

Q. ON THE QUESTION OF UPGRADES, WE DO HAVE 12 2640'S AND WE INITIALLY HAD ORDERS FOR 12 2644'S. WE FOUND IN OUR APPLICATION THAT THE 2400 BAUD RATE HAS NOT BEEN FAST ENOUGH, SPECIFICALLY FOR DISPLAYING SCREENS AND WE HAVE APPROACHED DIFFERENT WAYS SUCH AS DRAWING PORTIONS OF A SCREEN AT A TIME, TRICKING THE SYSTEM AND USING A SCREEN ALREADY UP AND ERASING SPECIFIC FIELDS OF IT. WE WOULD LIKE TO HAVE 9600 BAUD. WE UNDERSTAND THE LIMITATIONS OF THE 8080 CHIP VS. THE 8008 FOR MICROPROCESSOR. ARE THERE ANY PLANS TO UPGRADE A 2640 TO A FASTER PROCESSOR TO ALLOW A 9600 TRANSFER?

A. No. We are currently offering the 2645A which operates to 9600 baud.

Bill Foster

Bill Foster of Hewlett-Packard's General Systems Division was the first company representative to address the group. The topic of his presentation was "The Future Directions of HP3000 Software." This includes the research and development of 2000 and 3000 software trends and to some extent the hardware trends. At the beginning of his talk he mentioned that the user requests for improvement, correction and development would be seriously considered by HP (he was talking about the results of the Users Group questionnaire). Specifically, HP would consider the top 20 items on the list, but would also consider the rest of the list from an "ease of triplementation" point of view.

Getting back to the topic, Bill stated that MPE was HP's greatest accomplishment. He said that HP would continue to enhance and improve it. Some of these improvements would be in the from of new commands. Another area of improvement would be the implementation of firmware or micro-code instead of software for some of the commands. He used the HP3000 Series II as an example of some of these ideas.

A major part of Bill Foster's time was spent talking about the Series II. He discussed some of the differences between the Series I and the Series II and the reasons for these differences. Some of these differences are: a larger memory, a semi-conductor memory, a larger virtual stack area and the expansion of the number of processes allowed to run at any one time. These improvements increased the speed and reliability of the Series II machine.

The Series II has the capacity to expand the memory to a full 512K, and it is made of semi-conductors which allows the memory to be self error-correcting. A log is kept of all memory errors and isused to point out problems before they become major. The largest memory reduces the time spent swapping and thus increases the overall effectiveness of the computer. The memory management system (uses of CPU time) also add to the performance.

Finally, Bill stated HP's objectives as being: to decrease the price of its products, increase the performance of these products, increase their reliability

(both hardware and software), increase the abilities of MPE amd IMAGE, and to build interfaces to other machines besides IBM.

NOTE: Tim Robbs attended the HP meeting as a recorder. The above is a summary of Bill Foster's speech by Mr. Robbs.

Many of you know me as one who likes toys and that at each of these meetings I bring a toy. This time I brought a new toy and a new system. The new toy is the large screen video system that you have seen in use during these sessions. This vid o system is used at Boeing as a training aid and as a "show and tell" service for visiting firemen who come to see the system which I will present shortly. We will soon place the large screen system in our management information center as a management display tool.

The large screen display system consists of an adds envoy, a portable 5" CRT that has composite vidio output under the RS170 spec, and a videobeam projector. The videobeam projector is a color TV set that projects onto a 4' x 6' screen and has connections for a TV tape recorder. It is sold to taverns and to the guy who has nothing else to do with $4000 than buy one of these for his living room. We did have one problem with the thing. It is designed to work as a TV and not as a terminal. Therefore, it is setup with overscan rather than under scan as is found on computer terminals. The solution was for us to realign the videobeam. No small task as it is really three TV sets. One for each color, red, green and blue. We also had to go in and remove one circuit card and bypass the card as it buggers up the whole image when data is displayed.

We made one other modification this one to the terminal. Our presentations work off carriage return, so we added a remote carrage return key.

Now to talk about the system we developed at the Electronic Support Division of the Boeing Aerospace Company in Seattle. On screen you see an agenda for this presenation.

Let us look at the corporate structure of the Boeing companies. All of the Boeing are wholly owned by the Boeing Company. As is the company I work for, the Boeing Aerospace Company. Some of the other companies are the Boeing Commercial Airplane Company, Vertol, Boeing Computer Services, and others. The Commercial Airplane Company manufactures 747's, 707's, 737's, and our most popular airplane the 727

which we are currently producing one every other day. The Electronic Support Division, ESD, supplies electronic systems for these airplanes. The Commercial Airplane Company is our largest customer at ESD. Another item of interest is that we are a customer to BCS and BCS is a customer of ours. The Electronic Support Division of the Boeing Aerospace Company is the location we developed and installed our manufacturing system. ESD produces electronics packages for its own products and for other Boeing companies.

Now for a brief description of our hardware configuration. Our HP3000 is a CX which we use very heavily in a multi-user environment. We have a 7905 swapping disc, 3 ISS 47.5 mbyte data discs, two 1600 BPI tape drives, a 200 LPM printer, and two 16 line terminal controllers with 30 lines connected. Seven of those lines are dial-up and the others are hardwired.

We built our system around certain philosophical ideas. The system must be online as much as possible and any data input will be edited online in every way possible. Data entry is not just that, but includes edits for known spelling, usage in the present context, etc. We want to tell the person at the terminal about the error right then, not some time latter, because at the moment of input that person can usually correct the error. If we cannot correct the error we do not want the input until he knows the correct input. Our idea is to check a partnumber for accuracy at the moment of input, and not after a screen is filled. You have wasted the time inputting the balance of information.

In designing the system, we started with detail flow charts and then started programming in fortran. The PMS system is a heavy user of image with some direct access files.

Now to the modules that comprise the production management system. The first module is a master schedule. The schedule that shows when we will deliver those part numbers or assemblies that are known as end items, or they might even be purchased parts in the case of spares. The master schedule lists only the part numbers we are going to ship and the shipment date. No schedule file is maintained on sub-pieces.

This is handled by MRP.

The bill-of-material module contains that information relative to making an assembly. Nothing extra. It is basically what I call a pear tree. A record consists of the assembly part number, the component part number, and how many of that component is used on that assembly. We get into such things as assembly notes; but for the interest here I will not go into those items.

The inventory control module maintains such things as store bin location, quantity on hand, reserved quantitites, lead time, min, max, pansize, and all such buzz words.

Shop order control comes into play at the time a shop order is ready to be released. Shop orders are automically released unless there is a shortage of parts for that order. If there is a shortage then our production control group must decide whether to wait for the parts to arrive or start the order short those parts. The classic example is the knobs on the TV set.

Purchase order control. Here we keep track of the purchase orders we have requested. Their schedule, and the status of the purchase order for expidite action when necessary.

Material requirements planning. We use a regenerative MRP. A little later we will display in detail what happens in MRP. Basically, MRP combines the bills-of-material and the master schedule resulting in recommended schedules for all sub-assemblies and purchased parts on the way to developing schedules MRP will first consider inventory on hand and then combine orders to make economic order quantities.

Applied job routing and standards provides for the job flow and the information required for performance measurement.

Now let's look at MRP and what it really does. I was saying it works with an assembly and here we have assembly A. It consists of Parts C, B, and K, as you see on the screen. We will keep the definition of the assembly up in the corner for reference. Up in the right hand corner you will notice a schedule for one unit of assembly A as marked by the 'V'. So we know we must have an assembly available by

that point in time. And, of course, with reorder lead time we have to look forward and plan to have that order released at that point. Next, we have to plant parts to go on that assembly. Some of the parts are make and some are buy. Again we ask is the supply sufficient to satisfy demand? If not, we have to plan and order for those parts. Some are assemblies, and we go out and plan orders for their subparts, and on down the tree.

Now, that is basically what MRP does. It looks through the bill of material and plans all the orders needed to make the end item for the original demand. Of course, there are usually multiple demands spread through time. MRP must consider all of those demands and apply order rules to combine orders when applicable.

The system is image data base oriented and contains three image data bases and a set of auxilliary files we think of as a fourth data base. The first data base is our primary data base. It contains product structure, the master schedule, open orders, and recommended orders. There are 23 data sets and about 23 megabytes of disc space.

The second data base is information relative to current job charges. Five data sets, two megabytes and designed to work real fast. By using a unique data base for shop charges, locking conflicts are kept to a minimum and response time is maximized.

The third data base is used for performance on completed work. This data is used for performance reports, is not updated too frequently but can be used for management costing games with out impact on the production data bases.

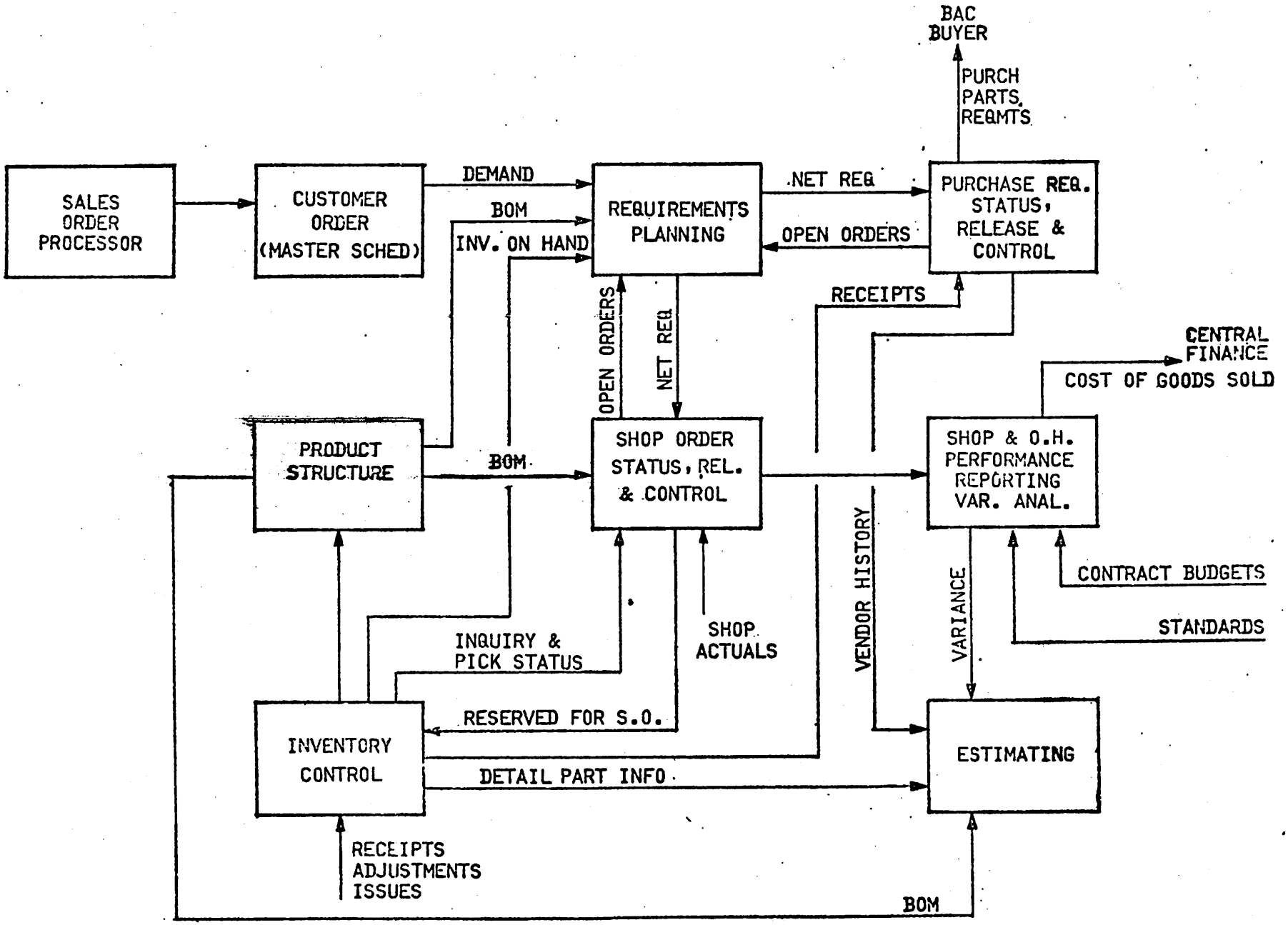
There are ten auxilliary files consuming about seven megabytes. They are direct access files. We use direct access files to minimize the size of the image data bases. The best example is the way we use an internal part number. We had to allow for a twenty-four digit partnumber. Now if we were to use a 24 digit partnumber in image our data base would be 60 percent larger than it is.

(There goes another 47.5 megabyte disc.) So inside the system we work with an integer number that represents the human recognizable partnumber. The integer partnumber is the disc record number corresponding to the real partnumber. To convert an

internal partnumber to a real partnumber takes only on disc read. To get from a real partnumber to an internal partnumber we use an image manual master data set with no related detail data sets. In this case, we take advantage of the fast hashing algorithm provided by image.

That is our system in a nut shell. I will now run a couple of our programs to demonstrate how we believe an interactive program should be written for an online environment. Any one who would like to see more or talk in more depth may contact me for a birds-of-a-feather session.

# ESD PRODUCTION MANAGEMENT SYSTEM



Jim Gunderson

I am Jim Gunderson, director of Data Processing for the Multnomah County Intermediate Education District in Portland, Oregon, to an educational service bureau in Portland. Telephone number is 503/255-1841 and my mailing address is P. O. Box 16657, 97216.

We have recently replaced an IBM 360 model 40 with two HP 3000's. As part of our analysis on the management of a multisystem shop, we came up with the idea of building some swithces for the I/O gear so that we could create an I/O pool. In one case, we wanted to share one of our three printers between the two systems. In a second case, a tape drive was to be shared between an HP2000 and an HP3000. Our specific problem was that we had an Op Scan 100 for test scoring, which produces an 800 BPI tape, and an IBM System/7 that we use for touchtone telephone data collection, which has a 1600 BPI tape. In a third case, we wanted to share the card reader between the two HP3000's.

Six years ago, we installed an HP2000 and subsequently a second 2000, each of which had one tape drive. One 2000 was upgraded to an access system. In that environment, we wanted to get by with only one tape on each 3000, and somehow use the 800 BPI tape that was on the 2000. We located an early-retired CE who agreed to build the tape switch initially. HP provided the cable schematics and we proceded. For simplicity, we used relays rather than solid state components. With pluggable relays, standard electronics, we took a very conservative approach and the switches have worked perfectly for a year. Physically, they are about 6 inches high by 12 inches wide by 18 inches deep.

We did the same for the card reader.

Finally, in order to allow our data control clerks to use either 3000 from a single terminal, we have had several smaller switches built for both the HP2640 and the DEC LA36. It is possible to log on to both systems and use the switch to access either one by simple turning a knob from I to II.

In an environment of multiple systems, it is very feasible to build these little devices in order to prevent loading all systems with all the I/O.

Switches are not particularly inexpensive. Furthermore, HP's CE's get nervous when they see the boxes sitting there because they are not sure where the maintenance is being paid.

We had to buy tape, printer, and card reader controllers for each 3000. We have a card reader controller on each system and a cable which comes out of the back of each controller into the switch and from there to the device. It is still significantly less expensive (perhaps 40%) than buying I/O for all the equipment.

The cables will run from $300 up to $1200. To get the switches built, we paid from $400 to $500 for the components plus labor. The terminal switches were $65 apiece. The man who did the work for us is available to build them for anyone who wishes. He is not particularly in business, but he has the resources and the capability to provide the devices. In addition, he can also test them at our shop in a live setting before he sends them on.

We would be happy to provide additional information.

Tom Harbron

This is the installation management forum. It is designed to be an interactive session. It has several purposes. I have had three people volunteer to work with me on the installation management committee, but none of them seem to be here at the moment. But, input from others who would be willing to make contributions in this area would be appreciated. As we go through the session here, I think you will get some feeling of the nature of the topics.

What I would like to do here in this session, is first of all, outline the mission of the installation management committee, solicit your suggestions and ideas along these lines. And, then in the time remaining, we could have a little question and answer session on topics of installation management. There are enough people here that we can find answers to most of the questions.

I would like to introduce Robert Young from the National Institute of Medical Research in London, who has volunteered to work with us. This is quite helpful because many of the problems European users face are quite different from those that we in North America have to deal with.

The mission of the installation management committee, as I outlined back in the newsletter of July, involves three kinds of activities. First of all, let me say this is not an exhaustive list and I am quite willing to amend it and add to it, as we go along. What I have outlined here are three kinds of activities.

One is meetings. We really didn't get a chance to get organized before this meeting. But, before the next one we can have some papers and sessions on installation management. If some of you have ideas and want to give a paper on that next year, please let me know.

The second thing is developing some standard operating procedures. There are a wide variety of things. To take one, for example, system backup procedures. How do you do this? How is it managed? And, we have had some questions on that here in this meeting. There are a lot of things like this that would be helpful

to many people to have; not just do it this way, but here is a variety of different procedures that are used by different installations. Take your pick.

The third activity that I think is going rather well, is the columns in the newsletter on topics that have been suggested, or in some cases that have already been produced, I have about a dozen of them here: backup procedures; connecting terminals, either hardwired or through modems; resource accounting, which goes into the question of billing and all of that; documentation; what to do until the CE comes; locality; virtual memory and structured programming, which are two different but related topics. I don't like the term, but I have not found a better one, vendor control, that which we have been discussing for the last hour is in this category. Hardware error detection on the Series II, which has some tools that were not available on the Series I; data design base. You take essentially the same programs and run it on data bases that are slightly different in design and it makes a world of difference in performance. Segmentation also heavily impacts performance. Those are some of the topics we have on the list anyway. I am sure there are a lot more that will come along.

Those are the activities. Now the subjects, and I am sure it is only a partial list: system integrity, operating techniques, hardware utilization, programming practices, system measurements, disc space management, data base design, etc.

The question is if and when we get replaceable disc packs, how should those be managed? I think we will have to have some information from HP on how it is going to be implemented before we can do anything on that. But, when the time comes it will be an important aspect of disc management. I can see potential problems in handling that. You take a pack off and a file a guy needs is gone, and why is it gone and what do you do about it?

I thank you for your input. It has been most helpful. I will thank you even more for substantial input in the forms of contributions to the newsletter columns, papers to be given at the next meeting, SOP's, or if you have any other substantial contribution.

Ed McCracken

After lunch the Users Group was addressed by Hewlett-Packard's General Systems Manager, Ed McCracken. He spoke on the "State of the General Systems Division." He specifically addressed the question, "Will HP survive in the computer business?" He answered that HP would survive and would strive to lead the industry.

Phase I of Hewlett-Packard's development began in 1968. This is the year that planning and development of the HP3000 started. HP invested heavily in SPL (System Programming Language) and in the MPE (Multi-programming Exeuctive Operating System). Phase II started in 1971. This phase was signaled by the introduction of the first HP3000's. This was the pre-CX 3000. Phase III, in 1973, showed an effort in the direction of general cleanup. Improvements were made in the hardware. The MPE and subsystems were becoming more reliable. In 1975 the HP3000 computer system made its first profits and contributed to the earnings of the company. In 1976 there were other important events. One of these was the introduction of the new Series II. The second was the creation of a separate division to maintain and improve the CX model.

Ed McCracken went on to list HP's general objectives for the future. The first objective is customer satisfaction. Hewlett-Packard will try to improve in four areas to keep their customers satisfied. These areas are: the general cleanup of the product specifications, increase in reliability of their systems, increase in responsiveness of the company toward the user, increase in effectiveness of customer and employee training, and production of better system documentation.

The second objective is to be increasingly professional. The introduction of the Series II is an example of this effort. The Series II has relatively few problems, and is much more reliable than the Series I. Before the announcement of the Series II, HP had 50 test sites in operation using the new computer. The MPE is compatible between the Series I and Series II computers. These areas show HP's attempt to offer the customer a more professional job.

The third objective is to make the required effort to meet sales, profit and shipment goals.

The fourth objective is to create a program capable of meeting all of the division's long range goals. This means increasing response to the users; to use MPE, which is considered a corner stone, to make a major contribution toward growth of the division; and to produce increasingly more powerful machines. It has been found that HP can produce a 30% increase in computing power without an increase in cost in their machines each year. The subsystems will continue to be developed and improved. The division is making an effort to develop its own disk drives. Hewlett-Packard will not produce a computer which is more expensive than the Series II. They will compete with the industry at this level.

The fifth and last objective is to develop a new division. Production capabili will be increased by a factor of three. The number of employees will double from 300 to 600. A new facility is to be completely developed at Santa Clara. Also the number of engineers in the field is to be doubled to provide better customer service. This effort is to make the division self supporting.

Ed said, "Hewlett-Packard's General Systems Division is manufacturing the highest quality systems possible, and is leading the industry in the market place." About 50% of HP's systems are sold to the smaller companies, and this part of the market is growing quickly. The rest of the sales are to the large companies for dedicated work or to O. E. M. houses. About 97% of all profits comes from sales of products and services and 3% from the sale of stocks.

The division has three profit centers. The first is the center headed by Dave Sanders. Among other things, Dave is responsible for the support and maintenance of the Series I machines. The second profit center is made up of the technical centers. These centers offer technical services and education. There will shortly be two of these technical centers, one on the east coast and one on the west coast. The third profit center is the customer service. The customer service will increase

their personnel and strive to provide better services to the customer.

NOTE: Tim Robbs attended the HP meeting as a recorder. The above is a summary of Ed McCracken's speech by Mr. Robbs.

Sex really does fit with the computer! I am thinking in terms of, perhaps, the secretary, perhaps yours, perhaps another. She walks over to the terminal where you have given her a task to perform, she sits down at that terminal, and prepares to enter some data. Of course, the first thing she does is snag her stockings on the inside wall of the terminal cabinet because it was made of very rough paint. She then attempts to log on. You had told her to log on with HELLO, MANAGER:SYS, slash, password. She is a pretty good typist but on the terminal she is a little shaky so she mistypes the password. What does she get returned? ERR8. It means your entry is not acceptable; your password is incorrect. But, why doesn't the message say that? There are other ways of looking at cryptic messages, too.

Let's consider her background. She is probably some nice young girl who is very morale, obviously attended church her whole life, minded her parents, yet the computer might have come up with some other messages using such words as ILLEGAL or BAD. Consider the psychological ramifications on this poor kid who has just sat down to her first job and you have given her this task. Rather traumatic. My favorite is the error message you get when at the console, no matter what you do if your input is not acceptable, the system responds with INVALID.

Can you identify the real problem? Sure. It is PEOPLEWARE. In contrast to hardware and software. For those of you who have known me for the four or five years that I have been associated with the Users Group know that my favorite topic is PEOPLEWARE. This is really the human factor or user interface. This area concerns all the same hardware and software

aspects of design, development, integration, production and maintenance. Some of the items involved are used in dialog at the terminal, user manuals, user problems and their handling, user support, user communication. Specifically, this involves psychology at the terminal. A hardware related example might consider the RETURN key. Why call it RETURN? Why not ENTER or SEND? Some people have put ENTER on the key because that is what the user does. Let's go one step further. How about coloring the key? What color should it be? How about green? How about the break key? How about red? Ever try to find that break key on the 2640? It is buried among several other keys; I always hit the wrong one. Or, why not use a two-color ribbon? For hard copy dialog you could distinguish between your input and the computer output. This idea makes it very simple to see "who" has done what. Of course, if you suggest that to some manufacturer you know what they are going to come out with--red and black. And all your input is going to be red; how about green and black. Why don't manufacturers take the human factor more into account? Well, there is a simple reason. First, it is expensive. If you recall, an early military system, something I started out with in computers, the SAGE system. The Semi Automatic Ground Environment was supported by the US Government and that is probably why they had enough money to implement human factor efforts. For every three programmers there was one psychologist on the staff to sit down and analyze those people who were going to use the air defense display system. Those airmen third class didn't want to know how the computer ran, but they sat at this silly thing and had to punch the ring button at the right time. The color of the lights, the seating arrangements, the location of the buttons, were all done by the human factor personnel who also generated simulation exercises.

Furthermore, the typical manufacturer of computer systems does not

understand the user or the user problem. He does not employ human factor analysts or psychologists, generally. Yet the most important element of the computer system is PEOPLE. Without them there is no need for computers. Right? What I am really saying here is that I feel that what we need at this point in time is to make the computer bend to the needs of the user not bend the user to the needs and requirements of the computer. Certainly the manufacturer's concerns for hardware and software are essential. I wonder really what a user is to HP? You might ask Ed McCracken.

But what sells computers? The answer is simple--hardware and software. Those of you who are managers, what do you look at, the hardware specifications. What will the software do for me? But, unless the user, the user sitting down at the terminal, can be a success; and that is the key to our world here, success; with the computer, simply and easily, the most sophisticated computer system in the world is of extreme diminished value.

When you are at the terminal what is the mean time between confusing, misleading, nonexistent responses from the computer system? Or, for that matter, HP? How often do you spend extra time trying to find out what the problem is? What really went wrong? When information is not in the manual, you guess by trying this and that. My engineers spend a lot of time sitting there and going through lemmas to determine just how that FORTRAN statement really works. Consider the error message "parameter in error." Why didn't they just give the parameter in error? There is a whole string of parameters in FOPEN, 15 or something like that. How do you know which one is in error? Why not point it out to the user?

The data processing situation right now in the world is as follows: hardware costs are going down for sure. Bill Foster said the other day that software costs are going up. HP is petrified at the amount of software and they

don't want to write the users' software. It is expensive. The use of batch processing is going down; the use of interactive processing is going up, e.g. time sharing, online. We all have HP3000 computers--interactive processing is one of the basic elements of the HP3000. Sit down at an HP3000 terminal and do what you want to do. Yes, it can process batch; but, I think most people here will use the HP3000 more interactively.

Up at the Birds of a Feather session last night we were talking about terminals across the nation feeding into a computer, all interactive, online, ready to go. The problem as pointed out by the Hughes Aircraft Company DP manager the other day, sort of hit me. He was saying the problem is not the cost of hardware, at Hughes we have two IBM 370/165 computers with three megabytes of core to fill this room and another room full of mag tapes and terminals all over the place. What he is saying, and I believe him, is that the cost of the software development is going up, but one of the highest costs is the transfer of information from the user to the system and from the system back to the user. That is the area User Interface.

What is the computer manufacturer doing to enhance the user interface? In the future more noncomputer oriented users will be interfacing directly with computer systems. They will be scientists, managers, housewives, and kids. They all want results. They want to perform their tasks <u>easily</u> and <u>simply</u>. The thought process, that process that goes on in the mind, for the end user is different from the computer scientist. We here are generally computer science oriented. That user out there is not! Computerese should not be necessary at the terminal. Besides, why not make it easy for the computer scientists? Should I really have to guess if the numbers output by a PMAP are octal or decimal? Why not at least put a percent sign in front? Even though I am a computer scientist and I have a master's degree in

computer science, doesn't mean that I magically know all the magic things that you output from the computer. Make it easy for me. Make my job so I can get my work done.

Shouldn't the computer system designers consider what the needs of the user are? Who is doing research and development in human factors today? How many manufacturers are actually having people do work out there, doing research, taking a look at the end user criteria? Are human factors a real high priority concern for manufacturers? Certainly not for most hardware manufacturers. Today I feel we do not really have friendly computers, let alone forgiving systems. Friendly is being able to come up with good vocabulary so that users can easily deal with computers; and forgiving, in the concept that I make an error and the computer doesn't return with abort, percent sign, and garbage. How do we get there from here? It is clear to me that some human factor analysis needs to be performed. Sure I realize that most manufacturers do not understand the tasks. It is only natural that the most knowledgeable person to perform this is the user. The implication is good, frequent, free flowing, information between user and manufacturer. No longer can systems design criteria not take into account the human factors. No longer can manufacturers generate design specifications in a vacuum. Manufacturers must swallow their pride and break the NIH barrier. NIH; not invented here. And, heavens, heavens of heavens, the manufacturers must make, must initiate personal, direct contact with the user to, watch it gentlemen, to see what his or her requirements are. On the other hand, the user needs to make the manufacturer aware, aware of the problems at hand and help other users get around the current problems through dialog, through documentation.

How do we implement this approach? The Users Group certainly provides one medium for user exchange with user and manufacturer. Another medium, and I think probably of equal importance, is the Newsletter. You are going to have to spell it out and contribute your ideas, your problems, your techniques, your solutions. Nothing in the computer system is obvious, no program is too simple. Bill Bryden and I were joking about that the other day. "I will just whip up this simple program," he said. Three hours later he was still debugging and it was just a simple program. It is difficult to get from here to there.

Only through communication, from you the user, the person most closely related to the computer system, can we, I mean we--user and HP--become a success in an easy and simple manner. In another of the Birds of a Feather sessions last night, one of the fellows was talking about a COBOL routine or a COBOL statement and the code generated. He was talking about the inefficient code the compiler generated. He knew exactly that this statement generated inefficient code and that another statement generated much more efficient code. That is great, that is good for him, but what about all of you out there? You are going to have to put it in a Newsletter, you are going to have to express it in meetings here up front. HP cannot do that for us. We must do that. The user has the responsibility to fully describe his problem. Not only to describe his problem, but to justify them in terms of both his needs and economics then generate as full a solution as possible. Just yelling about the problem is not going to get the solution and manufacturers are not going to pay attention unless they have a real understanding of what that user need is and how they can apply the solution to meet other user's needs. Very definitely economics is HP corporate objective number one. You MUST make a profit.

Your suggestions cannot be used by the manufacturer unless they make the manufacturer a bigger success easily and simply. More important your description of how to use a system, techniques, application solutions, learned information about the system is invaluable to other users, and ultimately your success. We have got to be careful about our relationship with the manufacturer. We cannot just beat on the manufacturer. Truly the greatest burden is on the user. You must be thorough in your analysis; you must site good examples, provide good solutions, provide good alternatives. There is not only one road to these problems for a solution; they are not answers; they are solutions. You must write good documentation. I know that hurts. Most programmers, most EPers do not write. Why, I do not know. Certainly there is some fear to commit themselves to paper. But, documentation has got to be learned. In fact, it seems to me that if I had a lot of programmers under me, one of the first courses I would send them to is a writing course--not computers--writing; how to communicate! When they have a problem they must be able to put it down correctly.

Second to last, we have got to gather concensus. That is what our questionnaires do.

Of course, last, we must be reasonable and businesslike. I realize that this is very frustrating. It is very frustrating for me at times and I do get carried away in terms of not being businesslike. But dealing with these manufacturers we must approach them in a reasonable fashion. Likewise, the manufacturer must realize that we are business people and approach us in a straightforward and honest manner. If you do not do these your good energy is wasted and you only perturb the manufacturer. Throughout the relationship I have had with Hewlett-Packard, I have gained a great respect for that corporation. I have been very personally involved with it through the

Users Group. I have a high respect for the corporation, its people, and the way they handle themselves. I think they do it a little bit differently and I think that makes a difference. Their aim is most definitely an aim for quality. We can say how come you don't do it like DEC does; there are reasons HP is a higher quality corporation in many ways.

Well, what are our conclusions here? Significant progress, advancement of our computer system is best made with full cooperation, understanding and easy communication between users and users, between user and manufacturer. Until the perfect system is generated, we must accept the current status of the system. Only you can expose the real user techniques, problems, and required solutions. That is you must grin and bear it.

Larry Meyers

I am going to run through this much faster than I had anticipated. I am going to give you the background of an HP3000 application in an environment that I think is foreign to most of you, a jail. In fact, the largest single correctional facility in the world, Cook County Jail.

The system we are going to talk about is called CIMIS--the Correctional Institution Management Information System. It is a decision making tool that will be used by everybody from the warden, for making major policy decisions, down to the line officers who make the thousands of decisions every day controlling population movement.

The system accomplishes this through a series of data base inquires and modifications called transactions. This raises the first topic that should be of general interest to you. We are going to talk today about transaction processing on the 3000. We never intended to put 192 terminals on the system, but Gary Green liked that number so much that he included it as the title of our talk. You are going to find out later that you could have 192 terminals on a 3000, or even more, depending upon the environment in which you are processing transactions. We will have a minimum of 41 terminnals on-line simultaneously, growing to between 60 and 70 by December of 1977.

One of the major things we are going to try to do here this afternoon is make sure you understand what we are talking about when we say "transaction processing". Alan Dale is going to deal with that in a general sense which, I think, will be most interesting to many of you.

I am going to give you an example of one specific type of processing that we are going to do on the 3000 to use as a benchmark for conceptualizing potential uses for this technique in your own shops. I am going to begin by explaining, very rapidly, why we are doing this and how we evolved to the 3000 and the SRI front end/message control system to be described in a minute.

In the fall of 1973, two events occurred that placed us on the road that lead here today. Firstly, a new administration came into Illinois after the 1972 elections. They had just completed a period of transition and familiarization with the headaches that they had inherited in state government. They had their set of goals and objectives upon which they had campaigned and more than a full measure of problems and frustrations particularly in the area of criminal justice.

The second event was that David Coldren and myself were just finishing working on a $2 million U.S. Department of Labor program to improve the quality of services, particularly manpower services, in the State's correctional institutions.

Dave described our emphasis in Illinois on operationalizing justice. To reiterate, that means trying to achieve equal treatment in the criminal justice system, making decisions based on fact, on the record, and subject to review. That is the Justice Model.

The experience that Dave and I had in corrections indicated that in this particular criminal justice system component, decision makers were not receiving the kind of information that supported good decision making in terms of what we thought justice model objectives should be. Dave and I suggested an information system be developed and made available to the widest possible audience in the custodial environment with the emphasis on line staff and their decisions affecting living locations, work assignments, disciplinary actions, and the like. In custodial environments, such issues are critically important as evidenced by demand lists presented by inmate groups involved in prison disturbances before and after Attica.

Yesterday, with groups of users cheering them on, a few people suggested that HP misrepresented the number of users that could be adequately supported on a 3000. We will have over 40 initially, as I said, and it is ambitious task. But, it wasn't the first thing we tried to do.

In the first effort, we wanted to determine if it was possible to implement anything that involved line staff directly in an information system targeted toward decision making. Line staff in a major Illinois penitentiary reported entry, exit, assignment, and housing changes to a three-person CIMIS staff who updated a system of files that constituted an end-of-day picture of the institution, that is what it would look like the first thing in the morning when they unlocked the doors. We called this the "snapshot approach". From these files, reports were generated at night and distributed to staff the following morning. Our objective was to give line decision makers all relevant information that related to their particular areas of concern. For example, the housing committee at Stateville Penitentiary had a complete housing map that was produced daily. They used this to balance the population across cell houses while considering issues of security and specific inmate proximity.

This "snapshot approach" operated effectively, but at significant dollar cost, with data base changes formated off-line through an intelligent terminal and processed in batch through the resources of a service bureau. This is the schematic (slide) of the computers that were used at the McDonald-Douglas Automation Company. Access was, at this point, limited only to CIMIS personnel and the update cycle occurred once daily.

In our second effort, we sought to maintain more than the most recent "snapshot" of the institution so that the top management could periodically review the decision processes of line staff. A monthly management review cycle was implemented in which a series of "snapshots" were retained in a "photo album model" via the use of a mini computer in a transaction processing environment. This environment, which I might add, effected a 90% cost savings in data processing charges vis a vis the service bureau, used screen formats to access and edit online records of any particular inmate. The primary role of the system, however, remained that of disseminating information to the decision makers through daily hardcopy reports.

In the initial mini-computer configuration, access continued to be restricted to CIMIS staff while updates throughout the day were made possible at a marginal cost of zero. The once daily report cycle prevailed for some time. We used IV-Phase System IV/70 hardware, and these systems are still in place in two Illinois institutions.

The next step in the evolution was to change the role of the system from that of an institutional mirror to the operational control mechanism that we had always intended it to be. With CIMIS, by this time a proven product, the institutional managers took the next step of placing terminals in a control center environment so that all decisons effecting housing and assignments were cleared through CIMIS before actually being implemented. A cycle of daily movement, planning, and control began to take shape. The concept of planning in a correctional institution was unheard of to this point. No longer was access limited to system staff. Rather, line staff maintained up-to-date files from a single control center throughout the day. For example, at Menard Penitentary the security shifts do not change until the institutional headcount, by gallery, is reconciled with the headcount in CIMIS. Since the guards in Illinois are unionized, that consitutes about 300 people being paid time-and-a-half while awaiting shift change. Additionally, officers become extremely irritated when required to stay after completing their eight hour shift. The cost of bad information in the system became significantly greater.

Retaining the thirty day management review capability (the "photo album model" that I showed you earlier), file updates began to occur with increased frequency edging the CIMIS system into what we call the "moving picture model" of institutional status in which hundreds of "snapshots" of the institution are recorded each day yielding a dynamic picture of institutional movement and decision making.

That is our history.

It brings us to the status of CIMIS currently under development on the 3000, real time inmate tracking in the Cook County Department of Corrections. Remember, in our parlance tracking is defined as the thousands of population control decisions made daily in an institution, the analysis of which forms the basis for all management decision making. Far removed from the active, but fairly stable environment of a penitentiary, the Cook County Department of Corrections is perhaps the most fluid correctional population anywhere in the United States. We have 60,000 people booked annually, many of them every Friday night. We have about 400 people going to court every day. We have an average daily population of 6,000. Now you might say, New York has more people than that in jail, and they do. But, they have eleven facilities spread all over the five burroughs. All of the facilities I will show you in a moment are on a contiguous fifty-six acre site. We have substantial scheduled movement to work assignments and court (since a jail is primarily a warehouse for court in which ninety percent of the 6,000 inmates are unsentenced). We also have significant unscheduled movement such as, lawyer visits, family visits, and the like.

In the summer of 1975, a rash of escapes at the Cook County Department of Corrections prompted investigations by the Sherriff's Office, the press, and citizen's groups, resulting in charges of inadequate population control, poor middle management, and poor administration and planning. Our agency was called in and responded with a package of five managemet support programs, ranging from an internal affairs division to a middle management development program, with the keystone in the program package being CIMIS. I want you to make no mistake, as you might begin to think that perhaps we should be locked up, we embarked upon this project in an atmosphere of crisis. In Chicago today, the only person whose picture appears in the paper more than Mayor Daley or Snoopy is the Director of the Cook County Department of Corrections. The Department is a "hot" issue.

Given the success of the penitentiary based CIMIS, and given the R&D role of ILEC/CJIS described earlier, we were given the task of tailoring the CIMIS system to the Cook County Department of Corrections. Specifically, we were told, track inmates real time, produce periodic and on demand management reports, and monitor staff adherence to procedures. This last point is particularly important as it is a direct reflection of decision making, and it was both the major criticism and defense of the operation of the Department of Corrections. Critics claimed no, or at best, vague procedures existed to guide staff in their decision making, and the administration claimed that it could not adequately monitor staff adherence to its established procedures.

The Cook County Department of Corrections, as I have shown, is composed of five contiguous divisions distributed over a fifty-six-acre site. The numbers on the slide refer to the different divisions. The first being a male division, maximum security, bonds of $5,000 and up. Division number two represents a male a division for less serious offenses with bonds below $5,000. Division three is women. Division four is a new work release center and an overflow area for division one. The three crosshatched areas (slide) represent buildings now under construction that will add another 1800 beds by December 1977. This is our growth requirement for terminals. Each division is separately administered but all report to the director of the department who is located onsite.

To develop the notion of inmate tracking as a transaction process on the 3000, and to hopefully make the general description of the message control system more meaningful, I would like to present a simple illustration of a CIMIS transaction on the 3000. To do this I have to define three terms.

The first is sphere of control. Paradoxically, the more narrow the sphere of control in a custodial environment, the safer the inmates, thus enhancing the freedom you can have in the atmosphere of confinement. There is simply no justice if a correctional institution is less safe than the most dangerous street corner in Chicago or the Baltimore Hilton. Examples of spheres of control at CCDOC are a tier (sixty cells and a day room through which inmates can move during the day), or work assignments, like the bakery, in which the supervisor has the responsibility for a crew of inmates. A sphere of control is an area in which inmate movement options are limited, but not totally restricted, and in which specific line staff members or a single member have custodial responsibility.

two or more spheres of control. These points imply a change in sphere of control with, and this is very important, rules or procedures to govern such changes, personnel authorized to make such changes, and a given line staff member to carry them out. An example might be moving an inmate from his tier to the dispensery past certain check points. When an inmate passes an accountability point, his location and status are altered in the CIMIS files, which are maintained in an IMAGE data base.

The final term is administrative area. These are primarily records offices which effect changes in the inmate's status by scheduling court appearances, recording changes in bond, and the like.

Our analysis of the Department of Corrections' spheres of control, accountability points, and administrative areas led to the terminal configuration shown on the slide. You see forty-one "T's" with the numbers indicating the level above ground. You will note no terminals are shown in the three buildings still under construction. It has not yet been made clear as to how these facilities will be administered and operated.

As I said before, the jail is a warehouse for the courts. Now I am going to illustrate two transactions that are going to be typical in CIMIS, relating to the daily court call process.

Each morning, men are "dropped" in accordance with the scheduled court call from cell blocks that are composed of four tiers connected by a stairway that runs into a subterranean level called "boulevard". There is no other route off these tiers. "Dropping" simply refers to moving the men from the tiers to the "boulevard". A "boulevard" officer calls for a single cell block at a time for one of two holding areas. The first holding area is for the criminal courts building which is connected to the jail by a tunnel. The second, since we are a centralized holding facility for the whole county, is used for men awaiting vehicles for transportion to suburban or downtown courts. For example, if ten inmates from A Block are to be dropped to the suburban holding area for transportation to the suburbs, the boulevard officer at this post will type on his terminal, "/Drop, A,S". That means that he is going to drop A Block to the suburban holding area. This transaction does a variety of things. First of all, it records that all ten of those inmates are no longer on the tier. They moved from one sphere of control to another. Secondly, it indicates that an action was taken in accordance with a schedule that was made up the night before. The third parameter "S" it indicates that they are going to the suburban holding area.

turned over to transportation units that may service as many as ten suburban branch courts. Thus, by entering "/Dispatch, Court Code 1, Court Code 2, ..., Court Code 10" the holding area officer will record that the inmate is being transferred to the custody of an external authority. When he types this transaction a format comes to the screen. He will type in the badge number of the transportation officer who is also signing for the inmates that he is taking. In addition, the system will record that a dispatch was made in accordance with the schedule, and record the departure of a group of inmates from the global sphere of control of the department. One note about jails and judges, you are dealing with separate administrations. If we send those ten people to court in the morning, we don't know who will be coming back because an individual could be in jail this morning with a $1 million bond for five counts of murder, and he could go into court and have the charges dropped. Therefore, he does't get transported back in custody like somebody who is still under indictment. He goes home. You never know who is coming back when you send inmates to court.

These transactions are, of course, simplifed. They use the IMAGE data base sub-system. Multiple data base accesses are associated with both transactions and this, far more than the number of different terminals talking to the 3000, is the limiting factor in our systems' performance.

Upon completing our analysis of the system at the Cook County Depatment of Corrections, RFP's were let with the specifications shown in summary on the slide. I won't read them all, but HP could meet all but the first, "62 simultaneous terminals as a minimum". Our options included: 1) Trying fewer terminals. We went back and took a look, but fewer terminals would have diluted the spheres of control so far that they would not have been productive. In addition, there would be no growth potential to accomodate the new buildings. 2) HP could offer no solutions, save many dial up terminals within the 32 simultaneous terminal limit. Given an extremely unsophisticated users group, guards, and an environment often referred to pre-twenthieth century, this was unacceptable. We had neither the in-house expertise nor the time to develop our own front end. Therefore, we looked for vendors (option 3). One was Westinghouse Learning Corporation. They were talking about building a transaction processing front end to a 3000 but they were a couple of years away. There was a Canadian firm as well, the name I don't even remember, who wasn't even sure they would proceed with the project, and there was a company called Systems Research, Inc. (SRI). We decided, based upon what we saw being developed on a 21 MX, a front end and message control system, that we would go with the 3000 with a 90 day no-penalty cancellation clause. This was

and see what would happen. At the conclusion of the 90 day period we faced a decision. SRI was developing a 21 MX front end to a 21 MX transaction processing system for a medical application. Secondly, SRI had experience that they could demonstrate in writing other transaction processing systems, and, of critical importance, after three months of playing with the Skokie HP Sales Office's 3000, (when it was up) we realized that the number of terminals was not a terribly difficult hurdle to surmount. Rather, process control and message control within the 3000 were more critical. SRI was modifying software that they had developed for a Burroughs' system to control messages internally. Unfortunately, by the time the 90th day came along, and it went to day 90 before we said "go", SRI was only able to demonstrate a small portion of the interface between the front end and the 3000. We basically took a calculated gamble on SRI.

The results have been a structure for coding that have permitted four programmers, totally inexperienced in transaction processing, to write independent transaction modules in two to three weeks per transaction with that time frame decreasing rapidly. SRI, unfortunately, hit a dead end here and there that delayed implementation, but, this was not unexpected. HP is not quite as predictable; we found that once in a while MPE or, worse yet COBOL, sticks up its head and goes "surprise!".

We know right now, for example, that if we don't get our hands on COBOL C pretty soon, we will be in serious trouble due to symbol table overflow. However, right now we are confident that CIMIS in the Cook County Department of Corrections is going to be a reality, and that you can do transaction processing on a 3000, (even on an MX) with a lot of terminals, by using some common sense in writing the code and by-using the SRI front end-message control system. Depending upon your application, you can probably do it too, with even as many as 192 terminals.

Before I introduce the message control wizard, Alan Dale, let me say that we are so confident, that we have ordered two 3000 Series II's that will provide a much more elegant, efficient, and trouble free transaction processing environment to be used in the Illinois State Department of Corrections with a network of 21 MX front ends.

I am going to let Alan Dale tell you generally how this message control system works and how you can perhaps apply it to your application.

Lanny Norensberg

I am here to speak on SPSS. Many people have called me to say that they would be here to listen. SPSS is not up yet, however, we are looking forward to getting something up in the near future. Unfortunately, it is a slow process. As you know, SPSS is now up on various 360's, 370's, CDC, PDP 10's. and other machines. There is a "version" on the 2000 written in Basic, but it is not a version sanctioned by SPSS. As of now, there is no version on the 3000.

About two years ago, in response to an internal need that we at NERA had, I contacted SPSS to see what could be done to implement the system on the HP3000. They said the only thing they could do is send me a tape and a McGraw-Hill book and their best wishes. And, that is what they did. They sent me a tape with 93,000 source cards.

The problems with SPSS are as follows. Before FORTRAN B, much of the task was impossible. It could not be done without spending large sums of manpower, money, time, etc. The problems are as follows. Before FORTRAN, you did not have options such as entry; if that option was needed, and SPSS has used it considerably, you had to rewrite the subroutine. And, considering that there are several hundred subroutines in that package, this was a mammoth task. Also, you never know if they just stay at that entry point or whether they actually go back to the beginning of the subroutine. You have to trace everything down. We were spending a great deal of time on it. Also, it must be realized that we cannot devote six people to this effort full-time. We are a profit organization. My computer department services the economists at NERA. SPSS is just a sideline. We initially went into it to serve ourselves. As it turns out, we may end up servicing the entire Hewlett-Packard community.

Some of you may not know what SPSS is. It stand sofr "Statistical Package for the Social Sciences." SPSS was developed by the National Opinion Research Organization at the University of Chicago. They developed it for their own internal needs and found out that it was very marketable. However, as I pointed out at the beginning, it was marketed for the IBM360. Eventually, they moved it up to the IBM370 and then eventually others converted it to the other machines.

Only one of the problems with SPSS was pointed out. There are several others. For instance, many of the key routines were written in BAL, which is IBM's assembly language, which makes it impossible to run those routines on our system. We either have to convert them or forget about them. Where we can, we are choosing to forget about them, at least for the moment. Where we are able, we hope to convert them. The only problem here is to find out what those routines do. SPSS, on many long calls to them, and in some letters to them, has been unable to tell us completely what those routines do.

Now that we have FORTRAN B, some of the problems I mentioned before, such as entry, double integer and a few others have been solved. But, not all the problems have been solved. For instance, they use LOGICAL *1 quite often. The question arises as to whether logical one is used as a character or is used as a logical. I have found both instances in subroutines. The same variable has been used as a byte character and as a logical. The question is how to convert it. Do we create new variables, have two variables running down the line, etc?

There is another problem with SPSS. Apparently, at least, our compiler has been flagging it. When on subroutine calls another subroutine, it passes the variable either as real, double precision, integer, etc., in the same dummy position. They will call a routine five or six times all with different parameters. The FORTRAN B compiler flags this. FORTRAN B, as soon as it gets the first subroutine call, looks at the parameter list, keeps a record of it, and then when it gets the second subroutine call it says, "Something is wrong." This subroutine call does not match the first one, and flags it. This means I have to go back to the original routine and find out why it wants it in one of these formats and rewrite the subroutine call to make sure it gets the data that it wants at that precise moment. It is not very simple, especially when you have several hundred routines and any 80 of them at one time may be calling the same subroutine. It is a huge problem.

Over the summer, I had two and one-half people work on it full-time for two months just trying to figure out which subroutine called which subroutine. We are finally getting to the point where we may lock that problem. Now we have to resolve it.

These are only the minor problems. Then major ones start. The version we have is version six, which is 156K version that is on 360. We should have no problem fitting in the same size data base simply taking advantage of the virtual memory aspect of the 3000. They rely totally on the whole thing residing incore. We can segment the program a little better than they can. But, one of the fears in segmenting it, is that we will oversegment. If it takes one-tenth of a second to call in a segment, if you oversegment it, especially if you are calling a subroutine a thousand times, you can end up running very slowly. So, that is one of the problems we have to solve.

I have requested that Hewlett-Packard provide some aid and assistance in this, only to be turned down. They are not a very helpful company, especially since they seem to be vending the SPSS package for me. I receive calls from around the country from people saying that a Hewlett-Packard salesman has told them that I have a finished product and that I am selling it. And, the Hewlett-Packard salesman seems to be selling the 3000 on the basis of my product. This I find very distasteful, considering the fact that when I call them up and ask for help they turn me down saying it is not one of their products; it is one of NERA's products and they do not want to be bothered with it. But they seem to be doing a very good job of marketing it.

What we are hoping for eventually is to come out with a small version of this package, not a full version. From our analysis, of not only our work at NERA, but from discussions with others dealing in economic research, I have discovered that there are two main programs everybody wants. One is a regression analysis--a super sophisticated one. And the other is a cross-tab with all its accompanying routines. The other things that SPSS has are nice and people would like them, and hopefully we

will be able to get them up, but immediately we are striving to put up a stripped down version of regression and cross-tabs. If we are lucky, and everything goes our way, and the 3000 is cooperative and stays up, and the staff is still available to me, I hope to get something up sometime before April.

Admittedly, it will not be as neat to use in some cases as SPSS. We will have to take away some of those super fast read routines, at least temporarily, until we substitute them with SPL routines that can read a little faster. In the meantime, what we will do is stay with FORTRAN, get a nice, neat FORTRAN version up that will read your data, that will read your control cards, strip them down, analyze them, etc. I do not guarantee that it will run fast. For those of you that are using it in a high production shop, or intend to use it in a high production shop where you must meet instantaneous deadlines, I do not believe the first version will be for you. In fact, it may not even be for us. We have an interest to speed it up. The first version will just simply produce accurate regressions.

We are going to have to spend much time augmenting the system. Every subroutine has to be gone through with a fine tooth comb to figure out exactly what it does. Then we must take out the unnecessary things. These are the problems we are going to face with SPSS.

It is now September 30. Tomorrow is October 1. Some of you have talked to me over the past six or seven months and always hear a new deadline pushed out in the future. But, consider the complexities of it and consider the fact that we keep finding more problems. Every time we correct one bug, we recompile it only to find out that the compiler missed several other bugs along the way.

There is another problem we have. They use hexadecimal throughout. They use it in data statements. FORTRAN B flags hexadecimal in data statement. A core-to-core conversion or reading it from some dummy file must be used just to produce the same results they do. They have character strings throughout. We have to make sure we are reading the same amount of characters they are reading.

I have tried to trace down several routines myself only to get lost in what the main control cards do and where they go. The manual is simply the same manual you can buy in any bookstore. The documentation SPSS has supplied is nil. All they have are quite a few comments in each program, which are good. But, as you can see we are running slow.

In the meantime, what NERA has done (if any of you want to send us a letter, we will be glad to make some of these available to you) is to develop some of our own internal routines to compensate. We have developed our own internal regression programs that do much of what SPSS does but without some of the output, and without the same need control cards. We have also developed our own cross-tab package. We had developed these packages before we contacted SPSS. We just polished them up a little bit better since then. If anyone is interested, feel free to write to me at National Economic Research Associates, Inc., 80 Broad Street, New York, New York 10004.

SPSS is not the only package we viewed. For instance, we looked at IDA. At Rockville, Maryland, we had converted IDA from 2000 BASIC to 3000 BASIC only to find out it was a "bomb." IDA did not serve our purpose. It works on small data sets, it was difficult to handle, no one wanted it. IDA is an interactive data analysis package. It is something they developed on a 2000. I think Chicago also did that. It is available at NYU's computer center; it is available on several others. You can get it for whatever the contributive library costs you. IDA is on one of the reels and there are about 30 or so BASIC subroutines. You can compile it. There are still problems. You have to figure out what IDA does. Just like SPSS. There is very little documentation on it. It is only a user's guide. But, it is simpler to put up than SPSS. For many of you who work on small data sets, that may be an advisable solution. Simply take the programs, compile it. The only thing is, of course, if you are going to compile it you have to figure out the entry points of each subroutine. And the entry points are not known in advance. You have to trace it down as we did. We do not have it completely worked out, but I will be glad to

send you what we do have. Again, you can contact me for that and you can finish working on it yourself. If I ever get a working version, I will put it into the library.

Besides SPSS, we looked at ECON. We are converting ECON. As soon as we get it completely converted, we will put it into the library. It is a little simpler than SPSS. There are fewer routines, very little source code, and as soon as we finish it we will distribute it.

The big question is why did NERA get involved in all of this? We are a profit organization. We are not a university. We are not a public research firm. We are a private research firm. We work for utilities; we work for major corporations in antitrust. We are a firm of several dozen economic analysts who analyze a company's situation, not as a market study, but as a litigation situation, such as an antitrust or rate setting case.

We were finding out that many of the antitrust cases require several years of data over several companies. We were analyzing 10,000 observations, 20,000 observations--which may not seem like much to many other companies, but to companies like NERA, which was used to something like 60 or 70 observations at a time, it was an immense task.

Our economists found themselves using these large packages. Now we were obtaining the Hewlett-Packard 3000, a considerable investment. We had a decision to make. Do we continue going outside while we had this computer sitting inside or look to use it. Many people were using SPSS. They suggested we get it. We got it. Unfortunately, it is not in a state to help anybody.

I have discovered one thing in my life; there is no such thing as portability. IBM will prevent that. They go out of their way to prevent that. They do not care what the standards are; they come with IBM standards. Meanwhile, everyone else is developing along the other standard. Hewlett-Packard, however, should have come out with some of these standard items a long time ago. Entry, for instance, has been

standard for a long time. Why it had to wait for FORTRAN B, I do not know. We converted a spline program which we received from some university; and fortunately it was a small one. It had entry all over the place. I had to resubroutine the thing.

You must realize that SPSS was specifically written for a 360 and it took advantage of whatever features IBM had. For instance, routines were written in BAL. You cannot expect Hewlett-Packard to convert to BAL. However, you can expect Hewlett-Packard to implement LOGICAL *1. I can expect them and do expect them to implement the hexadecimal data statement. That is a problem. We get many packages like that. RAPE has that, TSP has that. I could go through a whole list where they wrote BAL routines. What are we going to do? Perhaps, if Hewlett-Packard were a nice company, which lately they have shown otherwise, they would sit down and write a little assembler or something that would transfer BAL to SPL. Easy enough. They have much manpower out there. Let them do it. They want to sell these packages. They want to sell their machines. Other conversion programs have been written. I, myself, would undertake it if I knew BAL, but I do not know BAL well enough.

IMSL provides a package of subroutines. When I was at the Federal Reserve, we had purchased that package of subroutines. Our business is not to market SPSS. We are putting it up for ourselves. If it happens that it works, we will market it. We are not out to market it, per se. We are not going to make a profit on this thing. Most of the people who want it are universities and their rates are low. We are doing it for ourselves. IMSL is a good package. Remember, they have also written them for IBM systems. You will have to convert them. We had to convert them for a Burroughs 6700. And, a Burroughs 6700 is a much more powerful machine than any of the IBM machines that are used for the ISML package. There are many other packages out there. For those of you that have needs that are immediate, there is nothing wrong in going into the scientific subroutine package now that it is totally converted. You have the 1130 subroutine package which is an SL on the system. You have all these things. There is nothing wrong in writing your own front ends.

People are reluctant in writing their own programs these days. It is the calculator generation. No one knows how to add one plus one. They know how to press one, enter, plus one. That is all they know how to do, but they cannot add one and one on a piece of paper. One of the things that you have to learn is that if you do not have a package, do it yourself. That has been my philosophy. Do it yourself. You lift up a pencil and paper, you write out the code and you have a front end. The subroutines are available all over the place. They are public domain. They are free. You do not even have to pay exorbitant fees for them. Some are good; some are bad. You throw away the bad ones. If I came out and said SPSS was up, alive and working, and I guarantee it, you would accept it as is? When we received our 3000, I had a staff sitting down for days testing out every bit of a FORTRAN compiler to see if every single part of it worked according to specifications. And, it did not. I learned a long time ago not to believe anybody. You mean you are going to believe manufacturers? If you are interested in a regression package, you can get the Wampler and Longley test data. It is very good test data. Immediately, you can tell if the programs are bad. At the Federal Reserve, I had someone sit down and make up phony data and sit there and calculate the things by hand. The whole staff calculated it out to twenty-three decimal placed by hand. And then we ran the data through our regression programs to get the exact same results.

As I was saying before, just as a quick summary, SPSS is a very difficult package to convert. I may be somewhat masochistic in doing it. I may also be more than that, but, I am not making any total definite promises. May I suggest that nobody buy a 3000 on the basis that I will have SPSS up. I can think of many good reasons to buy a 3000 for those of you who are potential customers. It happens to be what with all my kidding (there are many faults in the system) better than most choices you can make. But, if your sole purpose for buying the 3000 is because I am going to have SPSS up, may I suggest that you forget it.

I hope to have SPSS up. I am going to try. We are working on it. If we can get something up in the near future, you will have it. But, as you can see with all the problems I am having (I probably have not even encountered half the problems yet) because as soon as I get a clean compile, I find out there are a thousand other problems in the system. So who knows when we will solve that. If you do want some of the packages we have developed, the regression package, the cross-tab package, contact me.

Stan Shell

My name is Stan Shell and I work in the Marketing organization of the General Systems Division. I would like to briefly go through an announcement of a new subsystem on the 3000 Series II: an APL compiler. Announced last week at an APL users convention in Canada, it took approximately 15 man years to develop this subsystem. APL was developed at our HP Advanced R & D facilities in Palo Alto, and it will be marketed by General Systems Division. It runs under MPE, and is currently being tested at Yale University. By the way, the reason why it runs only on the Series II, and not Series I, is APL involves such features as four word floating point, and special instructions that are only available on the Series II.

Now, what is the significance of the APL Compiler? We feel that it is quite a significant advancement in that it is the first example of a full scale, full capability, APL on a relatively "small" computer.

The next slide mentions a couple of the features that we are quite proud of. With APL there is a feature called work space, in which all programs, data, variables, etc., fit into an APL "environment". Typically on other APL offerings this is limited to 80Kbytes and has been significant resource restriction. With APL/3000 we have implemented a relatively unlimited work space. (It is limited only by the size of disc storage on the system.) As a comparison between the typical 80Kbytes working space, APL/3000 starts each user with a work space of 800Kbytes. There is special hardware that assists the implementation of this dramatically larger work space. It consists of several chips which fit onto the extended instruction set board. These chips implement 11 new instructions which implement the virtual work space.

Another unique feature of HP's APL is a coexistent editor. Typically, APL Editors are primitive, however, HP's is a sophistcated one similar in many respects to the Text Editor. While it doesn't have all the capabilities that you can find in the Editor/3000 advanced features, it is a little cleaner for some of the more simpler commands. For instance, it has an undo command; if you don't like what you have

modified with the APL Editor, you can say "Undo it", and the text will return to its original state.

APL/3000 coexists with our MPE file system. It also has a feature that makes it the very significant APL announcement. That feature is called a Dynamic Incremental Compiler facility. APL is typically an interpretive operation, whereby each line of code is interpreted a line at a time. Machine instructions are generated, then executed and then the next line of source is interpreted by the interpreter, machine instructions generated, executed and so it goes a line at a time.

With APL/3000, as a line is interpreted, the corresponding instructions generated are saved. During the next pass through that part of the code, an evaluation is made as to whether the object code has changed such that a reinterpretation of the source code would be required. Of course, if need be, the source code is reinterpreted and new instructions are generated and executed. However, if it so happens that the line need not be reinterpreted, then reinterpretation does not occur. It simply executes the previously interpreted instructions. So with APL/3000, the first pass through the program might be relatively slow as is typical with all interpreters. But, the second pass through the program and the third and fourth, etc., will probably be much more rapid since APL didn't have to interpret all the instructions. This feature is a Dynamic Incremental Compiler.

In short, we see this new subsystem of the Series II as being a significant advancement for users of APL. The availability of an APL this powerful on a machine the size of the HP 3000 is really a significant achievement, and makes APL even more effective in solving the problems that APL users have today. And, we also believe that if you take a look at APL, it might make your life a little bit easier, too.

Stan Shell

My name is Stan Shell, SE and Training Support Manager at GSD. Today I would like to talk to you about our role in training and software support, and some of the other areas that touch on User Services.

I would like to keep this as interactive as I can, as it is important that we get a dialogue going. So what I will do is very quickly skim through some of the topics I am going to be discussing, allow you to organize your questions, and then ask them at the appropriate time.

First, I am going to talk about the philosophy of support, repeating a little bit of what Ed (McCracken) said in the last meeting. Next, I'll be talking about the role of the Customer Engineer, System Engineer, and the Product Specialist, in both the field and the factory. Then, I'll touch briefly on the service division, CX and pre-CX systems and customer training and the Eastern and Western technical centers. I'll also cover phone-in consulting services.

In the last meeting, Ed briefly mentioned that what we want to do in providing user services is to be competent partners in your success. If you are successful, then so are we. Conversely, if you are not successful with your installation, then we have problems.

In addition to providing you with services that you need, we would like to maximize the impact of the services that we provide. We would like to make the services that we provide satisfy a majority of the user community. For instance, if someone would want assistance in interfacing a solar energy cell to a 3000, that might be one that we would look at with caution. However, if there is something that we are not doing, that we should be doing, and it would benefit the majority of users, that is what we want to hear about. We also want to know where we are not doing a good job.

There is a little bit of confusion that exists in the user community about the HP representatives that the user may interface with; specifically, the Customer Engineers, the Systems Engineers, and the Product Specialists. It is pretty easy

to identify a salesman. It is kind of his job to identify himself. Where things may get a little blurry is in the area of "What does the SE do?", and "What does the CE do?", and "Who is this person called a Product Specialist?"

The CE's, of course, are the individuals that are primarily hardware oriented and provide you with support, primarily of a hardware nature. In the factory at GSD, we have a group that parallels the CE group in the field. When there are problems that cannot be answered in the field, the factory is where the field should turn for assistance.

Product Specialists are the superstars of software knowledge . . . in MPE. These are the individuals who will be called in by the CE when you have a problem in your operating system. Once again, we have a Product Specialist group in the factory that is responsible for training these product specialists, and for providing backup when they, in turn, need help.

The SE's are yet another group. They have not spent years with the operating systems like the product specialists, but they are concerned with helping you with subsystem software problems. Also, they work in a sales environment. SE's can be found in the presales environment during technical discussions. In a postsale environment, if you have trouble with COBOL or IMAGE, it is your SE that you turn to for assistance.

Let's talk about CX (and pre-CX systems). I have about 30 seconds to talk about this support, and I am sure you have about two hours worth of questions out there. So, let me start by saying that we will support CX and pre-CX systems. I think Ed pointed that out in the last meeting, and I would just like to reemphasize that. There is a group in the factory that will specialize in assisting the field in satisfying problems relating to CX and pre-CX systems, MPE and hardware areas. Now, for subsystems. It would probably make sense to look at the fact that subsystems on both Series I and Series II are essentially the same. There are some minor differences related to four word floating point and double integer, but FORTRAN is

FORTRAN . . . there is one FORTRAN compiler. There are two compiler libraries that will cause a program compiled in FORTRAN on a Series I to have a different object code than a FORTRAN program compiled on a Series II. But, it is still one subsystem and it will still be supported out of my group in the factory, and, of course, your local SE.

I would like to briefly talk about some of the ways that I think you will be getting better quality support from your SE's. My group is charged with responsibility of training new SE's and, also, with advanced training of experienced SE's. There are a couple of things that are coming out that will provide the SE's with more tools to solve your problem.

The training that we have for the HP SE's right now is a little thin. We haven't in the past done enough to provide them with the proper tools. We will be offering to our SE's, in the very near future, three to four new training packages. A couple of the courses will be related to programming techniques, and one course will be devoted to system performance, system evaluation, and system optimization. What we are attempting to do is put together a cohesive package so that the SE can converse more knowledgeably on what to expect from a system in a particular situation. I think this will make the SE more valuable to you on a consulting status. I would anticipate in two to three months, we will be starting to train our SE's in the performance area.

We will be looking very strongly at providing specilization courses for our SE's. In my mind, at least, it makes sense for our SE's to begin to specialize more in the commercial environment or the scientific oriented products. What we will be doing is putting together specialization packages to train our SE's so that they can more adequately support you.

On customer training, the eastern and western technical centers will be the area training sites. The courses we currently offer are held at these centers. Training on the 3000 follows an interesting course. When I started working at HP

(4 years ago), the 3000 was visualized as being a super mini type machine with a problem solving, scientific orientation. Therefore, I can recall back three or four years ago, we were gearing our customer training in that direction. We had, in the preparation stages, classes that were going to address such things as I/O drivers, MPE internals, etc., for the knowledgeable user. Well, we found out the 3000 was also a very good general purpose commercial system. Having a relatively high technical content, courses for a customer from an RPG installation might confuse them. Also, not only was it difficult for some of the commercial customers to comprehend it, but it wasn't necessary to solve their application. So, we took a look at customer training and fashioned courses a little more for general purpose commercial community.

Now it is becoming evident, as our customer base is growing, that we are in the situation where people need more information than we are communicating in our customer courses. There are some of you out there who have some very sophisticated tasks that you would like to accomplish. Our training has not effectively addressed that. But, we will be offering such courses in the near future. One month from now will be the debut of a class tentatively called Series II Special Capabilities. It will be programming oriented, as opposed to the syntax oriented ones we have had in the past. For example, we will cover how to utilize some of the intrinsics that are available to users, how to use the debug package, etc. It is probably going to be difficult for individuals that purchase a system now to take our introductory classes and then run right into our advanced user class. I think eventually we will have an intermediate course for users. So, essentially, for those users who are not interested in complex utilizations on the 3000, we still have our basic courses that we teach on an introductory basis. For those of you that are interested in more advanced techniques on the 3000, we will have offerings in that area also.

We will also be offering, at selected installations, on-site training. The majority of on-site training will be done by professional instructors out of the Western Technical Center.

This slide mentions PICS. What is phone-in consulting? The service that it will offer is to answer specific problems that you might have, where the answer to your problem, hopefully, is no longer than a telephone call away. The individuals on the phone will have access to our factory SE group. If they get into trouble with a question you have, they have the factory SE group to fall back on. If the factory SE group cannot answer the question, assuming it is appropriate for PICS, then the Lab, of course, is available. That is what the individual manning the PICS phone will have as his method of information of retrieval. He has full access to our SE group and in turn, the Lab to answer customer problems.

Now, let's talk about software problems and PICS. I think one of the significant offerings of PICS is to allow you to have immediate response to problems you are encountering with your software. The phone-in consulting service will be the focal point for the software problem reporting system at HP in relation to the 3000. It will be a funnel, a filter, and will allow the customer when he calls PICS to immediately describe his problem and get an immediate response, "Yes, that is a known problem," or "No, it is not."

Here is how I envision the system working. You are out there and you have a problem. Well, what you can do with PICS is to call the Tech Center and describe your situation. Now, if it is a simple problem, the PICS SE will try to give you a work around and will submit a software problem report if necessary.

Now, let's take something a little more complex. You have multiple discs with a complex data base. You run this program for half an hour and you get the wrong answer. What is wrong there? Unfortunately, this type of problem you are not going to be able to describe over the phone and have the PICS person verify it. It is obviously of a much more complex nature. What we will have to do at that point is have the PICS individual contact the field SE. The appropriate field individual will travel to your installation, and with you the customer, reduce that problem to a manageable entity. And, at that point, the customer with the SE's assistance, will fill out a problem report and send it in, if appropriate.

Before I start, I would just like to ask a question of you out there. How many of you have used APL? How many have used BASIC? Now, the toughy. How many have used APL on the 3000? No one? That is because up until last week it was not available on the 3000. We announced APL on the 3000 last week and it will be available in November.

Since I assumed that most of you were not users of APL, I thought I would start out with some very basic questions. The first question that popped to my mind as a question that someone might ask is What is APL? APL is an acronym and it stands for A Programming Language. They really went to a lot of work to figure that one out. This language was developed at IBM and has been available from IBM for the last few years. It was developed around 1967, '68, and it was not actually supported by IBM until a few years ago. APL is an interactive language. It differs from languages such as FORTRAN and COBOL in that you don't have to specifically request that the system compile your programs. Most versions of APL are interpreters. But the new version of APL available as APL\3000 is actually an interpretive compiler. It compiles code as it needs to and keeps the compiled code available in the work-space. I will get into that later.

Another question that popped to my mind is Who can use APL? Most people can use APL. It is a very easy language to learn and to start programming in APL is quite simple. We have some people working..... I should have prefaced my remarks by saying I'm from Hewlett-Packard Research Laboratories in Palo Alto where the APL program was generated. I do not work directly for the General Systems Division, but the product we developed in the labs will be available on the HP-3000 system

and will be supported by General Systems. We have one fellow who is about 12 or 13 working with us at HP. He comes in every day and he sits down at the terminal and writes APL programs. He picked up APL by sitting at a terminal and playing with it. You can start at a very elementary level because of the interactive nature of APL. You can simply type 2 + 2 and you get back an answer. You don't have to first define the fact that what you are dealing with is integers and that you want an array that contains two integers and all sorts of things like this. You don't have to get into that level of detail in order to do useful programming in APL.

The third question I have here is What is APL good for? APL is a language which because of the power built into the language, can be used in a wide variety of applications, both in the scientific world and in the business world. I will have some examples of that further on. I just wanted to go into a list of some of the features of APL that make it a powerful language. (See slide 1).

One of the first features that I have already mentioned is that APL has a calculator mode of operation in that you can sit down at a terminal and type statements in an algebraic language which are executed immediately on the machine. There are powerful primitive operators in the APL system which allow you to deal not only with scalars, which is the normal data type that most other programming languages deal with, but you can deal with arrays as an entire entity. You can define an array and then you can define operations on an array which allow you to do some very useful things.

There is another feature of APL\3000 which is only available on a few other APL's in the world. We have a virtual workspace. The concept of a workspace is one that sometimes is confusing to people. So I have a little slide here that illustrates what a workspace looks like. (See slide 2). In most languages you deal with a

series of programs that interact with data that you have stored in some other part of the system. For example, you may write a COBOL program which reads and writes into files which are available on the system. You tend to use files to hold the data and programs have to be grouped together in order to provide a useful system. But, in APL we have what is known as a workspace. An APL workspace contains not only your data but the programs which you are working with. All of these are named-objects in your workspace, and the workspace itself has a name. So what the workspace is is a collection of both program and data objects, that you need to solve a particular problem. The idea of having a workspace and collecting together everything you need to solve a particular problem makes programming much easier. You don't have to rely on the file system or link editors and segmenters and a lot of other things in order to get a programming job done in APL. So this relates back to the earlier question which I posed. What is APL good for?

The concept of a workspace and the ability to group things together like this and the interactive mode in which APL runs, allows you to write and debug programs much more rapidly than you can in any other language.

I just returned from an APL conference in Ottawa, Canada, where HP announced its APL. At this conference there was a fellow from Europe who was discussing the usage of APL in Europe. What he did was to go into several small companies that were using APL in Europe and find out the kinds of applications they were using it for. He looked at the number of lines of code generated by the various companies. He came to the conclusion that the total amount of program development time spent by these companies was three times less than it would be in any other programming language. So, what is APL good for? It is especially good for cutting down the amount of time the programmer has to be involved in actually writing an application or solving a problem in APL.

Let me go on and talk about some more of the other features of the APL\3000. I have talked about the primitive operators, the virtual workspaces. We have interactive debugging in APL. What does this mean? Let's suppose you write a program and it has a variable V. And this particular variable you have forgotten to define. You know what happens in a FORTRAN, or ALGOL, or SPL program? If you reference variable V that you haven't declared the compiler gives you an error and stops and won't let you go any further. Well, in APL when you get to the point where you are trying to use the value of V, the APL interpreter realizes there is no value for V. It prints out the line in which the particular instance of V occurred, puts a pointer under the variable V, and tells you that it has a value error. It doesn't know what the value of V is. At that point, you can enter the value of V and continue the program from where you left off. If you have a syntax error, mismatched parenthesis or something like that, at the point the APL system finds this error, it stops and points to the place at which it found the error. It tells you it has found a syntax error. You can go in and change the program where the error occurred and continue from that point or some other point in that program. As you sit and debug your programs in APL, the entire power of APL is available to you in the debugging process.

We have a built-in editor in APL. APL functions and APL character arrays can be called into the editor. You can make changes and delete lines. The flavor of the built-in editor in APL\3000 is very similar to the editor that already exists on the 3000. The editor in APL knows about APL data types and it knows about APL programs, so you never have to exit the APL system in order to edit programs or data.

Another feature that we have in APL 3000 which doesn't exist in any other APL that I know of, is that we have two languages which coexist, and operate on the same data. You can write programs in either of these two languages. The two languages are APL and APLGOL. APLGOL is an ALGOL-like control structure that is placed around APL statements and I have an example of that later on. What APLGOL tends to do is make the flow of control in your program more readily visible to someone who is reading these programs later on. You can also put comments in your APL program and that doesn't hurt you, because when the APL system recognizes a comment it simply ignores the comment from then on. The comment only takes up space off line in the disk and is never actually swapped into the main core where it would take up room. The only time it is swapped in is when you are back translating an APL program in order to edit it or change it and then you would want to look at the comments and see what they were.

The last feature of the APL\3000 that differs from other APL's is that it is a dynamic compiler. By that we mean that it compiles instead of actually interpreting the code. The first time it sees a line it has never run before it compiles some code for that line and that code is kept in the workspace. When you come back to run the function again it just reexecutes the compiled code. We have seen speed-ups on the order of two or five to one, depending on the exact problem, between a compiled and a noncompiled version. The first time you run a function it is noncompiled and it will get compiled as you run it. From then on it will stay compiled providing the types and shapes of the variables involved do not change. If they do change, the system automatically knows it has to recompile code because the code generated before is no longer valid.

I would like to show you a couple of examples of the kinds of things you can do in APL. I have one slide here that illustrates the calculator mode of operation. (See slide 3). At the bottom of the slide it shows how to calculate an average in APL. This illustrates both the calculator mode of operation and the powerful functions available because we have here a variable X which has been assigned the vector of values 100, 86, 75, 95, etc. This statement sets up the variable X and allocates storage for it to be placed in without the user having to get involved. You can then calculate the average of those numbers by summing all the elements of X and dividing by the number of elements in X. Since X is stored as an APL data type which is a vector, the APL system knows about its length and knows how to add all the elenents of that vector. These operations are available within APL and you don't have to write a separate program to do them.

To give you a little more of the flavor of APL, I have an example here of a problem that actually came up in the research lab where I work. One of the fellows wanted to calculate the following integral. It is a fairly standard integral. It is the SIN(T) over T. He wanted to find the value of that integral from zero to X where X was 3.2. Now this integral does not have a closed form solution. So one way of approximating the integral is by taking a summation. You simply divide the interval up into a number of little spaces, each one of width dt, and then you evaluate the sin at each of those points and add the whole thing together. So a series approximation for the integral, taking advantage of the fact that some common terms can be cancelled, looks something like this. (See top of slide 4). As an exercise I programmed this example in APL and BASIC.

The BASIC solution for this problem is very similar to the FORTRAN solution or the ALGOL solution, for that matter. (See middle of slide 4). First you have to declare the variables and tell the system what they are going to be. (Line 10). In this case we are going to use long floating point. Otherwise, the answer would not have been accurate enough. Then we had to initialize some variables and set up a little FOR loop which says FOR I = 1 to 4. (Lines 20 through 30). Essentially, what we did was to evaluate the integral for 4 different values of the increment of summation, DX. We wanted to compare the value of the integral we got to see how small a value of DX we needed to give us the appropriate accuracy we were looking for. So you have an outer FOR loop that does the four different values of DX and then there is another FOR loop, FOR I = 1 to L, where L is the number of things you are going to sum. (Line 70). You then do a sum which is S plus the SIN term (Line 80), and store the answer when you are done with this innermost FOR loop. (Line 100). When you are all done you can do a matrix print Y (Line 120) which in BASIC prints out the entire array of the four solutions that are shown here. So this problem is about a ten or twelve line program in BASIC.

That same solution in APL is given in one line. (See bottom of slide 4). What I did was write a function which could then be called with the different values of DX. The first three lines are a request for a display of the function FUN, and then the fourth line calls the function with the various values of DX. The answers are printed out automatically.

What I am trying to show here is that the amount of programming involved to do the same thing in APL as opposed to BASIC is significantly different. I will read the APL solution from right to left because that is the way APL operates. I generated 3.2 which was the range over which we wanted to do the summation. I took DX and divided it into 3.2 and then I took the index generator in APL; (⍳) it generates all the numbers from one to the maximum number that we want. Essentially I generated the integer indices that were needed and assigned them to a variable I called XSI. Then I added a -.5 so that the series of integers becomes the series of half integers .5, 1.5, 2.5, etc., that is the denominator of the expression. The numerator of this fraction that occurs as one term in the series is the same series of half integers multiplied by DX, which is the increment size. So we take the SIN of DX times this list of half integers and divide that by the list of half integers and then the whole thing is summed using the reduction operator in APL. The result is put into R. R is the result of the function. When you run a function the result is then available to be assigned into something else. Since I didn't assign it in the line where I called the function, it simply gets printed out on the console. In APL, if you don't specifically assign something to a variable, it will get printed on your console. You don't have to write a special output statement to get things to print out for you. That is the flavor of the APL solution of this problem. I might point out some other differences.

In BASIC, the greater than sign (>) is the prompt that tells you that you are ready to input another BASIC command. However, in order to get a program which you have written to run you have to enter the command RUN. (See line following line 120). In BASIC you are normally in a function definition mode. In order to actually run a

program you have to get out of that mode and tell it to RUN. In APL you are normally in an execution mode. In order to define a function you have to flip into the function editor so that you can actually define a function.

There are matrix operations available in BASIC but you have to preface the statement by MAT to signal the BASIC interpreter that a line contains a matrix operation (e.g. line 120). In APL all of the variables here can be matrices and in most cases they are. APL treats matrices in the same way it treats scalars, etc. You don't have to do anything special to get it to handle matrices.

I have another example which is a little more lengthy, but it is fairly straightforward. It shows how APL can be used in the business community. Here is an example of the kind of things that are fairly easy to do in APL. We have here some data which is revenue for several different salesmen for several different products (See slide 5). Each number in this matrix represents the revenue generated by a particular salesman for a particular product. For example, Smith sold $140 worth of shoes, $19 worth of hats, Hall sold $659 worth of pants, etc. We have another matrix which represents the expenses each salesman incurred when selling each of those products. So these two matrices can be primitive data objects in APL. You can talk about relations between the matrices.

These matrices were entered into the APL workspace as shown in Slide 6. You simply state that the revenues are going to be a 4 x 5 matrix and here are the numbers. You input the numbers and can ask for the matrix of numbers to be printed out and it is printed out in the format that you have specified. It is stored internally as a matrix with four rows and five columns. To calculate the commissions (See slide 7) you subtract the expenses from the revenues and that is the actual

revenue each salesman has brought in. Then you multiply that revenue by the appropriate constant you are using for commissions. In a single line of APL you can generate an entire matrix of commissions for each of the salesman for each of the products.

Let's calculate the profits (See slide 8). In order to get the profits you take the revenue and subtract from that the sum of expenses and the commissions. A single APL statement calculates the profit. The result is a matrix of profits by salesman and by product line. If you want the profits by product line, what you really want is the sum over the rows, and that gives you for every row in the matrix the total amount of profit for that product. If you also want the profits by salesperson, then you do the sum over the other coordinate of the matrix which is to say you sum down the columns. And, in that case you get the profits by salesperson. If you want the total profits, that is the sum over all profits. In APL this is written +/+/PROFITS which gives you the total profit for your company. By taking advantage of the array capabilities that are available in APL, you can structure your problem in such a way that it is much easier to think about and much easier to extract information out of that data base by the operators that are available in APL.

I said earlier that we have two languages which coexist within our APL system, both APL and APLGOL. I have a brief example that shows the difference between the two languages. (See slide 9). Here are two identical programs called Toss and Flip. When you call TOSS, it prints out either "HEADS" or "TAILS" based on a random number that it gets from the APL system. In APL the function would look like this. (Top of slide 9).

In line 0 we have Toss, which is a function name, and COIN is a local variable, local to that function. COIN is assigned a random number between one and two in line 1. Then line 2 determines whether COIN is equal to 1, if so you print out the string 'HEADS'. If COIN is 2 you go down to line 5 and print out the string 'TAILS'. That is the way it would look in APL and that is probably not the clearest way to structure your program.

In APLGOL, this function looks very much like the ALGOL procedure you might write to do the same thing. If coin equals one, then print 'HEADS', otherwise print 'TAILS'. Essentially, what APLGOL does is generate the appropriate branch statements that are needed to get this function to work within the APL system. As I said, the APL and the APLGOL functions are compatible. You can call one from the other and you can write your programs in whatever language you feel comfortable with. Another fact that came out of the talk on APL usage in Europe which I referred to earlier was that people tend to write large systems of APL programs as collections of small procedures of ten and forty lines each. These procedures do specific things and they all have names, etc. This feature is very helpful in debugging your programs because you don't have to bring in long, long, long source programs of many thousand of lines and try to correct an error in line 1500 of some incredibly long program. If you keep your programs short, it is much easier to debug and to find out the flow of control within your system.

Let me just recap by going back to one of the earlier slides (See slide 1 again). After recapping, I can answer any questions you might have. The features of the APL 3000 are: 1) This is the only true calculator mode language that is available on the HP-3000. 2) It has very powerful operators built into the language. For

example, there is an operator available in APL which will sort a string of numbers in ascending order. You don't have to write your own sort procedure to do that. 3) You have virtual workspaces which allow you to collect together all the programs and data which you want to use to solve a particular application. 4) Interactive debugging which helps you write programs faster and debug them more quickly. 5) The built-in editor which allows you to edit both programs and data within your APL workspace and keep it in a fashion that is compatible with APL. 6) The coexistence of APL and APLGOL in the same workspace environment, 7) The dynamic compiler which speeds the execution of the APL programs so that you don't continually pay for the interpretive overhead.

At this point I will open it up to any questions anyone has about the system.

Q: Can you call FORTRAN subroutines from APL?

A. You cannot at this time write a FORTRAN subroutine and call it from APL. However, you can access files that may have been generated by a FORTRAN program.

Q. Do you have an interface to IMAGE?

A. We don't have a connection to image. We do have a connection to the file system, only, at this point. However, we are looking toward connections to image and other languages as a possible future extension.

Q. It sounds like APL has reduced the distinction between code and data which is built into the 3000. Is this true?

A. We found that a lot of the newer programming languages that are becoming available and a lot of the features that are required for people in the university environment, etc., tend to go away from the concept of distinct code and data. One thing, for example, you can do in APL very easily, is write an APL program which can, from the input it receives, write another APL program and go off and execute that program.

This is one of the reasons why we went to APL. I might just add that one of the reasons why I got interested in APL originally is because I had an application where, based on information I got from the user, I needed to write a program which would then do some special things for the user. I wrote an application in about three months in APL which then took about eighteen man months to translate into ALGOL. What this program does is take input from the user and generates an ALGOL program which runs on our system. That was a much more difficult task in ALGOL than in APL. However, there is still a distinction between code and data in your workspace. It is a fairly rigid distinction. You can turn a piece of code into a piece of data and then work on it as a piece data and then turn it back into a piece of code. There are safeguards in the APL system that do not let you read the code directly as data; you have to actually go through a system operator that will turn it from a piece of code into a piece of data. It will still give you protection.

Q: How do you interact with the file system in APL?

A: The way we interact with the file system is through a concept known as shared variables. You can take a variable in your workspace and specify that this variable is shared with a particular file on the 3000. Then every time you assign to this variable you put something out to the file and everytime you read from the variable you read something in from the file. Essentially, we give you access to all of the file controls which are available on the 3000.

The reason we have not put a lot of effort into designing a file system, as many of the other APL systems have, is because we felt that since we give you a virtual workspace, you should be able to store all the data and programs you need in

the workspace without having to go to a file to do it. We also wanted to give you the full capability of the MPE file system. We don't have any special operators which store an object as an APL data object which then cannot be read by anyone else unless they know what APL data objects look like.

Q: Is APL good for writing simulators?

A: APL is very well suited for writing simulators. As a matter of fact, one of the things that one of our fellows in the lab did for fun was to write a simulator for the INTEL 8080 microprocessor. The simulator has all the commands of the microprocessor as an APL function. He can run that simulator in APL and he can also produce 8080 machine instruction. The HP-2641 terminal has an 8080 in it and you can actually download instructions into the 8080 and run them right there. This was kind of an interesting APL program which wrote programs for the terminal and could be used to change the operations of the terminal. We are not necessarily recommending that you do that.

Q: What kind of protection is available in APL?

A: When you load your workspace, you load in all your functions. If you want to copy a particular function, you can say )COPY followed by a list of functions from the particular workspace. APL will tell you which ones it couldn't copy. There is a protected copy which will not wipe out a function by the same name in your workspace and if you use that feature APL will tell you which objects it was unable to copy.

If you lock a function, then no one can get in and alter that function.

We have some interesting additions to our APL system in the area of local variables. If you have a variable declared as local in a function, it will shadow whatever value the global variable had when you are running inside the function.

However, it is possible during debugging or under program control to go back and find out the value of the global variable. In most APL systems that is not possible, but in ours it is.

Q: How much do you pay for all the power of APL?

A: You do pay some costs for the interpretive overhead, i.e. the ability APL gives you to fix up your programs when you have an error. How much you pay depends on your particular application. Again, let me refer to the paper on the European APL study. They found in some systems the costs was essentially the same, because the amount they spent on additional CPU time, they saved on being able to program their solutions more quickly. A typical program only took about three or four minutes of CPU time to run, but it had to be run once a month and updated continuously. They found that APL was cost effective in that case. In general, you do pay something in overhead for all the power available to you, but it is not that bad.

Q: How big is APL?

A: APL is a subsystem on the 3000 similar to BASIC, etc. It takes a fair amount of code, but code is shared between all APL users.

Let me just say I will be giving a demonstration of the APL 3000 on our 2641 APL terminal at one of the birds of the feather sessions this evening. If you have further questions please come to that session and we will give you some online demonstrations of the system over the phone lines to Palo Alto. I will also be glad to answer your questions after this session. Right now we are about out of time.

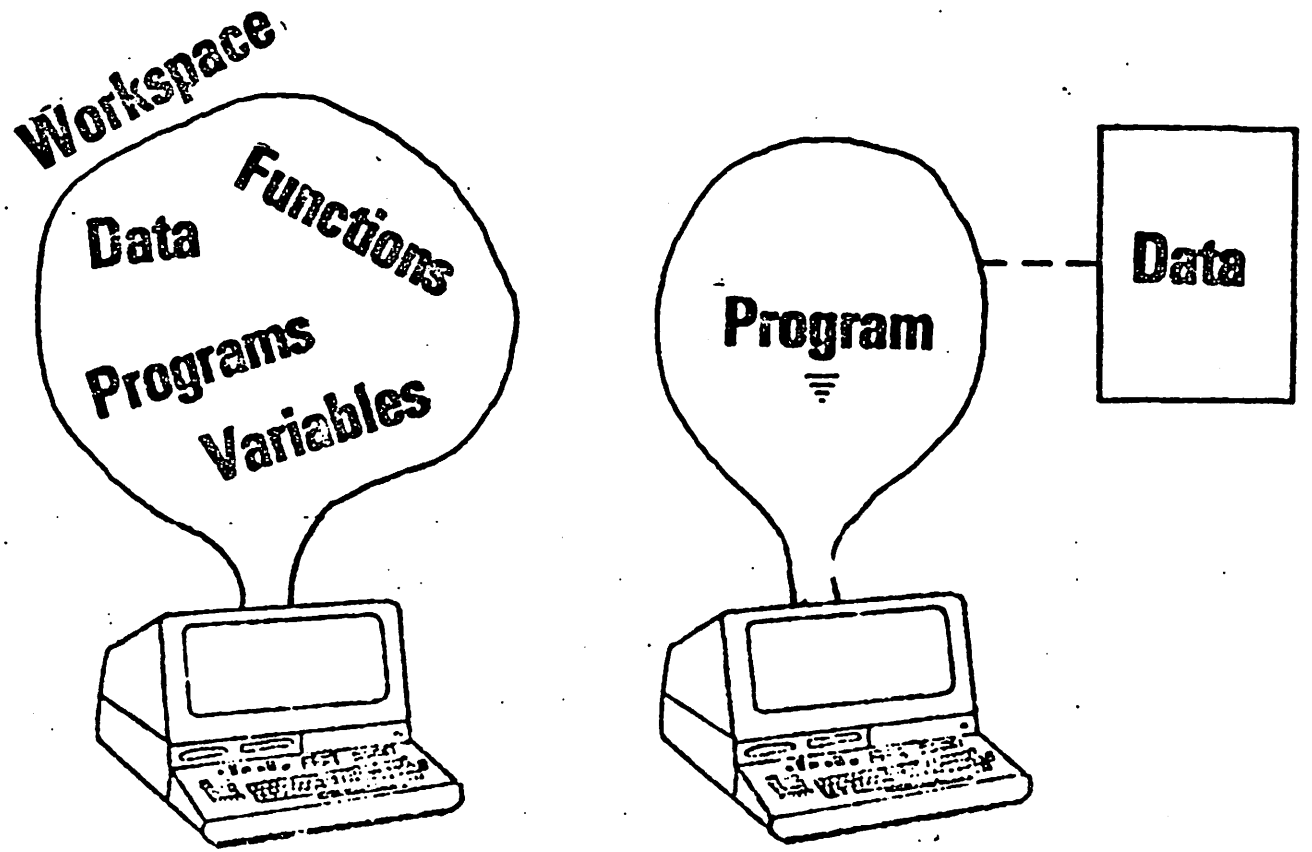
# Features of APL 3000

1. Calculator Mode Operation
2. Powerful Primitive Operators
3. Virtual Workspaces
4. Interactive Debugging
5. Built-in Editor
6. APL and APLGOL
7. Dynamic Compiler

SLIDE 1

HEWLETT hp PACKARD

# APL



HEWLETT PACKARD

# APL

* MANY OPERATORS AND FUNCTIONS
* NO HIERARCHY OF OPERATION
* HANDLES ARRAYS AS EASILY AS SINGLE NUMBERS
* CALCULATOR MODE OPERATION

CALCULATING AN
AVERAGE IN APL

```
      X←100 86 75 95 73 97 85 90
      AVERAGE←(+/X)÷⍴X
      AVERAGE
87.625
```

SLIDE 3

$$Si(3.2) \simeq \sum_{N=1}^{(N-\frac{1}{2})dx \leq 3.2} \frac{SIN[(N-\frac{1}{2})\,dx]}{(N-\frac{1}{2})}$$

# Basic Solution

```
>LIST
ARPADB
  10 LONG D,S,Y[4]
  20 D[1]=.1
  21 D[2]=.01
  22 D[3]=.001
  23 D[4]=.0001
  30 FOR I=1 TO 4
  50   L=CEI(3.2/D[I])
  60   S=0
  70   FOR N=1 TO L
  80     S=S+SIN((N-.5)*D[I])/(N-.5)
  90   NEXT N
 100   Y[I]=S
 110 NEXT I
 120 MAT PRINT Y
>RUN
ARPADB
 1.851523548519127L+00          1.851402182774109L+00
 1.851400917321872L+00          1.851400899101099L+00
```

# APL Solution

```
      ⎕CR'FUN'
 R←FUN DX
 R←+/(1○DX×XSI-.5)÷⍨.5+XSI←⍳⌈3.2÷DX

      FUN .1 ◇ FUN .01 ◇ FUN .001 ◇ FUN .0001
1.851528522
1.851402173
1.85140091
1.851400897
```

SLIDE 4

HEWLETT PACKARD

# APL

## EXAMPLE

AS A SALES MANAGER OF A SMALL COMPANY
YOUR REVENUES AND EXPENSES ARE:

*REVENUES*

| | *JONES* | *SMITH* | *WALL* | *HARRI* | *HALL* |
|---|---|---|---|---|---|
| *SHOES* | 190.00 | 140.00 | 1926.00 | 14.00 | 143.00 |
| *HATS* | 325.00 | 19.00 | 293.00 | 1491.00 | 162.00 |
| *PANTS* | 682.00 | 14.00 | 852.00 | 56.00 | 659.00 |
| *BOOTS* | 829.00 | 140.00 | 609.00 | 120.00 | 87.00 |

*EXPENSES*

| | *JONES* | *SMITH* | *WALL* | *HARRI* | *HALL* |
|---|---|---|---|---|---|
| *SHOES* | 120.00 | 65.00 | 890.00 | 54.00 | 430.00 |
| *HATS* | 300.00 | 10.00 | 23.00 | 802.00 | 235.00 |
| *PANTS* | 50.00 | 299.00 | 1290.00 | 12.00 | 145.00 |
| *BOOTS* | 67.00 | 254.00 | 89.00 | 129.00 | 76.00 |

SLIDE 5

HEWLETT PACKARD

# APL

THESE WERE ENTERED INTO AN APL WORKSPACE AS FOLLOWS:

```
      REVENUES←4 5⍴190 140 1926 14 143 325 19 293 1491 162.□
□:
      682 14 852 56 659 829 140 609 120 87

      REVENUES
  190  140  1926    14  143
  325   19   293  1491  162
  682   14   852    56  659
  829  140   609   120   87
```

SLIDE 6

# APL

```
      EXPENSES
120   65   890   54   430
300   10    23  802   235
 50  299  1290   12   145
 67  254    89  129    76

      REVENUES
190  140  1926   14  143
325   19   293 1491  162
682   14   852   56  659
829  140   609  120   87
```

## COMMISSION

```
      .062×0⌈REVENUES-EXPENSES
 4.34   4.65  64.232   0       0
 1.55    .558 16.74   42.718   0
39.184  0      0       2.728  31.868
47.244  0     32.24    0        .682
```

```
COMMISSION←.062×0⌈REVENUES-EXPENSES
```

HEWLETT hp PACKARD

# APL

## PROFITS

```
      PROFITS←REVENUES-EXPENSES+COMMISSION

      PROFITS
 65.66      70.35     971.768   ¯40       ¯287
 23.45       8.442    ¯253.26    646.282    ¯73
592.816    ¯285       ¯438       41.272    482.132
714.756    ¯114        487.76    ¯9         10.318
```

## PROFITS BY PRODUCT LINE

```
      +/PROFITS
780.778  858.434  393.22  1089.834
```

## PROFITS BY SALES PERSON

```
      +/PROFITS
1396.682  ¯320.208  1274.788  638.554  132.45
```

## TOTAL PROFITS

```
      +/+/PROFITS
3122.266
```

SLIDE 8

# APL\3000

TWO IDENTICAL PROGRAMS
THE APLGOL ONE IS EASIER TO READ

## APL

```
[0]     TOSS;COIN
[1]     COIN←1?2
[2]     →(~COIN=1)/T
[3]     'HEADS'
[4]     →0
[5]     T:'TAILS'
```

```
      TOSS
TAILS
      TOSS
HEADS
      TOSS
HEADS
      TOSS
TAILS
      TOSS
TAILS
```

## APLGOL

```
[0]   PROCEDURE FLIP.COIN;
[1]     COIN←1?2;
[2]     IF COIN=1 THEN
[3]       'HEADS'
[4]     ELSE
[5]       'TAILS';
[6]   END PROCEDURE
```

```
      FLIP
HEADS
      FLIP
TAILS
      FLIP
HEADS
      FLIP
TAILS
```

## NEW PRODUCT ANNOUNCEMENTS

## WILL WORKMAN/HP

## DISTRICT SALES MANAGER - ROCKVILLE

Before I announce the HP1000 System, let me apologize for the delay you probably experienced during registration. About a week ago with Gary Green's encouragement, we decided to go "on-line" and provide a data base using our 3000 System in Rockville, Maryland, to handle registration and later for sorting the data for the USERS GROUP. As you probably observed, we were running behind schedule on the implementation.

Well, what about the HP System 1000? You are obviously aware that we have a 3000 System and possibly the System 2000. At our Data Systems Division in Cupertino, California, they decided it was time for the System 1000! What is it? The System 1000 will be announced in a press release on October 4th and followed by feature articles in the trade press during October and November. The System 1000 utilizes the newest member of the 2100 minicomputer family and will be used for a broad spectrum of computational, data acquisition and control, measurement system and operations management applications. The major features of the System 1000 are the faster 21MXE processor, a 2645A CRT Terminal with dual mag tape cartridges for program and data storage (no paper tape!), IMAGE 1000 for data base management, desk-style

configurations, on-line microprogramming support under RTE and lower prices!

So we see it replacing, in some respects, various configurations that you might be familiar with . . . the 9600 family of data acquisition systems, including the 9640. The System 1000 will be the primary nomenclature for minicomputer systems coming out of the Data Systems Division in Cupertino, California.

Let me try to compare the three families of computer systems that we now have on the market. The 3000 Systems that you are familiar with we see as a general purpose business data processing system for on-line plus batch operations. The ACCESS 2000 System which has been seen as a 32 terminal T/S system supporting BASIC . . . has been enhanced to support RJE using HASP capability (IBM 370) and a data entry library to support formatted data entry operations. The System 1000 that we are announcing is seen as a computation/instrumentation/data acquisition and control/operations management system for dedicated applications that can be written in ASSEMBLY/FORTRAN/BASIC with limited data base management requirements. The System 1000 should be important to owners of the HP3000 as a "front-end" processor for real time data acquisition/control applications and data entry/operations management applications in a distributed system environment.

At time of introduction, we will have four different models of the System 1000. Looking at the slide, you will notice that the physical configuration is based on a desk. The 21MXE processor is in the left-hand bay with the disc and disc controller

in the right-hand bay. The system console is the 2645A CRT Terminal with dual mag tape cartridges. This configuration, using the 7905 (15M-byte) disc and 64K-byte of memory in the 21MXE processor with RTE-II software, is priced at $37,500. The Model 31 uses the 7900 (5M-byte) disc in an up-right cabinet rather than the Model 30 desk and is priced at $33,500.

The Model 80 System 1000 includes a 128K-byte 21MXE processor, same disc and console as the Model 30, desk with single-bay cabinet housing a 1600 CPI 9-track mag tape drive, 200 LPM line printer with RTE-III operating system and IMAGE 1000 data base management software (with QUERY) and is priced at $62,600. The same equipment configured in dual cabinets (no desk) is identified as the Model 81 and is priced at $63,600. Delivery for all models is running from 8 to 12 weeks. If you compare the System 1000 pricing to our 9600 family and the price/performance compared to our competitors . . . I think you will find that HP is becoming more aggressive!

Let's take a closer look at the 21MXE processor! As the 2100 minicomputer has matured, we have had the opportunity to explore microcode implementation for speed and efficiency. Our analysis of the 21MX processor showed that most instructions could be implemented in 4 microcycles rather than the six currently used. Therefore, on the 21MXE, we have used variable microcycle length instructions to optimize the speed of the machine. As a result, the "XE" processor runs about 60% faster than the MX and many FORTRAN programs with floating point operations is running

about twice as fast. We have achieved this speed improvement for approximately 10% increase in price over the MX! Another important feature of the System 1000 is that microprogramming of the 1KW WCS option is now supported under the RTE operating system. Users of the System 1000 can write and debug 21MXE microcode from the system console or another terminal.

As far as future enhancements for the System 1000, we are working on software to link the System 1000 RTE to the 3000 MPE for program-to-program communications, remote terminal access to the 3000 via RTE and file transfer. On the processor side, we expect to announce faster memory and floating point hardware during 1977 to provide a significant increase in speed.

For those of you not familiar with the HP2645A CRT Terminal announced September 1st . . . I would like to describe some of its features which led to our decision to use it as the System 1000 console. The 2645A CRT Terminal uses a new microprocessor (INTEL 8080) which provides significant speed and memory addressing capability over the earlier 2640 and 2644 CRT Terminals. The 2645A CRT Terminal supports up to 9600 baud asynchronous communications in half and full duplex modes. We have also designed a new data communications card for supporting IBM bi-sync in non-HP computer environments. The eight special function keys can now be programmed to initiate the transmission of up to 80 characters for complex commands to the computer system, for controlling and formatting data on the mag tape cartridges or for issuing

"log-on" commands when connected to a time-share network with single keystroke operation. These "soft programming" capabilities of the 2645 CRT Terminal can be dynamically loaded from the computer or mag tape cartridges and will be used on the System 1000 to initiate system commands like a FORTRAN compilation using one keystroke . . . making the RTE operating system "friendlier". The mag tape cartridges will be used for loading system diagnostics or for program/data storage that the user can take away from the system for security reasons.

Let me summarize . . . the System 1000 will be the main system offered by Data Systems Division. As the year evolves, you will see a number of enhancements, certainly in the 21MXE processor. We will be announcing additional distributed systems software that will provide a powerful link between the System 1000 and the 3000 System you are familiar with. The System 1000 is aggressively priced using a desk-style configuration that will fit better in the commercial DP environment. We look forward to providing 3000 owners a broader solution to solving your instrumentation and data processing requirements with the System 1000.

That is all I have for new product announcements. I will be happy to answer any questions . . .

QUESTION . . . Will the 21MX minicomputer be discontinued?

ANSWER . . . No, not in the near future. The 21MX will remain in production as we add enhancements to the higher priced 21MXE processor. This will give our customers a broader range of alternatives.

NOTE: As of March 1977, HP is announcing the following enhancements to the System 1000:

1) Models 20 and 21 using the new RTE-M memory-based operating system with an entry price of $21,100 for the Model 20 which can support up to 608K-bytes of 4K RAM semiconductor memory and the HP-IB.
2) New 512K-byte flexible disc drive ("floppy") for an additional $4,500.
3) Faster 4K RAM 16K word memory (20-30%) with price reductions on the current 16KW memory modules.
4) RTE-M/BASIC for programming in the BASIC language from several terminals.

Delivery of these new System 1000 models and the hardware/software enhancements is in the 8 to 10 week range. The Model 31 has been reduced to $21,500 and the configurations and prices for the System 1000 are shown in the following table:

**TABLE 1. HP 1000 Basic Configurations**

| | Model 20 | Model 21 | Model 30 | Model 31 | Model 60 | Model 81 |
|---|---|---|---|---|---|---|
| Processor | 21MX E-Series | | | | | |
| Cabinet | Desktop | Upright | Desktop | Upright | Desktop | Upright |
| Main Memory Std Size (KB) | 64 | 64 | 64 | 64 | 64 | 64 |
| Opt. Expansion To | 304 | 608 | 304 | 608 | 304 | 608 |
| Operating System Standard/ Optional | RTE-M | RTE-M | RTE-II/RTE-III | RTE-II/RTE-III | RTE-III + IMAGE/1000 | |
| Disc Subsystem Type | Flexible Disc (optional) | | Cartridge Disc | | | |
| Capacity (MB) | 512K | 512K | 15M | 5 or 15M | 15M | 15M |
| Opt. Expansion | 2048K | 2048K | 365M | 365M | 365M | 365M |
| System Console | 2645 Display with Dual Mini cartridges (110KB) | | | | | |
| Standard Peripherals | | | | | | |
| Line Printer | None | None | None | None | 200-1250 lpm | |
| Mag Tape | None | None | None | None | 800-1600 bpi | |
| Base Prices | 21,000 or 25,500 | 22,000; or 26,500 | 36,500 | 31,500 or 36,000 | 62,000 | 63,000 |

Robert Young

ARTHUR - JOB and PROCESS Monitor for CX and Pre-Cx Machines

I must start, of course, by offering nearly all of you here congratulations on the 200th anniversary of the out-break of organized crime. It is organized crime as far as I am concerned because your ancestors were rebelling against my monarchy. From the tone of the meeting, it would appear that the spirit of the Revolution has not died but, instead of it being directed at George III, the imperial power, remote and majestical, is now Hewlett-Packard.

The origins of ARTHUR, like King Arthur himself, lie back in the mythology. The HP 3000 was a mythological sort of machine to us when we first got it. It had stacks and segments and a virtual memory and was a whole heap bigger than anything any of us had ever tackled at first hand. In 1973, when we took delivery of the machine, it came into our Institute as a mythological beast of unknown and awesome temperament. The staff of the Institute were mostly traditional chemists, that is they used the techniques of wet chemistry. In 1970 a new Director arrived who introduced more physical methods and it was he who was behind the selection of the HP 3000 for the Institute. Of the 600 scientific and technical staff at the Institute, there were no more than 10 who had any experience of computing of any kind at all at that time.

In those early days we used MPE 'B' and for the sort of work our users were doing, this was quite adequate. However, as time went on and we trained the staff, that is the Institute staff not the computing staff, these naïve users became increasingly confident and adventurous in their use of system resources. The first real bottle-neck was the absence of spooling. We overcame this by prevailing successfully on Hewlett Packard to supply us with a pre-release version of MPE 'C'. It was around then that our users seemed to gain a lot of confidence in a very short period and we quickly ran into trouble with other system limitations.

Repeated attempts to get Hewlett Packard to produce the necessary system performance monitoring software were of no avail and it was with considerable reluctance that we decided to do this work ourselves. Fortune smiled upon us and we were able to take on a new member of staff in the Computer Laboratory.

He came straight to us from University where he had been studying chemistry. He had been doing some work using ICL equipment but it was only as an adjunct to his chemistry; by no stretch of the imagination was John Sowden a systems programmer. The other staff of the Laboratory were, of course, committed very heavily to the existing work programme and Dr. MacIntyre, the Head of the Laboratory at that time, dropped the problem into the lap of this new lad. It is to his considerable credit that he was able to boot-strap himself into the position of producing ARTHUR in 9 months.

Hewlett-Packard people were only of use to us at the casual advice level. We were able to do the work only because we had already procured some documentation (by a process I shall not mention here): a book of system tables, a 'bedsheet' and a microfiche set, all of MPE 'B'. By using these three sets of documents and a considerable amount of native wit we were able to steer John through the early stages and then he was able to fly on his own and produce the necessary goods.

ARTHUR appeared in two stages. The first one monitored the system tables and produced a print-out on the consol of CPU times and stack sizes for any in-core processes that ARTHUR had picked up. The work we really wanted from ARTHUR was for him to forceably re-schedule work which was being run from a session and consequently in the CQ on our machine down to the DQ where we run JOBS, when that was appropriate. It is a characteristic of teaching a large, näive user base how to use the machine with the power of the 3000 that the users' programmes get bigger and bigger and bigger, so that without them being too aware of it there comes a point where sessional users are using stacks of 20k or more. Efficiency of computation is not something you can introudce to the näive user when he first closes with the machine. The introduction of the idea of computation efficiency has to be delayed until such time as the man himself becomes concerned about improving his throughput. Only then is he prepared to learn more about the details of the use fo the machine; prior to that such information would be considered by him to be useless.

We released information whereby the sort of stack size and CPU time that we considered to be legitimate for sessional work were explained, and by the aid of ARTHUR we were able to have the results monitored. With the first edition of ARTHUR we tapped people on the shoulder to have the ordinances enforced, but when the final version of ARTHUR became available, it became quite automatic.

Re-scheduling is done on the basis of a Process since it is Processes that take time and absorb resources. It is quite obvious that for a Process to have any existence at all there must be means within the operating system to set them up and destroy them. What we did not know was if it would be possible to re-schedule a Process once it had been set up by the operating system. A good look at the priviledged intrinsics in the microfiche set resulted in us finding REMOVE and INSERT which do just what their names suggest.

A copy of ARTHUR on magnetic tape, and the listing of the software, written in SPL are lying on the table outside and anyone who may wish to inspect it or have a xerox copy may do so. The tape is intended for the User Group Library but anyone who would like a copy before this becomes available should send a tape to me at Mill Hill directly; we can only handle 800 BPI.

ARTHUR runs as a JOB and makes use of GETPRIVMODE and so does not appear flagged as PRIV in SYSTEM DUMPS - HP shrug off PRIV USER mode DUMPS. It only takes a few tens of seconds CPU time in 24 hours so is not heavy on resources. You talk to it via the switch register and the console. The programme is very well documented: I claim that as being my particular contribution, there is hardly a line without a comment. (At this point the meeting applauded).

In the IBM Systems Journal number 4, 1975, there is an article entitled "Performance Analysis of Virtual Memory Time Sharing Sytems": dear HP, can we please have a similar article about the 3000 and NOT JUST FOR THE SERIES TWO MACHINES.

There is a useful utility not supported by Hewlett Packard called OVERLORD. Like ARTHUR, it runs as a JOB. Please, HP, could not ARTHUR and OVERLORD be

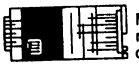
taken up and become a composite, supported utility? This would improve the overall use of the machine without asking HP to make alterations in the operating system with unknown and indeterminate effects throughout the system.

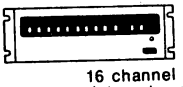
Up the Revolution.

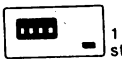
Tom McShane


The Vadic Corporation
505 East Middlefield Road
Mountain View, CA. 94043
(415) 965-1620

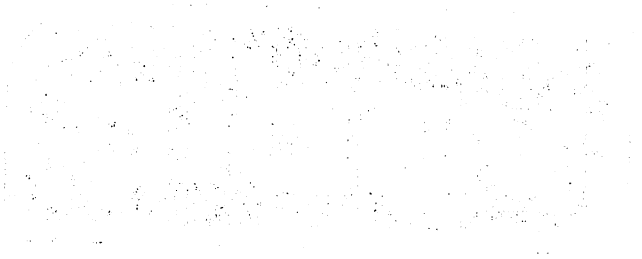

Bell & IBM type modem modules


Modem modules on PC boards

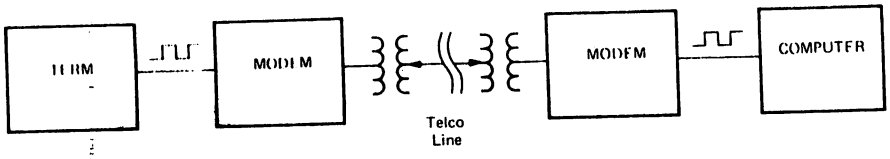

16 channel multiple data set system

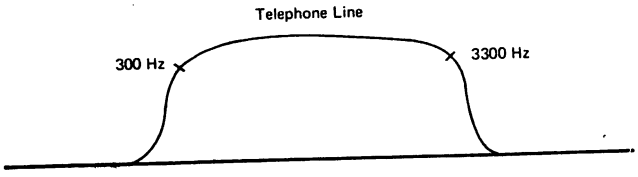

1 & 2 channel stand-alones

# Telecommunications From The User's Viewpoint

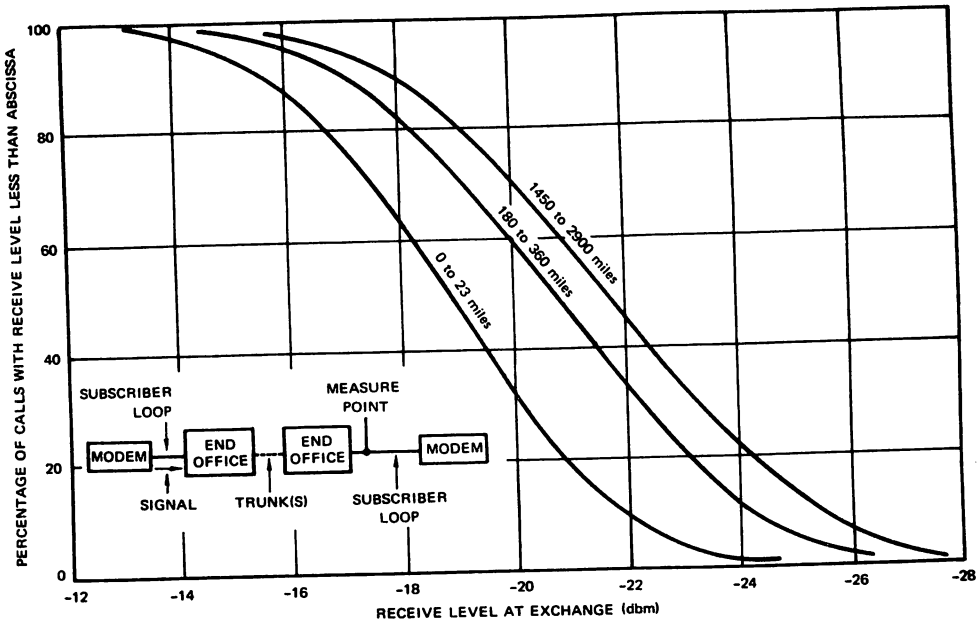


## WHY MODEMS?



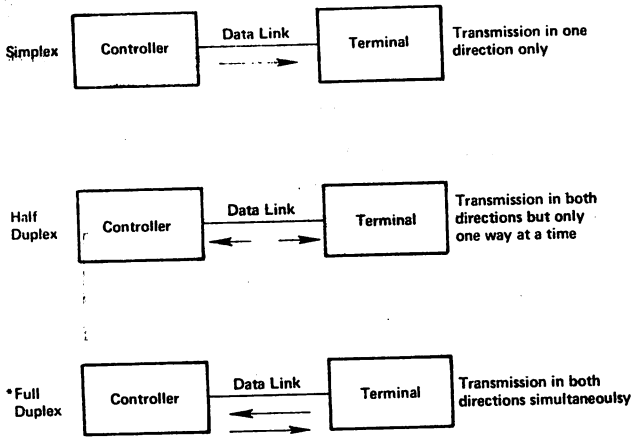

MO-DEM is a contraction of the words modulator - demodulator and is used to condition digital signals for transmission over the telephone network which was designed for analog voice signal transmission. The telephone network has a bandwidth of approximately 300 Hz to 3300 Hz, so the modems used on the telephone network must condition the signals to fit within the band.




**End Office Receive Levels on Direct Distance Dial Network**

Expected receive level between telephone exchanges on the DDD network is shown on a probabilistic basis. Subscriber loop loss at each end must be added to the interexchange loss.

**Communication Modes**




*Per ASCII standard full duplex implies that the same data rate exists in both directions simultaneously; i.e., 1200/150 bps is not full duplex, 1200/1200 bps is full duplex.

Communications terminology can be confusion. When the term "communication mode" is applied to modems the following nomenclature is used:

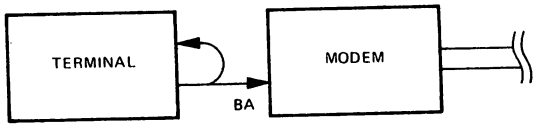
Simplex — transmission in one direction only with no way of responding. Your home TV reception of a signal transmitted from the TV station is an example of simplex communication.

Half Duplex — transmission in two directions, but only one way at a time. CB radio operators either transmit or receive, but cannot do both simultaneously on a single channel. At the end of transmission it is necessary to advise the other party when through transmitting and ready to receive by saying "over". Then the other operator can begin transmitting.

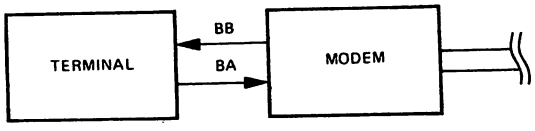
Full Duplex — transmission in both directions simultaneously. In the CB example, if two channels were used - one for transmitting and another for receiving - two simultaneous transmissions could be made.

## HALF/FULL DUPLEX OPERATION
## Terminal or Computer

**Half Duplex**
(implies that local copy is provided)

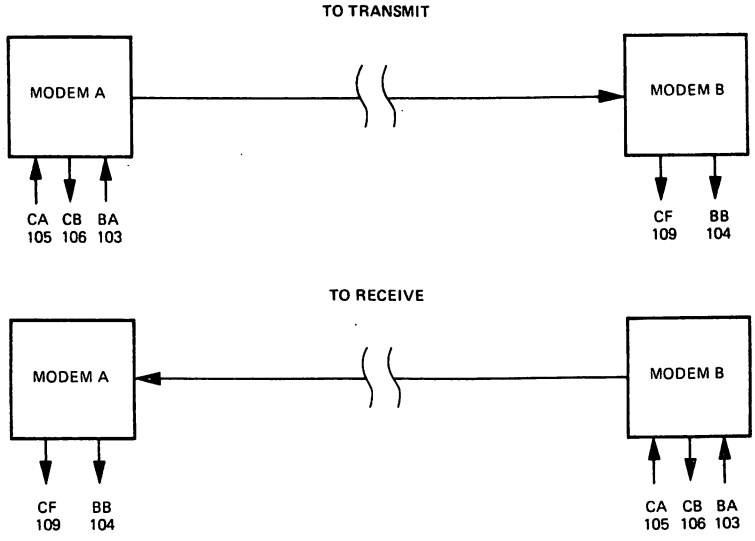



**Full Duplex**
(implies that no local copy is provided)
Means no local copy



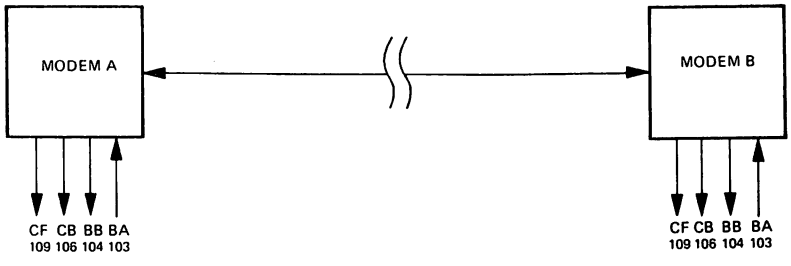

Terminal manufacturers often use the terms half duplex and full duplex to mean whether local copy is provided or whether the far end loops back (echoplexes) that which was transmitted. The presence or absence of local copy has nothing to do with the communications mode of the data link.

# HALF DUPLEX MODE



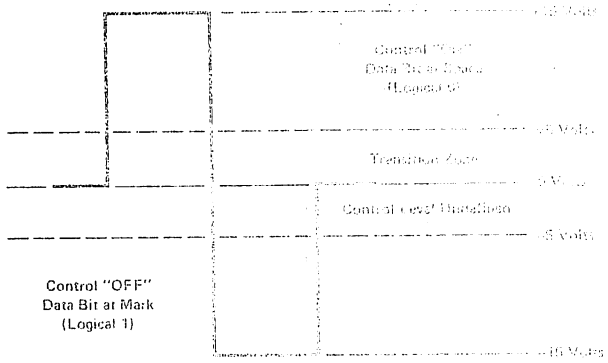

Control signals required to turn on modem A's transmitter and cause modem B's receiver to respond are shown in the top of the slide. The process is reversed in the lower portion.

# FULL DUPLEX MODE




Interface signals for transmitting and receiving data simultaneously are shown at both ends of the full duplex data link.

RS232 Voltage Transition Levels




EIA standard RS-232-C defines the interface between data terminal equipment and data communications equipment employing serial binary data. Mechanical and electrical specifications are provided. The slide shows the voltage transition levels at the interface point when a connection exists between the terminal and communications equipment.

The lights display:

BA Transmit Data (103)
BB Receive Data (104)
CA Request to Send (105)
CB Clear to Send (106)
CC Data Set Ready (107)
CD Data Terminal Ready (108/2)
CE Ring Indicator (125)
CF Carrier Detector (109)

The slide shows the more common EIA interface leads as they appear on the diagnostic panel of VADIC's sixteen channel multi chassis. International designations per CCITT standards appear in parenthesis.

# HOW ARE MODEMS SPECIFIED?

I. INTERFACE Computer or Terminal
- A. RS232C
- B. Bell Telephone
- C. IBM
- D. MIL 188
- E. CCITT
- F. Other

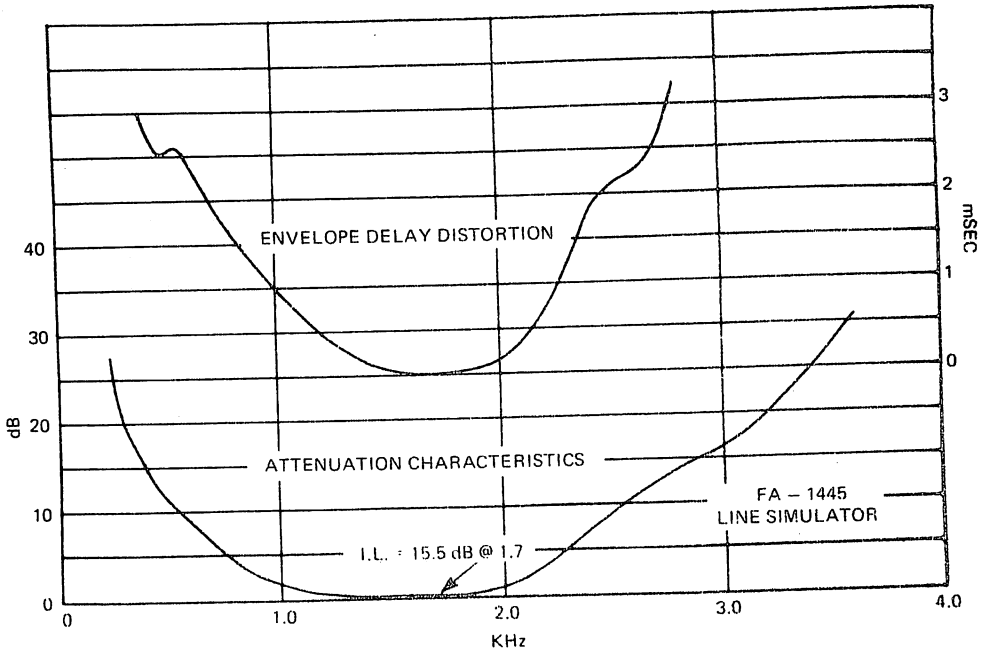
II. INTERFACE Telephone Line
- A. Dial or switch network
- B. Private, leased or dedicated line
  - 1. 2 wire
  - 2. 4 wire
- C. Voice grade
- D. DDS
- E. Special Carrier

III. SPEED

| | | |
|---|---|---|
| A. | Low speed | 0 to 300 BPS |
| B. | Medium speed | 0 to 1800 BPS |
| C. | High speed | 2000 to 9600 BPS |
| D. | Very high speed | 19.6K BPS to 250K BPS |

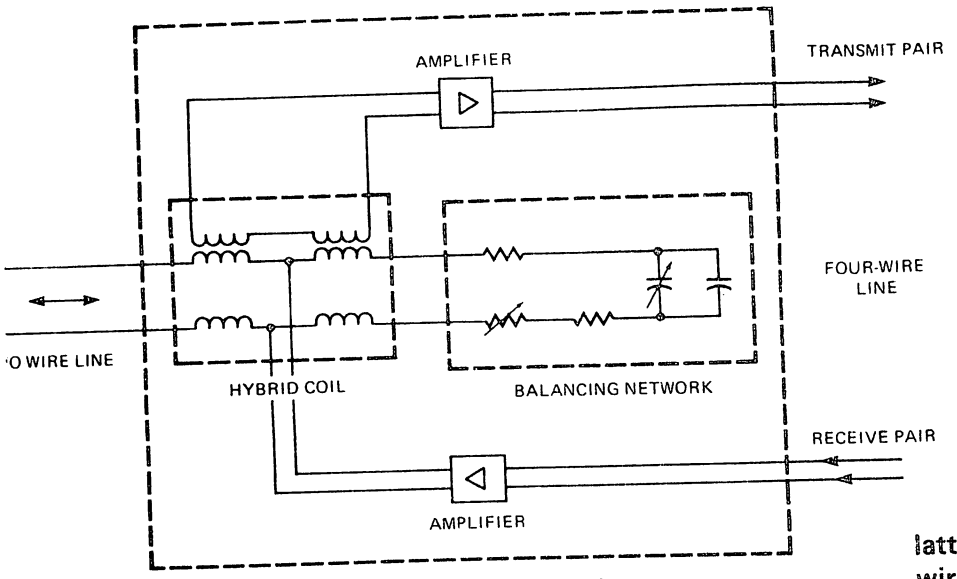
IV. COMMUNICATION MODE
- A. Simplex
- B. Half duplex
- C. Full duplex

Modem specifications include business machine and phone line interface as well as speed, communications mode, and performance criteria.




**Envelope Delay Distortion & Attenuation**
**3002 unconditioned voice grade line**

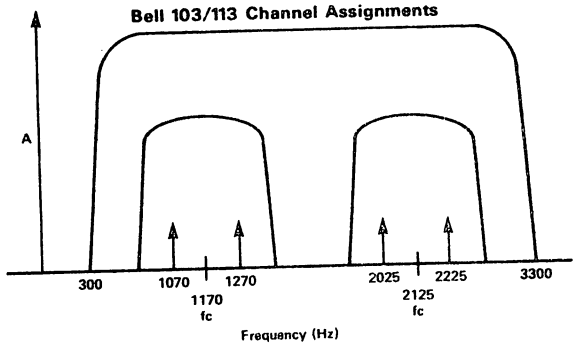
Because different frequencies encounter different amplitude attenuation and propagation delay times through the telephone network, not all of the bandwidth can be utilized for the transmission of digital data. These differences are largely immaterial in voice communications, but can be very detrimental to data transmission, particularly at speeds above 2400 bps.




2/4-Wire Terminating Set

The subscriber loop is basically a continuous loop of wire without amplifiers. At the central office the transmitted signal (voice or data) is sent to the far end office on a unidirectional channel called the transmit pair. This channel is simplex; one way only due to the audio amplifiers which exist. The far end transmitted signal appears at the near end on the receive pair. The transmit pair and receive pair are referred to by telco personnel as 4-wire lines. Leased circuits can be provided on a 4-wire or 2-wire basis. The latter case involves combining the signals in a 2/4 wire terminating set (hybrid). The subscriber loop is a 2-wire bi-directional channel for dial up calls.

## Bell 103/113

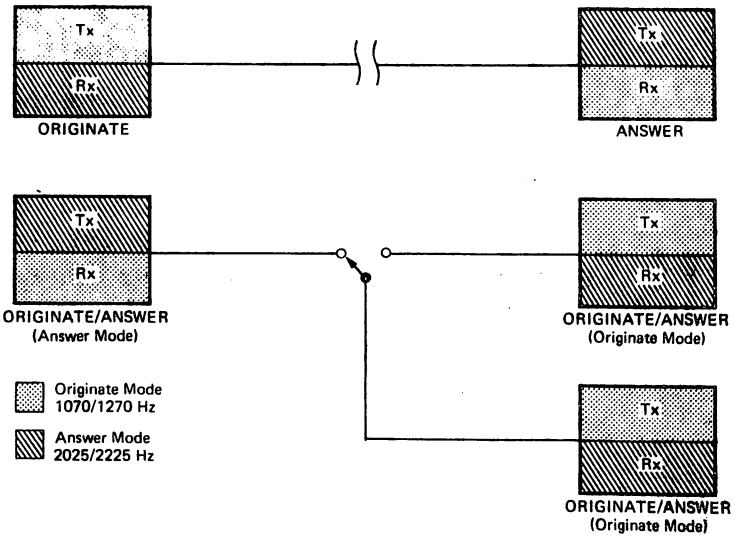



**Specifications**

| | | | |
|---|---|---|---|
| **Data:** | Serial, binary, asynchronous, full duplex | | |
| **Data Transfer Rate:** | 0 to 300 bps | | |
| **Modulation:** | Frequency Shift Keyed (FSK) FM | | |
| **Frequency Assignment:** | | Originating End | Answering End |
| | Transmit | 1070 Hz space<br>1270 Hz mark | 2025 Hz space<br>2225 Hz mark |
| | Receive | 2025 Hz space<br>2225 Hz mark | 1070 Hz space<br>1270 Hz mark |
| **Transmit Level:** | 0 to -12 dBm | | |
| **Receive Level:** | 0 to -50 dBm simultaneous with adjacent channel transmitter at as much as 0 dBm | | |

Specifications and channel assignments for the full duplex 300 bps asynchronous Bell 103/113 modem are shown in this slide.

# 103/113
# ORIGINATE/ANSWER




**The Bell 103 modem can transmit and receive in either the low or high band. The ability to switch modes has been termed "originate and answer". The Bell 113A operates only in the originate mode; the Bell 113B operates only in the answer mode.**

# BELL 103/113 FAMILY
# 0 to 300 BPS

DIAL-UP

103A2 or A3
- Originate/answer
- Single channel—self contained
- $25 to $35/month

103E Data Station—Computer Site
- Originate/answer
- 40 channels max
- $25 to $35/month

113A Terminal Data Set
- Originate only
- Single channel—self contained
- $10 to $15/month

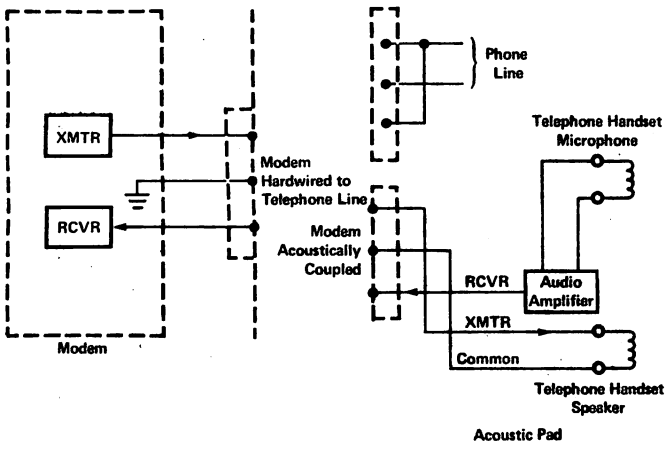
113B Data Station—Computer Site
- Answer only
- 20 channel increments—240 max
- $12 to $20/month Average $16 to $18

LEASED LINE

103F
- Originate/answer
- Single channel—self contained
- $25 to $35/month

**Bell 103/113 models and prices are shown in this slide.**

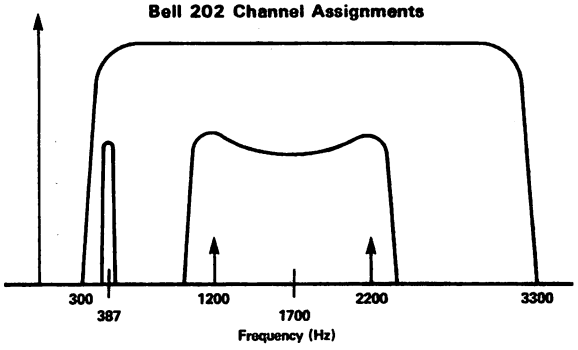
Acoustic vs. Hardwired Modem




Modems can be directly connected to telephone lines via a "hardwired" electrical connection. Where portability is required low speed modems can be acoustically connected via the transducers in an acoustic pad and the telephone handset. There is usually some performance degradation due to the distortion introduced by transducers reducing the maximum speed capability and receiver sensitivity of acoustic couplers compared with hardwired modems.

# Bell 202

Bell 202 Channel Assignments



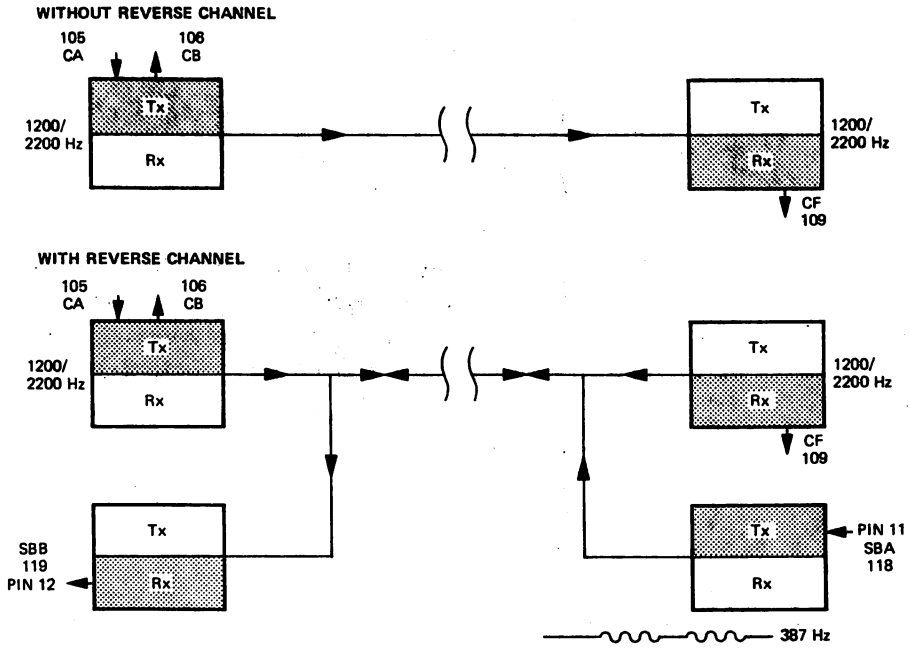

**Specifications**

| | |
|---|---|
| Data: | Serial, binary, asynchronous, half duplex on 2 wire lines |
| Data Transfer Rate: | 0 to 1200 bps – switched network<br>0 to 1800 bps – leased lines with C2 conditioning<br>Optional 5 bps AM reverse channel transmitter and receiver available for switched network units |
| Modulation: | Frequency Shift Keyed (FSK) FM |
| Frequency Assignment: | Mark – 1200 Hz<br>Space – 2200 Hz |
| Transmit Level: | 0 to -12 dBm |
| Receive Level: | 0 to -50 dBm switched network<br>0 to -40 dBm leased network |

Bell's 202 modem provides for 0 to 1200 bps communication over the dial network and 0 to 1800 bps over leased lines with $C_2$ conditioning per Bell's specifications, but in practice the 202's do not require conditioned lines even at 1800 bps. The 202 is half duplex with a two wire circuit (leased or dial up) and full duplex with a 4-wire circuit. Optionally available is a 5 bps reverse channel which operates in the low end of the 300 Hz to 3300 Hz channel and is primarily used for supervisory control. For instance, this channel could be used to transmit an "out of paper" signal from a remote printer back to the CPU to terminate data transmission until the paper could be reloaded in the printer.

A significant factor to consider when using the 202 type modem is the line turnaround time required for switching from the transmit mode to the receive mode. Echo suppressors in the telephone equipment that are required for voice transmission must be turned off by the modem to transmit digital data. These echo suppressors turn on if there is no energy in the band for 100 miliseconds (.1 second). The modem must provide a 200 ms signal to the line to turn off the echo suppressors every time it goes from transmit to receive mode - hence, if short records are being transmitted, such as in a graphic terminal application, the turnaround time can slow your anticipated 1200 baud (120 character/second) throughput to possibly 60 characters/second or less on an actual basis. Also since the CPU must be able to control the Request to Send lead of the Bell 202 for half duplex transmission, you should check your computer specifications to determine if this control is available. Most mini's do not have the capability of supporting half duplex 202, but can support 103 line discipline.

# 202 DIAL UP




**This slide shows the interface signals required to operate the half duplex 202 - with or without the reverse channel.**

## BELL 202 FAMILY

**DIAL-UP**

202C5, 9 or 11
- 0 to 1200 BPS
- Single channel, self contained
- Without reverse channel
- $35 to $50/month

202C6, 10 or 12
- 0 to 1200 BPS
- Single channel, self contained
- With reverse channel
- $35 to $50/month

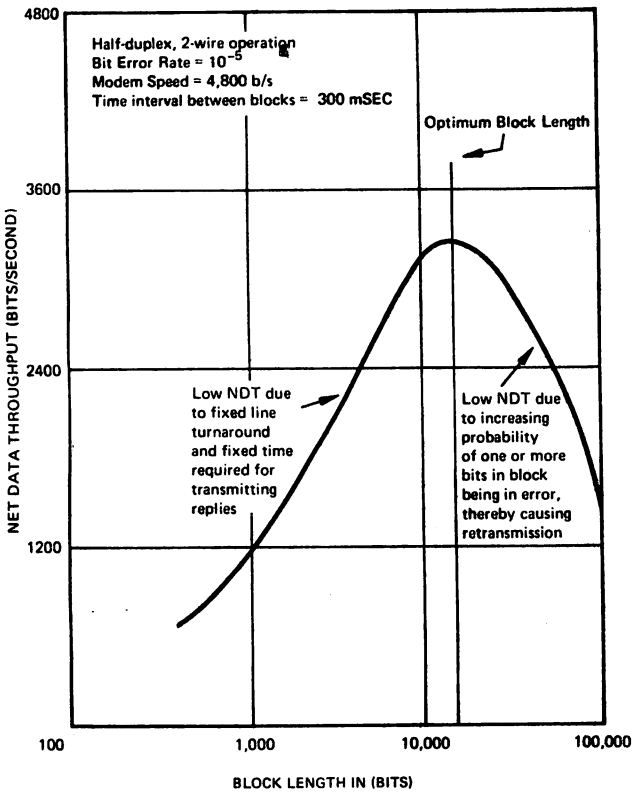
202S
- 0 to 1200 BPS
- Single channel or up to 8 channels in one housing
- Reverse channel optional
- $20 to $40/month

**LEASED LINE**

202D/202R
- 0 to 1800 BPS
- Single channel, self contained
- $35 to $50/month

202T
- 0 to 1800 BPS
- Single channel or up to 8 in one housing
- $20 to $40/month

**Bell 202 models and prices are shown in this slide.**

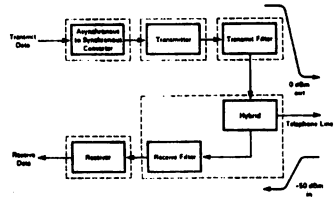



**Effect of Block Length on Net Data Throughput (NDT)**

**This slide shows the effect of block length on net data throughput. The curve is taken from an excellent article on throughput which appeared in the July/ August 1974 issue of Data Communications magazine.**

# Vadic VA3400

**VA3400 Block Diagram**



**Specifications**

| | |
|---|---|
| Data: | Serial, binary, asynchronous or synchronous, full duplex |
| Data Transfer Rate: | 1200 bps |
| Modulation: | Dibit Phase Shift Keying (DPSK) |
| Transmit Level: | 0 to -12 dBm |
| Receive Level: | -15 to -50 dBm (or 0 to -35 dBm) simultaneous with adjacent channel transmitter at up to 0 dBm |

Channel Assignment:

| | Originating End | Answering End |
|---|---|---|
| Transmit | High Band | Low Band |
| Receive | Low Band | High Band |

**Phase Shift**

| Dibit | VA3400 |
|---|---|
| 00 | 90° |
| 01 | 0° |
| 11 | 180° |
| 10 | 270° |

**A block diagram and specification summary of VADIC's full duplex VA3400 modem is shown. This modem operates at 1200 bps full duplex on 2-wire dial up or leased lines. The full duplex line discipline makes it possible to simply change modems and quadruple data speed over the Bell 103.**

# VA3400 FEATURES

1200 bps full duplex on dial up or 2 wire leased lines

Full duplex Protocol similar to Bell 103 (V21)

Abort timer disconnect

Loss of carrier disconnect

Remote controlled loopback

Echoplex operation

10:1 Improvement in error rate compared to FSK 1200 bps Modems

Terminal break capability provided

Eliminates half duplex turn around delay

Positive indication of connect status

Simplifies terminal operation

**The slide shows the VA3400 features.**

# Bell 201B/C

**Specifications**

| | |
|---|---|
| **Data:** | Serial, binary, synchronous, half duplex on 2 wire lines |
| **Data Transfer Rate:** | 2400 bps |
| **Modulation:** | Dibit Phase Shift Keying (DPSK) |
| **Transmit Level:** | 0 to -14 dBm |
| **Receive Level:** | -15 dBm to -50 dBm or 0 dBm to -35 dBm |
| **RTS/CTS Delay:** | 0, 7.5 msec, or 150 msec |

**Phase Shift**

| Dibit | 201B/C | 201A |
|---|---|---|
| 00 | 45° | 0° |
| 01 | 135° | 90° |
| 11 | 225° | 180° |
| 10 | 315° | 270° |

Note: *201A 2000 bps dial up lines replaced by 201C.*

**Bell 201 modems are 2400 bps synchronous half duplex 2-wire or full duplex 4-wire units that use a modulation scheme similar to the VADIC VA3400. The 201A is an obsolete 2000 bps unit, 201B is primarily for private or leased line applications, and the 201C is for dial up applications.**

# BELL HIGH SPEED
# All Synchronous

**201A**
2000 BPS
Single channel, self contained
Dial-up
$55 leased; $70 dial-up

**201B**
2400 BPS
Single channel, self contained
Leased line
$55 leased; $70 dial-up

**201C**
2400 BPS
Single channel or up to 6, 12 or 30
In common cabinet
Dial-up or leased line
$55 leased; $70 dial-up

**Bell 201 models and prices are shown in this slide.**

## BELL HIGH SPEED

**203—OBSOLETE**

**208A**
4800 BPS
Single channel, self contained
Leased line—no conditioning
$125/month

**208B**
4800 BPS
Single channel, self contained
Dial-up
$150/month

**209A**
9600 BPS or multiplexed increments of 2400 bps
Single channel, self contained
Leased line, D-1 conditioning
$230/month

**Bell high speed as well as other Bell modems are discussed.**

## OTHER BELL MODEMS

**303 Series**
19.2, 50 and 230.4K BPS
Single channel, self contained
Special 5000 & 8000 com lines

**403 Series**
Up to 60 Char/Sec
Single channel, self contained
Leased or dial-up
$40 to $60/month

# DATA ACCESS ARRANGEMENTS

**Manual:** CDT – 1000A
Manual only operation
$2.00 to $3.00/month

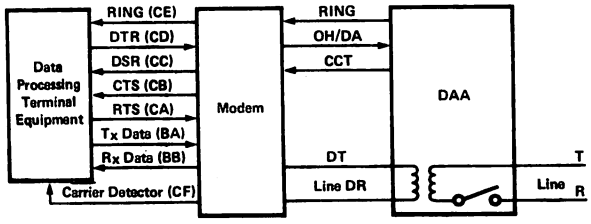
**Automatic:** CBS – 1001A or 1001F
Automatic operation
Voltage interface
$6.00 to $7.00/month

CBT – 1001B or 1001D
Automatic operation
Contact closure interface
$4.00 to $5.50/month

**Functions**

- Protect terminal and modem from line surges
- Provide DC isolation/impedance matching via transformer
- Protect network for excessive power to line
- Perform line transfer–handset to modem and vice versa
- Auto answer/dial (CBT or CBS only)

**Typical Automatic DAA Configuration**

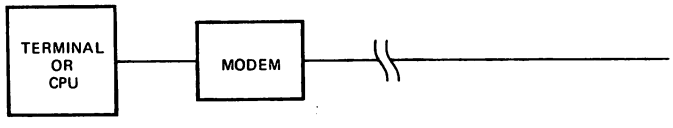



**Connecting a non-telco modem to the switched network has required the use of a protective device called a data access arrangement (DAA). During 1975 the California Public Utilities Commission began certifying non-telco DAA's so that independent manufacturers such as VADIC could connect their equipment to the network via a customer provided (VADIC provided) DAA. The FCC is now proposing that all the state PUC's and telephone companies allow the DAA to be part of the equipment provided by independent manufacturers.**

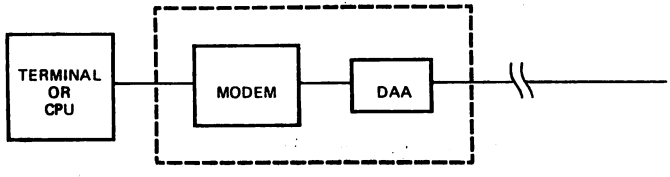
# REPLACING BELL MODEMS

1. Leased line
   No DAA required
2. Dial-up
   DAA required

BELL




NON-BELL




**Until the California PUC decision in mid 1975 the non-Bell modem had to have a separate telco provided DAA connected to it. The Bell modem connected directly to the line, as shown in the top of the slide.**

# IBM Line Adapters

**Specifications**

**Data:** Serial, binary, asynchronous, half duplex

**Data Transfer Rate:** Limited Distance Adapters - 0 to 134.5 bps (& 600 bps)
Leased Line Adapters – 0 to 600 bps
Shared Line Adapters – 0 to 134.5 bps

**Modulation:** Frequency Shift Keyed (FSK) FM

**Specifications Summary**

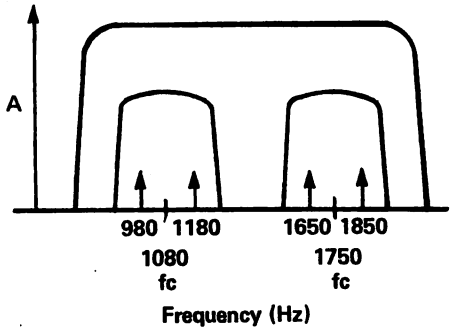
| Line Adapter | Type | Max. Miles (km) | Modulation | Bit Rate | 2W or 4W | Mark Freq. (Hz) | Space Freq. (Hz) | Input Impedance ohms | Output Impedance ohms | Termination Resistance (ohms) |
|---|---|---|---|---|---|---|---|---|---|---|
| Limited Distance | 1A | 4.75 (7.65 km) | FSK | 134.5 | 2W | 1170 | 1830 | 3000 | 3000 | As Req. |
| | 1B | | | | 4W | | | | 5000 | |
| | 2A1 | 8.25 (13.1 km) | FSK | 134.5 | 2W | 1000 | 2200 | 600 or 8200 | 600 or 8200 | As Req. |
| | 2A2 | | | 600 | | | | | | |
| | 2B | 8.25 (13.1 km) | FSK | 600 | 2W | 1000 | 2200 | 600 or 5000 | 600 or 5000 | As Req. |
| Leased Line | 1A | No Limit | FSK | 600 | 2W | 1400 | 2000 | 600 | 600 | 680 |
| | 1B | | | | 4W | | | | | |
| Shared Line | 1A | No Limit | FSK | 134.5 | 2W | 820<br>1230 | 990<br>1400 | 600 | 600 | 680 |
| | 1B | | | | 4W | 1640<br>2050 | 1810<br>2200 | | | |

IBM manufactures a line of asynchronous modems which they call line adaptors for use on leased lines. The specifications are summarized on the slide.

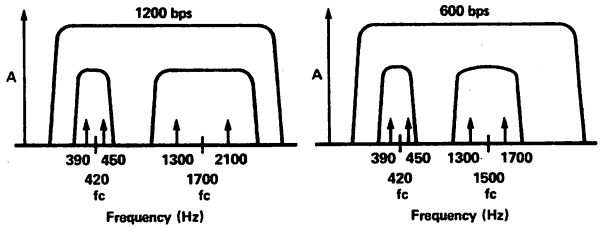
# CCITT Modems

| | | |
|---|---|---|
| V21 | (0 to 300 bps) | (similar to Bell 103/113) |
| V23 | (0 to 1200 bps) | (similar to Bell 202) |
| V26 | (2400 bps) | (similar to Bell 201) |

**V21 Channel Assignments**




**V23 Channel Assignments**

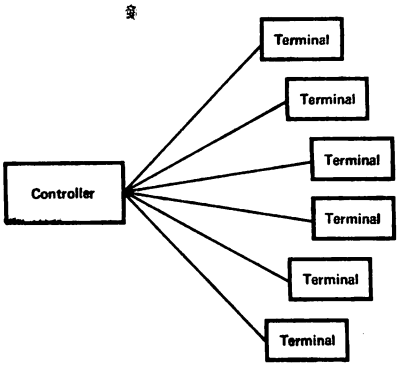



**V26 Data Encoding**

| DIBITS | Type A Phase Shift | Type B Phase Shift |
|---|---|---|
| 00 | 0° | 45° |
| 01 | 90° | 135° |
| 11 | 180° | 225° |
| 10 | 270° | 315° |

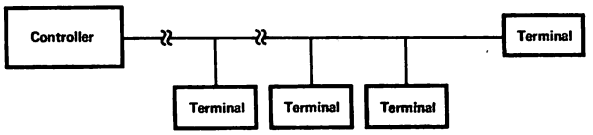
Outside the U. S. data transmission standards are set by the International Telegraph & Telephone Consultative Committee (CCITT) which is a part of the International Telecommunications Union in Geneva. Standards V.21, V.23, and V.26 describe modems similar to the Bell 103, 202, and 201 respectively.

Point-to-Point Network



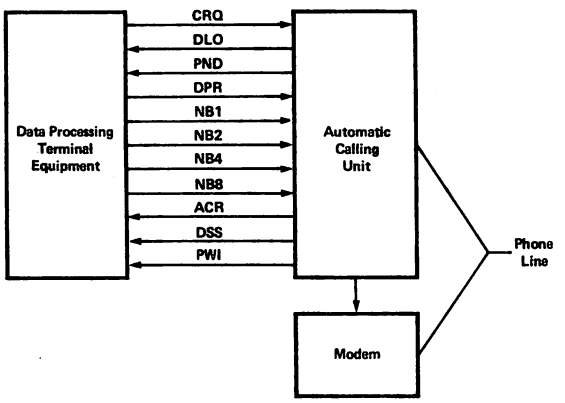

Point-to-point leased line networks are similar to what is shown at the top of the slide. All dial up connections are point-to-point. Multipoint networks have more than one remote station for each central station.

Multipoint Network




# Bell 801

Automatic Dialing




801A Pulse Dialer

PND on 100 ms times # pulses, then off 1 sec; i.e., approx. 15 sec to dial 10-digit number

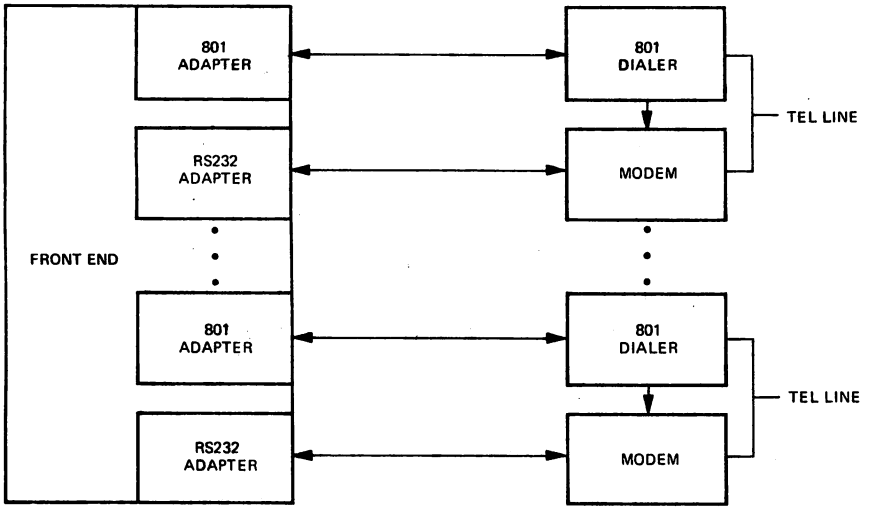
801C Touch Tone Dialer

PND on 50 msec, then off 70 msec; i.e., approx. 1 sec to dial 10-digit number

Automatic dialing via the Bell 801 auto calling unit requires a unique interface between the business machine and ACU. This interface is defined in EIA specification RS366. The interface signals are as follows:

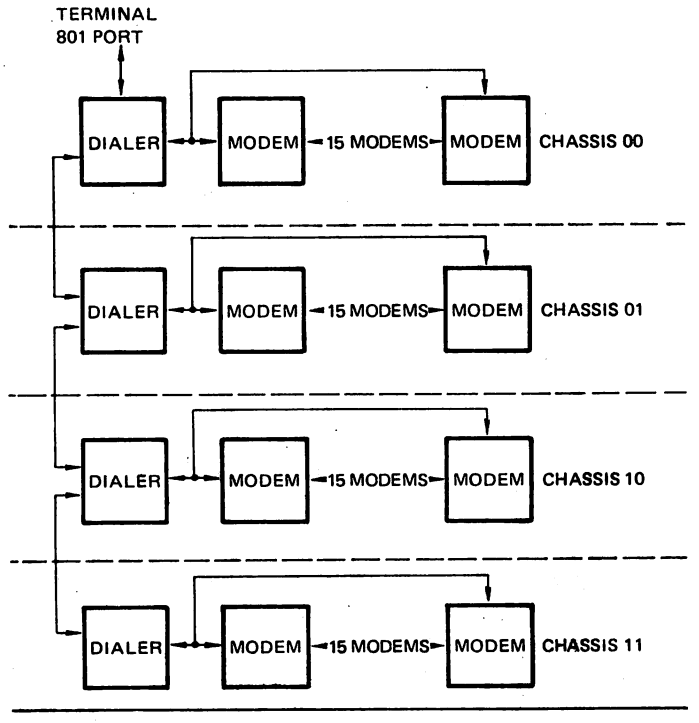
| | |
|---|---|
| CRQ | Call Request |
| DLO | Data Line Occupied |
| PND | Present Next Digit |
| DPR | Digit Present |
| NB1 | Digit Value 1 |
| NB2 | Digit Value 2 |
| NB4 | Digit Value 4 |
| NB8 | Digit Value 8 |
| ACR | Abandon Call & Retry |
| DSS | Data Set Status |
| PWI | Power Indication |

## NORMAL ACU CONFIGURATION



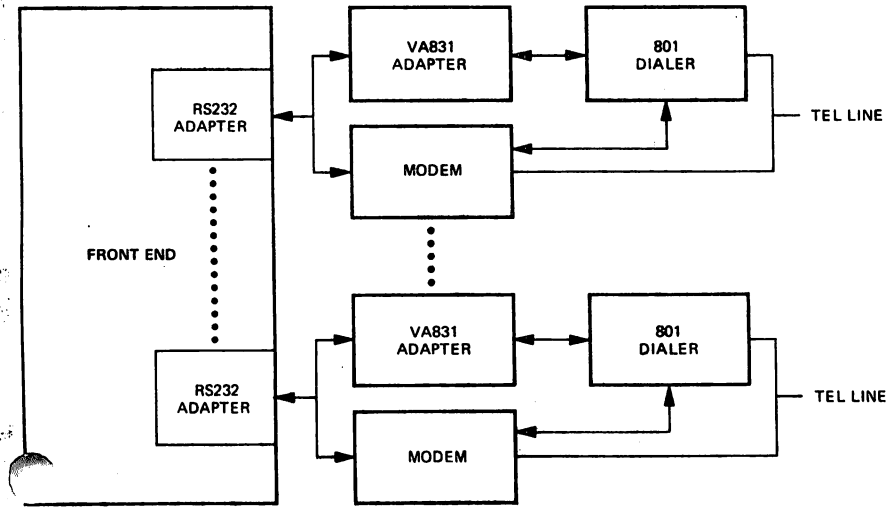

Each 801 auto calling unit requires its own front end 801 RS366 adaptor. A single modem is associated with each dialer. The modem/ACU combination is attached to a single phone line.

**MACS Block Diagram**



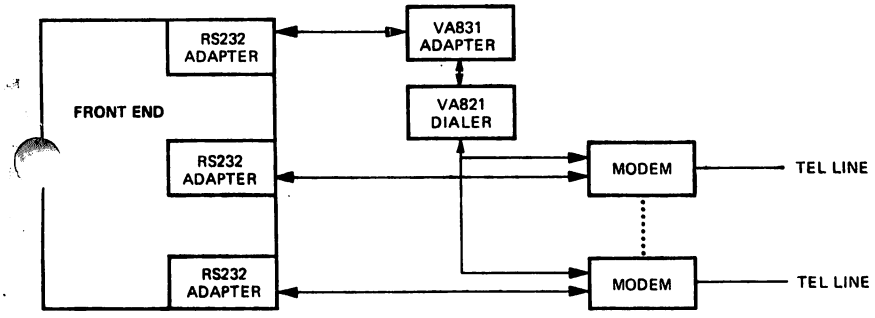

VADIC's multiline automatic calling system is diagramed in this slide. A single RS366 ACU port connects to as many as 60 phone lines. Each MACS dialer, VA821, can handle up to 15 modems.

## SYSTEM WITH RS232 TO RS366 ADAPTER

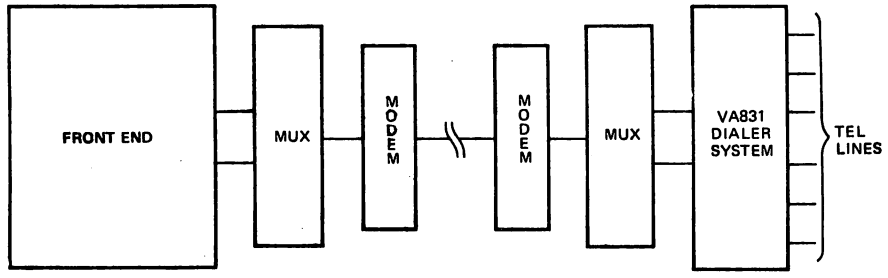



VADIC's VA831 adaptor enables a computer to initiate the dialer and transmit dialing information – digits and supervision – to an 801 pulse or touch tone dialer via an RS232 modem interface. All signaling takes place through characters sent from the computer on Transmit Data, and received by the computer on Received Data.

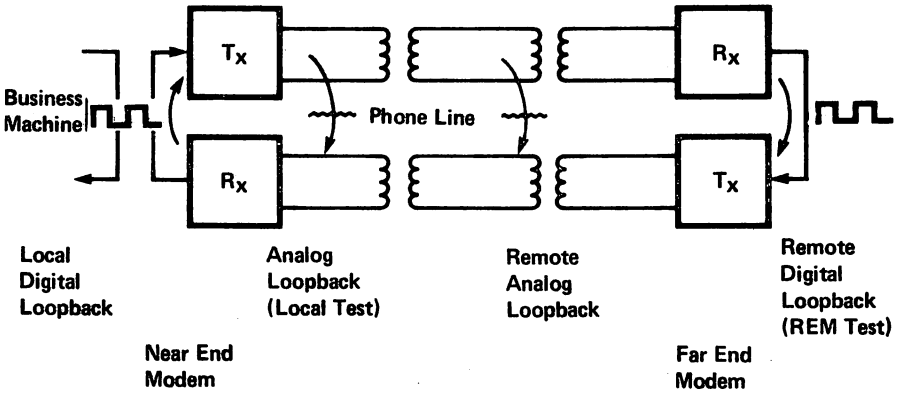
## SYSTEM USING VA821 MULTILINE DIALER




## MULTIPLEXER CONFIGURATION

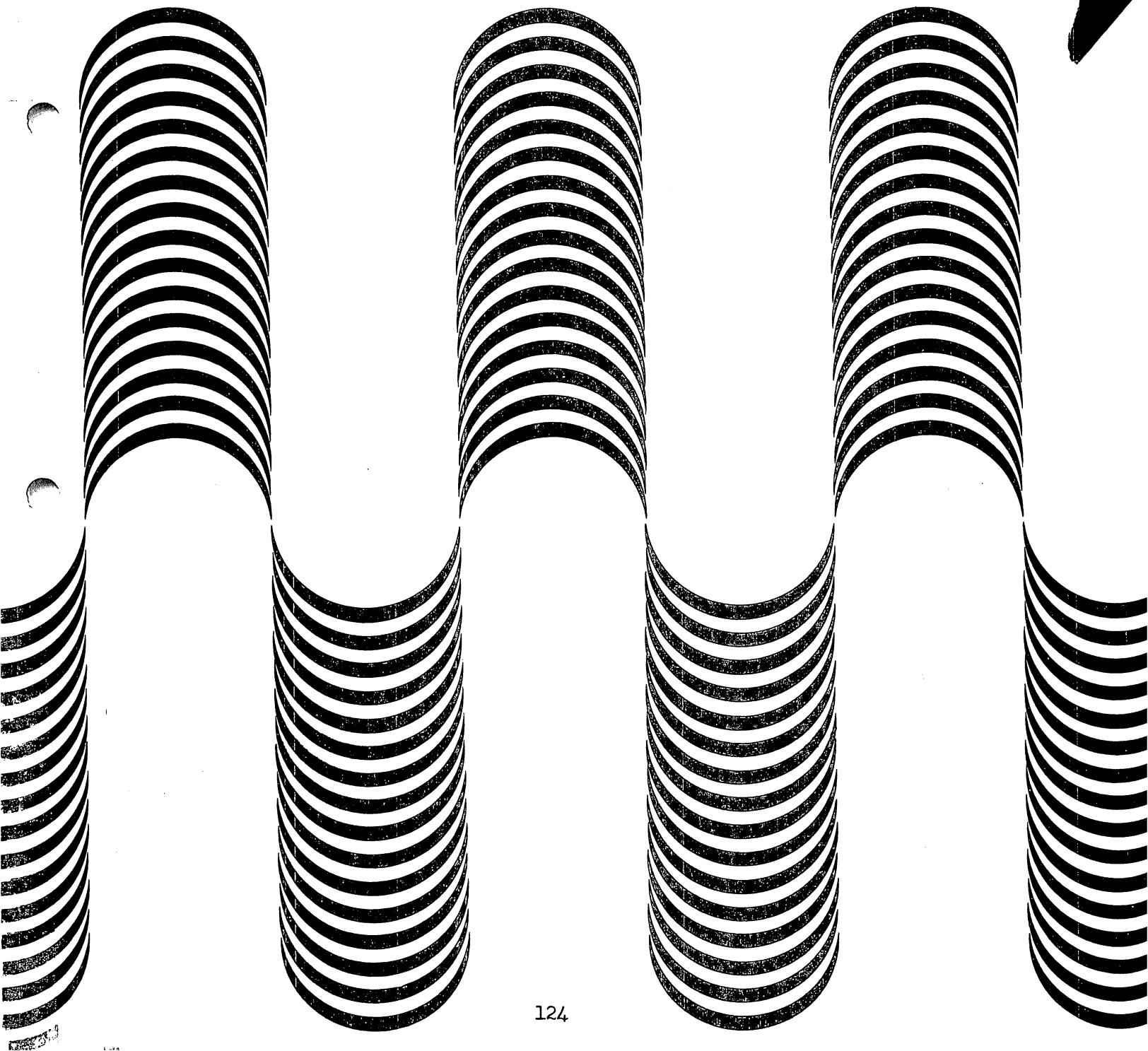



When VADIC's VA821 multiline dialer is used the RS232-RS366 adaptor requires a dedicated RS232 port. The VA831 makes it possible to have automatic calling through a multiplexer port that does not pass 801 interface signals. As shown in the lower portion of the slide, an automatic dialer can be located at the distant end of a multiplexer link, and accessed through a conventional RS232 interface without hardware modifications.




Analog loopback refers to connecting the transmitter's (analog) output to the receiver's input of the same modem. (For full duplex modems the transmitter and receiver must be switched into the same channel since normally they are in opposite channels.) The receiver demodulates the transmitted signal; agreement between transmit and receive data thus confirms that all parts of the modem are working correctly.

Digital loopback refers to connecting the receiver's (digital) output to the transmitter's input. A message sent from the near end modem which is looped back digitally at the far end and successfully received is a complete check that both modems and the communication link are working satisfactorily.

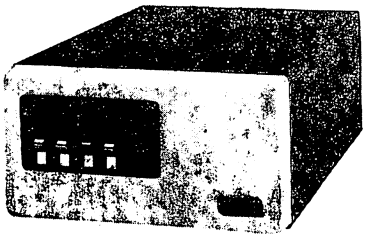


# Vadic Modems—A Whole Family of Modules, Cards & Chassis for OEM's & End-users.

## MODULAR CONCEPT FOR THE OEM


Vadic's data set modules (Bell 103, 202, IBM, CCITT) are specifically designed for incorporation inside remote terminals and data communication systems. The modules combine superior performance and low cost with packaging flexibility that enables rapid and inexpensive custom printed circuit board configuration. Vadic's modules also provide broad applications flexibility—full transmit/receive (with or without reverse channel for the 202 and CCITT data sets), transmit only/receive only operation, switched or leased lines, point-to-point or multipoint. Vadic manufactures custom circuit boards to EOM specifications for interconnect characteristics and physical configuration. All VA300, VA1200, VA2400, VA2700, and VA3400 Series modems as well as automatic dialers are available in custom configurations.

## MDS CONCEPT FOR THE END USER



Picture a whole new generation of computer-connected modem hardware and you've got Vadic's Multiple Data Set System (MDS). There's nothing like it for size, flexibility, and performance. Bell 103, 202, 201 compatible modems, Vadic's unique VA3400, IBM modems, and automatic dialers fit in seven inches of rack space—intermixed in any way. And to put the system even further ahead, we built in display and diagnostic functions that make troubleshooting an entire communications system easy.
We didn't stop there. We added a one- and two-channel chassis for terminal clusters and multiplexer sites—with identical diagnostics plus cabinets to hold the modems and associated telephone line connecting equipment.
In data communications, Vadic matches both ends of the line.

## VA3400: 1200 BAUD FULL DUPLEX, 2-WIRE MODEM

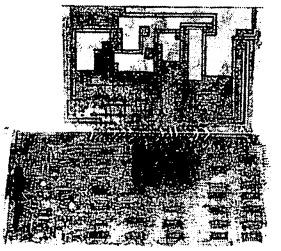
Here's a modem to make you sit back and take note—full duplex, 1200 baud on a two-wire circuit! This precedent setting modem has the speed of a Bell 202, with the simple line protocol of the 103 and can replace either one without any software modifications; it can be used on switched networks or leased lines.

### IT'S FAST

If your front end and remote terminals are capable of 1200 baud operation, you can make your data system run four times as fast simply by replacing 103's or Vadic VA300's with the VA3400's. Take two minutes to change board straps and a VA3400 can resemble a Bell 202 or VA1200. But since the VA3400 is full duplex, with continuous carrier in both directions, turnaround delay is zero. On some systems this means substantial performance improvement. You can also make it easier to deal with the disconnect and line outage problems so common in half-duplex operations, because the VA3400 incorporates an optional abort timer that disconnects a call if carrier is not detected within 12 seconds of answering a call and a carrier-off disconnect that automatically puts the data coupler on-hook if incoming carrier is lost.

### ASYNCHRONOUS/SYNCHRONOUS

VA3405 modems have an integral asynchronous/synchronous buffer, which allows use of the identical simple line discipline used by Bell 103 or VA300 modems. The VA3410's are synchronous and offer something new if you want to design your 1200 baud system to have higher throughput and lower line costs.



## VA300: 300 BAUD BELL 103 COMPATIBLE MODEM

Vadic's VA300 series features every 103 or 113 configuration offered by Bell, plus system capabilities and options not available from Bell. Standard Vadic transmitter and receiver modules form the analog functions of each unit. We add controls, interfacing, and diagnostics hardware to create a family of modems that give unexcelled performance and flexibility, yet hold their own in price.

### SWITCHED NETWORK MODEMS

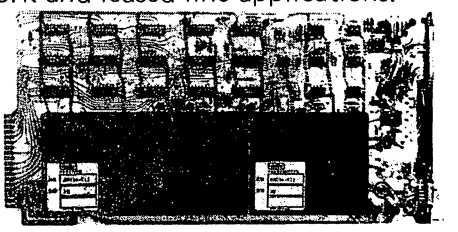
The VA300 family includes modem cards completely compatible with Bell's 103A2, 103E, and 113A/B switched network models. In particular, the VA305C/D card features all the disconnect and control capabilities of the 103E with a bonus—automatic dialing, using Vadic's VA801 type dialer. Applying advanced circuit techniques, Vadic packs full originate/answer operation into the same space taken by answer-only or originate-only models.

### LEASED LINE MODEMS

This series also incorporates 103F compatible versions, right down to programmable local test capability. They can be used in originate-only or answer-only modes.

### SPECIFICATIONS AND MODELS

There are 12 different models, plus strapping options, in the VA300 Series—for all kinds of switched network and leased line applications.



## VA1200: 1200 BAUD BELL 202 COMPATIBLE MODEM

Vadic's VA1200 Series mirrors the VA300 in concept and scope. Built around Vadic's proven line of 1200 baud transmitter and receiver modules, it offers more than Bell in performance, flexibility, and test capabilities.

### SWITCHED NETWORK UNITS

The VA1200 family covers all Bell 202 configurations, with or without 5-baud reverse channel. All models in the MDS series can also be used with Vadic's VA801 type automatic dialer. With its unique modular construction, Vadic units can be configured for transmit-only or receive-only applications.

### LEASED LINE UNITS

Vadic's VA1200 series also satisfies all 202D- or 202R-compatible requirements, but adds enough flexibility to meet virtually all leased line 1200 or 1800 baud applications. Basic units provide field straps for adjusting Clear-To-Send, carrier detect, soft carrier, and turn-around delays. These control features provide versions for two or four wire connection, point-to-point or multipoint system arrangements.

### NO EQUALIZER

One option Vadic does not require is a strappable equalizer. A new detection method obviates a compromise equalizer for typical phase and amplitude distortion. Operation over both short and long haul circuits is excellent—without field adjustment.

## VA2400 SERIES— BELL 201 COMPATIBLE MODEM

Vadic's VA2400 Series modems provide synchronous data transmission at 2400 bits per second. Vadic's modems utilize unique detection techniques that improve performance and enable rapid carrier acquisition over worst case lines.

### SWITCHED NETWORK MODEMS

The VA2400 family is fully compatible with all Bell 201C configurations plus provides automatic dialing using Vadic's VA801 type dialer.

### LEASED LINE MODEMS

The VA2400 Series also is compatible with all Bell 201B modems. Both 2-wire and 4-wire models are available for use in point-to-point or multidrop applications. RTS/CTS delay is strappable for 7.1 msec to optimize throughput.

### MDS DIAGNOSTICS

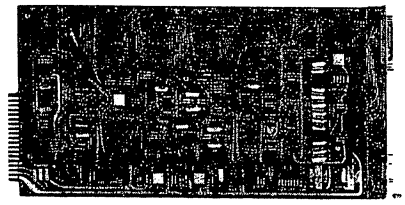
All VA2400 Series modems are completely interchangeable with Vadic's standard packaging and diagnostic hardware. Digital and analog loopback test in-service operation without additional equipment and comprehensive interface display pictures modem and network operation.

### LOW POWER

CMOS circuits are used to minimize power consumption, making the modem ideal for incorporation inside remote terminals or data communications systems.

### SPECIFICATIONS AND MODELS

There are five different models, plus strapping options via pencil switch control, to cover all types of switched network or leased line applications.



## VA2700: IBM COMPATIBLE MODEM

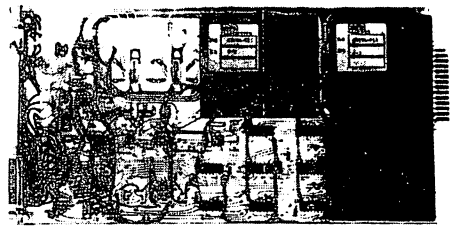
IBM supplies a series of modems, called line adapters, for IBM data communications systems. Vadic offers a complete set of modems compatible with IBM line adapters, but featuring package interchangeability with Bell types and diagnostics unique to Vadic's MDS concept. By adding a free standing cabinet, Vadic can replace and augment IBM 2711 installations.

LDA's—Limited Distance Adapters—for use with privately owned lines of limited length.

SLA's—Shared Line Adapters—up to four pairs of modems can operate independently over one privately owned line, each pair using a different frequency.

LLA's—Leased Line Adapters—IBM's version of modems for lines leased from a common carrier.

These IBM compatible modems can be mixed with all other Vadic modems in any chassis and their diagnostics capabilities are right up with all other MDS modems.



## CCITT COMPATIBLE MODEMS

Vadic also manufactures a complete line of CCITT compatible low and medium speed modems. Contact the factory for information.

## VA801: AUTOMATIC DIALERS

Dialers are numbered VA801A and VA801C, following Bell designations 801A and 801C for pulse and touch-tone automatic calling units, respectively. These units are functional equivalents to Bell dialers, with all strap options available. Computer systems replacing Bell dialers require *NO* software or hardware changes.

### OPERATING FEATURES

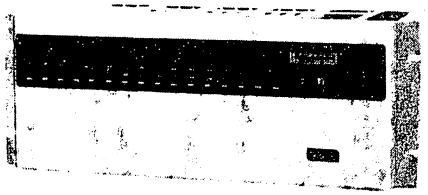
Vadic dialers feature all operations necessary for reliable automatic dialing, plus several functions that optimize system performance:

- Positive dial tone detection
- Positive answer tone detection (2025 Hz or 2225 Hz)
- 7 to 90 second abandon call and retry timer
- End of number code detection
- Forced 1½-second hang-up
- Full compatibility with all Bell 103 and 202 type modems
- Field conversion between pulse and touchtone dialers
- Busy detection for early retry
- CBS or CBT Data Couplers
- Tandem Dialing (optional)
- Invalid digit detection

# Vadic Multiple Data Set Systems

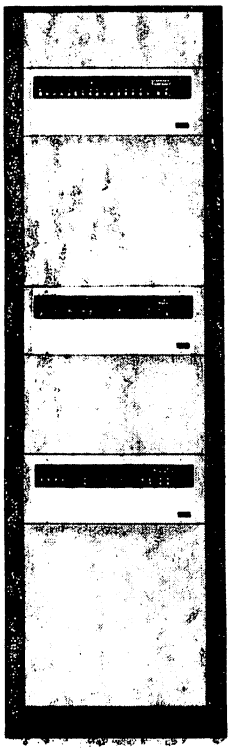
## VA1616: 16 CHANNEL CHASSIS

The VA1616 chassis leads the group. With its capacity to hold up to 16 intermixed Vadic modems, switched or leased line, or automatic dialers in one seven inch high chassis, it is truly a space saver. And of course, redundant power supplies, programmable diagnostics, and selectable front-panel display are part of the package. This means 128 modems—or 48 modems and 48 DAA cards—in a standard Vadic 19-inch wide cabinet!



### DIAGNOSTICS

Sharp diagnostics figure keenly in Vadic's achievement. We knew we could not just cut Bell's prices—we had to offer maintenance techniques that would make you want Vadic hardware for superior system performance. Our integral test and display system actually replaces telephone company services and hardware. Our customers tell us it's nice to know exactly what's wrong ***before*** phoning Bell. Look at the VA1616 chassis. With this elegant and powerful control panel at the central site you can:



- Display all EIA interface lines from a selected modem
- Test any modem locally
- Test any modem from remote station
- Send built-in test signals to remote, or use for local test
- Busy-out any modem
- Watch dynamic channel status of any modem
- Test remote terminals
- Test telephone lines

Vadic adds support hardware to the VA1616 that makes life easier for everyone. Standard prewired cabinets with housings for Bell Data Couplers simplify installation, maintenance, and the floor plan.

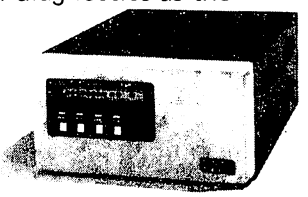
## VA1601 & 2: 1 & 2 CHANNEL CHASSIS

For the remote terminal end, Vadic matches its MDS with a complete family of standalone modems. These units express the same sense of style and performance that sets the MDS apart. Two basic chassis, the VA1601 and VA1602, accommodate any of over 47 different plug-in modem cards—the entire Bell 103, 202, and 201 compatible line, IBM modems, and Vadic's advanced VA3400 Series.

The VA1601 holds one modem and the VA1602 holds two. Now, add the VA1601ACU automatic calling unit chassis and you have more data communications and packaging flexibility than you can find anywhere.

### DIAGNOSTICS AND DISPLAY

Even though they look nice and small, each standalone chassis has the same powerful diagnostics as the VA1616.

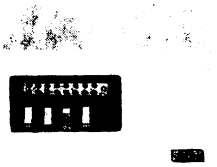


## VA1601ACU: AUTOMATIC CALLING UNIT

Packaging is in the clean and simple style of other Vadic standalone chassis, but inside there are three cards: any one of the MDS modems, a VA801 Series touchtone or pulse type dialer and an audible line monitor card. Enough room is left for a power supply that can even supply a type CBT data coupler, if necessary.

### SPECIAL DIAGNOSTICS

To let you always see what the modem and dialer are doing, the VA1601ACU has separate diagnostic indicators for each. The upper row of eight lights monitors the dialer and the lower monitors the modem interface leads. Of course, these diagnostics are in every way up with the rest of Vadic MDS and standalone units. The audible line monitor card and speaker is an add feature that allows you to hear the dial and connect sequence as it actually occurs. A rear panel switch silences it for normal operation.



vadic
505 EAST MIDDLEFIELD ROAD, MOUNTAIN VIEW, CALIFORNIA 94043 (415) 965-1620
22 EVELYN AVENUE, VINELAND, NEW JERSEY 08360 (609) 696-1440

Adelin M. Clesse

# Hewlett Packard 3352 Gas Chromatograph System to HP 3000 Link

## (Micro Processor Communication Link for Data Collection)

A communication link has been established from an HP 3352C Gas Chromatograph Data System and an HP 3000CX. This link allows scientists and technicians running Gas Chromatographs to automatically transfer their data to a 3000 Disc file.

The purpose of this paper is to describe the hardware/software problems which were encountered and our solutions to those problems. Of particular interest is the fact that a "link controller" based on a microprocessor was designed and constructed to interface the two systems.

## BACKGROUND

In December, 1975, the Scientific Systems Development section of Norwich Pharmacal Company launched a Laboratory Automation project which was to aid the Company in its Drug Research and Development activities. This commitment to computers in the laboratory came as a result of an extensive survey and planning period which closely involved all persons who were potential users of the system.

The overall configuration which was selected to satisfy the diverse needs is shown in Fig. 1 and is composed of three separate Hewlett Packard systems.

The first system, designated as the host, is the HP 3000. The configuration for this system includes: 48K memory, two 7905 discs drives, card reader, line printer and the R.J.E. subsystem. The function of this system is to provide time sharing service to users in the Research Center running a mix of BASIC, FORTRAN, EDITOR, assorted data entry programs, limited Image/Query applications and remote job entry to an IBM 370/145.

The second system is an HP 9600 coupled to the 3000 using the Programmable Controller subsystem. This configuration includes: 21MX with 32K of memory, 32 channel Analog/Digital converter, paper tape reader, and an assortment of digital interface cards. The purpose of this system is to automate 12 analytical instruments (Auto-analyzers, Coulter counters, spectrophotometers, electronic balance, Differential Scanning Calorimeter and CHN Elemental

Analyzer). Programs to support these various instruments are presently being written by SSD personnel.

The third system is the HP 3352 Gas Chromatograph Data System. This "turnkey" system consists of a 21MX with 32K of memory, paper tape reader, five terminals for reports and twelve Analog/Digital modules required to attach as many instruments. The purpose of this system was to automate the time consuming task of data analysis for a specific group of analytical instruments. This group includes gas chromatographs, liquid chromatographs and spectrodensitometers. The software as provided by HP collects data from each instrument and analyses that data (i.e. integrates peaks, performs baseline corrections, and calculates concentrations). The 3352 system then outputs a final report which identifies the peaks and indicates their retention time, area and concentration.

The vendor software does not provide a convenient way to manipulate data beyond the final report phase. This is in part due to the diverse needs of the many users. Since the 3000 could provide this capability, it was decided to perform these additional calculations using that system.

This paper deals primarily with the link which was established from the 3352 to the 3000 to make these additional manipulations possible.

GROUND RULES OR RESTRAINTS

Due to the problems encountered when one attempts to modify existing vendor software, it was decided that we would avoid making any changes to the present system software. On the GC Data System, we would have to use a single user BASIC interpreter to handle the transfer of data out of that system. For the 3000, we would use present existing system software and FORTRAN to handle data entering the system through the asynchronous terminal multiplexer.

Since the 3352 BASIC can process data for only one GC run at a time (other requests are queued up), the transfer of data would have to be as fast as possible to minimize any delay in the start of a new instrument run. One other problem which we encountered was that the single user BASIC would not allow us to perform Input/Output to more than one serial device. It did, however, permit the use of parallel interfaces defined as paper tape reader and punch.

Since the link would be terminated using one of the multiplexer ports on the 3000, it would be necessary that we programatically define the link as a "data device". Once the data had been entered and stored on the 3000, it would have to be conveniently available to all users immediately after the termination of that particular run.

Since the file space on the 3000 is limited, it was decided that the data would remain available for a limited period of time (7 days) after which it would be automatically purged.

The last requirement was that, if a problem develops or maintenance be required on either system, the other should continue to operate albeit without transfer capabilities. For the 3352 this meant that if the 3000 was not available gas chromatographic data analysis could continue to be performed without automatic storage of the generated data.

## SYSTEM DESCRIPTION

Figure 2 shows a block diagram of the link and will be helpful in describing data flow during transfer operations. The link controller shown in the diagram performs a number of functions:

1) Converts data from parallel to serial and serial to parallel format.
2) Buffers an entire record between the two computers.
3) Monitors current status of the 3000.
4) Maintains synchronization between the two systems.

This controller has been implemented with state of the art technology. The heart of the controller is a microprocessor with capabilities similar to many small computers. Amazingly, the cost of this device itself is only about 20 dollars. Other components in the controller increase the total cost somewhat. The microprocessor approach is still cost effective due to the flexibility acquired through its use.

The need for the controller may be more obvious to those who have tried to enter data through a multiplexer port from a peripheral device. Before sending data to a 3000, the device must

ensure that the 3000 is ready to receive data. This requires at a minimum that the device file be opened and that a "READ" operations be pending at that particular moment. The time required to perform these operations is influenced by system loading. When speed is essential, unnecessarily long delays are undesirable. One feature of the multiplexer which we were able to make use of is the automatic transmission of a "DC1" when the 3000 is ready to accept data. This particular sequence is used to trigger 2640 type terminals. We have used it to indicate a 3000 ready-to-receive status.

In an idle state, the controller maintains synchronization with the 3000 by satisfying the pending read every fifteen seconds. The controller then waits for a new "DC1" from the 3000. This idle polling sequence allows the Fortran/3000 program to periodically check a flag file (every sixty seconds). If it finds this flag "set" the program will close opened files, release the multiplexer port and terminate itself. This program normally runs as a batch job with CS priority 23 hours/day.

When a gas chromatograph run is complete, the 3352 system calculates final results and displays these at a terminal near the particular instrument. After the final report has been printed, the single user BASIC is automatically invoked. A short BASIC program then requests the final report data from the 3352 system and passes that data to the controller a line at a time.

As each character is sent to the 3000 by the controller, an echo is received. This echo is then checked with the character

sent. This verification process guarantees that the data stored on the 3000, located six hundred feet away, is identical to the data printed in the final reports.

If the HP 3000 does not respond to an attempted data transfer within fifteen seconds, the controller will time-out and indicate the failure to the 3352. If synchronization is lost with the 3000, any attempt to transfer data will be immediately aborted. For either condition the 3352 prints a message at the local terminal to indicate that the instrument data had not been stored.

When a complete report has been received by the 3000, the data file is locked and the updating is performed. After unlocking the file, a file number and date stamp are sent to the controller to be used by the scientist for subsequent access to his data. This information is conveniently displayed at the bottom of his final report.

A program is executed daily which discards all data stored for more than seven days. If extended storage is required, the data can be extracted from the GC link account and stored in the users own files. This technique allows us to keep the file down to a reasonable size (132,000 bytes).

To extract data from the file all that a user need do is write a short BASIC/3000 program which calls a FORTRAN subroutine. This FORTRAN subroutine opens the file and locks it to assure exclusive access, reads the required information and then passes that data to

the calling BASIC program. Since the subroutine has been stored in the System SL, the scientist is only concerned with his run parameters and remains completely isolated from the required file manipulations.

## TECHNICAL DETAILS OF THE LINK CONTROLLER

The use of a microprocessor for the link controller was not a part of the original design. The original design was based on a parallel to serial integrated circuit (UART) with a few other logic modules for control. It was only after a few unexpected complications came to our attention that we realized that our digital logic circuit was getting very complex and difficult to modify. Although the functions to be implemented by the microprocessor were complicated, our experience in solving computer problems made the solution to our interfacing problems more manageable than if we had used the very complex digital logic.

The use of the microprocessor allowed us to simplify the operations in both minicomputers. The link controller could now easily buffer records, check the echo on serial transmission and indicate to the 3352 if the 3000 was not available. The link controller is completely automatic in that no operator actions are required for daily startup. The link controller operates continuously.

The microprocessor which we chose for this application is the National SC/MP (Simple to use, Cost effective microprocessor). We chose this particular microprocessor for several reasons:

- Cost: An evaluation kit which included RAM, ROM, clock components and interface was available for 99 dollars.
- Software Development: A cross assembler was available on the G.E. Timesharing Network.
- Availability: The above evaluation kit was available from a local distributor off the shelf.

The evaluation card was functional the same day we received it. Using the preprogrammed PROM, we were able to load memory, execute simple programs and list the contents of memory. Our final configuration is shown in Figure 3. It consists of:

- The SC/MP microprocessor
- PROM containing the "monitor program" used to enter programs and start execution.
- RAM 512 bytes containing the programs we wrote to control data flow from one system to the other. 256 bytes are used as a line buffer.
- Input/Output interfaces to match signal levels with those on the 21MX printed circuit assemblies.

Writing programs for a microprocessor is really not much different from writing for a minicomputer, assuming one has experience with an assembler language. The interfacing, however, requires a person who has an in-depth understanding of digital logic. We were fortunate to have acquired expertise in both of these areas prior to this project.

In writing this description of the link controller, I realized that this particular approach could be used to enter data from a variety of instruments directly into a 3000. This would be especially true if only a minimal amount of real time processing of the data were required. For example, if it was necessary to log data from various temperature sensors, the sensors could be easily interfaced to the microprocessor and transferred as a record to the 3000.

As a summary, I would like to itemize the Problems which we encountered and the Solutions to those problems.

3352 GC DATA SYSTEM

P- Single User BASIC did not allow us to specify a particular terminal for input/output.

S- Use parallel interfaces defined as reader and punch.

P- Single User BASIC did not timeout on data entry from reader.

S- Use a "smart" controller which would timeout and send a special code to 3352.

3000 SOFTWARE

P- IOPENTERMDEV locked up terminals under certain situations.

S- Use this routine carefully, close terminal file properly and don't abort the program.

P- "DC1" transmitted by multiplexer.

S- Use this character to our advantage to detect ready status.

P- How do you stop a program running as a job and yet close files and preserve pointers.

S- Use a flag file which is set by a short program.

P- Passing string arrays from FORTRAN to BASIC.

S- Learn to manipulate the logical length of BASIC arrays using an SPL routine.

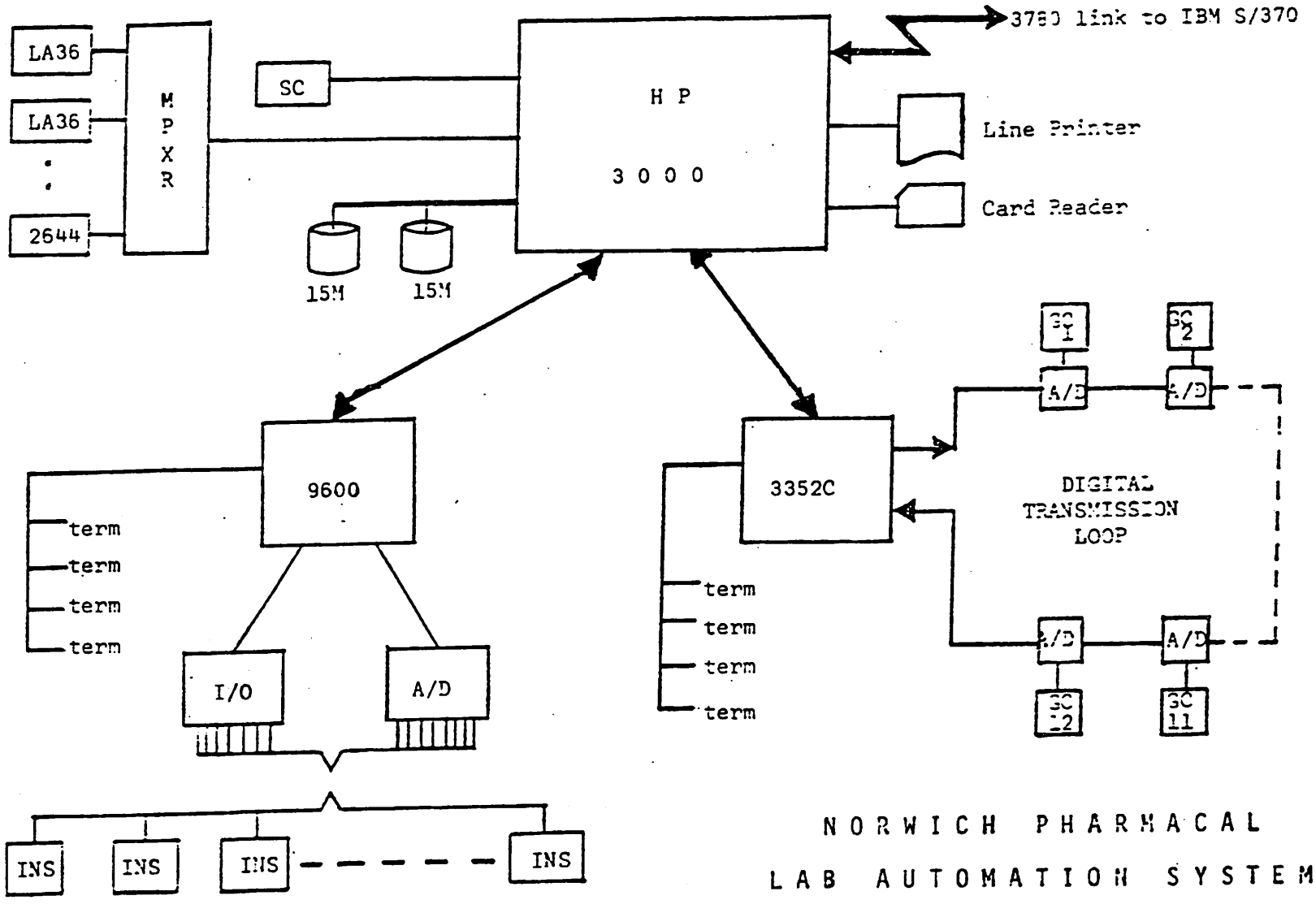
LINK CONTROLLER

P- Checking echo at 2400 Baud was difficult.

S- Analyze the echo generated by the MPXR for timing variation at different BAUD rates. (The echo is offset from the incoming pulse.)

This initial success with microprocessors has encouraged us to re-examine certain applications which we might have otherwise ignored. For analytical instruments it is generally not feasible to dedicate a minicomputer to each instrument. By interfacing these instruments using microcomputers we could perform a certain amount of data reduction before sending the data to the larger computer in serial format. With the cost of micros as low as it is, I feel that a new dimension has been added to distributed processing.

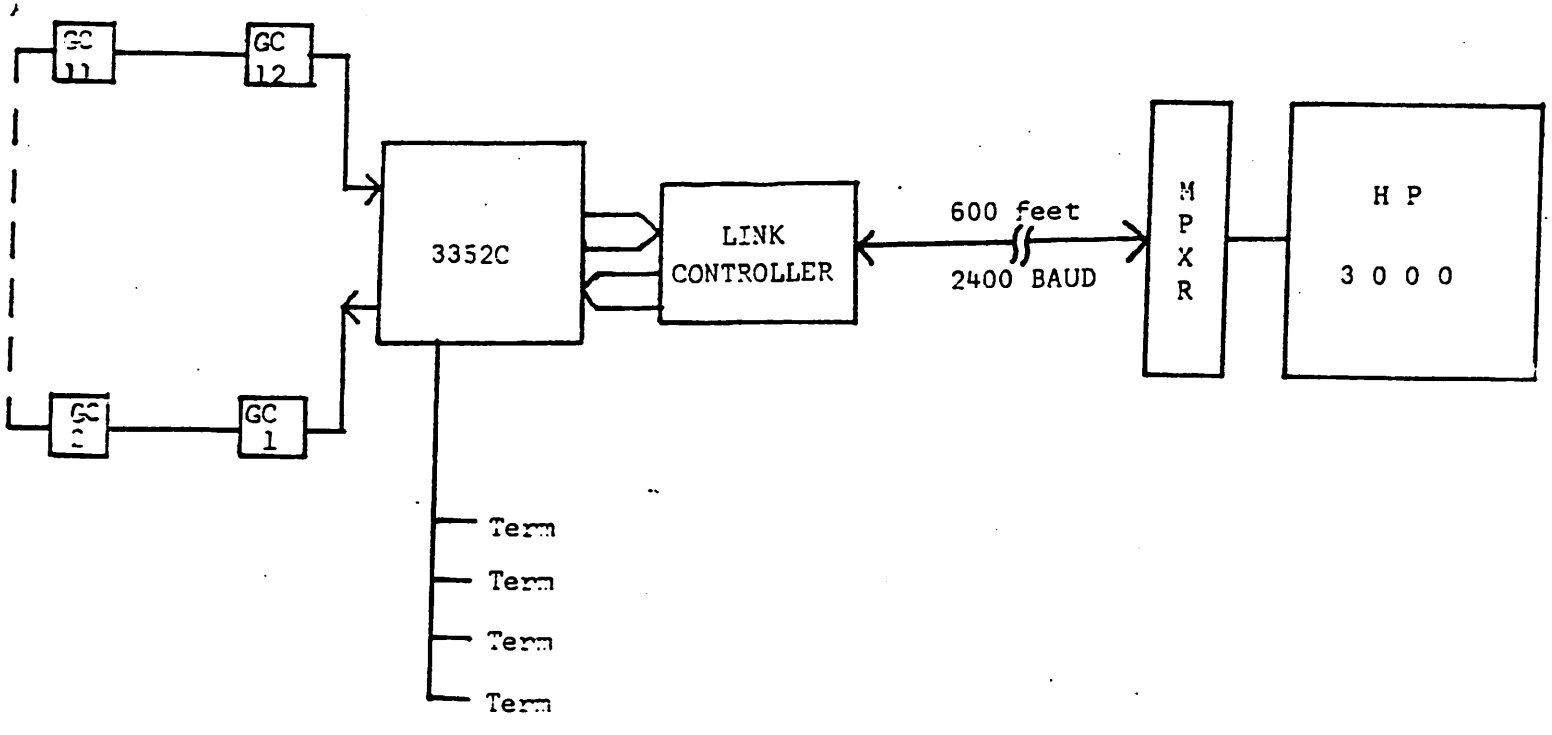
FIG. 1




NORWICH PHARMACAL

LAB AUTOMATION SYSTEM

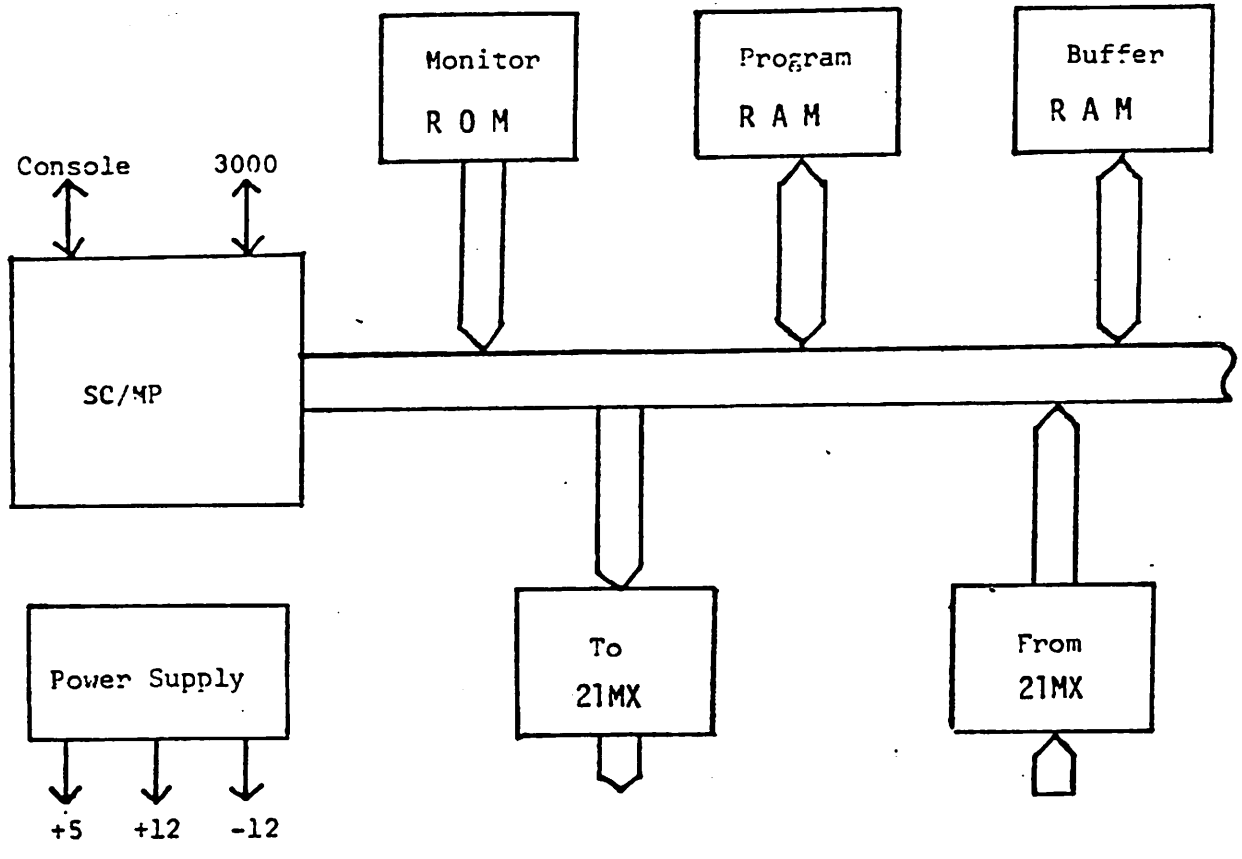
FIG. 2




140

G C LINK HARDWARE

FIG. 3




LINK CONTROLLER

## Introduction

The ALTER procedure is intended to be used as a replacement for the EDIT/3000 MODIFY command. The principal features are an expanded set of flexible editing commands and the capability to perform multiple editing operations from the same command line. The ALTER procedure has been implemented through the PROCEDURE command of EDIT/3000.

## Definitions used in this document

Original line: the text line as passed to the ALTER procedure from EDIT/3000.

ALTER line: the copy of the original line to which editing operations are applied.

Final line: the text line, with alterations, as passed back to EDIT/3000.

Command line: a line of ALTER commands.

HOLD string: a space that can hold a string of characters and is used in a manner similar to the HOLD file of EDIT/3000.

Z string: a space that can hold a sequence of ALTER commands and is used in a manner similar to Z:: of EDIT/3000.

Pointer: a pointer that moves through the ALTER line as the result of the ALTER commands. The action of most commands commences at the current pointer position.

Break character: a character used to separate operands and commands in the command line.

## General operating procedures

When altering a line, as many commands may be used in a single line as desired. A break character (typically a control G) is used to separate the operand of one command from the subsequent command. Alteration of a line is terminated by entering a null command line.

Any numbers associated with a command are optional, if a number is not specified, the default value (typically one) is used.

If a command is incorrectly specified or its use would cause some limit to be exceeded, all subsequent commands in the same command line are ignored and an error message is printed that indicates the point in the command line at which the error was detected. If the user is not accessing the ALTER procedure interactively, all subsequent commands pertaining to the current line are ignored.

When accessing the ALTER procedure in a non-interactive mode (typically from a JOB) all trailing blanks are stripped from the command line. In this mode of operation a break character should be inserted after any desired trailing blanks to ensure their inclusion in the command line.

## Limitations of the ALTER procedure

Maximum line length is 256 characters.

Maximum command line length is 132 characters.

Maximum HOLD string length is 132 characters.

Maximum Z string length is 72 characters.

No number in a command may exceed 256.

## Syntax notation

The elements of the ALTER syntax are described below. Each element is enclosed in "<>", followed by its definition and a list of commands in which the element appears.

<brk-chr> - Any ASCII character; the initial value is control G(BELL). <brk-chr> is used to separate data from subsequent commands.

Commands: INSERT, JOIN, LOCATE, MUNCH, REPLACE, SUBSTITUTE, EXTEND, Z=

<character> - Any ASCII character except <brk-chr>.

Commands: ADD, BREAK, CHANGE, FIND, GRAB, KILL, MUNCH

<string> - A sequence of <character>s to be used for input, comparison, or commands.

Commands: ADD, CHANGE, INSERT, JOIN, MUNCH, REPLACE, SUBSTITUTE, EXTEND, LOCATE, Z=

<c> - An unsigned integer that indicates how many copies of the specified characters are to be manipulated.

Commands: ADD, CHANGE, INSERT, JOIN, MUNCH, REPLACE, SUBSTITUTE, EXTEND, HOLD

<n> - An unsigned integer that indicates the number of characters affected by the command, the number of times to repeat the command, or the particular occurrence of a character or string.

Commands: DELETE, ERASE, HOLD, QUASH, SUBSTITUTE, INVERT, UNDO, Z, SPACE, FIND, GRAB, KILL, LOCATE, MUNCH

<w> - An unsigned integer which represents a field or line width.

Commands: JOIN, OUTLINE, QUASH, WIDTH

<column> - An unsigned integer which represents a column position in the ALTER line.

Commands: OUTLINE, POSITION

# - indicates that input for the command will come from the HOLD string rather than from a <string> following the command, or that characters are to be appended to the HOLD string.

Commands: INSERT, JOIN, LOCATE, MUNCH, REPLACE, SUBSTITUTE, EXTEND, HOLD

* - indicates that the string referenced in the last GRAB, HOLD, or LOCATE command is to be affected by the command, or that the length of the string is to be used as a field width.

Commands: DELETE, ERASE, HOLD, SPACE, SUBSTITUTE, INVERT, JOIN, QUASH

Elements enclosed in "[]" are optional and may be omitted if the default value for the element is desired.

## Example notation

The examples that follow each command are delimited by "!". A "U" at the beginning of the line indicates a user input line, comments follow the right-hand "!". Blank lines are used for purposes of clarity only, and are not generated by the actual ALTER procedure.

For demonstrative purposes, the break character used in the examples is "/".

## Implementation notes

The ALTER procedure is a single SPL procedure named "M" contained in the file ITELEDIT.SOURCE.LIB. The procedure should be compiled and added to an appropriate SL, typically the system SL.

Accessing the procedure from EDIT/3000 is done in the following manner:

```
/P M,{G P S} <rangelist>
```

The first parameter is the procedure name "M", the second parameter (G, P, or S) indicates the sequence of SLs that will be searched for the procedure, and the third parameter is a range list specifying which lines will be affected.

## ADD

Purpose: Add characters to the line.

Syntax:

Form 1 - [ <c> ] A <character>

Description:

Form 1 - Add <c> copies of <character> in front of the current pointer position.

Defaults - <c> defaults to 1.

Pointer - The pointer is positioned after the last character added.

Comments:

ADD is typically used to add a single character into the line.

Examples:

```
U!  /P M,S,1                          !
 !  ALTER     1                       !
 !  THIS IS THE SAMPLE LINE.          !
U!                    AS              ! 1
 !  THIS IS THE SAMPLES LINE.         !
U!  3A$                               ! 2
 !  $$$THIS IS THE SAMPLES LINE.      !
U!                                    ! 3
 !  /                                 !
```

1 - Add an "S" in front of the space between "SAMPLE" and "LINE".

2 - Add three "$"s in front of the first character of the line.

3 - End of alteration.

## BREAK

Purpose: Change the definition of <brk-chr>.

Syntax:

Form 1 - B <character>

Description:

Form 1 - Use <character> as the ALTER <brk-chr>.

Defaults -

Pointer - The pointer is unaffected by this command.

Comments:

<brk-chr> is initially set to a control G.

The break character can be changed to a character that is more convenient to type than control G. It should be set to a character that does not appear in the text since the ALTER procedure can not differentiate between the break character used as text or as a break.

Examples:

```
U!  /P M,S,1                              !
 !  ALTER     1                           !
 !  THIS IS THE SAMPLE LINE.              !
U!  B/                                    ! 1
 !  THIS IS THE SAMPLE LINE.              !
U!  IHERE, /F.I??                         ! 2
 !  HERE, THIS IS THE SAMPLE LINE??.      !
U!                                        ! 3
 !  /                                     !
```

1 - Change the break character to be a "/".

2 - Insert the string "HERE, " in front of the first character of the line, find the next occurrence of "." and insert "??" in front of it. Note that a break character was used to separate the string of the first INSERT command from the FIND command, but that no break character was required after the second INSERT command since there are no subsequent commands on that line.

3 - End of alteration.

## CHANGE

Purpose: Change characters in the line.

Syntax:

Form 1 - [ <c> ] C <character>

Description:

Form 1 - Replace characters of the line starting at the current pointer position with <c> copies of <character>.

Defaults - <c> defaults to 1.

Pointer - The pointer is positioned after the last character changed.

Comments:

CHANGE is typically used to replace a single character in the line.

Examples:

```
U!  /P M,S,1                          !
 !  ALTER     1                       !
 !  THIS IS THE SAMPLE LINE.          !
U!  FACI                              ! 1
 !  THIS IS THE SIMPLE LINE.          !
U!  F.2C??                            ! 2
 !  THIS IS THE SIMPLE LINE??         !
U!                                    ! 3
 !  /                                 !
```

1 - Find the first occurrence of an "A" and change it to an "I".

2 - Find the first occurrence of a "." and change that character and the next to "??".

3 - End of alteration.

## DELETE

Purpose: Delete characters from the line.

Syntax:

Form 1 - [ <n> ] D

Form 2 - * D

Description:

Form 1 - Delete the next <n> characters starting at the current pointer position.

Form 2 - Delete the string recorded by the immediately preceding GRAB, HOLD, or LOCATE command.

Defaults - <n> defaults to 1.

Pointer - The pointer is positioned after the last character deleted.

Comments:

If the original pointer position is less than <n> characters from the end of the line, only the characters through the end of the line are deleted and the pointer is positioned after the last character of the line.

Examples:

```
U!  /P M,S,1                          !
 !  ALTER     1                       !
 !  THIS IS THE SAMPLE LINE.          !
U!  3FS7D                             ! 1
 !  THIS IS THE LINE.                 !
U!  2LIS /*D                          ! 2
 !  THIS THE LINE.                    !
U!                                    ! 3
 !  /                                 !
```

1 - Find the third "S" in the line and delete seven characters starting at that point.

2 - Locate the second occurrence of the string "IS " and delete it.

3 - End of alteration.

## ERASE

**Purpose:** Erase (change to blank) characters in the line.

**Syntax:**

Form 1 - [ <n> ] E

Form 2 - * E

**Description:**

Form 1 - Erase the next <n> characters starting at the current pointer position.

Form 2 - Erase the string recorded in the immediately preceding GRAB, HOLD, or LOCATE command.

Defaults - <n> defaults to 1.

Pointer - The pointer is positioned after the last character erased.

**Comments:**

If the original pointer position is less than <n> characters from the end of the line, only the characters through the end of the line are erased and the pointer is positioned after the last character erased.

**Examples:**

```
U!  /P M,S,1                              !
 !  ALTER     1                           !
 !  THIS IS THE SAMPLE LINE.              !
U!  2FI2E                                 ! 1
 !  THIS    THE SAMPLE LINE.              !
U!  2LTH/*E                               ! 2
 !  THIS      E SAMPLE LINE.              !
U!                                        ! 3
 !  /                                     !
```

1 - Find the second occurrence of "I" in the line and erase that character and the next.

2 - Locate the second occurrence of "TH" in the line and erase it.

3 - End of alteration.

## FIND

Purpose: Position the pointer at a specific character.

Syntax:

Form 1 - [ <n> ] F <character>

Description:

Form 1 - Find the <n>th occurrence of <character> starting from the current pointer position.

Defaults - <n> defaults to 1.

Pointer - The pointer is positioned at the <n>th occurrence of <character>.

Failure - The command fails if the <n>th occurence of <character> is not found.

Comments:

Examples:

```
U!  /P M,S,1                          !
 !  ALTER    1                        !
 !  THIS IS THE SAMPLE LINE.          !
U!  3FECT                             ! 1
 !  THIS IS THE SAMPLE LINT.          !
U!  4FSIXXX                           ! 2
 !     ^                              !
 !  *** ERROR *** COMMAND FAILED      !
 !  THIS IS THE SAMPLE LINT.          !
U!                                    ! 3
 !  /                                 !
```

1 - Find the third "E" in the line and change it to a "T".

2 - Finding the fourth "S" will fail since there are only three "S"s in the line. The insert command is not executed.

3 - End of alteration.

## GRAB

Purpose: Record the position and length of a string.

Syntax:

Form 1 - [ <n> ] G <character>

Description:

Form 1 - Record the position and length of the string from the current pointer position up to, but not including, the <n>th occurrence of <character>. This command does not change the line, but the information it records can be used by subsequent commands to alter the line.

Defaults - <n> defaults to 1.

Pointer - The pointer is unaffected by this command.

Failure - The command fails if the <n>th occurrence of <character> is not found.

Comments:

The GRAB command is used to specifiy a string for use by the DELETE, ERASE, HOLD, SPACE, SUBSTITUTE, or INVERT commands; or to define a field width for the JOIN and QUASH commands.

Examples:

```
U!  /P M,S,1                          !
 !  ALTER     1                       !
 !  THIS IS THE SAMPLE LINE.          !
U!  3FSG *V                           ! 1
 !  THIS IS THE sample LINE.          !
U!                                    ! 2
 !  /                                 !
```

1 - Find the third "S" in the line, grab all characters up to the next " " in the line and down shift the string recorded.

2 - End of alteration.

## HOLD

Purpose: Place characters in a HOLD string from which they may later be retrieved.

Syntax:

Form 1 - [ <n> ] [, <c> ] H

Form 2 - [ <n> ] # [ <c> ] H

Form 3 - * [ <c> ] H

Form 4 - * # [ <c> ] H

Description:

Form 1 - Place <c> copies of the <n> characters which begin at the current pointer position into the HOLD string.

Form 2 - Append <c> copies of the <n> characters which begin at the current pointer position to the contents of the HOLD string.

Form 3 - Place <c> copies of the string recorded by the preceding GRAB, HOLD, or LOCATE command into the HOLD string.

Form 4 - Append <c> copies of the string recorded by the immediately preceding GRAB, HOLD, or LOCATE command to the contents of the HOLD string.

Defaults - <n> and <c> default to 1.

Pointer - The pointer is unaffected by this command.

Comments:

Note the difference between Form 1(3) and Form 2(4): in Form 1(3) the designated string replaces whatever was previously contained in the HOLD string; whereas in Form 2(4) the designated string is appended to the existing contents of the HOLD string.

If the original pointer position is less than <n> characters from the end of the line, only the characters through the end of the line are held.

The contents of the HOLD string are retained between calls to the ALTER procedure.

The length of the HOLD string is limited to 132 characters.

The HOLD string is used in a manner similar to the hold file of EDIT/3000, but the two areas are not related.

Examples:

```
U!  /P M,S,1                          !
 !  ALTER     1                       !
 !  THIS IS THE SAMPLE LINE.          !
U!  2LIS /*H*D                        ! 1
 !  THIS THE SAMPLE LINE.             !
U!  #I                                ! 2
 !  IS THIS THE SAMPLE LINE.          !
U!  L THIS/*#H                        ! 3
 !  IS THIS THE SAMPLE LINE.          !
U!  8DF.#I                            ! 4
 !  THE SAMPLE LINE IS THIS.          !
U!                                    ! 5
 !  /                                 !
```

1 - Locate the second occurrence of "IS " in the line, place it in the HOLD string and then delete it from the line.

2 - Insert the HOLD string in front of the first character of the line.

3 - Locate the first occurrence of " THIS" and append it to the HOLD string.

4 - Delete the first eight characters of the line, find the "." and insert the HOLD string in front of it.

# INSERT

Purpose: Insert characters into the line.

Syntax:

Form 1 - [ <c> ] I <string> [ <brk-chr> ]

Form 2 - # [ <c> ] I

Description:

Form 1 - Insert <c> copies of <string> in front of the current pointer position. <string> must be followed by a <brk-chr> if additional commands follow on the same command line.

Form 2 - Insert <c> copies of the HOLD string in front of the current pointer position.

Defaults - <c> defaults to 1.

Pointer - The pointer is positioned after the last character inserted.

Comments:

INSERT is used to add an indefinite number of characters into the line.

Examples:

```
U!  /P M,S,1                              !
 !  ALTER     1                           !
 !  THIS IS THE SAMPLE LINE.              !
U!  IIS /2FI3DF.I NOW                     ! 1
 !  IS THIS THE SAMPLE LINE NOW.          !
U!                                        ! 2
 !  /                                     !
```

1 - Insert "IS " in front of the first character of the line, find the second occurrence of "I" (starting after the inserted characters), delete three characters, find the "." and insert " NOW". Note the use of the "/" to separate the string of the first INSERT command and the subsequent FIND command.

2 - End of alteration.

# JOIN

Purpose: Insert characters into a field without affecting the portion of the line to the right of the field.

Syntax:

Form 1 - [ <w> ] [, <c> ] J <string> [ <brk-chr> ]

Form 2 - * [ <c> ] J <string> [ <brk-chr> ]

Form 3 - * # [ <c> ] J

Description:

Form 1 - Within a field which begins at the current pointer position and has width <w>, join <c> copies of <string> to the front of the field. <string> must be followed by a <brk-chr> if additional commands follow on the same command line.

Form 2 - Within a field which begins at the current pointer position and has a width equal to the length of the string recorded by the last GRAB, HOLD, or LOCATE command, join <c> copies of <string> to the front of the field. <string> must be followed by a <brk-chr> if additional commands follow on the same command line.

Form 3 - Within a field which begins at the current pointer position and has a width equal to the length of the string recorded by the last GRAB, HOLD, or LOCATE command, join <c> copies of the HOLD string to the front of the field.

Any characters that would be moved past the end of the field because of the insertion will be deleted.

Defaults - <w> and <c> default to 1.

Pointer - The pointer is positioned after the last character inserted.

Comments:

If the width of the field is not known, the GRAB command can be used to establish the field width. Examples:

```
U!  /P M,S,2                              !
 !  ALTER     2                           !
 !  SMITH           ,ANN        ,F,29     !
U!  FAG,*JMARY                            ! 1
 !  SMITH           ,MARYANN    ,F,29     !
U!                                        ! 2
 !  /                                      !
```

1 - Find the first "A" in the line, grab up to the next "," and insert "MARY" at the front of the field recorded without affecting the portion of the line to the right of the ",".

2 - End of alteration.

# KILL

Purpose: Delete characters up to a specific character.

Syntax:

Form 1 - [ <n> ] K <character>

Description:

Form 1 - Delete all characters from the current pointer position up to but not including the <n>th occurrence of <character>.

Defaults - <n> defaults to 1.

Pointer - The pointer is positioned at the <n>th occurrence of <character>.

Failure - The command fails if the <n>th occurrence of <character> is not found.

Comments:

Examples:

```
U!  /P M,S,1                          !
 !  ALTER     1                       !
 !  THIS IS THE SAMPLE LINE.          !
U!  2KI                               ! 1
 !  IS THE SAMPLE LINE.               !
U!                                    ! 2
 !  /                                 !
```

1 - Delete all characters up to but not including the second occurence of "I" in the line.

2 - End of alteration.

## LOCATE

Purpose: Locate a string of characters.

Syntax:

Form 1 - [ <n> ] L <string> [ <brk-chr> ]

Form 2 - [ <n> ] # L

Description:

Form 1 - Locate the <n>th occurrence of <string> starting from the current pointer position. <string> must be followed by a <brk-chr> if additional commands follow on the same command line.

Form 2 - Locate the <n>th occurrence of the HOLD string starting from the current pointer position.

Defaults - <n> defaults to 1.

Pointer - The pointer is positioned at the first character of the <n>th occurrence of <string> or the HOLD string.

Failure - The command fails if the <n>th occurrence of <string> or the HOLD string is not found.

Comments:

Examples:

```
U!  /P M,S,1                          !
 !  ALTER     1                       !
 !  THIS IS THE SAMPLE LINE.          !
U!  LLINE/*E                          ! 1
 !  THIS IS THE SAMPLE     .          !
U!                                    ! 2
 !  /                                 !
```

1 - Locate the first occurrence of "LINE" in the line and erase it.

2 - End of alteration.

# MUNCH

Purpose: Replace characters up to a specific character.

Syntax:

Form 1 - [ <n> ] [, <c> ] M <character> <string> [ <brk-chr> ]

Form 2 - [ <n> ] # [ <c> ] M <character>

Description:

Form 1 - Replace all characters from the current pointer position up to but not including the <n>th occurrence of <character> with <c> copies of <string>. <string> must be followed by a <brk-chr> if additional commands follow on the same command line.

Form 2 - Replace all characters from the current pointer position up to but not including the <n>th occurrence of <character> with <c> copies of the HOLD string.

Defaults - <n> and <c> default to 1.

Pointer - The pointer is positioned after the last character replaced.

Failure - The command fails if the <n>th occurrence of <character> is not found.

Comments:

MUNCH is equivalent to a KILL command followed by an INSERT.

Examples:

```
U!  /P M,S,1                                  !
  !  ALTER     1                              !
  !  THIS IS THE SAMPLE LINE.                 !
U!  3FS2MLHEAD                                ! 1
  !  THIS IS THE HEADLINE.                    !
U!  3,3MHEXTRA                                ! 2
  !  EXTRA EXTRA EXTRA HEADLINE.              ! 3
U!  /                                         !
```

1 - Find the third "S" in the line, replace all characters up to but not including the second occurrence of "L" with the string "HEAD".

2 - Replace all characters up to but not including the third occurrence of "H" in the line with three copies of the string "EXTRA ".

3 - End of alteration.

## OUTLINE

Purpose: Generate a column position template.

Syntax:

Form 1 - [ <w> ] [, <column> ] O

Description:

Form 1 - Generate a column position template <w> columns wide starting in <column>.

Defaults - <w> and <column> default to 1.

Pointer - The pointer is unaffected by this command.

Comments:

The template generated by the OUTLINE command is useful when column dependent editing is being performed.

Examples:

```
U!  /P M,S,1                                  !
 !  ALTER     1                               !
 !  THIS IS THE SAMPLE LINE.                  !
U!  300                                       ! 1
 !          1         2         3             !
 !  123456789012345678901234567890            !
 !  THIS IS THE SAMPLE LINE.                  !
U!                                            ! 2
 !  /                                         !
```

1 - Print a column template thirty columns wide.

2 - End of alteration.

## POSITION

Purpose: Position the pointer at a specified column.

Syntax:

Form 1 - <column> P

Description:

Form 1 - Position the pointer at <column>.

Defaults - <column> defaults to 1.

Pointer - The pointer is positioned at <column>.

Comments:

The leftmost column of the ALTER line is column one.

Examples:

```
U!  /P M,S,1                              !
 !  ALTER     1                           !
 !  THIS IS THE SAMPLE LINE.              !
U!  10PIAB/PCX                            ! 1
 !  XHIS IS TABHE SAMPLE LINE.            !
U!                                        ! 2
 !  /                                     !
```

1 - Position the pointer at column ten, insert the string "AB" in front of that column, return the pointer to column one and change that character to an "X".

2 - End of alteration.

## QUASH

Purpose: Delete characters from a field without affecting the portion of the line to the right of the field.

Syntax:

Form 1 - [ <w> ] [, <n> ] Q

Form 2 - * [ <n> ] Q

Description:

Form 1 - Within a field which begins at the current pointer position and has width <w>, delete the first <n> characters.

Form 2 - Within a field which begins at the current pointer position and has a width equal to the length of the string recorded by the last GRAB, HOLD, or LOCATE command, delete the first <n> characters.

Blanks are inserted at the right end of the field to maintain the original field width.

Defaults - <n> and <w> default to 1.

Pointer - The pointer is positioned after the last character deleted.

Comments:

If the width of the field is not known, the GRAB command can be used to establish the field width.

Examples:

```
U!  /P M,S,2                                  !
 !  ALTER     2                               !
 !  SMITH           ,MARYANN    ,F,29         !
U!  2FMG,*4Q                                  ! 1
 !  SMITH           ,ANN        ,F,29         !
U!                                            ! 2
 !  /                                         !
```

1 - Find the second occurrence of "M" in the line, grab all characters up to but not including the next "," and delete four characters from the field recorded without affecting the portion of the line to the right of the ",".

2 - End of alteration.

## REPLACE

Purpose: Replace characters.

Syntax:

Form 1 - [ <c> ] R <string> [ <brk-chr> ]

Form 2 - # [ <c> ] R

Description:

Form 1 - Replace characters of the line starting at the current pointer position with <c> copies of <string>. The number of characters replaced is equal to <c> times the length of <string>. <string> must be followed by a <brk-chr> if additional commands follow on the same command line.

Form 2 - Replace characters of the line starting at the current pointer position with <c> copies of the HOLD string. The number of characters replaced is equal to <c> times the length of the HOLD string.

Defaults - <c> defaults to 1.

Pointer - The pointer is positioned after the last character replaced.

Comments:

REPLACE is used to change an indefinite number of characters in the line.

Examples:

```
U!  /P M,S,1                              !
 !  ALTER     1                           !
 !  THIS IS THE SAMPLE LINE.              !
U!  2FIRPRETTY                            ! 1
 !  THIS PRETTY SAMPLE LINE.              !
U!                                        ! 2
 !  /                                     !
```

1 - Find the second occurrence of "I" in the line and replace characters from that point on with "PRETTY".

2 - End of alteration.

## SUBSTITUTE

Purpose: Substitute one string for another.

Syntax:

Form 1 - [ <n> ] [, <c> ] S <string> [ <brk-chr> ]

Form 2 - [ <n> ] # [ <c> ] S

Form 3 - * [ <c> ] S <string> [ <brk-chr> ]

Form 4 - * # [ <c> ] S

Description:

Form 1 - Substitute for the next <n> characters starting at the current pointer position <c> copies of <string>. <string> must be followed by a <brk-chr> if additional commands follow on the same command line.

Form 2 - Substitute for the next <n> characters starting at the current pointer position <c> copies of the HOLD string.

Form 3 - Substitute for the string recorded by the last GRAB, HOLD, or LOCATE command <c> copies of string. <string> must be followed by a <brk-chr> if additional commands follow on the same command line.

Form 4 - Substitute for the string recorded by the last GRAB, HOLD, or LOCATE command <c> copies of the HOLD string.

Defaults - <n> and <c> default to 1.

Pointer - The pointer is positioned after the last character inserted.

Comments:

SUBSTITUTE is most often used when replacing a string of characters with <string> which is of different length.

Examples:

```
U!  /P M,S,1                              !
 !  ALTER    1                            !
 !  THIS IS THE SAMPLE LINE.              !
U!  2FT3SA                                ! 1
 !  THIS IS A SAMPLE LINE.                !
U!  2LIS/*SILLUSTRATES                    ! 2
 !  THIS ILLUSTRATES A SAMPLE LINE.       !
U!                                        ! 3
 !  /                                     !
```

1 - Find the second occurrence of "T" in the line and substitute the string "A" for the three characters starting at that point.

2 - Locate the second occurrence of the string "IS" in the line and substitute the string "ILLUSTRATES" for it.

3 - End of alteration.

## TRANSPOSE

Purpose: Transpose two adjacent characters.

Syntax:

Form 1 - T

Description:

Form 1 - Transpose (reverse the relative positions) of the next two characters starting at the current pointer position.

Pointer - The pointer is positioned after the two transposed characters.

Comments:

Examples:

```
U!  /P M,S,1                          !
 !  ALTER    1                        !
 !  THIS IS THE SAMPLE LINE.          !
U!  FNT                               ! 1
 !  THIS IS THE SAMPLE LIEN.          !
U!                                    ! 2
 !  /                                 !
```

1 - Find the first occurrence of "N" in the line and transpose it with the next character in the line.

2 - End of alteration.

Syntax:

Form 1 - [ <n> ] U

Description:

Form 1 - Restore the line to its <n>th previous state. The only values for <n> currently defined are one and two. A value of one causes the ALTER line to be restored to the immediately previous state, a value of two(or any larger value) causes the ALTER line to be restored to the original line.

Defaults - <n> defaults to 1.

Pointer - The pointer is positioned at the beginning of the line.

Comments:

Note that all previous states back to the original are not saved, only the immediately previous state and the original line. Thus, issuing the UNDO command twice in succession is equivalent to performing a '2U' command, the line is restored to its original state.

Examples:

```
U!  /P M,S,1                          !
 !  ALTER     1                       !
 !  THIS IS THE SAMPLE LINE.          !
U!  4V                                ! 1
 !  this IS THE SAMPLE LINE.          !
U!  5D                                ! 2
 !  IS THE SAMPLE LINE.               !
U!  F.S HERE?                         ! 3
 !  IS THE SAMPLE LINE HERE?          !
U!  U                                 ! 4
 !  IS THE SAMPLE LINE.               !
U!  U                                 ! 5
 !  THIS IS THE SAMPLE LINE.          !
U!                                    ! 6
 !  /                                 !
```

1 - Downshift the first four characters of the line.

2 - Delete the first five characters of the line.

3 - Find the first occurrence of a "." and substitute the string " HERE?" for it.

4 - Restore the line to its immediately previous state.

5 - The second UNDO command in succession will restore the line to its original state.

6 - End of alteration.

## INVERT

Purpose: Change the case of alphabetic characters.

Syntax:

Form 1 - [ <n> ] V

Form 2 - * V

Description:

Form 1 - Invert the case of the next <n> characters starting at the current pointer position. Upper case characters are downshifted, lower case characters are upshifted, and numeric and special characters are unaffected.

Form 2 - Invert the case of all characters in the string found in the last GRAB, HOLD, or LOCATE command.

Defaults - <n> defaults to 1.

Pointer - The pointer is positioned after the last character scanned, this is not necessarily the last character actually inverted.

Comments:

If the original pointer position is less than <n> characters from the end of the line, only the characters through the end of the line are inverted and the pointer is positioned after the last character of the line.

Examples:

```
U!  /P M,S,1                                  !
 !  ALTER    1                                !
 !  THIS IS THE SAMPLE LINE.                  !
U!  6V                                        ! 1
 !  this iS THE SAMPLE LINE.                  !
U!  7V                                        ! 2
 !  THIS Is THE SAMPLE LINE.                  !
U!                                            ! 3
 !  /                                         !
```

1 - Invert the case of the first six characters in the line. Note that this is the first six of any characters, not the first six alphabetics.

2 - Invert the case of the first seven characters in the line.

## WIDTH

Purpose: Specify maximum final line length.

Syntax:

Form 1 - [ <w> ] W

Description:

Form 1 - Set the maximum final line length to <w>.

Defaults - <w> defaults to 1.

Pointer - The pointer is unaffected by this command.

Comments:

The initial value for <w> is 72.

After alteration of the line is complete, the final line length is checked against <w>. If the final line length exceeds <w> , the user is given three options:

1) The line may be altered further to bring the line length down to the specified <w>.
2) The line may be truncated at <w>.
3) The line may be left as is (i.e., longer than <w>).

If the user is not accessing the ALTER procedure interactively, option 3 is chosen automatically.

EDIT/3000 does not issue a warning if the user creates a line that extends beyond the right margin.

The value of <w> is retained until a subsequent WIDTH command changes it.

Examples:

```
U!  /P M,S,1                                !
 !  ALTER    1                               !
 !  THIS IS THE SAMPLE LINE.                 !
U!  15W                                      ! 1
 !  THIS IS THE SAMPLE LINE.                 !
U!                                           ! 2
 !  ***WARNING*** LINE LENGTH( 24) > MAX( 15),
 !                ALTER/TRUNCATE/RETURN
U!  *A/T/CR?A                                ! 3
 !  THIS IS THE SAMPLE LINE.                 !
U!  72W                                      ! 4
 !  THIS IS THE SAMPLE LINE.                 !
U!                                           ! 5
 !  /                                        !
```

1 - Set the maximum final line length to fifteen.

2 - End of alteration.

3 - Since the final line length exceeds the maximum specified, the user is given the options of further altering the line, truncating the line at the maximum final line length, or returning the line to EDIT/3000 as is.

4 - Set the maximum final line length to seventy-two.

5 - End of alteration.

## EXTEND

Purpose: Append characters to the end of the line.

Syntax:

Form 1 - [ <c> ] X <string> [ <brk-chr> ]

Form 2 - # [ <c> ] X

Description:

Form 1 - Append <c> copies of <string> to the end of the line. <string> must be followed by a <brk-chr> if additional commands follow on the same command line.

Form 2 - Append <c> copies of the HOLD string to the end of the line.

Defaults - <c> defaults to 1.

Pointer - The pointer is positioned after the last character appended.

Comments:

Examples:

```
U!  /P M,S,1                                  !
 !  ALTER.    1                               !
 !  THIS IS THE SAMPLE LINE.                  !
U!  X NEXT LINE                               ! 1
 !  THIS IS THE SAMPLE LINE. NEXT LINE.       !
U!  U4X /XFILLER                              ! 2
 !  THIS IS THE SAMPLE LINE.    FILLER        !
U!                                            ! 3
 !  /                                         !
```

1 - Extend the line with the string " NEXT LINE".

2 - Restore the line to its previous state and extend the line with four blanks followed by the string "FILLER".

3 - End of alteration.

# Z

Purpose: Save and use ALTER commands with the Z string.

Syntax:

Form 1 - Z= <string> [ <brk-chr> ]

Form 2 - [ <n> ] Z

Description:

Form 1 - Save the sequence of ALTER commands contained in <string> in the Z string. <string> must be followed by a <brk-chr> if additional commands follow on the same command line.

Form 2 - Execute the commands saved in the Z string <n> times.

Defaults - <n> defaults to 1.

Pointer - Form 1 leaves the pointer unaffected. Form 2 has the same effect as <n> copies of <string> in the command line.

Comments:

The length of the Z string is limited to 72 characters.

The contents of the Z string are retained between calls to the ALTER procedure.

Examples:

```
U!  /P M,S,1                          !
 !  ALTER    1                        !
 !  THIS IS THE SAMPLE LINE.          !
U!  Z=FSCX                            ! 1
 !  THIS IS THE SAMPLE LINE.          !
U!  2Z                                ! 2
 !  THIX IX THE SAMPLE LINE.          !
U!                                    ! 3
U!  /P M,S,1                          ! 4
 !  ALTER    1                        !
 !  THIX IX THE SAMPLE LINE.          !
U!  Z                                 ! 5
 !  THIX IX THE XAMPLE LINE.          !
U!                                    ! 6
 !  /                                 !
```

1 - Assign the string "FSCX" to the Z string. This represents the commands "find next S, change it to an X".

2 - Perform the commands contained in the Z string twice.

3 - End of alteration.

4 - Invoke the ALTER procedure again.

5 - Perform the commands contained in the Z string once. Note that the previous definition is retained.

6 - End of alteration.

## SPACE

Purpose: Move the pointer left or right in a line.

Syntax:

Form 1 - [ <n> ] -

Form 2 - [ <n> ] <space bar>

Form 3 - * <space bar>

Description:

Form 1 - Position the pointer <n> spaces to the left of the original pointer position.

Form 2 - Position the pointer <n> spaces to the right of the original pointer position.

Form 3 - Position the pointer immediately to the right of the string recorded by the last GRAB, HOLD or LOCATE command.

Defaults - <n> defaults to 1.

Pointer - The pointer is positioned as described above.

Comments:

In form 1, if positioning the pointer would move it past the beginning of the line, the pointer will be positioned at the first character of the line. In form 2, if positioning the pointer would move it past the end of the line, blank characters are appended to the end line to the point where the pointer is positioned.

Examples:

```
U!  /P M,S,1                          !
 !  ALTER     1                       !
 !  THIS IS THE SAMPLE LINE.          !
U!  F.2-T                             ! 1
 !  THIS IS THE SAMPLE LIEN.          !
U!       CT                           ! 2
 !  THIS IT THE SAMPLE LIEN.          !
U!  LSAM/* 2D                         ! 3
 !  THIS IT THE SAME LIEN.            !
U!                                    ! 4
 !  /                                 !
```

1 - Find the first occurrence of a "." in the line, space left two positions and transpose the two characters starting at that point.

2 - Space right to the second "S" in the line and change it to a "T".

3 - Locate the first occurrence of the string "SAM", space right over it and delete the next two characters.

4 - End of alteration.

## DEBUG

**Purpose:** Call the intrinsic DEBUG.

**Syntax:**

Form 1 - ?

**Description:**

Form 1 - Call the intrinsic DEBUG.

Pointer - The pointer is unaffected by this command.

**Comments:**

This allows poking around in ALTER and EDIT/3000.

## QUOTE

Purpose: Manipulate ASCII control characters in a line.

Syntax:

Form 1 - '

Description:

Form 1 - The special characters flag is toggled each time ' is encountered as a command.

Defaults -

Pointer - The pointer is unaffected by this command.

Comments:

The special characters flag is set to false before each command line is scanned.

When the special characters flag is true, data tokens of the form "'nnn" are converted to a single ASCII character with value nnn. These characters may be manipulated in the same manner as any other characters. When the special characters flag is false, data tokens of the form "'nnn" are treated as separate ASCII characters. "nnn" should be the decimal representation of the desired ASCII character.

Examples:

```
U! /P M,S,1                                  !
 ! ALTER     1                               !
 ! THIS IS THE SAMPLE LINE.                  !
U! 'F.A'10                                   ! 1
 ! THIS IS THE SAMPLE LINE                   !
 !                                           !
U! 'F'10D                                    ! 2
 ! THIS IS THE SAMPLE LINE.                  !
U! A'10                                      ! 3
 ! '10THIS IS THE SAMPLE LINE.               !
U! 3D                                        ! 4
 ! THIS IS THE SAMPLE LINE.                  !
U!                                           ! 5
 ! /                                         !
```

1 - Turn the special characters flag on, find the first occurrence of "." in the line and add a line feed character ('10) in front of it.

2 - Turn the special characters flag on, find the line feed character and delete it.

3 - Since the special characters flag is off, the string "'10" is inserted in front of the first character of the line.

4 - Delete the first three characters from the front of the line.

5 - End of alteration.

Charles W. Jackson

# SHORTHAN

An easy-to-use utility program that expands simple abbreviations into COBOL source code without requiring you to learn a new language.

## HOW IT OPERATES

A COBOL programmer codes simple abbreviations rather than lengthy COBOL code. SHORTHAN then expands these abbreviations into valid COBOL verbs, syntax, and/or data element names. These abbreviations reside on a permanent tape or disk file and are dynamically loaded into an internal core table each time the SHORTHAN program is used. Subsequent table look-ups are performed using an internal binary search algorithm.

You may determine your own abbreviations, use our list of "STANDARD" abbreviations, or a combination of both. Our abbreviations are simple and easy to use. They are usually the first letter of each expanded word. For example: CONFIGURATION SECTION would be coded CS and expanded accordingly.

In some instances a COBOL coder may need one or more abbreviations that are not contained in your list of standard abbreviations. In such a situation, SHORTHAN allows the coder to submit "TEMPORARY" abbreviations. These "TEMPORARY" abbreviations will be added to the standard abbreviations each time they are submitted to SHORTHAN.

SHORTHAN will not alter the original coding indentation, line format, etc. If an abbreviation substitution would cause the source code to extend beyond column 72, SHORTHAN will automatically generate an appropriately coded continuation card-image.

## WHAT IS PROVIDED

The source code is sent directly to you. Complete installation instructions and System Documentation is provided. Since you are a HP3000 user, no source code changes are necessary. Also provided is a sample program which is used to verify the SHORTHAN installation. SHORTHAN also has a one (1) year software warranty/support. Support for subsequent years is provided for $100 per year.

## WHAT IT WILL DO FOR YOUR COMPANY

SHORTHAN can save your company time and money in three major areas:

- PROGRAM DEVELOPMENT

  Development/maintenance programmer time is reduced because less code is required.

- KEY TRANSCRIPTION/VERIFICATION

  Key transcription/verification time correspondingly decreases because the amount of code is reduced.

- DATA CENTER

  COBOL compiler diagnostics and subsequent reruns are reduced. Less is key transcribed and, therefore, fewer errors are introduced into your source code.

Robert A. Goodman

Professional
Computer Services
Incorporated

2321 E. 28th St.
Long Beach, Ca. 90806
213/426-9561

Professional Computer Services (PCS) is a computer service company located in Long Beach, and has been providing quality computer services for 10 years.

PCS has specialized in the business applications of computers, and has developed package programs for Accounts Payable, Payroll, Accounts Receivable and General Ledger/Financial Statements. These package programs are now running on the Hewlett-Packard System 3000, Series II, and can run on any model of of the HP System 3000.

PCS has used the package programs to serve many different companies of varying sizes and types. The packages are extremely flexible and mature.

The attachment describes the package programs in general. For details, further information on custom programming or questions on the attached, please contact Bob Goodman by phone or mail at the above address and phone number.

PACKAGE SOFTWARE

## Accounts Payable

The Accounts Payable package uses a vendor master file which stores the vendor number, name and address, and terms. When an invoice or expense has been authorized to pay, it is input to the system. An Invoice Register is printed. The due date and discount are computed from the invoice date and the vendor terms.

When it is time to print checks, a "pay through" date is provided. The system pays all invoices with computed due dates less than or equal to the "pay through" date. Provision is made to override the computed decision to pay or not pay. All of the invoices which are to be paid for a particular vendor are collected and one check is printed with a remittance advice showing those invoices being paid. A Check Register is printed.

The system provides for the entry of handwritten checks and prints a Check Register and a Cash Disbursement Journal. At the end of an accounting period, a General Ledger Distribution is printed. All entries against each account are listed with a description, and the summary posting for the account is printed. Entries are generated to the cash, accounts payable, and discount accounts and the report zero balances. This General Ledger Distribution becomes input to the General Ledger and serves as a subsidiary ledger. The distribution for more than one month can be accumulated at the same time so that entries for future periods can be made before finishing with the current or past periods.

The following reports can be generated at any time:

- Listing of all open items by due date or vendor
- Aging of open items
- Vendor file list
- Vendor labels

## Payroll

The Payroll package uses an employee master file to store employee number, name, address, pay rates, tax status, deductions, and personnel data for each employee. When a payroll is to be processed, payroll entries are made for each employee to be paid. The following variables are provided on payroll entries:

- Type Pay - Regular hours, sick, vacation, holiday, excludable sick, meals, tips, 1099, miscellaneous gross.
- Rate - Up to five different rates can be stored in the employee master to provide for different rates for different shifts and job classifications.
- Department - Employee hours can be split across multiple departments.
- Overtime - Time and a half or double time.

The system computes gross payroll, taxes, and other deductions. Up to nine deductions per employee are allowed on a fixed, one time or up to a limit basis. A Payroll Register prints a listing of all employees showing the gross-to-net computation. A Deduction Register prints a listing of all employees who have other deductions with each deduction type listed in a separate column. The Paycheck includes the employee address, and the stub shows the elements making up the gross pay, the current pay taxes, the year-to-date taxes, and the other deductions taken.

A monthly Payroll Report is printed which becomes input to the General Ledger. The account number, the breakdown of gross by the tax boundaries, and the tax and other deductions are listed for each department.

The quarterly 941A and DE3 continuation sheets are printed on appropriate forms and the tax calculations are done. The annual W2 forms are printed.

The following optional reports can be selected as required:

- Employee Master List
- Individual Earnings Records
- Workmans Compensation Report
- Bank Reconciliation
- Deduction Reports
- Time Cards

## Accounts Receivable

The Accounts Receivable package uses a customer master file which stores the customer name and address and sales analysis variables. Sales, payments, and adjusting entries are input to the system. An Invoice Register, Cash Receipts Register, and adjustment registers are printed. All entries are posted to an open item file and to sales analysis files which have been selected. Sales analysis is available by customer, product category, salesman and department. An account number is computed from client option parameters and each journal is posted to a General Ledger Distribution.

At a selected frequency, a trial balance is printed showing all unpaid items for each customer. Each entry is aged according to the invoice date and customer terms.

Each customer has a statement frequency and statements are run at the selected frequency for all qualifying customers. Sales Analysis Reports are printed at required intervals. At the end of each accounting period the General Ledger Distribution is printed. Entries are generated against the cash and accounts receivable accounts and the report zero balances. The General Ledger Distribution becomes input to the General Ledger system and serves as a subsidiary ledger.

## General Ledger and Financial Statements

The General Ledger system uses a Balance Forward master file to store account number, descriptions, balances, and report attributes of the accounts. The output of the Payroll, Accounts Payable, and Accounts Receivable systems are inputs. Miscellaneous general entries are input, also.

The system produces journals, General Ledger, and Financial Statements. Some of the features of the system are listed below:

- Up to two financial statements in different formats and up to five supporting schedule groups in different sequences and formats.
- Two, four, or six column financial statements for use in budgetary statements. Annual or monthly budgeting. Whole dollar option.
- Completely optional financial statement captions and total descriptions.
- Four levels of totals on financial statements. Progressive totals on all levels. Tabulate or print the accounts making up any total level.
- Page change and line skip options at any point on financial statements.
- Optionally print percentages or dollars per unit above and below a percentage base accumulation.
- Year-to-date net change financial statements.
- Source and application of funds statements.
- Any chart of accounts up to seven digit account numbers. Print or nonprint on financial statements.
- Detail description on input.
- Automatic year-end closing.
- Direct entry for corrections without running General Ledger.
- Automatic debit and credit of accounts based on percentage of input or account balance.
- Automatic cost of sales entry.
- Audit report of invalid and special accounts.

Professional
Computer Services
Incorporated

2321 E. 28th St.
Long Beach, Ca. 90806
213/426-9561

# ACCOUNTING APPLICATIONS
# PACKAGE PROGRAMS
# PRICE SCHEDULE

| Application Program | Purchase Price |
|---|---|
| General Ledger and Financial Statements | $3500 |
| Payroll | 3000 |
| Accounts Payable | 3000 |
| Accounts Receivable | 3500 |
| Medical Billing | 2500 |
| Inventory | 2500 |

## Notes to Price Schedule

1. Four hours of technical support at our Long Beach facility are provided with each package. Time and expense for travel outside of the Greater Los Angeles area is paid by the buyer.
2. All programs are warranted against program defects for 90 days.
3. Program maintenance contracts are available at 1% of the purchase price per month or on a time and expense basis.
4. Programs may be rented. The monthly rental is 10% of the purchase price plus a $200 installation charge. Fifty percent of all charges for the first 90 days may be applied toward purchase.

Norman Wright

The need for high volume, low cost data entry is leading increasingly to the use of optical mark sense equipment in a wide variety of applications. Frequently the greatest part of the effort of the beginning user goes into assessing the capabilities of the various available OMR media. Yet in most applications, this will represent only the opening pages of OMR implementation. This discussion will concentrate on such factors as the impact of forms design on the accuracy and completeness of OMR data. Consideration will also be given to philosophies of error correction as well as the implications of OMR data entry on overall systems design. The main thrust of the discussion is on the non-mechanical aspects of humanely and accurately "marking sense".

"The Moving Finger writes: and having writ, moves on: nor all thy piety nor wit shall lure it back to cancel half a line, nor all thy tears wash out a word of it." (From The Rubaiyat of Omar Khayyam, Edward Fitzgerald, translator.)

When the poet Omar Khayyam penned these words in 12th Century Persia, he did not rule out the possibility that, while we could not change a line of what is writ, we could at least subject it to intensive study and re-interpretation. And while Omar could not have foreseen the modern corollary, he almost certainly would have understood it: to capture every line--preferably in machine-readable form. This modern need for high volume data entry leads us, not completely by punsmanship, to our modern OMR--the optical mark reader.

Historically the optical mark reader is closely associated with educational applications. One of its first uses was in scoring the large volumes of standardized educational tests so familiar to post-war generations of students. At first mechanical, and later optical mark sensing equipment freed countless teachers and clerks from the tedious, time consuming, and error-prone task of hand scoring the thousands of student examinations that went along with mass assessments of student achievement. The almost immediate acceptance of the mark sense response sheet may be attributed to its reliance upon the simple and familiar tools of paper and

pencil, and to its close analogy to its non-machine-scorable counterpart. The OMR has continued to dominate this field. Tests, surveys, and questionnaires of all kinds today remain the most widespread and familiar of all OMR applications.

Because of its use in education, it is not surprising that educational administrators were one of the first groups to see the value of the technology in gathering demographic data from students. In schools and colleges around the country, OMR data entry was soon being used not only to assess students, but to register them, assign them to classes, and report their final grades as well. This natural succession of events was perhaps accentuated by the fact that OMR did best what educators needed to do--to capture large amounts of data in a very short time at minimal cost. These non-testing uses of the OMR helped form a broader base upon which many more recent applications have been built.

Increasingly optical mark sense technology is finding its way into diverse applications in business, industry, and government. It has been used to capture census, personnel, and survey information closely analogous to its uses in education. It has also been used for many traditional applications in business--entering routine statistical and sampling data, quality control, inventory, and time accounting. Recently the OMR has been used to replace hand-counted paper ballots in a voting system where the legal requirements of ballot layout prohibited conventional voting machines.

OMR media today are available in a wide variety of styles and sizes. Documents range from punched card size, to standard $8\frac{1}{2}$" by 11" sheets, to documents which fold out into a long continuous form. Many of the readers for card-sized documents accommodate punched as well as marked information.

What all of the available media have in common is requiring the respondant to record his data by gridding--filling in pre-printed boxes, ovals, or circles with pencil marks. The gridded marks are read by the photo sensors on the mark reader and translated by hardware and software into dicital data. The sample sheet

(fig. 1), used by the American College Testing Program to register students for its examinations, shows some of the variety of information that can be captured in this manner. This is in addition, of course, to the more usual multiple choice type of responses.

The example is notable for the relatively high amount of written and numerical information which the applicant is requested to grid. Unfortunately, no studies have been done, as they have been for CRT's, testing the psychological attitude and tolerance of respondants to this form of data entry. Unquestionably the gridding of the information takes longer than printing it--about 3 minutes for gridding the sample document, versus about 1½ minutes for simply entering the data in printed form. In some sense at least, the respondant is required to function as a data transcriber as well as to provide the required information. Undeniably this form of data entry depends upon the time, effort, and care of the respondant.

Largely for this reason it was once felt that filling out OMR documents should take place only in carefully supervised environments. While this is still frequently practiced in schools and colleges, many OMR users have recently had good results with documents filled out by the general public in uncontrolled situations. The sample document in figure 2 is designed for home use. Users of data sheets and surveys find that they can usually rely upon the respondant's good will and his desire for personal data to be correctly entered. There is also some evidence to suggest that the present population consists increasingly of "trained" data transcribers, since OMR is becoming widespread.

In applications requiring use by the untrained respondant or the general public, the layout and design of the printed form is crucial to the success of any OMR project. Essential to successful layout is some ordered and clear method of presenting data for response. With sheet sized documents, the user can frequently include explanations and examples on the OMR document itself. If the information is not to be filled out in a controlled environment, it is almost always desirable to include accompanying instructions clearly specifying the desired content of each

field. The material should also give the traditional "Grid only one response" instructions for marking OMR format: however, too frequently this is done to the detriment of elaboration on field content.

Almost all OMR projects require that related documents be grouped into some form or other, whether by classes and schools in educational applications, or by production units in a business environment. The older method for entering such control information was to depend on the respondant to enter it correctly on the response document itself. For example, students would be asked to grid a school number and the first 5 letters of their teacher's name. Increasingly, however, OMR users prefer to enter such control information on separate sheets or documents, usually ones which are identifiable to the scanning software or hardware itself. Because of its pivotal importance control data is thus separated from respondant data. Frequently it may be entered by a more highly trained or carefully controlled person--a teacher or a clerk at the processing center.

At the U.S. Civil Service Commission's OMR processing center, for example, written exams are shipped to the center from 65 area offices around the country. Personnel at the center fill out a scannable document known as an Area Office Header to identify each group of answer sheets received. The header sheet is color coded and corner cut, borrowing techniques from card processing. This header itself becomes a record on tape along with job applicant documents. It identifies all subsequent records on tape as being input from a particular area office, as well as giving processing and control information about the documents.

In the simplest of all imaginable mark sense applications incorrectly gridded data is of two sorts: either an applicant grids more than one response on a set of grid positions requiring only one--known as a set of grid positions requiring only one--known as a double grid; or the respondant fails to grid any response--an omit. In some applications, the software, either at the point of scanning or in subsequent processing, can be programmed to predictably circumvent or default when

encountering such data. More usually the data suspected to be in error will be examined to see if some determination can be made of the intent of the respondant. Double grids and omits give greatest problem in demographic data; fortunately it is also here that they are most likely to be correctable. Almost universally, OMR documents requiring gridded data such as name and address request that the respondant both print and grid his information. In such cases most errors can be resolved by resorting to the written information.

Some users of OMR depend on extensive pre-screening of the mark sense documents in an effort to detect gridding errors before they are input to the machine. Such an examination can never be totally effective, of course. More importantly, it tends to use a great deal of human time and energy in perusing documents which will need no attention. For this reason, it is usually much more cost effective to concentrate on errors after they have been detected by computer.

In most applications, then, the raw data from the scanner will pass through some kind of edit program designed to detect errors peculiar to OMR entry. Usually other edits and checks on data validity, similar to those for other methods of data entry, will also be performed. The resolution and correction of the detected errors will usually represent by far the greatest amount of time and effort spent in any system involving OMR entry.

The correction process can be considerably more complex using OMR technologies where only one side of a document can be read at a time, or where a single respondant's data spans several documents or cards. In such applications where the order and relatedness of documents can usually not be depended on, a match is done to bring together the different pages or documents which form data on a single individual. Many schemes for pre-gridding or pre-punching match information have been devised. Unless some provision is made for strict control on the match fields, resolution of mis-matching pages can be a monumental task.

The actual method by which corrections are applied to files of OMR data presents an interesting paradox. Since the data is already in machine readable form on the

mark sense document, and since it can usually be corrected by simply erasing or re-gridding the mis-gridded information, it would seem natural to apply the corrections simply by re-scanning the corrected document. While some OMR users do opt for this method, most high volume users do not. They object that it disrupts the production flow of batches and complicates the process of undating data with corrected information. They argue that, while OMR is good at the task of rapid mass data entry, keyed entry is more suitable to correcting the data once it is at a data center.

In an average, well designed OMR application, the documents containing errors or suspected errors will represent somewhere between 5 and 20 percent of the total number of documents entered. The exact figures will, of course, depend on many factors including the amount of data on each document, the age and experience of the respondants, and the complexity of the edit criteria themselves. Typically, the number of actual corrections to be applied to the data will be sonderably less than the number of documents examined for suspected error. Most systems utilizing OMR entry today provide for field corrections on erroneous data rather than requiring total record replacement. This design factor brings about important economies in the data entry phases of correction. The percentage of fields in error, of course, is drastically less than the percentage of documents in error.

Recent advances in CRT technology, and the cost reductions which have accompanied them, open broad new avenues for error detection and correction in OMR data. A few of the newest systems have been designed utilizing some form of CRT correction entry. One can envision the entire process of error detection and correction taking place at the CRT. The stack of OMR documents would provide the only source material needed for the correction process. In such a set up, a single analyst would be responsible for both the resolution of the error and the actual correction of the data on file.

Regardless of their differences in application and technique OMR users point to many of the same advantages. Data is captured in machine readable form directly

from the respondant, thus eliminating the time consuming task of keyed entry. The OMR document is highly portable and requires no special equipment to fill out. While data gathered in this manner will almost always contain errors, a well-designed system can detect and recover from such errors effectively. OMR's are highly efficient at entering large masses of data at low cost. Thus in many applications involving information from and about people, optical mark readers will continue to represent one of the best means available for data aquisition.

## INTEREST INVENTORY

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 1 | 13 | 25 | 37 | 49 | 61 | 73 | 85 |
| 2 | 14 | 26 | 38 | 50 | 62 | 74 | 86 |
| 3 | 15 | 27 | 39 | 51 | 63 | 75 | 87 |
| 4 | 16 | 28 | 40 | 52 | 64 | 76 | 88 |
| 5 | 17 | 29 | 41 | 53 | 65 | 77 | 89 |
| 6 | 18 | 30 | 42 | 54 | 66 | 78 | 90 |
| 7 | 19 | 31 | 43 | 55 | 67 | 79 | |
| 8 | 20 | 32 | 44 | 56 | 68 | 80 | |
| 9 | 21 | 33 | 45 | 57 | 69 | 81 | |
| 10 | 22 | 34 | 46 | 58 | 70 | 82 | |
| 11 | 23 | 35 | 47 | 59 | 71 | 83 | |
| 12 | 24 | 36 | 48 | 60 | 72 | 84 | |

## STUDENT PROFILE SECTION

TURN ANSWER FOLDER WITH LEFT MARGIN OF THIS PAGE UP TO MARK RESPONSES TO ITEMS 1-192 IN THIS SECTION

MRC FORM NO. 5301

PAGE 1

# U.S. Civil Service Commission

FORM APPROVED
OMB NO. 50-R0437

FORM **B**

## MID-LEVEL DATA SHEET

**GENERAL INSTRUCTIONS:** The questions on this form request information about your background, interests, and availability. Before you make any entries, make sure you have read all instructions--on this form, accompanying Form 1056A, and any other instructional material provided by the area office with this form. Recording information that does not accurately reflect your availability, employment preferences, and qualifications may cause you to lose consideration for some positions.

**This form can only be processed if you -- (1.) Use a number 2 (or softer) lead pencil. (2.) Completely blacken the oval corresponding to your response choice. (3.) Completely erase any mistakes. (4.) Make no stray marks.**

**NOTE: BE SURE TO WRITE IN YOUR RESPONSE AS WELL AS BLACKENING THE OVALS IN THE BLOCKS WHERE THIS IS REQUIRED.**

**1**

NAME: _____
(Please Print)

DATE: _____
(Please Print)

**2 SOCIAL SECURITY NUMBER**
Record your SSN in the spaces provided. Blacken the appropriate ovals below.

| | | | – | | – | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | | 1 | 1 | 1 | 1 | 1 | 1 |
| 2 | 2 | 2 | | 2 | 2 | 2 | 2 | 2 | 2 |
| 3 | 3 | 3 | | 3 | 3 | 3 | 3 | 3 | 3 |
| 4 | 4 | 4 | | 4 | 4 | 4 | 4 | 4 | 4 |
| 5 | 5 | 5 | | 5 | 5 | 5 | 5 | 5 | 5 |
| 6 | 6 | 6 | | 6 | 6 | 6 | 6 | 6 | 6 |
| 7 | 7 | 7 | | 7 | 7 | 7 | 7 | 7 | 7 |
| 8 | 8 | 8 | | 8 | 8 | 8 | 8 | 8 | 8 |
| 9 | 9 | 9 | | 9 | 9 | 9 | 9 | 9 | 9 |

**3 GENDER**
Blacken the appropriate oval.

O Male O Female

**4 GRADE LEVEL (S)**
Refer to the enclosed qualifications guidelines. Blacken the oval(s) beside the grade level(s) for which you are available and believe you are qualified.

O GS-9 O GS-11 O GS-12

**5 TRAVEL**
Are you willing to accept a position that requires travel away from the city of your normal work location at least 25 percent of the time?

O Yes O No

**6 TEMPORARY EMPLOYMENT**
Are you willing to accept temporary employment?

O Yes O No

**7 VETERANS PREFERENCE**
Refer to the Form 1056B instructions. Blacken the oval that corresponds to your preference claim.

O Disability 10-pt (CP) O 5-pt (TP)
O Other 10-pt (XP) O None (NV)

**8 PART-TIME EMPLOYMENT**
Are you willing to accept part-time employment (fewer than 40 hrs. per week)?

O Yes O No

**9 EDUCATION LEVEL**
Blacken the oval beside the highest education level you have attained, or will attain in the next 9 months. **Only one oval may be blackened.**

O Ph.D
O Masters
O LL.B. or J.D. Degree
O BA Degree
O Associate Degree
O Some College
O High School or Equivalency
O Less than High School

**10 LANGUAGES**
Blacken the oval beside each language *(other than English)* which you read and speak fluently.

O French O Spanish
O Russian O Chinese
O German O Other

**11 FEDERAL EMPLOYEE**
Are you now a Federal employee?

O Yes O No

**12 STATE/LOCAL AVAILABILITY**
Are you interested in being considered for State or local Government employment?

O Yes O No

FIGURE 2
COURTESY OF THE U.S. CIVIL SERVICE COMMISSION

192

CSC FORM 1056B MAY 1976

## 13 STATE OF LEGAL or VOTING RESIDENCE

Blacken the oval beside the State of your legal or voting residence. **Only one oval may be blackened.**

- O Alabama
- O Alaska
- O Arizona
- O Arkansas
- O California
- O Colorado
- O Connecticut
- O Delaware
- O District of Columbia
- O Florida
- O Georgia
- O Guam
- O Hawaii
- O Idaho
- O Illinois
- O Indiana
- O Iowa
- O Kansas
- O Kentucky
- O Louisiana
- O Maine
- O Maryland
- O Massachusetts
- O Michigan
- O Minnesota
- O Mississippi
- O Missouri
- O Montana
- O Nebraska
- O Nevada
- O New Hampshire
- O New Jersey
- O New Mexico
- O New York
- O North Carolina
- O North Dakota
- O Ohio
- O Oklahoma
- O Oregon
- O Pennsylvania
- O Puerto Rico
- O Rhode Island
- O South Carolina
- O South Dakota
- O Tennessee
- O Texas
- O Utah
- O Vermont
- O Virginia
- O Virgin Islands
- O Washington
- O West Virginia
- O Wisconsin
- O Wyoming

## 14 PREFERENCE

The purpose of this item is to ask you to identify the state or area where you most prefer to work. **Please refer to the local area office's mid-level information sheet.** State of preference will only be used for well qualified candidates for that location. Place the code you select in the spaces provided. Blacken the appropriate ovals.

| | |
|---|---|
| | |
| (A) | (A) |
| (B) | (B) |
| (C) | (C) |
| (D) | (D) |
| (E) | (E) |
| (F) | (F) |
| (G) | (G) |
| (H) | (H) |
| (I) | (I) |
| (J) | (J) |
| (K) | (K) |
| (L) | (L) |
| (M) | (M) |
| (N) | (N) |
| (O) | (O) |
| (P) | (P) |
| (Q) | (Q) |
| (R) | (R) |
| (S) | (S) |
| (T) | (T) |
| (U) | (U) |
| (V) | (V) |
| (W) | (W) |
| (X) | (X) |
| (Y) | (Y) |
| (Z) | (Z) |

## 15 SPECIAL SKILLS and KNOWLEDGES

Listed below are nine special skills and knowledges related to positions at the level for which you are applying. **You may select up to four** of these skills and knowledges for which you have applicable experience and/or training. *Do not select an item if you would not be interested in a job which requires that special skill or knowledge.* Blacken the oval beside each choice.

- O **Program planning or development** -- **systematically applying broad program goals to develop new or improved programs.**
- O **Program evaluation** -- evaluation of ongoing programs to improve their effectiveness. Identifying shortcomings and recommending alternative approaches.
- O **Program management** -- directing and controlling programs. Deciding content, objectives and priorities, and allocating organizational resources.
- O **Supervising** -- planning and directing the work of other employees.
- O **Preparation of Written Reports, Program Proposals.**
- O **Adapting Operations and Procedures to ADP Processes.**
- O **Teaching/Training** -- serving as an instructor at training sessions. Involves preparation of course material.
- O **Grants Review** -- reviewing grant applications and determining grant awards.
- O **Knowledge of Engineering and Scientific Fundamentals.**

## 16 OCCUPATIONAL CODES

**Please refer to the local area office's mid-level information sheet.** Record the codes you choose in the spaces provided. Blacken the appropriate ovals below.

| | | |
|---|---|---|
| | | |
| 0 0 0 | 0 0 0 | 0 0 0 |
| 1 1 1 | 1 1 1 | 1 1 1 |
| 2 2 2 | 2 2 2 | 2 2 2 |
| 3 3 3 | 3 3 3 | 3 3 3 |
| 4 4 4 | 4 4 4 | 4 4 4 |
| 5 5 5 | 5 5 5 | 5 5 5 |
| 6 6 6 | 6 6 6 | 6 6 6 |
| 7 7 7 | 7 7 7 | 7 7 7 |
| 8 8 8 | 8 8 8 | 8 8 8 |
| 9 9 9 | 9 9 9 | 9 9 9 |

## 17 COLLEGE MAJOR

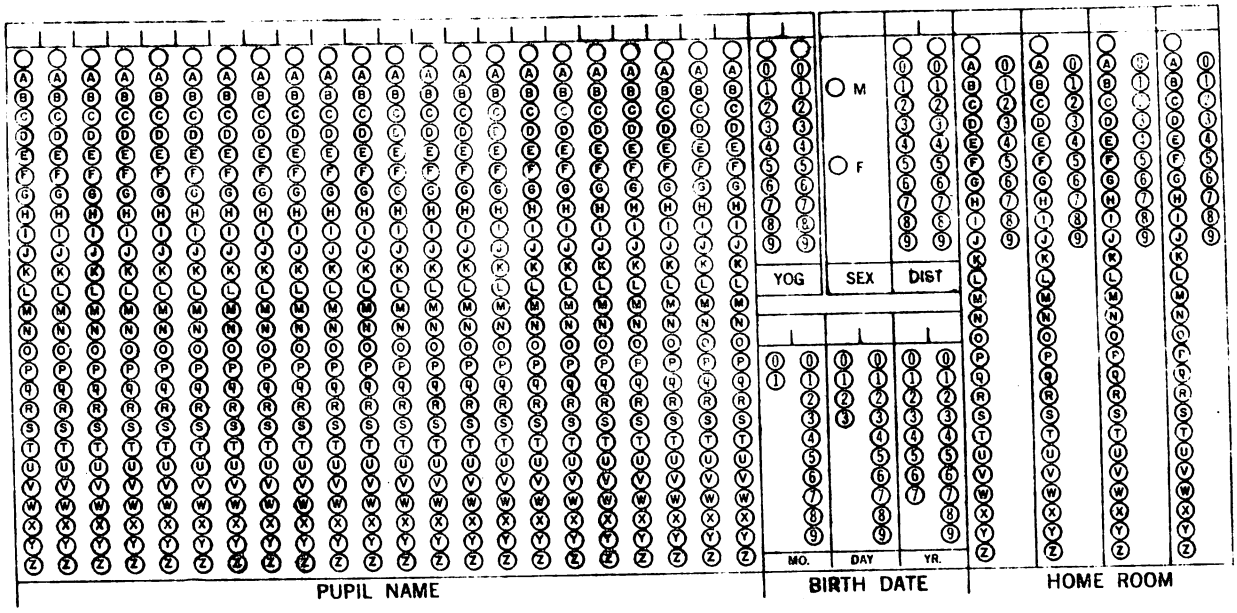
If you have a college degree (or expect to receive one within the next 9 months), blacken the oval corresponding to your major field of study. If you are qualified on the basis of education alone, you must have at least 2 years of graduate study or a Master's Degree. **Only one oval may be blackened.**

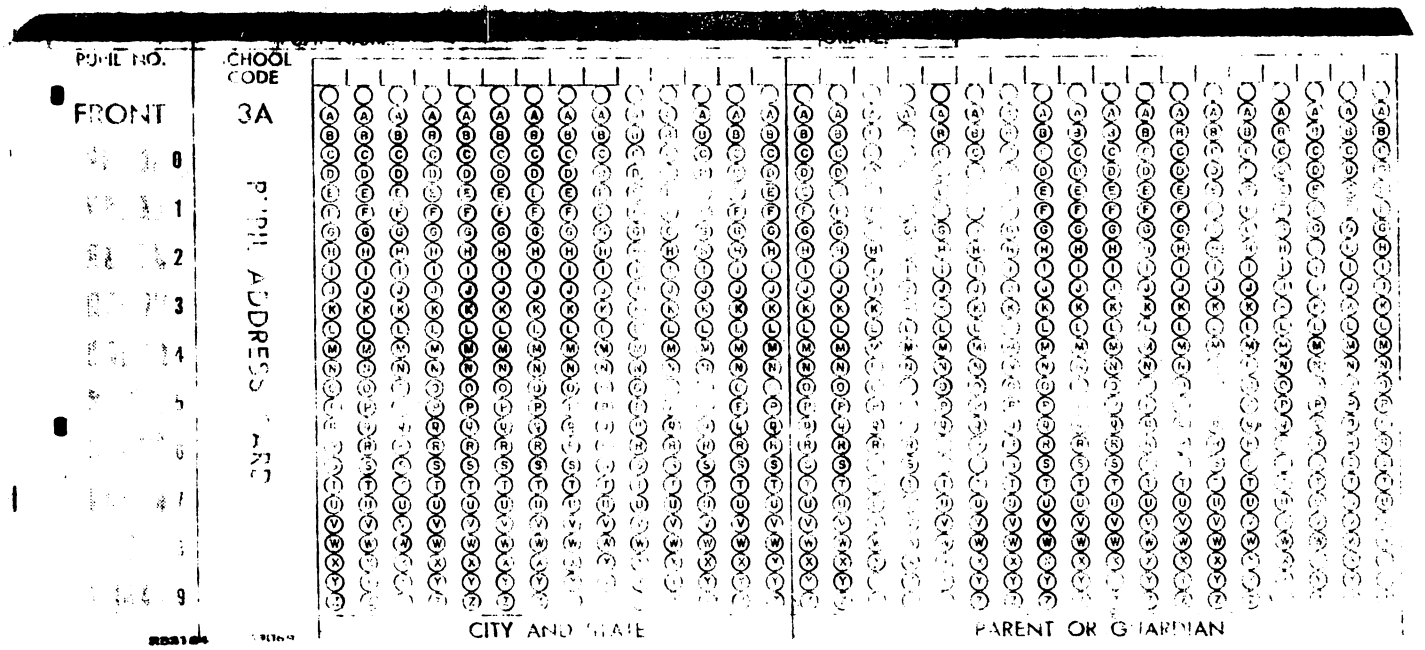
- O Accounting
- O Anthropology
- O Agriculture, Fish and Wildlife Management, Park Management, Forestry
- O Archeology
- O Biological Sciences
- O Business Administration
- O Communications
- O Computer Science
- O Criminology
- O Economics
- O Education
- O English, Journalism
- O Engineering (except Industrial)
- O Fine or Applied Arts
- O Foreign Languages
- O Geography
- O History
- O Industrial Engineering
- O Law
- O Library Science
- O Mathematics or Statistics
- O Personnel or Industrial Relations
- O Pharmacy
- O Physical Science
- O Political Science, Public Administration
- O Psychology
- O Public Health
- O Social Work or Sociology
- O Social Sciences (other than those listed)
- O Transportation
- O Urban Planning
- O Other

## 18 GEOGRAPHIC AVAILABILITY

Refer to **Form 1056B instructions.** Blacken the ovals beside your choices.

| | | | | |
|---|---|---|---|---|
| O 01 | O 11 | O 21 | O 31 | O 41 |
| O 02 | O 12 | O 22 | O 32 | O 42 |
| O 03 | O 13 | O 23 | O 33 | O 43 |
| O 04 | O 14 | O 24 | O 34 | O 44 |
| O 05 | O 15 | O 25 | O 35 | O 45 |
| O 06 | O 16 | O 26 | O 36 | O 46 |
| O 07 | O 17 | O 27 | O 37 | O 47 |
| O 08 | O 18 | O 28 | O 38 | O 48 |
| O 09 | O 19 | O 29 | O 39 | |
| O 10 | O 20 | O 30 | O 40 | |



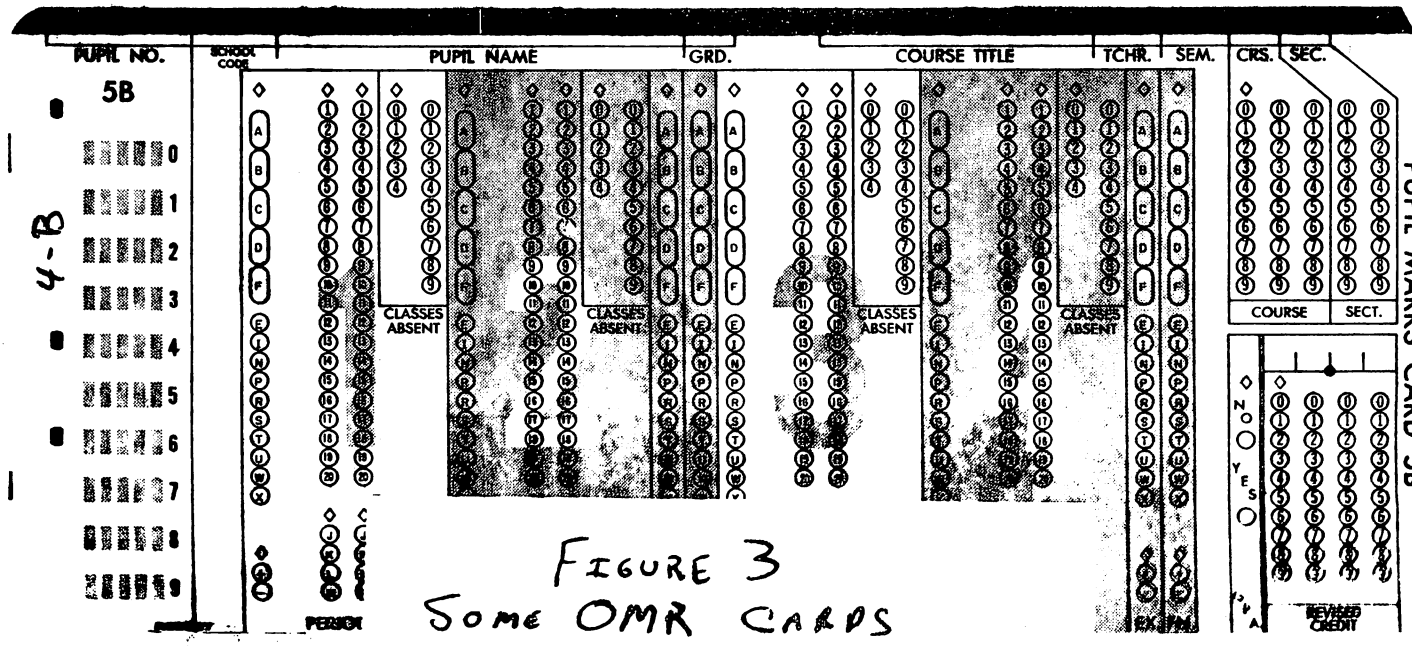




FIGURE 3
SOME OMR CARDS

Charles Van Ausdall

Abstract

It is not our intent to define nor exhaust the possible approaches to transaction logging but to point out the importance of and need for such a facility, specifically, in an on-line environment utilizing the HP 3000 Data Base subsystem IMAGE.

I. What is transaction lossing?

II. Why lose transactions?
(batch/on-line implications)
A. Rerun/recovery
B. Audit capability
C. System evaluation, whoenteredhowmanyofwhatwhen
1. Who - operator/process ID
2. Enteredhowmany - provided by txn. count
3. Ofwhat - transaction ID
4. When - time and date stamp

III. What should we lose?
(as a function of why we lose)
A. All transactions
B. Only actual modifications to the "data base"

IV. Who loss the "what" we choose to lose?
(optimization of utility)
A. Users responsibility?
B. Vendors "opportunity"?

V. How do we lose the "what"?
(the medium chosen as a function of resources/hardware)
A. Tape or disc as "system" resources
B. Unsupported (by vendor) devices configured as terminals

VI. Summary/Proposal
A. Reliability of interested systems
B. Efficiency
C. Operational considerations

## Transaction Logging for On-Line Systems
## Integrity of Image Data Bases

Welcome to the wonderful world of on-line computing!!! I am sure that by now we all believe that on-line processing is possibly the greatest thing since peanut butter. However, as with all "good things" we sometimes find "nits" which only come to light after making the commitment to innovate. One particularly important such item is the management of system integrity for such an amorphous

entity as an on-line computing facility.

For our purposes let us define and limit "on-line" systems to be those facilities providing an operator, typically the end user, with the capability of dealing directly with the computer. Further, let us provide this capability in an interactive mode, allowing the user to enter data and providing on the spot results. This process is to be differentiated from conventional batch systems which accept data in bulk quantities prepared in advance in some computer compatable nonconsumable medium. This data is typically gathered by the user, "punched" by a data entry operator via such devices as keypunch, key-tape, key-disc, etc... and finally submitted by a computer operator to the "machine" for processing. The results are then routed back to the user to be verified and possibly resubmitted to Data Entry for another run.

There are many variations and combinations of the batch/on-line mix and as many valid arguments for the use of each. Our primary interest here is in the implications of the user dealing directly and interactively with the computer.

## Data Integrity

Since the user enters data directly to the computer there is typically no "computer compatable nonconsumable medium" produced. This would be the case with video type terminals. This point is significant in that it implies the possible loss of the capability to rerun the process. If we are operating in an environment in which data is entered from a source document, it would be necessary to manually reenter all data processed since the last backup. In a non-source environment, however, i.e., phone order processing recovery may not be possible.

The process of lossing each transaction as it occurs would provide us with the required history of values and sequence in which data was processed against the data base. This "lose" together with periodic system backup procedures should provide us with the information to restore our "environment" to its status at any given point in time.

A second important consideration to the question of data integrity is the implication of the multi programming environment in which several processes may be concurrently effecting the data environment. This is particularly important if the sequence in which transactions are processed against the data base effect their outcome. In this situation, the result of any one transaction may depend on either the value of elements already in the data base or the possibility that another transaction against the data base may be pending.

The above considerations may be placed in perspective by the following question: "How do we recover our data base following a system failure? (hardware or software)". We are not so much concerned with who or what caused the failure but without ability to pick up the pieces after the cause has been remedied.

## Providing Integrity

We have attempted to handle this situation in our shop in a straight forward manner. Simply stated: ...We do not allow system crashes... This policy, however, has not proven to be entirely successful and we, therefore, have been forced to consider alternatives.

Periodic backup provides a partial answer allowing us to recover up to some point in time T(n) at which the system was last backed up. (We have even implemented an S.O.P. - Standard Operating Procedure - to provide the backups). However, we have not been successful in setting the "system" to cooperate in scheduling its crashes. (What about the transactions from T(n) to the time of crash?)

## Logging

Our next approach was to explore the process of Transaction Lossing. In a nutshell, Transaction Lossing provides a "copy" in which it occurs. These transactions are then available and together with the backup from time T(n) provide the information to restore the data environment to the status at the exact time of failure (assuming, of course, that some facility is provided to "rerun" these transactions against the data base).

In our survey we found several questions which must be asked to define this "Lossing" process.

What do we lose? Our answer is primarily a function of why we lose. If we lose transactions for data recovery only then we need lose only those transactions which modify our data base and more exactly only that data which has been actually updated. If we would also like to analyze all activity against the system we must necessarily store more transactions with more information. This approach could provide information for audit trail or system evaluation as in whoenteredhowmanyofwhatwhen.

Having defined the why and what we move to the next logical process, the definition of how we implement logging and implicitly the restore procedures. We considered several avenues of approach. At the outset lossing appeared to be a relatively application dependent process, similar to yet sufficiently different for each process to require many lose and restore procedures and involve much effort to create and maintain the recovery software.

After studying the data storage technique chosen, HP Image 3000, we developed what appeared to be a generalized lossing/recovery process using the data base itself to define the structure and content of the data base at recovery time. Our approach was to provide user callable procedures with which to accomplish the file handling and transaction record output. However, in our attempt to implement this facility we have encountered some difficulties yet to be resolved to our satisfaction:

1. Multiple processes must be able to concurrently share a single file (of variable record length to allow maximum blocking efficiency) and to maintain a single I/O buffer for that file to insure that records are lossed in the sequence the transactions are processed. The overhead (coding, maintenance and execution) required to define, open and write to the lose file might possibly be more efficiently performed by IMAGE.
2. Where do we create the lose file? Preferably offline to cassette or discette or some similar relatively inexpensive medium.

3. User written lossing procedures cannot inteface with the Data Base inquiry subsystem QUERY. Our choice must then be to either strictly control the use of QUERY since QUERY modifications to the Data Base would not be lossed or to prohibit the use of this program.

## In Summary

The subject of integrity in the data environment of an on-line computer facility should weigh heavily in the design of that facility and in its daily management and operation. Data base management technology is one area in which there is currently great interest and with which an attempt has been made to integrate user data into a common working structure to improve cohesion, reliability and availability. It is felt that by providing a lossing and restore facility to the data base structure would greatly improve the reliability of on-line processing, and add significantly to the market potential of the system.

Further, on the basis of our experience in attempting to provide this capability, we believe that optimization requires that the lossing utility take advantage of operating under system rather than user control.

We feel the subject of data integrity to be of sufficient importance to the user in an on-line environment to warrant the interest of the vendor and request endorsement by the HP 3000 International Users Group in the request that Hewlett-Packard initiate a hi-priority project to provide a transaction lossing facility as an enhancement to the IMAGE/QUERY 3000 Data Base Management Subsystem.

Following are our lossing procedure definitions and the code developed to perform the lossing function. Please note this code was developed with the concept of sharing a common buffered variable record length disc file. It is currently our belief that the process should lose to an unbuffered shared device possibly a tape cartridge to provide an inexpensive offline lossing medium.

```
**********************************************************************
*      DATA BASE LOG  06.08.76:C.R.Van Ausdall                       *
**********************************************************************
      All activity which in any way changes any of the supported
      (by program reference) H.P. 3000 IMAGE data bases with the
      exception of H.P. QUERY access will be logged to provide the
      capacity to restore data base integrity subsequent to system
      or program crashes

      * the format of the logged record will be as follows:
        1) Current time and date from system clock
              DATE YYDDD (Julian date from procedure CALENDAR     )
                                                    1word-integer
              TIME HHMMSSTT (from procedure CLOCK)  2word-integer
              (see system procedure definitions for description  )
              (of format in MPE OPERATING SYSTEM reference manual)
        2) DATA BASE NAME                            6byte-character
        3) DATA SET NAME                            16byte-character
        4) MODE - add, change, or delete code        1word-integer
        5) STATUS - contents of status array returned by
                    last D.B. call                  10word-integer
                    (see IMAGE reference and D.B. Call ie...DBPUT)
                    (for definition of format                    )
        6) LIST - a concatenated string of item names passed
                  as a parameter to the D.B. call ie...DBPUT
                    (see IMAGE reference for definition          )
        7) LIST LENGTH - an integer, the value of which is the
                  length of the array LIST including delimeters
                  and terminator.
        8) BUFFER - is an array containing the values of the
                  fields defined by the array LIST in a
                  corresponding sequence
        9) BUFFER LENGTH - an integer, the value of which is
                  the length of the array BUFFER.

      * procedure DBLOG(MPEFNO,DBASE,DSET,DMODE,DSTAT,DLIST,DLSTLEN,
                        DBUFF,DBUFLEN)

        ARRAY(word) : DBASE, DSET, DSTAT, DLIST, DBUFF
        INTEGER : MPEFNO, DMODE, DLSTLEN, DBUFLEN

        All variables with the following exceptions are defined
        in the IMAGE reference manual.
        MPEFNO - the mpe file number of the logging file
                 as opened by DBOPNLOG.
        DMODE - the value of mode is determined by the
                transaction against the data base which
                preceeded (triggered) the call to DBLOG.
                This may have been a DBPUT, DBUPDATE, DBDELETE
                MODE = 1 if a DBPUT is to be logged.
                MODE = 4 if a DBUPDATE is to be logged.
                MODE = 7 if a DBDELETE is to be logged.
        This procedure will actually write a log record to the
        Data Base logging file LOGFIL.
```

```
* procedure DBOPNLOG(MPEFNO)

  INTEGER : MPEFNO

  This procedure defines and opens a variable length
  MPE file designated as the file LOGFIL.group.account
  and returns the MPE file number to the calling routine,
  to be used in calls to the logging procedure DBLOG.
```

```
GE 0001   HEWLETT-PACKARD 32102A.01.6   FORTRAN/3000     WED, JUL 14, 1976,  7:25 PM

001000  $CONTROL USLINIT,MAP,LABEL
002000        PROGRAM DRIVLOG
003000  C
004000  C*****************************************************************
005000  C
006000  C     This program is an example of the use of
007000  C     D.B.  Logging procedures DBOPNLOG and DBLOG...06/30/76:cv
008000  C
009000  C*****************************************************************
010000  C
011000        IMPLICIT INTEGER (A-Z)
012000        CHARACTER *1
013000       ^   DBASE *26,
014000       ^   DSET *16,
015000       ^   DSTAT *20,
016000       ^   DLIST *24,
017000       ^   DBUFF *24
018000        DIMENSION
019000       ^   IBASE (13),
020000       ^   ISET  ( 8),
021000       ^   ISTAT (10),
022000       ^   ILIST (12),
023000       ^   IBUFF (12)
024000        EQUIVALENCE
025000       ^   (IBASE(1),DBASE),
026000       ^   (ISET (1),DSET ),
027000       ^   (ISTAT(1),DSTAT),
028000       ^   (ILIST(1),DLIST),
029000       ^   (IBUFF(1),DBUFF),
030000       ^   (ISTAT(2),I)
031000        DISPLAY "BEGIN DRIVLOG...DBLOG DRIVER PROGRAM"
032000  C
033000  C     OPEN LOG FILE AS SHARED AND RETURN MPE FILE NUMBER.
034000        CALL DBOPNLOG (MPEFNO)
035000        DISPLAY "I'VE OPENED THE LOGGING FILE"
036000        DISPLAY "PLEASE ENTER DATA BASE NAME XXXXXXXX"
036500        ACCEPT DBASE
037000        DSET = "SET NUMBERXXXXXX"
038000        DLIST = "LIST XXXXX123456789 1234"
039000        DBUFF = "BUFF XXXXX123456789 1234"
040000        LSTLEN = 12
041000        BUFLEN = 12
042000        DO 30 I = 1,100,2
043000        DO 30 J = 1,7,3
044000        MODE = J
045000  C
046000  C     KILL TIME...
047000        DO 20 K = 1,100
```

```
048000          Y = Y+1
049000          Y = Y-1
050000    20    CONTINUE
051000          CALL DBLOG (MPEFNO,IBASE,ISET,MODE,ISTAT,ILIST,
052000         ^             \LSTLEN\,IBUFF,\BUFLEN\)
053000          REC =  REC + 1
054000    30    CONTINUE
055000          DISPLAY "END DRIVLOG"
056000          DISPLAY REC, " RECORDS LOGGED."

GE 0002    DRIVLOG
057000          STOP
058000          END

SYMBOL MAP
ME                TYPE         STRUCTURE    ADDRESS                    NAME                T
FLEN              INTEGER      SIMPLE VAR   Q+ 12                     DBASE               C
LOG               SUBROUTINE                                           DBOPNLOG
UF                CHARACTER    SIMPLE VAR   Q+ 11,I                   DLIST               C
ET                CHARACTER    SIMPLE VAR   Q+  4,I                   DSTAT               C
                  INTEGER      SIMPLE VAR   Q+  7,I                   IBASE               I
UFF               INTEGER      ARRAY        Q+ 10,I                   ILIST               I
ET                INTEGER      ARRAY        Q+  3,I                   ISTAT               I
                  INTEGER      SIMPLE VAR   Q+ 14                     K                   I
TLEN              INTEGER      SIMPLE VAR   Q+ 19                     MODE                I
EFNO              INTEGER      SIMPLE VAR   Q+ 13                     REC                 I
                  INTEGER      SIMPLE VAR   Q+ 15

 LABEL MAP
 STATEMENT    CODE     STATEMENT    CODE     STATEMENT    CODE     STATEMENT    CODE     STA
   LABEL     OFFSET      LABEL     OFFSET      LABEL     OFFSET      LABEL     OFFSET      L

     20        324         30        345

**  NO ERRORS,  NO WARNINGS;   PROGRAM UNIT COMPILED  ****
MPILATION TIME    2.344 SECONDS    ELAPSED TIME     5.706 SECONDS
TAL COMPILATION TIME   0:00:03
TAL ELAPSED TIME       0:00:07
```

```
C
PAGE 0001   HEWLETT-PACKARD 32100A.06.0  SPL  WED, JUL 14, 1976,  7:26 PM  (C) COP
0
00002000  00000 0   $CONTROL NOWARN,INNERLIST
00003000  00000 0   $CONTROL MAP,SUBPROGRAM
00004000  00000 0   <<                                                          >>
00005000  00000 0   << DBOPNLOG...06/29/76:CV                                  >>
00006000  00000 0   << THIS PROCEDURE WILL OPEN AN MPE VARIABLE LENGTH FILE    >>
00007000  00000 0   << "LOGFIL" TO BE USED BY PROCEDURE "DBLOG" TO RECORD      >>
00008000  00000 0   << TRANSACTIONS AGAINST AN IMAGE DATA DBASE.  THIS LOG FILE >>
00009000  00000 0   << TOGETHER WITH A STORE TAPE OF THE DATABASE SHOULD PROVIDE>>
00010000  00000 0   << DATA INTEGRITY VIA A RESTORE PROGRAM "DBRERUN" TO RESTORE>>
00011000  00000 0   << A CRASHED DATA DBASE TO THE STATUS PRIOR TO THE CRASH.   >>
00012000  00000 0   <<                                                          >>
00013000  00000 0   << NOTE THAT ONLY TRANSACTIONS WHICH ACTUALLY MODIFY THE   >>
00014000  00000 0   << CONTENT OF THE DATA BASE SHOULD BE LOGGED...IE DBUPDATE >>
00015000  00000 0   << ...DBPUT AND DBDELETE.                                   >>
00016000  00000 0   <<                                                          >>
00017000  00000 0   << PROCEDURES FOR RESTORING FROM THE LOG FILE WILL BE FOU   >>
00018000  00000 0   << IN THE LOGGING SYSTEM REFERENCE AND OR THE RELOAD PROG..M>>
00019000  00000 0   << "DBRERUN".                                               >>
00020000  00000 0   <<                                                          >>
00021000  00000 0   BEGIN
00022000  00000 1   PROCEDURE DBOPNLOG (MPEFNO);
00023000  00000 1   INTEGER MPEFNO;
00024000  00000 1      BEGIN
00025000  00000 2      INTEGER E;
00026000  00000 2      BYTE ARRAY LOGFIL(0:6);
00027000  00000 2      INTRINSIC FOPEN,FCHECK;
00028000  00000 2      MOVE LOGFIL:="LOGFIL ";
                                    00000    ADDS,003              035003
                                    00001    LRA   S- 000          171700
                                    00002    LSL ,000,001          010201
                                    00003    STOR  Q+ 002          051402
                                    00004    ADDS,003              035003
                                    00005    LRA   Q+ 002,I        1734(
                                    00006    LRA   P+ 003          170003
                                    00007    LSL ,000,001          010201
                                    00010    BR    P+ 000          140000
                                    00011    INSERT OR FIXUP       046117
                                    00012    INSERT OR FIXUP       043506
                                    00013    INSERT OR FIXUP       044514
                                    00014    INSERT OR FIXUP       020040
                                    00015    LDI ,007              021007
                                    00016    MVB ,000,003          020043
00029000  00017 2      MPEFNO:=FOPEN(LOGFIL,1,%345);
                                    00017    ZERO, NOP             000600
                                    00020    LOAD  Q+ 002          041402
                                    00021    LDI ,001              021001
                                    00022    LDI ,345              021345
```

```
                                                00023     ADDS,013                035013
                                                00024     LOAD   P+ 000            040000
                                                00025     PCAL,000                000000
                                                00026     STOR   Q- 004,I          053604
00030000  00027 2          IF <> THEN BEGIN
                                                00027     BE     P+ 000           141200
00031000  00030 3                               FCHECK(MPEFNO,E);
                                                00030     LOAD   Q- 004,I          043604
                                                00031     LRA    Q+ 001           171401
                                                00032     ADDS,003                035003
1
PAGE 0002    HEWLETT-PACKARD
0
                                                00033     LDI ,030                021030
                                                00034     PCAL,000                000000
00032000  00035 3                               END;
00033000  00035 2       END;
                                                00035     EXIT,001                031401
```

| IDENTIFIER | CLASS | TYPE | ADDRESS |
|---|---|---|---|
| E | SIMP. VAR. | INTEGER | Q +001 |
| FCHECK | PROCEDURE | | |
| FOPEN | PROCEDURE | INTEGER | |
| LOGFIL | ARRAY | BYTE | Q +002 |
| MPEFNO | SIMP. VAR. | INTEGER | Q -004 |

```
00034000  00000 1    END.
```

| IDENTIFIER | CLASS | TYPE | ADDRESS |
|---|---|---|---|
| DBOPNLOG | PROCEDURE | | |

```
 PRIMARY DB STORAGE=%000;     SECONDARY DB STORAGE=%00000
 NO. ERRORS=000;              NO. WARNINGS=000
 PROCESSOR TIME=0:00:03;      ELAPSED TIME=0:00:12
/
```

```
C
PAGE 0001    HEWLETT-PACKARD 32100A.06.0  SPL  WED, JUL 14, 1976,  7:26 PM  (C) COP
0
00001000   00000 0    $CONTROL NOWARN,INNERLIST
00002000   00000 0    $CONTROL MAP,SUBPROGRAM
00003000   00000 0    <<                                                          >>
00004000   00000 0    << DBLOG...06/24/67:CV                                      >>
00005000   00000 0    << This procedure will log transactions against an IMAGE    >>
00006000   00000 0    << Data Base to a previously defined and opened Shared MPE  >>
00007000   00000 0    << variable length file referenced by the MPE file number   >>
00008000   00000 0    << MPEFNO.  This LOGFIL together with the STORE tape of     >>
00009000   00000 0    << the Data Base taken prior to D.B. modifications should   >>
00010000   00000 0    << provide "FAIL SOFT" capability and allow restoration of  >>
00011000   00000 0    << data integrity to the instant prior to the failure.      >>
00012000   00000 0    <<                                                          >>
00013000   00000 0    << NOTE THAT ONLY TRANSACTIONS WHICH ACTUALLY MODIFY THE    >>
00014000   00000 0    << CONTENT OF THE DATA BASE SHOULD BE LOGGED...IE DBUPDATE  >>
00015000   00000 0    << ...DBPUT AND DBDELETE.                                   >>
00016000   00000 0    <<                                                          >>
00017000   00000 0    << PROCEDURES FOR RESTORING FROM THE LOG FILE WILL BE FOU...>>
00018000   00000 0    << IN THE LOGGING SYSTEM REFERENCE AND OR THE RELOAD PROGRAM>>
00019000   00000 0    << "DBRERUN".                                                >>
00020000   00000 0    <<                                                          >>
00021000   00000 0    BEGIN
00022000   00000 1    PROCEDURE DBLOG(MPEFNO,DBASE,DSET,DMODE,DSTAT,DLIST,DLSTLEN,
00023000   00000 1                    DBUFF,DBUFLEN);
00024000   00000 1    VALUE DLSTLEN,DBUFLEN;
00025000   00000 1    INTEGER DMODE,MPEFNO,DLSTLEN,DBUFLEN;
00026000   00000 1    ARRAY DBASE;
00027000   00000 1    ARRAY DSET;
00028000   00000 1    ARRAY DSTAT;
00029000   00000 1    ARRAY DLIST;
00030000   00000 1    ARRAY DBUFF;
00031000   00000 1       BEGIN
00032000   00000 2       INTEGER DATE,LENGTH;
00033000   00000 2       ARRAY LOGREC(0:1023);
00034000   00000 2       LOGICAL PARM,WAIT;
00035000   00000 2       INTEGER POINTER SDATE := @LOGREC(0);
00036000   00000 2       INTEGER POINTER LMODE := @LOGREC(24);
00037000   00000 2       DOUBLE POINTER STIME  := @LOGREC(1);
00038000   00000 2       INTRINSIC FWRITE,FUNLOCK,FLOCK;
00039000   00000 2       INTRINSIC FCHECK,FCONTROL;
00040000   00000 2       INTRINSIC CALENDAR,CLOCK;
00041000   00000 2       <<  BUILD LOG RECORD AFTER GETTING TIME STAMP
00042000   00000 2       PARM                    := TRUE;
                                    00000    ADDS,011                 035011
                                    00001    LRA   S- 000             171700
                                    00002    STOR  Q+ 003             051403
                                    00003    LOAD  P+ 000             040000
                                    00004    ADDS,000                 035000
```

```
                                        00005     LOAD  Q+ 003           041403
                                        00006     STOR  Q+ 006           051406
                                        00007     LOAD  Q+ 003           041403
                                        00010     ADDI,030               022430
                                        00011     STOR  Q+ 007           051407
                                        00012     LOAD  Q+ 003           041403
                                        00013     ADDI,001               022401
                                        00014     STOR  Q+ 010           051410
                                        00015     LDNI,001               025001
                                        00016     STOR  Q+ 004           051404
1
PAGE 0002   HEWLETT-PACKARD
0
00043000  00017 2       WAIT                := TRUE;
                                        00017     LDNI,001               025001
                                        00020     STOR  Q+ 005           051405
00044000  00021 2       LENGTH              := DLSTLEN + DBUFLEN + 35;
                                        00021     LOAD  Q- 006           041606
                                        00022     ADDM  Q- 004           071604
                                        00023     ADDI,043               022443
                                        00024     STOR  Q+ 002           051402
00045000  00025 2       SDATE               := CALENDAR;
                                        00025     ZERO, NOP              000600
                                        00026     PCAL,000               000000
                                        00027     STOR  Q+ 006,I         053406
00046000  00030 2       STIME               := CLOCK;
                                        00030     DZRO, NOP              000700
                                        00031     PCAL,000               000000
                                        00032     STD   Q+ 010,I         163410
00047000  00033 2       MOVE LOGREC(3)      := DBASE,(13);
                                        00033     LDXI,003               021403
                                        00034     LRA   Q+ 003,I,X       177403
                                        00035     LRA   Q- 013,I         173613
                                        00036     LDI  ,015              021015
                                        00037     MOVE,004,003           020023
00048000  00040 2       MOVE LOGREC(16)     := DSET,(8);
                                        00040     LDXI,020               021420
                                        00041     LRA   Q+ 003,I,X       177403
                                        00042     LRA   Q- 012,I         173612
                                        00043     LDI  ,010              021010
                                        00044     MOVE,004,003           020023
00049000  00045 2       LMODE               := DMODE;
                                        00045     LOAD  Q- 011,I         043611
                                        00046     STOR  Q+ 007,I         053407
00050000  00047 2       MOVE LOGREC(25)     := DSTAT,(10);
                                        00047     LDXI,031               021431
                                        00050     LRA   Q+ 003,I,X       177403
                                        00051     LRA   Q- 010,I         173610
                                        00052     LDI  ,012              021012
                                        00053     MOVE,004,003           020023
```

```
00051000   00054 2       MOVE LOGREC(35)          := DLIST,(DLSTLEN);
                                        00054     LDXI,043                021443
                                        00055     LRA    Q+ 003,I,X       177403
                                        00056     LRA    Q- 007,I         173607
                                        00057     LOAD   Q- 006           041606
                                        00060     MOVE,004,003            020023
00052000   00061 2       MOVE LOGREC(35+DLSTLEN):= DBUFF,(DBUFLEN);
                                        00061     LDI ,043                021043
                                        00062     ADDM   Q- 006           071606
                                        00063     STAX, NOP               004300
                                        00064     LRA    Q+ 003,I,X       177403
                                        00065     LRA    Q- 005,I         173605
                                        00066     LOAD   Q- 004           041604
                                        00067     MOVE,004,003            020023
00053000   00070 2       FLOCK(MPEFNO,WAIT);
                                        00070     LOAD   Q- 014,I         043614
                                        00071     LOAD   Q+ 005           041405
                                        00072     PCAL,000                000000
00054000   00073 2       FWRITE(MPEFNO,LOGREC,LENGTH,0);
                                        00073     LOAD   Q- 014,I         04361-
1
PAGE 0003    HEWLETT-PACKARD
0
                                        00074     LOAD   Q+ 003           041403
                                        00075     LOAD   Q+ 002           041402
                                        00076     ZERO, NOP               000600
                                        00077     PCAL,000                000000
00055000   00100 2       FCONTROL(MPEFNO,6,PARM);
                                        00100     LOAD   Q- 014,I         043614
                                        00101     LDI ,006                021006
                                        00102     LRA    Q+ 004           171404
                                        00103     PCAL,000                000000
00056000   00104 2       FUNLOCK(MPEFNO);
                                        00104     LOAD   Q- 014,I         043614
                                        00105     PCAL,000                00000-
00057000   00106 2       END;
                                        00106     EXIT,011                031411
```

| IDENTIFIER | CLASS | TYPE | ADDRESS |
|---|---|---|---|
| CALENDAR | PROCEDURE | LOGICAL | |
| CLOCK | PROCEDURE | DOUBLE | |
| DATE | SIMP. VAR. | INTEGER | Q +001 |
| DBASE | ARRAY | LOGICAL | Q -013 |
| DBUFF | ARRAY | LOGICAL | Q -005 |
| DBUFLEN | SIMP. VAR. | INTEGER | Q -004 |
| DLIST | ARRAY | LOGICAL | Q -007 |
| DLSTLEN | SIMP. VAR. | INTEGER | Q -006 |
| DMODE | SIMP. VAR. | INTEGER | Q -011 |

```
DSET                    ARRAY           LOGICAL     Q -012
DSTAT                   ARRAY           LOGICAL     Q -010
FCHECK                  PROCEDURE
FCONTROL                PROCEDURE
FLOCK                   PROCEDURE
FUNLOCK                 PROCEDURE
FWRITE                  PROCEDURE
LENGTH                  SIMP. VAR.      INTEGER     Q +002
LMODE                   POINTER         INTEGER     Q +007
LOGREC                  ARRAY           LOGICAL     Q +003
MPEFNO                  SIMP. VAR.      INTEGER     Q -014
PARM                    SIMP. VAR.      LOGICAL     Q +004
SDATE                   POINTER         INTEGER     Q +006
STIME                   POINTER         DOUBLE      Q +010
WAIT                    SIMP. VAR.      LOGICAL     Q +005


)0058000  00000 1    END.

      IDENTIFIER          CLASS          TYPE          ADDRESS

    DBLOG                 PROCEDURE


PRIMARY DB STORAGE=%000;    SECONDARY DB STORAGE=%00000
NO. ERRORS=000;             NO. WARNINGS=000
PROCESSOR TIME=0:00:05;     ELAPSED TIME=0:00:20
```

ARTICLE FOR HP 3000 USER'S GROUP NEWSLETTER

At the Baltimore User's Group Meeting, questions were submitted to HP concerning the operation of, and enhancements to, the HP 3000. Responses to those questions which deal with how the system works, how to work around an error condition, what an error message means, etc., are given below. HP's response to questions regarding enhancements or new products are either included in the User's Group Questionnaire or are considered confidential information which we do not feel comfortable releasing to our User's Group (and thus potentially our competitors).

SECTION I. Questions on MPE/SYSTEM

1. Some computers log disc hardware/software retries to enable the user or CE to monitor the error rate of a particular disc. In many cases preventative maintenance can be performed before the disc actually fails and crashes the system. Since the Series II has memory chip error logging, would it be possible to expand this concept to other physical devices?

   [Unlike memory, the system devices return error conditions via the controller to the I/O system in MPE. If I/O error logging is enabled, such errors are recorded in the current log file for subsequent listing using the system Utility LISTLOG2. Note that each error is listed, and no analysis performed.]

2. A tape read error during a RELOAD is absolutely fatal, even though only one file is usually involved. Why can't error recovery be included to:
   1. attempt several rereads, and
   2. if the rereads fail, purge the affected file from the directory and go on to the next?

   [During a RELOAD, tape parity errors detected within a user file actuates a purge of the file and a forward space to the next file. Currently, no retries take place but we are intending to perform this function in the near future. Realize that a parity error occurring within the MPE part of the RELOAD will be fatal....always.]

3. OUTCLASS parameter on JOB card. Can this be used to get all output to a file other than $STDOUT? We have not been able to make this work, but the documentation in the manual indicates this should be possible.

   [The OUTCLASS parameter on a :JOB commands provides a means to define $STDLIST for a job. The documentation is not too specific in places - in summary, the device specified must be configured as a serial output device.]

4. A Series II from CX change you cannot drop the outfence to zero in the Series II machine, to print priority one files. Why not print => outfence, and allow the user to put priority zero on his file command for complete deferment of output?

   [The feature to enable users to completely defer output by specifying an OUTPRI=1 was considered to be the simplest mechanism, enabling an operator to distinguish between User and System deferments.]

5. LOG files. Why is old log file not reopened and closed properly after a crash? At least why not a utility program that the System Manager can direct toward the improperly closed file? How does the System Manager reset the numbering of the log files?

   [Log files are not automatically cleaned up after a system crash because of two reasons. First, it is not at all obvious as to which was the last open log file and secondly, the internal structure prevents a simple file open/file close mechanism. We recognize the suggestions to provide this facility, plus a reset of the log file number by the System Manager and both will be implemented.]

6. LISTF: Why do LISTF options 0 and 1 not indicate which files are currently in use, as the manual says they should?

   [A :LISTF with option $\geq$ indicates by an * which files are currently open.

   Option 0, disagrees with documentation, in not doing the same. The reason is that we decided not to access every file label in order to detect an open file. Disc access overhead is thus the trade-off. If it is universally felt that it should be done, it shall be, otherwise the documentation will be updated.]

7. What is the remedy to the following message on the System Console: (3000 CX)

JOB OVERLOAD, TYPE 4

("no job process count table JPCNT entry available" says the Operator's Manual.)

["Job overload type 4." It appears that this would occur having the job LIMIT set to the maximum configured limit, executing the maximum number of jobs plus at least one HIPRI job. To remedy this, keep the operator established job LIMIT less than the total configured limit with a negative adjustment allowing for the maximum number of concurrent HIPRI jobs.]

8. Is there any way to configure the Card Reader Driver IOCDRDO, so that it will accept punched cards in COLUMN BINARY CODE FORMAT (supported by the hardware with Operating Mode 1)? If not, is there any other way to read binary cards?

[Would need to change Filesize and Driver to do binary open.]

9. Is the Series II multiplexer limited to a max of 2400 baud?

[Yes.]

MPE/SYSTEM

10. I'm running RPG and I get spoofles which go over the 32 extents which are normally allocated. Therefore, my report just quits printing prematurely. How can I stop this from happening?

[The only current solution to prevent reaching the 32 extent limit on spoofles is to reconfigure the system and increment the "# of sectors per spoofile extent". Note that the document is incorrect - there is no upper limit of 1024.]

## SECTION II. Communications

1. Is HP considering software to support an IBM link that allows a 3000 to appear like a HASP workstation?

   [HASP workstation is in investigation phase.]

2. European Modems: Compatible with 3000? Will distributed processing work in Europe?

   [Distributed processing project is in development and will be supported in Europe.]

3. On Series II HP 3000, will the running of 3780 Emulator degrade the system as it does on Series CS when it is locked in memory while you wait online for mainframe response?

   [No, a different line driver is used on Series II.]

4. How far will terminal I/O advances go? Many smaller, less powerful systems have faster I/O than the HP 3000. What are your objectives or goals in this area? Some terminals (264X) will go 9600 baud. I've heard rumbles of 19200 baud in the future. Is this high speed I/O being considered for the HP 3000?

   [High speed terminal I/O is considered an important area for HP 3000 enhancement, and several projects are in development phase.]

## SECTION III. Data Base Management System

1. Does HP plan in QUERY to chain detail data sets based on a search item?

   [There is no short term plan to modify QUERY.]

2. When will IMAGE be improved to lock only records for update, put, delete? The IMAGE enhancement allowing "read only" accesses not to lock the entire data base was great. Set locking is now under consideration. Locking only particular records is the next logical step. Is this possible? How soon? This is absolutely necessary to make a truly online system run efficiently.

   [There is no plan to add record locking at this time because it would require major modifications to IMAGE. Data set locking is a simpler enhancement, and it is currently under investigation.]

3. IMAGE UTILITY ERRORS: Should display what the error was, i.e., wrong # at data sets. The octal address of where the error occurred is of little use to the customer...he doesn't have the source code/compile list to find what the error was. This problem has cost us many hours in situations where the manual does not indicate whether the proposed change to the schema is allowed or disallowed, especially in common situations such as adding a data set, changing set type or changing set linkages.

   [In some cases, these errors are associated with unsupported capabilities. Check with local HP support personnel.]

4. IMAGE SCHEMA RESTRUCTURE: Experience (a very costly teacher!!!) has proven that many seemingly simple data base instructural modifications are prohibitively costly due to the lack of utility flexibility (i.e., DBLOAD/DBUNLOAD) or sequence/occurrence (i.e., add/delete) sets in schema.

   1. We feel it is a must that these considerations be explained in the Manual.
   2. We feel the vendor should demonstrate greater interest in making the product it has provided more human engineered in terms of the implications of restrictions in real life situations.

   [An attempt will be made to make this information more visible in future manuals. The next enhancement release will provide a utility for quickly modifying password and security scheme.]

5. IMAGE DATA INTEGRITY: In the area of system/product enhancement, it is felt that the subject of Data Integrity subject to system reliability...i.e., the cost of system crash in a data base environment, be reviewed. We feel an integral function to data base maintenance (add/delete/update) should be to log these modifications to a non volatile media (tape/diskette). A major limitation in the approach to user designed/written logging systems would be the inability to interface to QUERY.

   [Transaction logging and recovery are currently in development.]

6. Empirical evidence indicates that QUERY locks the data base while performing a find operation without regard to the mode in which the data base was opened. The implication being that system management while attempting to provide optimum system response concurrent with system accessability is faced within our case an online order entry process accessing the data base in Mode 1 which can be locked out for up to 10 minutes by a user opening the D.B. in mode 5 and performing a serial read.

   1. Why lock the D.B. for a FIND operation?
   2. Why lock the D.B. if your mode of access doesn't require it?

   [QUERY has not been modified to include the latest IMAGE enhancements. These enhancements are currently under consideration.]

7. I have a problem with QUERY/IMAGE - Is there any way to allow different users to update data, find information and report information and report information concurrently? I'm using modes 1 and 5. Mode 1 for update, mode 5 for finds and reports.

   [Modes 1 and 5 allow concurrency if used in a user written program; QUERY enhancements are currently under consideration.]

8. IMAGE schemas have to be changed. How about a DBLOAD program that will load data to a changed schema? (Additions of data sets should pose no problems.)

   [A restructuring enhancement is possible but it requires substantial effort. These enhancements are being considered.]

9. Is there any reason HP cannot write and support a general purpose program that will allow a user to modify a search item in a detail data set?

   [No, however this can be done simply with two IMAGE calls today and it is low on our enhancement list.]

10. Is the data entry language (DEL) project continuing? What enhancements are planned? When can we expect new releases, and what, if any, plans are there to support other than 2640 terminals?

    [Yes, work is continuing on improved data entry facilities. Direct entry to files is being considered.]

## SECTION IV. Languages/Utilities

### COBOL

1. Will HP improve the object code generated by COBOL? For example, a figurative character comparison always generates at least 21 words of code including a PCAL when 12 or less are sufficient for equal lengths strings. There are more common statements that need simple optimization. (We have COBOL Ver. B).

   [COBOL object code optimization will be released through MIT 1646. The constructs that will produce more efficient object codes include:
   1.1 Character comparisons of the same length,
   1.2 Literal moves,
   1.3 COMP moves of the same size and characteristics.]

2. When can we expect system intrinsics which will interface with COBOL and the provision of good usable examples in the manuals?

   [The COBOL-74 compiler will be able to interface with system intrinsics directly.]

3. When can we expect a COBOL compiler which will produce compile time messages, rather than just ignoring the statement?

   [We need an example that the COBOL compiler ignores an erroneous statement without producing a compile time message.]

4. I understand that the COBOL project is alive and well. When can we expect the next version and what will be the improvements? When can we have ANS compatibility and improved debug features?

   [We are hard at work on a COBOL with ANS 74 compatibilities and the debug feature.]

### BASIC/FORTRAN

1. Is there a way, besides using an SPL subroutine, to get the PARM value from the RUN command by FORTRAN and/or BASIC programs? An example in the FORTRAN manual indicates this may be possible but does not actually show how to do it.

   [Compiled BASIC programs are intended to execute exactly as they did in the BASIC Interpreter. For this reason, no new features were added for the singular use by compiled programs. Since it would always be necessary to input programs which are intended to be compiled through the Interpreter, any new language constructs must have equally valid and equivalent meanings in the BASIC Interpreter. This requirement rules out some ostensibly "nice" extensions such as programmable use of control-Y. To permit programmable control over this function would necessitate giving up the debugging capabilities which the Interpreter offers.]

[The compiled BASIC programmer cannot make use of the PARM because the PARM is already being used for another purpose. See the BASIC Compiler manual, p. 2-1.]

[There is no way to get the PARM value in FORTRAN.]

2. Valid need exists for BASIC Interpreter to have the equivalent of the EDITOR's K-files. Feasible?

   [Since the BASIC workspace (roughly equivalent to the EDITOR K-file) is currently maintained in the program data stack rather than in a file, a disc back-up mechanism would impose a significant overhead which would have a very noticable affect on the BASIC response time. For this reason, we feel that a reliable back-up mechanism is not feasible.]

4. BASIC-CONVERT: We have found the bug which permits a "number" like "OA" to be converted to a numeric zero with no error branch taken to be a problem. Is this now being corrected? If not, why not?

   [With regard to the "OA" bug: this is not a BASIC bug. It was first discovered by FORTRAN programmers and was a problem in the Compiler Library. The bug was fixed in Compiler Library 32211C.04.02 dated July 12, 1976.]

5. BASIC-CONVERT: We have had, under some circumstances the conversion of a number to a blank. How can this be possible? Is it a known bug?

   [Has a bug report been submitted? We are not aware this problem has occurred.]

6. Are there any plans for expanding I/O option? e.g., READ USING (at INPUT USING) similar to PRINT USING? Is there a possibility that BASIC will ever be [updated] ?backdated? to use FORTRAN type input/output formatting, as is done on the HP 9830A?

   [There are no plans to incorporate fixed-field or formatted input into BASIC at this time. Nonetheless, we will keep this suggestion on file for consideration with other useful commercial features.]

ETC. (SORT, FCOPY, EDITOR, STAR)

1. STAR: Will STAR be available on Series II? If not, why not? Is there a convenient way of converting ASCII files to STAR files? If currently available, could a subroutine be implemented, callable from any language?

   [We have no plans to make STAR available on HP 3000 II, because of its low usage and generally inferior specification relative to other packages available. Users of STAR are encouraged to contribute utilities to the Users Group Library.]

2. By what date, if at all, does HP plan to enhance EDIT/3000 to include the type of capabilities used at ITEL leasing?

   [Ross Scroggs has contributed an intra-line editing feature to the contributed library which is usable through the PROCEDURE command of the Editor.]

3. What is HP's response to Ross Scroggs' fixes and new features?

   [Editor A.5.01 contains many fixes, including those from Ross Scroggs of Itel.]

4. FCOPY
   a) Why must the program loop forever when copying a variable length record of length zero as produced on the LOG file when there is a system failure?
   b) Why can one not space forward in a variable length record file?
   c) Why can I not fool FCOPY by using a file equation and MR to space forward in a log file?

   [a) Has a bug report been submitted? We are not aware of this problem.]
   [b) FSPACE spaces the blocksize when record is variable, and not record size, viz.

   ```
   BUILD X; REC = -80, 10, V, ASCII
   LISTF X, 1
         REC = -800 BF = 1]
   ```

   c) Probably because of b) above.]

HP3000 USERS GROUP MEMBERSHIP
April 1977

Adams, R.
Scott Paper
604-688-8131

Aeberhardt, Jean-Paul
Credit Lyonnais
508-70-00

Agrusti, Ray
Wayne Board of Education
201-694-7126

Ahrend, R. C.
Hewlett-Packard
201-265-5000

Alder, Rene
Hewlett-Packard
0222-3516213

Alexander, Barry
Shoffner Industries

Anderson, Eric
Vydec Incorporated
201-822-2100

Anderson, Gary
McMaster University

Arenales, Pedro
Instituto Nutricion
43-7-62

Asner, Bernard
University of Dallas

Atamanchuk, Bill
Domtar Limited
514-282-5606

Baker, Major B. E.
Canada Forces
613-996-4624

Bale, Jonathan
Hewlett-Packard

Balnys, Paul
Hewlett-Packard

Barkley, Don
Hewlett-Packard
408-249-7020

Becker, A.
Hughes Aircraft
714-871-3232

Bennett, Robert
Spring Anesthesia Group
213-868-9779

Bensen, Owen
Hewlett-Packard
612-636-0700

Berchtold, Roger
Hughes Aircraft
714-871-3232

Bhuta, Manny
Allied Chemical

Biggs, Gary
University of Dallas
214-438-1123

Blanchard, Reginald
L'Hotel Dieu De Quebec
418-694-5364

Blood, K. L.
Sattelite Computing
804-545-8429

Booth, Bud
Automation Analysis

Borden, John
Baltimore Gas & Electric
301-234-5781

Bottegal, Thomas
George Washington University

Bowden, Milton
Allan Hancock College

Boydstun, E. E.
Teledyne Systems
213-886-2211

Braendi, E. R.
Technikum Winterthur

Brodowski, Richard
Independent Press-Telegram
213-435-1161

Brown, Dale
Mary Washington College
373-7250

Brown, Dana
Employee Benefits Insurance
408-998-4490

Brown, Paul
Diversified Systems
314-727-7988

Byrant, J. H.
Virginia Commonwealth University
804-770-7371

Bryden, William
San Bernadino Valley MWD
714-824-2200

Burgess, Guy
Hewlett-Packard

Burggrabe, W. F.
Nooter Corporation
314-621-6000

Buten, Richard E.
Procter & Gamble
513-562-0643

Campbell, Bruce
Hewlett-Packard

Caraza, Luis
Technologico De Monterrey
91-463-26090

Chambers, Carolyn
Liberty Communications
503-343-3301

Chernoff, Dennis
Hughes Aircraft
213-648-3416

Churchman, Vici
Pioneer Oil
817-531-3776

Clark, Ben
Dow Badische
804-887-6688

Clark, Gordon
Far-West Data Systems
714-556-4585

Clarke, Peter
School of Signals

Clayton, V. A.
American Permalite
703-586-8565

Clifton, Jim
Harbor General Hospital
213-328-2380

Comegys, George
Anne Arundel County
301-969-9000

Cook, Howard
Barth & Dreyfuss
213-627-6051

Corcoran, John
Hewlett-Packard

Covington, Wayne

Crabtree, Jim
Becton-Dickinson Corp.
201-460-3237

Cressie, Jim
Cook Co. Dept. of Correction
312-847-5627

D'Abate, Joann
Standard Oil of Ohio
216-575-5178

D'Agostino, William
Hoerbiger Corp. of America
516-484-5454

Dalvi, V. S.
Northern Telecom
514-931-5711

Daniels, Robert
Zischke Organization
415-986-6568

Day, Victor
Norwich Pharmacal
607-335-2427

De Carlo, Gianni
Hewlett-Packard
0039-2-6251

De Foucault, Charles
Hewlett-Packard
613-225-6530

De Meyer, Christian P.
L. L. & N. De Meyer
091-220011

Dewachi, A. I.
Iraq Mins. Industry & Minerals

Dobruk, Jerry
Lynes United Services
403-262-4501

Donaghue, G. A.
Nedco (1975) Ltd.
514-341-3700

Drynan, Gil
Boeing Aerospace
206-773-8114

Dumas, Richard
Computer Resources
415-941-4646

Eads, Bill
Hewlett-Packard

Eagle, Dennis
Hewlett-Packard
303-667-5000

Earls, John
Arthur A. Collins Inc.
214-661-2928

Edwards, Constance
St. Francis Xavier Univ.
902-867-2295

Edwards, Paul
Hewlett-Packard
214-231-6101

Edwards, Robert
Nat'l Conf. State Legislatures
202-624-5479

Eickhoff, Dennis
Nationwide Financial Services
314-567-9070

Engberg, Anton
Advanced Data Group
408-446-4346

Engstrom, Ed
Petro-Canada
403-264-7015

Erie, Scott P.
Donaldson Company
612-887-3131

Esmanhoto, Luiz Maria
Promon Engenharia

Farrell, Michael
General Mills
612-540-7067

Feinstein, David
University of Wisconsin
715-425-3358

Feldman, Daniel
Bose Corporation
617-879-7330

Feng, Roger
University of Santa Clara

Ferris, W. L.
Pacific Mutual Life

Fischer, Lee
Qume Corporation
415-783-6100

Flack, William
Southern College of Optometry

Flagel, Edwin
Rainier National Bank
206-587-2150

Fleck, G. C.
Premier Cablevision
604-324-4247

Fochtman, Gerald
Kalamazoo Public Schools
616-385-0596

Fontana, T. R.
Bechtel International

Foster, Bill
Hewlett-Packard
408-249-7020

France, J. E.
Northwest Territories
403-873-7521

Freyburger, Janice
Kalamazoo Public Schools
616-385-0596

Fuller, R. D.
Santa Monica

Furukawa, Shiro
Alexander Grant
213-786-3551

Gabbert, Jackie
ESL
408-734-2244

Gabet, Jean-Michel
Gnostic Concepts
415-854-4672

Gagne, J. Pierre
Northern Telecom
514-931-5711

Gardiner, J. A.
Pilkington Glass
416-694-3401

Garland, Art
Texaco
713-722-8381

Gates, William
Longs Drug Stores
415-937-1170

Gilchrist, Don
McMaster University
416-525-9140

Glenn, Lewis
Institut Cerac
021-349801

Godwin, Glen
Paktank Corporation
713-623-0000

Goldberg, Joe
Bose Corporation
617-879-7330

Goldberg, Michael
Financial Data Planning
305-858-1675

Gomez, Luis
Industrial Minera Mexico
564-70-66

Gonggryp, J. J.
Handelsvereniging Tradax
020-803803

Gordon, Paul
Signal Insurance

Gorfinkel, Martin
Los Altos Research Center
415-941-9310

Gott, David
Data Transformation

Graham, Brenda
Bose Corporation
617-879-7330

Graham, Gary
Hewlett-Packard

Gray, Larry
Granite Rock
408-724-5611

Gray, Norm
Newport Laboratories
714-540-4914

Green, Gary
Maryland Dept. of Education
301-796-8300

Green, Michael
Tandem Computers
408-255-4800

Green, Ronnie
Northern Telecom
408-961-9340

Griffin, Rick
Octal Systems

Griffith, William
Toro Company
714-688-9221

Groft, Garth
Borg-Warner

Guerrero, Jorge
Cetys College
903-768-1801

Guhl, Robert
Afsa Data
213-373-8661

Gunderson, Jim
Multnomah County IED
503-255-1841

Haase, Niels
IFV

Hackleman, William
College of the Mainland
713-938-1211

Hackman, Linford
Vydec Corporation
201-822-2100

Haefner, Paul
Hewlett-Packard

Hall, Craig
Taylor Produce & Storage
616-392-7171

Haman, Vince
Johnson Co. Data Processing
319-338-5933

Harbron, Tom
Anderson College
317-644-0951

Hayes, Garry
Union Oil
213-486-7260

Heath, Odis
Heath and Company
08-28-9480

Heda, Sharad
Vydec Incorporated
201-822-2100

Herbert, Tony
Maunsell & Partners Pty, Ltd.
60-1135

Herman, Martin
Harbor General Hospital
213-328-2380

Hill, Jack
Hewlett-Packard

Hines, Robert
Hitco
213-321-8080

Hoff, Marc
Hewlett-Packard
201-265-5000

Huggins, James
University of Dallas

Huttunen, Arto
Ahlstrom
952-63100

Hvamb, Widar
Tinius Olsens Tekniske Skole
034-32330

Hyvarinen, Ossi
Finland Post & Telegraph
90-717522

Iungerich, Conrad
Hewlett-Packard

Jackson, Charles
Collier-Jackson & Assoc.
813-985-1107

Jackson, Robert
Wake Forest University
761-5261

Jardine, Robert
Hewlett-Packard
415-493-1501

Jeans, Kenneth
Nooter Corporation

Jenkins, Harold
Fairfax Co. Public Schools
703-536-2600

Johnston, Chuck
Ferris Busscher & Lorman
616-392-8534

Johnston, Frank
Macon Telegraph
912-743-2621

Jolly, John
Becton-Dickinson Corp.
201-460-3237

Jones, Moran
Futura Press
512-442-7836

Jordan, M. G.
Boeing Aerospace
206-773-8114

Karnos, Carolyn
Hydrocomp Incorp.
415-493-5522

Keller, Darryl
Donaldson Company
612-887-3131

Kelly, Neal
Hewlett-Packard
301-948-6370

Kenfield, John
Hewlett-Packard
707-525-1400

Key, David
784-1436

Klamm, Phil
Hitco
714-833-9601

Klaus, George
Hughes Aircraft
714-871-3232

Kleineidam, Bodo
Hewlett-Packard
07031-667508

Klett, Donald
Sangamon State University
217-786-6549

Knudtsen, Dick
Hewlett-Packard
493-1501

Kopish, Frank
Hewlett-Packard
415-493-1501

Kragh, Nancy
Weyerhaeuser
206-593-7073

Kramer, A. C.
Solid State Scientific
215-855-8400

Kuehner, Warren
Hewlett-Packard

Lamb, Dennis
Universal Motor Fuel
316-264-9387

Lancaster, H.
Koppers

Laraway, C. M.
Monitor Information Systems
213-395-3217

Lasser, Daniel
Business Time Sharing Services
703-527-1424

Law, Jack
Hewlett-Packard

Lay, Sam
Draper's
714-540-7904

Leite, Denis
Promon Engenharia
282-4242

Lessey, Ken
Reichhold Chemicals
208-572-5600

Librarian
Conestoga College

Lindstedt, Sven
Sahlgren's Hospital

Lorren, Larry
Hewlett-Packard
408-249-7020

Lovestedt, Bob
Boeing Commercial Airplane
206-237-1397

Lovitt, John
Hewlett-Packard
314-567-1455

Lumb, Art
Proctor & Gamble
513-562-0700

Lund, Karen
University of Rochester
716-275-2269

Lundy, Clayton
Computer Resource Services
602-242-9121

Lysenko, Jim
Cosigma Incorporated
514-288-1031

MacKinnon, Ronald
Nova Scotia Educ. Compt. Net
902-867-2278

Machado, Luiz C. M.
Promon Engenharia
011-282-4242

Machado, P.
Scientific Micro Systems
415-964-5700

Mahadeen, Ramsey
Atlas Industrial Mfrg.
201-779-3970

Mahoney, Larry
R. W. Beck & Associates

Mahugh, Richard
Boeing Commercial Airplane
206-655-4526

Makela, Mariti
Kone Oy
914-13700

Manager, HP3000 Computer
Hewlett-Packard

Maines, Ralph
Hewlett-Packard
408-249-7020

Marchese, Buzz
Computer Composition Sales
314-652-4622

Marriott, Michael
MJK Associates
408-247-5102

Martin, Rick
Systems Consultants
212-736-6305

Martinez, Hector
Promon Engenharia

Matas, Claudio
Instituto Univ. De
Technologia

Matoza, Marc
Hewlett-Packard
514-697-4232

Matteson, Lewis
Matteson Development
713-465-6523

Maus, John
M. H. Golden Construction
714-291-8181

McCarthy, Raymond
Hewlett-Packard

McClain, Kenneth
General Computer
216-467-0880

McCleary, Gloria
Chester Engineering
412-262-1035

McEvoy, Chase
Birks McEvoy & Company
713-772-6633

McFarland, John
Metro Transportation Comm.

McLean, Scott
Hewlett-Packard
031-331-1000

Mecham, Douglas
Hughes Aircraft
714-871-3232

Merritt, Jim
Smith Industries
813-531-7781

Meyers, Larry
Ill. Law Enforcement Comm.

Miller, William
Principia College
618-466-2131

Milligan, Glen
Chrysler Corporation
313-956-2312

Mitchell, Robert
Automix Keyboards
206-885-8852

Neal, R. H.
Automated Business Services
804-355-4361

Neely, Dwayne
Hewlett-Packard
213-649-2511

Neibergs, George
ESB-Exide
215-342-8000

Nemeth, Lou
Philadelphia Water Department
215-MU6-3925

Neumyer, Richard
Western Electric

Nierengarten, John
University of Wisconsin
608-784-6050

Nokes, C. A.
Hughes Aircraft
714-871-3232

Nordman, Gary
Malkin & Pinton
Industrial Sup

Norman, Richard
Columiba Union College
301-270-9200

Oney, James
Rosenberg Capital Management
415-777-5474

Park, Han
Hewlett-Packard

Parker, Don
Sherwin Williams

Penrose, Doug
Hewlett-Packard

Peressini, M.
Airco Incorporated
201-464-8100

Petersen, Paul
Santa Rosa Jr. College
707-527-4385

Philpott, David
Philpott Hulme & Fisher
415-651-0111

Pierce, Tony
Boeing Aerospace

Pivak, Gerald
Becker Securities
212-747-4000

Pleasants, Joyce
Aurora Public Schools
303-344-8060

Pocklington, William
Wesley-Jessen
312-346-2000

Podkomorski, John
Hewlett-Packard
312-255-9800

Podolsky, Joe
Hewlett-Packard

Polk, Paul
ESL
408-734-2244

Powers, Dennis
William C. Brown Co.
319-588-1451

Price, John
Hewlett-Packard
408-249-7020

Pyatt, Gary
Pacific Mutual Life
714-640-3281

Ray, Jerry
Data Systems Consultants
402-289-3381

Rea, Stu
Hewlett-Packard

Reck, Robert
Kalamazoo Public Schools
616-385-0596

Reed, John
Airco Incorporated
201-464-8100

Reithner, Robert
National Econ. Research ASC
212-747-3939

Riley, Nancy
Long Beach Comm. Hospital
213-597-6655

Roberts, Ken
Munson Sporting Goods
714-979-0112

Rodrigues, Coelho Antonio
Persico Pizzamiglio
287-9955

Romer, Brett
College of the Desert
714-346-8041

Roseman, Garry
Iredell County
704-872-9501

Ross, Patrick
ESL
408-734-2244

Rubin, Paul
National Security Agency
301-688-6060

Sager, Kurt
Ciba-Geigy Photochemie

Salvadori, Leonel
Moinho Santista

Sanders, Dave
Hewlett-Packard
408-249-7020

Sanford, Dave
Pacific Outdoor Advertising
213-222-8185

Saunders, Jr., Richard
Wansona Manufacturing
704-694-4146

Sawyer, John
Wake Forest University
919-761-5354

Schiefer, Douglas L.
Investment Counseling Services
804-874-8603

Schmidt, John
Hewlett-Packard

Schmidt, Paul
Southern Baptist Convention

Schuler, Kurt
University of Dallas
214-438-1123

Schwar, Jim
Lafayette College
215-253-6281

Schwartz, Gerald
Hartford Insurance Group
203-547-3669

Seah, Ching-Kie
Hewlett-Packard

Secrest, Wilbur
Wesley-Jessen
312-346-2000

Seligson, David
Yale-New Haven Hospital
203-436-2765

Seveborg, Karl
Bofors
0586-36000

Shroads, James
HID Research
516-328-3941

Shuler, Scott
National Publishers Service
302-654-5688

Shumate, D. C.
Warren & Van Praag
314-994-3050

Shurman, Gary
Hospital Computer Systems
504-729-7391

Simon, George
Northern Telecom
514-931-5711

Simpson, Jad
British Columbia Hydro & Pr.
604-663-3395

Slater, Ted
Hewlett-Packard
604-254-0531

Smith, James
Inco Limited
416-363-6311

Smith, Sharon
National Bank of Detroit
313-225-3699

Smith, Terry
Milo Beauty & Barber Sup.
216-923-9953

Snider, Jim
AFSA Data
213-373-8661

Sohnle, Ronald
Canada Reg. Econ. Expansion
306-665-4388

Solt, Sam
Hewlett-Packard
415-493-1501

Sorenson, John
Val Paraiso University
219-464-5179

Speers, R. J.
Canada Reg. Econ. Expansion
613-996-3636

Spieler, Charles
Simco
415-785-8100

Starck, Richard
Ocean Air International
412-681-7533

Stiefel, E.
Habergger Ltd. Engineering
Works

Storaasli, Paul
Hewlett-Packard
493-1501

Stover, Lou
Denver Manpower Admin.
303-892-7131

Stump, Dale
Nevada Controllers Office
702-885-4330

Surowiec, Al
Airco Incorporated

Symonds, G.
Canada Nat'l Health & Wel.
613-995-9030

Taboada, Napoleon
La Constancia
22-0733

Tahtinen, Matti
Enso-Gutzeit Osakeyhtio

Tankersley, Jim
Proctor & Gamble
513-562-2262

Taylor, James
Hopper Associates
313-559-8530

Tembrock, Joe
Hewlett-Packard

Thistle, Carl
College of Cape Breton

Thomas, Roland
Rocky Mountain Bank Note
308-232-7171

Thomas, Walter
Advanced Technology
415-452-1401

Thompson, Don
Hewlett-Packard
206-454-3971

Thompson, Terry
Donaldson Company
612-887-3131

Thomson, James B.
R. Shriver Associates
201-335-7800

Thorland, Edward
Luther College
319-387-1177

Townsend, Charles
Kalamazoo Public Schools
616-385-0596

Townsley, M.

Trippet, L. D.
Hughes Aircraft
714-871-3232

Ulfers, Chuck
Hewlett-Packard

Van Ausdall, Chuck
Commercial Office Products

Van DePellel, F.
Bank Mendes Gans
020-238181

Vanderlugt, Garrett
Kalamazoo Public Schools
616-385-0596

Vandervoort, Dorothy
Admiral
312-292-5630

Vidal, Miguel
Universidad Catolica
47-51-11

Villa, Charles J.
Itel Corporation
415-983-0300

Vislosky, Michael
Mandate Corporation
216-861-8100

Vogel, John
Wells Fargo Bank
415-396-3969

Vollmer, Richard
ASARCO
915-532-7961

Vuillod, Bernard
Instituto Univ De
Technologia

Vuohu, Esko
Finland Geological Survey
90-464819

Wade, Gerald
Hewlett-Packard
505-526-2485

Waldo, Bill
Georgia Dept. of Admin.
Services

Walker, Merrill B.
Victor O. Schinnerer & Co.
202-686-2960

Wang, Scott
Hewlett-Packard
303-221-5000

Waterson, Gerald
Rutgers University
609-757-6065

Watson, David
Armament Systems
714-635-1524

Weber, Jon
University of Wisc.
608-784-6050

Welsch, John
Hewlett-Packard
408-249-7020

Welte, Larry
Hewlett-Packard

Wetherell, Russ
Columbia Union College
301-270-9200

Whitesell, Ann
ESL
408-734-2244

Whitney, Martin
Benetech Incorporated
916-481-1931

Wickel, Wilfried
Datendienst Telecomputer
0611-522364

Williams, Brian
Schlegel Corporation
716-244-1000

Williamson, Bob
Boeing Computer Services

Wilson, Doug
Conestoga College
519-653-2511

Woc, Rene
Telectro
6-1422

Wright, Norman
U. S. Civil Ser. Comm.
912-744-2078

Wright, William
Contra Costa Co. Water Dist.
415-682-5950

Yamada, Larry
Lockheed

Yauney, Bob
Silton Data
213-585-1161

Yergatian, Greg
Hewlett-Packard
617-890-6300

Yoskovitch, Israel
Nedco (1975) Ltd.
514-341-3700

Young, Robert
Nat'l Inst. Medical Research
01-959-3666

Zuercher, Peter
Hewlett-Packard

# EXECUTIVE BOARD

## CHAIRMAN

**Bill Gates**
Longs Drug Stores
141 North Civic Drive
Walnut Creek, Calif. 94596
(415) 937-1170

## RECORDS

**Gerald Schwartz**
Hartford Insurance Group
1 Hartford, Conn. 06115
(203) 547-3669

## HP INTERFACE

**William Bryden**
San Bernardino Valley Municipal Water District
1350 South "E" Street
San Bernardino, Calif. 92408
(714) 889-0433

## COMPUTER USAGE

**W. F. Burggrabe**
Nooter Corporation
1400 South Third Street
St. Louis, Missouri 63166
(314) 621-6000

## LIBRARY

**Dr. Gary Anderson**
Dept. of Biostatistics
School of Public Health
University of Washington
Seattle, Wash. 98103
(206) 543-1044

## MEETINGS AND REGIONAL USERS GROUPS

**Gil Drynan**
P.O. Box 313
Woodinville, Wash. 98072
(206) 773-8114

## DIRECTOR

**Gary Green**
Information Systems
Maryland Department of Education
P.O. Box 8717
Baltimore, Maryland 21240
(301) 796-8300

## PUBLICATIONS AND JOURNAL PAST PRESIDENT

**Doug Mecham**
Hughes Aircraft Company
P.O. Box 3310, Bldg. 601/H219
Fullerton, Calif. 92634
(714) 871-3232, x3077/3009

## HP REPRESENTATIVE

**Ralph Manies**
Hewlett-Packard
5303 Stevens Creek Blv.
Santa Clara, Calif. 95050
(408) 249-7020

## 1977 INTERNATIONAL MEETING HOSE

**Gil Drynan**
P.O. Box 313
Woodinville, Wash. 98072
(206) 773-8114