STRIDE

January 1986 — Vol II, Issue 1

CONTENTS:

tech notes

The Stride 400 Series

From $2995 to $60,000+

As you can see below, the Stride 400 Series has no shortage of features. We've recently added RAM expansion to 12M bytes, low-cost high-speed graphics and a revolutionary hands-free NOD™ cursor control device. But there's more to Stride Micro than hardware. We're an open company, sharing source programs and schematics with our users. That's the reason Stride and Sage (our former name) machines are preferred by many leading p-System developers. It's what we call "Performance by Design".™

Stride 400 Series Technical Specifications

- *12 MHz 68000 CPU standard*
- *VMEbus interface*
- *1M bytes of parity RAM standard Up to 12M bytes*
- *5¼" 640K byte floppy disk drives*
- *20M to 448M bytes hard disk storage*
- *Battery backed-up real-time clock*
- *4K bytes of battery backed-up CMOS RAM*
- *4 to 22 RS-232C serial ports*
- *Omninet networking hardware (opt.)*
- *p-System IV.21 w/LAN software*
- *High speed low-cost monochrome graphics (784 x 325 resolution)*
- *NOD cursor control device*
- *FPU hardware floating point*
- *Streaming ¼" tape drive backup*
- *MMU Memory management unit*
- *Stride configuration utilities*
- *TeleTalker communications software*
- *Centronics compatible parallel port*

The Stride Multiuser p-SYSTEM

Stride provides a complete multiuser system **standard** with every machine. This unique system allows multiple p-System (and CP/M-68K) operating systems to co-exist at the same time on a single machine. An easy-to-use beginner's program (MU.BUILD) will generate a multiuser system within minutes. Sophisticated users will appreciate the highly powerful and flexible MU.UTIL program which allows an incredible level of customization of the multiuser system. It is a multitasking, propriety BIOS developed in 1982 (the first commercial multiuser p-System), and has grown in strength and flexibility ever since. Stride also offers other popular multiuser operating systems such as RM/COS and UNIX.

STRIDE MICRO

It's In The Mail!

by Verlene Joyce Bonham

I'm sure you've heard *that* before! An overused excuse to delay yet another day and shift all the blame to the U.S. postal system. But the dodge won't work when applied to electronic mail or **email** as it is called. A note sent via the phone to Compuserve, for example, is there in minutes! The recipient has only to dial in and pick it up with his computer system. The *speed* of **email** is one of the major reasons for the phenomenal growth of the industry in the last few years. Why send a letter via Federal Express for $15 when a $1.50 **email** message gets there in less than 10 minutes?

Stride has used **email** more and more lately for customer support and international communications. Our phone lines are always humming. For example, the listing of the last six weeks of MUSUS (a p–System bulletin board) weighs over 14 pounds -- and it's printed in very small type on the laser printer. Traffic on the UNIX **net.news** in just 2 weeks totaled 3653 articles, or 6.686991 Mbytes of text. This originated from 661 different Usenet sites by 1469 different users to 197 newsgroups. It's easy to get behind on reading that!

In addition to sending fast, private business letters, a popular (and fun!) part of the **email** world are the **bulletin boards** or **BB**s. They work much like the bulletin board posted in the lunchroom of your company. Rather than send your message to one person, you "post" it to the BB and any member of the BB group can read it. Generally, each BB has a particular area of interest the members discuss. Some limit membership, most don't.

There are BBs for almost any topic: space, music, games, travel, health, kids, cars, movies, politics, sex and serious databases on finance, stocks, unusual diseases, and, of course, there are BBs on almost any language or operating system in the computer world. And it is inexpensive.

Sorry, guys. Don't tell me how poor you are. Email *is* cheap. Think of how many times you've made a phone call and he/she wasn't there. With **email**, you send the message and know that it at least got to its destination.

How much time have you spent trying to describe a computer operation verbally when a couple of lines of "TYPE THIS, SEE THAT" would solve the problem? Or how easy it would all be if you could just send him a little 20 line program!

How many times has your name or an important word been misspelled? (Valley becomes Sally and fox becomes socks, etc.)

It may be worth it to take another look at your company's phone and shipping costs and then price **email** services.

Modems are also reasonably priced. A simple 300 baud modem can be purchased for around $100. But you should try for the higher priced, higher speed 1200 baud or even 2400/1200 baud models with such goodies as auto–answer, auto–dial, and both synchronous and asynchronous modes. Auto–answer is a must if you want to remotely dial–in to your Stride, and the Hayes command set has become a *de facto* modem standard.

If you're not sure what it's all about, give it a test drive. You can pick up a Compuserve Starter Kit from you local RadioShack or computer store for about $25. Or order it for about $40 from Black Box Corporation, P.O. Box 12800, Pittsburgh, PA 15421 (order phone (412)–746–5530. There is also a starter kit for the Source from Black Box for about $48. For MCI Mail call 800–MCI–2255 or write 2000 M Street, NW Washington, DC 20036. Their startup "mailbox fee" is $18.

The starter kits include several hours of time on the air, just enough time to poke around and learn what services are offered. Individuals generally pay with a credit card. Company accounts are available.

The UNIX net.news doesn't cost anything except short haul phone time. But sometimes it is hard to find a site close to you that is willing to pass along the news to you.

Once you've found an interesting BB, you may get "stage fright". After all, what do you say to people you've never met face–to–face? Start just by browsing through the messages. Silence is a virtue until you get a feel for the "personality" of a specific board.

It does help to polish your messages a little and take time to say exactly what you mean. This is especially important if you are a commercial user sending business messages. Cold, naked text needs help to get your feelings across and to make sure you are not misunderstood.

Chuq Von Rospach, in an article on **net.news,** gives some good tips on etiquette for **email,** "*Be Careful with Humor and Sarcasm: Without the voice inflections and body language of personal communications, it is easy for a remark meant to be funny to be misinterpreted. Subtle humor tends to get lost, so take steps to make sure that people realize you are trying to be funny. The net has developed a symbol called the smiley face. It looks like this:*

" :–) "

and points out sections of articles with humorous intent." To see the face, tilt your head to the left and look at it that way.

Maybe you've got a better idea of the fascination of this new media now. If so, and you join a service, let Stride know. Most of the gang (mitch, buddy, dunlap, vjb, bill, davee, ...) can be reached via the "mail." And the next time someone tells you "it's in the mail" maybe it really will be! " :–) " □

IREX – Real Time For The p–System

by Stephen Buchalter

Users of minicomputers for real–time applications, notably the Hewlett–Packard HP1000 and the DEC PDP–series, traditionally have been unable to use microcomputers for their tasks. The difference in hardware is not always the reason for this. Although the modern supermicro is challenging the traditional mini in computational power, their operating systems (UNIX, PC–DOS, etc.) lack real–time facilities such as:

- *Exact control of task priorities in a multi–tasking environment. This assures time–critical applications of processor time.*
- *Time–based scheduling of tasks. This enables the execution of a task at a future time, either once or iteratively.*
- *Configuring critical tasks in memory for fast response.*
- *Provision for synchronization between tasks.*

All of these facilities, and more, may be found in a newly developed product called IREX. IREX is an interactive real–time executive specifically developed for the p–System. This operating system was developed at UCSD, the University of California at San Diego, and features a *de facto* standard Pascal for microcomputers.

Some 300,000 micros run the p–System. A key feature of its Pascal *vis a vis* other implementations is its ability to have asynchronously executing tasks within the same program.

IREX is a highly portable package that is hosted by the p–System and runs on such diverse processors as the Intel 286 of the IBM AT and the Motorola 68000 of Stride's 400 Series.

In order to function, IREX interacts directly with the p–System kernel. The additional requirements imposed by real–time considerations are met by maintaining control blocks for each task in parallel with those maintained by the operating system.

IREX is suitable for real–time applications such as data–acquisition, process control and even business applications where a task can be scheduled to print a lengthy report at midnight when nobody else needs the printer.

The package has a novel way of presenting to the user its many features. It is menu–driven in the modern user–friendly style brought about by the micro revolution. An on–line help facility is included in case the user has forgotten the format for an interactive schedule starting at some future time.

The status of a processes (task) may be inspected. IREX also incorporates hierarchical access levels with password protection. In a process control environment, for example, it can be made impossible for an operator to deschedule a critical watchdog timer task.

Asynchronously executing tasks may generate alarm messages which are buffered by the system. The existence of such messages is indicated by a real–time status indicator. The messages may be inspected on demand. Tasks may queue for the system console and the operator is made aware of their presence. The console may be allocated interactively to waiting tasks at the operator's initiative.

The package allows the adjustment of priorities on–line, something some minicomputer systems do not allow!

In addition, the elapsed time of scheduled processes as well as the CPU utilization of the host computer may be measured. This allows exact tailoring of the micro to its application.

With IREX, independent programs may be entered in a library. Only programs contained in the library may be executed. Such programs execute concurrently with scheduled tasks.

Programs may be executed by using a single keystroke. The application program may be compiled with one priority and executed at a different priority. The system editor, for instance, may be made to execute at a higher priority to provide crisp response to keyboard input if the other tasks in the environment consume an appreciable amount of processor time.

Apart from its interactive interface, IREX provides full access to its functions at application code level. As in an ADA environment, a program is called via IREX–coupled tasks through common separately compiled modules or units. The traditional mini system–common region and associated trap calls have been elegantly dispensed with.

IREX facilities are grouped in different units. Sleep calls, interrupt timeouts and concurrent console read/write facilities are all included in the package. Tasks may write to the screen without upsetting concurrent read operations. Task output is organized on a per–task basis rather than restrictive "window" areas.

IREX brings a seamless integrated environment to real–time applications. It does for real–time what the modern micro database package did for data processing. The operator is given a highly understandable, easily manipulated application interface which still retains tight security.□

Ed.Note: IREX is a product of MBM Computers, 4 Claydon House, Holdershil Road, London NW4 ILS, Tel:(441) 2081659 TWX:261507 attention Ref 2658

CMX – A Matrix Expression Calculator

by William L.Brogan

CMX is an extremely useful program for *interactively* doing most common matrix calculations. You need not know any programming to use it and it is a great way to learn matrix math.

The command syntax used is very simple. Matrix arrays are named with a single capital letter, A–Z, where I, Z, and X are reserved for special matrices. I is the name of the unit matrix, Z is the zero matrix and X is reserved for the "unknowns" array in the solution of simultaneous equations.

Operators

Single keystrokes + – * / indicate the common **algebraic** operators of addition subtraction, multiplication and inversion. The inverse only applies to square matrices. If used, the determinant is also calculated and displayed. The . (dot) multiplies a matrix by a scaler and the ' (quote) transposes a matrix. The ? operator solves simultaneous equations, even in the non– square case. If there are fewer equations than unknowns, *all* of the non–unique solutions are found. The rank of the array being operated upon is also shown.

The = **display** operator displays the array currently on the stack and saves it in a new array name. The) operator displays the current results but does not save them. In addition the ? and / operators always display results.

The **construction** operators are useful for working with partitioned matrices. & appends one array to another (side–by–side) to form a larger array.

The % operator appends one array to another (one above the other) to form a larger array.

The # operator forms the Kronnecker product of two arrays, i.e. **A#B** forms a larger array by multiplying each element a(i,j) in A by the entire B array. @ forms a smaller array by selecting a specified subsegment of a larger array.

The ! ^ < $ **program control** operators change the CMX mode. ! returns to the help screen. ^ returns to the data input screen. < is used when you want to abort an incorrect sequence of operations and start from the beginning.

$ toggles between a fully prompted mode of operation and a silent mode. In the prompted mode, the response to each prompt is either a single capital letter array name or one of the above operators, followed by a carriage return.

In the silent mode any meaningful combination of up to 72 characters (matrix names or operators) can be strung together in a single command string. The order of operations is strictly left to right. As illustrated in the examples that follow, this sometimes requires that intermediate results be saved via the = operator.

Data Entry

Helpful explicit error messages are provided. For example: what happens if you asked for the sum of two matrices of different sizes? This violates the conformability rules of matrix algebra and CMX will notify you of the problem.

The following two examples demonstrate the power of CMX and its simple command syntax.

Example #1: Using three suitably sized matrices: P, H and Q, verify the well–known matrix inversion lemma.

$$[P^{-1}+H^{T}QH]^{-1}=P-PH^{T}[HPH^{T}+Q^{-1}]^{-1}HP$$

Using the silent mode of CMX, the left side of this identity could be evaluated using the following four command strings:

```
H'=J  P/=S  Q/=U  J*Q*H+S/=L
```

J is defined as the transpose of H, and the inverses of P and Q are formed. The array L contains the left side of the given expression. To evaluate the right side R we need to evaluate one more intermediate array which is called T.

```
H*P*J+U^*H*P=T  P*J*T -1.-P=R
```

Note the multiplication by the scaler –1. A space or comma separator must be used after any numeric data in a string command. The matrix inversion lemma says that the arrays L and R are the same, and that they are within the single precision accuracy of CMX.

Example #2: For a previously defined array A and a compatible column Y, solve AX=Y for the unknown column X.

The command A?Y will find the result and place it in X, unless the equations are inconsistent. You could also use A/*Y=R. If A is square and invertible, R would give the same answer. In the first method, A need not be square.

Conclusion

CMX provides the well–known power of matrices as a tool for solving many types of problems. With its easy–to–use features, CMX also becomes a good method for learning matrix algebra, especially in conjunction with a text book on matrices.

CMX is priced at $55 and runs under CP/M–68K on the Stride 400 Series and Sage machines. For more information on CMX and other engineering software, write or call Estanco, Inc., 444 So. Cotner Blvd., Lincoln, Nebraska 68510. Tel: (402) 489–6435

Ed.Note: Dr. William L. Brogan is the author of "Modern Control Theory" Prentice–Hall, 1985. The examples in this article were taken from his book; specifically more information on Example #1 can be found on page 78 of that book.

EPA: an Overview of Essential Pascal Aids

by Jai Gopal Singh Khalsa

Essential Pascal Aids (EPA) is a treasury of UCSD Pascal programs and programming tools. Non-programmers also may be interested in the many major utilities in the package. The following programs are generally useful at almost any p-system site.

COMLINK

This is a telecommunications program for accessing electronic mail services such as CompuServe, Telemail, BIX, MCI Mail, etc. It supports auto-dialing with Hayes-compatible modems using a list of telephone numbers limited only by available disk space. Things such as baud rate, parity, data bits, linefeed after carriage-return, pulse or tone dialing, line-at-a-time handshake character, etc., may be configured separately for each telephone number. These parameters will be installed automatically when the number is selected.

Text files may be sent to remote computers and, of course, an entire on-line session may be captured and saved to a text file on your machine.

Files to be sent may be selected from alphabetical volume directory listings from within **COMLINK.**

DIR

This is a screen-oriented Filer-like utility with a number of convenient enhancements over the standard Filer. The V)olumes listing shows all mounted volumes with the number of files, blocks used, blocks unused, and the largest unused space on each volume. You can page back and forth through this listing and S)elect a volume to examine by moving an arrow with the standard cursor keys. A complete volume directory listing (also available by typing a volume name with optional wild card) appears in two columns on the screen.

Files may be sorted by Name (default), Date, Size, Type, or not sorted at all. Again, cursor keys are used to move an arrow with options to C)hange file name (or size!) by simply typing over the existing name, V)iew or E)dit text files, X)ecute code files, or S)elect/toggle any number of files. An asterisk (*) appears next to each selected file and options are offered to P)rint the selected file names, T)ransfer the selected files to another volume and/or R)emove the selected files.

Tests run at MicroStrategies indicate that multiple file transfers between disk volumes using **DIR** are faster than the standard p-System Filer.

Selected files may be transferred to serial volumes in which case a formfeed will be inserted after each file (default) or, optionally, a pause or nothing. It is extremely convenient when 'cleaning up' your disk to be able to V)iew text files and return immediately to the directory listing to (possibly) S)elect them for removal.

The F)ind option in **DIR** accepts a wildcard and searches all volumes for the specified file(s); this is particularly handy when you have 20+ volumes on a hard disk.

Other options in **DIR** will C)ompare two files, K)runch a volume, and set the P)refix and D)ate.

P-BACKUP

This utility allows hard disk volumes and files that are larger than a single floppy disk to be automatically broken into floppy-sized pieces and saved and restored. The program prompts for the appropriate floppies when needed. Of course, it is also excellent for backing up small files too!

Up to twelve (12) redundant copies of any file or volume may be maintained using **P-BACKUP.** Copies are stored and listed in the order in which they are saved. Entries may be deleted quickly without having to insert each floppy that holds a piece of the file being deleted. Files and volumes may be selected in a manner identical to that described in the **DIR** utility above.

In addition, a text file list of file names may be used to drive the backup process. Such a list may be created with the **DIR** utility or, better yet, the E)verything option in **P-BACKUP** may be used. This option scans all on-line volumes and checks each file against the **P-BACKUP** index. If the file date has changed since the last backup (or there is no record of a previous backup), the filename is printed to the text file list. The list may be edited, of course, before instructing **P-BACKUP** to proceed with the backup.

To speed the process further, an **oldest** option may be specified to automatically remove the oldest prior backup of a file before backing up the current version. For safety, this works only if there are two or more previous backups of the file/volume. An alphabetical log of all backups may be printed or viewed at any time.

PRINT

This is a modest but capable text formatting program provided in source code form. It supports top and bottom margins, page numbering with prefix and suffix, auto-page break at blank lines depending on a user-specified minimum number of lines left, page headings, boldface, underline, change pitch, etc. It may be enhanced as needed by the moderately experienced programmer.

RESCUE

Have you ever accidentally removed an important text file or lost an entire volume directory due to a hardware (or neural) glitch? **RESCUE** ignores the directory and scans the actual disk, looking for contiguous blocks of legal text. When it finds some, it indicates the beginning block number, the number

of contiguous blocks, and allows the user to P)age through the 'file' and J)ump to the E)nd or B)eginning to examine the text that has been found. If it's what you're looking for (harrah!), use the S)ave option to transfer the text to another volume.

Programming Tools:

For the programmer, EPA offers a rich set of UCSD Pascal library units based on thousands of hours (and 50,000+ source lines) of programming experience. For both the expert and novice, these tools will greatly speed the development of casual or sophisticated programs. The library units include:

EpaKernel

This unit provides a number of KERNEL-dependent procedures and functions such as reading and setting the system date, setting the prefix volume, getting information about an open file (name, length, volNum), and getting information about volumes (isBlocked and volName given volNum or volNum given volName). Not only does it relieve the need for the standard SysInfo and FileInfo units (and is much smaller) but, thanks to Stephen Pickett of REC Software, it works on *both* 4.13 and 4.21 versions of SYSTEM.PASCAL!

CrtUnit

CrtUnit provides many basic and commonly needed functions such as GetChar, UnGetChar, GetString, GetInteger, StrToInt, fillField, stripEnds, padLeft, padRight, AllUpperCase, scrnTitle, menu and promptline functions, and a very handy **PrintWhere** procedure. This makes it very easy to write terminal-independent programs using the standard ScreenOps unit.

EpaMasks

The utility **MASKLIB.CODE** allows screen and print masks to be compactly stored and viewed. A mask is nothing more than a number of lines of text created in the standard editor, usually to represent some kind of form or help screen. The **EpaMasks** unit is used to retrieve the masks in application programs, either for display on the screen or one line at a time for printing. An **emPutField** procedure greatly simplifies placement of string 'fields' within other strings.

UpiUnit & UpiLib.CODE

The **UpiUnit** unit allows application programs to have a 'Universal Printer Interface'. Utility **UPILIB.CODE** uses **UpiUnit** to store definitions of printer control sequences in a library file to be easily accessed via the unit. Commands include changing pitch, underline and boldface, and four on/off pair of 'misc' commands. Though not intended as a 'ConsoleOps', it works well for storing terminal commands such as 'Pass-Thru-Printer-Port' and 'inverse video'.

Int32Ops

A complete set of 68000 assembly operations to support *fast* 32-bit integers (implemented and donated to the USUS library by Tom Cattrall).

EpaDates

This unit contains routines such as dsGetDate,dsDayOfWeek, dsDaysPer-Month, dsDaysBetween and dsLater (a date comparison routine) plus dsJulianDate, dsDateMath, dsDateToStr, and dsStrToDate.

ocUnit

ocUnit is intended to be a complete, intelligent module for handling printer-type output. This output may go to the console or to a file. However, the formatting characteristics remain essentially identical (i.e., page numbering, line count, top-of-form, etc.).

It is especially useful when printing reports that consist of columns of data as placement of data within the columns, justification, centering on the page, and re-ordering of the columns is very easy. Another good reason to use this unit for all printed output is that a serial printer attached to the printer port of a terminal is easily (transparently) supported.

EpaTime

EpaTime provides tuEnterTime, tuRead-Time, tuSetTime, and tuGetTime for convenience in accessing the system clock.

ArrowMenu

This unit provides a mouse-like ability to point to items on the screen using the standard cursor keys and select them for further processing by an application program. The items are arranged on the screen in one or more columns automatically by the unit. An arrow (-->) is placed pointing to an item on the screen by the amMoveArrow function. Depending on the keyboard input, the value of arrowPos may be changed OR the item may be selected OR the next/previous page of items may be displayed. It has been found that such direct interaction with the screen is often far more effective and less confusing than single-key menu selection, especially when the number of choices is large.

FLock

This unit is designed to provide Read and Write file-locking capability to cooperating application programs on a multi-user Stride/Sage computer. The key word here is *co-operating*. The unit does nothing to actually prevent files from being accessed by other users. The application programs must consistently use the unit and successfully lock a file before operating on it in order to insure data integrity in the multi-user environment. Also, source code is provided.

DirOps

This unit allows complete access to and manipulation of volume directories. It is significantly more powerful than the standard DirInfo

unit as it allows any or all attributes of a file (i.e., name, size, date, type, lastByte) to be changed. Files may be removed from the directory or created, supplying any or all of the file attributes. Unlike the operating system, a *smallest–space–that's–large–enough* algorithm is used for new files.

DirOps makes the directory appear to be a sequential file and can sort the records (i.e., files) by Name, Date, Size, or Type. Through the use of an opaque type, more than one directory at a time may be open, allowing for very efficient programming of things like multi–file transfers. Source code for a **VolStuff** unit is provided that demonstrates the use of DirOps.

DiskIO

Four major improvements over conventional Pascal file I/O are offered by this unit:

1) GET and PUT to more than MaxInt record positions. A 32–bit integer may be used as a SEEK parameter.

2) **Subfiles** – Block–ranges within a conventional file may be treated as files themselves with different record sizes and a 'record zero' for each subfile. Though each subfile has a separate **diFIB,** they all share a common UCSD FIB. Subfiles allow logically related data of different types to share a single UCSD file, minimizing both the burden on UCSD's 77 files–per–volume limit and the potential for data inconsistency when using separate files.

3) Dynamic file allocation is handled transparently by the unit, allowing linked lists or arrays of files allocated at runtime. Also, total data space required by FIBs is substantially reduced because the shared UCSD FIBs are untyped and unbuffered and therefore much smaller than FIBs generated by the compiler for a FILE OF SOMETHING.

4) Data of arbitrary type and size may be read/written to arbitrary locations in the (sub)file using 'startBlk/startByte' parameters.

HashFile

Hashing is a technique for mapping a very large number of possible data values onto a relatively small address space. For example, a terms file might have a key field of STRING[5] and values such as 'A001', 'B2234', 'K1', etc. The number of possible values of the key field is extremely large and yet the application may use at most a few hundred (or thousand) terms.

The hashing technique transforms the key field to a record number between 1 and maxRecs and handles *collisions* if two or more values happen to map to the same location.

If the algorithm is good, the number of attempts required to find a particular record (or an empty location) is quite low (and fast), regardless of the total capacity of the file.

The EPA HashFile unit will handle keys of type STRING, Integer32, or conventional (16–bit) integer. Also, it supports true removal of hash records without degrading access performance.

Conclusion:

Essential Pascal Aids is being constantly maintained and enhanced based on the "real programming" needs of a large application. New library units that are currently in the 'alpha' stage will be added to EPA as they mature.

If you are thinking about writing programs, starting with a package of tools like this will likely give you a big productive and creative boost.

For the non–programmer, the **COMLINK, DIR, P–BACKUP, PRINT,** and **RESCUE** utilities are ample reasons for investing in the package. □

Ed.Note: EPA is available directly from MicroStrategies, 350 Village St. Millis, MA 02054, Tel: (617) 376–4722 or from Stride for $125 plus shipping.

STRIDE FAIRE '86 INFORMATION

Make your plans now for Stride Faire! February 28 through March 2 is your chance to talk to the team at Stride and many of the vendors, programmers, and users in Stride's community. You need to do just three things to attend:

Faire Registration

Call Stride Reno (702) 322–6868 for a registration package. Entrance fees are only $25.

Hotel Rooms

Call the Nugget Hotel/Casino for accomodations. (800) 648–1177 or (702) 356–3355. Special rates for the Faire start at $57.78 per night, half what you would pay for equivalent rooms in most cities.

Transportation

Call McGinley World Travel for special discount air fares at (800)–553–5366 or (702) 329–6218.

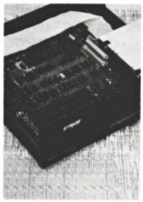
The Nugget runs a courtesy shuttle to and from the Reno airport at 5:50 AM, 6:20, 7:00 and then every half hour until 11:00 PM. At other times call 356–3300 and you can arrange for the shuttle to pick you up. The Nugget is less than 10 minutes from the airport.

Early Birds

If you're an early bird at the Faire, coming in Thursday night, wander up the Nugget escalator to the waiting area in the conference room. Maybe you'll see someone to chat with. No official events are planned for Thursday night, but you might want to get in early to make sure you don't miss Stride's main opening speech Friday morning. See you there! □

STRIDE '86 Faire

Hardware, software, skiing, food and gaming — all the action's at **Stride Faire '86.** Come to Reno and find out what's new with the Stride Micro community. And every paid attendee ($25) has a chance to win a Stride 440 with a terminal. Attendees can get up to 70% discounts on air fair. Plan to attend now!



Fri., Sat., Sun.
FEB. 28 — MAR. 2, 1986
RENO, NEVADA — USA

For more information; call Stride Faire (702) 322-6868 or write:

Stride Micro, formerly Sage Computers builds 68000 micro computers.

STRIDE MICRO

P.O. Box 30016, Reno, NV 89520-0016

Here's your chance to learn Modula-2!

MODULA-2 For C Programmers

An introduction to the strengths and weaknesses of Modula-2 from a C viewpoint.

A full day seminar taught by
Dr. Richard Wiener and Claude Wiatrowski

WHEN: Stride Faire, March 3, 1986 (Monday)
WHERE: Nugget Convention Center, Reno, Nevada
COST: $200 Contact: Stride Faire (702) 322-6868

UNIX Q&A

How do I access the Stride BIOS under UNIX?

You don't. UNIX does not allow direct user access to the BIOS. However, it does provide a great variety of system calls which will allow you to do raw I/O in a safe and easy manner. If you wish to talk to a device you've installed yourself, you must write and install a driver.

Can I copy a p-System floppy under UNIX?

Yes, the **dd** command will copy the #4: (rfd0) diskette to #5: (rfd1) as long as the diskette is in 1280 block format.

```
dd if=/dev/rfd0  of=/dev/rfd1  bs=320k
```

If you have lots of memory, you can increase the buffer size (bs) to a max of 640k (the size of the floppy).

Also, it will duplicate a UNIX floppy, of course. If you do this very often, and you are running **csh,** you may want to "alias" it in your **.cshrc** file.

If you have only one floppy disk drive, you need to copy the diskette to a file, then copy the file to the new diskette. The commands for this are:

```
dd if=/dev/rfd0  of=<filename>  bs=320k
dd if=<filename>  of=/dev/rfd0  bs=320k
```

How do I format a floppy under UNIX?

Use the **format** program. It uses the same format as that of the p-System formatter under UTIL. Insert the diskette, then use one of these statements:

```
format fd0  (left floppy)
format fd1  (right floppy)
```

Can I change the date while in multiuser UNIX?

Yes, but you may be surprised if you do. The **cron** routine is a system background job, or **daemon,** and can get very busy and burden the system if you change the time on the fly.

Here's the preferred way. On the system console, login as **root** (or **su** for superuser) and type **telinit s** to drop the system into single user mode. Then type **killall** which halts **cron,** and other processes. Then do the date/time change using **date.** Type **halt-q** to halt the system, then reboot to the multiuser system, which starts **cron** again.

What do you mean by multiuser UNIX? I thought UNIX was always multiuser?

Yes, it is. The single user mode is basically a maintenance mode (see the question on changing the date above). In the single user mode, only the kernel processes (**swapper, init, xsched**) plus a user shell (**root**) are running. The multiuser mode brings in many other processes to support additional users.

My UNIX machine needs a name-- any rules on how to name it?

UNIX machines generally have colorful and descriptive names depending on the whimsy of the owners. UNIX machines frequently talk to each other with **uucp** logins, the "mail," so you need a name that is unique.

You should not choose a common first name such as Jack, Sally, etc. Use all lower case. Can others easily remember, spell and pronounce it? You might settle for your company's name if it is unique. A short name of 5-7 characters is recommended if you plan to be a **uucp** site. Otherwise, a maximum of nine characters is allowed.

Avoid special characters in the name, as many have meanings to the mail systems protocol. Don't use dashes: there is a current bug in the UNIX accounting program **wtmpfix** which doesn't like dashes.

How do I use my modem with a Stride UNIX machine?

First set the Hayes dip switches: 1up 2up 3down 4up 5up 6up 7up 8down. Then check that your modem cable (Part number CC00136B or CC00137B) has a pin in position 20 and not position 4. If necessary, move the pin from 4 to 20. Connect the modem and turn it on. Generally, you leave the modem powered up at all times.

***

Stride's UNIX is somewhat unique in that it supports both dial-out and dial-in modes at the same time on the same modem!

***

The dial-in mode allows you to login from a remote location. To install dial-in, make an entry in **/etc/inittab** such as

```
d?:2:respawn:/etc/getty  ttyd?
```

where ? is the number (0-F Hex) of the serial port connected to the modem.

To allow dialing out, your **/etc/remote** file should be changed to look like this:

```
#
# General dialer definitions used below
#
dial1200|1200 Baud Able \
   Quadracall attributes:\
     :dv=/dev/ttyh1,/dev/ttyh2,/dev/ttyh?:\
        :br#1200:at=hayes:du:dt#30:
dial300|300 Bizcomp 1022 attributes:\
     :dv=/dev/ttyh1,/dev/ttyh2,/dev/ttyh?:\
        :br#300:at=hayes:du:
```

where the ? in **/dev/ttyh?** is the number (0-F Hex) of the modem port. Once this is done, you can dial out using routines such as **tip.** □

Starting Out In "csh"

by Bill Rainey

One of the extra features of Stride's UNIX release is support for **csh,** the Berkeley **C shell.** This shell has a number of powerful features not available to the standard **Bourne shell** program **sh.**

If you want to change your account to use **csh,** first get your system manager to specify the **csh** option in the **/etc/passwd** file for you. Then you're ready to "tune" up your account. Here are some ideas on where to start.

Each **csh** user has a set of files in his home directory that start with dots. These files provide a way to custom tailor UNIX just for that user. The three main ones are:

```
.login .logout .cshrc
```

These files are **csh** script files, you should review the **csh(1)** section of the *on–line* manual.

.login

The commands in the **.login** file are executed after you give your login name and passwd. Below is an example of a **.login** file.

umask masks the permissions of each file created by the user. When set at 02, other users have read and execute permission, but not write. Reference **chmod(1)**and **sh(1).**

The **set path** line defines the directories searched and the order they are searched in for an executable program. Without this, you would have to type the full pathname of the program to be executed.

stty has many options for setting terminal parameters, refer to **stty(1).** Here, the backspace character is defined as the ERASE key, the program interrupt key is Ctrl/C and the kill key (erase current line) is Ctrl/U. See also **termio(7).**

setenv is used here to load the environment variable TERM with the terminal type returned by **tset** . If the terminal type is **dialup,** then **tset** will ask if **qume** is the correct terminal type. Reference **tset(1)** and **csh(1)** for **setenv** in the on–line manual.

The commands between **if ...** to **...endif** starts a "clock" display on your screen in the status line area. The **set** command assigns a value to a shell variable. Here it is being used to setup terminal parameters. The terminal type is compared to a Wyse WY–50 (current Stride terminal) and if true, properly sets up the status line for use by **sysline.** The time and date displayed on the status line will be updated every 30 seconds.

The **biff y** command arranges for the **mail** program to notify you *immediately* when any new mail has arrived.

The **clear** command clears the screen.

```
umask 02
set      path=(. $HOME/.bin /usr/local /usr/ucb /bin /usr/bin /usr/games)
stty     erase 'th' intr 'tc' kill 'tu'
#        setup terminal environment
setenv EREM  M'/usr/ucb/tset   – –Q –mdialup:?qume'
#        sysline setup
if ($TERM == wyse50) then
        set syslinepid = `(setenv TERM wyse50–78;sysline  –i –h +30)`
endif
#                                  *** NOTE ****
biff y                  The two characters ` and ' have different meanings.
clear                   ` is a single quote (value = 29 dec. 27 hex)
from                    ' is a grave accent (value = 96 dec. 60 hex)
fortune                 The stty line uses the single quotes, the other
                        lines shown have grave accents.
```

The **from** command shows you a current list of mail you have waiting.

fortune is a "game" which gives you a funny message chosen at random from a database.

.logout

Below is an example of a **.logout** file:

```
if ($?syslinepid) then
    kill –HUP $syslinepid
endif
clear
fortune
```

The commands in this file are executed when you say CTRL/D, or **logout** at the end of your UNIX session.

The **.logout** file shown in the example is very simple, the commands from **if...** to **...endif** terminate the **sysline** process which was displaying the time in the terminal's status line.

The **clear** and **fortune** commands do the same functions as they did in the **.login.** file.

.cshrc

Another file, **.cshrc** is also run at login time and is used to define how your prompt looks and to setup aliases needed by subshells. It is also run whenever a csh subshell is executed. One nice touch to add to your **.cshrc** is this: adding :

```
if ($?prompt) then
    set prompt='mycomputer(myname)[\!]
endif
```

Now your prompt will show your machine name, your name and how many commands have been typed so far.

```
mycomputer(myname)[1]
```

A little time spent trying combinations of these commands and others found in the references should help you build a user environment to your taste.

Again, refer to **csh(1)** in the on–line manual for details on how the **C shell** works. □

Finding p–System New File Lengths

by Chuck Emery

The p–System directory structure is sequential: that is, the files are assigned one after another on the volumes. If you remove or rewrite a file, the old area is marked <empty> and is available for a new file. After awhile, the directory may have lots of little <empty>s in it and only a few big areas. The Filer **K(runch** operation is required in order to consolidate all valid files at the top of the volume leaving one big <empty> area at the end.

When a program opens a new file with a **REWRITE** , the new file is assigned the largest empty area by the system. The problem is: *How does a program know how big that area is?* A very simple program could keep on writing until it got an error and dump the data written so far. This is not very user–friendly or secure.

Another simple, and workable, technique is to always specify the maximum size of the file. This is done by adding the size needed in brackets after the filename. For example: **newfile[10]** would get allocated in the first 10 free blocks on the volume. For a good description of how new files are assigned in the directory, refer to page 258 of *Advanced UCSD Pascal Programming Techniques* by Willner and Demchak. Assigning the size works if you know how much data you want to write (as when copying from another file) but in most cases the size of the new text is unknown.

The system unit FILEINFO does provide function **F_Length** as a way to get the length of an existing file. (Refer to the *Program Development Manual*). However, on a just–opened file, **F_Length** returns 0! Again, this is not what is needed.

The unit **TFLength** shown on the next page does a direct access to the **KERNEL** unit to find the size of the newly opened "temporary" file. (Note that the **Volume** name of the file cannot be a number such as "#4:" but must be a character name). The program below, **TryTFileLen** shows how to call the unit.

As you can see, most of the main body of the unit is code to check for a valid file name, determine the volume, and bring in the volume directory.

The five lines in **FetchFileInfo** match the directory entry with the file requested and calculate the size by subtracting the first block of the file from the last block of the file. This gives the current size of an existing file or the maximum available area for a new, "temporary" file. (Remember that for TEXT files,the first 2 blocks of this area are required as a header and not available to your program).

Another way to determine the area available for a new file is to use the function **dFreeSpace** in the **DIRINFO** unit. This function returns the volume size, total number of free blocks and the largest empty space.

However, using this assumes that the system will grant you that largest space for your file when you open it. This may not be the case under the Liaison networking system where other users may be accessing the same volume. If you also use the **dLock** and **dRelease** functions to control access to the directory, the assumption that the largest space is yours becomes more valid.

The program show here however, will *always* be correct since it requires that you successfully open the file first.

It also requires less room to run since **KERNEL** is always present in memory, but the units **DIRINFO** (2992 words) or **FILEINFO** (172 words) are in **SYSTEM.LIBRARY** and would have to be loaded into memory. □

(A source listing of the Unit is shown on the next page.)

Ed.Note: Chuck Emery can be reached at TDI Canada, 208 Evans Ave, STE 104, Toronto, Ontario M8Z–1J7, Canada. TEL: (416) 252–7181.

```
{TryTFileLen: the main program used to test
 the unit TFLength on next page.}
PROGRAM TryTFileLen;

USES {$U TF.Length.code} TFileLen;

VAR FileName:STRING;
    FID:TEXT;
    Len:INTEGER;
BEGIN
 WRITE('Name of Text File: ');
 READLN(FileName);
 IF (POS(FileName,'.text')=0)
   AND (POS(FileName,'.TEXT')=0) THEN
     FileName:=CONCAT(FileName,'.TEXT');
 REWRITE(FID,FileName);      {Create a new file}
 Len:=TFLength(FileName);
 IF Len>0 THEN
   BEGIN
     WRITELN('Max blocks available = ',Len);
     {... code to write new file would go here....}
     {... remember you lose 2 blocks to the header.}
     CLOSE(FID,LOCK);
   END ELSE WRITELN('Error: ',Len);
END.
```

```
{TFileLen: Returns the length, in blocks, of the file named
 in the argument.  Works for both IV.13, IV.2x versions}

Unit TFileLen;
INTERFACE
 FUNCTION TFLength ( Name : String ) : INTEGER;
IMPLEMENTATION
  USES  {$U UNITS:KERNEL.CODE} KERNEL;

  FUNCTION TFLength { Name : String ) : INTEGER} ;
    VAR
      I : INTEGER;
      Vol : VID;
      FL : File;

  PROCEDURE FetchFileInfo;
    BEGIN                                { Match Name in GDIRP }
      I:=1;
      While (I<77) AND (SYSCOM^.GDIRP^[I].DTID<>Name) Do I:=I+1;
      If SYSCOM^.GDIRP^[I].DTID=Name THEN TFLength:=
        SYSCOM^.GDIRP^[I].DLASTBLK-SYSCOM^.GDIRP^[I].DFIRSTBLK
       ELSE TFLength:=-1;                { -1 = file not found }
    END;


   FUNCTION GoodName:BOOLEAN;
     BEGIN
      { uppercase conversion }
      For I:=1 to LENGTH(Name) do IF Name[I] in ['a'..'z'] THEN
        Name[I]:=Chr(Ord(Name[I])-Ord('a')+Ord('A'));
      If POS ('[',Name)<>0 THEN Name:=COPY(Name,1,POS ('[',Name)-1); { trim }
      I:=POS (':',Name);                                             { size }
      If POS ('*',Name)=1 THEN           { We have SYVID reference }
        BEGIN
          Vol:=SYVID;
          If I=0 THEN Name:=COPY(Name,2,LENGTH(Name)-1)
          ELSE IF I=2 THEN Name:=COPY(Name,3,LENGTH(Name)-2)
          ELSE Vol:='';                  { bad volume name }
        END
      ELSE IF I=0 THEN Vol:=DKVID        { we have a DKVID reference }
      ELSE IF I=1 THEN                   { we have a DKVID reference }
        BEGIN                            { and an extra character }
          Vol:=DKVID;
          Name:=COPY(Name,2,LENGTH(Name)-1);
        END
      ELSE                               { we have an explicit volume }
        If I-1>VIDLENG THEN              { the volume name is too long }
          Vol:=''
        ELSE
          BEGIN
            Vol:=COPY(Name,1,I-1);
            Name:=COPY(Name,I+1,LENGTH(Name)-I);
          END;
      I:=Length(Name);
      GoodName:=NOT(  (I>TIDLENG) OR (I<1) OR (Vol='') );
      END;

    Begin {TFLength }
      IF GoodName THEN                   { We have legal Vol & Name }
        BEGIN
          If Vol<>SYSCOM^.GDIRP^[0].DVID THEN { Volume is not GDIRP }
            BEGIN
              {$I-} RESET (FL,CONCAT(Vol,':')); {$I+} { force Directory in }
              If Vol=SYSCOM^.GDIRP^[0].DVID THEN { Volume is NOW in GDIRP }
                FetchFileInfo
               ELSE TFLength:=-2;        { -2 = volume not on line }
            END
          ELSE FetchFileInfo;
        END ELSE TFLength:=-3;           { -3 = illegal file/volume name }
      END;


BEGIN
END.
```

USUS Spring National Meeting To Be Held in Dallas, Texas

The USUS group will hold its Spring National Meeting for p-System users in Dallas on **April 25–27, 1986 (Thurs., Fri., Sat., and Sun.).** New members and browsers are welcome. As always USUS is looking for demonstrators and speakers for the meeting. If you are interested in giving a p-System related talk of either a general or technical nature please call Carl Van Dyke at (609) 683- 0220. In addition, Carl would welcome offers to lead workshops on specific topics such as communications, data base management, or ASE macros.

Location: The Harvey Hotel (formerly Harvey House Hotel) is located at 7815 LBJ Freeway (LBJ is also known as I-635), on the west-bound access road just west of Central Expressway (US75) at Coit Road. Transportation to the hotel is available from DFW International Airport via "The Link". The one-way fare from DFW International Airport is $9. Taxis and car rentals are available from both DFW International Airport and Love Field. The mailing address is **Harvey Hotel, 7815 LBJ Freeway, Dallas, Texas 75240, Tel: (214) 960-7000 or (800) 900-0222.** Contact the Harvey Hotel directly to make room arrangements. The special room rate for USUS attendees will be $49 + 9% tax for a single or double. To get this rate, you *must* say that you are attending the USUS conference when you make your reservations.

Food, Shopping & Weather: The Harvey Hotel is five minutes by car from restaurants and six major shopping centers: NorthPark, the Galleria, Sakowitz Village, Preston Center, Valley View, and Prestonwood Malls. The unique Olla Podrida is 1/2 mile south of the hotel, and the art galleries and antique shops of the Oak Lawn area are only ten minutes away. There are a number of computer stores within this shopping area, including the original CompuShop, two ComputerLands, a Tandy Computer Center, The Computer People, Jade, the Micro Store (formerly Portia Isaacson's store), and an IBM Product Center. Dallas' weather in late April should be sunny and mild, so bring your bathing suit and enjoy the pool, Jacuzzi, and sauna.

Meeting Reservations: To register for meeting attendance, please send $25 per person to Bob Peterson, P.O. Box 1686, Plano, TX 75074 (214) 596-3720. You can also register at the door for $35 when you arrive. □

Remember the USUS interim meeting will be held during Stride Faire, Feb.28–Mar.2. The general USUS membership and any interested parties are welcome to attend.

Q&A

How can I get a text file from a PC to my Stride p-System?

A program called **PDOSTRAN** will do this for you. This utility is available from the USUS p-System user's group, ask for library volume 32. If you aren't a member of USUS see page 14 for the address and write for information. Also, the USUS officers will also be at Stride Faire. You can join up and browse through the many useful programs in the library.

What is the significance of the p-System filer "Warning units #x & #y have the same name "

This most often happens when a diskette is inserted with the same name as a volume on the hard disk. Problems can occur if you try to write a file to the volume using just the NAME:. If you use the number, all is well. However, if you use just the NAME:, the IV.21 p-System will write to the volume with the lowest unit number. Older versions of the p-System may leave an unfinished file in one volume. An unfinished file will have ???? in its directory listing. It may also be a duplicate file name. The best way to remove unfinished files is with a "R #x:?" which will prompt you to remove each file.

The first line of my WORD7 letter doesn't print out. What's wrong?

Go into the Printer Configuration. The definition of "Initialization" should be "0", no other characters. Printers such as Qumes can handle an initialization sequence, but ignore other input for a short time while installing the new setup. WORD7 does not wait for the initialization sequence, but immediately sends the first line, which the printer ignores. Again the fix is to clear the "Initialization sequence".

Help! My program tries to use many of the special features of the terminal, but it doesn't work. How do I figure out what's wrong?

Most terminals have a debug mode (monitor submode). When in this mode normally non-printing characters are printed with special symbols. In addition special features are disabled. For example a backspace is displayed with a special character but not executed. In this way all characters sent can be observed.

A program can turn on 'debug' mode on a Stride (Wyse WY-50) terminal by sending an ESC U in the output stream. An ESC X or an ESC u will turn off the mode. The user may also control this from the keyboard in local mode. Shifted break at the keyboard will toggle local mode on and off.

Also there is an error in the Wyse WY-50 reference guide. The command to 'enter host message' is ESC f not ESC F . Also the last character position in this bottom row controls the display attribute for the beginning of the screen and is normally not used.

Do I get an upgrade to the Program Reference Manual with the purchase of the new IV.21 p-System Compiler?

No. However, the Release Notes serve as an addendum to the manual covering the new options.

What do the SDT (Stride Debugging Tool) commands LT and LA do?

Nothing. They were reserved for new functions which were never needed and therefore were not implemented.

Does the IV.21 p-System release change the Stride multi-user system?

Not really. Just a new bootstrap was needed. The Stride multi-user will support either IV.21 or IV.13 or both together. However, at the time of the release of the new p-System, Stride also released a new MU.BIOS which had new features and updates in it. This new MU.BIOS can be purchased separately (Part Number DK0100) for $25. □

People & Products

Jeff Brower and **Hyperception** have a new address and phone: 9550 Skillman, LB125, Ste. 307H, Dallas, TX 75243 TEL: (214) 828-3508. Hyperception has a business graphics package for Stride/Sage machines.

You'll be hearing some new voices on the phones at Stride when you call in. **Joye Lorentzen** is our new PBX operator and **Charlie DeGroff** is now heading up Technical Support.

Paul Lamar has just released a **65C02 Cross-assembler** that runs under CP/M-68K on the Stride machines. Mr. Lamar can be reached at Lamar Micro, 2107 Artesia Blvd., Redondo Beach, CA 90278 Tel: (213) 374-1673

Stride's new regional sales managers, **Fred Schmidt** and **Tom Le Donne** have finished some on-the-job training here in Reno and are now in the field rarin' to go. Fred has Chicago and the midwest, Tom has the east coast and works out of Boston. They'll have to work hard to match veteran **Joe Wilson** who has left the nest in Reno and is now heading up the Southwest region from Los Angles. (Phone numbers are given on the back page.) □

Editor: Verlene Joyce Bonham

In Stride Tech Notes is a publication of Stride Micro, issued eight times yearly. Subscriptions are $12 for one year and include the *In Stride* parent magazine which is published quarterly, for a total of 12 issues per year.

Tech Notes back issues are also available for $0.50 and *In Stride* back issues for $2.00 as supply lasts.

Purchase of a Stride computer includes a one-year subscription upon receipt by Stride Micro, Reno, of a fully completed owner's registration card.

Requests for subscriptions, reprint permission, ad rates, bulk orders or submission of prospective articles should be sent to the *In Stride* Editor at Stride Micro's Reno address.

*Stride™, *Performance By Design*™ and NOD™ are trademarks of Stride Micro.*

Postmaster: Change of address notices should be sent to:

In Stride **Editor**
Stride Micro
P.O. Box 30016
Reno, NV 90520 -0016
(702) 322-6868
TWX: 910-395-6073

Eastern Region
Boston, Massachusetts
(617) 639-0750

SouthCentral Region
Dallas, Texas
(214) 392-7070

SouthWest Region
Los Angeles, California
(818) 363-6814

NorthCentral Region
Chicago, Illinois
(312) 377-1234

STRIDE
MICRO