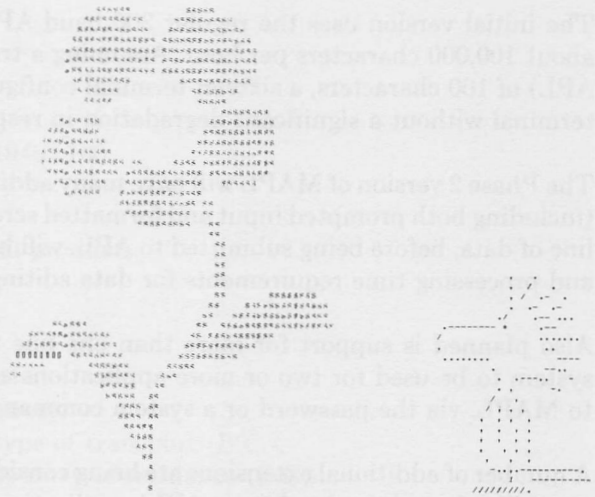
the I.P. Sharp newsletter

MAY-JUNE 1977
Volume 5 Number 3



MINIS SHARPEN APL USAGE

SHARP SPECIAL SYSTEMS

by Hugh O'Rourke

With an eye to making multi-terminal applications more cost-effective on *SHARP APL*, Sharp Special Systems is currently developing "MAPL", a minicomputer-based multi-terminal front-end for use with the *SHARP APL* system. MAPL will allow a number of terminals dedicated to the same application to be connected to one APL port.

MAPL will be particularly well suited to multi-terminal transaction-oriented data entry and/or enquiry applications where from four to sixteen terminals, all located within the same building, are all performing the same job. Order entry and inventory control are two good examples.

To illustrate, consider the order-filling aspect of an inventory control system for a nationwide distributor maintaining one central warehouse. Since the warehouse is large, and order turnaround time important, there is a need for, say, ten terminals at various locations throughout the plant. Assuming that each terminal needs to be on-line throughout the working day, based on the current $8 per hour rate, connect charges alone would amount to $640 per day. At these rates, the cost of the MAPL system would be recovered in about two months.

There will be additional, although less dramatic, savings. The terminals will be connected directly to MAPL, so only one local business line and modem will be required. MAPL will support most ASCII terminals, many of which are less expensive than the APL variety. And because all of the terminals will be connected to the same workspace, the problems inherent in updating a file from more than one workspace disappear.

Except for sign-on procedure, MAPL will be virtually transparent to both the user and the *SHARP APL* system. A number of CRT and/or hard-copy terminals will be attached to a small minicomputer. (Remember when all minicomputers were small?) One of these terminals will be designated the "master", and will be used to sign on and sign off *SHARP APL* in the regular fashion.

(continued)

MINIS (continued)

The Phase 1 version of MAPL assumes a rather simple application, where one input line to APL produces a reasonably short (say, less than 200 characters) response from APL. In certain cases the APL applications program will be written with an awareness of the multi-terminal aspect of the user, but all queuing of messages and correlating of input and output to the appropriate terminal will be done by the MAPL computer.

The initial version uses the regular 300-baud APL ports. A 300-baud line will, theoretically, transmit about 100,000 characters per hour. Assuming a transaction (including input to APL and response from APL) of 100 characters, a sixteen-terminal configuration will handle about 30 transactions per hour per terminal without a significant degradation in response time.

The Phase 2 version of MAPL will have many additional capabilities. Support for interactive applications (including both prompted input and formatted screens) will be available for data-entry applications. Each line of data, before being submitted to APL, will be checked for validity, thus reducing the programming and processing time requirements for data editing in the APL applications program.

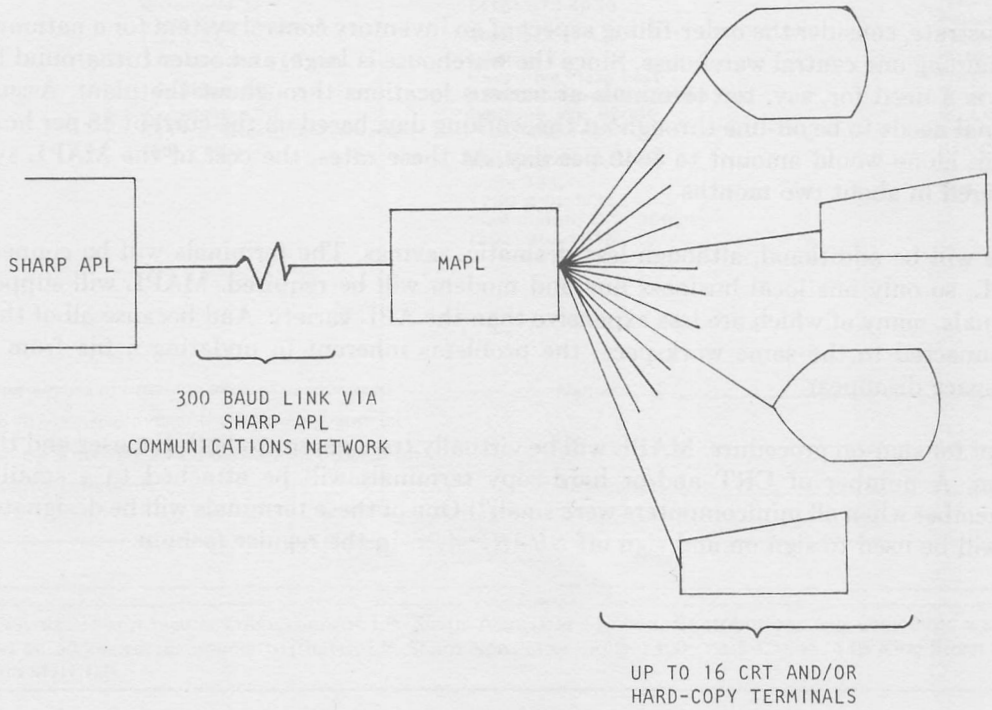
Also planned is support for more than one line to APL, allowing terminals connected via the MAPL system to be used for two or more applications concurrently. In this configuration, the user will signal to MAPL, via the password or a system command, which application is required.

A number of additional extensions are being considered for the future. A possible extension of the two-line capability is to have local (non-APL) applications in situ in the MAPL computer. An example of this might be local editing and/or balancing of data before it is transmitted to APL.

A "timeout" feature could be implemented in the MAPL system, designed to guard against unauthorized use of unattended terminals. And last, but certainly not least, the MAPL system opens up the possibility of collecting real-time data from, for example, a production process and feeding it on-line to APL for analysis.

We're sure some of you will have ideas for additional frills and features, and if so, we'd be happy to hear about them. If you would like further information, or to attend a demonstration of the Phase 1 version, contact Toronto Special Systems, or your local Sharp representative.

"MAPL" SYSTEM SCHEMATIC



NEW AVIATION DATA BASE - SERVICE SEGMENT DATA

by Rob Kronick

We are pleased to announce a new and very large data base available on the Sharp system: "Economic Regulation 586 - Service Segment Data" or ER-586 Data.

Until recently, service segment data was witheld from the public eye for a period of one year. After the recent institution of the Freedom of Information Act, this restriction was lifted, and the data is now timely. We now have on-line the data from January 1973 to December 1976. Updates will be done monthly, within 90 days of the end of every reporting period.

One of the outstanding features of this data base is that it is accessible in such an easy manner under the *MAGIC* system. *MAGIC* was designed specifically to simplify the analysis of time series data.

The data base is BIG! A total of 28 tape reels from the National Archives were used to create it. Certified route air carriers in the United States submit statistics for all scheduled flights to the Civil Aeronautics Board. Some 40-70 facts are retained for each flight segment: among others, the number of flights, enplaned freight, first-class passengers transported, and local non-priority U.S. Mail. The *SHARP APL* Aviation Newsletter (No. 4 - May, 1977) describes the data, the retrieval methods, and the interface with *MAGIC*. One of the examples from the newsletter appears below. It shows the first page of output from a small (19 lines) *MAGIC* program that prints a table for each of the approximately 50 flight numbers for flights between Seattle and Chicago. Only two tables are shown here.

(continued)

SEA→ORD
UA FLIGHT:140

	REVENUE DEPARTS PERF.	AVAIL SEATS	USED SEATS	LOAD FACTOR	TRANSP. FREIGHT (TONS	ENPLANED PASS	ENPLANED FREIGHT (TONS)	DEPLANED PASS (LOCAL)	DEPLANED FREIGHT (LOCAL)
JAN/76	31	5,476	3,105	56.7	9.2	3,105	9.2	2,928	7.7
FEB/76	29	5,110	2,834	55.5	11.9	2,834	11.9	2,619	11.3
MAR/76	30	9,744	3,154	32.4	88.5	3,154	88.5	3,153	88.3
APR/76	29	9,818	3,705	37.7	70.3	3,705	70.3	3,705	70.3
MAY/76	31	10,502	4,979	47.4	96.9	4,979	96.9	4,979	96.9
JUN/76	30	10,264	7,600	74.0	102.4	7,600	102.4	7,600	102.4
JUL/76	31	10,502	7,248	69.0	47.1	7,248	47.1	7,248	47.1
AUG/76	31	10,410	7,715	74.1	79.3	7,715	79.3	7,715	79.3
SEP/76	29	9,618	5,238	54.5	97.9	5,238	97.9	5,238	97.9
OCT/76	31	9,989	4,893	49.0	104.1	4,893	104.1	4,893	104.1
NOV/76	27	9,134	3,848	42.1	115.6	3,848	115.6	3,848	115.6
DEC/76	30	9,631	5,355	55.6	74.5	5,355	74.5	5,355	74.5
YEAR/76	359	110,198	59,674	54.2	897.7	59,674	897.7	59,281	895.4

SEA→ORD
UA FLIGHT:144

	REVENUE DEPARTS PERF.	AVAIL SEATS	USED SEATS	LOAD FACTOR	TRANSP. FREIGHT (TONS)	ENPLANED PASS	ENPLANED FREIGHT (TONS)	DEPLANED PASS (LOCAL)	DEPLANED FREIGHT (LOCAL)
JAN/76	29	6,970	3,821	54.8	55.8	71	2.4	57	0.5
FEB/76	29	6,863	3,250	47.4	57.7	248	4.6	228	2.8
MAR/76	31	7,466	3,956	53.0	44.8				
APR/76	30	7,133	4,139	58.0	32.1				
MAY/76	30	7,252	4,667	64.4	48.2				
JUN/76	30	7,262	5,559	76.5	92.2	831	16.8	684	15.9
JUL/76	30	7,260	5,981	82.4	53.0				
AUG/76	31	7,502	6,671	88.9	47.7				
SEP/76	30	7,147	5,707	79.9	54.4	4,274	51.1	3,732	40.4
OCT/76	30	7,260	4,936	68.0	68.4	3,886	67.0	3,406	28.1
NOV/76	28	6,776	3,839	56.7	80.2	423	19.4	408	9.8
DEC/76	28	6,663	4,770	71.6	47.9				
YEAR/76	356	85,554	57,296	67.0	682.2	9,733	161.3	8,515	97.4

ER-586 (continued)

We are optimistic about the usefulness of the new data base for the following reasons:

1. Service Segment data has been needed in the industry for some time. Now that the information is no longer restricted, we have it on-line, and easily accessed and manipulated with a little *MAGIC*.

2. This data is not available on-line anywhere else.

Rick Smith is our marketing coordinator for Aviation. Rick has been in the aviation industry for several years, and is headquartered south of Los Angeles at the Newport Beach office. Please contact Rick, and in Toronto, David Keith or Rob Kronick, for further information. A copy of the Aviation Newsletter is available free from the Toronto office.

THE SHARP APL MESSAGE PROCESSING FACILITY

All customers of I.P. Sharp Associates have access to a package called the "Message Processing Facility". Also referred to as the Mailbox, it provides the user with instant written communication with others. The internal memo is used extensively and successfully in most large organizations. The Mailbox facility provides the same service, but also provides two features the traditional memo systems lack. Mailbox is fast, and it is capable of sending the same message at once to any number of people in any location.

Its speed comes from the fact that all messages are entered directly into the computer, and made available to the recipients immediately. When each recipient signs on to the computer, he can ask the system to print the messages he has received up until the last second.

The user has complete control over the form of the message. It may be a simple statement or an elaborately-formatted document or report. The *SHARP APL* Text Editor makes it easy to modify text before printing it, or sending it through the Mailbox. The Mailbox also uses a comprehensive text editing facility of its own for correcting and adding to messages.

The Mailbox always provides a high level of security in message integrity regardless of the external classification of a letter. Each message is carefully protected; it is not possible for a user to examine or alter another person's mail. The user may also fully designate the security class, registration and urgency of a message before filing it. These markings are largely a matter of convention between the sender and the recipient. The system is designed to conform with the regulations governing and protecting common carriers in the telecommunication field.

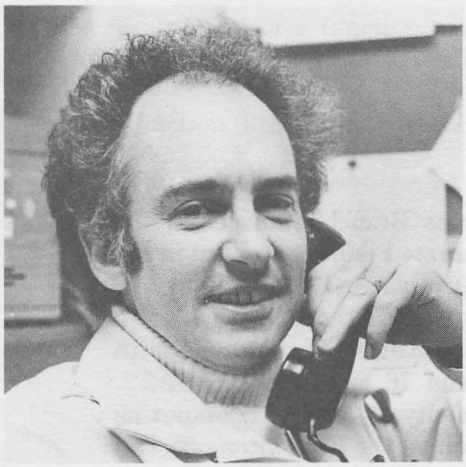
Each Mailbox member has an address code, usually his initials or an abbreviation of his name. A Mailbox member may also have certain group codes associated with him. A group usually contains individuals who have a common interest, or who are members of the same organization.

A Mailbox message may be addressed to a single person, to more than one, or to a group. Once sent, a message will remain in the Mailbox until each recipient has disposed of his copy. Even after a message is "accepted", it can be accessed for a limited time. There are programs for making inquiries on the status of messages, and for finding out the identity of other Mailbox users with the same security classification. The high level of security and its ease of use make the *SHARP APL* Mailbox facility a particularly powerful tool for rapid communication between offices, cities and continents. Please contact your Sharp representative for more information - and for a copy of the *SHARP APL* Mailbox brochure.

SHARP NEWS

NEW OFFICE IN THE U.S. - WASHINGTON, D.C.
We are pleased to announce that **Peggy Kueffer** has joined Sharp to manage the D.C. office. She can be found at:
Suite 307, 1730 K St. N.W.
Washington, D.C. 20006
(202)293-1895

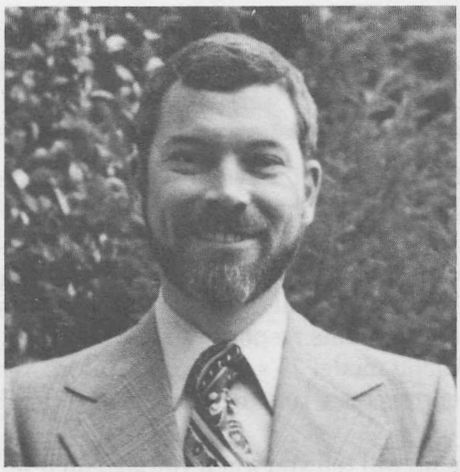
NEW OFFICE IN THE UNITED KINGDOM
Warrington, Cheshire - **Stephen Durnford** joined Sharp in March 1976 to look after business in the North and West of England. Local dial access was made available in May 1977 for users in **Liverpool** and **Manchester**. After several years in marketing and sales support in a non-computing environment, Stephen found APL in early 1975 as meeting his needs at the time, and has never looked back.
The office is at:
48A Horsemarket St.,
Warrington, Cheshire, England



Paul Berry

PALO ALTO
Paul Berry now has an office at:
Suite 110, 299 California St.
Palo Alto, Ca 94306
(415)327-1700

SEATTLE, WASHINGTON
Michael Crick has been manager of the Seattle office since it opened. We are pleased to announce that **Ross Hale** has joined Mike from Humboldt State University in Arcata, California. Prior to teaching APL and other computer courses at Humboldt, Ross s been working towards a PhD in Computer Science at the University of Delaware. He has had extensive experience dealing with customers in the real world having spent four years as a consultant with Arthur Anderson. He will add a great deal of strength to the Seattle office.
The new office address:
217 Executive Plaza East,
12835 Bellevue-Redmond Rd.,
Bellevue, Wa 98005 - U.S.A.
(206) 453-1771



Ross Hale

(continued)

SHARP NEWS (continued)

Ettie Ettinger

SOUTHERN CALIFORNIA - NEWPORT BEACH OFFICE

Ettie Ettinger became branch manager of the Newport Beach office in April. Ettie has been with the company since 1974. Her background is in education and programming and she has a Bachelor's degree in Business Information Systems (Data Processing) from California State University of Long Beach. During her years at Sharp she has gained a lot of experience with *MAGIC* and the various aviation data bases on the Sharp system, and has worked closely with the local aerospace contractors.

Frank Arthur

EDMONTON - ALBERTA

Frank Arthur has taken over as Branch Manager in the Edmonton office. Frank was Corporate Planning Analyst in the Calgary office, where he was also involved in building Financial Planning models, and in putting together *SHARP APL* Functions for Statistical Analysis. **Roger Hui** worked with Sharp during the summers of '75 and '76, mainly in *APLSTAT*.

Roger was class of '77 - University of Alberta (Computing Science and Mathematics), and joined the Edmonton office in May.

SYSTEM NEWS

Users of *SHARP APL* have probably noticed an increased level of general usage during the past month or so. The user load has been creeping up and peaks in excess of 160 are not uncommon.

Some months ago the system development group joined our two cpu's together, so that both could share the cpu load and the file system load during peak hours. This has worked well and we have been able to maintain reasonable response times. However, machinery does tend to malfunction every now and again and IBM 360 model 75's are no exception. When we lose a machine and one cpu has to carry the whole load, then response time can become intolerable.

In future we will tell you when you sign on if we are running on one cpu only. The message will only appear if the user load is 100 or more, and in effect it is warning that the service, although available, is liable to be seriously degraded. When we are running in a degraded mode, then response time is usually acceptable for trivial computing, function definition, etc., but can cause considerable back up of file operations. Thus file system intensive programs will tend to suffer most and should be avoided if at all possible.

Technical Supplement-11

REPLETE RESTARTABILITY

In TECHNICAL SUPPLEMENT-10, the techniques involved in the design of restartable Complex Sequential Updates (CSU) were reviewed. It will be recalled that the basic idea was to either provide redundant information that could be used to downdate the files to the condition that existed in the workspace at the time of a crash, or to segment the CSU into sets of smaller simple sequential updates, each one of which was readily restartable. In this issue, we shall consider another type of complex updating system.

A COMPLEX RANDOM UPDATE (CRU) is a process whereby a file is updated through the use of ⎕*REPLACE*; however, unlike a Simple Random Update (SRU) which replaces once for each ⎕*READ* from a source file, a CRU may replace a variable number of components or possibly none at all.

EXAMPLE: consider a source file, the first component of which contains a $2 \times N$ directory. The first row of this directory could conceivably contain part numbers, location codes, or perhaps even authorized user numbers. The second row contains the starting component of a variable number of consecutive components in which the information pertaining to the particular part number, etc., resides.

From time to time a zero might be inserted in *DIR*[⎕*IO*;*K*] indicating that the information concerning the particular part number **might** require change. In order to conclusively determine whether an update is indeed required, we assume the existence of a monadic explicit function called *CONDITION* which returns a 1 if the *ITEM* under consideration needs updating and a 0 otherwise. Moreover, if *ITEM* returns a 1, then *PROCESS* will update *ITEM* and replace the zero in *DIR*[⎕*IO*;*K*] with more meaningful information. A function that will accomplish all of the required updating is presented below:

```
      ∇ CR∆UPDATE;⎕IO;DIR;I;K;LIST;ITEM
[1]    ⎕IO←1 ◇ TIESOURCE
[2]    DIR←⎕READ SRC,1 ◇ I←1 ◇ K←0
[3]    LIST←(0=DIR[1;])/DIR[2;]
[4]   LP:ITEM←⎕READ SRC,LIST[I]+K
[5]    →(CONDITION ITEM)ρPRCS
[6]   INC:K←K+1
[7]    →(LIST[I+1]>LIST[I]+K)ρLP
[8]    I←I+1 ◇ K←0
[9]    →(I≤ρLIST)ρLP ◇ →0
[10]  PRCS:(PROCESS ITEM) ⎕REPLACE SRC,LIST[I ]+K
[11]   →INC
      ∇
```

Consider the difficulties associated with restarting *CR∆UPDATE* after a system crash, bearing in mind that the workspace environment could be as much as 4 seconds behind the condition of the source file. Specifically, we would have no way of knowing the last source file component number which was read. Furthermore, we do not know how many of the components associated with an item set were actually replaced.

In rendering this function restartable, we must look at the time stamps provided to us by *⎕RDCI* and compare them with an adjusted value of the time of the crash which can be taken from

```
¯1↑2 ⎕WS 4
```

as registered in the continue workspace. Hopefully, this should provide us with a synchronous point of re-entry into the body of *CR∆UPDATE* while at the same time affording us a mechanism to correct any potential damage that the directory might have incurred. The restarting process is detailed below:

```
    ∇ CR∆UPDATE;⎕IO;DIR;I;K;LIST;ITEM;TIME1 ;TIME2;INT;G
    .
    .
    .
[11]  →INC
[12] GOON:TIESOURCE ◇ TIME1←ADJUST ¯1↑2 ⎕WS  4 ◇ INT←1⌈I-1 ◇ G←0
[13] RDCI:TIME2←¯1↑⎕RDCI SRC,LIST[INT]+G
[14]  →(TIME1>TIME2)ρREQUIRED
[15] INCG:G←G+1
[16]  →((⎕SIZE SRC)[2]>LIST[INT]+G)ρRDCI ◇ →E ND
[17] REQUIRED:→(~CONDITION ⎕READ SRC,LIST[IN T]+G)ρNOUPDATE
[18]  I←1↑(⌽LIST[INT]+⍳G)⍳LIST ◇ K←0
[19]  DIR[1;DIR[2;]⍳LIST[I]]←(⎕READ SRC,1)[1; DIR[2;]⍳LIST[I]]
[20]  →LP
[21] NOUPDATE:→INCG
[22] END:
    ∇
```

As an alternative to this somewhat unwieldy *GOON*, the entire updating process can be easily converted to a simple sequential update with a much more manageable restart procedure. Specifically, a backup file will be maintained. Initially this file is empty, but as components are read from the source file, they are either directly appended or updated and appended to the backup file according to the result of *CONDITION*. Listed below then is a simple sequential version of the foregoing CRU.

```
      ∇ SS∆UPDATE;⎕IO;DIR;I;K;CMP;EOF
[1]    ⎕IO←1 ◇ TIEFILES
[2]    ⎕DROP BK,-/2↑⎕SIZE BK
[3]     DIR←⎕READ SRC,1
[4]    NEWD←DIR[;⍋DIR[2;] ◇ NEWD←NEWD,1,(⎕SIZE  SRC)[2]
[5]    NEWD ⎕APPEND BK ◇ I←1 ◇ K←0
[6]   START:CMP←NEWD[2;I]
[7]   NEXT:ITEM←⎕READ SRC,CMP+K
[8]    →((CONDITION ITEM)∧0=NEWD[1;I])⍴PRCS
[9]    ITEM ⎕APPEND BK ◇ →INC
[10]  PRCS:(PROCESS ITEM) ⎕APPEND BK
[11]  INC:K←K+1
[12]   →((CMP)<NEWD[2;I+1])⍴NEXT
[13]   I←I+1 ◇ K←0
[14]   →(I<¯1↑⍴NEWD)⍴START
[15]   (0 ¯1 ↓NEWD) ⎕REPLACE BK,1 ◇ →0
[16]  GOON:TIEFILES ◇ K←0
[17]   EOF←¯1+(⎕SIZE BK)[2] ◇ I←+/EOF>NEWD[2;]
[18]   NEWD[1;I+ 0 1]←(⎕READ BK,1)[1;I+ 0 1]
[19]   ⎕DROP BK,-1+EOF-NEWD[2;I]
[20]   →START
      ∇
```

In keeping with the summary remarks of Technical Supplement-8, (Nov.-Dec./76), we would like to point out that even without the introduction of the simple *GOON* logic, our *SS∆UPDATE* could have been restarted by →1, whereas attempting this approach with *CR∆UPDATE* would have been utterly disastrous. The conclusions therefore are self-evident - Keep Implementing Simple Systems.

CONTEST NO. 3: PARAGRAPHICS RESULTS

Congratulations to all of our Paragraphics contest participants. The brave entrants and their submissions are listed below:

*	Gary Buchwald	New England Mutual Life - Boston	*PARA∆GB*
*	Steve Clarke	I.P.S.A. - UK	*PARA∆SCL*
*	Clive Edwards	I.P.S.A. - Toronto	*PARA∆CED*
*	Don Emmens	National Energy Board - Ottawa	*PARA∆DE*
	Bob Hinton	Reed Ltd. - Quebec City	*PARA∆RPC*
	Don Howson	I.P.S.A. - Ottawa	*PARA∆DONH*
*	Don Leitch	Xerox - Toronto	*PARA∆JL*
*	David Liu	Xerox - Toronto	*PARA∆DL*
*	George Lount	I.P.S.A.- Toronto	*PARA∆GLO*
	Marvin Mandelbaum	I.P.S.A. - Rochester	*PARA∆MRM1+MRM2*
	Mike Powell	I.P.S.A. - Edmonton	*PARA∆POW1+POW2*
*	Robin Surtees	I.P.S.A.- UK	*PARA∆RSU*

All of these entries may be found in workspace 1436479 *PARAGRAPH*.

The names marked with asterisks indicate those entrants who most completely dealt with all of the output specifications. Of these however, only *PARA∆GLO* and *PARA∆JL* addressed themselves to the difficulties associated with not looping on the number of lines in the result as indicated in the procedural notes. Thus, on the basis of our primary judging criterion of completeness, the field was narrowed down to these two entries. Speed of execution on standard text then became the deciding factor as *PARA∆GLO* emerged more than 5.5 times faster than *PARA∆JL*. Accordingly, our first prize winner is ***George Lount*** while the best non-I.P.S.A. entry is awarded to ***Jon Leitch***!!! Our winners will be receiving a book prize of their choice in the near future. Congratulations again and thanx ⌊/⍳0 to all who took part.

*** CONTEST NO. 4 - THE MAXICON CONNECTION***

Required: A dyadic function with explicit result of the form:

```
Z←POS MAXICON MAT
```

which returns a boolean matrix *Z* of the same shape as *MAT* and where the 1's in *Z* define the **maxi**mally **con**nected region about and including the position *POS* in *MAT*.

Note:

(1) *POS* is a 2 element non-negative integer vector such that
`1=∧/POS≤⍴MAT`

(2) A connected region in *MAT* is a set of connected elements, where we define "connected elements" as follows:
Suppose *A*, *B*, and *C* are elements of *MAT*, then *A* is connected to *B* if both of the following conditions hold:

(i) *A*=*B*
(ii) *A* is vertically or horizontally adjacent to *B*

Moreover, if *A* is connected to *B*, and *B* is connected to *C*, then *A* will be considered to be connected to *C*. Also, it may be assumed that *A* is vacuously connected to itself.

(3) A maximally connected region is therefore the largest region, all of whose elements are connected to the element in position *POS*.

Examples:

(these examples assume that ⎕*IO* has been set to 1.)

```
      MAT
AAABA
ABBBA
BAABB

      2 3 MAXICON MAT
0 0 0 1 0
0 1 1 1 0
0 0 0 1 1
```

```
      MAT
4 1 2 2 1 4
1 4 3 3 4 1
2 3 4 4 3 2
2 3 4 4 3 2
1 4 3 3 4 1
4 1 2 2 1 4

      3 3 MAXICON MAT
0 0 0 0 0 0
0 0 0 0 0 0
0 0 1 1 0 0
0 0 1 1 0 0
0 0 0 0 0 0
0 0 0 0 0 0
```

Entries will be judged on the usual criteria of completeness, speed, conciseness, minimality of internal storage requirements, and elegance. Further examples and points of clarification will be added to workspace `999 CONTEST` as the need arises. The deadline for submissions is August 31. Please send your entries along with any other comments, criticisms and suggestions to the address below. Topics for future contests will also be welcomed.

Jerry Cudeck (Mailbox code: *JHC*)
I.P. Sharp Associates,
Suite 1400, 145 King Street West,
Toronto, Ontario,
M5H 1J8 CANADA

MAY-JUNE 1977

NTASK AND BTASK USAGE NOTES

by David Markwick

NTASKS and *BTASKS* have now been available in *SHARP APL* for some time and several methods of dealing with the various problems posed by these non-terminal tasks have been developed.

One of the problems that appears regularly with both types of tasks is that of the maintenance of a workspace containing an active ⎕*LX*. Obviously, the maintenance programmer loading the workspace does not wish the ⎕*LX* to be executed. One of the easiest methods to avoid the tasks automatically starting is to use ⌶28 (the terminal type). The value of ⌶28 in an *N*- or *B-TASK* is always ¯1.
Say ⎕*LX* has been assigned:

```
      ⎕LX←'Δ'
```

and the function Δ is:

```
      ∇Δ
[1]    →(¯1≠⌶28)⍴0
[2]    WORKΔTOΔBEΔDONE
∇
```

If, however, you only wish the task to run when it is, say, a *BTASK*, then the value of ⎕*RUNS* must be used. Assuming origin 1, the fourth column of ⎕*RUNS* determines the task type:

```
0 = TTASK
1 = NTASK
2 = BTASK
```

In addition, we need to know the *TASKID* of the task we are running, to determine which row of ⎕*RUNS* applies to our task. This may be taken from (2 ⎕*WS* 3)[11]. Thus, to determine what kind of task we have:

```
RUNS←⎕RUNS        ⍝ FIND VALUE OF ⎕RUNS
TASKIND←RUNS[;1]⍳(2 ⎕WS 3)[11]        ⍝ FIND ROW OF <RUNS>
                                        FOR OUR TASK
TASKTYPE←RUNS[TASKIND;4]              ⍝ FIND TASK TYPE
                                        FOR OUR TASK
```

We can now inspect the value of *TASKTYPE* to determine the kind of task we have.

Thus, to determine if our task is a *BTASK*, we could use the following function:

```
      ∇ BOOL←BTASK;R ;⎕IO
[1]    ⎕IO←1 ◇ R←⎕RUNS
[2]    BOOL←2∊R[R[;1]⍳(2 ⎕WS 3)[11];4]
∇
```

When the *NTASK* or *BTASK* has run to a successful conclusion, a workspace automatically gets saved, unless it is a clear workspace. To avoid a build-up of saved workspaces, the use of *CLEAROUT* as the last function called, will, in the event of successful conclusion, leave no saved workspace. Do not, of course, call *CLEAROUT* before everything else has completed, as termination of the task as a result of an error would also give a clear workspace. Using the example Δ, the function would now read:

```
  ∇Δ
[1]    →(¯1≠⌶28)⍴0
[2]    WORKΔTOΔBEΔDONE
[3]    CLEAROUT
∇
```

If an error occurs in an *NTASK* or *BTASK*, it always happens to someone else, or as a result of what someone else has done! It is most useful, then, to know which files were tied. A method of tracing files tied is to write functions which parallel the file "quad" functions and keep a note of ⎕*NUMS* and ⎕*NAMES* in global variables in the workspace. For example:

```
      ∇ FNAME ΔSTIE FTIE
[1]    FNAME ⎕STIE FTIE
[2]    ΔNUMS←⎕NUMS ◇ ΔNAMES←⎕NAMES
∇
```

and

```
  ∇ ΔUNTIE FTIES
[1]    ⎕UNTIE FTIES
[2]    ΔNUMS←⎕NUMS ◇ ΔNAMES←⎕NAMES
∇
```

This method would have to be applied to all uses of ⎕ file functions that changed the status of files tied in the workspace:
i.e.

⎕*TIE*, ⎕*STIE*, ⎕*ERASE*, ⎕*CREATE*, ⎕*UNTIE*, ⎕*RENAME*

A further sophistication is the use of tie numbers to indicate whether a file is tied exclusively, or share tied. A nice method is to say that even numbers are reserved for ⎕*TIE* (exclusive tie), and odd numbers for ⎕*STIE*. The function for Δ*TIE* and Δ*STIE* would then return the tie numbers. For example:

```
      ∇ R←ΔSTIE FNAME;⎕IO
[1]    ⎕IO←1 ◇ R←1+2×⍳20
                          ⍝ CREATE 20 POSSIBLE  STIE NUMBERS
[2]    R←''⍴(~R∊⎕NUMS)/R     ⍝ TAKE FIRST UNUSED S TIE NUMBER
[3]    FNAME ⎕STIE R
[4]    ΔNUMS←⎕NUMS ◇ ΔNAMES←⎕NAMES
∇
```

Based on this method, a function that takes the value of ΔNUMS and ΔNAMES, and reties all files, ΔRETIE, may be written:

```
      ∇ΔRETIE;I;⎕IO
[1]    ⎕IO←I←1
[2]   NFILE:→(I>⍴ΔNUMS)⍴0 ◇ →(2|ΔNUMS[I])⍴STI E
[3]    ΔNAMES[I;] ⎕TIE ΔNUMS[I] ◇ I←I+1 ◇ →NFI LE
[4]   STIE: ΔNAMES[I;] ⎕STIE ΔNUMS[I] ◇ I←I+1  ◇ →NFILE
      ∇
```

Thus in the event of an error, not only do we have ')SI', to indicate the point of the error, but also ΔNUMS and ΔNAMES to indicate the state of file ties.

These methods for file ties are obviously equally applicable to TTASKS.

The advent of the BTASK has enabled the user to create a self-perpetuating system. By appropriate manipulation of the request parameters for a BTASK, one BTASK may request the running of the next. An example of this could be a monitoring workspace which is required to run each day, at the end of the working day. By giving the AUTOBREQ function the correct NBTIME and NBDATE parameters, this is easily achieved. Assume the end of the working day is 6 p.m. Specify NBTIME(18:00) to ensure that the task does not run before 6 p.m. To ensure that the task is not run until the next day, we must calculate the date. Generally, we may use ⎕TS, and add one to the day. Hence:

```
      DATE←3⍴⎕TS
      DATE[3]←DATE[3]+1
```

At month end, this may produce some odd dates, for instance

```
      DATE
77 5 32
```

However, the BTASK scheduler is tolerant of these, as long as we are only trying to work one day ahead. A request for 77 5 32 will happily be run on 77 6 1 .

The NBDATE parameter expects the date to be specified as MONTH/DAY/YEAR, and the whole argument to AUTOBREQ must be character. We therefore have to rotate our variable DATE and make it character.

```
      ⍕1⌽DATE
5 32 77
```

We may now write a function that will return a character string of tomorrow's date, using the above statements.

```
      ∇R←TOMORROW;DATE;⎕IO
[1]    ⎕IO←1 ◇ DATE←3⍴⎕TS
[2]    DATE[3]←DATE[3]+1
[3]    R←⍕1⌽DATE
∇
```

The whole argument to AUTOBREQ would then be:

```
      'NBDATE(',TOMORROW,'),NBTIME(18:00):  MO NITOR MONERR 1000 2000'

NBDATE(5 32 77),NBTIME(18:00): MONITOR  MONERR 1000 2000
                        ⍝ IF DATE IS  5 31 77
```

We can now set the following in our monitoring workspace:

```
      ⎕LX←'Δ'
```

and:

```
      ∇Δ
[1]     →(¯1≠I28)ρ0
[2]     AUTOBREQ 'NBDATE(',TOMORROW,'),NBTIME(1 8:00): MONITOR
[3]     MONITORΔFUNCTION
[4]     CLEAROUT                                        MONERR 1000 2000'
∇
```

Thus we have a workspace which we may load to a *TTASK* for maintenance, will request its next run automatically each time it is run, and will leave no saved workspace if it runs without errors.

A similar system may be written to run a workspace on, say, the 2nd of each month. Once again we may write a function to give the character representation of the date one month hence.

The current month:

```
  MONTH←2ρ⎕TS
```

The value of the year is important (76 12 is followed by 77 1), and we must encode the year and month value, add one on and then decode it. A further complication is that encode and decode count fron zero up, while months are counted from 1 - so we have to subtract 1 from the month value. The encoded value of *MONTH* is:

```
  0 12 ⊥ MONTH-1
1627                    ⍝ IF MONTH IS 77 5
```

Add one for the next month, and decode the result, not forgetting that decode also counts from zero up - add 1 to the final result. Hence:

```
  1+0 12 ⊤1+0 12⊥MONTH-1
77 6
```

The function to return the character date for the second day of next month may be written:

```
  ∇R←NEXT2ND;MONTH
[1]     MONTH←2ρ⎕TS
[2]     MONTH←1+0 12⊤1+0 12⊥MONTH-1
[3]     R←⍕1⌽MONTH,2
∇
```

So to run our monthly workspace each month on the 2nd, all we need is:

```
      ⎕LX←'Δ'
```

and

```
      ∇Δ
[1]     →(¯1≠I28)ρ0
[2]     AUTOBREQ 'NBDATE(',NEXT2ND,'): MONTHLY WS MTHERR 500 750'
[3]     MONTHLYΔFUNCTION
[4]     CLEAROUT
∇
```

NEW SHARP APL TECHNICAL NOTES - See Page 11 for SATN INDEX

SATN-23 - COMPARISON TOLERANCE

by Bob Bernecky

SATN-23 is now available. It describes recent changes to the treatment of *⎕CT* in *SHARP APL*. The primitive relational functions now operate in a more reasonable fashion, and certain primitives operate much faster if *⎕CT*=0.

SATN-25 EXTENSIONS TO ARGUMENT PASSING

by Doug Forkes

Two extensions have been made to the way *SHARP APL* passes arguments to functions.

A) A user-defined function whose header indicates it is dyadic may be called monadically. The function begins execution with its left argument undefined.

B) The result-name of a function may match an argument name, in which case the function begins execution with the result initialized to the value (if any) of that argument.

A sample function illustrating both extensions:

```
      ∇R←L DIV R
[1]    ⍝ MIMIC ÷ BUT DOMAIN ERROR FOR 0÷0
[2]    →(0≠4 ⎕WS 'L')/3 ◇ R←÷R ◇ →0
[3]    R←L×÷R
      ∇


      2 DIV 2 4
 1  0.5
      DIV 2 4
0.5  0.25
      0 DIV 0 0

DOMAIN ERROR
DIV[3]    R←L×÷R
             ∧
```

MANAGEMENT STYLE TESTS AND SURVEYS

International Publications Limited, which is owned by Prof. W.J. Reddin, provides surveys that can give a rapid and objective assessment of managerial styles, employee morale and the climate of an organization - say before or after a major change.

Addresses of National representatives may be obtained from:

Tests and Surveys Section,
International Publications Limited,
Box 1022, Bank of Bermuda Building,
Hamilton, Bermuda.

A survey of it readers by the Financial Post Magazine drew 1,224 responses. Replies were mailed out on June 1st. We apologize for incorrectly specifying the publication in the previous issue. The survey was conducted by the Canadian "**Financial Post Magazine**".

APPLICATIONS SOFTWARE LIBRARY - UPDATE

The *SHARP APL* Systems Librarian, **Ed Stubbs**, is responsible for testing and installing new applications software into the public library. Maintenance of existing workspaces, especially to keep all packages up-to-date in terms of enhancements to the *SHARP APL* language, is another major part of the System Librarian's duties. **Jane Chung** has joined Ed - and the list of updates is consequently much longer this month! Grant McDorman has been helping them for the duration of the summer. A summary of updates made over the last two months appears below. Please incorporate the new or updated versions of existing packages into user applications. Contact your Sharp representative for assistance at any time.

NEW WORKSPACES

1	*NETWORK*	Contains Telenet sign-on procedure
5	*FONT*	Symbols corresponding to APL Graphics for terminals without an APL keyboard or typefont.
51	*FPSTOCKLIST*	This workspace prints the Company Codes (used by the Exchanges), and hence by the Financial Post System.
81	*CSNEWS*	News about the CANSIM (Registered Trade Mark of Statistics Canada's machine readable data base) and the *SHARP APL* CANSIM Minibase.
91	*BIGPRINT*	Incorporates former 91 *DOTCHARS* and 91 *BIGCHARS* and the function *LP* (from 91 *FUNSPACE*) to make all large print options available in one workspace.
39	*MAGIC*	This workspace is the same as 702 *JUSTMAGIC* - a general system for access and analysis of time series data bases.

CHANGES TO EXISTING WORKSPACES

1	*LOCKSMITH*	Cleaned up.
1	*FILEPRINT*	Contents of 2 *PRINT∆SIMUL* have been incorporated, with input conventions slightly changed.(See *SIMUHOW*).
7	*WSDOC*	*HSPRINT* replaces old *PRINTREQ/AUTOPREQ* functions.
32	*REGRESSION*	Three-stage least squares now available.
91	*FUNSPACE*	Cleaned up - and the function *LP* has been moved away to 91 *BIGPRINT*.
91	*PICTURES*	*HSPRINT* replaces old *PRINTREQ/AUTOPREQ* functions - and incorporates 91 *DRAW* and 91 *CALENDAR*, both of which were dropped.

WORKSPACES MERGED

NEW		FROM OLD:
7	*FNED*	7 *FNED* + 7 *FNEDHOW*
59	*AGEDAR*	59 *AGEDAR* + 59 *AGEDARHOW*
59	*ANALYZE*	59 *ANALYZE* + 59 *ANALYZEHOW*
59	*BUDGET*	59 *BUDGET* + 59 *BUDGETHOW*
61	*ROARK3*	61 *ROARK31* + 61 *ROARK32*
61	*PLANEFRAME*	61 *PLANEFRAME* + 61 *HOWFNS*
63	*COGO*	63 *COGO* + 63 *COGOHOW*

LIBRARY - UPDATE (continued)

WORKSPACES REMOVED

1	*FILES*	Obsolete since the introduction of "quad" file functions two years ago. (Now in 499 *FILES*1).
2	*PRINT∆SIMUL*	
91	*DRAW*	
91	*BIGCHARS*	
91	*CALENDAR*	
91	*DOTCHARS*	

SHARP APL COURSES

Introduction to APL:	JUN	JUL	AUG	SEP	OCT
EDMONTON (5-day)		18,19,25,26,	→2		
LOS ANGELES*	20-23				
NEW YORK CITY		18-22	15-19		
OTTAWA	06-10	04-08	08-12	12-16	
ROCHESTER	20-24	18-22	15-19	26-30	
SEATTLE			22-24		
TORONTO	20-22	11-13	08-10	06-08	04-06
TORONTO		25-27	22-24	19-21	18-20
U.K. (LONDON) 5-day		6→	10→		
U.K. (LONDON) New format - 3 days				12-14	17-19

Introduction/Intermediate

LOS ANGELES*			AUG 1-4		

Intermediate

LOS ANGELES*		JUL 18-21			
TORONTO			AUG 16-17		

Advanced

LOS ANGELES*			AUG 22-25		
U.K.				SEP 15-16	

Seminars

EDMONTON	"Statistical Analysis in SHARP APL"	July 6
OTTAWA⊛	"Monitoring & Controlling the use of APL"	May 3,5
	"Immedate & Deferred Batch APL"	June 20,21
U.K. (One-day courses arranged on demand)		
	"MAGIC for Time Series Analysis"	
	"Appreciaton of APL"	
	"Introduction to the SHARP APL system"	
TORONTO	"Saving Money (NTASKS and BTASKS"	July 20
	"Efficient Use of Basic APL"	July 20
	"Files and Data Structures"	July 21
	"Principles of Good APL Design"	July 21

APL auf Deutsch

Michael Riedel is teaching APL in Switzerland, Austria and Holland in the German language - courses arranged at your convenience.

* Please ccntact Chris Seri of the Newport Beach office (714) 644-5112 to register.

⊛ Ottawa course notes are available from Alan Daley (613)236-9942.

CANSIM

CANSIM is the registered trade mark for Statistics Canada's machine readable data base. When publishing any data retrieved from CANSIM, the following must be used as the source:
"These data originate from CANSIM which is the registered Trade Mark forStatistics Canada's machine-readable data base."
The *SHARP APL* CANSIM Minibase Supplement is updated regularly, and new series are added. Workspace 81 *CSNEWS* keeps interested users up-to-date with all new developments concerning this database as well as the CANSIM Minibase.

The annual, quarterly and monthly series on May 31, 1977 was as follows:

Matrix No.	Subject
ANNUAL:	
60	Estimated population of Canada by province.
169	Total cash receipts from farming operations. (Alberta).
235	Farm operating expenses and depreciation charges. (Alberta)
271	Income of farm operators from farm operations. (Alberta).
*T*378	Consumer Price Index (Edmonton-Calgary) 1949=100
*T*378	Consumer Price Index (Edmonton-Calgary) 1961=100
431	Consumer Price Index of Canada.
807	Wages and Salaries, by province.
809	Job vacancies, by province.
819	Employment inexes, by province.
838	Production of sawn lumber.
841	New motor vehicles, production, shipments, exports and sales.
844-845	Total electric power & electric power available by province.
846	Coal and Coke statistics
847	Supply and disposition of crude oil & natural gas.
848	Statistics of refined petroleum products.
852	Building permits by province.
853	Dwelling units started.
861	Farm cash receipts.
884	Operating statistics of Canadian pipelines.
1025-1049	Handbook of Agricultural Statistics.
1056	Production & value of honey.
1058	Production & value of greenhouse vegetables.
1136-1186	Handbook of Agricultural Statistics.
1190	Capital expenditures of construction, machinery, and equipment.
1194	Capital & repair expenditures of construction, machinery .....
1328-1359	Handbook of dairy statistics.
1383-1384	Fruit & vegetable reports.
1390,1399	Acreage yield production & value - from 1940.
2076	Basic labour force characteristics (British Columbia).
3305	Cost of fuel & electricity used.
3317	Total value of shipments & other revenue - Canada.
3331,3364, 3366,3454	Flows & stocks of fixed, non-residential capital in Canada.
3722-3723	Total value of construction.
3770-3771	Work performed, by type of structure, new & repair.
7140-7283	Supply and disposal of energy in natural units - Natural Gas Canada.

QUARTERLY

181	Operating statistics of Canadian oil pipeline carriers.
801-803	Immigration to Canada - from 1926.
2325-2328	Canadian balance of International payments.
3186	Total estimates of job vacancies.
3539	Capacity utilization rates in Canadian manufacturing.
3797-3799	Fire & casualty insurance companies.
3857-3859	Property & casualty insurance companies.

MONTHLY:

T198	Consumer Price Indexes 1949=100.
T375	Consumer Price Index (Edmonton-Calgary) 1949=100.
520	Manufacturing shipments by province.
626-645	Refined petroleum products.
671	Industry selling price index - petroleum & coal products.
951	Production of specified chemicals.
1552,1557, 1777	Employment indexes by industry.
1779	Average weekly hours by industry (Vancouver).
1794	New housing price indexes for metropolitain areas.
2671	International travellers entering Canada, by type of transport (B.C.).
2683	Non-resident travellers entering Canada, by country of residence (B.C.).

SATN-INDEX

```
SATN-0   1 JAN 76          SATN INTRODUCTION
SATN-1   1 JAN 76          TASKID
SATN-2   1 FEB 77 REV: 1   CONTROL MESSAGES
SATN-3   1 JAN 76          ⎕OUT
SATN-4  15 JUN 76 REV: 1   NTASKS AND BTASKS
SATN-5  15 JUL 76 REV: 1   BATCH APL
SATN-6   1 JAN 76          EXECUTE
SATN-7   1 JAN 76          LATENT EXPRESSION
SATN-8  15 AUG 76 REV: 1   HSPRINT
SATN-9   1 JUN 76          USAGE INQUIRY SYSTEM
SATN-10  1 JAN 77 REV: 1   SORTREQ
SATN-11  1 JAN 76          )RESET
SATN-12  1 JAN 76          )COPY
SATN-14 15 JAN 77          PACKAGE - A NEW VARIABLE TYPE.
SATN-15 20 APR 76          INDEX
SATN-16 20 APR 76          FILE SYSTEM MUST-WRITE BUFFERS.
SATN-17 15 JUL 76          FORMATTING PRIMITIVE
SATN-18  1 JUL 76          ⎕FMT
SATN-19  1 JAN 77          FILEPRINT
SATN-20 15 JAN 77          SYSTEM VARIABLES AND FUNCTIONS
SATN-21 15 JAN 77          ⎕WS AND ⎕FD
SATN-22 15 JAN 77          APL WORKSPACE TRANSFER
SATN-23  7 JUN 77          COMPARISON TOLERANCE
SATN-24 25 MAR 77          )SYMBOLS
SATN-25 15 MAY 77          EXTENSIONS TO ARGUMENT PASSING
```

I.P. Sharp Associates Limited

Head Office: Suite 1400, 145 King St.West, Toronto,Canada M5H 1J8 (416) 364-5361

Canada — Regional Offices

Calgary
Suite 2660, Scotia Centre,
700 — 2nd St. S.W.,
Calgary, Alberta
T2P 2W2
(403) 265-7730

Edmonton
Suite 505,
10065 Jasper Ave.,
Edmonton, Alberta
T5J 3B1
(403) 428-6744

London
Suite 510,
220 Dundas St.,
London, Ontario
N6A 1H3
(519) 434-2426

Montreal
Suite 1610,
555 Dorchester Blvd. West,
Montreal, Quebec
H2Z 1B1
(514) 866-4981

Ottawa
Suite 600,
265 Carling Ave.,
Ottawa, Ontario
K1S 2E1
(613) 236-9942

Vancouver
Suite 604,
1112 West Pender St.,
Vancouver, B.C.
V6E 2S1
(604) 682-7158

Victoria
870 Roy Rd.,
Victoria, B.C.
V8Z 2X3
(604) 479-1611

Winnipeg
Suite 909,
213 Notre Dame Ave.,
Winnipeg, Manitoba
R3B 1N3
(204) 947-1241

United Kingdom
I.P. Sharp Associates Limited

Birmingham
2nd Floor,
Radio House,
79/81 Aston Rd. North,
Birmingham B6 4BX
England
021-359-6964

Gloucester
29 Northgate St.,
Gloucester
0452 28106

London
132 Buckingham Palace Rd.,
London SW1W 9SA
(01) 730-0361

London
118 - 119 Piccadilly,
Mayfair, London W1V 9FJ
England
(01) 629-1564

Warrington
48A Horsemarket St.,
Warrington, Lancashire
England
W 55342

Europe

Intersystems, B.V.
Herengracht 244,
Amsterdam 1002,
The Netherlands
(020) 244050

I.P. Sharp Europe S.A.
Boite 1
191 - 7 Boulevarde de Souverain,
Bruxelles, Belgium
(022) 660-2900

I.P. Sharp GmbH
Leostrasse 62A,
4000 Düsseldorf 11,
West Germany
(0211) 57 50 16

I.P. Sharp A.G.
Badenerstrasse 141
8004 Zürich
Switzerland
241-52-42

I.P. Sharp ApS
Østergade 24B,
1100 Copenhagen K
Denmark
112 434 and 112 435

I.P. Sharp AB
Kungsgatan 65,
S111 22 Stockholm, Sweden
(08) 21 10 19

U.S.A.
I.P. Sharp Associates, Inc.

Boston
Suite 812,
148 State St.,
Boston, Mass. 02109
(617) 523-2506

Chicago
Suite 424,
8501 West Higgins Rd.,
Chicago, Ill. 60631
(312) 693-5895

Dallas
Suite 1148,
Campbell Centre,
8350 Northcentral Expressway,
Dallas, Texas 75206
(214) 369-1131

Houston
One Corporate Square,
Suite 405,
2600 Southwest Freeway,
Houston, Texas 77098
(713) 526-5275

Minneapolis
Suite 1371,
1 Appletree Square,
Bloomington, Minn. 55420
(612) 854-3405

New York City
Suite 250, East Mezz.,
Pan Am Bldg.,
New York, N.Y. 10017
(212) 986-3366

Newport Beach
Suite 1135,
610 Newport Centre Drive,
Newport Beach, Ca. 92660
(714) 644-5112

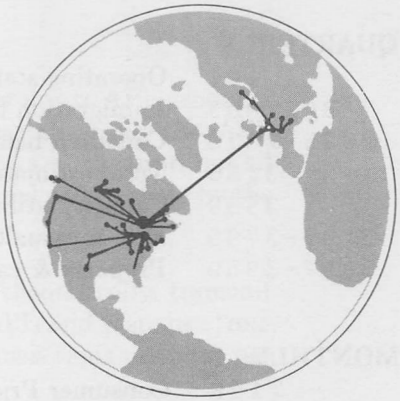
Palo Alto
Suite 110,
299 California Ave.,
Palo Alto, Ca. 94306
(415) 327-1700

Rochester
Suite 1150,
183 Main Street East,
Rochester, N.Y. 14604
(716) 546-7270

San Francisco
Suite C409,
900 North Point Street,
San Francisco, Ca. 94109
(415) 673-4930

Seattle
Suite 217,
Executive Plaza East,
12835 Bellevue-Redmond Rd.,
Bellevue, Wa. 98005
(206) 453-1661

Washington, D.C.
Suite 307,
1730 K St. N.W.,
Washington, D.C. 20006
(202) 293-2915



APL Operator (416) 363-2051

SHARP APL Local Access In:

Canada
Calgary
Edmonton
Halifax
Hamilton
Kitchener
London
Montreal
Ottawa
Quebec City
Regina
Saskatoon
Toronto
Vancouver
Victoria
Winnipeg

U.S.A.
Ann Arbor
Boston
Buffalo
Chicago
Dallas
Des Moines
Houston
Los Angeles
Minneapolis
Newport Beach
New York City
Rochester, N.Y.
San Francisco
Seattle
Stamford, Ct.
Syracuse, N.Y.
Washington, D.C.
White Plains

Telenet
Albany
Atlanta
Baltimore
Cincinnati
Cleveland
Columbus
Denver
Detroit
Hartford, Ct.
Hempstead
Indianapolis
Kansas City
Miami
Milwaukee
Newark
New Haven/Bridgeport, Ct.
New Orleans
Philadelphia
Phoenix
Pittsburgh
Portland
Sacramento
St. Louis
Salt Lake City
San Carlos
San Diego
San Jose

Europe
Amsterdam
Bruxelles
Copenhagen
Coventry
Düsseldorf
Gloucester
Liverpool
London
Manchester
Milan
Paris
Stockholm
Zürich

☐ Please amend my mailing address as indicated.
☐ Add to your mailing list the following name(s).
☐ Send me a SHARP APL publications order form.
☐ Note my comments: _____

Name: _____
Co.: _____
Address: _____

The Newsletter is a regular publication of I.P. Sharp Associates Limited. Contributions and comments are welcomed and should be addressed to: Jeanne Gershater, I.P. Sharp Newsletter, Suite 1400, York Centre, 145 King Street West, Toronto, Ontario M5H 1J8.