68' MICRO JOURNAL

Australia A $4.75 | New Zealand NZ $ 6.50
Singapore S $ 9.45 | Hong Kong H $23.50
Malaysia M $ 9.45 | Sweden 30 SEK

$2.95 USA

Motorola VME-MACINTOSH-S 50 & Other 68XXX Systems

6809 68008 68000 68010 68020 68030

OS-9

The Magazine for Motorola CPU Devices For Over a Decade! A User Contributed Journal

FLEX SK*DOS

This Issue:

"C" User Notes p.13
Mac-Watch
Macintosh Spellswell Review p.22
Basically OS-9 p.18
Software User Notes p.8
Graphics On FORTH p. 25
An RS-232 Breakout Box p.29

And Lots More!

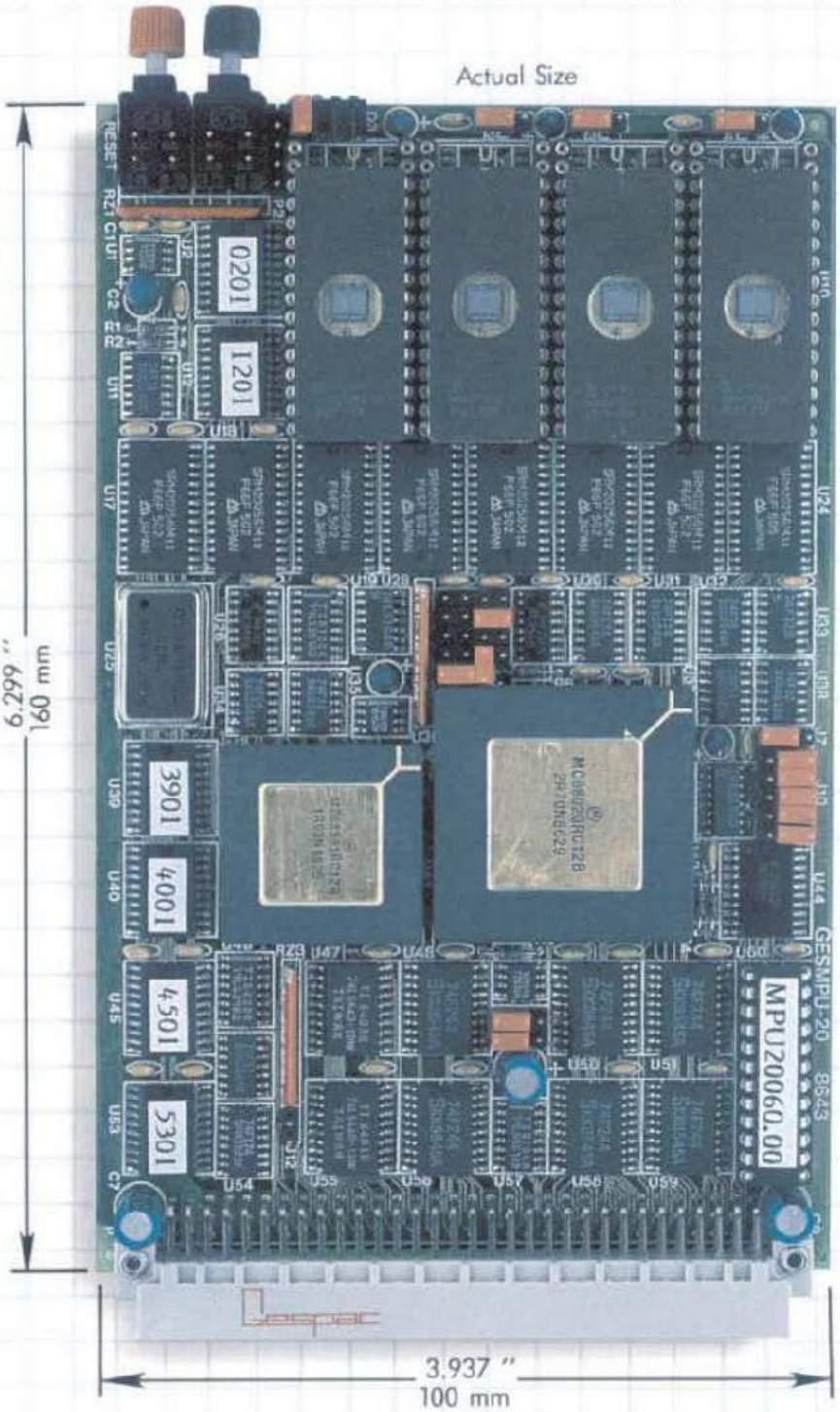
VOLUME IX ISSUE III • Devoted to the 68XXX User • March 1987

"Small Computers Doing Big Things"

SERVING THE 68XXX USER WORLDWIDE

03
0 74470 12810

GESPAC Gives You More Power Per Square Inch.



Here is the size, performance, and cost breakthrough you hove been waiting for: The 68020 based GESMPU-20 from GESPAC.

You con now unleash on unprecedented amount of power into your applications. On just 25 square inches, we hove squeezed a 12.5 MHz (16 MHz optional) 68020 32-bit microprocessor, a 68881 floating point coprocessor, 4 sockets for up to 512 Kilobytes of EPROM, and up to 512 Kilobytes of zero-wait-states CMOS RAM.

This board is totally expandable through its G-64 bus interface. And GESPAC has the largest variety of inexpensive memory, interfaces, controllers, and transducer cords anywhere. Plus real-time disk operating systems, high level languages, and other software tools. GESPAC has the total solution to your system integration needs.

Best of all, because our boards ore small, they cost less. The new GESMPU-20 is priced below $1000 in one hundred piece quantity orders.

So why wait? Contact us today for information on the GESMPU-20 or any of the 150 G-64 bus system components from GESPAC—the leader in single Eurocard microcomputer products worldwide.

Call (602) 962-5559.

Gespac

IN USA - CANADA
50A West Hoover Ave.
Mesa, Arizona 85202
Tel. (602) 962-5559
Telex 386575

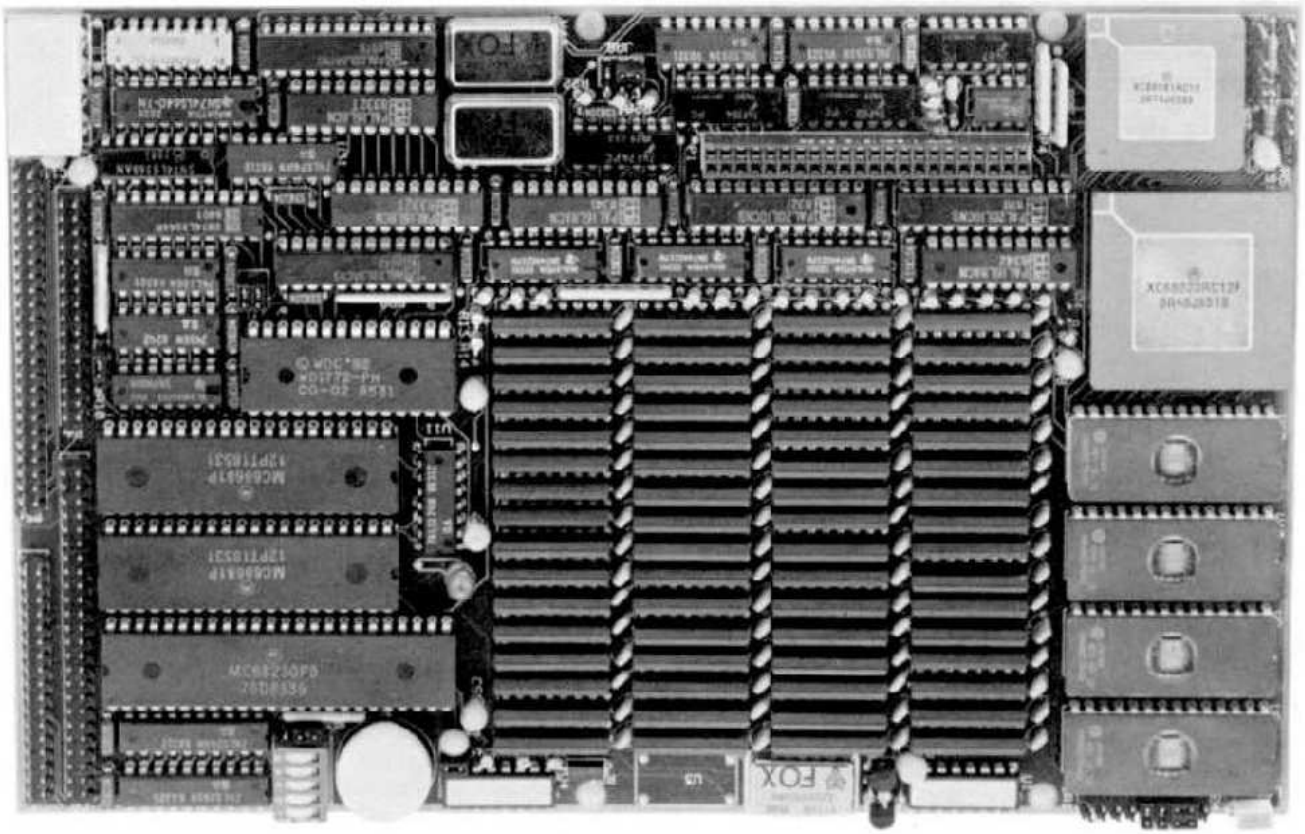
INTERNATIONAL
3, chemin des Aulx
CH-1228 Geneva
Tel. (022) 713400
Telex 429989

GMX™ Micro-20
68020 SINGLE—BOARD COMPUTER

Mainframe CPU Performance on a 5.75" x 8.8" Board
(benchmark results available on request)

$2565.00

12.5 MHz Version
Quantity Discounts Available



Features

- 32-Bit MC68020 Processor (12.5, 16.67, or 20MHZ)
- MC68881 Floating-point coprocessor (optional)
- 2 Megabytes of 32-bit wide, high-speed RAM
- 4 RS-232 Serial I/O Ports (expandable to 36)
- 8-bit Parallel I/O Port ('Centronics' compatible)
- Time-of-Day Clock w/battery backup
- 16-bit I/O Expansion Bus
- Up to 256 Kbytes of 32-bit wide EPROM
- Floppy Disk Controller for two 5¼" drives
- SASI Intelligent Peripheral Interface (SCSI subset)
- Mounts directly on a 5¼" Disk Drive
- Optional Boards include Arcnet, Prototyping, I/O Bus adapter, 60 line Parallel I/O, RS-422/485

Software

Included:

- GMX Version of Motorola's 020Bug Debugger with up/download, breakpoint, trace, single-step, and assembler/disassembler capabilities
- Comprehensive Hardware Diagnostics

Optional:

UNIX™ like Multi-user/Multi-tasking Disk Operating Systems

- OS-9/68000™ (Real-time and PROMable)
- UniFLEX™

Programming Languages and Application Software

- BASIC, C, PASCAL, ABSOFT FORTRAN, COBOL and ASSEMBLER
- Spreadsheet, Data Base Management, and Word Processing

COMPLETE EVALUATION SYSTEMS AVAILABLE

GMX INC. 1337 W. 37th Place Chicago, IL 60609

(312) 927-5510 • TWX 910-221-4055
State-of-the-Art Computers Since 1975

A Member of the CPI Family

68 Micro Journal

6800 6809 68000 68010 68020

10 Years of Dedication to Motorola Users

Editorial Staff

Publisher:
Don Williams Sr.

Executive Editor:
Larry Williams

Production Manager:
Tom Williams

Administration:
Office Manager:
Mary Robertson
Subscriptions:
Joyce Williams

Contributing & Associate Editors:

Ron Anderson
Ron Voigts
Doug Lurie
David Lewis
Dr. E.M. Bud Pass
Art Weller
Dr. Theo Elbert
& hundreds more of us

Contents

Software User Notes	8	Anderson
"C" User Notes	13	Pass
Basically OS-9	18	Voigts
Mac-Watch	22	Staff
Graphics On FORTH	25	Lurie
RS-232 Breakout Box	29	Baliski
Bit Bucket	36	All of Us
68' Micro 1986 Index	36	Current
Time .CMD	44	Drexler
Classifieds	51	

"Contribute *Nothing* - Expect *Nothing*" DMW 1986

COMPUTER PUBLISHING, INC.
"Over a Decade of Service"
"World Wide"

68 MICRO JOURNAL
CPI
Computer Publishing Center
5900 Cassandra Smith Road
PO Box 849
Hixson, TN 37343

Phone (615) 842-4600 - Telex 510 600-6630

Copyright © 1987 Computer Publishing, Inc.

68 Micro Journal is published 12 times a year by Computer Publishing, Inc. Second Class Postage paid ISSN 0194-5025 at Hixson, TN and additional entries. Postmaster: send form 3597 to 68 Micro Journal, POB 849, Hixson, TN 37343.

Subscription Rates

1 Year: $24.50 USA, Canada & Mexico $34.00 *a year. Others add* **$12.00** *a year surface, airmail add* **$48.00** *a year, USA funds!* **2 Years $42.50, 3 Years $64.50** *plus additional postage, for each* additional year.

Items or Articles for Publication

Articles submitted for publication must include authors name, address, telephone number and date, as well as a statement that the material is original and the property of the submitting author. Articles submitted should be on diskette, Macintosh, OS-9 or FLEX format. All printed items should be dark type and satisfactory for photo-reproduction. No blue ink! No hand written articles - please.

Please do not *format with spaces any text indents, chart items, etc. (source listings o.k.) WE will edit in ALL formatting.* Text should be flush left column and use ONLY a carriage return to separate paragraphs or other article text items! MacWrite, FLEX TSC, Stylo formatting acceptable.

Letters & Advertising Copy

Letters to the Editor should be original copy, signed! Letters of gripe as well as praise are acceptable. *We reserve the right to reject any letter to the editor or advertising copy material, for any reason we deem advisable.*

Advertising Rates: Commercial please contact 68 Micro Journal advertising department. Classified ads must be non-commercial. Minimum of $15.50 for first 15 words. Add $.60 per word after the first 15. All classifieds must be pre-paid. No classifieds accepted by telephone.

The VME BUS and OS-9:

Ultimate Software for the Ultimate Bus.

Modularity. Flexibility. High Performance. Future growth. These are probably the prime reasons you chose the VME bus. Why not use the same criteria when selecting your system software? That's why you should take a look at Microware's OS-9/68000 Operating System—it's the perfect match for the VME bus.

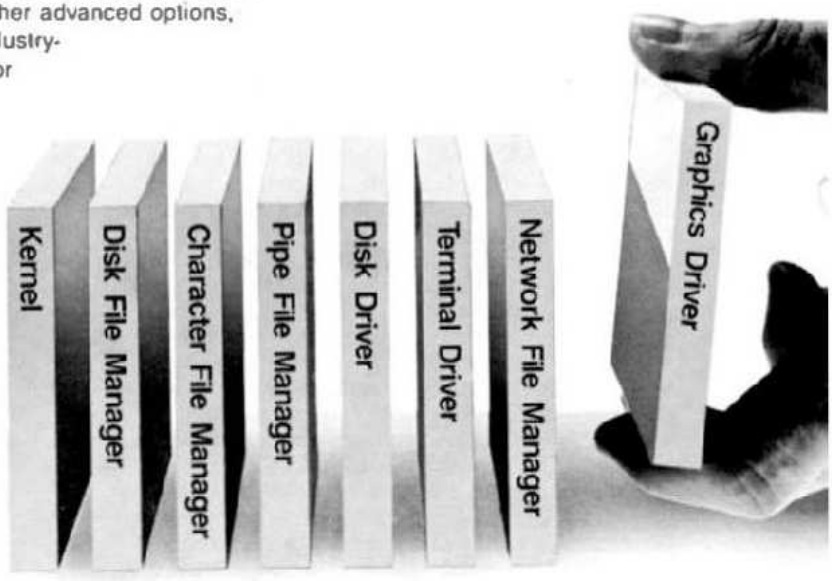
When you're working with VME you must have access to every part of the system. Unlike other operating systems that literally scream KEEP OUT!, OS-9's open architecture invites you to create, adapt, customize and expand. Thanks to its unique modular design, OS-9 naturally fits virtually any system, from simple ROM-based controllers up to large multiuser systems.

And that's just the beginning of the story. OS-9 gives you a complete UNIX-application compatible environment. It is multitasking, real time, and extremely fast. And if you're still not impressed, consider that a complete OS-9 executive and I/O driver package typically fits in less than 24K of RAM or ROM.

Software tools abound for OS-9, including outstanding Microware C, Basic, Fortran, and Pascal compilers. In addition, cross C compilers and cross assemblers are available for VAX systems under Unix or VMS. You can also plug in other advanced options, such as the GSS-DRIVERS™ Virtual Device Interface for industry-standard graphics support, or the OS-9 Network File Manager for high level, hardware-independent networking.

Designed for the most demanding OEM requirements, OS-9's performance and reliability has been proven in an incredible variety of applications. There's nothing like a track record as proof: to date, over 200 OEMs have shipped more than 100,000 OS-9-based systems.

Ask your VME system supplier about OS-9. Or you can install and evaluate OS-9 on your own custom system with a reasonably priced Microware PortPak™. Contact Microware today. We'll send you complete information about OS-9 and a list of quality manufacturers who offer off-the-shelf VME/OS-9 packages.



Modular Hardware Deserves Modular Software

MICROWARE.

Microware Systems Corporation
1866 N.W. 114th Street • Des Moines, Iowa 50322
Phone 515-224-1929 • Telex 910-520-2535

Microware Japan, Ltd.
41-19 Honcho 4-Chome, Funabashi City • Chiba 273, Japan • Phone 0473 (28) 4493 • Telex 781-299-3122

Micromaster Scandinavian AB
5:1 Persgatan 7
Box 1309
S751-43 Uppsala
Sweden
Phone: 018-138595
Telex: 76129

Dr. Rudolf Keil, GmbH
Porphystrasse 15
D-6905 Schriesheim
West Germany
Phone: (0 62 03) 67 41
Telex: 465025

Elsoft AG
Zelweg 12
CH-5405 Baden-Dattwil
Switzerland
Phone: (056) 83-3377
Telex: 828275

Vivaway Ltd.
36-38 John Street
Luton, Bedfordshire, LU1 2JE
United Kingdom
Phone: (0582) 423425
Telex: 825115

Microprocessor Consultants
92 Bynya Road
Palm Beach 2108
NSW Australia
Phone: 02-919-4917

Microdata Soft
97 bis, rue de Colombes
92400 Courbevoie
France
Phone: 1-768-80-80
Telex: 615405

OS-9 is a trademark of Microware and Motorola. PortPak is a trademark of Microware. GSS-Drivers is a trademark of Graphic Software Systems, Inc. VAX and VMS are trademarks of DEC. Unix is a trademark of AT&T.

MUSTANG -020 Super SBC™

DATA-COMP proudly presents the first Under $5000 "SUPER MICRO".

The MUSTANG-020™

MUSTANG-020™

The MUSTANG-020 68020 SBC provides a powerful, compact, 32 bit computer system featuring the "state of the art" Motorola 68020 "super" micro-processor. It comes standard with 2 megabyte of high-speed SIP dynamic RAM, serial and parallel ports, floppy disk controller, a SASI hard disk interface for intelligent hard disk controllers and a battery backed-up time-of-day clock. Provisions are made for the super powerful Motorola MC68881 floating point math co-processor, for heavy math and number crunching applications. An optional network interface uses one serial (four (4) standard, expandable to 20) as a 125 bit per second network channel. Supports as many as 32 nodes.

The MUSTANG-020 is ideally suited to a wide variety of applications. It provides a cost effective alternative to the other MC68020 systems now available. It is an excellent introductory tool to the world of hi-power, hi-speed new generation "super micros". In practical applications it has numerous applications, ranging from scientific to education. It is already being used by government agencies, labs, universities, business and practically every other critical applications center, worldwide, where true multi-user, multi-tasking needs exist. The MUSTANG-020 is UNIX C level V compatible. Where low cost and power is a must, the MUSTANG-020 is the answer, as many have discovered. Proving that price is not the standard for quality!

As a software development station, a general purpose scientific or small to medium business computer, or a super efficient real-time controller in process control, the MUSTANG-020 is the cost effective choice. With the optional MC68881 floating point math co-processor installed, it has the capability of systems costing many times over it's total acquisition cost.

With the DATA-COMP "total package", consisting of a heavy duty metal cabinet, switching power supply with rf/line by-passing, 5 inch DS DD 80 track floppy, Xebec hard disk controller, 25 megabyte winchester hard disk, four serial RS-232 ports and a UNIX C level V compatible multi-tasking, multi-user operating system, the price is under $5000, w/12.5 megahertz system clock (limited time offer). Most all popular high level languages are available at very reasonable cost. The system is expandable to 20 serial ports, at a cost of less than $65 per port, in multiples of 8 port expansion options.

The system SBC fully populated, quality tested, with 4 serial ports pre-wired and board mounted is available for less that $3000. Quantity discounts are available for OEM and special applications, in quantity. All that is required to bring to complete "system" standards is a cabinet, power supply, disks and operating system. All these are available as separate items from DATA-COMP.

A special version of the Motorola 020-BUG is installed on each board. 020-BUG is a ROM based bebugger package with facilities for downloading and executing user programs from a host system. It includes commands for display and modification of memory, breakpoint capabilities, a powerful assembler/disassemble and numerous system diagnostics. Various 020-BUG system routines, such as I/O handlers are available for user programs.

Normal system speed is 3-4.5 MIPS, with burst up to 10 MIPS, at 16.6 megahertz. Intelligent I/O available for some operating systems.

Hands-on "actual experience sessions", before you buy, are available from DATA-COMP. Call or write for additional information or pricing.

DATA-COMP
Installed Systems World-Wide
5906 Cassandra Smith Road
Hixson, Tn 37343
Telephone 615 842-4600
Telex 810 600-6630

MUSTANG-020, MUSTANG-08 Benchmarks

All timings by independent consultant	Time - seconds 32 bit Integer	Register Long
IBM AT 7300 Xenix Sys 3	9.7	no register
AT&T 7300 UNIX PC 68010	7.2	4.3
DEC VAX 11/780 UNIX Berkley 4.2	3.6	3.2
DEC VAX 11/750 " "	3.1	3.2
68000 OS-9 68K 10 Mhz	6.5	4.0
6809E OS-9 68K 8 Mhz	18.0	9.0
MUSTANG-08 6809E OS-9 68K 10 Mhz	**9.8**	**6.3**
MUSTANG-020 68020 OS-9 68K 16 Mhz	**2.2**	**0.88**
MUSTANG-020 68020 MC68881 UniFLEX 16 Mhz	**1.8**	**1.22**

```
Main()
    {
        register long i;
        for (i=0; i < 999999; ++i);
    }
```

Estimated MIPS - MUSTANG-020 — 4.5 MIPS.
Burst to 8 - 10 MIPS; Motorola Specs.

MUSTANG-020-

FEATURES

- 12.5 Mhz (optional 16.6 Mhz available) MC68020 full 32-bit wide path processor
- 32-bit wide data and address buses, non-multiplexed
- on chip instruction cache
- object code compatible with all 68XXX family processors
- enhanced instruction set - math co-processor interface
- 68881 math hi-speed floating point co-processor (optional)
  direct extension of full 68020 instruction set
  full support IEEE P754, draft 10.0
  transcendental and other scientific math functions
- 2 Megabyte of SIP RAM (512 x 32 bit organization)
- up to 256K bytes of EPROM (64 x 32 bits)
- 4 Asynchronous serial I/O ports standard
  optional to 20 serial ports
  standard RS-232 interface
  optional network interface
- buffered 8 bit parallel port (1/2 MC68230)
  Centronics type pinout
- expansion connector for additional I/O devices
  16 bit data path
  256 byte address space
  2 interrupt inputs
  clock and control signals
  Motorola I/O Channel Modules
- time of day clock/calendar w/battery backup
- controller for 2, 5 1/4" floppy disk drives
  single or double side, single or double density
  35 to 80 track selectable (48-96 TPI)
- SASI interface
- programmable periodic interrupt generator
  interrupt rate from micro-seconds to seconds
  highly accurate time base (5 PPM)
- 5 bit sense switch, readable by the CPU
- hardware single-step capability
- mounts directly to a standard 5 1/4" disk drive

Size 8 15/16 x 5 7/8

These *hi-speed* 68020 systems are presently working at NASA, Atomic Energy Commission, other Government Agencies as well as Universities, Business, Labs, and critical applications centers, Worldwide, where speed, math crunching and multi-user, multi-tasking UNIX C level V compatability and low cost is a *must*!

MUSTANG-020™ Software

OS-9

OS-9	$350.00
Basic09	300.00
C Compiler	500.00
Fortran 77	400.00
Microware Pascal	400.00
Cottagesoft Pascal	900.00
Style-Graph	495.00
Style-Spell	195.00
Style-Merge	175.00
Style-Graph-Spell-Merge	695.00
PAT w/C source	229.00
JUST w/C source	79.95
PAT/JUST Combo	269.50
Sculptor+ (see below)	995.00
CDM	125.00
Cross Assemblers	50.00
Checksum software	100.00
Disassemblers	100.00

UniFLEX

UniFLEX	$450.00
Screen Editor	250.00
Sort-Merge	200.00
BASIC/PacCompiler	300.00
C Compiler	350.00
COBOL	750.00
CMODEM software	100.00
TMODEM software	100.00
X-TALK (see Ad)	99.95
Cross Assembler	50.00
Fortran 77	450.00
Sculptor+ (see below)	995.00

Option & Expansion

8 Port expansion RS-232 (total of 20 serial ports supported)	495.00
Expansion for Motorola I/O Channel Modules	$195.00

** All Expansion boards

All expansion boards for old style cabinet will require the 101 expansion cable. Systems ordered with newer PC type cabinet do not require this cable.

101 Expansion Cable $39.95

Sculptor+: We are USA distributors for Sculptor+. Call or write for site or multiple system license & discounts. OEM/Dealer.

Special for complete MUSTANG-020™ system buyers - Sculptor+ $695.00. Save $300.00!

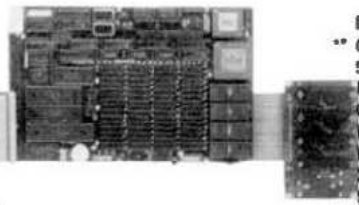
Software Discounts

All MUSTANG-020™ system and board buyers are entitled to discounts on all listed software: 10-70% depending on item. Call or write for quotes. Discounts apply after the sale as well.

For a limited time we are offering a $400.00 trade-in on your old 68XXX SBC. Must be working properly and complete with all software, cables and documentation. Call for details.

NOTE: UniFLEX is reported to run slower than OS-9 with more than several users on line - Also call or write for information on OS-9 Version 2. soon to be available. A full 68020 OS-9, with 68881 support.

MUSTANG-020 System component prices - Effective July 1, 1986
Prices subject to change - call for latest quotes.



	MUSTANG-020 (12.50 Mhz)	$2750.00
**	Cabinet (PC or as shown)	$299.95
	5"-80 track floppy DS/DD	$269.95
	Floppy cable	$39.95
	OS-9 68K	$350.00
	Winchester cable	$39.95
	Winchester Drive 25 Mbyte	$895.00
	Xebec H/D controller	$395.00
	Shipping USA UPS	$20.00
	Total	$5059.80

• DISCOUNT OFF COMPLETE SYSTEM **$1061.00**

Complete System :

25 Mbyte HD **$3998.80**

85 Mbyte HD **$5248.80**

OPTIONS ADD:

UniFLEX	$90.00
MC68881 I/p math processor	$275.00
16.67 Mhz MC68020	$375.00
16.67 Mhz MC68881	$375.00

This price subject to increase
Additional MUSTANG systems soon

Note: Current OS-9 (Ver. 1.2) does not address the MC68881 - Future revisions will. If the 68881 is anticipated in the future, it must be ordered with the system, when originally ordered. UniFLEX does support both the enhanced code of the 68020 and 68881 now.
OPTION BOARDS: ** *Option boards to be installed in Mustang-020 cabinets must be ordered with the extension cable.* The cabinet is too tight for direct plugon. Or specify our new PC type cabinet, with initial order.

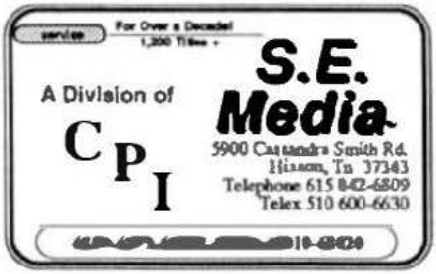
PAT - JUST

PAT

With 'C' Source

$229.00

All OS-9 68XXX Systems



PAT FROM S. E. MEDIA -- A FULL FEATURED SCREEN ORIENTED TEXT EDITOR with all the best of PIE. For those who swore by and loved **PIE**, this is for **YOU! All PIE features & much more!** Too many features to list. And if you don't like ours, change or add your own. **C source included.** Easily configured to your CRT terminal, with special configuration section. No sweat!

68008 - 68000 - 68010 - 68020 OS-9 68K $229.00

COMBO PAT/JUST

Special $249.00

JUST

JUST from S. E. MEDIA - - Text formatter written by Ron Anderson; for dot matrix printers, provides many unique features. Output formatted to the display. User configurable for adapting to other printers. Comes set-up for Epson MX80 with Graflex. Up to 10 imbedded printer control commands. Compensates for double width printing. Includes normal line width, page numbering, margin, indent, paragraph, space, vertical skip lines, page length, centering, fill, justification, etc. Use with **PAT** or any other text editor. The ONLY stand alone text processor for the 68XXX OS-9 68K, that we have seen. And at a very **LOW PRICE!** Order from: S.E. MEDIA - see catalog this issue.

68008 - 68000 - 68010 - 68020 OS-9 68K
With 'C' source $79.95

An Ace of a System in Spades!

MUSTANG-08™

ONE PENNY SALE

NOT 128K, *NOT* 512K
FULL 768K No Wait RAM

1¢

The MUSTANG-08™ system took every hand from all other 68008 systems we tested, running OS-9 68K!

The MUSTANG-08 includes OS9-68K™ and/or Peter Stark's SK*DOS™. SK*DOS is a single user, single tasking system that takes up where *FLEX™ left off. SK*DOS is actually a 68XXX FLEX type system *(Not a TSC product.)*

The OS-9 68K system is a full blown multi-user, multi-tasking system. All the popular 68000 OS-9 software runs. It is ~~fast whiz on disk~~ fact is: the MUSTANG-08 is faster on access than ~~the~~ 68XXX systems are on memory access. Now that is fast! And that is a small part of the ~~story~~! See benchmarks

*Introductory price of $1,~~~ (2-80 track DSDD floppy). Complete in ~~desk~~ style cabinet, heavy duty switching power supply, rf by-passing, ready to run, with your choice of OS-9 68K or SK*DOS.* Add $750 for a single floppy/25 megabyte hard disk system. For those that waited, DATA-COMP didn't forget.

Specifications: System includes OS-9 68K or SK*DOS - Your Choice

CPU	MC68008	10 Mhz
RAM	768K No Wait States	256K Chips
PORTS	2 - RS232 2 - 8 bit Parallel	MC68681 DUART MC6821 PIA
CLOCK	MC146818	Real Time Clock
EPROM	16K, 32K or 64K	Selectable
FLOPPY	WD1772	5 1/4 Drives
HARD DISK	Interface Port	WD1002 Board
Size:	5.75 X 8 inches - bolts directly to a floppy or HD	

1¢

C Compile times: OS-9 68K. Hard Disk file. LIST utility source from K&R.

MUSTANG-08	0 min - 32 sec
Other popular 68008 system	1 min - 05 sec
MUSTANG-020	0 min - 21 sec

Dual 5" Disk System
$1,998.08

25 Megabyte Hard Disk System
$1,998.09

~~$2,748.08~~

Unlike other 68008 systems there are several significant differences. The MUSTANG-08 is a full 10 Megahertz system. The ~~68~~ has No wait ~~states~~. As ~~this~~ is more MUSTANG type ~~features~~.

Also, allowing for addressable ROM/PROM the RAM is the maximum allowed for a 68008. The 68008 can only address a total of 1 Megabytes of RAM. The design allows all the RAM space (for all practical purposes) to be utilized. What is not available to the user is required and reserved for the system. ~~RAM~~ disk ~~of~~ 480K can be easily configured, leaving 288K ~~for~~ program ~~and~~ system RAM space. The RAM DISK can be ~~used~~ to ~~store~~ your application requires (system must ~~have~~ ~~more~~ for requirements). Leaving the ~~rest~~ for program use. Sufficient

~~$2,00~~ ~~Trade-in~~ ~~Prices see~~ ~~MUSTANG-02~~ ~~page #5~~

Limited Time SALE!

~~Save $~~

Mustang Hi-Speed Systems Only from Data-Comp Div.

	Seconds	32 bit Integer	Register Long
Other 68008 8 Mhz OS-9 68K	18.0	9.0	
MUSTANG-08 10 Mhz OS-9 68K	9.8	6.3	

```
main()
{
        /* int i; */
C Benchmark Loop    register long i;
        for (i=0; i < 999999; ++i);
}
```

MUSTANG-08 is a ~~trademark~~ of CPI
SK*DOS is a trademark of Star-K

Data-Comp Division

A Decade of Quality Service*
Systems World-Wide

Computer Publishing, Inc. 9900 Cassandra Smith Road
Telephone 615 842-4601 - Telex 510 600-6630 Hixson, Tn 37343

* Those with SWTPC hi-density FLEX 5" - Call for special info

SOFTWARE USER NOTES

A Tutorial Series

By: Ronald W. Anderson
3540 Sturbridge Court
Ann Arbor, MI 48105

From Basic Assembler to HLL's

With this column I am starting a third 8" disk of text and example program listing files. The second contains 22 columns and it is not filled to the end, but I decided it was time to relegate it to the archives and start fresh again. I am wondering (seriously) at this point how long it will be before 8 inch floppy disks and drives are totally obsolete. When I bought mine, my 35 track single sided single density 5 1/4" drives held 85K bytes of information. The new 8 inch DSDD drives held just about a megabyte, or the equivalent of 12 5 1/4" diskettes. My collection of some 100 small disks would all fit on 8 of the larger ones, though I didn't organize things quite that way. I put BASIC programs on one disk, my System files on another, letters on a third, '68' Micro Journal columns on a 4th, etc.

Now an 80 track double sided double density disk can hold almost 800K bytes. There is an access speed advantage to the 8" drives, but otherwise they are large and heavy. My two 8" drives are in a box that is larger than my computer. The new 1/2 height 5 1/4" drives occupy much less space and hold almost as much data.

The real reason that I fear the demise of the 8" drives, however, is not the competition from the slower small floppies, but from the ever declining cost of a hard disk drive. My two 8" drives cost about $1100 plus power supply, plus $350 for a disk controller board. Now I can buy enough hardware (drive and controller) to run a 20 Mbyte hard disk for less than that. Furthermore, regular '68' readers will know that the software drivers are available thanks to Leo Taylor and others, in public domain. Are my very expensive 8" drives to become very large paperweights, or anchors for rowboats?

The real reason that I fear the demise of the 8" drives, however, is not the competition from the slower small floppies, but from the ever declining cost of a hard disk drive. My two 8" drives cost about $1100 plus power supply, plus $350 for a disk controller board. Now I can buy enough hardware (drive and controller) to run a 20 Mbyte hard disk for less than that.

Back From Brazil

Yes, it was a very good trip. We made many new friends and in fact, I talked to our exchange student just today (Christmas day). We had been trying to get a call through to Brazil all day (the lines really get busy on Christmas), and we re-

ceived a call from there about 6:00. I will summarize briefly, since this is not a travel column. The country is beautiful. The people are VERY friendly. The food is excellent. Most of our traveling was done in the more temperate climate of the state of Sao Paulo, about as far south in latitude as Miami or Cuba are North. We were impressed by the coffee plantations everywhere in the area where we stayed primarily. Other industries down there are Sugar Cane, Rubber Trees, Citrus Groves, and Silk. Brazil converts a great deal of sugar cane to alcohol to use as a fuel. Many of the automobiles are equipped to run on alcohol, which is considerably less expensive than gasoline there.

I promised a report on computing there. I can report only one encounter with computers. A business in the town where we stayed, (Garca, a town of 35,000 about 250 miles Southwest of Sao Paulo city) was manufacturing Apple II clones. It was apparently a going business, though I didn't learn a great deal about it, since the owner / engineer spoke only Portugese and our student / interpreter was not with me the day we visited the computer business. The company seemed to be working on data processing applications. Unfortunately, in the government's zeal to encourage industry within Brazil, they have very high import duties on electronic equipment, up to 250%, I understand. This severely limits imports of personal computers, video equipment, etc.

How's That Again?

Sorry but I have run across a number of statements that I think are funny and I am going to share them with you. First from a "Voice Synthesizer" manual from Jameco Electronics, "Changes are periodically made to the information contained hearin." If I am hearin' correctly, of course they mean "herein".

A recent news item from Detroit (on the 11:00 news on ABC) said that there was a fire in "a vacant building occupied by homeless people". If the building was occupied, it wasn't vacant, and how could the people be living in it and be homeless?

From a manual on a D.C. motor speed control: "If the armature is to be disconnected and reconnected with AC power applied the Inhibit Circuit must be simultaneously activated and deactivated." Anyone have a switch that turns on and off simultaneously?

Back to Computing

I have had a few calls from people reporting problems getting PAT running on their Mustang systems, and unfortunately, the trip and the catching up that I had to do when I returned have delayed my getting at the project of fixing a few reported bugs and getting corrected copies ready for distribution. The other day I went to fire up the Mustang and the power supply died. A little investigation shows that the power transistor in the switching supply has "punched through" It measures zero ohms from collector to emitter. The board is on its way off to Data Comp for repair. As soon as it is returned I will get into that version of PAT and clean it up. I have had one complaint about JUST not running correctly with an Epson MX-80 with Graftrax Plus. I am going to get into that one immediately now. JUST was originally developed using an identically equipped Epson, but perhaps when I switched to a later RX-80 model I made a change that is incompatible with the older model. Since I still have the older Epson, I am going to set it up and find the cause of the problem.

Standardization

We die hard individualists who like our old 6809 systems and know that they are as good as if not better than at least the first and second generation "big blue" offerings, face a dilemma Let me explain from personal experience why almost everyone jumps on the IBM bandwagon.

Pardon the example once again, but in order to speak from personal experience, I must use PAT. I started writing PAT a long time ago as an exercise to see what I could do with a screen editor, writing it in a language at a higher level than assembler. I lost interest after some initial work on screen handling, but one day I decided to give it a go and see if I could finish a fairly difficult project and make it as good as or better than anything else I had seen (a highly biased judgement, of course).

After a year or so of working on it in my spare time and some that I really couldn't spare, I sent out some copies to several people well known to me to be 6809 FLEX enthusiasts. They were most helpful in reporting a substantial number of bugs to me, and also in suggesting useful features that I could add to it. One tester in particular called and told me that PAT should to take advantage of the fact that terminals scroll when you get to the bottom of the screen. I realized that I didn't have to rewrite the whole screen. Just force it to scroll and write the last line and the status line again. It will be much faster, particularly on a terminal running below 9600 baud. That feature made handling a particular situation an exception, but editors are full of exceptions anyway, so I added the software to do it, and noticed little difference on my 19.2K baud terminal. I had terminal configurations for the old dumb terminal (ADM-3A) an ADDS Viewpoint Plus, and several Televideo versions.

I figured it was about time to start selling PAT, and Don Williams agreed to offer it, providing me with an adequate royalty per copy, and advertising it and handling production of disks and shipping, at no expense to me. About a week after the first copies were sold, I received a call from someone with an ANSI standard terminal. I knew about those, but figured most 6809 users wouldn't have one. I did a PATANSI special version for such terminals, changing the cursor positioning procedure and making provision for longer control strings than I had previously allowed. Next call was from someone with a terminal that could either allow cursor positioning or scroll, but not both. I had to do a special PATNS (No Scroll) version for that individual, removing the additions I had made to handle that situation at the bottom of the screen. Next someone with an original FLEX 09 version (the very first offering of Technical Systems Consultants of the 6809 version) called. That version is missing one jump vector that allows access through FLEX to the routine that inputs a character from the terminal without echoing it. I had to write such a routine and access the serial port directly with another version that I called PATF9 to solve that problem.

I realized that the market would be limited for PAT at the beginning, and this is not a complaint in any way, just a relating of "how it is" in the FLEX software market. At any rate, after the first flurry of "it doesn't work with my system", things settled down a little, though I presently have a few customers who have been very patient about problems specific to their systems. I found it necessary for a while to get some other things done, and those problems are still with me, waiting to be solved as I mentioned above.

Several months ago I decided to try to reduce PAT a little, that is shrink the code, and while I was at it I added a few last features, such as a search that can be case insensitive, a "bookmark" feature whereby you can mark a place in a text and go back to that place simply, and a means of saving the present file and loading another one without exiting PAT. Those things are all implemented, but I have not yet released an upgrade version because I am still finding a bug now and then.

Also several months ago, I decided to do a "C" version of PAT for the Mustang 68020 system to run under OS-9/68K. I have been using that version on my Mustang system for some time, but have not upgraded to the changes that I made in my last pass on the 6809 version. A few people have written or phoned to say that the "C" version simply does not run on their system. They get error messages and return to OS-9 or for some reason the configuration file is not found, etc.

All in all, to date, my royalties on PAT have amounted to about $1.00 per hour of the time I spent writing it!

We simply are too small a group to represent a serious market to any software writer that wants to make a living from his work. Probably very few of us have systems that have not been customized in one way or another. We have homebrew terminals, homebrew hard disk systems, etc. The reason the IBM and its clones are so popular is the standardization. The terminal is part of the system. Once you learn how to put the cursor somewhere on the screen in one IBM system, you can handle all the others and compatibles the same way. The software supplier doesn't have to support 100 different terminals. The IBMs, like it or not, represent a much larger and more standardized market. Selling several thousand copies of a software item for $45 each is obviously a better situation than selling 75 copies for $75 or so.

Don will agree that many of the more specialized computer magazines (related to the Tandy Color Computer, for example) went out of business because the market anticipated by their advertisers never materialized. The advertisers didn't sell enough product to allow them to pay for their ads, let alone make any profit. The advertisers simply folded with their ad bills unpaid, and the magazines folded because they couldn't collect from bankrupt advertisers, nor could they attract new advertisers since it was realized that the market wasn't there.

So, with all the DISadvantages of the IBM and its clones, it has one advantage that overrides. It is STANDARD and it represents a large market. In many respects, PC-DOS or MS-DOS are no major improvement over FLEX. I consider them harder to use than FLEX. If they represent any improvement over FLEX, it is only in the area of tree structured directories. Certainly neither is as capable as OS-9, since they do not have facilities for multi-users or multi-tasking.

Unfortunately (in my opinion) if you want graphics or the capability to use some of the new laser printers (or even to make best use of some of the Epson (IBM) dot matrix graphics to do fancy type fonts), you have to go IBM or Apple Mac. An almost ex FLEX user friend of mine keeps sending me samples of what "Fontasy" on an IBM compatible can do with a dot matrix printer. Of course there is no reason someone couldn't do a nice graphics board for the 6809 systems (color even), and the software for multi fonts with a graphics printer wouldn't be difficult to do, but why bother? The market is too small to make it worthwhile. Such is progress.

I get involved in the industrial use of microprocessors a great deal. I see that area going 680XX pretty much. The biggest and best CAD systems run on Apollo or Sun 68020 machines, which are a real step upward from the 80286 based IBM clones. The 80386 systems are slow in coming, and they will suffer from precedents, as did the 6809. There is a lot of 80286 code now, and it will be easy to adapt it to the 80386 rather than write code to take advantage of the new processor, just as our first 6809 software was simply reassembled 6800 code that didn't take advantage of the extra registers and new instructions that made the '09 better and faster. A couple of years went by before software written specifically for the 6809 began to appear.

Let's face it. Our 6809 systems suffer from lack of identity. When someone asks me what kind of computer I have, I instinctively answer "You wouldn't recognize the name". The other day someone asked me at work if we had built our own development system. The SWTPc logo was not recognized. Only we diehards recognize Southwest Technical Products, GMX, and Peripheral Technology as "name brands" in computers. I truly hope all of these can find a market niche and remain living proof that small and innovative is better than big, mediocre and standard.

Only we diehards recognize

Southwest Technical Products, GMX, and Peripheral Technology as "name brands" in computers. I truly hope all of these can find a market niche and remain living proof that

small and innovative is better than big, mediocre and standard.

STRUCTURED LISTINGS

I've recently been translating a BASIC program to PL/9 and in the process I formed some opinions and made some observations that I would like to pass along to you. The program in question was published in BYTE to illustrate an algorithm for inverting a large matrix, but the purpose of the program is irrelevant to the discussion. The listing was just about 1 1/2 pages long, and written in a form that I consider to be almost unintelligible. My PL/9 version is just over 6 pages long, and my first reaction was that the PL/9 must surely be a lower level language that requires a great deal more typing in order to write a program. However, on further thought, I realized that the BASIC program is written in the style that surely must be a carry-over from the days when each byte of memory was crucial and we traditionally crammed as much information into each line of a BASIC program as was possible.

Listing A is extracted directly from the program in question. There are 12 statements in three lines.

TSC Extended BASIC allows statements to be indented from the line numbers, as do many BASIC interpreters. By following indenting rules established in other languages, i.e. everything included in a loop is indented, and putting one statement on a line, the fragment expands to the form in listing B. The 3 lines are now 13 lines, but surely the structure is more apparent and the doubly nested loop is much more readable.

Next I decided that the PL/9 version could also be crammed into three lines and followed the function of the original as closely as possible. The result is listing C which you will agree is just about as unintelligible as the original BASIC lines. Lastly, listing D shows the PL/9 translation done in the same structured manner as listing B.

The point of all this is that it is not the inefficiency of PL/9 that makes the listing much longer, but the approach of the programmer. If the aim is to make the program run fast, there is some reason for cramming the BASIC statements all together. Since the point of the article in question was to show an algorithm, I think there was little excuse for writing the program in the manner in which it was presented. The primary aim of any programmer trying to communicate ideas to others should to be to make the program as readable as possible! I've tried here to show that readability is not so much a function of the language used as of the programmer's use of a few simple rules that include:

1. One statement to a line.

2. Indenting to show the scope of loops.

3. Minimum use of GOTO statements.

We didn't discuss the last rule above. It is difficult to follow this rule in BASIC, though a REPEAT UNTIL or WHILE DO loop can be simulated with a GOTO and some REM statements. Perhaps we will explore writing structured programs in BASIC at some greater length in the next column.

```
LISTING A

FRAGMENT OF BASIC PROGRAM

100 FOR I=1 TO N:R0=0:S0=0:FOR J=1 TO N
110 R0=R0+ABS(A(I,J)):S0=S0+ABS(A(J,I)):NEXT J
120 X(1,I)=R0:E(1,I)=S0:NEXT I:T1=0:T2=0
```

```
LISTING B

SAME WRITTEN IN A STRUCTURED MANNER:
100 FOR I=1 TO N
110    R0=0
120    S0=0
130    FOR J=1 TO N
140       R0=R0+ABS(A(I,J))
150       S0=S0+ABS(S(J,I))
160    NEXT J
170    X(1,I)=R0
180    E(1,I)=S0
190 NEXT I
200 REM FOLLOWING ARE ACTUALLY PART OF NEXT LOOP
INITIALIZATION
210 T1=0
220 T2=0
```

```
LISTING C

UNSTRUCTURED PL/9 VERSION:

I=0; WHILE I<N BEGIN J=0; R0=0; S0=0 WHILE J<N
BEGIN
R0=R0+ABS(A(I*N+J)); S0=S0+ABS(A(J*N+I)) J=J+1;
END;
X(I)=R0; E(I)=S0; I=I+1; END; T1=0; T2=0;
```

```
LISTING D

STRUCTURED PL/9 VERSION:

I=0;
WHILE I<N BEGIN
    J=0;
    R0=0;
    S0=0;
    WHILE J<N BEGIN
       R0=R0+ABS(A(I*N+J));
       S0=S0+ABS(A(J*N+I));
       J=J+1;
    END;
    X(I)=R0;
    E(I)=S0;
    I=I+1;
END;

T1=0;
T2=0;
```

EOF

FOR THOSE WHO NEED TO KNOW **68 MICRO JOURNAL™**

The C Programmers Reference Source. Always Right On Target!

C User Notes

A Tutorial Series

By: Dr. E. M. 'Bud' Pass
1454 Latta Lane N.W.
Conyers, GA 30207
404 483-1717/4570
Computer Systems Consultants

INTRODUCTION

This chapter continues the discussion of the proposed ANSI C standard and the discussion of common problem areas in the use of the C language and its libraries.

PROPOSED ANSI C STANDARD

The header file "stdlib.h" declares several functions and one type. These functions and type provide string conversion, random number generation, memory management, and environment facilities.

The type is onexit_t, which is the type of the argument and the type of the value returned by the onexit function.

The string conversion functions are as follows

```
double atof(const char *nptr);
   converts the string pointed
   to by nptr to type double
int atoi(const char *nptr);
   converts the string pointed
   to by nptr to type int
long atol(const char *nptr);
   converts the string pointed
   to by nptr to type long
double strtod(const char *nptr,
  char **endptr);
   converts the string pointed
   to by nptr to type double
   and sets endptr to address
   of next character in string
long strtol(const char *nptr,
   char **endptr, int base);
   converts the string pointed
   to by nptr in radix base
   (2-36, 0=decimal) to type long
   and sets endptr to address
   of next character in string
```

The random number generation functions are as follows:

```
int rand(void);
   returns next element in a
   sequence of random numbers
   in the range 0 to 32767
void srand(unsigned int seed);
   seeds the random number
   generator method used by
   the rand function
```

The memory management functions are as follows

```
void *calloc(unsigned int nelem,
  size_t elsize);
   attempts to allocate sufficient
   space for nelem objects, each
   of length elsize, and returns
   either a pointer to this
   space (which is set to binary
   zeroes) or NULL
void free(void *ptr);
   deallocates the object
   pointed to by ptr
void *malloc(size_t size);
   attempts to allocate sufficient
   space for an object of length
   size size, and returns
   either a pointer to this
   space or NULL
void *realloc(void *ptr,
  size_t size);
   deallocates the object
   to which ptr points, then
   attempts to allocate sufficient
   space for an object of length
   size size, and returns
   either a pointer to this
   space or NULL
```

The environment functions are as follows:

```
void abort(void);
   terminates the current task
   unsuccessfully (usually after
   closing all open files) unless
   SIGABRT signal is being ignored
void exit(int status);
   terminates the current task
   successfully (after calling
   all functions registered by
   onexit and after closing
   all open files) and returns
   the value of status to the
   invoker of the current task
char *getenv(const char *name);
   searches an environment list
   for a string of the form
   name=value, and returns either
   a pointer to value or NULL
onexit_t onexit(onexit_t func);
   registers the function pointed
   to by func to be called without
   arguments, at program exit
   a pointer to value or NULL
int system(const char *string);
   passes the string pointed to
   by string to the host to be
   acted upon by a command
   processor, and normally
   returns a value indicative
   of the success of the
   execution of the string
```

The header file "string.h" declares several functions useful in manipulating character strings. For each function, a pointer provides the beginning address of a character string.

The string copying functions are as follows:

```
void *memcpy(void *s1,
   const void *s2, size_t n);
   copies n characters from
   the string pointed to by s2
   to the string pointed to by s1
   then returns the value of s1
void *memset(void *s,
   int c, size_t n);
   copies the value of c
   (cast to unsigned char)
   into the first n characters
   to the string pointed to by s
   then returns the value of s
void *strcpy(char *s1,
   const void *s2);
   copies characters from
   the string pointed to by s2
   to the string pointed to by s1
   until a NUL character is moved
   then returns the value of s1
void *strncpy(void *s1,
   const void *s2, size_t n);
   copies up to n characters from
   the string pointed to by s2
   to the string pointed to by s1
   until a NUL character is moved
   then returns the value of s1
```

The string concatenation functions are as follows:

```
void *strcat(char *s1,
   const void *s2);
   copies characters from the string
   pointed to by s2 to the end of
   the string pointed to by s1
   until a NUL character is moved
   then returns the value of s1
void *strncat(void *s1,
   const void *s2, size_t n);
   copies up to n characters from
   the string pointed to by s2
   to the end of the string pointed
   to by s1 until a NUL character is
   moved then returns the value of s1
```

The string comparison and length functions are as follows:

```
int memcmp(const void *s1,
   const void *s2, size_t n);
   compares n characters in
   the string pointed to by s2
   with the string pointed to by s1
   then returns a value indicating
   whether s1 is lexicographically
   less than, equal to, or greater
   than s2
int strcmp(const void *s1,
   const void *s2);
   compares characters in
   the string pointed to by s2
   with the string pointed to by
   s1 (both NUL-terminated)
   then returns a value indicating
   whether s1 is lexicographically
   less than, equal to, or greater
   than s2
```

int strlen(const void *s);
returns the number of characters in the string pointed to by s, not counting the terminating NUL character

int strncmp(const void *s1, const void *s2);
compares up to n characters in the string pointed to by s2 with the string pointed to by s1 (both NUL-terminated) then returns a value indicating whether s1 is lexicographically less than, equal to, or greater than s2

The string search functions are as follows:

void *memchr(const void *s, int c, size_t n);
compares up to n characters in the string pointed to by s for the first occurrence of c (cast to unsigned char) then returns either a pointer to the matching character in s or NULL

void *strchr(const char *s, int c);
compares characters in the string pointed to by s for the first occurrence of NUL or c (cast to unsigned char) then returns either NULL or a pointer to the matching character in s

size_t strcspn(const char *s1, const char *s2);
returns the length of the longest initial segment of the string pointed to by s1 not containing any of the characters contained in the string pointed to by s2 (both NUL-terminated)

char *strpbrk(const char *s1, const char *s2);
returns a pointer to the first location in the string pointed to by s1 containing any of the characters contained in the string pointed to by s2 (both NUL-terminated) or returns NULL

char *strrchr(const char *s1, int c);
compares characters in the string pointed to by s for the last occurrence of c (cast to unsigned char) then returns either or a pointer to to the matching character in s or NULL

size_t strspn(const char *s1, const char *s2);
returns the length of the longest initial segment of the string pointed to by s1 containing any of the characters contained in the string pointed to by s2 (both NUL-terminated)

char *strtok(const char *s1, const char *s2);
parses the string pointed to by s1 into tokens separated by characters in the string pointed to by s2 and returns on each successive call a pointed to a NUL-terminated section of s1 or NULL on the final call

The header file "time.h" declares several functions, three types, and one macro. It provides a facility for processing times and dates.

The macro is CLK_TCK, which is the number per second of the value returned by the clock function.

The types are as follows:

clock_t: type of the clock function,

time_t: type of the time function,

struct tm: broken-down time structure:
int tm_sec; seconds after minute
int tm_min; minutes after hour
int tm_hour; hours since midnight
int tm_mday; dayof month (1-31)
int tm_mon; month of year (0-11)
int tm_year; years since 1900
int tm_wday; days since sunday
int tm_yday; day of year (0-365)
int tm_isdst; dst if nonzero

The time measurement functions are as follows:

clock_t clock(void);
returns the number of CLK_TCK units since some implementation-dependent reference point in

```
    time or -1 (cast to clock_t)
    if no timer is available
time_t time(time_t *timer);
    returns the number of seconds
    since some implementation-
    dependent reference point in
    time or -1 (cast to time_t)
    if no timer is available;
    if timer is not NULL, this value
    is also placed into timer
```

The time manipulation functions are as follows:

```
char *asctime(const struct tm
  *timeptr);
    converts the time structure
    pointed to by timeptr into
    a representation of the time
    in the following format:
    dow mon dd hh:mm:ss yyyy\n
char *ctime(const time_t *timer);
    converts the value of time
      pointed to by timer into
      a representation of the time
      in the following format:
      dow mon dd hh:mm:ss yyyy\n
    double difftime(time_t time2,
      time_t time1);
      returns the number of seconds
      from time 1 to time2
    struct tm *gmtime(const time_t
  *timer);
    converts a time pointed to by
    timer into a time structure
    (expressed relative to GMT)
    and returns a pointer to this
    time structure
struct tm *localtime(const time_t
  *timer);
    converts a time pointed to by
    timer into a time structure
    (expressed relative to the
    local time zone) and returns a
    pointer to this time structure
```

C PROBLEM

The proposed ANSI C standard suggests that conforming C compilers generate warnings in at least the following situations which may arise in C programs:

a character constant contains more than one logical character

the character string /* is encountered in a comment

an implicit cast which causes a narrowing of data type is encountered, whch as the assignment of a double to an int

a function is called but no prototype has been supplied

the arguments in a function call do not agree in number or in type with those of the formal parameters in a prototype for that function

a declaration with no apparent effect is encountered

a value is given to an object of an enumeration type other than by assignment of an enumeration constant that is a member of that type

a statement may never be reached

a statement with no apparent effect is encountered

a block is entered at other than the beginning

a function has return statements with and without expressions or is of non-void type and has return statements without expressions

identical identifiers with external linkage disagree in type or length

For the C problem, suggest additional lint-like checks which might be applied by C compilers to assist the programmer in writing, debugging, porting, and maintaining C programs.

EXAMPLE C PROGRAM

Following is this month's example C program; it provides a function which parses a character string and returns pointers to the tokens comprising it. The test driver program reads a character string from standard input, parses it, and writes it to standard output. This function could be used in processing language text, generating a cross-reference, checking spelling, etc.

```
#include <stdio.h>

/*
 * parses character string into separate
 * arguments using whitespace to delimit
 * arguments, and stores the arguments
 * in an array which is passed.
 * returns address of pointer array.
 */

#define BLANK ' '
#define TAB '\t'
#define YES 1
#define NO 0

main(argc, argv)
int argc;
char **argv;
{
    char **args(), l[256], *a[128], **p;

    while (fgets(l, 256, stdin))
    {
        printf("%s", l);
        for (p = args(l, a); *p; ++p)
            printf("%s\n", *p);
    }
    exit(0);
}

char **args(buffer, argv)

/* string to be broken into arguments */
char *buffer;
/* pointer array for arguments */
char **argv;
{
    /* save address to be returned */
    char **ret_val = argv;
    /* save argument switch */
    int save = YES;

    /* eliminate leading white space */
    while (*buffer == BLANK ||
      *buffer == TAB)
      buffer++;
    while (*buffer)
    {
      if (save)
      {
        /* save argument in array */
        *argv = buffer;
        save = NO;
      }
      if (*buffer != BLANK &&
        *buffer != TAB)
        /* inside of argument */
        buffer++;
      else
      {
        /* at end of argument */
        *buffer++ = '\0';
        save = YES;
        /* inc pointer to next slot */
        argv++;
        /* eliminate white space */
        while (*buffer == BLANK ||
          *buffer == TAB)
          buffer++;
      }
    }

    /* put null pointer as last entry */
    *++argv = NULL;
    return (ret_val);
}
```

Following is a sample execution of the program listed above. In each group, the first line is sample input and the remaining lines are the output generated by the program.

```
11111 22222 33333 44444 55555
11111
22222
33333
44444
55555
now is the time for all good people to go
now
is
the
time
for
all
good
people
to
go

EOF
```

Basically OS-9

Dedicated to the serious OS-9 user.
The fastest growing users group world-wide!
6809 - 68020

A Tutorial Series

By: Ron Voigts
2024 Baldwin Court
Glendale Heights, IL
60139

OS-9 PLUMBERS

Done any plumbing lately? With OS-9? Unless you've tried laying a few OS-9 pipes, you may think it a useless idea. One time I did! I tried to configure an early system saving as much memory as possible. I eliminated whatever I could from the OS9Boot. Yes, PIPE, PIPER, and PIPEMAN went too. Some things can't be removed. You can't throw away the terminal modules, or the disk drive modules. Otherwise everything would come to grinding halt. But PIPE, PIPER and PIPEMAN don't effect normal system operation. I didn't miss them (right away!).

That is what I thought! I wrote a little program that used the standard paths to remove some control characters from a file. I planned to redirect a listing through it. I entered:

```
list file ! strip >newfile
```

and I got an error. I must have typed it wrong. I tried again. Same error! Something was wrong. I checked the modules in the commands directory. LIST was OK. So was STRIP. FILE listed fine by itself. But when I redirected LIST' output to STRIP, it wouldn't work. Then the lights went on. I was missing the Pipe modules. Suddenly the power of the system went down by a few notches. I immediately redid my boot disks and put the Pipe modules back. I never have regreted it. The difference in memory was about 500 bytes. About 2 pages.

The above example with LIST and STRIP is a good sample of creating a pipe. It does nothing more, then to take the output of the left program and pass it to the input of the right. Hence the name 'pipe', because it controls the flow of one program to another. An analogy is a plumber who lays pipes to control the flow of water. Only we are OS-9 plumbers.

The !, used to direct the flow of data, is not part of the pipe modules, but belongs to the OS-9 shell. The SHELL recognizes it as a pipe directive. It redirects the output of one program to another. For each ! encountered, it creates a pipe. Take the following example:

```
list afile ! strip ! upper >/p1
```

The SHELL creates two pipes, the flow goes from LIST to STRIP to UPPER. So the standard input is from LIST and not the keyboard. STRIP's standard output is redirected to UPPER. And its standard output goes to the printer. In this case, it is /P1. Since standard input and output are used, this line could easily be rewritten:

```
strip <afile ! upper >/p1
```

STRIP's input would come directly from afile and only one pipe would be created.

A really nice feature is that all the programs in the pipeline are executed concurrently. They are all synchronized so that the output of one cannot get ahead of the next program in the pipline. Flow moves along smoothly. The slowest program determines the flow rate.

Pipe applications are only limited by your imagination. They can be created for character manipulations, file formatting, and changing the output flow. Filters are popular. They remove undesired characters. The program STRIP I mentioned earlier was designed to remove non-printable characters. The OS-9 commands set includes one item called TEE. It redirects an output to a number of selected paths, including the standard one.

```
list afile ! tee afile.lst /p
```

directs AFILE to the standard output, a disk file called AFILE.LST and the printer.

The modules that control Pipes are interesting in themselves. As you probably already know there are three modules associated with a device. They are the file manager, the device driver and the device descriptor. For pipes, they are named PIPEMAN, PIPER, and PIPE. What makes them interesting is that there is no real device associated with them. Hence there is no need for a driver. However, a driver is necessary for continuity. Usually, an entry for a standard OS-9 driver would appear:

```
LBRA INIT
LBRAREAD
LBRA WRITE
LBRA GETSTA
LBRA SETSTA
LBRA TERM
```

Each line takes 3 bytes. PIPER's entry table is:

```
CLRB
RTS
NOP
```

repeated 6 times. The NOP is necessary to make each entry 3 bytes long. (Actually, NOP does not appear in the last line of code. It is in the others to create the 3 byte offset. But nothing is offset after the 6th entry point, so it is not needed. If this doesn't make sense, pretend it's there. Either way it will work.) This all means the driver does nothing. It only clears B to prevent an error report and returns. The Pipe Manager does all the work. Even parts of it do nothing. SEEK, GETSTT and SETSTT do nothing. CHGDIR, DELETE and MAKDIR return with error #208--Illegal Service Request. The CREATE and OPEN for Pipeman are the same code. All this makes it ideal for its job.

Creating a PIPE can be very interesting. The SHELL usually creates it for you, but you can make your own in your programs. For example, your program may want to direct its output through a filter at some appropriate moment. You could have it create a PIPE, redirect its standard output to the filter, FORK to the filter, and have the filter's input come from the PIPE. You can even get more creative if you want. Once you know how to do it.

There is one important concept to understand, before we get into creating a real PIPE. It is how to save the standard input and output paths while creating new ones. We use the system call I$DUP, which will duplicate a path. Here is the procedure. We duplicate a standard path, say #0, the standard input. Now we close #0, while saving the duplicate path number. Now with I$DUP again, we duplicate some other path that we want to be our new standard input. I$DUP assigns the lowest available path number, which has to be #0. The standard input is now from the a new path. When finished, we reverse everything we did. We close #0, duplicate the saved path which is our old standard input path and close its duplicate. Confused? Let's look at how to use with a PIPE.

I will outline how to do it. As you will see the method can be transferred to almost any language. You must be able to access IDUP, FFORK and open and close paths. We'll demonstrate using:

```
LIST FILE ! STRIP
```

Here is the outline.

1. OPEN /PIPE in UPDATE mode.
2. Duplicate path #1. Save the duplicate and Close #1.
3. Duplicate path of pipe. Now #1 will print to the PIPE.
4. FORK to LIST with FILE as its parameter.
5. Restore the standard output. Close #1. Duplicate the saved path. It becomes #1. Close its duplicate.
6. Duplicate #0. Save the duplicate and Close #0.
7. Duplicate the PIPE path and #0 will input from the pipe.
8. FORK to STRIP
9. Close #0. Duplicate the saved path. #0 is now restored and close its saved path.
10. Finally, close the path to the pipe.

Nothing to it, right? If you find this complicated, confusing and something you just don't want to do. Don't worry! The SHELL takes care of it all for you.

A WORD WRAPPING PIPE

Last month's column was concerned with the OS-9 Editor. If you've used it before, you know that is a decent little program for creating text files. It does lack many features of the more advanced word-processors. One of them is wrap around. Wrap around is a nice feature that lets you set the line length. Then whenever a line's length is exceeded, the remainder of it is placed on the next line without chopping a word apart. Usually this is done during the text entry, but it can be part of the paginator too. With this in mind, I created a PIPE that handles wrap around. It is the C program listed at the end of the column.

I called the program WRAP.C. It has 4 possible commands. These are in the text file that it processes. They are:

.NW No wrap around
.WR Wrap around
.LN=n Sets line length to n
.PP New paragraph

Anytime you want to change a feature use one of these on a line. The dot in front of each command tells WRAP that this is a control line. The dot must be in column one. .NW and .WR toggle on and off the wrap feature. .LN sets the maximum line length. .PP causes a new paragraph to be started. You may want to change this last command. Right now, it terminates any pending paragraphs and prints an extra carriage return. So paragraphs are separated by a blank line. You can change this if you like. Maybe you want your paragraphs indented a few spaces. Better yet add two new commands. .PPIN= can set the paragraph indent and .PPSP can set the blank lines between paragraphs.

Using this is relatively easy. Let's say you have a file named REPORT.TXT and the first two lines in the file read:

.WR
.LN=60

Entering the following line will print the file to another file with lines no more than 60 characters long.

LIST REPORT.TXT ! WRAP >REPORT.NEW

If a command line, one that starts with a dot (.), does not fit the model for the four commands that WRAP recognizes, it will be passed through. This means you can create more pipes to further process you text. Perhaps you create program to add margins and line numbers. Call it PAGER. Another one can justify lines, adding spaces between words to right justify line. It can be JUSTIFY. Now entering:

LIST REPORT.TXT ! WRAP ! PAGER ! JUSTIFY >REPORT.NEW

will create a sharp formatted version of the original text file with word wrap around, margins and justifying..

I'll leave these innovations up to you. As you can see using pipes can add a powerful new dimension to what you do with OS-9. Use you imagination. If you don't write C programs, try BASIC09 or Pascal. Whatever the language you work with, you can use it with PIPES. Just remember, always remember to use only the standard inputs and outputs.

LISTING

```
 1 /* Name: WRAP.C
 2    By: Ron Voigts
 3    Date: 15-OCT-86
 4    To compile: CC1 WRAP.C
 5
 6    This program will add wrap around to
       standard
 7    text files, like files created with the
 8    OS-9 Editor.  The features included here
      are
 9          .NW     wrap OFF
10          .WR     wrap ON
11          .LG=n   change line length to n
12          .PP     new paragraph            */
13
14 #include <stdio.h>
15 #include <ctype.h>
16 #define OFF 0
17 #define ON 1
18 #define FALSE 0
19 #define TRUE 1
20
21 char line[133];     /* input line */
22 char t[133];        /* temporary area */
23 direct int length,  /* line length */
24        position,    /* cursor position */
25        wrap;        /* wrap status  */
26
27 main()
28 {
29    /* default values */
30    wrap=OFF;
31    length=60;
32    position=0;
33
34    /* input a line and process it */
35    while (gets(line)!=NULL) {
36      if (line[0]=='.')
37          command(line);
38       else
39          if (wrap)
40             output(line);
41          else
42             printf("%s\n",line);
43    }
44 }
45
46 /* process line commands */
47 command(s)
48 char *s;
49 {
50    register int i;
51    int q;
52    /* copy t to s */
53    copy(s,t);
```

```
54
55    /* match lines to commands */
56    q=-1; /* default value */
57    if (compare(t,".WR")==0) q=0;
58    if (compare(t,".NW")==0) q=1;
59    if (compare(t,".PP")==0) q=2;
60    if (compare(t,".LG")==61) q=3;
61
62    /* process the command type */
63    switch (q) {
64    case 0:
65       wrap=ON;
66       position=0;
67       break;
68    case 1:
69       wrap=OFF;
70       if (position!=0)
71          cr(1);
72       break;
73    case 2:
74       if (wrap && (position>0))
75          cr(2);
76       else
77          cr(1);
78       break;
79    case 3:
80       length=atoi(s+4);
81       break;
82    default:
83       printf("%s\n",s);
84    }
85 }
86
87 /* send out carriage returns */
88 cr(i)
89 int i;
90 {
91    register int j;
92    for (j=0; j<i; j++)
93       printf("\n");
94 }
95
96 /* compares 2 strings */
97 compare(s,t)
98 char *s, *t;
99 {
100   register int i;
101   i=0;
102   while (s[i]==t[i])
103      if (t[i++]=='\0')
104         return(0);
105   return(s[i] - t[i]);
106 }
107
108 /* out put lines with wrap feature */
109 output(s)
110 char *s;
111 {
112   int space;
113   register int i;
114   /* set up initial values */
115   space=FALSE;
116   i=0;
117   /* process line dividing it into words */
118   while (*s!='\0') {
119      if (*s==' ')
120         space=TRUE;
121      t[i++]=*s++;
122      if ((*s!=' ') && (space)) {
123         t[i]='\0';
124         print(t);
125         i=0;
126         space=FALSE;
127      }
128   }
129   /* wrap up any leftovers */
130   if ((i!=0) && (t[0]!=' ')) {
131      t[i++]=' ';
132      t[i]='\0';
133      print(t);
134   }
135 }
136
137 /* output a word and adjust position */
138 print(t)
139 char *t;
140 {
141   if (position+strlen(t)>length) {
142      cr(1);
143      position=0;
144   }
145   printf("%s",t);
146   position=position+strlen(t);
147 }
148 /* copy command line to temporary line
149    with capitals if necessary        */
150 copy(s,t)
151 char *s, *t;
152 {
153   register int i;
154   for (i=0; s[i]!='\0'; i++)
155      t[i]=toupper(s[i]);
156   t[i]='\0';
157 }
158
```

EOF

FOR THOSE WHO NEED TO KNOW

68 MICRO JOURNAL™

The Macintosh™ Section

Reserved as a

A place for your thoughts

And ours.....

Mac-Watch

Spellswell A Spelling Checker & More

Back in those early days of pre-PC history there were a group of us wandering the countryside of do-it-yourself computing. Remembering those days? Forgotten by many, denied by some later-comers and embellished in tales by 'old timers', there was one common thread that held the whole thing together; we knew that better things were coming (and most of us were having fun). Applications to do something really useful with those klunky but marvelous machines.

Everyone had his, or her, personal desires and goals. Some wanted games, better games, faster games, harder games and on. Others wanted faster and higher resolution in math crunching, while still others wanted educational applications, business applications and the list was unending. For my part I wanted word processing. And we got it.

It didn't come fast. In fact it was a slow and painful process. We were the **'Orkin Man'** of the early micro days. There wasn't a digital or hexadecimal bug that we were not prepared to face and do battle with. Most times we won, but I knew some poor souls who buckled under and salvaged their sanity by dragging off to become insurance salesmen, auto mechanics or for a few, computer gurus. Me, I settled for writing and publishing stuff about them, rather than remain the *pure hacker*.

From nothing but hex keypads or toggle switches, to TV writers that posed as CRT drivers, and old (and some not so old TV receivers), to paper tape, to CRTs, to cassette tape (300-1200 baud), to intelligent terminals, to mini disk drives and then we tumbled into a world of delicious Winchester technology, 80 bit math crunchers, 32 bit address jumbos and other majestic digital marvels! And all that time I plodded along, and gladly embraced all the new soft/hardware that made my job easier. For in the end, the whole lot of them were nothing but *tools*, crude tools at first but slowly maturing.

Today we can prepare our entire magazine, all the way from idea to press ready signatures, with an economy micro costing less than what we once paid for just one Winchester drive. And that system has a CPU that is a direct descendent of those early day marvels of digital wonderment. Remember?

When the Apple Macintosh arrived we bought one of the very first. After all it had a 68000. It was a wonderfully laid out concept, just waiting for the rest of the micro-world to catch up. It had superior graphics, and the slowest disk access system we had ever seen. It was slower than our earliest kludged disk systems. It was practically an orphan in so far as support applications were concerned. There was MacWrite and MacPaint, both furnished by Apple, and that was about all. It was going to be awhile before the rest of the world caught up with the Mac.

Today things are different. The Mac has finally matured. Applications are springing up like wild onions in March. And the strange thing about it is that most of them are really quality products. Some sneaking over from the PC arena and others homegrown.

Spreadsheets you wouldn't believe, accounting applications that were too tough for the better minis just a few years back, applications that can find a place in about any business, anywhere. And they all are very affordable. But, for me the crowning finally came when *Desktop Publishing* arrived. We embraced it as if it was an end in itself. Of course, it isn't, it is still just a tool. However, it finally allowed us to do what we had wanted from those early days of adventure. *That is to make it accomplish our task!* Simple really, wasn't it?

Today I have at my disposal a variety of word processing applications. From the very simple to the mind boggling sublime. I can output to everything from modems, networks, servers, dot matrix printers to daisy wheel printers to nearly professionally perfect laser printers and much more.

However, for some reason which I find hard to understand, but only recently has there been a good, note, I say - *good* spelling checker. On the S50 bus we had really good ones years ago, but I guess we were ahead of our time.

In the Macintosh market community there are several spelling checkers to choose from. We tried several, but I guess I was spoiled by spellers running on some of our 68XX(X) systems, none seemed to quite cut the mustard. That is until we received one named **Spellswell**.

Spellswell does do it well. Fact is, we decided to review this product alone, as it is heads and shoulders above all the others, for the Mac, that we have used. Not only am I impressed with the product but I am especially impressed with the *'after'* support. A quality product from quality folks.

Spellswell is supplied with a 93,000 plus or a 60,000 or so word dictionary. Depending on the storage size of your disks drives, you have a choice. Also included in the package is a homonym checker. It like the main dictionary is user expandable and editable.

After your file is digested by the speller you are presented with several options for correcting the file or massaging the dictionaries. Among it's more significant features are the following.

1. As you edit the suspected words in your file you have several 'automatic' options. You may deal individually with each word each time it appears, or you may option to have them automatically dealt with each time thereafter. This feature can save a lot of key bashing.
2. Proper nouns that are not capitalized are detected and flagged for your attention. Capitalization, fonts and other editor options are maintained for the entire document.
3. Incorrect hyphenation is flagged and questioned. Missing apostrophes are dealt with in a like manner. Missing spaces between words and sentences are detected, as well as flagging for action duplicated words, such as the the, etc.
4. Abbreviations that are not correct are caught and all can be corrected immediately.

In addition to plain text documents, Spellswell will scan and correct documents from most all Macintosh editing applications, including MacWrite 4.5, WORD 1.0 and 1.05, WORKS, Living Videotext More, ThinkTank, Jazz and many more (but not Telescape, or at least our version).

At this point we ran into the one and only problem experienced with Spellswell. The files that were ported into the Mac via Telescape (a modem program) would not agree with the internals of Spellswell. And that seemed to be a big'un as we use(ed) Telescape as our primary modem program.

As soon as the problem was discovered (real soon like) I had a call placed to the publisher of Spellswell to alert them to our findings. Having dealt with several other Macintosh product vendors, I was surprised at the reception my call received.

Immediately, as soon as I told the young lady answering the telephone that I had a problem to report, I was connected with someone who could discuss the problem with me. No waiting, no telephone symphony, immediate action!

After I explained the problem the fellow on the other end request that I send the file to them, Federal Overnite Express, COD. We did. And believe it or not but within 18 hours of dispatching the disk to them they were back on the phone to acknowledge that indeed there was a problem and that it would be reckoned with immediately! Also they asked us to send them another, FedX and COD but this time in raw format. We did and less than 24 hours later they called again to say that the problem had been identified and that a, **get this,** *a special filter program would be written immediately and sent to us to filter out the extra nulls our modem program was putting in the file!*

However, by this time I had dug into the document file and discovered that extra nulls were being introduced into our ported files, following each $0D carriage return. We had already switched modem programs to the one furnished in our SideKick collection, named MacTerm. It works like a good program should.

Even though we cured the problem by switching to another program, it was a real treat being dealt with as we were. I cannot say too much about the excellent support the folks who supply Spellswell went to. Isn't it a shame that others don't do likewise.

Spellswell is not a DA. It is a full blown application. You close your text file and then select it from within Spellswell. You must, one time, tell Spellswell where the dictionary is on the disk, and from then on it remembers. Spellswell then loads in your document, up to about 8,000 words at a

time. Documents larger are checked in 8,000 word blocks until finished. This makes it fairly fast.

As each suspect word is encountered you have several options, as stated above. You are prompted to skip, add, replace or delete the suspected word. As each suspect word is presented for action a dictionary scroll box will be filled with words that fall close to the spelling of the suspect word. Also a 'suggested replacement' window contains a word the speller feels you might want to use. You have the option of editing that window or click selecting a word from the dictionary window. Either way gets your suspect word corrected, provided of course that it really is wrong. From the dictionary window you can scan the entire dictionary. All standard Macintosh protocols hold for windows and editing actions while using Spellswell. Replacement words not in the dictionary will be added to the dictionary if you so reply to it's prompt requesting such action.

There is a Short Cuts menu. There you can permanently opt to have all the normal 'skip, add, replace and delete' prompts apply to all reoccurrence of suspect words. Each can be momentarily de-selected and you will get the 'replace all occurrences' question again.

When words are added to the dictionary you are prompted (if you have not select auto from the short cuts menu) for different suffixes and capitalization of your word to be added. If Quick Short Cut is selected no box is presented, however, the word is saved with whatever capitalization it had in your document. When it is tested capitalization is not considered, unless it is used as a replacement, then if you checked it to require capitalization, it will be placed into your document capitalized. Proper nouns, such as days of the week, or the names of months are forced to capitalization. This may be disabled by selection in the Short Cuts menu.

The delete command can be used on the dictionary window to remove words from the dictionary. Believe it or not, but this is a feature not supported by all the Mac spelling checkers.

Abbreviations, contractions and diacritical marks are all handled in a normal way. Characters such as Á and È are dealt with properly. Words containing diacritical marks may be replaced or corrected but are not added, as such, to the dictionary.

While homonym checking is fine for some, I keep it turned off. There is just too much asking me if it really should be - to, as used in to bed, or too, as in too much, or two as in two times is enough. This is a nice touch, but I pass on this feature. Although you can suppress certain groups of words being prompted or simply stop checking them all. Or they are just forgotten if checked off in the Short Cuts menu. A homonym dictionary is maintained, and can be added to via the program or edited by any editor, as it is maintained as a regular text file.

When closing, Spellswell notifies you of the total number of suspect words encountered and the total number of words contained in your document. Also you have the option of a file to be maintained for that particular document containing all the 'skipped' words. If saved, and the document re-edited, you will not have to go through the 'skip' process all over again.

Most of the functions used in correcting your document have keyboard-command key options. Saves a lot of mouse clicking once learned. Me, I caught on after one session with a 12,000 word document.

Spellswell is *not* copy protected. A scheme that has put grey hairs on most who have hard disk drives. Also registered owners are notified of bugs and feature updates.

When we started to review spelling checkers, this was one of several. All the others were either so flawed or had other traits I felt were undesirable, I decided to confine our review to this one particular product. *It is, by far, superior to any other we have seen or used.* Not only in operation, but in support. To me, support is of prime consideration. Did you ever have to sit there, telephone hung to your ear, for what seemed like hours, maybe listening to some lousy excuse for music, as your long distance meter kept rolling merrily along? If so, you will soon learn to appreciate good support. We got it here in spades!

If some equal or better comes along, we will let you know. But, for now, *Spellswell is the only one that does it all so painlessly! And they seem to care!*

Spellswell is available directly from:

Greene, Johnson, Inc.
321 Alvarado ~ Suite H
Monterey, CA 93940
(408) 375-2828

Or at most good Apple software retailers.

The price: $74.95

A Staff Review

EOF

FOR THOSE WHO NEED TO KNOW
68 MICRO JOURNAL™

FORTH

A Tutorial Series

By: R. D. Lurie
9 Linda Street
Leominister, MA 01543

GRAPHICS WITH FORTH

It was very easy to write graphics commands for FORTH. All I had to do was copy the ANSI commands from the instruction manual and there it was! I could draw circles and lines and boxes and you-name-ituntil my fingers got tired. And I could write in bright, blinking, underlined, reversed video text to the point that the whole thing was unreadable. It was a lot of fun! First, a little background is in order. My terminal is actually a Z80 computer running CP/M. No, I have not defected or taken leave of my senses. Instead, I have taken advantage of the situation created by IBM. This is one of those "standard" CP/M office computers which have been rendered obsolete by BIG BLUE. It has all of the usual bells and whistles of a good desk-top system, but it also comes with a program designed specifically to convert it into a smart terminal. This includes being able to store received text directly into a disk file, and to send a disk file directly to the host computer. Of course, it was expected that the host would be a mainframe or a mini, but I found that it works very well with my GIMIX, etc. system. About the only change I need to make in the GIMIX software is to add X-ON/X-OFF capability, since this is what the terminal program expects to see at the other end of the line. Right now, I am limited to 2400 baud between the two units, but the software patch should allow 9600 baud, possibly even 19200 baud. I definitely recommend this route to any one else looking for a first-class terminal for a very low price. Everything should be nearly perfect as soon as I can get a FORTH-83 for CP/M.

ANSII Graphics ESC Sequences

Now I know what Ron Anderson was talking about a few months ago with his complaints about the indeterminate length of the ANSI ESC sequences for controlling a computer terminal. Some of the graphics commands can run from 3 to 15 characters in length, depending on the details of the specific request. I hate to think of the problems of handling that variable length in most languages, but it is a snap in FORTH! You just call the command string, with the coordinates on the Data Stack, and let the computer do the rest. All of the ANSI codes of interest, here, consist of a prefix, one or more numbers separated by semicolons, and a suffix. The numbers are the parameters of the command; that is, where, how, or what kind? The suffix is the way the terminal can know what to do with the numbers. For instance, the only difference between "draw a line to X,Y" and "peek screen RAM at X,Y" is the difference between "L" and "R" as suffixes.

The main inconvenience with the ANSI commands is that you cannot embed any sort of null-character within the command string. This means that you cannot pad the string with <SP> or <NULL> or anything else. It is literally a case of what you send is what you get.

Fortunately, FORTH lets you control string formatting in a dynamic manner, so that you do not have to send any <SP>'s, if you don't want to. Normally, the only thing that can change within a particular string is the value of one or more graph coordinates. Therefore, if you just push your X and Y values onto the Data Stack as double numbers, the words <# #S #> TYPE will send exactly what you need to the terminal. I'll discuss the detail of this in a moment.

Command Prefix

Since this terminal actually uses only a subset of the possible ANSI commands, I did not have to worry about providing for a tremendous variety of different command prefixes. In fact, only 2 were necessary for graphics, and 3 more are required for text. This subset is exactly that required to

emulate the DEC VT100 terminal, which has been a standard for the mini industry for years. More about that at another time.

All commands begin with <ESC> and one more character, which, when taken together, form a prefix for any command within a specific group of commands. For example, the two-character combination of <ESC><?> always signals the beginning of a graphics command. Other combinations signal other jobs.

This installment has the program examples shown in their original FORTH screen format. This will make it easier to refer to different sections of the program.

Screen #107 has the definitions of .<ESC> and .<?> . The definitions are named this way because, in FORTH, the . is normally used to prefix the name of any command which causes printable output; and < and > are often used in text to indicate a single key or key-stroke; so I have used them in what I think is a logical manner.

The definition of .<ESC> is obvious, since we simply want to transmit the ASCII code 27 (decimal); however, .<?> was done differently in order to emphasize the purpose of the definition. I expect this self-documentation to be a great help in understanding the code months from now.

The other two words in Screen #107 are meant for the text cursor and a general purpose screen clearing command.

Transmitting Numbers

Since those commands which use more than one number require that the numbers be separated by a semicolon, it was necessary to define .<;> (which is also used in the prefix of many text commands). Screen #108 shows that this definition is exactly like that of .<?> in screen #107. Notice that careful attention must be paid to typing definitions of this class; a slip of the finger could cause an unwanted <SP> to be sent, and that bug could be the very devil to find!

Remember that sometimes a command has only one parameter, but a command can often have 2, or even more, parameters. As a result, it is convenient to have two routines, one for those cases requiring a semicolon, and one for those cases not requiring a semicolon. Notice the difference between NUM-OUT and NUM-OUT; . The semicolon is part of the name of the second word!

The first thing that happens in NUM-OUT is that a 0 (zero) is pushed onto the Data Stack. This automatically converts the top integer into a double number, which is required for the built-in number-to-ASCII conversion definitions. The value of a positive integer is not changed by this technique, but it will not work for a negative integer.

I did not bother to protect against a negative integer being on the Data Stack, since the graphics routines do not make any provision for negative arguments. I believe that this sort of protection, if desired, should be put in the calling routines, since it is not always needed; so, why waste the time?

The phrase <# #S #> TYPE will convert the double number on top of the stack into an ASCII string and send it to the display. No extra fill characters are sent with this form of the command.

NUM-OUT; differs only in that the number is followed by a semicolon. Nothing else is sent.

Graphics Cursor

All graphics work on this terminal is predicated on there being an invisible graphics cursor. The position of this cursor determines the center of circles and arcs and the beginning of lines. Therefore, there must be a command to control the position of this cursor. I have used GRXY , standing for GRaphics XY, as the name of this definition. It is entered with two integers, the X and Y coordinates, on the Data Stack. The value of Y must be on top of the stack (the last one entered). This is simply a bit of user-friendliness, because that is the order we learned to use in school for graph coordinates.

One key to good FORTH programming is the technique of "factoring" definitions. I have already shown one obvious case of this in the definition of NUM-OUT; , where most of the code is the same as for NUM-OUT . A similar case can be made for factoring the definition of GRXY , since the next eight definitions all begin with the same code, which is the heart of GRXY . I have named this factored portion of GRXY as (GRXY) , which is the first definition in screen #109.

Unfortunately, the actual operation of moving must be done first on the X axis. Therefore, the first step in the definition is SWAP , which simply exchanges X and Y on the Data Stack. This is followed by transmitting the ESC? preamble, the X coordinate, a semicolon, and the Y coordinate.

The definition GRXY simply calls (GRXY) and then transmits the letter "C", which terminates the command. The "C" is actually the portion of the command which tells the terminal that we want the graphics cursor moved to X,Y.

By the way, be sure to remember that the graphics cursor and the text cursor are not related in any way! They move completely independently of each other and may, or may not, occupy the same screen coordinates.

Marking a Point

My terminal allows five different ways of marking a point on the screen. The alternatives are ".", "*", "+", "o", or "x". In all five cases, the procedure is the same, except for the terminating sequence. The details are shown in screens #109-111.

DOT is unique, among the markers, in that it does not need a parameter. This, obviously, makes DOT the default case. Therefore, we only need transmit (GRXY) and "M" in order to mark the screen with a ".".

Since the other four point markers all require a parameter, as well as the "M" terminator, a semicolon must be included in the transmitted string. The easiest solution to this requirement is the addition of a semicolon to the definition of (GRXY) to make (GRXY); . Remember that the semicolon is part of the name of this definition.

Line

No graphics program would be complete without a convenient way to draw a line. Since LINE is a part of the EDITOR vocabulary, LINE could also be used here, but I wanted to emphasize the purpose of the definition, so I chose GLINE for its name. GLINE stands for Graphics LINE.

It appears that GLINE , like DOT , was considered to be a default operation, so no parameters are required, beyond the expected X and Y coordinates. As a result, the definition consists only of (GRXY) and the terminator "L". Notice that no semicolon is required.

Examples

I have included a couple of simple examples of using GLINE in graphics definitions, BOX and SOLID-BOX . BOX is written in FORTH-83 and uses PICK so that there is no need to provide for external storage. In contrast, SOLID-BOX uses external variables, so its definition could be used in FORTH-79, FORTH-83, and fig-FORTH (if you initialize the variables). Actually, I think that you would be using variables for storing the coordinates, anyway, so the technique used in SOLID-BOX would probably be more convenient, in the long run.

BOX just draws a rectangle from the supplied coordinates. It makes no difference which X,Y coordinate pair is entered first; the drawing of the box will begin and end with the first pair of coordinates entered. I think that the definition is adequately explained by the comments on the screen, so I will not take up much time with any additional description. I do want to point out that the "DS" in the comments refers to the Data Stack, and the sequence of X1, Y1, etc. is the order they follow on the stack; the right-most value is the next one available (the stack top).

SOLID-BOX draws a box which is filled by the foreground color. The box is drawn by repetitively drawing a horizontal line from left to right and from top to bottom. The DO ... LOOP limit is initialized by a subtraction, so you must make sure that the coordinates are entered in the order which will yield a positive number. There is no error trapping, as I did not want to complicate the example; add it if you want it.

The algorithm is quite simple. For each iteration of the loop, the graphics cursor is moved to the next available position, and a line is drawn from there to the corresponding point at the righthand side of the screen.

The rather cryptic phrase 1000 0 DO LOOP has nothing directly to do with drawing the figure. As I said in my opening remarks, I have not yet perfected the communication between the computer and the terminal; this phrase is here strictly for the purpose of slowing down the drawing operation so that the computer doesn't overrun the terminal.

I debated with myself whether or not to leave it in the example. I finally decided that some of you may have the same kind of problem that I am having, and, by leaving the extra loop in place, you could see a convenient technique and place for slowing your system. Leave it out, at first, and only add it if you actually find that you need it.

Well, that is enough on graphics, for now. The point in showing all of this is to illustrate that it is easy to send variable length data strings to a display device from FORTH. Even if you are not interested in graphics, the technique can be useful for any sort of program output.

I have also tried to show some examples of "factoring" FORTH definitions, so that repetitive elements can be combined into a few reusable definitions. That way, program readability is improved and RAM consumption is reduced.

OVERKILL

The joke is on me! Some friends at the local FIG chapter pointed out a problem with the C functions I included in an earlier column. Essentially, the problem is that I committed one of the cardinal sins of FORTH programming--I tried to spruce up a definition with overkill. I made the return of the FALSE flag so much more complicated than necessary that I made the definitions execute much more slowly than they could run. This is no problem for an occasional reference to a definition, but would be readily noticed in a program to scan a long text.

Fortunately, the solution to the problem is so simple that it really needs no explanation, only an example. Screen #114 shows a revised definition for ?ISSPACE which executes much quicker than the old definition. To make the change, you only need to delete the last line of the definition, put the semicolon after ENDCASE , and add FALSE SWAP just before ENDCASE .

If anyone has any more improvements, I would like to hear from you, please.

```
flist col0.lst

SCR # 107
  0 \ GRAPHICS
  1
  2 : .<ESC>  ( -- )                                  \ RDL  12/10/86
  3     27 EMIT ;
  4
  5 : .<?>    ( -- )                                  \ RDL  12/10/86
  6     ." ?" ;
  7
  8 : HOME    ( -- )                                  \ RDL  12/10/86
  9     .<ESC> ." [H" ;           \ text cursor
 10
 11 : CLS     ( -- )                                  \ RDL  12/24/86
 12     .<ESC> ASCII c EMIT 0 EMIT ;
 13
 14 -->
 15

SCR # 108
  0 \ GRAPHICS
  1
  2 : .<:>    ( -- )                                  \ RDL  12/10/86
  3     ." ;" ;
  4
  5 : NUM-OUT     ( n -- )                            \ RDL  12/16/86
  6     0                                 \ make a double number
  7     <# #S #> TYPE ;                   \ output the number
  8
  9 : NUM-OUT;    ( n -- )                            \ RDL  12/26/86
 10     NUM-OUT
 11     .<:> ;
 12
 13 -->
 14
 15

SCR # 109
  0 \ GRAPHICS
  1
  2 : (GRXY)  ( X Y -- )                              \ RDL  12/16/86
  3     SWAP                              \ proper order for X & Y
  4     .<ESC> .<?>                       \ preamble
  5     NUM-OUT;                          \ X coordinate
  6     NUM-OUT ;                         \ Y coordinate
  7
  8 : GRXY    ( X Y -- )                              \ RDL  12/16/86
  9     (GRXY)
 10     ." C" ;                           \ terminate sequence
 11
 12 : DOT    ( X Y -- )                              \ RDL  12/26/86
 13     (GRXY)
 14     ." M" ;                           \ terminate sequence
 15 -->

SCR # 110
  0 \ GRAPHICS
  1
  2 : (GRXY);     ( X Y -- )                          \ RDL  12/26/86
  3     (GRXY)
  4     .<;> ;
  5
  6 : .<+>    ( X Y -- )                              \ RDL  12/26/86
  7     (GRXY);
  8     ." 10M" ;                         \ terminate sequence
  9
 10 : .<+>    ( X Y -- )                              \ RDL  12/26/86
 11     (GRXY);
 12     ." 11M" ;                         \ terminate sequence
 13 -->
 14
 15

SCR # 111
  0 \ GRAPHICS
  1
  2 : .<0>    ( X Y -- )                              \ RDL  12/26/86
  3     (GRXY);
  4     ." 79M" ;                         \ terminate sequence
  5
  6 : .<X>    ( X Y -- )                              \ RDL  12/26/86
  7     (GRXY);
  8     ." 88M" ;                         \ terminate sequence
  9
 10 : GLINE  ( X Y -- )                              \ RDL  12/26/86
 11     (GRXY)
 12     ." L" ;                           \ terminate sequence
 13
 14 -->
 15

SCR # 112
  0 \ GRAPHICS
  1
  2 : BOX     ( X1 Y1 X2 Y2 -- )                      \ RDL  12/26/86
  3     3 PICK 3 PICK             \ DS=X1 Y1 X2 Y2 X1 Y1
  4       GRXY                    \ set starting point - X1,Y1
  5       1 PICK 3 PICK           \ DS=X1 Y1 X2 Y2 X2 Y1
  6       GLINE                   \ draw line to X2,Y1
  7       2OVF                    \ DS=X1 Y1 X2 Y2 X2 Y2
  8       GLINE                   \ draw line to X2,Y2
  9       3 PICK OVER             \ DS=X1 Y1 X2 Y2 X1 Y2
 10       GLINE                   \ draw line to X1,Y2
 11       2DROP                   \ DS=X1 Y1
 12       GLINE ;                 \ draw line to X1,Y1
 13
 14 -->
 15

SCR # 113
  0 \ GRAPHICS
  1
  2 VARIABLE X1     VARIABLE X2
  3 VARIABLE Y1     VARIABLE Y2
  4
  5 : SOLID-BOX   ( X1 Y1 X2 Y2 -- )                  \ RDL  12/26/86
  6     Y2 !  X2 !  Y1 !  X1 !              \ store coordinates
  7     Y2 @ Y1 @ - 0 DO                    \ DO ... LOOP limits
  8       X1 @ Y1 @ I + GRXY                \ start of line
  9       X2 @ Y1 @ I + GLINE               \ draw line
 10  1000 0 DO LOOP
 11     LOOP ;
 12
 13
 14
 15

SCR # 114
  0 : ?ISSPACE    ( char -- boolean )                 \ RDL  01/10/86
  1     CASE
  2       32 OF TRUE ENDOF                  ( <SP>                )
  3       09 OF TRUE ENDOF                  ( <TAB>               )
  4       10 OF TRUE ENDOF                  ( <LF>                )
  5       12 OF TRUE ENDOF                  ( <FF>                )
  6       13 OF TRUE ENDOF                  ( <CR>                )
  7       FALSE SWAP
  8     ENDCASE ;
  9
 10 \ ------------------------------------------------------------
 11
 12 \ Test for <SP><TAB><LF><FF><CR>)
 13
 14
 15
```

EOF

FOR THOSE WHO NEED TO KNOW

68 MICRO JOURNAL™

Build A RS-232 BREAKOUT BOX

By:
Barry Balitski
151 Midglen Place S.E.
Calgary, Alberta T2X 1H6

Oh, boy! I just got my new printer! Now let's get it hooked up and see how it works. Gee, I wonder if I have a cable that will work? Let's try this one...no won't work. Well, let's pull out the manual and see what we need. Oh no, the manual is a very poor translation of Japanese and after reading it twenty times I still can't understand it. Now where's the manual for my computer? I'll wade through that and see if I can find the RS-232 interface pin designations. Hmm... maybe if I hook this up and connect these with some jumpers and cut this one and splice it to here. Oh! Oh! Was that a spark I saw when the jumper slipped off this terminal and hit another? Well let's spend another three hours to find out that I blew the RS-232 driver chip.

Does this sound familiar to anyone??? After my latest episode of the "unstandard" RS-232 standard I decided it was time for me to go and buy a "RS-232 breakout box". A RS-232 breakout box is a device which has a ribbon cable with a male DB-25 plug on one end, another ribbon cable with a female DB-25 socket on it and a box with two sets of terminals which connect to either ribbon cable. The breakout box is used by connecting it to each device and placing jumpers between the connector terminal pairs until you get the proper connection.

A stroll to my favorite electronics supplier left me in shock as many of these exceeded the price of my printer. Sure, there were some very nice ones with pretty lights, logic probes and many other real nice goodies. But the price was way out of line for a hobbyist. This set me to thinking about how I could use their "caviar" ideas and build myself a "bread and butter" model. I began to round up the necessary parts; ten feet of twenty-five conductor ribbon cable, one each of a crimp type DB-25 male and female connector, a 4x6 plastic box (remember plastic!), and the heart of the whole thing, a 2 row by 34 contact solderless breadboard.

To assemble the whole thing what you need to do is as follows: Make sure the solderless breadboard is the type with a paper backing. This paper is quite easily removed from the breadboard by lifting a corner and peeling it away. Caution! Do not let the paper adhesive pull the small metal strips which will be exposed out of their slots. If necessary you can use a small amount of alcohol to soften the adhesive. These small metal strips will be where you will solder each of the 25 stripped wires on each ribbon cable. I cut a slot in each side of the box to run the ribbon cables in and cut a larger slot in the top which will be where the stripped ends of the wires meet the breadboard. Run the ribbon cables in their respective sides and up through the top. Strip all the wires and solder them carefully to each metal strip exposed on the bottom of the breadboard.

I chose to skip one conector every five, removed the metal strip and filled the empty chamber with white epoxy so I could not insert a wire from the top. I found this to be convenient as you can count by fives and you're less likely to be off on your count, but do it however you like. Guide the ribbon cables down and place the breadboard over the large cutout. Fasten the breadboard to the plastic box with the mounting holes (I hope yours has them). Crimp the DB-25 connectors to the cable by aligning the cable correctly in the connector and clamping the connector closed in a vise. The only way to correctly do the crimping is in a vise, there are other ways but I don't think you want to take a chance with open or shorted contacts. That's all there is to it. Now what you do is hook the breakout box between the two devices and make your trial connections with # 22 gauge solid wire from the one side of the breadboard to the other.

(615)842-4600 Telex 5106006630

South East Media

5900 Cassandra Smith Rd.
Hixson, TN 37343

for information
call (615) 842-4601

CoCo OS-9™ FLEX™

SOFTWARE

SPECIAL

K-BASIC

K-BASIC under OS-9 and FLEX will compile TSC BASIC, XBASIC and XPC Source Code Files.

$AVE $100

K-BASIC now makes the multitude of TSC XBASIC Software available for use under OS-9. Transfer your favorite BASIC Programs to OS-9, compile them, Assemble them, and BINGO -- useable, multi-precision, familiar Software is running under favorite Operating System!

!!! SPECIAL ~~$199.00~~ $99.00 !!!

SCULPTOR

Full OEM & Dealer Discounts Available!

THE SCULPTOR SYSTEM

Sculptor combines a powerful fourth generation language with an efficient database management system. Programmers currently using traditional languages such as Basic and Cobol will be amazed at what Sculptor does to their productivity. With Sculptor you'll find that what used to take a week can be achieved in just a few hours.

AN ESTABLISHED LEADER

Sculptor was developed by professionals who needed a software development tool with capabilities that were not available in the software market. It was launched in 1981 and since then, with feedback from an ever increasing customer base, Sculptor has been refined and enhanced to become one of the most adaptable, fast, and above all reliable systems on the market today.

SYSTEM INDEPENDENCE

Sculptor is available on many different machines and for most operating systems, including MS DOS, Unix/Xenix and VMS. The extensive list of supported hardware ranges from small personal computers, through multi-user micros up to large minis and mainframes. Sculptor is constantly being ported to new systems.

APPLICATION PORTABILITY

Mobility of software between different environments is one of Sculptor's major advantages. You can develop applications on a stand-alone PC and – without any alterations to the programs – run them on a large multi-user system. For software writers this means that their products can reach a wider marketplace than ever before. It is this system portability, together with high-speed development, that makes Sculptor so appealing to value added resellers, hardware manufacturers and software developers of all kinds.

SPEED AND EFFICIENCY

Sculptor uses a fast and proven indexing technique which provides instant retrieval of data from even the largest of files. Sculptor's fourth generation language is compiled to a compact intermediate code which executes with impressive speed.

INTERNATIONALLY ACCEPTED

By using a simple configuration utility, Sculptor can present information in the language and format that you require. This makes it an ideal product for software development almost anywhere in the world. Australasia, the Americas and Europe – Sculptor is already at work in over 20 countries.

THE PACKAGE

With every development system you receive:
- A manual that makes sense
- A periodic newsletter
- Screen form language
- Report generator
- Menu system
- Query facility
- Set of utility programs
- Sample programs

For resale products, the run-time system is available at a nominal cost.

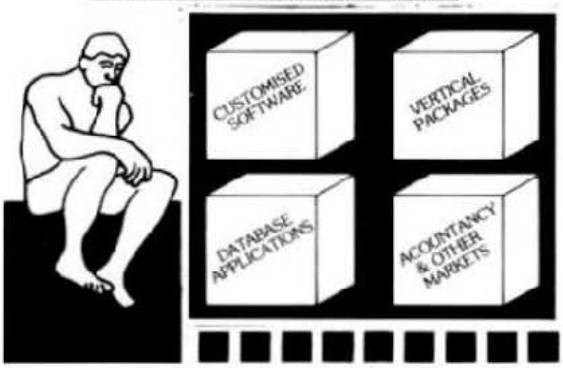
Facts **Features**

DATA DICTIONARY

Each file may have one or more record types described. Fields may have a name, heading, type, size, format and validation list. Field type may be chosen from:
- alphanumeric
- integer
- floating point
- money
- date

DATA FILE STRUCTURE
- Packed, fixed length records
- Money stored in base currency unit
- Dates stored as integer day numbers

Sculptor for 68020 OS-9 & UniFLEX $995



INDEXING TECHNIQUE

Sculptor maintains a B-tree index for each data file. Program logic allows any numbers of alternative indexes to be coded into one other file.

INPUT DATA VALIDATION

Input data may be validated at three levels:
- automatic by field type
- validation list in data dictionary
- programmer coded logic

ARITHMETIC OPERATORS

-	Unary minus
*	Multiplication
/	Division
%	Remainder
+	Addition
-	Subtraction

RELATIONAL OPERATORS

=	Equal to
<	Less than
>	Greater than
<=	Less than or equal to
>=	Greater than or equal to
<>	Not equal to
and	Logical and
or	Logical or
ct	Contains
bw	Begins with

SPECIAL FEATURES
- Full date arithmetic
- Echo suppression for passwords
- Terminal and printer independence
- Parameter passing to sub-programs
- User-definable date format

MAXIMA AND MINIMA

Minimum key length	1 byte
Maximum key length	160 bytes
Minimum record length	3 bytes
Maximum record length	32767 bytes
Maximum fields per record	32767
Maximum records per file	16 million
Maximum files per program	16
Maximum open files	Operating system limit

PROGRAMS
- Define record layout
- Create new indexed file
- Generate standard screen form program
- Generate standard report program
- Compile screen form program
- Compile report program
- Screen form program interpreter
- Report program interpreter
- Menu interpreter
- Query facility
- Reformat file
- Check file integrity
- Rebuild index
- Alter language and date format
- Setup terminal characteristics
- Setup printer characteristics

SCREEN FORM LANGUAGE
- Programmer defined options and logic
- Multiple files open in one program
- Default or programmer processing of exception conditions
- Powerful verbs for input, display and file access
- Simultaneous display of multiple records
- Facility to call sub-programs and operating system commands
- Conditional statements
- Subroutines
- Independent of terminal type

* Full Developement Package
** Run Time Only
*** C Key File Library

MUSTANG-020 Users - Ask For Your Special Discount!

	*	**	***		*	**	***
MUSTANG-020	$995	$199	$595	**IBM PC/XT/AT MSDOS**	$595	$119	$595
OS/9 UniFLEX	"	"	"	**AT&T 3B1 UNIX**	"	"	"
IBM Compatibles	"	"	"	**SWTPC 68010 UniFLEX**	$1595	$319	$798
Tandy CoCo III	"	"	"	**SWTPC 68010 UNIX**	$1990	$398	$995

...Sculptor Will Run On Over 100 Other Types Of Machines...
...Call For Pricing...
!!! Please Specify Your Make of Computer and Operating System !!!

VISA

DISASSEMBLERS

SUPER SLEUTH from Computer Systems Consultants Interactive Disassembler; extremely *POWERFUL!* Disk File Binary/ASCII Examine/Change. Absolute or FULL Disassembly. XREF Generator, Label "Name Changer", and Files of "Standard Label Names" for different Operating Systems.

Color Computer SS-50 Bus (all w/ A.L. Source)
CCD (32K Req'd) Obj. Only $49.00
F, $99.00 - CCF, Obj. Only $50.00 U, $100.00
CCF, w/Source $99.00 O, $101.00
CCO, Obj. Only $50.00
OS9 68K Obj. $100.00 w/Source $200.00

DYNAMITE+ -- Excellent standard "Batch Mode" Disassembler. Includes XREF Generator and "Standard Label" Files. Special OS-9 options w/ OS-9 Version.

CCF, Obj. Only $100.00 - CO, Obj. Only $ 59.95
F, " " $100.00 - O, object only $150.00
U, " " $300.00

PROGRAMMING LANGUAGES

PL/9 from Windrush Micro Systems -- By Graham Trott. A combination Editor Compiler Debugger. Direct source-to-object compilation delivering fast, compact, re-entrant, ROM-able, PIC. 8 & 16-bit Integers & 6-digit Real numbers for all real-world problems. Direct control over ALL System resources, including interrupts. Comprehensive library support; simple Machine Code interface; step-by-step tracer for instant debugging. 500+ page Manual with tutorial guide.

F, CCF - $198.00

PASC from S.E. Media - A Flex9 Compiler with a definite Pascal "flavor". Anyone with a bit of Pascal experience should be able to begin using PASC to good effect in short order. The PASC package comes complete with three sample programs: ED (a syntax or structure editor), EDITOR (a simple, public domain, screen editor) and CHESS (a simple chess program). The PASC package come complete with source (written in PASC) and documentation.

FLEX $95.00

WHIMSICAL from S.E. MEDIA Now supports *Real Numbers*. "Structured Programming" WITHOUT losing the Speed and Control of Assembly Language! Single-pass Compiler features unified, user-defined I/O; produces ROMable Code; Procedures and Modules (including pre-compiled Modules); many "Types" up to 32 bit Integers, 6-digit Real Numbers, unlimited sized Arrays (vectors only); Interrupt handling; long Variable Names; Variable Initialization; Include directive; Conditional compiling; direct Code insertion; control of the Stack Pointer; etc. Run-Time subroutines inserted as called during compilation. *Normally produces 10% less code than PL/9.*

F and CCF - $195.00

KANSAS CITY BASIC from S.E. Media - *Basic for Color Computer OS-9* with many new commands and sub-functions added. A full implementation of the IF-THEN-ELSE logic is included, allowing nesting to 255 levels. Strings are supported and a subset of the usual string functions such as LEFT$, RIGHT$, MID$, STRING$, etc. are included. Variables are dynamically allocated. Also included are additional features such as Peek and Poke. A must for any Color Computer user running OS-9.

CoCo OS-9 $39.95

C Compiler from Windrush Micro Systems by James McCosh. Full C for FLEX except bit-fields, including an Assembler. *Requires the TSC Relocating Assembler if user desires to implement his own Libraries.*

F and CCF - $295.00

C Compiler from Introl -- Full C except Doubles and Bit Fields, streamlined for the 6809. Reliable Compiler; FAST, efficient Code. More UNIX Compatible than most.

FLEX, CCF, OS-9 (Level II ONLY), U - $575.00

PASCAL Compiler from Lucidata -- ISO Based P-Code Compiler. Designed especially for Microcomputer Systems. Allows linkage to Assembler Code for maximum flexibility.

F and CCF 5" - $99.95 F 8" - $99.95

PASCAL Compiler from OmegaSoft (now Certified Software) -- For the *PROFESSIONAL*; ISO Based, Native Code Compiler. Primarily for Real-Time and Process Control applications. Powerful; Flexible. Requires a "Motorola Compatible" Relo. Asmb. and Linking Loader.

F and CCF - $425.00 - One Year Maint. $100.00
OS-9 68000 Version - $900.00

KBASIC - from S.E. MEDIA -- A "Native Code" BASIC Compiler which is now Fully TSC XBASIC compatible. The compiler compiles to Assembly Language Source Code. A NEW, streamlined, Assembler is now included allowing the assembly of LARGE Compiled K-BASIC Programs. Conditional assembly reduces Run-time package.

FLEX, CCF, OS-9 Compiler /Assembler $199.00

CRUNCH COBOL from S.E. MEDIA -- Supports large subset of ANSII Level 1 *COBOL* with many of the useful Level 2 features. Full FLEX File Structures, including Random Files and the ability to process Keyed Files. Segment and link large programs at runtime, or implemented as a set of overlays. The System requires 56K and CAN be run with a single Disk System. *A very popular product.*

FLEX, CCF; Normally $199.00
Special Introductory Price $99.95

FORTH from Stearns Electronics -- A CoCo FORTH Programming Language. Tailored to the CoCo! Supplied on Tape, transferable to disk. Written in FAST MI.. Many CoCo functions (Graphics, Sound, etc.). Includes an Editor, Trace, etc. Provides CPU Carry Flag accessibility. Fast Task Multiplexing, Clean Interrupt Handling, etc. for the "Pro". Excellent "Learning" tool!

Color Computer ONLY - $58.95

DATABASE ACCOUNTING

XDMS from Westchester Applied Business Systems

FOR 6809 FLEX-SK-DOS(5/8")

Up to 32 groups/fields per record! Up to 12 character filed name! Up to 1024 byte records! User defined screen and print control! Process files! Form files! Conditional execution! Process chaining! Upward/Downward file linking! File joining! Random file virtual paging! Built in utilities! Built in text line editor! Fully session oriented! Enhanced format! Boldface, Double width, Italics and Underline supported! Written in compact structured assembler! Integrated for FAST execution!

XDMS-IV Data Management System

XDMS-IV is a brand new approach to data management. It not only permits users to describe, enter and retrieve data, but

Telex 5108006630
(615)842-4600
SOUTH EAST MEDIA
5900 Cassandra Smith Rd.
Hixson, TN 37343
for information
call (615) 842-4601
CoCo OS-9™ FLEX™
SOFTWARE

!!! Please Specify Your Operating System & Disk Size !!!

Availability Legends--
F = FLEX, CCF = Color Computer FLEX
O = OS-9, CCO = Color Computer OS-9
U = UniFLEX
CCD = Color Computer Disk
CCT = Color Computer Tape

* OS-9 is a Trademark of Microware and Motorola
* FLEX is a Trademark of Technical Systems Consultants

SOUTH EAST MEDIA
5900 Cassandra Smith Rd
Hixson, TN 37343
info (615) 842-4601
CoCo OS-9™ FLEX™
SOFTWARE

** Shipping **
Add 2% U.S.A.
(min. $2.50)
Add 5% Surface Foreign
10% Air Foreign

VISA

also to process entire files producing customized reports, screen displays and file output. Processing can consist of any of a set of standard high level functions including record and field selection, sorting and aggregation, lookups in other files, special processing of record subsets, custom report formatting, totaling and subtotaling, and presentation of up to three related files as a "database" on user defined output reports.

[POWERFUL COMMANDS]

XDMS-IV combines the functionality of many popular DBMS software systems with a new easy to use command set into a single integrated package. We've included many new features and commands including a set of general file utilities. The processing commands are Input-Process-Output (IPO) oriented which allows almost instant implementation of a process design.

[SESSION ORIENTED]

XDMS-IV is session oriented. Enter "XDMS" and you are in instant command of all the features. No more waiting for a command to load in from disk! Many commands are immediate, such as CREATE (file definition), UPDATE (file editor), PURGE and DELETE (utilities). Others are process commands which are used to create a user process which is executed with a RUN command. Either may be entered into a "process" file which is executed by an EXECUTE statement. Processes may execute other processes, or themselves, either conditionally or unconditionally. Menus and screen prompts are easily coded, and entire user applications can be run without ever leaving XDMS-IV!

[IT'S EASY TO USE!]

XDMS-IV keeps data management simple! Rather than design a complex DBMS which hides the true nature of the data, we kept XDMS-IV file oriented. The user view of data relationships is presented in reports and screen output, while the actual data resides in easy to maintain files. This aspect permits customized presentation and reports without complex redefinition of the database files and structure. XDMS-IV may be used for a wide range of applications from simple record management systems (addresses, inventory ...) to integrated database systems (order entry, accounting...). The possibilities are unlimited...

FOR 6809 FLEX-SK-DOS(5/8") $249.95

ASSEMBLERS

ASTRUK09 from S.B. Media -- A "Structured Assembler for the 6809" which requires the TSC Macro Assembler.
F, CCF - $99.95

Macro Assembler for TSC -- The FLEX STANDARD Assembler.
Special -- CCF $35.00; F $50.00

OSM Extended 6809 Macro Assembler from Lloyd I/O. -- Provides local labels, Motorola S-records, and Intel Hex records; XREF. Generate OS-9 Memory modules under FLEX.
FLEX, CCF, OS-9 $99.00

Relocating Assembler/Linking Loader from TSC. -- Use with many of the C and Pascal Compilers.
F, CCF $150.00

MACE, by Graham Trott from Windrush Micro Systems -- Co-Resident Editor and Assembler; fast interactive A.L. Programming for small to medium-sized Programs.
F, CCF - $75.00
XMACB -- MACE w/Cross Assembler for 6800/1/2/3/8 *F, CCF - $98.00*

CROSS ASSEMBLERS

TRUE CROSS ASSEMBLERS from Computer Systems Consultants -- Supports 1802/5, Z-80, 6800/1/2/3/8/11/HC11, 6804, 6805/HC05/ 146805, 6809/00/01, 6502 family, 8080/5, 8020/1/2/35/C35/39/40/48/C48/49/C49/50/8748/49, 8031/51/8751, and 68000 Systems. Assembler and Listing formats same as target CPU's format. Produces machine independent Motorola S-Text.

68000 or 6809, FLEX, CCF, OS-9, UniFLEX
any object or source each - $50.00
any 3 object or source each - $100.00
Set of ALL object $200.00 - source $300.00

XASM Cross Assemblers for FLEX from S.E. MEDIA -- This set of 6800/1/2/3/5/8, 6301, 6502, 8080/5, and Z80 Cross Assemblers uses the familiar TSC Macro Assembler Command Line and Source Code format. Assembler options, etc., in providing code for the target CPU's.
Complete set, FLEX only - $150.00

CRASMB from LLOYD I/O -- Supports Motorola's, Intel's, Zilog's, and other's CPU syntax for these 8-Bit microprocessors: 6800, 6801, 6303, 6804, 6805, 6809, 6811 (all varieties); 6502, 1802/5, 8048 family, 8051 family, 8080/85, Z8, Z80, and TMS-7000 family. Has MACROS, Local Labels, Label X-REF, Label Length to 30 Chars. Object code formats: Motorola S-Records (text), Intel HEX-Records (text), OS9 (binary), and FLEX (binary). Written in Assembler -- e.g. Very Fast.

	CPU TYPE - Price each:			
For:	MOTOROLA	INTEL	OTHER	COMPLETE SET
FLEX9	$150	$150	$150	$399
OS9/6809	$150	$150	$150	$399
OS9/68K	------	------	------	$432

CRASMB 1632 from LLOYD I/O -- Supports Motorola's 68000, and has same features as the 8 bit version. OS9/68K Object code Format allows this cross assembler to be used in developing your programs for OS9/68K on your OS9/6809 computer.
FLEX, CCF, OS-9/6809 $249.00

UTILITIES

Basic09 XRef from S.E. Media -- This Basic09 Cross Reference Utility is a Basic09 Program which will produce a "pretty printed" listing with each line numbered, followed by a complete cross referenced listing of all variables, external procedures, and line numbers called. Also includes a Program List Utility which outputs a fast "pretty printed" listing with line numbers. Requires Basic09 or RunB.
O & CCO obj. only -- $39.95; w/ Source - $79.95

BTree Routines - Complete set of routines to allow simple implementation of keyed files - *for your programs* - running under Basic09. A real time saver and should be a part of every serious programmers tool-box.
O & CCO obj. only - $89.95

Lucidata PASCAL UTILITIES (Requires LUCIDATA Pascal ver 3)

XREF -- produce a Cross Reference Listing of any text; oriented to Pascal Source.

INCLUDE -- Include other Files in a Source Text, including Binary - unlimited nesting.

PROFILER -- provides an Indented, Numbered, "Structogram" of a Pascal Source Text File; view the overall structure of large programs, program integrity, etc. Supplied in Pascal Source Code; requires compilation.
F, CCF --- EACH 5" - $40.00, 8" - $50.00

DUB from S.E. Media -- **A UniFLEX BASIC decompiler** Re-Create a Source Listing from UniFLEX Compiled basic Programs. Works w/ ALL Versions of 6809 UniFLEX basic.
U - $219.95

LOW COST PROGRAM KITS from Southeast
Media -- The following kits are available for FLEX on either 5 or 8 inch disk.

1. **BASIC TOOL-CHEST $29.95**
   BLISTER.CMD: pretty printer
   LINEXREF.BAS: line cross-reference
   REMPAC.BAS, SPCPAC.BAS, COMPAC.BAS: remove superfluous code
   STRIP.BAS: superfluous line-numbers stripper
2. **FLEX UTILITIES KIT $39.95**
   CATS.CMD: alphabetically-sorted directory listing
   CATD.CMD: date-sorted directory listing
   COPYSORT.CMD: file copy, alphabetically
   COPYDATE.CMD: file copy, by date-order
   FILEDATE.CMD: change file creation date
   INFO.CMD (& INFOGMX.CMD): tells disk attributes & contents
   RELINK.CMD (& RELINK82): re-orders fragmented free chain
   RESQ.CMD: undeletes (recovers) a deleted file
   SECTORS.CMD: show sector order in free chain
   XL.CMD: super text lister
3. **ASSEMBLERS/DISASSEMBLERS UTILITIES $39.95**
   LINEFEED.CMD: 'modularise' disassembler output
   MATH.CMD: decimal, hex, binary, octal conversions & tables
   SKIP.CMD: column stripper
4. **WORD - PROCESSOR SUPPORT UTILITIES $49.95**
   FULLSTOP.CMD: checks for capitalization where required
   BSTYCIT.BAS (.BAC): Stylo to dot-matrix printer program
   NECPRINT.CMD: Stylo to dot-matrix printer filter code
5. **UTILITIES FOR INDEXING $49.95**
   MENU.BAS: selects required program from list below
   INDEX.BAC: word index
   PHRASES.BAC: phrase index
   CONTENT.BAC: table of contents
   INDXSORT.BAC: fast alphabetic sort routine
   FORMATER.BAC: produces a 2-column formatted index
   APPEND.BAC: append any number of files
   CHAR.BIN: line reader

FULL SCREEN FORMS DISPLAY from Computer Systems Consultants -- TSC Extended BASIC program supports any Serial Terminal with Cursor Control or Memory-Mapped Video Displays; substantially extends the capabilities of the Program Designer by providing a table-driven method of describing and using Full Screen Displays.
F and CCF, U - $25.00, w/ Source - $50.00

SOLVE from S.E. Media - OS-9 Levels I and II only. A Symbolic Object/Logic Verification & Examine debugger. Including inline debugging, disassemble and assemble. SOLVE IS THE MOST COMPLETE DEBUGGER we have seen for the 6809 OS-9 series! SOLVE does it all! With a rich selection of monitor, assembler, disassembler, environmental, execution and other miscellaneous commands, SOLVE is the MOST POWERFUL tool-kit item you can own! Yet, SOLVE is simple to use! With complete documentation, a snap! Everyone who has ordered this package has raved! See review - 68 Micro Journal - December 1985. No 'blind' debugging here, full screen displays, rich and complete in information presented. Since review in 68 Micro Journal, this is our fastest mover!
Levels I & II only - OS-9 Regular $149.95
SPECIAL INTRODUCTION OFFER *$69.95*

Telex 5108006630
(615)842-4600
SOUTH EAST MEDIA
5900 Cassandra Smith Rd.
Hixson, TN 37343
for information
call (615) 842-4601
CoCo OS-9™ FLEX™
SOFTWARE

DISK UTILITIES

OS-9 VDisk from S.E. Media -- For Level I only. Use the Extended Memory capability of your SWTPC or Gimix CPU card (or similar format DAT) for FAST Program Compiles, CMD execution, high speed inter-process communications (without pipe buffers), etc. - SAVE that System Memory. Virtual Disk size is variable in 4K increments up to 960K. Some Assembly Required.
Level I OS-9 obj. $79.95; w/ Source $149.95

O-F from S.E. Media -- Written in BASIC09 (with Source), includes: REFORMAT, a BASIC09 Program that reformats a chosen amount of an OS-9 disk to FLEX Format so it can be used normally by FLEX; and FLEX, a BASIC09 Program that does the actual read or write function to the special O-F Transfer Disk; user-friendly menu driven. Read the FLEX Directory, Delete FLEX Files, Copy both directions, etc. FLEX users use the special disk just like any other FLEX disk
O - 6809/68000 $79.95

LSORT from S.E. Media - A SORT/MERGE package for OS-9 (Level I & II only). Sorts records with fixed lengths or variable lengths. Allows for either ascending or descending sort. Sorting can be done in either ASCII sequence or alternate collating sequence. Right, left or no justification of data fields available. LSORT includes a full set of comments and errors messages.
OS-9 $85.00

HIER from S.E. Media - *HIER is a modern hierarchal storage system for users under FLEX.* It answers the needs of those who have hard disk capabilities on their systems, or many files on one disk - any size. Using **HIER a regular (any) FLEX disk (8 - 5 - hard disk) can have sub directories.** By this method the problems of assigning unique names to files is less burdensome. Different files with the exact same name may be on the same disk, as long as they are in different directories. For the winchester user this becomes a must. Sub-directories are the modern day solution that all current large systems use. **Each directory looks to FLEX like a regular file, except they have the extension '.DIR'.** A full set of directory handling programs are included, making the operation of HIER simple and straightforward. A special install package is included to install HIER to your particular version of FLEX. Some assembly required. Install indicates each byte or reference change needed. Typically - 6 byte changes in source (furnished) and one assembly of HIER is all that is required. No programming required!
*FLEX - SK*DOS $79.95*

!!! Please Specify Your Operating System & Disk Size !!!

Availability Legends--
F = FLEX, CCF = Color Computer FLEX
O = OS-9, CCO = Color Computer OS-9
U = UniFLEX
CCD = Color Computer Disk
CCT = Color Computer Tape

*OS-9 is a Trademark of Microware and Motorola
*FLEX is a Trademark of Technical Systems Consultants

SOUTH EAST MEDIA
5900 Cassandra Smith Rd. Hixson, TN 37343
info (615) 842-4601
CoCo OS-9 FLEX SOFTWARE

** Shipping **
Add 2% U.S.A.
(min. $2.50)
Add 5% Surface Foreign
10% Air Foreign

COPYMULT from S.E. Media -- Copy LARGE Disks to several smaller disks. FLEX utilities allow the backup of ANY size disk to any SMALLER size diskettes (Hard Disk to floppies, 8" to 5", etc.) by simply inserting diskettes as requested by COPYMULT. No fooling with directory deletions, etc. COPYMULT.CMD understands normal "copy" syntax and keeps up with files copied by maintaining directories for both host and receiving disk system. Also includes BACKUP.CMD to download any size "random" type file; RESTORE.CMD to restructure copied "random" files for copying, or recopying back to the host system; and FREELINK.CMD as a "bonus" utility that "relinks" the free chain of floppy or hard disk, eliminating fragmentation.

Completely documented Assembly Language Source files included. ALL 4 Programs (FLEX, 8" or 5") $99.50

COPYCAT from Lucidata -- *Pascal NOT required.* Allows reading TSC Mini-FLEX, SSB DOS68, and Digital Research CP/M Disks while operating under FLEX 1.0, FLEX 2.0, or FLEX 9.0 with 6800 or 6809 Systems. COPYCAT will not perform miracles, but, between the program and the manual, you stand a good chance of accomplishing a transfer. Also includes some Utilities to help out. Programs supplied in Modular Source Code (Assembly Language) to help solve unusual problems.

F and CCF 5" - $50.00 F 8" - $65.00

VIRTUAL TERMINAL from S.E. Media - Allows one terminal to do the work of several. The user may start as many as eight task on one terminal, under *VIRTUAL TERMINAL* and switch back and forth between task at will. No need to exit each one; just jump back and forth. Complete with configuration program. The best way to keep up with those background programs.

O & CCO - obj. only - $49.95

FLEX DISK UTILITIES from Computer Systems Consultants -- Eight (8) different Assembly Language (w/ Source Code) FLEX Utilities for every FLEX User's Toolbox: Copy a File with CRC Errors; Test Disk for errors; Compare two Disks; a fast Disk Backup Program; Edit Disk Sectors; Linearize Free-Chain on the Disk; print Disk Identification; and Sort and Replace the Disk Directory (in sorted order). -- PLUS -- Ten XBASIC Programs including: A BASIC Resequencer with EXTRAs over "RENUM" like check for missing label definitions, processes Disk to Disk instead of in Memory, etc. Other programs Compare, Merge, or Generate Updates between two BASIC Programs, check BASIC Sequence Numbers, compare two unsequenced files, and 5 Programs for establishing a Master Directory of several Disks, and sorting, selecting, updating, and printing paginated listings of these files. A BASIC Cross-Reference Program, written in Assembly Language, which provides an X-Ref Listing of the Variables and Reserved Words in TSC BASIC, XBASIC, and PRECOMPILER BASIC Programs.

ALL Utilities include Source2 (either BASIC or A.L. Source Code).

F and CCF - $50.00
BASIC Utilities ONLY for UniFLEX -- $30.00

GAMES

RAPIER - 6809 Chess Program from S.E. Media -- Requires FLEX and Displays on Any Type Terminal. Features: Four levels of play. Swap side. Point scoring system. Two display boards. Change skill level. Solve Checkmate problems in 1-2-3-4 moves. Make move and swap sides. Play white or black. **This is one of the strongest CHESS programs running on any microcomputer,** *estimated USCF Rating 1600+ (better than most 'club' players at higher levels)*

F and CCF - $79.95

COMMUNICATIONS

CMODEM Telecommunications Program from Computer Systems Consultants, Inc. -- Menu-Driven; supports Dumb-Terminal Mode, Upload and Download in non-protocol mode, and the CP/M "Modem7" Christensen protocol mode to enable communication capabilities for almost any requirement. Written in "C".

FLEX, CCF, OS-9, UniFLEX, 68000 & 6809 with Source $100.00 - without Source $50.00

X-TALK from S.E. Media - X-TALK consists of two disks and a special cable, the hookup enables a 6809 SWTPC computer to dump UniFLEX files directly to the UniFLEX MUSTANG-020. This is the ONLY currently available method to transfer SWTPC 6809 UniFLEX files to a 68000 UniFLEX system. Gimix 6809 users may dump a 6809 UniFLEX file to a 6809 UniFLEX five inch disk and it is readable by the MUSTANG-020. The cable is specially prepared with internal connections to match the non-standard SWTPC SO/9 I/O Db25 connectors. A special SWTPC S+ cable set is also available. Users should specify which SWTPC system he/she wishes to communicate with the MUSTANG-020. The X-TALK software is furnished on two disks. One eight inch disk contains S.E. Media modem program C-MODEM (6809) and the other disk is a MUSTANG-020 five inch disk with C-MODEM (68020). Text and binary files may be directly transferred between the two systems. The C-MODEM programs are unaltered and perform as excellent modem programs also. X-TALK can be purchased with or without the special cables, but this special price is available to registered MUSTANG-020 users only.

X-TALK Complete (cable, 2 disks) $99.95
X-TALK Software (2 disks only) $69.95
X-TALK with CMODEM Source $149.95

XDATA from S.B. Media - A COMMUNICATION Package for the UniFLEX Operating System. Use with CP/M, Main Frames, other UniFLEX Systems, etc. Verifies Transmission using checksum or CRC; Re-Transmits bad blocks, etc.

U - $299.99

EDITORS & WORD PROCESSING

JUST from S.E. Media -- Text Formatter developed by Ron Anderson; for Dot Matrix Printers, provides many unique features. Output "Formatted" Text to the Display. Use the FPRINT.CMD supplied for producing multiple copies of the "Formatted" Text on the Printer INCLUDING EMBEDDED PRINTER COMMANDS (very useful at other times also, and worth the price of the program by itself). "User Configurable" for adapting to other Printers (comes set up for Epson MX-80 with Graftrax); up to ten (10) imbedded "Printer Control Commands". Compensates for a "Double Width" printed line. Includes the normal line width, margin, indent, paragraph, space, vertical skip lines, page length, page numbering, centering, fill, justification, etc. Use with PAT or any other editor.

* Now supplied as a two disk set:

Disk #1: JUST2.CMD object file, JUST2.TXT PL9 source:FLEX - CC

Disk #2: JUSTSC object and source in C: FLEX - OS9 - CC

The JTSC and regular JUST C source are two separate programs. JTSC compiles to a version that expects TSC Word Processor type commands, (.pp .sp .ce etc.) Great for your older text files. The C source compiles to a standard syntax JUST.CMD object file. Using JUST syntax (.p .u .y etc.) With all JUST functions plus several additional printer formatting functions. Reference the JUSTSC C source. For those wanting an excellent BUDGET PRICED word processor, with features none of the others have. This is it!

Disk (1) - PL9 FLEX only - F & CCF - $49.95
Disk Set (2) - F & CCF & OS9 (C version) - $69.95
OS-9 68K000 complete with Source - $79.95

PAT from S.B. Media - A full feature screen oriented TEXT EDITOR with all the best of "PIE". For those who swore by and loved only PIB, this is for you! All PIE features and much more! Too many features to list. And if you don't like these, change or add your own. PL-9 source furnished. "C" source available soon. Easily configured to your CRT, with special config section.

Regular FLEX $129.50
** SPECIAL INTRODUCTION OFFER * $79.95*
SPECIAL PAT/JUST COMBO (w/source)
FLEX $99.95
OS-9 68K Version $229.00
SPECIAL PAT/JUST COMBO 68K $249.00
Note: *JUST in "C" source available for OS-9*

CEDRIC from S.E. Media - A screen oriented TEXT EDITOR with availability of 'MENU' aid. Macro definitions, configurable 'permanent definable MACROS' - all standard features and the fastest 'global' functions in the west. A simple, automatic terminal config program makes this a real 'no hassel' product. Only 6K in size, leaving the average system over 165 sectors for text buffer - appx. 14,000 plus of free memory! Extra fine for programming as well as text.

Regular $129.95
SPECIAL INTRODUCTION OFFER FLEX $69.95

BAS-EDIT from S.B. Media - A TSC BASIC or XBASIC screen editor. Appended to BASIC or XBASIC, BAS-EDIT is transparent to normal BASIC/XBASIC operation. Allows editing while in BASIC/XBASIC. Supports the following functions: OVERLAY, INSERT and DUP LINE. Make editing BASIC/XBASIC programs SIMPLE! A GREAT time and effort saver. Programmers love it! NO more retyping entires lines, etc. Complete with over 25 different CRT terminal configuration overlays.

FLEX, CCF, STAR-DOS Regular $69.95
Limited Special Offer: $39.95

SCREDITOR III from Windrush Micro Systems -- Powerful Screen-Oriented Editor/Word Processor. Almost 50 different commands; over 300 pages of Documentation with Tutorial. Features Multi-Column display and editing, "decimal align" columns (AND add them up automatically), multiple keystroke macros, even/odd page headers and footers, imbedded printer control codes, all justifications, "help" support, store common command series on disk, etc. Use supplied "set-ups", or remap the keyboard to your needs. Except for proportional printing, this package will DO IT ALL!

6800 or 6809 FLEX or SSB DOS, OS-9 - $175.00

SPELLB "Computer Dictionary" from S.E. Media -- **OVER 150,000 words!** *Look up a word from within your Editor or Word Processor (with the SPH.CMD Utility which operates in the FLEX UCS).* Or check and update the Text after entry; ADD WORDS to the Dictionary, "Flag" questionable words in the Text, "View a word in context" before changing or ignoring, etc. SPELLB first checks a "Common Word Dictionary", then the normal Dictionary, then a "Personal Word List", and finally, any "Special Word List" you may have specified. SPELLB also allows the use of Small Disk Storage systems.

F and CCF - $129.95

STYLO-GRAPH from Great Plains Computer Co. -- A full-screen oriented WORD PROCBSSOR -- (uses the 51 x 24 Display Screens on CoCo FLEX/STAR-DOS, or PBJ Wordpak). Full screen display and editing; supports the Daisy Wheel proportional printers.

NEW PRICES 6809 CCF and CCO - $99.95,
F or O - $179.95, U - $299.95

STYLO-SPELL from Great Plains Computer Co. -- Fast Computer Dictionary. Complements Stylograph.

NEW PRICES 6809 CCF and CCO - $69.95,
F or O - $99.95, U - $149.95

(615)842-4600 Telex 5106006630
SOUTH EAST MEDIA
5900 Cassandra Smith Rd.
Hixson, TN 37343
for information
CALL (615) 842-4601
CoCo OS-9™ FLEX™
SOFTWARE

STYLO-MERGE from Great Plains Computer Co. -- Merge Mailing List to "Form" Letters, Print multiple Files, etc., through Stylo.

NEW PRICES6809 CCF and CCO - $59.95,
F or O - $79.95, U - $129.95

STYLO-PAK -- Graph + Spell + Merge Package Deal!!!

F or O - $329.95, U - $549.95
O, 68000 $595.00

MISCELLANEOUS

TABULA RASA SPREADSHEET from Computer Systems Consultants -- TABULA RASA is similar to DESKTOP/PLAN; provides use of tabular computation schemes used for analysis of business, sales, and economic conditions. Menu-driven; extensive report-generation capabilities. Requires TSC's Extended BASIC.

F and CCF, U - $50.00, w/ Source - $100.00

DYNACALC -- Electronic Spread Sheet for the 6809 and 68000.

F, OS-9 and SPECIAL CCF - $200.00, U - $395.00
OS-9 68K - $595.00

FULL SCREEN INVENTORY/MRP from Computer Systems Consultants -- Use the Full Screen Inventory System/Materials Requirement Planning for maintaining inventories. Keeps item field file in alphabetical order for easier inquiry. Locate and/or print records matching partial or complete item, description, vendor, or attributes; find backorder or below stock levels. Print-outs in item or vendor order. MRP capability for the maintenance and analysis of Hierarchical assemblies of items in the inventory file. Requires TSC's Extended BASIC.

F and CCF, U - $50.00, w/ Source - $100.00

FULL SCREEN MAILING LIST from Computer Systems Consultants -- The Full Screen Mailing List System provides a means of maintaining simple mailing lists. Locate all records matching on partial or complete name, city, state, zip, or attributes for Listings or Labels, etc. Requires TSC's Extended BASIC.

F and CCF, U - $50.00, w/ Source - $100.00

DIET-TRAC Forecaster from S.E. Media -- An XBASIC program that plans a diet in terms of either calories and percentage of carbohydrates, proteins and fats (C P G%) or grams of Carbohydrate, Protein and Fat food exchanges of each of the six basic food groups (vegetable, bread, meat, skim milk, fruit and fat) for a specific individual. Sex, Age, Height, Present Weight, Frame Size, Activity Level and Basal Metabolic Rate for normal individual are taken into account. Ideal weight and sustaining calories for any weight of the above individual are calculated. Provides number of days and daily calendar after weight goal and calorie plan is determined.

F - $59.95, U - $89.95

!!! Please Specify Your Operating System & Disk Size !!!

Availability Legends-

F = FLEX, CCF = Color Computer FLEX
O = OS 9, CCO = Color Computer OS-9
U = UniFLEX
CCD = Color Computer Disk
CCT = Color Computer Tape

* OS-9 is a Trademark of Microware and Motorola
* FLEX is a Trademark of Technical Systems Consultants

SOUTH EAST MEDIA
5900 Cassandra Smith Rd
Hixson, TN 37343
info (615) 842-4601
CoCo OS-9 FLEX
SOFTWARE

** Shipping **

Add 2% U.S.A.
(min. $2.50)
Add 5% Surface Foreign
10% Air Foreign

There are places where you can buy pre-stripped and formed wires ideal for this type of thing. Because you have only used 25 of the contacts on each side you will have a small area of contacts on the one end which you can use for those unusual interface problems. The only other thing I should mention is that some day you will run into the problem where your break-out box has a male and a female connector and your two devices will be both male or female. To get around this problem what you do is build yourself a couple of "gender changers". Take one male solder type DB-25 connector and one male wire-wrap DB-25 connector, place the wire-wrap pins in the solder type connector back to back and solder the 25 connectors all around. This may be used to change from female to male. You should also get a female solder type connector and a female wire-wrap and do the same thing. This will give you the capability to interface to any type of connector easily and the gender changers can be used by themselves. After you have got the devices talking to each other properly write down the connections and make yourself a cable following your diagram.

Amazing! A cable that works the first time. If you do a lot of interfacing between devices I'm sure you will find this to be very useful, I don't know how I got along without it.

May '68' MICRO JOURNAL publish for eternity!

Editor's Note: Well Barry, eternity is sorta a'long time. but I won't complain. For now, however, I do want to thank you, and all the others who care enough to share. From thousands of other users - THANKS! And please, all of you, keep that good stuff coming!

DMW

Bit-Bucket

By: All of us

"Contribute Nothing - Expect Nothing", DMW '86

Dear Mr. Williams:

Please find enclosed a file named **INDEX86.TXT**, which is my most recent installment to my 68 Micro Journal key word index. It may be appended to my previous seven year index, for a full eight year index to 68 Micro Journal. For the benefit of readers who may not be familiar with this index, I have taken the liberty of repeating the instructions:

This index is a standard Flex text file which has proved very useful to me. One of the main values of 68MJ is the useful little tidbits that are often included in letters, asides, and the Bit Bucket. It's very difficult to track these back down some months later. I've tried to develop a comprehensive index including all bits of info I felt might be useful later. It's invaluable for finding those patches to contributed software that appear some months later.

The index is a key word index. Each line starts with month, year and page number of an article or item, and usually includes the author's name. I've also tried to identify the item as article, letter, program, utility, etc. Then follows certain key words selected to characterize the topic(s) covered. I've attempted to stay within the 128 character limitation of the Flex line buffer.

You may then use Leo Taylor's FIND.CMD to locate a specific topic of interest. When FIND locates a match, it prints the entire line including the date and page. This makes the whole operation quite simple, and I didn't even need to write any software! FIND.CMD syntax:

+++FIND,<STRING>,<FILENAME

prints all lines containing "string". Default extension is .TXT. The only confusion seems to be with names that are hyphenated or slashed. To simplify things, I have generally just deleted these extra characters. Examples:

SS30 NOT SS-30, CPM NOT CP/M, OS9 NOT OS-9, MPS2 NOT MP-S2, CFM3 NOT CFM/3, IO NOT I/O, PL9 NOT PL/9

Thank you for the opportunity to once again show my appreciation for your efforts and those of the contributors to 68 Micro Journal.

Sincerely,
John Current
2675 Pebble Dawn
San Antonio, TX 78232

Editor's Note: John, I want to thank you for the supplement to your excellent series of Indexes of 68 Micro.

Not only are they what our readers, especially new ones, need but we use the heck out of them also. We get calls every day from readers needing to order a back issue with some particular article. Your Index is what we use. It is great folks, like yourself that contribute, which keeps us afloat. I really appreciate all you who keep things going.

DMW

Index 68 Micro Journal - 1986

JAN 86 P7 ANDERSON FLEX USER NOTES OS9 ON SWTPC TANDY 1200HD 2764 PAT FLEX IN ROM

JAN 86 P8 PASS C USER NOTES RANDOM FILES COPY FILE SUBSTITUTE VARIABLE LIST FILES PARALLEL COLUMNS C LISTINGS

JAN 86 P13 DIBBLE OS9 USER NOTES ASSEMBLY LANGUAGE STRING FUNCTIONS C REFERENCE MANUAL ASSEMBLY LISTINGS

JAN 86 P18 VOIGTS BASIC OS9 STRINGS ASCII CHECKING ACCOUNT BALANCE PASCAL LISTING

JAN 86 P19 ELBERT ADA 68000 ADVANCED TASKING

JAN 86 P21 LEWIS REVIEW SCULPTOR ISAM BTREE DATA BASE MANAGEMENT SYSTEM OBMS

JAN 86 P25 HERNANDEZ REVIEW EMERALD E381 68008 SBC CPMBBK

JAN 86 P27 LARIMORE AND BIRDSEYE ARTICLE ADD HARD DISK TO FLEX OR OS9 SBC PT69 S350 HD XEBEC

JAN 86 P35 FRANCK ARTICLE RADIX SORT FOR 68000 ASSEMBLY LANGUAGE

JAN 86 P44 JOHNSHOY ARTICLE RATBAS RATIONALIZED BASIC PREPROCESSOR TSC XBASIC XPC

JAN 86 P46 BROMLEY SHRINK BASIC PROGRAMS COCO

JAN 86 P48 STARK LETTER LISTING STARDOS COMMANDS

JAN 86 P48 LYNDE LETTER OS9 _STUCHECK ROUTINE

JAN 86 P49 JONES LETTER TSC XBASIC TOKENS HASH CODE ERROR MESSAGES STOP AND

FEB 86 P8 ANDERSON FLEX USER NOTES PAT EDITOR LOCAL VS GLOBAL VARIABLES DISK DRIVE ALIGNMENT TIME AND DATE

FEB 86 P9 PASS C USER NOTES DATE PACKAGE SET FUNCTIONS PERMUTATION GENERATOR HELP BRAIN TEASER C LISTING

FEB 86 P14 VOIGTS BASIC OS9 REAL TIME CLOCK RTC TIME DATE ASSEMBLY LISTING

FEB 86 P17 DIBBLE OS9 USER NOTES MICROWARE SEMINAR

FEB 86 P19 LUCIDO 68000 USER NOTES OSRAM DEBUGGER

FEB 86 P21 FERGUSON ARTICLE INTERFACE WINCHESTER HARD DISK TO S350 BUBS S980 WELLWRITTEN

FEB 86 P26 WILLIAMS MUSTANG020 UPDATE #1 68020 DATACOMP BENCHMARKS

FEB 86 P27 BURG ARTICLE FLEX KERMIT COMMUNICATION PROTOCOL IN CONTROL C

FEB 86 P29 BUSSCHE ARTICLE MOD SWTP DMAF2 TO SUPPORT 5" SCHEMATIC 6809 ASSEMBLY LISTING

FEB 86 P38 MICHAL ARTICLE 6809 IN THE LAB A TO D D TO A SCHEMATIC 6800 ASSEMBLY LISTING

FEB 86 P43 BALITSKI FLEX PRINTER SPOOLER CONTROL 6809 ASSEMBLY LISTING

FEB 86 P45 DREXLER FLEX UTILITY MEMDISK RESIDENT COMMANDS PAUSE FORM FEED LAST 6800 ASSEMBLY LISTING

FEB 86 P46 FEDERICI LETTER PUBLIC DOMAIN FORTH 83 FOR FLEX

FEB 86 P47 FIG ANNOUNCES FORTH 83 STANDARD

FEB 86 P47 MICROWARE ANNOUNCES OS9 NETWORK FILE MANAGER

FEB 86 P48 BELGARD ARTICLE USING FORTH

FEB 86 P49 LARIMORE STARDOS UTILITY HELP SYSTEM 6809 ASSEMBLY LISTING

FEB 86 P50 JONES LETTER TSC XBASIC LISTING .BAC FILES CHAINING

MAR 86 P7 ANDERSON FLEX USER NOTES PL9 COMPILER OPTIMIZATION WITH ASSEMBLER CODE 6809 ASSEMBLY LISTING

MAR 86 P9 DIBBLE OS9 USER NOTES SUN 68010 WORK STATION WINDOWS

MAR 86 P10 PASS C USER NOTES TEXT EDITOR HELP EXTENSION C LISTINGS

MAR 86 P14 ELBERT ADA AND 68000 COMPILER COMPARISON

MAR 86 P16 LUCIDO 68000 USER NOTES ASSEMBLY LANGUAGE

MAR 86 P19 VOIGTS BASIC OS9 SPEAKING OS9

MAR 86 P21 MUSTANG020 BENCHMARKS C LISTING

MAR 86 P23 LEWIS AND HARDIN ARTICLE SCULPTOR DESCRIPTION OBMS

MAR 86 P26 BURLINSON REVIEW COMPACTA UNIBOARD 6809 SBC 6809 ASSEMBLY LISTING

MAR 86 P35 CURRENT 7 YEAR 68 MICRO JOURNAL KEYWORD INDEX, 1985 INDEX

MAR 86 P40 ADAMS LETTER SIEVE AND FLEX RANDOM FILE TIMING COMPARISON

MAR 86 P40 PRYZYBYL LETTER FIX BUGS IN UDM COCO BUSINESS SOFTWARE BASIC LISTING

MAR 86 P41 OPYRCHAL LETTER BRITISH 6809 DRAGON SHOW COCO COMPUSERVE

MAR 86 P42 WILLIAMS MUSTANG020 SAGA PHOTOS

MAR 86 P43 ROBINSON ARTICLE MVME101 TERMINAL CONFIGURATION VME BUSS 68000 MOTOROLA WYSE 50 XON XOFF SCHEMATIC

MAR 86 P45 BOOTT FLEX UTILITY DCAT IA LIST ALL FILES CAT SORTED BY DATE PL9 LISTING

MAR 86 P47 ANDERSON FLEX UTILITY EXTENSIBLE TABLE DRIVEN LANGUAGE RECOGNITION PL9 LISTING

MAR 86 P48 GOULETTE FLEX UTILITY DELETE ENHANCED 6809 ASSEMBLY LISTING

APR 86 P6 ANDERSON FLEX USER NOTES PAT TEXT EDITOR PL9 ANSI TERMINAL COMPILER FEATURES

APR 86 P10 VOIGTS BASIC OS9 MEMORY MODULE DIRECTORY ALPHABETIZER C LISTING

APR 86 P13 PASS C USER NOTES WINDRUSH MCCOSH COMPILER UPDATES TEXT EDITOR TERMINFO UNIX TSC TO STYLO C LISTING

APR 86 P17 DIBBLE OS9 USER NOTES TOP MENU LIBRARY FUNCTIONS MISMATCHED CHARACTER ALGORITHM C LISTING

APR 86 P29 LOE ARTICLE OPL COMPILER DESCRIPTION

APR 86 P22 MCCARTHEY AND GROSFILEX ARTICLE MOTOROLA 68020 SYSTEM SCHEMATIC

APR 86 P27 CROSSTREAMS REVIEW MICROTIME II CLOCK CALENDAR S530 6809 ASSEMBLY LISTING

APR 86 P34 SIS ARTICLE MOTOROLA MC68HC11 RESET FUNCTIONS SCHEMATIC

APR 86 P41 GROSS ARTICLE FAST FOURIER TRANSFORM FFT PASCAL LISTING

APR 86 P46 LARIMORE LETTER FIX BUG CREXLER'S MEMDISK IN FLEX LC 6809 ASSEMBLY LISTING

APR 86 P47 FISHER LETTER TANDY MC10 USER GROUPS

APR 86 P47 DODGE LETTER C STACK CHECKING

APR 86 P49 COMPILER PRODUCTS UNLIMITED ANNOUNCEMENT OPL COMPILER

APR 86 P50 MICROPROCESSORS DEVELOPMENTS UNLIMITED TRANSFERRING SCULPTOR DATA FILES BETWEEN DIFFERENT SYSTEMS

MAY 86 P6 ANDERSON FLEX USER NOTES COMMON FLEX ROUTINES FILTERS STYTOPAT COUNT BYTES MUSTANG020 PL9 AND 6809 ASSEMBLY LISTINGS

MAY 86 P12 VOIGTS BASIC OS9 MICROWARE C COMPILER C VARIABLES MATRIX.C LISTING

MAY 86 P15 DESCRIPTION OS9 NTEWORK OSNET

MAY 86 P17 PASS C USER NOTES CURSES TERMINFO DEMO POINTERS C LISTING

MAY 86 P23 DIBBLE OS9 USER NOTES INTERACTIVE COMPACT DISKS CDI TINY TRANSLATOR TINY SEARCH 6809 ASSEMBLY LISTING

MAY 86 P25 WILLIAMS MUSTANG020 UPDATE UNFLEX

MAY 86 P28 GALITSKI MEMORY TEST UTILITIES PIC ROTABIT SUNSTEST COMDATA 6809 ASSEMBLY LISTING

MAY 86 P28 HOGLUND ARTICLE ACCESSING FLEX DIRECTORY FROM BASIC TSC XPC LISTING

MAY 86 P35 ANDERSON REVIEW OMEGASOFT PASCAL FOR OS9 MUSTANG020 68020 COMPILER

MAY 86 P37 DESCRIPTION SOUTHEAST MEDIA LOW COST PROGRAM RITS BLISTER REMPAC SPOPAC COMPAC STRIP LINEREF CATS CATD

MAY 86 P39 WELLER REVIEW OPL COMPILER A PRODUCT OF COMPILER PRODUCTS UNLIMITED

MAY 86 P41 STUCKLEN PROGRAM KEYPAD SCAN ROUTINE PIA 6820 6800 ASSEMBLY LISTING

MAY 86 P42 ISAACSON FLEX UTILITY EXECUTE COMMANDS FROM TEXT FILE PL9 LISTING

MAY 86 P43 JOHNS LETTER TSC XBASIC DECIMAL HEX CONVERSION LOGIC FUNCTIONS

MAY 86 P44 MILLS LETTER KERMIT FILE TRANSFER PROTOCOL

MAY 86 P45 DODGE LETTER BUGS FIXES MARCH PII MARCH LIST P22 PL9 ADTEST MICROWARE C

MAY 86 P47 SWTPC 68 DEALER FEST 68010 VME 3600 SCULPTOR CAD

MAY 86 P49 MOTOROLA ANNOUNCEMENTS 68020 8 BIT DISCRETIMINATIONS

MAY 86 P50 MUSTANG020 BENCHMARKS

MAY 86 P50 MICROWARE ANNOUNCEMENTS COM OS9 COMMUNICATIONS PACKAGE MODEM

JUN 86 P7 VOIGTS BASIC OS9 EXECUTABLE MEMORY MODULE TEXT COMPRESSION PACK 6809 ASSEMBLY LISTING

JUN 86 P10 PASS C USER NOTES C COMPILER UPDATES CURSES ONE SCREEN EDITOR RPN CALCULATOR C LISTINGS

JUN 86 P16 REVIEW SARDIS 512K RAM EXPANSION

JUN 86 P17 DIBBLE OS9 USER NOTES STANDARDS TERMINALS

JUN 86 P18 PRODUCTIVITY ARTICLE Gimix BUSS GESPAC FIGURES

JUN 86 P21 AHRENS ARTICLE MOTOROLA MC14500B 1 BIT CPU SCHEMATIC

JUN 86 P25 SYATEK AND PERKINS MOTOROLA RASTER MEMORY SYSTEM RMS VIDEO GRAPHICS MC6847 MC6487 SCHEMATICS

JUN 86 P37 ARTICLE OPL COMPILER PRODUCTS UNLIMITED

JUN 86 P40 MOORSFOOT ARTICLE MODEM600 UPDATE

JUN 86 P42 WELLER REVIEW PASC COMPILER

JUN 86 P43 FRANCOIS FLEX UTILITY RPTERR PRIORITY 6809 ASSEMBLY LISTING

JUN 86 P45 WILLIAMS ARTICLE PT68 OS9

JUN 86 P45 BERGVALL ARTICLE BOOLEAN EPROM PROGRAMMER PLA BOOLROM BASIC09 LISTING SCHEMATIC

JUN 86 P47 TAYLOR LETTER PUBLIC DOMAIN UTILITIES GIVEAWAY UPDATES

JUN 86 P47 ANNOUNCEMENT PERIPHERAL TECHNOLOGY COCO OS9 DRIVERS PT68

JUN 86 P48 WILLS LETTER DISK SECTOR INTERLEAVE BASIC LISTING

JUN 86 P48 ADAMS LETTER WESTCHESTER XDMS DBMS DIFFERENCES BETWEEN LEVELS

JUN 86 P49 ANNOUNCEMENT MICRO CONCEPTS MICROBOX III 68000 MOTOROLA RMS GRAPHICS

JUN 86 P51 FIGURE RELATIVE PERFORMANCE MOTOROLA 68020/68881 COMBINATION

JUN 86 P51 JONES LETTER TSC KBASIC INT(X) FUNCTION

JUL 86 P7 ANDERSON FLEX USER NOTES MUSTANG020 OS9/68K FCB FCC RMB EQU DVORAK KEYBOARD C LISTING

JUL 86 P10 VOIGTS BASIC OS9 DATE TIME SORT C AND 6809 ASSEMBLY LISTINGS

JUL 86 P12 PASS C USER NOTES COMPILER BUGS CUG CURSES FUNCTIONS TEST SIGN EXTENSION C LISTING

JUL 86 P18 GERWITZ ARTICLE XDMS DBMS EXAMPLE EXPENSE BUDGET LISTING

JUL 86 P24 WILLIAMS ARTICLE SPRING COMDEX OS9 ATARI VOICEWRITER KRS COLOR CATCHER

JUL 86 P25 DIBBLE OS9 USER NOTES 68K HACK LINKER TIMING SYSTEM CALLS DYNACALC SPREADSHEET REVIEW

JUL 86 P28 ARTICLE OPL COMPILER TEXT FORMATTER LISTING

JUL 86 P36 WILLIAMS MUSTANG020 UPDATE BURN IN 16 MHZ

JUL 86 P36 ZYCHLINSKI LETTER PL9 AND FFT AM9511 PL9 AND ASSEMBLY LISTING GENPL9.BAS LISTING

JUL 86 P44 MOTOROLA ANNOUNCEMENT MC68HC11A8 BUFFALO MONITOR EVALUATION BOARD

JUL 86 P45 MUSTANG020 ANNOUNCEMENT MOTOROLA CHANNEL MODIFIES

JUL 86 P48 MICROWARE ANNOUNCEMENT CD ROM STANDARD CDI OS9/68K

JUL 86 P47 JONES LETTER TSC XBASIC *R COMMAND REPEAT FLEX COMMAND

JUL 86 P48 SE MEDIA INFO ON BAS.EDIT TSC BASIC

JUL 86 P48 COMPILER PRODUCTS UNLIMITED ANNOUNCEMENT ADVENTURE GAME GENERATOR

JUL 86 P49 VAN GELDER LETTER BUG FIX FOR GAGESS FFT IN APRIL

JUL 86 P50 PETERSON LETTER BUG FIX FLEX PRINTER SPOOLER INTERRUPTS STROBE 6809 ASSEMBLY LISTING

JUL 86 P50 HOLZHAKER LETTER BUG FIX COMPACTA UniBOARD 6809 SBC

AUG 86 P7 ANDERSON USER NOTES FFT BUG UHRICH ALGORITHM PASCAL AND PL9 LISTING

AUG 86 P13 VOIGTS BASIC OS9 MAKING SYSTEM DISK

AUG 86 P16 PASS C USER NOTES C COMPILERS FOR 68000 UNIX EX OS9/68K CURSES TEXT EDITOR B TREE MODIFY C LISTING

AUG 86 P21 FERGUSON HIER REVIEW HIERARCHICAL FLEX FILE SYSTEM FOR HARD DISK

AUG 86 P24 LURIE ARTICLE READ FLEX DISK SIR FROM FORTH FF9 FORTH LISTING

AUG 86 P29 WILLIAMS ARTICLE LAS VEGAS NCC GIMIX 68020 SCULPTOR MICROWARE OS9 CDI

AUG 86 P37 TAYLOR ARTICLE INTERACTIVE KARNAUGH MAPPING BOOLEAN LOGIC TRUTH TABLE C LISTING

AUG 86 P44 PIACENZA LETTER RENUMBER.BAS DATTR.CMD BASIC LISTING

AUG 86 P48 JONES LETTER TSC XBASIC DECIMAL TO HEX ASC VAL STR$

AUG 86 P49 FEDERICI LETTER FLEX PRINTER DRIVER IRQ 6809 ASSEMBLY LISTINGS

SEP 86 P7 ANDERSON USER NOTES 64K RAM 68XXX PAT BUGS TANDY OT1

SEP 86 P12 PASS C USER NOTES ANSI STANDARD OUTCHAR LOAN AMORTIZATION C LISTING

SEP 86 P17 VOIGTS BASIC OS9 COCO OS9 LEVEL II ATTACH SYSGEN 6809 ASM AND BASIC09 LISTING

SEP 86 P22 LURIE FORTH COLUMN DESCRIPTION

SEP 86 P25 WILLIAMS REVIEW CHILTON'S GUIDE TO MACINTOSH REPAIR

SEP 86 P26 GILCHRIST FLEX UTILITY UNSQUEEZE XMODEM C LISTING

SEP 86 P30 INFORMATION MANAGEMENT SYSTEM MANUAL IMS DBMS

SEP 86 P40 MOTOROLA DATA MC68020 RELATIVE POWER

SEP 86 P43 MOTOROLA ANNOUNCEMENT MC68881 FPCP FUNCTIONS

SEP 86 P45 JONES LETTER TSC XBASIC INPUT INCH$

SEP 86 P47 SCHAEFER LETTER RELOCATABLE OBJECT CODE

SEP 86 P48 PIACENZA FLEX UTILITY CONT. DATTR.CMD FILE PROTECTION 6809 ASSEMBLY LISTING

OCT 86 P9 ANDERSON USER NOTES OMEGASOFT PASCAL PAT BASIC UTILITY VAR OS9 IO FFT UHRICH BUG

OCT 86 P13 PASS C USER NOTES ANSI STANDARD STRINGS PUNS HUMOR UNSIGNED LONG VARIABLES C LISTING

OCT 86 P18 VOIGTS BASIC OS9 XBASIC REVIEW NEWTONS GRAPH POLYNOMIAL XBASIC LISTING

OCT 86 P23 REVIEW READY SET GO MAC DESK TOP PUBLISHING

OCT 86 P25 WOODWARD COCO GAME C LISTING

OCT 86 P28 NORVICOV AND SHARPE ARTICLE MACH1 FORTH MULTITASKING GB64MPU 68000

OCT 86 P37 WOLF ARTICLE MACINTOSH BUILD A MAC SCHEMATIC

OCT 86 P39 STARK ARTICLE SKDOS 6809 68000 DOS PT68 MUSTANG020

OCT 86 P41 SHIRK LETTER TSC UNIFLEX C COMPILERS 80XD AND PASS REPLY

OCT 86 P44 GROVES LETTER RAM UPGRADE COMPUTER EXCELLENCE

OCT 86 P45 PARR FLEX UTILITY REPEAT COMMAND LINE AND MODIFY 6809 ASSEMBLY LISTING

OCT 86 P47 PRESTON LETTER ASSEMBLING COMPACTA UNIBOARD 6809 SBC SYSTEM

OCT 86 P48 WELLER FLEX UTILITY REMINDER REMIND IMPORTANT DATES PL9 LISTING

NOV 86 P8 ANDERSON USER NOTES PT68K FLEX LOGO DUMP PIC 6809 ASSEMBLY LISTING PATOSTY

NOV 86 P16 VOIGTS BASIC OS9 RAMDISK ISAM BTREE TREES PASCAL LISTING

NOV 86 P19 PASS C USER NOTES ANSI STANDARD FWRITE ESCAPE CODES DUMP TO BINARY C LISTING

NOV 86 P25 WILLIAMS REVIEW 68000 DISK REPAIR UTILITY SPECIALTY ELECTRONICS

NOV 86 P27 VOIGTS REVIEW BTREE ISAM APPLIED COMPUTER TECHNOLOGY

NOV 86 P27 REVIEW MICROCOMPUTER DEVELOPMENT SYSTEM TEC IBM 8805

NOV 86 P37 MILLS ARTICLE DRIVES NOT READY FLEX UTILITY DISKS 6809 ASSEMBLY LISTING

NOV 86 P40 LURIE COLUMN FORTH VERSIONS DATE FORTH LISTING

NOV 86 P42 JONES LETTER TSC XBASIC LSET RSET NEW : XBASIC

NOV 86 P43 PETERS OS9 UTILITY ATTACH 6809 ASSEMBLY LISTING

NOV 86 P45 DODGE LETTER OS9/68K C COMPILER BUGS

NOV 86 P47 MOTOROLA ANNOUNCEMENT MC68030 32 BIT CPU

NOV 86 P49 MOTOROLA ANNOUNCEMENT MC68882 ENHANCED EFPCP

DEC 86 P8 ANDERSON USER NOTES PIC HLL DMA 6809 ASSEMBLY LISTING

DEC 86 P12 PASS C USER NOTES ANSI STANDARD PREPROCESSING BTREE C LISTING

DEC 86 P16 VOIGTS BASIC OS9 REGISTERS KBASIC KO ASSEMBLER

DEC 86 P19 FERGUSON ARTICLE CRYPTOQUOTE GAME KBASIC ABASIC 6809 ASSEMBLY LISTING

DEC 86 P22 LURIE FORTH COLUMN C LIBRARY FUNCTIONS CASE CHARTEST FORTH LISTINGS

DEC 86 P26 LUNT ARTICLE CUSTOM FLEX 6809 SYSTEM WD2793 FDC NEWDISK BOOTROM

DEC 86 P29 WILLIAMS MICROWARE GROUNDBREAKING PHOTOS

DEC 86 P36 86 MICRO JOURNAL WRITERS GUIDE EDITORIAL FORMAT TEXT FILES

DEC 86 P37 GREEN ARTICLE ADD WINCHESTER HARD DISK HD TO SWTP SS50 SYSTEM SCHEMATICS

DEC 86 P43 REVIEW TOPS OFFICE PRINT SHOP FOR MAC HD BACKUP HFS LOCATOR PBI SOFTWARE

DEC 86 P45 WILLIAMS ARTICLE MC68HC11 EASY TO USE FORTH BUFFALO MONITOR

DEC 86 P48 SCHNELL LETTER DRAGON COMPUSENSE

DEC 86 P48 LLOYD IO ANNOUNCES VANTAGE EDITOR/CROSS ASSEMBLER/SYNTHETIC DEBUGGER FOR 6 BIT ON OS9MK

DEC 86 P49 JONES LETTER TSC KBASIC LOGIC OPERATORS AND OR NOT

DEC 86 P50 MICROWARE ANNOUNCEMENT OS9 LEVEL II FOR COCO III

DEC 86 P51 MOTOROLA ANNOUNCES REAL TIME OPERATING SYSTEM FOR 68020

EOF

GESPAC EXPANDS TO NEW FACILITY

Mesa, AZ. - Due to the impressive success of its operation, GESPAC moves to larger premises. The company now occupies a new building with 8,000 square feet of office and warehouse space.

A 300% growth in sales in 1986, and an account base of 200 North American Companies required the company to double its staff of sales and engineering support. Furthermore, the company plans to begin production of boards in the U.S. in the second quarter of 1987. The additional space will provide GESPAC with the room necessary to carry out its operations.

The new address is:
GESPAC Inc.
50 W. Hoover Ave.
Mesa, AZ. 85202

GESPAC INCREASES U.S. G-64 PRODUCT OFFERING BY 27 BOARDS

BUSCON, LOS ANGELES, CA.,--GESPAC INC. announces that it will be distributing the G-64 product line of MPL AG in the U.S. and Canada. MPL is a European company located in Zurich Switerland.

MPL has been in the G-64 bus business for six years, and has been designing boards that are complementary to GESPAC'S product line. The MPL catalog includes 27 truly different and complementary boards.

The MPL product line includes several relay boards, isolated analog converters, isolated serial communication cards and other industrial application related products. MPL also has a few CMOS CPU boards based on the 6809 and the 68000.

Under the terms of the agreement GESPAC Inc. will insure the promotion, stocking, technical support and maintenance of the MPL boards in the United States and Canada.

With these 27 new MPL boards, and its own product line of close to 100 function, GESPAC now offers to the American G-64 customer, the largest and most diversified family of board level products from a single vendor.

A 15 page catalog of MPL products is available, free of charge, from GESPAC.

GESPAC OPENS A REGIONAL SALES OFFICE IN SILICON VALLEY

GESPAC opens a regional sales office in Santa Clara, CA. This office, located in the heart of Silicon Valley, is headed by Mr. Richard Soundy.

Mr. Soundy was a long time salesman of board level products in the area, and manager of his own rep. company, before he joined GESPAC in this assignment.

The office is located at:
1333 Lawerence Expressway -150
Santa Clara, CA. 95051
(408) 241-2876

GESPAC opened this office in order to better support its important customer base of Northern and Southern California. It also intends to further develop this important market, which accounts for one third of the national market for the board level products.

GESPAC GOES PUBLIC ON THE SWISS OVER-THE-COUNTER0 MARKET

On September the 15, 1986, GESPAC made a aucceasful initial public offering of its shares on the Swiss over-the-counter market.

The company sold 1800 investment units, each composed of a company share and nine bearer participatory certificates. The proceeds of this sale raised 5.4 millions Swiss Francs (3.2 Million Dollars) of capital. Until this stock offering, the company was privately held by its founders.

The company agreed to the public offering in order to finance its multinational development, including new factories in France and the U. S..The capital raised is also anticipated to fund important research and development projects necessary to maintain strong technological leadership. The company has also invested heavily in state-of-the-art engineering work stations and advanced production equipment for surface mounted technology.

GESPAC was established in 1979 and has focused all of its activities on the design, manufacture and sale of board level products for the industrial OEM. Gespac's boards comply with the standard G-64 bus, which the company defined, and licensed in 1981 to Thomson CSF.

GESPAC and the G-64 bus have grown to a prominent position in the European market. The company is also enjoying signifigant growth in the U.S. where it began its operation in 1984. In 1986, U.S. sales grew 300%, and the company is now serving over 200 accounts.

Reflecting the outstanding reputation of GESPAC in the European electronics industry, the offering was one of the most successful on record in the Swiss securities market. The company's share value doubled in the first two weeks of trading. As of January 15, 1987, the shares were traded at three times their original value.

For more information contact:
Cosma Pabouctsidis, President
GESPAC, Inc.
50 West Hoover Ave.
Mesa, Az. 85202
(602) 962-5559

Continued From Last Month
HEIR UNIX Modifications
by Bradford Taylor

```
    }

    /*
    *** change directory from string
    */
    change_dir(path)
    char *path;
    {
          unsigned trk_sect;        /*  track/sector of file  *

          if(!*path)
           trk_sect = ROOT;       /* default    */
          else if(*path == SLASH)

     {
      trk_sect = ROOT;     /*  start at top  */
      ++path;
     }
    else
     trk_sect = current_dir();   /*  use current directory */


    /*  follow path      */
    if(*path)
     {
      do
       {
        path = parse_name(path);
        /* Check for possible parent move */
        if(strcmp(fname,"..")==0)
         trk_sect = find_is(trk_sect);
        else
         trk_sect = parent(trk_sect);
       }
      while(*path++ == SLASH);
      --path;
     }

    if(*path == '\0')         /* proper end of string  */
     update_current(trk_sect);
    else
     not_found();

}

/*
*** parse directory name
*/
parse_name(path)
char *path;
{
       char c;
       int i;

       for(i=0;i<8;++i)       /*  null out name area  */
        fname[i] = 0;

       for(i=0;i<8 && *path  && *path != SLASH;++i)
        {
         c = *path++;
         fname[i] = (c>='a' && c<='z') ? c : toupper(c);
        }

       return(path);

}

/*
*** Directory not found
*/
not_found()
{
      puts("directory specification error");
      exit();
}

 *

  *** Find track sector of directory
  */
  find_ts(ts)
  unsigned ts;
  {
        int error;

        fp -> drive = drive;        /* Simulate open directory  */
        fp -> buffer[0] = ts>>8;
        fp -> buffer[1] = ts&255;
        fp -> data_index = 0;

        fp -> function = GET_INF;

        /*  Find desired file in directory   */
        do
         {
          error = do_fms(fp);
          if(error == -1 || fp->filename[0] == 0)
           not_found();
         }
        while(!(scompare(fp->extension,"DIR",3)
              !scompare(fp->filename,fname,8)));

        return(to_int(&(fp -> start_track)));

}

/*
*** string compare by count
*/
scompare(p1,p2,cnt)
char *p1,*p2;
int cnt;
{
   while(--cnt >0 && *p1++ == *p2++);

   return(cnt?0:1);
}

/*
*** Make fms call and parse error
*/
do_fms(fp)
struct fcb *fp;
{
     int error;

     error = _fms(fp,'m');     /* call file system */
     if(!fp->error)
      return(error);

     flea_rpterr(fp);        /* report error */
     exit(0);                /* then die     */

}

/*
*** Return track/sector of current directory
*/
current_dir()
{
      fp -> drive = drive;
      fp -> function = IS_OPEN;  /*  Open dir  */
      do_fms(fp);

      fp -> function = GET_INF;  /*  Get information */
      do_fms(fp);

      return(to_int(&(fp -> buffer[24])));

}

/*
*** Update current directory
*/
update_current(ts)
unsigned ts;        /*  new track and sector */
{

      current_dir();     /*  open current directory */
      fp->buffer[24] = ts>>8;
      fp->buffer[25] = ts&255;
      fp->function = WR_SEC;
      do_fms(fp);        /*  write the new information */

}

/*
*** Return track/sector of parent directory
*/
parent(ts)
unsigned ts;
{

      fp->drive = drive;
      fp->current.track = ts>>8;
      fp->current.sector = ts&255;

      fp->function = RD_SEC;
      do_fms(fp);                /* get that information     */

      ts =to_int(&(fp->buffer[4]));


      return(ts?ROOT:ts);      /*  don't backup beyond home  */
```

```
/*
*** convert pointed to bytes to an integer
*/
to_int(ip)
int *ip;
{
      return(*ip);
}

/*****************************************
*                                       *
*  List a directory given a UNIX-like path    *
*                                       *
*  ***CATT [<drive>:][<path>][<match>]      *
*                                       *
*  path ::= a UNIX like directory path      *
*  match ::= defines the type of files CATT *
*            looks at.                  *
*                                       *
*  Example: ***CATT 2:/TOOLS/SOURCES/.TXT   *
*                                       *
*  Prints a short horizontal directory list *
*  given an optional drive and path list.   *
*  Designed to work with the HIER package   *
*  written by Ray Goff.                 *
*                                       *
*  Bradford Taylor                      *
*  Sharm Engineering                    *
*  Box 97                               *
*  Mulvane, KS 67110                    *
*  Tel. (316) 777-0108                  *
*                                       *
*****************************************/
#include <stdio.h>
#include <flex.h>

#define ROOT 0x0005             /* home track and sector   */
#define UCCCFLG (*mem(0xCC49))  /* user-configurable upper-case flag    */
#define UCC     0x40            /* set for upper-case only              */
#define DELETED 255             /* file deleted code       */

/*  GLOBALS   */

struct fcb *fp;
char drive;             /*   drive number                  */
char fname[16];         /*   file name area                */
char dname[9];          /*   current directory name        */
unsigned current;       /*   current directory trk/sector  */
unsigned free_size;     /*   number of free sectors        */
char mask[9],m_ext[4];  /*   Match   information           */

/*
*** Entry
*/
main(argc,argv)
int argc;
char **argv;
{
      char *path;


      fp = &FLEX_DATA.sysfcb;     /* set pointer to system fcb *

      path = argv[1];
      if(argc < 2)
       path = "\0";
       if(isdigit(*path))
        {
         drive = (*path)&15;
         if(*++path == ':')
          ++path;
        }
       else
        drive = FLEX_DATA . work_drive;   /*  default to work drive */

       cwfnct = current_dir();
       open_dir(path);
       show_dir();                      /*  show files in directory */

}


/*
*** open directory from string
*/
open_dir(path)
char *path;
{
      char *ptr;
      unsigned trk_sect,t;      /*  track/sector of file  */

      trk_sect = current;         /*  use current directory */
      if(*path == '/')
       {
        trk_sect = ROOT;    /*  start at top  */
        ++path;
       }

      /*  follow path     */
      t = 0;
      if(*path)
       {
        do
         {
          if(!strcmp("..",path) || !strcmp("../",path,3))
           {
            trk_sect = parsec(trk_sect);
            path += 2;
           }
          else
           {
            path = parse_name(path);
            if((t = find_ts(trk_sect)) != -1)
             trk_sect = t;
           }
         }
        while(*path++ == '/' && t != -1);
        --path;
       }

      if(*path)                 /* proper end of string  */
       not_found();
      if(t == -1)       /*  Match information found */
       {
        for(t=0,ptr=fname;t!=8 && *ptr!='.';++ptr,++t)
         mask[t]=*ptr;
        mask[t] = 0;
        while(*ptr && *ptr!='.')
         ++ptr;

        if(*ptr)                    /*  jump over '.'       */
         ++ptr;
        strcpy(m_ext,ptr,3);    /*  copy over extension */
       }
      else
       mask[0]=m_ext[0] = 0;

      /*  get directory name for target directory */
      read_sector(trk_sect);      /*read target sector   */
      printf("Directory: ");
      printn(&(fp->buffer[8]),8);

      /*  Set up fcb for GET_DIR  calls  */
      store_int(fp -> buffer,trk_sect);
      fp -> data_index = 0;
      fp -> drive = drive;

}

/*
*** parse directory name
*/
parse_name(path)
char *path;
{
       char c,i;

       for(i=0;i!=12 && (c = *path) && c != '/';++i,++path)
        {
         fname[i] = UCCCFLG != UCC? c : toupper(c);
        }
       fname[i] = 0;          /* end string */

       return(path);

}

/*
*** Directory not found
*/
not_found()
{
      flex_puterr("Path name error\4");
      exit();
}

/*
*** Find track sector of directory
*/
find_ts(ts)
unsigned ts;
{

      fp -> drive = drive;        /* Simulate open directory  */
      store_int(fp -> buffer,ts);
      fp -> data_index = 0;

      /*  Find desired file in directory   */
      do
       {
        if(get_info() == -1 || !fp->filename[0])
         return(-1);
       }
```

To Be Continued Next Month

Peter Bendall
Kallieser Stieg 8
2358 Kaltenkirchen
West Germany

29 December 1986

HIGH SPEED CASSETTE & CASSETTE FILE MANAGER
(JPC Products Co)

Dear Don
Thanks very much for your kind help when I phoned today. I would indeed be extremely grateful if you would publish this small note.

I have the JPC Products "TC3" high speed cassette interface running on ALL of my SS50 machines, 6800 and 6809, as well as on a very old 6800 machine that I take to school once a year for Activity Week. These run with the Cassette File Manager "CFM3" operating system from the same company at 4800 baud.

We now have a lot to do with the DRAGON computer, a Spanish made 6809E machine originally designed in Wales(!), that is very similar to the "old faithful" Tandy COCO.

I read in a back number somewhere that JPC once offered a version of the cassette interface and software tailored to the COCO. JPC seem to have disappeared and we would like to get the system running on the Dragon. I would very much like to contact someone who has a working interface or a kit that we could have, or maybe, if JPC are still supported somewhere, they might like to let us have a listing of the software and the circuit of the hardware adaptor.

There is no real problem in reading CFM3 data files on the DRAGON or the COCO since it is a simple timing loop reading a bit on a parallel input port, although there is potentially a small problem with the system timer. Apart from wanting to be compatible with existing users there is a simple facility within DRAGON BASIC (and I assume COCO BASIC as well) whereby two look up table "Stubs", one for Commands and one for Functions, are available for the user to add his own commands to BASIC.

If anyone can help I would be pleased to hear from them. I am reachable by post at the above address and if anyone is close enough to phone I am at home most evenings on international number +49 (Germany) 4191 6536. Also if anyone has access to the international networks I am on BITNET as PETER@DHHEMBL5.

Thanks again for your kind help, I look forward to hearing from someone out there, perhaps there might be an article in it somewhen soon!

Best wishes for 1987
Peter Bendall
DJ0JR, G3NBU
Sysop Dragon-Board

Peter Bendall

MICRONICS
RESEARCH CORP.

Microcomputers - Hardware and Software
GIMIX® Sales, Service and Support

33383 LYNN AVENUE,
ABBOTSFORD,
BRITISH COLUMBIA,
CANADA, V2S 1E2

Dear Don,

H'm! Seems I can't get away from discussing XBASIC. My reader-friends sure keep me supplied with new points of interest, so I decided to reply via 68MJ so that everyone could share in our discoveries.

The first point, which came all the way from Australia, centres on my earlier statement that XBASIC can accommodate a maximum of 255 characters per statement-line (not 127 as stated in TSC's manual). This reader tried several configurations of input-lines, and found that (depending on the nature of the statements in the line) the maximum acceptable to XBASIC was a variable amount. I think the most he could get was a 149-character line. Which is a long, long way from the supposed 255! Perhaps I should have emphasised that the maximum is 255, which is not to say that you will always get up to 255 characters per line.

To elaborate, XBASIC has two 255-character buffers - one of which accepts characters as they are entered from the keyboard (let's call this one A), and the second of which is used to tokenise your input-line and parse it before storing it into your main program (let's call this one B). Now, if you'll cast your mind back to my very first article on this subject, you'll recall my example of a few primitive line-entries and the corresponding tokenised lines, and how amazed we were at the disproportionate amount of code that seemed to be generated. Herein lies the key to the '255' problem. If, due to the nature of the line keyed into Buffer A, the tokenising happens to generate less code in Buffer B than in Buffer A, then you can fill up with 255 characters in Buffer A. On the other hand, if Buffer A generates more code in Buffer B, then Buffer B determines the limit of 255 at which to cut off tokenising. Hope this makes it all clear.

The second point has been raised by a few far-flung friends, which indicates that the subject of random files gives rise to little difficulties now and again. Perhaps I can best illustrate this by reproducing the essence of a little program which appeared in the May 1982 issue of 68MJ. Here's the program :

```
10 PRINT "FILE CREATION PROGRAM"
40 DIM Q$(5)
50 OPEN NEW "CHECKFILE" AS 1
60 FIELD #1, 50 AS Q$(1),50 AS Q$(2) ..... 50 AS Q$(5)
70 FOR I=1 TO 5
80 Q$(1)=" .... 50 SPACES .... "
90 NEXT I
```

```
100 DF=200
110 FOR I=1 TO DF: PUT #1, RECORD I: PRINT I;: NEXT I
150 CLOSE 1: PRINT
180 PRINT "DONE WITH FILE CREATION"
```

Although the above program may appear to be quite straightforward, it nevertheless demonstrates an incomplete grasp of random files and how to create them. So, at the risk of repeating stuff which most readers already know, I'll take it from the beginning. But first, try entering and RONning the program yourself. You should observe the program displaying the opening message, then a 1, followed by a long wait, then a rush of 5 more numbers, another wait, 5 more numbers, and so on till it reaches a count of 200, at which time Line 180's message will be displayed. And then, if you examine the file on disk with DISKEDIT, you'll find 200 records filled with '00's. So what's wrong then???

Let's repeat the above, replacing the 50 SPACEs of Line 80 with a message such as "Hello World" and pad it out to a total of 50 characters. Again we would find a file of 200 records filled with '00's, even though Line 110's job is to create 200 records, each of which should consist of 5 sub-records composed of "Hello World". Why did this not occur? OK, random-file tutorial coming up .....

We are OK in our program down to Line 50, where an instruction to open a file named CHECKFILE on Channel 1 is encountered. As a side-note, the file will not actually be opened until an attempt is made to GET or PUT a record to it. Line 60 is also OK, where we define each record as being composed of, or FIELDed as, 5 sub-records of 50 characters each. But then we proceed, in Lines 70 - 90. to completely cancel our line 60. You see, you can only put data into, or modify, the various FIELDs by means of the LSET or RSET statements. By simply defining the 5 Q$s with an ordinary LET (or implied LET), any prior FIELD definition using the same variables is effectively wiped out. Line 80 should therefore have read :

```
80 LSET Q$(I)="message"
```

without the need of any trailing SPACEs, as the FIELD statement would automatically pad out to the defined length of 50.

Our program is quite workable at this stage, but why does the displayed count go in little spurts? The explanation is that when a random file is first created it allocates only one record to the file. Actually it reserves 3 sectors, but the first two are used for 'housekeeping' purposes, Sector 3 being our actual Record #1. So, when our Line 110 (110 - 140 in the original program) PUTs the first record to the file, all available sectors have been used up, so the FMS section of DOS takes time out to extend the file a little before it can PUT a few more. The original program is, of course, PUTting absolutely nothing into each record - not even a subrecord, due to the cancellation of the FIELD statement - which accounts for the completely NUL file. All that the PUTs are doing is to extend the file a bit at a time as the program loops from 1 to DF.

A better approach would have been to extend our file out to 'DF' (that is, 200) records before we begin PUTting anything at all. To extend a random file, we merely PUT to a non-existent record, and the file will automatically be extended to that limit. Thus :

```
100 DF=200: POT #1, RECORD DF
```

It will take a little while to extend to 200 records, but then you should see the numbers 1 to 200 just clicking up on the screen with no pauses between.

I'm beginning to get a little feed-back on my RBASIC, with quite useful suggestions on features to add. I shall wait a while, then pick the most useful for incorporation. A limiting factor is going to be that I only have about 7 unused tokens left, one of which I've already reserved for the statement 'FLEX', rather than the present command of the same name. I'll go into the rationale of this in a later submission. So now I'm down to 6 tokens only, and even then I'd like to reserve maybe two for some future enhancements not anticipated at this time. Some good news for 68000 owners - some of my friends have convinced me that a 68K version of RBASIC would be appreciated, so this is now in the works. Going to be quite a job, but not an impossible one (I hope). This 68000 code takes a little getting used to!

I'll keep in touch.

Don Williams,
68 Micro Journal.
5900 Cassandra Smith Road,
Hixson, TN 37343

Sincerely,

R. Jones
President

Charles River NEWS

FOR ADDITIONAL INFORMATION:

Edgar R. Geithner
Geithner McGowan, Inc.
(617) 875-3821

FOR IMMEDIATE RELEASE

CHARLES RIVER DATA SYSTEMS CLOSES CONTRACT WITH RADIO NEW ZEALAND

FRAMINGHAM, Mass., Jan. 22 -- Radio New Zealand, the government radio agency, has placed a substantial order for Charles River Data Systems's new 68020-based Universe"/400 super microcomputers.

The order was placed through Mitsui Computer Systems, Ltd., the Charles River distributor in New Zealand.

The systems will be located in radio stations throughout New Zealand and will track advertising time each station has available for sale and generate advertising invoices from a central location. In addition, the systems will support each station's advertising sales, traffic, and accounting activities, and produce its daily station log.

The Universe 400 is based on the industry standard VMEbus, and can use either a Motorola 68000 or 68020 microprocessor. It has a memory capacity of up to 16M bytes, and can support up to 4.2G bytes of disk storage. It has 12 VME slots, up to eight of which are available for expansion. A typical Universe/400 is priced under $10,000.

Charles River Data Systems was founded in 1973 and introduced the Universe 68, the first 32-bit, computer system based on a microprocessor, in September, 1981. The company now offers the VMEbus-based Universe/200, Universe/400, and Universe/600 computer systems, and a VERSAbus-based family of computers. Both computer families use the Motorola 68020 and 68000 microprocessors, and run under the UNOS* operating system, which is derived under license from AT&T UNIX® System V.

TIME .CMD

Kenneth Drexler

Dear Mr. Williams:

It has been several months since I have done my share to keep '68 Micro Journal plump. I hope you can stand another real-time-clock-for-FLEX article.

This attached article, "TIME.CMD", describes both the hardware and software needed to implement such a clock on a FLEX computer. The hardware is simple and inexpensive, approximately $10 at current prices. The software is designed to install itself in FLEX and then keep the date registers in FLEX up to date whether the computer is on or off. The software can be run by a STARTUP.TXT file and it will automatically start and maintain FLEX's date registers. The article gives the details. The article and software is all original and is my property. It was suggested by the article entitled, "Clock/Calendar for 6809", which appeared in the July 1981 Microcomputing magazine.

Now that there are several 680X0 machines which run SK*DOS/68K and OS-9/68K, it would be great to see a series of articles which surveyed the available machines – not just the Mustang line. You ran such a series on the 6809 boards. (The benchmarks 68 Micro runs suggest that the Mustang machines will come out just fine in such a series.) Hazelwood, Emerald, Peripheral Technology, GIMIX, Mizar, TLM and, perhaps, others have 680X0 offerings running OS-9/68K or SK*DOS/68K.

Keep up the good work.

Sincerely,

Kenneth Drexler
365 Drake's View Drive
Inverness, California 94937
(415) 485-1330

Editor's Note: Thanks Ken for the article. For 10 bucks, can't go wrong.
As to the benchmark series; we have tried. I offered to run them here but none of the others seemed too keen on that idea. Then we offered to accept reviews from others who might have done likewise. Again, we drew a blank. Fact is, several stated flatly that they just did not want to publish their figures. The ones we publish should answer that. However, should we get any new figures, on any other machine, in the same ball-park, I will let all of you know.

DMW

TIME.CMD and the simple hardware described below will give you a real time clock for your FLEX computer. It will also let your computer keep its own date registers up to date whether the computer is on or off. The only requirements for use of this hardware and software are an unused PIA (Peripheral Interface Adapter) and FLEX or a compatible operating system such as SK*DOS.

The hardware used by TIME consists of a MSM5832 clock chip and a PIA. It costs less than $10. When used with the TIME.CMD listed below, this hardware allows you to display the current time, set the MSM5832 chip, install the software into FLEX which keeps the date current, disable the up date software and display a help message. TIME.CMD can be called from a STARTUP.TXT file and will automatically install itself, set FLEX's date registers and display the date and time.

The Hardware

The hardware involved in this project could not be simpler. It consists of the following seven parts:

MSM5832 Chip
Unused PIA
32.768 KHz Crystal
2 - 22 pf. caps.
1N914 Germanium Diode
3-5 volt battery

That is it. No pullups, no interface gates, switches or transistors. The cost is under $10.

The heart of the clock is the MSM5832 clock chip. This chip is a 16-pin integrated circuit containing a CMOS real time clock. The chip contains 13 registers in which it keeps the current date, time and day of the week. It has four address, four data and four control lines which allow the computer to read and write the chip's data registers. It uses a 32.768KHz crystal. The access time for the MSM5832 is up to 6 microseconds so the chip cannot be driven directly by the 6809. To deal with this fact, in this design the clock chip is driven through a PIA.

The circuit used with the MSM5832 clock chip is shown below. It uses both halves of an unused

PIA. Because of the simplicity of the circuit, no fancy installation is needed. The MSM5832 can be mounted in a spare socket or it can be glued on an vacant corner of the board where the unused PIA is located.

When connecting the MSM5832, it is important to connect it to the PIA as shown in the schematic below. The secret of the circuit's simplicity is in the hookup shown: the MSM5832's data lines are connected to PIA pins PA0 to PA3 and the MSM5832's address and control lines are connected to PIA pins PB0 to PB7. The reason for this using this hookup is that the input/output lines on the B-side of the PIA are three-state and do not have pullups. This means that when the PIA is reset or when outputs are not enabled, the B-side pins float, neither high nor low. This allows the pulldown resistors built into the MSM5832 to pull the control and address lines down to their inactive state (low). As a result, resetting the PIA has no adverse effect on the data stored in the clock chip.

There is another advantage of using a PIA to directly drive the MSM5832. The PIA has an output high voltage specification which is sufficiently high to more than meet the input high voltage specification of the MSM5832. If a regular TTL circuit were used, pullup resistors might be required.

The battery backup circuit consists of two diodes which isolate the +5 power supply and the backup battery. A germanium diode is used between the backup battery and the clock chip to minimize the voltage drop between the battery and the MSM5832. This allows use of a compact, low cost 3 volt lithium battery for backup. (The MSM5832 needs a minimum of 2.2 volts for backup.)

The Software

The source code for TIME.CMD is listed below. It is written in 6809 assembly language and is designed for use on a computer using General FLEX sold by Technical Systems Consultants. However, the program can be modified for assembly by a 6800 or 6802 or for other versions of FLEX. It can also be used with any FLEX-compatible operating system such as SK*DOS. The code is heavily commented and is mostly straight forward.

TIME has five functions: Display Time, Start Clock, Set Clock, Stop Clock and Help. An option character typed on the command line after the "TIME" program name determines which function is executed.

Start Clock is the most complex function. When it is called, it checks to see if the MSM5832 clock chip contains a valid time and date. If the data is not valid, the program prompts the user for the correct data. Once valid data is obtained, this data is used to set FLEX's date registers in RAM at $CC0E and in the MSM5832. The clock chip then is checked again for valid time and date data. If the data is valid, the program then moves the code which keeps FLEX's date registers up to date out of the utility command space where TIME runs. This code segment is then linked to FLEX's WARMS entry point through the DWARM vector (see below). Once this is done, whenever a program calls WARMS, the update code reads the current date from the MSM5832 and updates FLEX's date registers, if necessary.

The code which handles the updating of FLEX's date registers must remain in RAM after TIME is run. In order to allow other commands to use the utility command space in FLEX, the update code is moved to RAM at the location named RAMLOC in the source listing. RAMLOC should be set to an unused section of RAM which is at least $78 bytes long. If no such RAM is available, RAMLOC can be set to $0, in which case the date update code will automatically be placed at the top of RAM, below MEMEND. MEMEND is then adjusted to protect the moved code.

The installation code in TIME listed below uses the DWARM entry point in FLEX's disk drivers. This entry point is a part of general version of FLEX and, perhaps, others. This entry point is called whenever WARMS is executed. It is designed to perform any needed updating of the disk drivers. DWARM is located at $DE18 and consists of a JMP instruction ($7E) followed by a two byte address. The program preserves the DWARM vector by storing the address portion of the JMP instruction in the date update code. The address of the date update code is then substituted for the address in the DWARM jump. After a date update is completed, the program jumps to the address to which the DWARM call originally pointed.

If your version of FLEX has a $7E at $DE18, the code below can be used without change. If it

does not have a jump instruction at that location you will have to modify the program listed below to modify the address part of the WARMS vector itself.

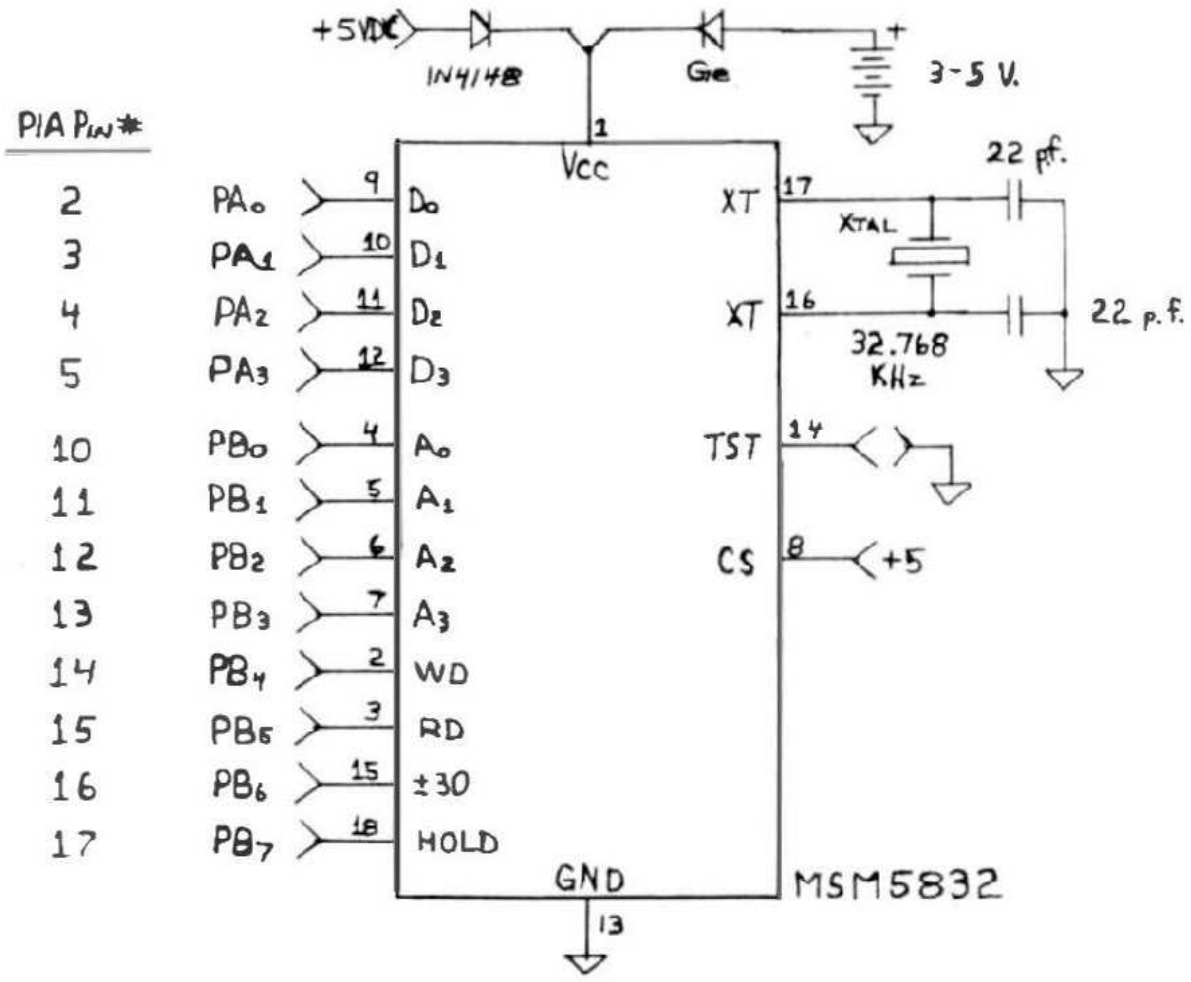
The only other tricky part of the TIME program is the storage of the data needed to set the MSM5832 chip and confirm that it is within the chip's range. The complexity involves the fact that all of the information needed to set the clock chip is two characters long except the day of the week, which is one character. This would not be a problem except that the day of the week character is in the middle of the register set of the MSM5832. To compensate for this and to allow a compact clock setting routine, SETCLOCK reads a two character day-of-the-week number and then adjusts the data to eliminate the extra byte before the data packet is stored in the clock chip. The code accepts both upper and lower case answers to its questions. As written the code maintains time in 2400 hour format.

The Stop Clock function uses the copy of the address portion of DWARM which is stored in the date update code to restore the original DWARM call. Stop clock does not attempt to restore MEMEND to its original value because it has no way of knowing if other programs have been placed in RAM below the update code.

In order to use the software, there are only two equates which must be set in the source listing. These are RAMLOC and CLKADR. RAMLOC is discussed above. CLKADR is the address of the PIA to which the MSM5832 is attached. All other addresses used in the listing are FLEX standard addresses and need not be changed. If your version of FLEX does not have a JMP instruction at $DE18, you will have to slightly rewrite the code between TIME2 and TIME3 and between TIME8 and TIME10 to point at WARMS+1 rather than DWARM+1.

Enjoy.

TIME.CMD HARDWARE



```
 NAM TIME PROGRAM - VERSION 1.0
 OPT PAG
 PAG
*************************************************
*
*  TIME COMMAND
*
*  Date: April 8, 1986
*  Revised December 22, 1986
*  By Ken Drexler, 365 Drakes View Drive
*  Inverness, California
*
*  This command starts and runs a MSM5832 clock
*  chip on a Flex 9 system. The program can be
*  called from the STARTUP file or used as a
*  utility.
*
*  See, "Clock/Calendar for the 6809" by
*  David R. Rawson, Microcomputing, July 1981.
*  page 132.
*
*  Command format:  "TIME <Options>"
*
*  Options:          None     Display Time and Date
*                    +        Start Clock
*                    S        Set Clock
*                    Z        Stop Clock
*                    ?        Help Message
*
*  Startup is the most complex function:
*  The program first checks if the clock chip has
*  valid data. If it does it moves code to RAM
*  which will use the chip to set the Flex date
*  registers on every warm start of Flex. This will
*  keep Flex up-to-date.
*
*  If the clock data is not valid, the program asks
*  for the current date and time and uses this
*  information to set the Flex date registers and
*  the clock chip. The clock chip is then checked
*  for valid data again. If the data is valid, the
*  code which will set the Flex registers on warm
*  start is moved to RAM and enabled.
*
*  If the Clock is set, the program exits by
*  displaying the current date and time.
*
*  Hours are entered using a 24 hour format.
*
*  The Warm start code is placed in RAM at RAMLOC
*  or, if RAMLOC = 0, it is placed at the top of
*  memory and MEMEND is adjusted to below it.
*
*************************************************
 PAG
*************************************************
*
* SYSTEM EQUATE
FLEX EQU $C000
*
* RAM EQUATES
RAMLOC EQU $F180 LOC. TO STORE SETSYS ROUTINE

* FLEX EQUATES
WARMS EQU FLEX+$0D03
GETCHR EQU FLEX+$0D15
PUTCHR EQU FLEX+$0D18
INBUFF EQU FLEX+$001B
PSTRNG EQU FLEX+$0D1E
PCRLF EQU FLEX+$0D24
NXTCH EQU FLEX+$0D27
GETDEC EQU FLEX+$0D48
DWARM EQU FLEX+$1E18 DISK WARM START (JMP)
EOLCHR EQU FLEX+$DC02 END OF LINE CHAR.
SYSMON EQU FLEX+$C0E SYSTEM MONTH
SYSDAY EQU FLEX+$C0F SYSTEM DAY
SYSYR EQU FLEX+$C10 SYSTEM DAY
MEMEND EQU FLEX+$0C2B
*
* MSM5832 EQUATES
CLKADR EQU $F40C CLOCK ADDRESS
DATREG EQU 0 DATA REG.
DATCTL EQU 1
ADDREG EQU 2 ADDRESS REG.
ADRCTL EQU 3

* ADDRESS REGISTER EQUATES
WRITE EQU $10
READ EQU $20
ADJ EQU $40
HOLD EQU $80

* DATA REGISTER EQUATES
SEC1 EQU 0
SEC10 EQU 1
MIN1 EQU 2
MIN10 EQU 3
HR1 EQU 4
HR10 EQU 5 D3: 1=24hr, 0=12hr
DOW EQU 6
DAY1 EQU 7
DAY10 EQU 8 D2: 1=Leap, 0=Not Leap
MON1 EQU 9
MON10 EQU 10
YR1 EQU 11
YR10 EQU 12
 PAG
*************************************************
*
 ORG FLEX+$100
TIME BRA TIME1

VER FCB 10 VERSION 1.0

* VARIABLES AND DATA

* VARIABLES
*
DATA RMB 14 (MUST be even!)

MAX EQU * MAXIMUM DATA VALUE TABLE
 FDB 59,23,6,31,12,99
 SPC 3
* SELECT OPTIONS
TIME1 JSR NXTCH GET NEXT CHARACTER
 CMPA #$0D NO PARAMETERS, DATE, TIME ONLY
 LBEQ TIME10
 CMPA EOLCHR END OF LINE?
 LBEQ TIME10 YES, DATE,TIME ONLY
 CMPA #'+ STARTUP?
 BEQ TIME4
 ANDA #$5F FORCE UPPER CASE
 CMPA #'Z CLOCK OFF?
 BEQ TIME2
 CMPA #'S SET CLOCK CHIP?
 BEQ TIME5

* ANYTHING ELSE IS HELP
 LEAX HELPMS,PCR PRINT HELP MESSAGE
 BRA TIME7 PRINT AND EXIT

* TURN OFF CLOCK, RESTORE ORIGINAL DWARM CODE
TIME2 LDX DWARM+1 GET DWARM VECTOR
```

```
* CHECK CLOCK CODE INSTALLED
 LDD ,X GET DATA THERE
 CMPD SETSYS TIME CODE?
 BNE TIME3 TIME CODE NOT INSTALLED
 LDD RETRN-SETSYS,X GET OLD VECTOR
 STD DWARM+1 REPLACE IT
TIME3 LDX #CLKADR POINT AT CLOCK CHIP
 CLRB
 STB ADRCTL,X
 STB ADDREG,X
 LDB #$04
 STB ADRCTL,X
 BRA EXIT

* STARTUP CLOCK
TIME4 LBSR INITRD INITIALIZE CLOCK, GET ADDR IN X
 LBSR HOLDON STOP CLOCK
 LBSR CKSET CLOCK DATA VALID
 BCC TIME8 DATA OK
TIME5 LBSR SETCLOCK GET DATA, SET FLEX, CHIP
 LBSR CKSET SET NOW?
 BCC TIME8 YES, CONTINUE
 CLRB NO, RELEASE HOLD AND EXIT
 STB ADDREG,X CLEAR HOLD
TIME6 LEAX NOTVAL,PCR PRINT INVALID DATA MSG.
TIME7 JSR PSTRNG
 BRA EXIT

* SET UP CODE FOR FLEX WARMS
TIME8 LDD DWARM+1 SAVE OLD JUMP DESTINATION
 STD RETRN
 LEAX ENDMOV,PCR SET END POINTER
 PSHS X STACK IT
 LEAX BEGMOV,PCR GET START
 LDY #RAMLOC GET DESTINATION

* IF RAMLOC ZERO, MOVE MEMEND
 BNE TIME9 SKIP IF <>0
 LDD MEMEND GET OLD MEMEND
 SUBD ENDMOV-BEGMOV
 STD MEMEND SAVE RESULT
 TFR D,Y MOVE DESTINATION TO Y
TIME9 PSHS Y SAVE LOCATION OF CODE
TIME91 LDA ,X+ MOVE DATA
 STA ,Y+
 CMPX 2,S DONE?
 BNE TIME91
 PULS X,Y GET DESTINATION, CLEAN STACK
 STX DWARM+1 STORE LOC. OF CODE IN DWARM OPERAND
TIME10 LBSR PRDATE PRINT DATE/TIME
EXIT JMP WARMS
 PAG
* THE FOLLOWING PROGRAM SEGMENT IS MOVED OUT
* OF THE UTILITY SPACE AND LINKED TO WARMS.
* IT KEEPS THE FLEX DATE REGISTERS CURRENT.
BEGMOV EQU *

* SET FLEX DATE REGISTERS
*
SETSYS BSR INITRD SET PIA, GET CLKADR
 BSR HOLDON STOP CLOCK
 LDB #DAY1
 BSR RD2BIN
 STB SYSDAY
 LDB #MON1 GET MONTH
 BSR RD2BIN
 STB SYSMON
 LDB #YR1
 BSR RD2BIN
 STB SYSYR
 CLRB CLEAR HOLD
 STB ADDREG,X
 JMP >0 TWO BYTES, TO BE PATCHED BEFORE USE

RETRN EQU *-2
 SPC 3
** INITIALIZE PIA AND CHECK CHIP
* OUT:  X = CLOCK ADDRESS
*
INITRD LDX #CLKADR POINT AT CLOCK
 CLRA
 STA DATCTL,X
 STA ADRCTL,X
 LDB #$3E
 STD DATREG,X SET DDRA = INPUT
 COMA
 STD ADDREG,X SET DDRB = OUTPUT
 RTS
 SPC 3
* READ CLOCK DIGIT TO A
*  IN:  X - CLOCK
*       B - REGISTER NO.
*       PIA INITIALIZED FOR READ
*       CLOCK ON HOLD
* OUT:  A - BCD DATA
*       B,X,Y PRESERVED
*
RDIGIT PSHS B,Y SAVE REGS
 LEAY <MSKTBL,PCR POINT AT MASK TABLE
 ORB #HOLD+READ ENABLE READ
 STB ADDREG,X
 NOP DELAY
 LDA DATREG,X
 PULS B GET NO.
 ANDA B,Y MASK DATA
 PULS Y,PC

* MASK TABLE
MSKTBL FCB $F,$7,$F,$7,$F,$3,$7,$F,$3,$F,$1,$F,$F
 SPC 3
* READ TWO CLOCK DIGITS TO D
*  IN:  X - CLOCK
*       B - FIRST REG. TO READ
*       PIA INITIALIZED FOR READ
*       CLOCK ON HOLD
* OUT:  D - BCD DATA A:MSB B:LSB
*
RD2DIG BSR RDIGIT GET FIRST DIG.
 PSHS A SAVE FIRST
 INCB
 BSR RDIGIT MSB TO A
 PULS B,PC LSB TO B, RETURN
 SPC 3
* READ BINARY DIGIT ROUTINE
*  IN: X - CLOCK
*      B - 1ST REG. NO.
*
RD2BIN BSR RD2DIG GET DIGITS, FALL THRU TO . .

* BCD TO BINARY ROUTINE
*  IN:  D - 2 BCD DIGITS A:MSB B:LSB
* OUT:  B - BINARY SUM OF DIGITS
*
BCDBIN PSHS B SAVE LSB
 LDB #10 MSB*10
 MUL
 ADDB ,S+ ADD LSB
 RTS
 SPC 3
* CLOCK HOLD ON ROUTINE
*  IN: X - CLOCK ADDRESS
*
HOLDON PSHS A SAVE A
 LDA #HOLD
```

```
 STA ADDREG,X
 PULS A RESTORE A AND FALL THRU TO . . .
 SPC 3
** DELAY ROUTINE
*  DELAY = 160 USEC.
*
DELAY PSHS  A
 LDA #16
DELAY1 LBRN DELAY 5 CYCLES
 DECA 2 CYCLES
 BNE DELAY1 3 CYCLES
 PULS A,PC
 SPC 3
ENDMOV EQU *
* END OF MOVED PROGRAM SEGMENT
 PAG
*****************************************
*
* READ TWO ASCII DIGITS
*
*  IN:  D - 2 BCD DIGITS
* OUT:  D - 2 ASCII DIGITS A:MSB B:LSB
*
RD2ASC BSR RD2DIG READ 2 DIGITS
 ADDD #$3030 CONV. TO ASCII
 RTS
 SPC 3
* CHECK CLOCK DATA FOR RANGE
*  IN:  X - CLOCK
*       PIA INITIALIZED FOR READ
*       CHIP HOLD ON
* OUT:  CARRY =1 IF INVALID, ELSE =0
*
CKSET LDB #SEC1 SET REG.
 BSR RD2BIN
 CMPB #59 IN RANGE?
 BGT BADIG
 LDB #MIN1
 BSR RD2BIN
 CMPB #59
 BGT BADIG
 LDB #HR1
 BSR RD2BIN
 CMPB #24
 BGT BADIG
 LDB #DOW
 BSR RDIGIT
 CMPA #6
 BGT BADIG
 LDB #DAY1
 BSR RD2BIN
 CMPB #31
 BGT BADIG
 LDB #MON1
 BSR RD2BIN
 CMPB #12
 BGT BADIG
 LDB #YR1
 BSR RD2BIN
 CMPB #99
 BGT BADIG
 CLRB
 RTS

BADIG COMB
 RTS
 SPC 3
* PRINT DATE ROUTINE
*
PRDATE LBSR INITRD INZ. PIA, GET CLKADR
 BSR HOLDON STOP CLOCK
 LDD #$0D0A PRINT CR LF
 BSR PRNTD
 LDB #MON1 PRINT MONTH
 BSR RD2BIN
 DECB ADJUST TO ZERO BASE
 PSHS X
 LEAX MONTBL,PCR POINT AT NAMES
 LDA #3
 MUL
 LEAX D,X POINT AT STRING
 BSR PRNT3S PRINT 3 CHAR. NAME
 PULS X
 LDB #DAY1
 BSR PR2DIG PRINT DAY
 LDD #',*256+$20
 BSR PRNTD
 LDD #256*'1+'9
 BSR PRNTD
 LDB #YR1 PRINT YEAR
 BSR PR2DIG
 LDA #$20 SPACE
 BSR OUTCHR
 LDB #DOW PRINT NAME OF DAY
 LBSR RDIGIT GET DOW # IN A
 LDB #3
 MUL
 PSHS X SAVE X
 LEAX DAYTBL,PCR POINT AT TABLE
 LEAX B,X POINT AT STRING
 BSR PRNT3S
 PULS X RESTORE X
 LDB #HR1 PRINT HOUR
 BSR PR2DIG
 LDA #':
 BSR OUTCHR
 LDB #MIN1
 BSR PR2DIG
 LDA #':
 BSR OUTCHR
 LDB #SEC1
 BSR PR2DIG
 LDD #$0D0A CRLF
 BSR PRNTD
 CLRB RELEASE HOLD
 STB ADDREG,X
 RTS

PR2DIG LBSR RD2ASC GET 2 ASCII, FALL THRU TO . . .

PRNTD BSR OUTCHR PRINT A
 TFR B,A

OUTCHR JMP PUTCHR

* PRINT THREE CHAR. STRING
*  IN:  X - ADDR. OF STRING
*
PRNT3S LDB #3 SET COUNTER
P3S1 LDA ,X+ GET CHAR
 BSR OUTCHR
 DECB
 BNE P3S1
 LDA #$20 SPACE
 BRA OUTCHR
 SPC 3
** SET CLOCK ROUTINE
* OUT: X - CLOCK ADDRESS
*      CLOCK CHIP SET
*      FLEX DATE REGISTERS SET
*
SETCLOCK LBSR INITWR SET PIA -WRITE, GET CLKADR

* GET DATA IN INVERSE REGISTER ORDER (2 BYTES EACH)
```

```
 LEAU MAX,PCR POINT AT END OF DATA
 LEAY INTRO,PCR
 BSR PRSTR PRINT INTRO AND YEAR?
 LBSR GETNUM
 BSR PRSTR MONTH?
 LBSR GETNUM
 BSR PRSTR DAY?
 BSR GETNUM
 BSR PRSTR DAY OF THE WEEK?
 BSR GETNUM
 BSR PRSTR HOUR?
 BSR GETNUM
 BSR PRSTR MINUTE?
 BSR GETNUM
 CLR ,-U SEC1,SEC1D = 0
 CLR ,-U U NOW POINTS TO DATA START
 BSR PRSTR LEAP YEAR?
 JSR GETCHR
 ANDA #$5F FORCE UPPER CASE
 CMPA #'Y
 BNE SCLK2 NOT LEAP YEAR
 LDA DAY1D+1,U
 ORA #$04 SET LEAP YEAR BIT
 STA DAY1D+1,U
SCLK2 LDA HR1D,U
 ORA #$08 SET 2400 HOURS MODE
 STA HR1D,U
 BSR PRSTR REDO?
 JSR GETCHR
 ANDA #$5F FORCE UPPER CASE
 CMPA #'Y
 BEQ SETCLOCK

* ADJUST DATA TO ELIMINATE 2ND DOW BYTE
 LEAX DOW+1,U
 LDB #6 SET COUNTER
SCLK3 LDA 1,X
 STA ,X+
 DECB
 BNE SCLK3
 LDX #CLKADR
 BSR SET PUT DATA IN CHIP
 BSR PRSTR PUSH KEY TO START
 JSR GETCHR
 CLRA
 STA ADDREG,X RELEASE HOLD
 STA DATCTL,X SET DDRA TO INPUT
 STA DATREG,X
 LDA #$3E
 STA DATCTL,X

* SET FLEX DATE REGISTERS
 LDB #DAY1 DAY
 BSR GETBIN
 STB SYSDAY SET FLEX
 LDB #MON1 MONTH
 BSR GETBIN
 STB SYSMON
 LDB #YR1 YEAR
 BSR GETBIN
 STB SYSYR
 RTS
 SPC 3
** SET CLOCK SUBROUTINES
*
** PRINT PROMPTS
*  IN: Y - POINTER TO STRING DATA
*
PRSTR JSR PCRLF PRINT LINE
 BRA PRSTR1

PRSTRD JSR PUTCHR PRINT CHAR.

PRSTR1 LDA ,Y+ GET NEXT CHAR.
 CMPA #$04 END?
 BNE PRSTRD NO, PRINT IT
 RTS
 SPC 3
* GET BINARY ROUTINE
*  IN: B - REG. NO. OF DATA
*      U - DATA STORAGE
* OUT: B - BINARY VALUE OF 10*[REG,U]+[REG+1,U]
*
GETBIN LDD B,U GET DATA
 EXG A,B
 LBSR BCDBIN CONV. TO BINARY
 RTS
 SPC 3
** GET NUMBER ROUTINE
*
*  IN:  U - DESTINATION FOR DATA
*       D - HEX DIGIT
* OUT:  D - TWO BCD DIGITS STORED [,--U]
*       THE MSBYTE IS STORED AT THE HI. ADDR.
*
GETNUM JSR INBUFF
 JSR GETDEC
 BCS GETERR ERROR
 TSTB
 BEQ GETERR ERROR
 CMPX 10,U IN RANGE?
 BGT GETERR NO
 TFR X,D MOVE TO D
 BSR DTOBCD
 EXG A,B ADJUST POSITION
 STD ,--U SAVE IT
 RTS

GETERR LEAX ERRORS,PCR PRINT ERROR MSG
 JSR PSTRNG
 BRA GETNUM
 SPC 3
** HEX TO BCD ROUTINE
*  NOTE:  RANGE IS LIMITED TO 0 TO 99 DECIMAL.
*  IN:  0 - HEX DIGITS
* OUT:  D - TWO BCD DIGITS
*
DTOBCD CLRA
DBC01 SUBB #10 LSB - 10
 BCS DBCD2 TOO MUCH?
 INCA COUNT 10'S
 BRA DBCD1

DBCD2 ADDB #10 ADD 10 BACK
 RTS
 SPC 3
** INITIALIZE PIA AND CHECK CHIP
* OUT:  X = CLOCK ADDRESS
*
INITWR LDX #CLKADR GET CLOCK ADDRESS
 CLRA INITIALIZE PIA
 STA DATCTL,X
 STA ADRCTL,X
 COMA
 STA DATREG,X SET DDRA AND DDRB TO OUTPUTS
 STA ADDREG,X
 LDA #$3E SET PIA STATUS
 STA DATCTL,X
 STA ADRCTL,X
 RTS
 SPC 3
** SET DATA IN CLOCK CHIP
*  IN:  X = CLOCK ADDRESS
*       U = DATA STORAGE
*
```

```
SET LBSR HOLDON STOP CLOCK
 LDB #HOLD+SEC1
 PSHS U
SET1 STB ADDREG,X
 LDA ,U+
 STA DATREG,X
 ORB #WRITE
 STB ADDREG,X
 ANDB #$FF-WRITE
 STB ADDREG,X
 CMPB #HOLD+YR10
 BEQ SET2
 INCB
 BRA SET1

SET2 PULS U,PC
 SPC 3
** MESSAGES AND TABLES
*
** MONTH TABLE
*
MONTBL FCC /JanFebMarAprMayJun/
 FCC /JulAugSepOctNovDec/

DAYTBL FCC /SunMonTueWedThuFriSat/

** MESSAGES
*
NOTVAL FCC /Clock data invalid!!/
 FCB 7,4

INTRO FCC /Set Date & Time:/
 FCB $D,$A

YEAR FCC /Year?  19/
 FCB 4

MONTH FCC /Month?  /
 FCB 4

DAY FCC /Day?  /
 FCB 4

DOWN FCC /Days after Sunday?  /
 FCB 4

HOUR FCC /Hour? (2400 hour clock): /
 FCB 4

MINUTE FCC /Minute?  /
 FCB 4

LPYR FCC /leap Year? (Y or N): /
 FCB 4

ERROR1 FCC /Redo entries? (Y or N): /
 FCB 4

START FCC /Press any key to start Clock >>>>/
 FCB 4

ERRORS FCC /Error, ReEnter Data:  /
 FCB 7,4

HELPMS FCC /Usage:  TIME <Option>/
 FCB $D,$A
 FCC /Options: (None) - Display date-time/
 FCB $D,$A
 FCC /   + - Start Clock/
 FCB $D,$A
 FCC /   S - Set Clock/
 FCB $D,$A
 FCC /   Z - Stop Clock/
 FCB $D,$A
 FCC /   ? - Help/
 FCB 4

 END TIME
```

EOF

Classifieds *As submitted - No Guarantees - As is!*

DAISY WHEEL PRINTERS
Qume Sprint 9 - $900 Qume Sprint 5 - $800

Winchester 10 Megabyte Drive - Seagate Model #412 $275.

3 - Dual 8" drive enclosure with power supply. New in box. $125 each.

5 - Siemens 8" Disk Drives. $100 each.

Tano Outpost II, 56K, 2 5" DSDD Drives, FLEX, MUMPS $495.

TELETYPE Model 43 PRINTER - with serial (RS232) interface and full ASCII keyboard. $250 ready to run.

SWTPC S/09 with Motorola 128K RAM, 1-MPS2, 1-Parallel Port, MP-09 CU Card - $900 complete.

CDS-1- 20 Meg Hard Disk System with controller $500.

(615) 842-4600 M-F 9 AM to 5 PM EST

•••

Wanted to buy DMF-3 Boards. Please Call
Doug (208) 939-8813.

•••

Wanted 1 Hazelwood MC 20 Disk Controller.
John Searfoss (512) 781-2361.

•••

S/09 w/64K, dual DSSD mini-floppies, 4 serial, 3-parallel, CT-82 terminal, MX-80 w/Graftrax, FLEX, software. Excellent condition. Make offer.
Keith (405) 624-0621 evenings.

•••

Several SS50, SS30 cards,reasonable. Send SASE for complete list or call.
(503) 485-2796 Crosby Stone, 1941B W17th St. Eugene,Or.97402

FOR THOSE WHO NEED TO KNOW **68 MICRO JOURNAL™**

Call for Software:
68000, C, Basic09
Sculptor

We are receiving calls and letters from numerous sources, including users, business and others looking for OS-9 68000 software; applications, etc.

Many of you have developed software that with little change could be adapted for others. If you are interested in selling it, please let us know. There is a growing market out there now. Get in on the ground floor!

If you can use additional income and have something that might be of interest, call and talk to Larry or Don.

S.E. MEDIA Division - CPI
POB 849
Hixson, TN 37343

Telephone (615) 842-6809
Telex (510) 600-6630

SK★DOS™

The Generic DOS™ for 68000 applications in

- ★ Industrial Control
- ★ Business Use
- ★ Educational Computing
- ★ Scientific Computing
- ★ Number Crunching
- ★ Dedicated Systems
- ★ Turnkey Systems
- ★ Data Collection
- ★ Single-Board Computers
- ★ Bus-oriented Computers
- ★ Graphics Workstations
- ★ One-of-a-kind Systems
- ★ Advanced Hobbyist Use

SK★DOS is a single user disk operating system for computers using Motorola 32 bit CPUs such as the 68008, 68000, 68010, and 68020. It provides the power of a full DOS, yet is simple and easy to use, and will run on systems from 32K to 16 megabytes. Because SK★DOS is easily implemented on a new system, we call it "The Generic DOS™" which allows programs written for one system to be run on many others.

SK★DOS comes with over 40 commands and system programs, including a 6809 emulator which allows 68K SK★DOS to run application programs and languages developed for 6809 SK★DOS and other systems. Assemblers, editors, and higher level language support are available from third party software vendors and through public domain software.

SK★DOS is available for single-copy or dealer sales, as well as OEM licensing. Single copies cost $125 (inquire as to available systems). Extremely attractive OEM licensing terms are also available. An optional Configuration Kit contains a detailed Configuration Manual and two disks of source code for system adaptation, including source code for a system monitor/debug ROM and other programs useful for adapting SK★DOS to new systems.

SK★DOS
is available from

STAR-K
SOFTWARE SYSTEMS CORPORATION
P.O. BOX 209 - MT. KISCO, NY 10549 - 914/241 0287
TELEX 6108018774

SOFTWARE FOR 680x AND MSDOS

SUPER SLEUTH DISASSEMBLERS
EACH $99-FLEX $101-OS/9 $100-UNIFLEX
OBJECT-ONLY versions: EACH $50-FLEX,OS/9,COCO
interactively generate source on disk with labels, include xref, binary editing
specify 6800,1,2,3,5,6,9/6502 version or Z80/8080,5 version
OS/9 version also processes FLEX format object file under OS/9
COCO DOS available in 6800,1,2,3,5,6,9/6502 version (not Z80/8080,5) only
NEW: 68010 disassembler $100-FLEX,OS/9,UNIFLEX,OS/9-68K,MSDOS

CROSS-ASSEMBLERS WITH MACRO CAPABILITIES
EACH $50-FLEX,OS/9,UNIFLEX,MSDOS ANY 3 $100 ALL $200
specify for 180x,6502,6801,6804,6805,6809,Z8,Z80,8048,8051,8085,68000
modular cross-assemblers in C, with load/unload utilities NOW: OS/9-68K
8-bit (not 68000) sources for additional $50 each, $100 for 3, $300 for all

DEBUGGING SIMULATORS FOR POPULAR 8-BIT MICROPROCESSORS
EACH $75-FLEX $100-OS/9 $80-UNIFLEX
OBJECT-ONLY versions: EACH $30-COCO FLEX,COCO OS/9
interactively simulate processors, include disassembly formatting, binary editing
specify for 6800/1, (1)6805, 6502, 6809 OS/9, Z80 FLEX

ASSEMBLER CODE TRANSLATORS FOR 6502, 6800/1, 6809
6502 to 6809 $75-FLEX $85-OS/9 $80-UNIFLEX
6800/1 to 6809 & 6809 to position-ind $50-FLEX $75-OS/9 $60-UNIFLEX

FULL-SCREEN XBASIC PROGRAMS with cursor control
AVAILABLE FOR FLEX, UNIFLEX, AND MSDOS

DISPLAY GENERATOR/DOCUMENTOR	$50 w/source, $25 without
MAILING LIST SYSTEM	$100 w/source, $50 without
INVENTORY WITH MRP	$100 w/source, $50 without
TABULA RASA SPREADSHEET	$100 w/source, $50 without

DISK AND XBASIC UTILITY PROGRAM LIBRARY
$50-FLEX $30-UNIFLEX/MSDOS
edit disk sectors, sort directory, maintain master catalog, do disk sorts, resequence some or all of BASIC program, real BASIC program, etc.
non-FLEX versions include sort and resequencer only

SMODEM TELECOMMUNICATIONS PROGRAM
$100-FLEX,OS/9,UNIFLEX,MS-DOS,OS/9-68K,UNIX
OBJECT-ONLY versions: EACH $50
menu-driven with terminal mode, file transfer, MODEM7, XON-XOFF, etc.
for COCO and non-COCO, drives internal COCO modem port up to 2400 Baud

DISKETTES & SERVICES

5.25" DISKETTES
EACH 10-PACK $12.50-SSSD/SSDD/DSDD
American-made, guaranteed 100% quality, with Tyvek jackets, hub rings, and labels

ADDITIONAL SERVICES FOR THE COMPUTING COMMUNITY

CUSTOMIZED PROGRAMMING
we will customize any of the programs described in this advertisement or in our brochure for specialized customer use or to cover new processors; the charge for such customization depends upon the marketability of the modifications

CONTRACT PROGRAMMING
we will create new programs or modify existing programs on a contract basis, a service we have provided for over twenty years. the computers on which we have performed contract programming include most popular models of mainframes, including IBM, Burroughs, Univac, Honeywell, most popular models of minicomputers, including DEC, IBM, DG, HP, AT&T, and most popular brands of microcomputers, including 6800/1, 6809, Z80, 6502, 68000, using most appropriate languages and operating systems, on systems ranging in size from large telecommunications to single board controllers; the charge for contract programming is usually by the hour or by the task.

CONSULTING
we offer a wide range of business and technical consulting services, including seminars, advice, training, and design, on any topic related to computers; the charge for consulting is normally based upon time, travel, and expenses.

Computer Systems Consultants, Inc.
1454 Latta Lane, Conyers, GA 30207
Telephone 404-483-4570 or 1717

We take orders at any time, but plan long discussions after 5, if possible.

Contact us about catalog, dealer, discounts, and services.
Most programs in source: give computer, OS, disk size.
25% off multiple purchases of same program on one order.
VISA and MASTER CARD accepted; US funds only, please.
Add GA sales tax (if in GA) and 5% shipping.
(UNIFLEX tm Technical Systems Consultants; OS/9 Microware; COCO Tandy;MSDOS Microsoft)

Now! *"TOPS The OFFICE PRINT Shop™"* Makes *professionals* of us all. For *less* than just affordable!

DeskTop Publishing

NOTE: The following includes a 3 day, *hands-on*, instructional session for the entire *"The OFFICE PRINT Shop™"* system. A full 6 months, *no extra charge*, telephone or in-house (our offices, normal business hours) follow-up advisory service. We feature **full** Apple™ service and also national service by Honeywell. This includes *next day*, on site service. Over 95% of all service requests are completed on the initial call. ***We understand the importance of fast service!***

100 **System 100** *Very Affordable Save up to 90% on your printing*

The *TOPS* System 100
consists of the following items:

- A special Apple Macintosh Plus™ 1 Megabyte computer, including a double sided, double density 800,000+ character disk drive.
- An Apple LaserWriter™ *typeset quality* printer.

Page make-up software
galley typesetting software
3 day - on site - instructions
6 months instructional support & much more!

Option:
We also offer software to drive *most all* the large commercial typesetters. You can save a bundle by doing your own typesetting and proofing, and then down-loading to the commercial system.

Leasing with payout available for all systems.

Also available with 20 million character storage 'Hard Disk'

Data-Comp Division

CPI — A Decade of Quality Service' — *Systems World-Wide*

Computer Publishing, Inc. 5900 Cassandra Smith Road
Telephone 615 842-4601 - Telex 510 600-6630 Hixson, Tn 37343

This document *composed, typeset and printed* with a TOPS System 100

TOPS The OFFICE PRINT Shop is a trademark of Computer Publishing, Inc.
Apple Macintosh Plus is a trademark of Apple Computer Company, Inc.
LaserWriter is a trademark of Apple Computer Company, Inc.

OS-9 UniFLEX
MUSTANG-020, 68020, 68881 AND MORE
HANDS-ON EXPERIENCE

The DATA-Comp Division of Computer Publishing Corporation announces their new and innovative HANDS-ON 68020 computer familiarization two day event. A chance to TRY BEFORE YOU BUY!

For two full days (Monday through Friday - excluding legal holidays) each participant will be furnished the exclusive use of a 68020 computer (MUSTANG-020). Each system will have available native C compilers, BASIC, assembler and other high level languages. Each system will be equipped with the Motorola MC 68881 math co-processor, where applicable.

Each demonstration room will contain not more than two work stations. Each system will be equipped with floppy disk, 20 megabyte winchester technology hard disk, and 2 megabyte of RAM. RAM is partitioned as 690K bytes of RAM disk and 1.2 megabyte of user RAM space.

Participants are encouraged to bring along any source level projects, for evaluation, in C, BASIC or assembler. Call for availibility of other HHLs.

Although this is not a training seminar, Data-Comp personnel are available for assistance and consultation. This event is scheduled for hands-on evaluations of the 68020 CPU, 68881 math co-processor and MUSTANG-020 system, operating in a functional environment.

Transportation to and from the airport and hotel/motel will be provided. Lunch provided both days. Chattanooga airport is serviced by American, Delta, Republic and other airlines.

COST

One person - $375.00

Two persons - $595.00

* Motel single $22.00, double $26.00
Includes satellite TV - convenient to food and shopping

DATA-COMP

A Division of
Computer Publishing, Inc.

5900 Cassandra Smith Road
Hixson, Tn 37343
Telephone 615 842-4600
Telex 510 600-6630

Systems available for both OS-9 and UniFLEX. Reservation should be made 15 days in advance. Attendee should initially indicate OS-9, UniFLEX or both. Special facilities available on request. Please write or call for additional information.

NOTE: Both OS-9 and UniFLEX are Unix type operating systems. Each as been enhanced in some aspect or another. Prospective attendees should have some working knowledge or expetience with one of these operating systems, to gain full benefit of the session. However, a newcomer will find that it is a simple matter to be fairly proficient in using these systems in the allocated time. Special system instruction available on request. Call or write.

* Hotel/Motel cost are separate cost, not included in the basic cost shown.

THE 6800-6809 BOOKS

..HEAR YE.....HEAR

OS-9™ User Notes

By: Peter Dibble

The publishers of 68' Micro Journal are proud to make available the publication of Peter Dibbles
OS9 USER NOTES

Information for the BEGINNER to the PRO, Regular or CoCo OS9

Using OS9
HELP, HINTS, PROBLEMS, REVIEWS, SUGGESTIONS, COMPLAINTS, OS9 STANDARDS, Generating a New Bootstrap, Building a new System Disk, OS9 Users Group, etc.

Program interfacing to OS9
DEVICE DESCRIPTORS, DIRECTORIES, "FORKS", PROTECTION, "SUSPEND STATE", "PIPES", "INPUT/OUTPUT SYSTEM", etc.

Programming Languages
Assembly Language Programs and Interfacing; Basic09, C, Pascal, and Cobol reviews, programs, and uses; etc.

Disks Include
No typing all the Source Listings in. Source Code and, where applicable, assembled or compiled Operating Programs. The Source and the Discussions in the Columns can be used "as is", or as a "StartinR Point" for developing your OWN more powerful Programs. Programs sometimes use multiple Languages such as a short Assembly Language Routine for reading a Directory, which is then "piped" to a Basic09 Routine for output formatting, etc.

BOOK $9.95

Typeset -- w/ Source Listings
(3-Hole Punched; 8 x 11)
Deluxe Binder - - - - - - - - - - $5.50

All Source Listings on Disk

1-8" SS, SD Disk - - - - $14.95
2-5" SS, DD Disks - - - $24.95

FLEX™ USER NOTES

By: Ronald Anderson

The publishers of 68 MICRO JOURNAL are proud to make available the publication of Ron Anderson's **FLEX USER NOTES**, in book form. This popular monthly column has been a regular feature in 68' MICRO JOURNAL SINCE 1979. It has earned the respect of thousands of 68 MICRO JOURNAL readers over the years. In fact, Ron's column has been described as the 'Bible' for 68XX users, by some of the world's leading microprocessor professionals. The most needed and popular 68XX book available. Over the years Ron's column has been one of the most popular in 68 MICRO JOURNAL. And of course 68 MICRO JOURNAL is the most popular 68XX magazine published.

Listed below are a few of the **TEXT** files included in the book and on diskette.

All TEXT files in the book are on the disks.

LOGO.C1	File load program to offset memory — ASM PIC
MEMOVEC1	Memory move program — ASM PIC
DUMPC1	Printer dump program — uses LOGO — ASM PIC
SUBTEST.C1	Simulation of 6800 code to 6809, show differences — ASM
TERMEM.C2	Modem input to disk (or other port input to disk) — ASM
M.C2	Output a file to modem (or another port) — ASM
PRINT.C3	Parallel (enhanced) printer driver — ASM
MODEM.C2	TTL output to CRT and modem (or other port) — ASM
SCIPKG.C1	Scientific math routines — PASCAL
U.C4	Mini-monitor, disk resident, many useful functions — ASM
PRINT.C4	Parallel printer driver, without PFLAG — ASM
SET.C5	Set printer modes — ASM
SETBAS1.C5	Set printer modes — A-BASIC

NOTE: .C1,.C2, etc.=Chapter 1, Chapter 2, etc.

**Over 30 TEXT files included in ASM (assembler)-PASCAL-PIC (position independent code) TSC BASIC-C, etc.

Book only: **$7.95** + $2.50 S/H

With disk: 5" **$20.90** + $2.50 S/H

With disk: 8" **$22.90** + $2.50 S/H

Shipping & Handling $3.50 per Book, $2.50 per Disk set
Foreign Orders Add $4.50 Surface Mail
or $7.00 Air Mail
If paying by check - Please allow 4-6 weeks delivery
* All Currency in U.S. Dollars

Continually Updated In 68 Micro Journal Monthly

VISA

Computer Publishing Inc.
5900 Cassandra Smith Rd.
Hixson, TN 37343

(615) 842-4601
Telex 5106006630

"FLEX is a trademark of Technical Systems Consultants
"OS9 is a trademark of Microware and Motorola
"68' Micro Journal is a trademark of Computer Publishing Inc.

Hard Disk Subsystem for SS-50 Computers

THIS PROVEN SUBSYSTEM ADDS HARD DISK SPEED AND STORAGE CAPACITY TO YOUR COMPUTER YET REQUIRES ONLY ONE SS-50 SLOT. SOFTWARE (WITH SOURCE) IS INCLUDED FOR YOUR CHOICE OF FLEX9* OR SK*DOS*, OS-9 LEVEL I OR LEVEL II, OR OS-9 68K OPERATING SYSTEMS. THE SOFTWARE HONORS ALL OPERATING SYSTEM CONVENTIONS. THE SOFTWARE IS DESIGNED FOR THE XEBEX S1410 CONTROLLER INTERFACING TO ANY HARD DISK DRIVE THAT CONFORMS TO THE ST506 STANDARD. FOUR SUBSYSTEMS ARE AVAILABLE:

1) 27 MB (FORMATTED) CONTROL DATA CORPORATION WREN HARD DISK, XEBEC S1410A CONTROLLER, SS-30 INTERFACE CARD, ALL CABLES, AND SOFTWARE FOR $2850;
2) 7.3 MB (FORMATTED) TANDON TM-603 HARD DISK, REST SAME AS ABOVE FOR $895;
3) NO HARD DISK, REST SAME AS ABOVE FOR $600; AND
4) SS-30 INTERFACE CARD AND SOFTWARE FOR $200.

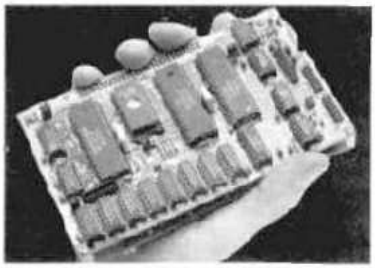
ALL PRICES INCLUDE SHIPPING. WE ACCEPT VISA AND MASTERCARD WITHOUT ADDING A SURCHARGE. TEXAS RESIDENTS MUST ADD SALES TAX. THE SUBSYSTEM MAY BE MOUNTED WITHIN YOUR COMPUTER CHASSIS OR IN A SEPARATE ENCLOSURE WITH POWER SUPPLY. PLEASE WRITE OR PHONE (INCLUDE YOUR DAY AND EVENING PHONE NUMBERS) FOR MORE INFORMATION. WE WILL RETURN NORTH AMERICA CALLS SO THAT ANY DETAILED ANSWERS WILL BE AT OUR EXPENSE.

WELLWRITTEN ENTERPRISES

P.O. Box 9802 - 845
AUSTIN, TEXAS 78766

VISA *** (512) 244-6530 *** MasterCard

FLEX IS A TRADEMARK OF TECHNICAL SYSTEMS CONSULTANTS, INC.
SK*DOS IS A TRADEMARK OF STAR-KITS
OS-9 IS A TRADEMARK OF MICRONAME AND MOTOROLA



512K RAM Expansion

Compact Flexible 6809 Computer

The new ST-2900 system — a complete 64K small business or hobbyist computer is only one of its many possible configurations. Among its features are:

- Small enough to hold in your hand! (Eurocard size: 3.9" x 6.3")
- Three board "system" for greater versatility than single board computers.
- CPU Board — powerful 6809E processor, 16K or 64K RAM, 1K-32K EPROM, 2 RS232 serial ports with software programmable baud rates, 16 bit counter/timer. Run the CPU board all by itself, or plug your own custom board or our FDC board and/or RAM-512 board into the expansion connector.
- FDC Board — double-sided/double-density floppy disk controller with adjustment free digital data separator and write precompensation. 2 8-bit parallel ports, 2 16-bit counter/timers, prototyping area.
- **RAM-512 Board** — 524,288 bytes of RAM on a 4.15" x 6.3" board! Low power. Includes RAM-Disk software for FLEX/STAR-DOS or OS-9.
- **FLEX, STAR-DOS,** and **OS-9** supported — **software selectable.**
- OS-9 Conversion Package lets you use the low cost Radio Shack CoCo version of OS-9 on our ST-2900 system. Save $131 off the suggested list price of OS-9! **No programming is involved.** Supports CoCo OS-9, standard OS-9, and MIZAR OS-9/68K disk formats. Compatible with PC-XFER to let you **read/write/format MS-DOS disks!**

CPU bare board/plus EPROM	$45	OS-9 Conversion Package	$49
FDC bare board	$38	FLEX Conversion Package	$29
RAM-512 board A&T **(w/o RAM)**	**$299**	CPU + FDC + OS-9 Conversion	$119
CPU + FDC board set assembled and tested			$329

- Add $5 shipping/handling ($10 overseas). These prices are in U.S. funds. Canadian orders: call or write for prices. Terms: check, money order, VISA.

(trademarks FLEX – Technical Systems Consultants; OS-9 – MicroWare & Motorola; MS-DOS – Microsoft)

Source code for ST-2900 OS-9 device drivers now available $99

SARDIS TECHNOLOGIES

Call or write for free catalog and complete price list
(604) 255-4485

2261 E. 11th Ave. Vancouver, B.C., Canada V5N 1Z7

LLOYD I/O™ INC.

Lloyd I/O is a computer engineering corporation providing software and hardware products and consulting services.

19535 NE GLISAN • PORTLAND, OR 97230 (USA)
PHONE: (503) 666-1097 • TELEX: 910 380 5448 LLOYD I O

Computer Engineers

K-BASIC™ *IS HERE*

K-BASIC is a TSC XBASIC (XPC) compatible COMPILER
for OS9 & FLEX... price $199

Here at last is a compiler for BASIC that will compile all your XBASIC programs. K-BASIC compiles TSC's XBASIC and XPC programs to machine code. K-BASIC is ready now to save you money and time by teaching your computer to:

• Think Faster • Conserve Memory • Be Friendlier

Call (503) 666-1097 for our CATALOG.
We have many programs for serious software developers!

DO™

Micro BASIC for OS9... $149

A structured mirco BASIC for general system control featuring: Parameter passing, 10 string variables, 26 numeric variables, subroutines, nested loops, interactive I/O, sequential files, and time variables (for applications executing in the background required to execute procedures such as disk or file backups.) Includes the SEARCH and RESCUE UTILITIES™. (For OS9 ONLY.)

SEARCH and RESCUE UTILITIES™

for OS9... $35

A super directory search utility. Output may be piped to the included utilities to perform file: COPIES, DELETES, MOVES, LISTING (pagination), and FILTERING. Some filtering utility programs are included: of interest is the FILE DATE CHECKING utilities YOUNGER and DRAFT (Level 2). (For OS9 Level 1 and 2.)

PATCH™

Modem Communications for OS9... $39

PATCH is a modem communications program for OS9 featuring: KEY MACROS, ASCII TEXT AND BINARY FILE UP/DOWN LOADING, PRINTER COPY, and HELP MENUS. We use it several times each day with our TELEX service. PATCH is convenient and easy to use. Key macros may be pre-stored and loaded at any time.

CRASMB™

CROSS ASSEMBLER PACKAGE
for OS9 & FLEX... all for $399
Motorola CPU's... $150
Intel CPU's... $150, Others... $150

CRASMB is the highly acclaimed cross-assembler package for OS9 and FLEX systems. It turns your [illegible] into a development station for [illegible] get CPUs:

6800 6801 6805 6809 6811 6802
6803 8048 8051 8085 Z80
(6800 [illegible] ...$249)

NOW FOR OS9/68K $432

CRASMB features [illegible], Conditional, Long symbol names, Symbol cross reference table, [illegible] code in 4 formats (OS9, FLEX, S-1-S9, INTEL HEX).

VISA, MC, COD, CHECKS, ACCEPTED

USA:	LLOYD I/O (503 666 1097),	S.E. MEDIA (800 338 6800)
England:	Vivaway (0582 423425),	Windrush (0692 405169)
Germany:	Zacher Computer (65 25 299),	Kell Software (06203 6741)
Australia:	Parts Radio Electronics (344 9111)	
Japan:	Microboards (0474) 22-1741	Seikou (03) 832-6000
Switzerland:	Elsoft AG (056 86 27 24)	
Sweden:	Micromaster Scandinavian AG (018 - 138595)	

K-BASIC, DO, SEARCH and RESCUE UTILITIES, PATCH, CRASMB, and CRASMB 16.32 are trademarks of LLOYD I/O
OS9 is a ™ of Microware, FLEX is a ™ of TSC

SOFTWARE FOR THE HARDCORE

** FORTH PROGRAMMING TOOLS from the 68XX&X **
** FORTH specialists — get the best!! **

NOW AVAILABLE — A variety of rom and disk FORTH systems to run on and/or do TARGET COMPILATION for

6800, 6301/6801, 6809, 68000, 8080, Z80

Write or call for information on a special system to fit your requirement.

Standard systems available for these hardware—

EPSON HX-20 rom system and target compiler
6809 rom systems for SS-50, EXORCISER, STD, ETC.
COLOR COMPUTER
6800/6809 FLEX or EXORCISER disk systems.
68000 rom based systems
68000 CP/M-68K disk systems, MODEL II/12/16

tFORTH is a refined version of FORTH Interest Group standard FORTH, faster than FIG-FORTH. FORTH is both a compiler and an interpreter. It executes orders of magnitudes faster than interpretive BASIC. MORE IMPORTANT, CODE DEVELOPMENT AND TESTING is much, much faster than complied languages such as PASCAL and C. If Software DEVELOPMENT COSTS are an important concern for you, you need FORTH!

firmFORTH™ is for the programmer who needs to squeeze the most into roms. It is a professional programmer's tool for compact rommable code for controller applications.

* tFORTH and firmFORTH are trademarks of Talbot Microsystems
* FLEX is a trademark of Technical Systems Consultants, Inc
* CP/M-68K is trademark of Digital Research, Inc

tFORTH™ from TALBOT MICROSYSTEMS NEW SYSTEMS FOR 6301/6801, 6809, and 68000

---> tFORTH SYSTEMS <---

For all FLEX systems: GIMIX, SWTP, SSB, or EXORciser Specify 5 or 8 inch diskette, hardware type, and 6800 or 6809.

** tFORTH — extended fig FORTH (1 disk) $100 ($15)
with fig line editor.
** tFORTH + — more! (3 5" or 2 8" disks) $250 ($25)
adds screen editor, assembler, extended data types, utilities, games, and debugging aids.
** TRS-80 COLORFORTH — available from The Micro Works
** firm FORTH — 6809 only. $350 ($10)
For target compilations to rommable code.
Automatically deletes unused code. Includes HOST system source and target nucleus source. No royalty on targets. Requires but does not include tFORTH +.
** FORTH PROGRAMMING AIDS — elaborate decompiler $150

** tFORTH for HX-20, in 16K roms for expansion unit or replace BASIC $170

** tFORTH/68K for CP/M-68K 8" disk system $290
Makes Model 16 a super software development system.

** Nautilus Systems Cross Compiler
— Requires: tFORTH + HOST + at least one TARGET:
— HOST system code (6809 or 68000) $200
— TARGET source code: 6800-$200, 6301/6801—$200
same plus HX-20 extensions— $300
6809—$300, 8080/Z80—$200, 68000—$350

Manuals available separately — price in ( ).
Add $6 system for shipping, $15 for foreign air.

TALBOT MICROSYSTEMS 1927 Curtis Ave., Redondo Beach, CA 90278 (213) 376 9941

Installed Systems World-Wide
OVER 10 YEARS OF DEDICATED QUALITY!

CPI A Division of Computer Publishing, Inc.
5900 Cassandra Smith Road
Hixson, Tn 37343
Telephone 615 842-4600
Telex 510 600-6630

DATA-COMP SPECIAL
Heavy Duty Power Supplies

For A limited time we are offering our HEAVY DUTY SWITCHING POWER SUPPLY. These are BRAND NEW units and will not last long. Also note that these prices are less than 1/4 the normal price for these high quality unit.

Make: Boschert

Size: 10.5 x 5 x 2.5 inches - including heavy mounting brackets and hardware.

Rating: in 110/220 volts ac (strap change) Out: 130 watts

Outputs: +5v - 10 amps
+12v - 4.0 amps
+12v - 2.0 amps
-12v - 0.5 amps

Mating Connector: Terminal strip
Load Reaction: Automatic short circuit recovery

Each
SPECIAL: $59.95
2 or more 49.95

Add: $7.50 each S/H

Make: Boschert

Size: 10.75 x 6.2 x 2.25 inches

Rating: 110/220 ac (strap change) Out: 81 watts

Outputs: +5v - 8.0 amps
+12v - 2.4 amps
+12v - 2.4 amps
+12v - 2.1 amps
-12v - 0.4 amps

Mating Connectors: Molex
Load Reaction: Automatic short circuit recovery

Each
SPECIAL: $49.95
2 OR MORE 39.95

Add: $7.50 S/H each

PLμS-68k (PL/9 for the 68000) running under OS/9-68000

- Built-in screen editor
- Built-in source-level debugger
- Byte, Integer, Long and Real variables
- Direct source to object compilation
- Generates fast, efficient object code
- Compiles over 2000 lines/min
- 99% upward compatible with PL/9
- Comprehensive instruction manual
- No license fees to pay

PLμS-68k is a complete re-write of PL/9, for the 68000-series μP. It is a structured high-level language, designed for programming stand-alone applications as well as those in which an operating system is present. It is easy to learn, very quick to use (the compiler is resident with the editor and produces executable object code in one pass) and generates efficient, position-independant code upon which no royalties are payable.

The built-in tracer/debugger helps you debug your programs symbolically and at source level, so you need know next to nothing about the assembly-language of the 68000 in order to develop programs.

PLμS is here now, it carries the Windrush commitment to support, and it's cheap. Do you need a better reason to try it?

PLμS-68k for OS/9-68000.....$299.00

PLμS-68k is also available for FLEX™ systems, supplied with a 68008 second processor, case and power supply. (This reduced price no longer includes a free upgrade to the OS/9 version).....$499.00

For further information, phone or write:

Worstead Laboratories
North Walsham
Norfolk NR28 9SA
England

Tel (44) 692 404086
Telex 975548 WMICRO G

'68' MICRO JOURNAL

OK, PLEASE ENTER MY SUBSCRIPTION

Bill My: Mastercard ☐ VISA ☐

Card #_____ Exp. Date_____

For 1 Year ___ 2 Years ___ 3 Years ___

Enclosed: $_____

Name _____

Street _____

City _____ State ____ Zip _____

My Computer Is: _____

Subscription Rates

U.S.A.: 1 Year $24.50, 2 Years $42.50, 3 Years $64.50

***Foreign Surface: Add $12.00 per Year to USA Price.**

***Foreign Airmail: Add $48.00 per Year to USA Price.**

***Canada & Mexico: Add $9.50 per Year to USA Price.**

***U.S. Currency Cash or Check Drawn on a USA Bank !**

68 Micro Journal
5900 Cassandra Smith Rd.
POB 849
Hixson, TN 37343

☎

Telephone 615 842-4600

Telex 510 600-6630

MasterCard VISA

CSG IMS

CSG IMS is a general purpose information management system designed to make the development of file-intensive applications as quick and easy as possible. IMS is a full featured database manager with the added benefit of a structured, general purpose application language. Some popular applications are: accounting, inventory, data acquisition, cataloging, membership and mailing lists.

SYSTEM FEATURES

- CSG IMS uses B+Tree index structures for fast database access and reliability. Record, index and file sizes are virtually unrestricted.
- Supported data types are: text, BCD floating point (14 digits), short and long integers, and date.
- Menu driven executive program for ease of operation.
- User definable screen forms and reports are supported.
- The interactive environment provides access to databases and most language features allowing quick ad hoc queries.
- CSG IMS includes a recursive, compiled language supporting program modules with full parameter passing.
- The CSG IMS run-time interpreter is available separately for user developed and distributed applications.
- Comprehensive 320 page manual with tutorial section.

CSG IMS for OS9/6809 LII and OS9/68000: $495.00
Run time interpreter for CSG IMS: $100.00
CSG IMS manual only: $20.00
PRICES IN US DOLLARS
ADD $5.00 S&H FOR CONTINENTAL USA. FOREIGN ORDERS ADD $10.00 S&H

Clearbrook Software Group

To order CSG IMS or to receive further information write:
CLEARBROOK SOFTWARE GROUP
P.O. Box 8000-499
Sumas, WA 98295-8000
or phone:
(604) 853-9118

Send for a free catalog describing all of our OS9 products.
We welcome dealer inquiries.

OS9 is a registered trademark of Microware and Motorola.

OS-9™ SOFTWARE

L1 UTILITY PAK—Contains all programs formerly in Filter kits 1 & 2, and Hacker's kit 1 plus several additional programs. Complete "wild card" file operations, copies, moves, sorts, del, MACGEN shell command language compiler, Disassembler, Disk sector edit utility, new and improved editions, approx. 40 programs, increases your productivity. Most programs applicable for both level I & II 6809 OS-9. **$49.95 ($51.95)**

Call or send Self Addressed Stamped Envelope for catalog of software for color Computer OS-9 and other OS-9 systems.

BOLD prices are CoCo OS-9 format disk, other formats (In parenthesis) specify format. All orders prepaid or COD, VISA and MasterCard accepted. Add $1.50 S&H on prepaid, COD actual charges added.

SS-50C
MEMORY LIQUIDATION SALE!
(While Supply Lasts)

1 MEGABYTE RAM BOARD

Full megabyte of ram with disable options to suit any SS-50 6809 system. High reliability, can replace static ram for fraction of the cost. $399 for 2 Mhz or $439 for 2.25 Mhz board assembled, tested and fully populated.

2 MEGABYTE RAM DISK BOARD

RD2 2 megabytes dedicated ram disk board for SS-50 systems. Four layer circuit board socketed for 2 **Megabytes!** Special sale price of **$399.00** includes only 256k of ram installed (you add the rest), includes OS-9 level I and II drivers for Ram disk, (note: you can re-boot your system without losing ram-disk contents). (Add $6 shipping and Insurance.)

Please call for answers to your technical questions concerning these products.

D.P. Johnson, 7655 S.W. Cedarcrest St.
Portland, OR 97223, (503) 244-8152
(For best service call between 9-11 am Pacific time.)

OS-9 is a trademark of Microwave and Motorola Inc.
MS-OOS is a trademark of Microsoft Inc.

COMPILER EVALUATION SERVICES

BY: Ron Anderton

The S.E. MEDIA Division of Computer Publishing Inc. is offering the following SUBSCRIBER SERVICE:

COMPILER COMPARISON AND EVALUATION REPORT

Due to the constant and rapid updating and enhancement of numerous compilers, and the different utility, appeal, speed, level of communication, memory usage, etc., of different compilers, the following services are now being offered with periodic updates.

This service, with updates, will allow you who are wary or confused by the various claims of compiler vendors, an opportunity to review comparisons, comments, benchmarks, etc., concerning the many different compilers on the market, for the 6809 microcomputer. Thus the savings could far offset the small cost of this service.

Many have purchased compilers and then discovered that the particular compiler purchased either is not the most efficient for their purposes of does not contain features necessary for their application. Thus the added expense of purchasing additional compiler(s) or not being able to fully utilize the advantages of high level language compilers becomes too expensive.

The following COMPILERS are reviewed initially, more will be reviewed, compared and benchmarked as they become available to the author:

PASCAL 'C' GSPL WHIMSICAL PL/9

Initial Subscription - $39.95
(includes 1 year updates)
Updates for 1 year - $14.50

S.E. MEDIA - C.P.I.
5900 Cassandra Smith Rd.
Hixson, TN 37343
(615) 842-4601

68000 68020 68010
68008 6809 6800

Write or phone for catalog.

AAA Chicago Computer Center

120 Chestnut Lane — Wheeling IL 60090
(312) 459-0450

Technical Consultation available most weekdays from 4 PM to 6 PM CST

Stop!

Get a 25 MegaByte Hard Disk practically FREE - only 1¢

Be Sure to Consider the SPECIAL MUSTANG 1¢ Sale on page 5

When it's over, IT'S OVER!

We don't know how long this very, very low price can be maintained, don't miss it!

Data-Comp Div. - CPI

6809<>68XXX UniFLEX

X-TALK

A C-MODEM/Hardware Hookup

Exclusive for the MUSTANG-020 running UniFLEX, is a new transfer program and cable set from DATA-COMP (CPI). X-TALK consist of 2 disks and a special cable, this hook-up enables a 6809 SWTPC UniFLEX computer to port UniFLEX files directly to a 68XXX UniFLEX system.

This is the only currently available method to transfer files, text or otherwise, from a 6809 UniFLEX system to a 68000 UniFLEX system, that we have seen. A must if you want to recompile or cross assemble your old (and valuable) source files to run on a 68000 UniFLEX system. GIMIX users can directly transfer files between a 6809 GIMIX system and our MUSTANG-020 68020 system, or GIMIX 68020 system. All SWTPC users must use some sort of method other than direct disk transfer. The 6809 SWTPC UniFLEX disk format is not readable by most other 68000 type systems.

The cable is specially prepared with internal connections to match the non-standard SWTPC SO/9 DB25 connectors. A special SWPTC+ cable and software is also available, at the same price. Orders must specify which type SWTPC 6809 UniFLEX system they intend to transfer from or to.

The X-TALK software is furnished on two disks. One 8" disk containing the 6809 software and one 5" disk containing the 68XXX software. These programs are also complete MODEM programs and can be used as such, including X-on X-off, and all the other features you would expect from a full modem program.

X-TALK can be purchased with/without the special cables, however, this SPECIAL price is available only to registered MUSTANG-020 owners.

X-TALK, w/cable $ 99.95
X-TALK only 69.95
X-TALK w/source $149.95

DATA-COMP
5900 Cassandra Smith Rd.
Hixson, TN 37343

Telephone 615 842-4601
Telex 510 600-6630

Note: Registered MUSTANG-020 owners must furnish system serial number in order to buy at these special low prices.

68 MICRO JOURNAL
Reader Service Disks

Disk- 1	Filesort, Miniscat, Mmicopy,Mmifms, **Lifetime, **Poetry, **Foodlist, **Diet.
Disk- 2	Diskedit w/ inst.& fixes, Prime, *Prmod, **Snoopy, **Football, **Hexpawn,**Lifetime.
Disk- 3	Chug09, Sec1, Sec2, Find, Table2, Intext, Disk-exp, *Disksave.
Disk- 4	Mailing Program, *Finddat, *Change, *Testdisk.
Disk- 5	*DISKFIX 1, *DISKFIX 2, **LETTER, **LOVESIGN, **BLACKJAK, **BOWLING.
Disk- 6	**Purchase Order, Index (Disk file indx).
Disk- 7	Linking Loader, Rload, Hardness.
Disk- 8	Crest, Lanpher (May 82).
Disk- 9	Datecopy, Diskfix9 (Aug 82).
Disk-10	Home Accounting (July 82).
Disk-11	Dissembler (June 84).
Disk-12	Modem68 (May 84).
Disk-13	*fmtmf68, Tesmf68, *Cleanup, *Dskalign, Help, Date.Txt.
Disk-14	*Init, *Test, *Terminal, *Find, *Diskedit, Init.Lib.
Disk-15	Modem9 + Updates (Dec. 84 Gilchrist) to Modem9 (April 84 Commo).
Disk-16	Copy.Txt, Copy.Doc, Cat.Txt, Cat.Doc.
Disk-17	Match Utility, RATBAS, A Basic Preprocessor.
Disk-18	Parse.Mod, Size.Cmd (Sept. 85 Armstrong), CMDCODE, CMD.Txt (Sept. 85 Spray).
Disk-19	Clock, Date, Copy, Cat, PDEL.Asm & Doc., Error.Sys, Do, Log.Asm & Doc.
Disk-20	UNIX Like Tools (July & Sept. 85 Taylor & Gilchrist). Dragon.C, Grep.C, LS.C, FDUMP.C.
Disk-21	Utilities & Games - Date, Life, Mailmerge, Touch, Goblin, Starshot, & 15 more.
Disk-22	Read CPM & Non-FLEX Disks. Fraser May 1984.
Disk-23	ISAM, Indexed Sequential file Accessing Methods, Condon Nov. 1985. Extensible Table Driven. Language Recognition Utility, Anderson March1986.
Disk-24	68' Micro Journal Index of Articles & Bit Bucket Items from 1979 - 1985, John Currens.
Disk-25	KERMIT for FLEX derived from the UNIX ver. Burg Feb. 1986. (2)-5" Disks or (1)-8" Disk.
Disk-26	Compacta UniBoard review, code & diagram, Burlison March '86.
Disk-27	ROTABIT.TXT, SUMSTEST.TXT, CONDATA.TXT, BADMEN.TXT.
Disk-28	CT-82 Emulator, bit mapped.
Disk-29	**Star Trek
Disk-30	Simple Winchester, Dec.'86 Green.
Disk-31	*** Read/Write MS/PC-DOS (SK*DOS)

NOTE:
This is a reader service ONLY! No Warranty is offered or implied, they are as received by 68' Micro Journal, and are for reader convenience ONLY (some MAY include fixes or patches). Also 6800 and 6809 programs are mixed, as each is fairly simple (mostly) to convert to the other. Software is available to cross-assemble all.

* Denotes 6800 - ** Denotes BASIC
*** Denotes 68000 - 6809 no indicator.

Specify 8" disk $19.50
5" disk $16.95

Add: S/H - $3.50
Overseas add: $4.50 surface - $7.00 Air Mail, USA Dollars

68 MICRO JOURNAL
PO Box 849
Hixson, TN 37343
615 842-4600 - Telex 510 600-6630

6809/68008 SINGLE BOARD COMPUTERS

The Peripheral Technology Family of Single Board Computers is a Low-Cost Group Which Ranges From an Entry Level 8-Bit Version to a Powerful 68008-Based Board. A Product is Available to Fit Almost Every User's Requirements.

PT69-5

- 6809 Processor/2MHZ Clock
- 4 RS-232 Serial Ports
- 2 8-Bit Parallel Ports
- 4K-16K EPROM/60K Ram
- Parallel Printer Interface
- DS/DD Controller for 35-80
- Track Drives Ranging From SS/SD-DS/DD
- Winchester Interface Port

PRICE: $349.00

PT69-3

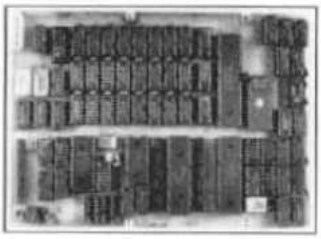
- 6809 1 MHZ Processor
- 2 RS-232 Serial Ports
- 2 8-Bit Parallel Ports
- 4K EPROM/59K User Ram
- DS/DD Controller for 35-80 Track Drives Ranging From SS/SD-DS/DD

PRICE: $269.95
OS9 L1 For
PT69 BOARDS: $200.00
SK*DOS: $ 49.95

PT68K-1

- MC68008 10 MHZ Processor
- 768K RAM/64K EPROM
- 2 RS-232 Serial Ports
- Winchester Interface Port
- Floppy Disk Controller for 2 5+" Drives
- 2 8-Bit Parallel Ports

BOARD: $595.00
WITH OS9: $749.95
WITH SK*DOS: $675.00



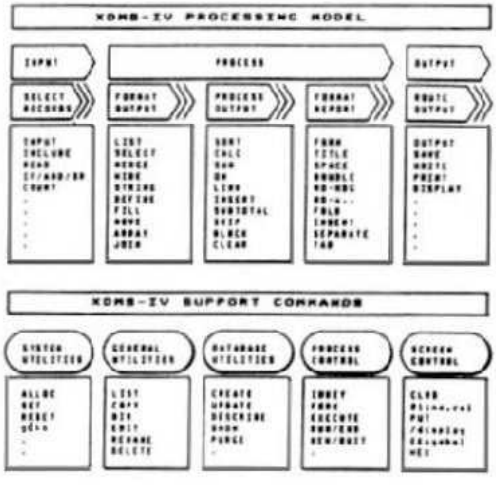
PERIPHERAL TECHNOLOGY
1480 Terrell Mill Road, Suite 870
Marietta, Georgia 30067
(404) 984-0742 Telex # 880584
VISA/MASTERCARD/CHECK/C.O.D.

*OS9 Is A Trademark Of Microware and Motorola

Send For Catalogue For Complete Information On All Products.

XDMS-IV

Data Management System



Save $100.00 - Limited Time
Regular $350.00 - ~~Now Only~~
$249.95

Technical telephone assistance: Tel 914-941-3552 (Evenings)
FLEX™ Technical Systems Consultants, SK*DOS™ STAR-KITS Corp.

FOR 6809 FLEX-SK-DOS(5/8")

Up to 32 groups/fields per record! Up to 12 character filed names! Up to 1024 byte records! User defined screen and print control! Process files! Form files! Conditional execution! Process chaining! Upward/Downward file linking! File joining! Random file virtual paging! Built in utilities! Built in text line editor! Fully session oriented! Enhanced format Boldface, Double width, Italics and Underline supported! Written in compact structured assembler! Integrated for FAST execution!

XDMS-IV Data Management System

XDMS-IV is a brand new approach to data management. It not only permits users to describe, enter and retrieve data, but also to process entire files producing customized reports, screen displays and file output. Processing can consist of any of a set of standard high level functions including record and field selection, sorting and aggregation, lookups in other files, special processing of record subsets, custom report formatting, totaling and subtotaling, and presentation of up to three related files as a "database" on user defined output reports.

POWERFUL COMMANDS!

XDMS-IV combines the functionality of many popular DBMS software systems with a new easy to use command set into a single integrated package. We've included many new features and commands including a set of general file utilities. The processing commands are Input-Process-Output (IPO) oriented which allows almost instant implementation of a process design.

SESSION ORIENTED!

XDMS-IV is session oriented. Enter "XDMS" and you are in instant command of all the features. No more waiting for a command to load in from disk! Many commands are immediate, such as CREATE (file definition), UPDATE (file editor), PURGE and DELETE (utilities). Others are process commands which are used to create a user process which is executed with a RUN command. Either may be entered into a "process" file which is executed by an EXECUTE statement. Processes may execute other processes, or themselves, either conditionally or unconditionally. Menus and screen prompts are easily coded, and entire user applications can be run without ever leaving XDMS-IV!

IT'S EASY TO USE!

XDMS-IV keeps data management simple! Rather than design a complex DBMS which hides the true nature of the data, we kept XDMS-IV file oriented. The user view of data relationships is presented in reports and screen output, while the actual data resides in easy to maintain files. This aspect permits customized presentation and reports without complex redefinition of the database files and structure. XDMS-IV may be used for a wide range of applications from simple record management systems (addresses, inventory ...) to integrated database systems (order entry, accounting...). The possibilities are unlimited.

Visa & Master Card Excepted

Telephone: 615-842-4601 or Telex: 510 600-6630
Or Write: S.E. Media, 5900 Cassandra Smith Rd., Hixson, Tenn. 37343

GMX Micro-20 prices

MICRO 20 (12.5 MHz).....$2565.00
MICRO 20 (16.67 MHz).....$2895.00
8 PORT RS232 BOARD SET (SBC-8S).....$ 498.00
PROTOTYPING BOARD (SBC-WW).....$ 75.00
BACK PANEL PLATE (BPP-PC).....$ 44.00
I/O BUS ADAPTER (SBC-8A).....$ 195.00

QUANTITY DISCOUNTS ARE AVAILABLE ON THE ABOVE ITEMS AS FOLLOWS:4-9,LESS 5%; 10-24,LESS 10%; 25-99,LESS 20%; 100 UP,LESS 30%.

MC68881RC12.....$ 295.00
MC68881RC16.....$ 395.00
SBC ACCESSORY PACKAGE (M20-AP).....$1690.00
For other configurations and options, contact GMX.
MOTOROLA 68020 USERS MANUAL.....$ 18.00
MOTOROLA 68881 USERS MANUAL.....$ 18.00

TO ORDER BY MAIL: SEND CHECK OR MONEY ORDER OR USE YOUR VISA OR MASTER CHARGE. Please allow 3 weeks for personal checks to clear. U.S. orders add $5 handling if order is under $200.00. Foreign orders add $10 handling if order is under $200.00. Foreign orders over $200.00 will be shipped via Emery Air Freight COLLECT, and we will charge no handling. All orders must be prepaid in U.S. funds. Please note that foreign checks have been taking about 8 weeks for collection so we would advise wiring money, or checks drawn on a bank account in the U.S. Our bank is the Continental Illinois National Bank of Chicago, 231 S. LaSalle Street, Chicago, IL 60693, account number 73-32033.

BASIC-09 and OS-9 are trademarks of Microware Systems Corp. and MOTOROLA, Inc. FLEX and UniFLEX are trademarks of Technical Systems Consultants, Inc. GIMIX, GHOST, GMX, CLASSY CHASSIS, are trademarks of GIMIX, Inc.

GMX
1337 WEST 37th PLACE
CHICAGO, ILLINOIS 60609
(312) 927-5510 • TWX 910-221-4055

GMX S-50 BUS prices

68020 SYSTEM

For the user who appreciates the need for a bus structured system using STATIC RAM and powered by a ferro resonant constant voltage transformer, DMA transfers, high speed MMU, we have the UNIFLEX-VM 68020 development system.
The system CPU provides protection to the system and other users from crashes caused by defective user programs.
The system's intelligent serial I/O processor boards significantly reduce system overhead by handling routine I/O functions.
The UniFLEX VM Operating System is a demand-paged, virtual memory operating system written in 68020 Assembler code for compactness and efficiency. It allows up to 4 Megabytes of Virtual Memory per user. All systems include 1MB of static RAM, one 3-port Intelligent Serial I/O board, DMA Controllers, a 5" 80 track floppy drive.

PRICES

#020 UniFLEX VM with 25MB HD.....$10,988.20
#020 UniFLEX VM with 85MB HD.....$12,488.20

YOU CAN EXPAND THESE 020 SYSTEMS WITH:

60MB STREAMER.....$ 2,400.00
REMOVABLE PACK DRIVE.....$ 1,200.00

INTELLIGENT I/OS

#14 3 Port Serial-30 Pin.....$ 498.14
#13 4 Port Serial-50 Pin.....$ 618.13
#12 Parallel-50 Pin.....$ 538.12

CABLE SETS FOR I/OS

95 Cable Sets Specify Card.....$ 24.95
51 Cent. B.P. Cable for #12 & #44....$ 34.51
53 Cent. Cable Set.....$ 36.53

6809 SYSTEM

The number 39 systems include:#05 CPUwBAT; #19 Classy Chassis; 256K Static RAM; a # 43 2 port serial card & cables; #68 DMA Controller; all necessary cables, power regulators, and filler plates;

System # 39 OS-9 GMX II Dual 80 DSDD...$ 2,998.39
" w19MB.....$ 4,698.39
" w72MB.....$ 6,298.39

The Software included in this System: GMXBUG monitor; FLEX; and OS-9 GMXII. You can software select either FLEX or OS-9. Also includes OS-9 Editor, Assembler, Debugger, BASIC-09, RUNB, RMS, DD, and GMX-VDISK for FLEX,

System # 39 UniFLEX w25MB.....$ 4,698.39
" w85MB.....$ 6,298.39

The UniFLEX Operating System is included.

6809 SYSTEMS USING THE GIMIX III CPU & INTELLIGENT I/O PROCESSOR BOARDS

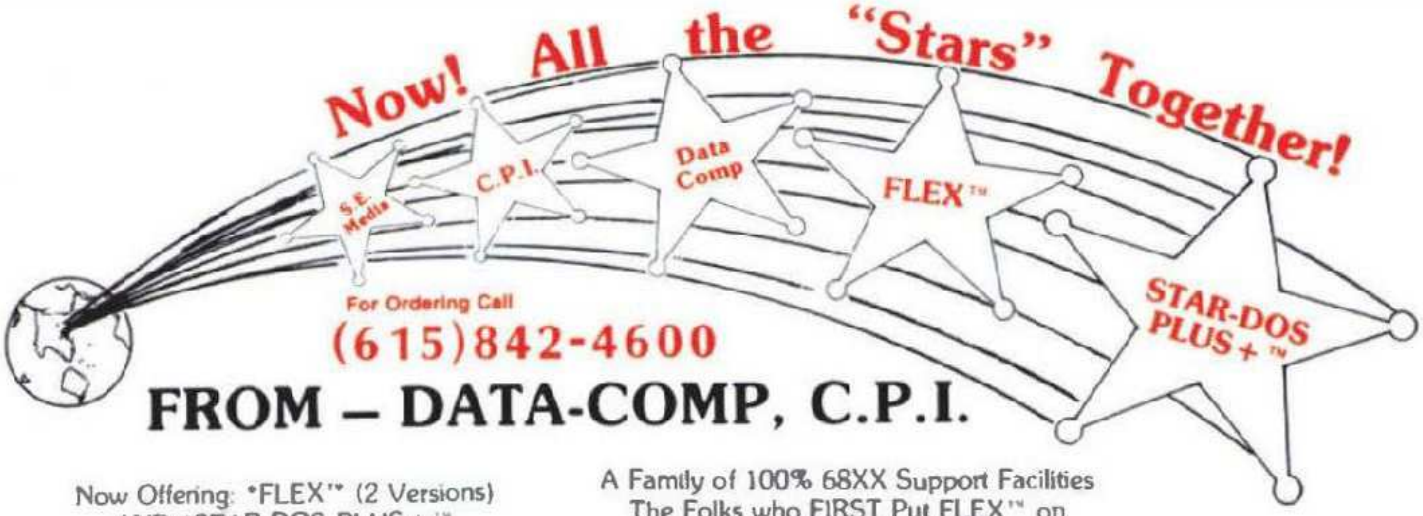
These System include: GMX6809 CPU III; one #11 3 port intelligent serial I/O & Cables; #19 Classy Chassis; 256K Static RAM; #68 DMA controller; all necessary cables, power regulators, and filler plates.

System # 79 OS9 GMX III Dual 80 DSDD...$ 4,498.79
" w25MB.....$ 6,498.79
" w85MB.....$ 7,998.79

The # 79 System Software includes: OS9 GMXIII; OS9 Editor, Assembler, Debugger, BASIC 09, RUNB, RMS, DD, RAMdisk, O-FLEX; GMXBUG; FLEX. The GMX Support ROM and the hardware CRC board are exclusive features iacluded in this system.

System # 89 UniFLEX III w25MB.....$ 6,798.39
w85GB.....$ 8,298.39

The UniFLEX GMX III Operating System is included.



FROM – DATA-COMP, C.P.I.

Now Offering: *FLEX™ (2 Versions) AND *STAR-DOS PLUS+™

A Family of 100% 68XX Support Facilities
The Folks who FIRST Put FLEX™ on The CoCo

FLEX-CoCo Sr. with TSC Editor, TSC Assembler, Complete with Manuals. Reg. $250.00 **Only $79.95**

STAR-DOS PLUS+ $34.95
- Functions Same as FLEX
- Reads - writes FLEX Disks
- Run FLEX Programs
- Just type: Run "STAR-DOS"
- Over 300 utilities & programs to choose from.

FLEX-CoCo Jr. without TSC Editor & Assembler $49.95

PLUS

ALL VERSIONS OF FLEX & STAR-DOS• INCLUDE

TSC Editor Reg $50.00 **NOW $35.00**

TSC Assembler Reg $50.00 **NOW $35.00**

+ Read-Write-Dir RS Disk
+ Run RS Basic from Both
+ More Free Utilities
+ External Terminal Program
+ Test Disk Program
+ Disk Examine & Repair Program
+ Memory Examine Program
+ Many Many More!!!

CoCo Disk Drive Systems

2 THINLINE DOUBLE SIDED DOUBLE DENSITY DISK DRIVES SYSTEM WITH POWER SUPPLY,CABINET,DISK DRIVE CABLE,J&M NEW DISK CONTROLLER JFD-CP WITH J-DOS,RS-DOS OPERATING SYSTEMS. $469.95

* Specify What CONTROLLER You Went J&M, or RADIO SHACK

THINLINE DOUBLE SIDED DOUBLE DENSITY 40 TRACKS $129.95

Verbatim Diskettes

Single Sided Double Density	$ 24.00
Double Sided Double Density	$ 24.00

Controllers

J&M JFD-CP WITH J-DOS	$139.95
WITH J-DOS, RS-DOS	$159.95
RADIO SHACK 1.1	$134.95
RADIO SHACK Disk CONTROLLER 1.1	$134.95

Disk Drive Cables

Cable for One Drive	$ 19.95
Cable for Two Drives	$ 24.95

MISC

64K UPGRADE FOR C,D,E,F, AND COCO 1.1	$ 29.95
RADIO SHACK BASIC 1.2	$ 24.95
RADIO SHACK DISK BASIC 1.1	$ 24.95
DISK DRIVE CABINET FOR A SINGLE DRIVE	$ 49.95
DISK DRIVE CABINET FOR TWO THINLINE DRIVES	$ 69.95

PRINTERS

EPSON LX-80	$289.95
EPSON MX-70	$325.95
EPSON MX-100	$495.95

ACCESSORIES FOR EPSON

8148 2K SERIAL BOARD	$ 89.95
8149 32K EXPAND TO 128K	$169.95
EPSON MX-RX-80 RIBBONS	$ 7.95
EPSON LX-80 RIBBONS	$ 5.95
TRACTOR UNITS FOR LX-80	$ 39.95
CABLES & OTHER INTERFACES CALL FOR PRICING	

DATA-COMP
5900 Cassandra Smith Rd.
Hixson, TN 37343

SHIPPING
USA ADD 2%
FOREIGN ADD 5%
MIN. $2.50

(615)842-4600
For Ordering
Telex 5106006630

Introducing

S - 50 BUS / 68XX

Board and/or Computer
Terminals-CRTs-Printers
Disk Drives-etc.

REPAIRS

This

Not This

NOW AVAILABLE TO ALL S50/68XX USERS

The Data-Comp Division of CPI is proud to announce the availability of their service department facilities to **'ALL'** S50 Bus and 68XX users. Including all brands, SWTPC - GIMIX - SSB - HELIX and others, including the single board computers. *Please note that kit-built components are a special case, and will be handled on an individual basis, if accepted.

1. If you require service, the first thing you need to do is call the number below and describe your problem and *confirm a Data-Comp service & shipping number!* This is very important, Data-Comp will not accept or repair items not displaying this number! Also we cannot advise or help you troubleshoot on the telephone, we can give you a shipping number, but **NO** advice! Sorry!

2. All service shipments must include both a minimum $40.00 estimate/repair charge and pre-paid return shipping charges (should be same amount you pay to ship to Data-Comp).

3. If you desire a telephone estimate after your repair item is received, include an additional $5.00 to cover long distance charges. Otherwise an estimate will be mailed to you, if you requested an estimate. Estimates *must be requested.* Mailed estimates slow down the process considerably. However, if repairs are not desired, after the estimate is given, the $40.00 shall constitute the estimate charge, and the item(s) will be returned unrepaired providing sufficient return shipping charges were included with the item to be serviced. Please note that estimates are given in dollar amounts only.

4. Data-Comp service is the oldest and most experienced general S50/68XX service department in the world. We have over $100,000.00 in parts in stock. We have the most complete set of service documents for the various S50/68XX systems of anyone - **YET, WE DO NOT HAVE EVERYTHING!** But we sure have more than anyone else. We repair about 90% of all items we receive. Call for additional information or shipping instructions.

DATA-COMP
5900 Cassandra Smith Rd.
Hixson, TN 37343

(615)842-4607
Telex 5106006630