

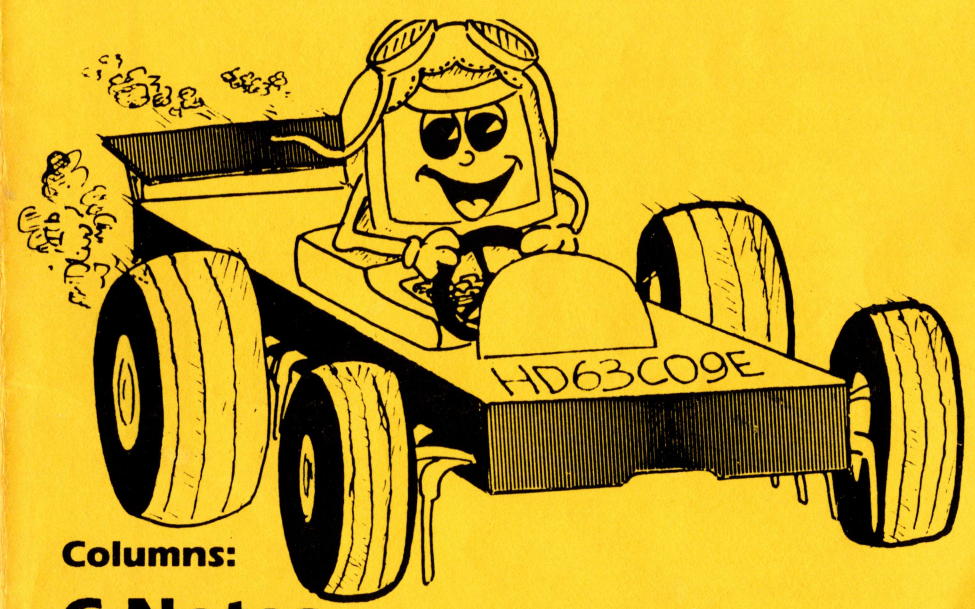
\$2⁰⁰

The OS9 Underground[®]

Magazine Dedicated to OS9 / OSK Users Everywhere!

Volume 1, Issue 1 "Premiere Issue" June 1992

Feature: Test Driving the 6309



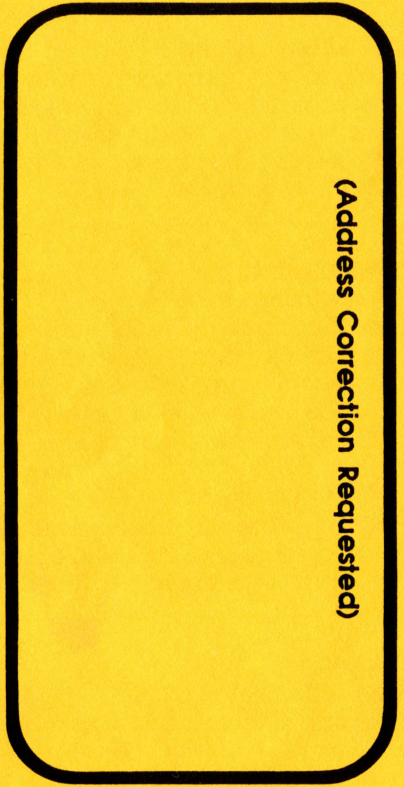
Columns:
C-Notes
BASIC Training
Software Engineering

Programs:
ConvC - Text Utility
Doing the Split - Split.B09

"The OS9 Underground Magazine"

Fat Cat Publications
4650 Cahuenga Blvd. Ste. #7
Toluca Lake, Ca. 91602

(Address Correction Requested)



Coming next month...

Underwater Robots, NV Ram and OS9



SYSTEM IV

The 68000 Computer serving customers world-wide

This high-quality, high performance 68000 computer was designed for and is accepted by industry. Perfect low-cost work-station, development platform or fun machine. Powerful, flexible and expandable inexpensively. Run MS-DOS software with the optional ALT-86 card. Supports up to 4 operating systems.

Prices start at \$999.00 with Professional OS-9

FREE

G-WINDOWS DEMO

for SYSTEM IV and PT68K4/2 owners.

Multi-tasking - processes continue running when windows are made inactive or are hibernating.

Windows may be re-sized, moved, overlaid, etc.

GUI to start processes by selecting an icon or, start processes from your custom menu or from the command line.

Copy and Paste between windows.

Adds command line editing, command history, and file name expansion.

Runs existing OSK software without modification.

Number of windows and processes limited only by your memory.

Includes GIF viewer.

Includes G-VIEW demo.

G-WINDOWS with DESKTOP \$199.00

G-WINDOWS Developer's Pak \$199.00

LIMITED TIME OFFER - both for \$399.00

OS-9/68000 SOFTWARE

QUICK ED - Screen Editor and Text Formatter
VED ENHANCED - Text Editor
SCULPTOR - Development and Run-Time Systems
FLBKBLINT V4.00 - The C Source Code Checker
WINDOWS - C Source Code Windowing Library
IMP - Intelligent Make Program

CALC-9 - Spreadsheet
VPRINT - Print Formatter (for VED)
M6809 - OS-9 6809 Emulator/Interpreter
DISASM_OS9 - OS-9/68K Disassembler
PROFILE - User State Program Profiler
PAN UTILITIES

delmar co

Middletown Plaza - PO Box 78 - Middletown, DE 19709
302-378-2555 FAX 302-378-2556



New Stuff From...



SUB-ETHA SOFTWARE

"In Support of the CoCo"



CheckBook+09 by Joel Mathew Hegberg

"Point 'n click" checking account manager with graphing! Fully mouse/joystick driven. Visualize your account with Circlr, Bar and Line graphs which can be exported for use with other software. Pop-up currency calculator with memory, "cut 'n paste" editing sorting, printer support...plus more! (RS-DOS version reviewed October 1991 Rainbow, page 54.) Req: 512K CoCo3, OS9 Level 2, 80 Column Monitor \$24.95

MiniBanners09 by Allen C. Huffman

Print Single or multiple line banners on ANY printer!
Supports graphi or non-graphic printers (like Daisy wheels, plotters, or even non-standard printers like the TP-10)! Independently sized text on each line (up to 16). Comes with over 30 fonts. Fully menu driven and easy to use. Now output to any device or file! (RSDOS version reviewed October 1991 Rainbow page 54.) Req: 512K CoCo3, OS9 Level 2 Any Printer \$19.95

MAC to DMP Print Utility by Carl England

Dump MACintosh picture files direct from disk to your DMP printer! Now even a CoCo 1 or 2 (which cannot adequately view a MAC picture) can print them out. Comes with 6 sample pictures to get you started. Req: CoCo, OS9 Level 1 or 2, DMP 105/106/130 Printer 14.95

Worlds at War by Trevor Milne

A complete WAR GAME DESIGNER for OS-9!
NEW! War game fans unite! Create multi-color icons for game maps and pieces. Define terrain (damage, supplies, shielding, etc.), artillery (attack, distance, units, fuel capacity, cargo payload, fire power, etc.) and more! Multi-screen playfield. Powerful graphics interface. Plays and looks great.
(For two players) Req: 512K CoCo3, OS9 Level 2 29.95

Document Printer by Carl England

Print booklets on your DMP/Epson printer!
NEW! Feed it text files and it prints sideways on a Tandy DMP or Epson printer. Fold the sheets and instant booklets. (OS-9 and RS-DOS versions included.)
Req: CoCo3, OS9 Level 2 or RS-DOS \$24.95

Sub-Etha Software
P.O. Box 152442
Lufkin, Texas 75915
(409) 639-ETHA [3842]

Call or Write for Information!
Add \$2.50 S&H and \$4.50 C.O.I.D.
Texas residents add 8.25% tax.
"Don't Panic — We ship Fast!"

BBS Listings

BBS Name	City/State	Network	Hours	Max Baud	Par	Phone
Narnia BBS	Marysville, CA	StG Net	11p-11a	2400	8N1	(916) 749-1107
Night Gallery	El Monte, CA	StG/Usenet	24hrs	2400	8N1	(818) 448-8529
Zog's Cavern	Toluca Lake, CA	StG/UseNet	Nightly	2400	8N1	(818) 761-4135
Stone Wall	Tijunga, CA	FID:102/823	24hrs	2400	8N1	(818) 781-9506
Milky Way	Novato, CA	RCIS	24hrs	2400	8N1	(415) 883-0696
Cassie's Corner	Huntington Bch, CA	ACBBS	24hrs	2400	8N1	(714) 841-0116
Kzin BBS	Surrey, BC, Can	FIDO/StG	24hrs	2400	8N1	(604) 589-5545
So, Alberta Bul	Lethbridge, AB, Can	StG Net	24hrs	2400	8N1	(403) 329-6498
Sci Fi	Poco, BC, Can	StG/FIDO	24hrs	2400	8N1	(403) 944-6390
The Dutchess	Wappingers Fls, NY	ACBBS	24hrs	2400	8N1	(914) 838-1261
SandV BBS	La Grange Pk, IL	StG/Usenet	24hrs	9600	8N1	(708) 352-0948
Nobody's Home	Ocala, FL	StG Net	24hrs	2400	8N1	(904) 245-6585

Don't see your favorite BBS listed here? Have your SysOp fill out the form below with the following information on his/her BBS to the address below.

BBS Name _____	Send to:
SysOp Name _____	The OS9 Underground
Address _____ City _____	BBS Listings
State/Prov _____ Zip/Postal Code _____	4650 Cahuenga Blvd.
Network? Yes ___ No ___ Network _____	Suite # 7
BBS Phone _____ Voice Phone _____	Toluca Lake, Ca.
(Address and Voice Number are treated as confidential)	91602

Want Ads

For Sale:	
(2) Coco 2 64K	\$25.00 ea.
MPI (26-3034)	\$35.00
Metric 101 Ser/Par Intfc ..	\$25.00
RS Sound Speech Pak	\$15.00
Coco Disk Cont (26-3022) ...	\$20.00
Microworks DS-69 Digitizer ..	\$20.00
64K Serial Printer Buffer ..	\$35.00
64K Parallel	\$35.00
ADC (BSR) 1200 Bd Ext Modem	\$35.00
RS Hi-res Mouse interface ..	\$10.00
(6) -1/4 Ht DDDS Bare Drives	
(Yes, these are 1/4 HT!)	\$25.00 ea.
CGP-115 Plotter (Gray Case)	\$25.00
(2) Black Joysticks	\$ 5.00 ea.
Light Pen (Coco Joystick) ..	\$15.00

Ask for Dan
(818) 781-6573 (9am to 9pm PST)

For Sale:	
CoCo2 - 64K	\$20.00
CoCo2 - 16K	\$15.00
Ask for Alan (9am to 7pm PST)	
(818) 761-4135	

Wanted:
CoCo3 Basic Book, the one you get when you buy a Coco 3
Ask for Dan
(818) 781-6573 (9am to 9pm PST)

Advertising Index

Company	Page
delmar Co.	IFC
OS9 Underground Subscriptions	4
Sirius Software/Hardware	5
DCS Banners and Signs	12
AniMajik Productions	15
Gale Force Enterprises	18
OS-9 Users Group	21
OS9 Underground Advertising	28
Sub-Etha Software	IBC



Coming next month...
"Underwater Robots, NV Ram ad OS9"

The OS9 Underground

Magazine Dedicated to OS9/OSK Users Everywhere!

CONTENTS

Under it All (Editor's Ramblings) 4

FEATURE:
"Test Driving the 6309" 6
by Mike Orloff

COLUMN:
"BASIC Traing" 8
by Jim Vestal

PROGRAM:
"Doing the Split" 9
by Wayne Campbell

COLUMN:
"Software Engineering" 10
by Leonard Cassady

FEATURE PROGRAM:
"ConvC - Text Utility" 13
by Andy DePue

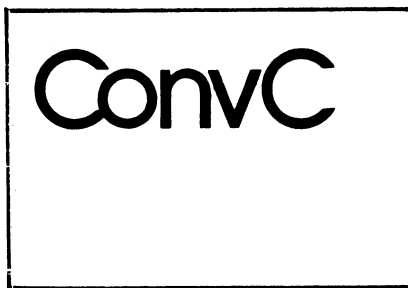
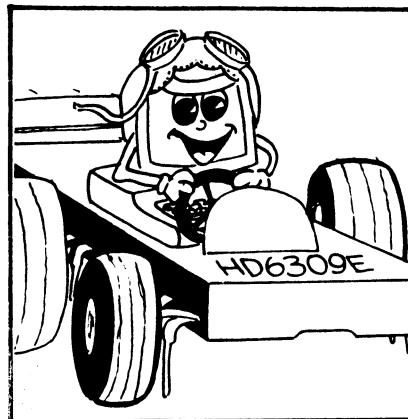
CENTER PAGE SPECIAL:
"Chicago, Chicago! It's Fest Time!" . 16

COLUMN:
"My CoCo PC" 22
by Caroyl Johnson

COLUMN:
"C-Notes! - Using Data Modules" 24

COLUMN:
"Reviving the Old Spirit" 29
by MOTD Editor, Scott McGee

Want Ads BBS Listings Ad Index 30



Under it all...



In this, the premiere issue of THE OS9 UNDERGROUND we feature an article about the 6309 chip that has been discussed and bandied about on various computer networks. Mike Ortloff's compilation will show some of the pros and cons of replacing the Coco's 6809E with Hitachi's "6809 Clone", the 6309.

Three programming columns, one for Basic09, one for C and yet another that will cover software engineering for both, will be begin with this issue. Jim Vestal's article on "Structured B09" Programming will help you to break some bad programming habits, while Andy DePue's excellent tutorial shows you how to make use of data modules in C. Leonard Cassidy, no novice to C, also starts his fine series on Software Engineering and how to program for business.

Of course, a magazine isn't complete without a Question and Answer column, and our celebrated Paul Pollock will be there to field your Questions from programming to Hardware.

I round this issue out with a special "Congratulations to the CocoFest" and to Dave Meyers, who has helped to keep our community "in touch".

This is my first issue, but not my first venture into publishing. I have worked in the graphics trade for almost 20 years now and have produced advertising and newsletters for many major corporations, such as McDonald's, Max Factor and Holiday Inn, to name a few.

I have tried to use OS9 where ever possible in conjunction with producing this magazine. Dynastar is my editor of choice and SDB for my subscription list. I even wrote my own custom "mini-DTP" in b09 just for this magazine.

Each month, I hope to bring you the best in articles about our favorite operating system, OS9, in all of it's varieties.

Program and article submissions are always welcome by you and encouraged. Deadlines for programs, articles and ads are the 10th of each month for publication in the following month.

I would also appreciate any comments, complaints, suggestions, kudos you may have about the "OS9 Underground". Who knows, I may even have to have a "letters to the editor" column too. See you next month with our coverage of the Chicago CocoFest!

Do your OS9/OSK Machine a Favor... Subscribe to

The OS9 Underground®

Magazine Dedicated to OS9/OSK Users Everywhere!

One (1) Year subscription (12 issues) \$18.00
(\$23.00 Canadian, 27.00 overseas)

Special Intro Price...!

Send in your Subscription before July 1st 1992 and you'll get \$6.00 off the regular subscription rate.

Send your Check or M.O. (U.S.) to:
(Make check payable to: AniMajik Productions)

"The OS9 Underground Magazine"
Fat Cat Publications
4650 Cahuenga Blvd. Ste #7
Toluca Lake, Ca 91602

Reviving the Old Spirit...

By OS-9 UG MOTD Editor, Scott McGee

[Included here is an excerpt from an interview with the President of the newly-formed OS9 User Group, Boisy G. Pitre. -Ed]

I had a chance to talk with Boisy and asked him a few questions about his feelings concerning the OS-9 Users Group. I have included below the responses he gave me to questions I thought readers might like to hear.

Tell me about the OS-9 Users Group?

The OS-9 Users Group is a self-sufficient organization dedicated to the growth and expansion of the OS-9 operating system. We BELIEVE in OS-9. Our officers are all experienced in the operating system and have equitable ability to provide the professionalism that OS-9 deserves.

Why start another (New) OS-9 Users Group?

Well, "new" is a trivial term to use. I think "revived" better describes the OS-9 Users Group. Its a revival of the spirit of the old Users Group. In its time, the old group served well. Now, there are new systems that promise to be excellent choices or OS-9 users, and this growing base will need a professional organization to represent it.

What purpose will the Users Group Serve?

Our primary goal is the dissemination of information concerning the OS-9 operating system. As our theme says, we are "dedicated to excellence in OS-9 computing." The OS-9 Users Group also gives OS-9 users a 'home base' to meet other OS-9 users and share information with them.

What is the likely audience for the Users Group?

Just about every professional OS-9 user out there! Engineers, students, programmers, casual users... because we have something to offer for everyone, from OS-9/6809 to OS-9000, everyone benefits.

What systems do you support? Software, Hardware, revisions?

The Users Group has made it a point to support all versions of OS-9, from OS-9/6809 to OSK to OS-9000. Our goal has been to encompass every OS-9 user,

and offer them practical information on their particular version of OS-9. As for hardware, we will only support systems as they relate to OS-9.

How is this Users Group different from other user groups?

The OS-9 Users Group is a professional organization. I don't think that we are different from the OS-9 Community Network or foreign user groups as far as the mission goes, but we do have different means to our goals.

The OS-9 Users Group Librarian, Alan Sheltra, is putting together a library of software disks which will be available to our members. And with new software being added regularly, the library is expanding.

Is the software/newsletter available to non-members?

The MOTD is available to paid members of the OS-9 Users Group. However, copies of our premiere issue may be shipped with the MM/1 from IMS, and the Tomcat systems from Frank Hogg Labs and other machines.

How do I join?

Becoming a member is easy! Just pick up an application from any one of the many available information services, or write to:

The OS-9 Users Group
P.O. Box 434
Farmington, Utah 84025

with your name, address, system type, and diskette size/format. Along with this information, all that is needed is a money order for \$25.00 to secure your membership for one year. Upon receipt of your membership application, you will be processed and your membership will go into effect immediately.

What do I get when I join?

Along with your one-year subscription to the MOTD newsletter, you will the introdisk for your system from the Users Group Library.

How do I contribute programs?

You may contribute programs to the OS-9 Users Group Library by sending your program and documentation to the address in care of Alan Sheltra, the Users Group Librarian. It will be evaluated and, if it merits, be included in the OS-9 Users Group Library. Articles for the MOTD are also welcomed and may be sent for submission to the same address above.

```
*****
* Data Module for use with 'Chat' program.
* Copyright (C) 1992, by Andy DePue.
*
* This module is currently designed for two users only.
*
```

```
TypeLang equ $40 * Define Module Type as Data and Language as Data
AttRev equ $81 * Module Attributes Set to Re-Entrant - Revision is 1
Edition equ $01 * Module Edition is 1
Stack equ $00 * Since This is a Data Module, No Stack is Required.
*****
```

```
*****
* Data Module for use with 'Chat' program.
* Copyright (C) 1992, by Andy DePue.
*
* This module is currently designed for two users only.
*
```

```
TypeLang equ $40 * Define Module Type as Data and Language as Data
AttRev equ $81 * Module Attributes Set to Re-Entrant - Revision is 1
Edition equ $01 * Module Edition is 1
Stack equ $00 * Since This is a Data Module, No Stack is Required.
```

```
psect ChatMod,TypeLang,AttRev,Edition,Stack,DataStrt
```

```
*****
* Actual Data of the Module
*
* Data Module is Arranged as Follows:
*
* Offset Size Description
*-----
* $0000 $0001 Number of Records in Data Module
* $0001 $0073 * Records Records For Users (Chatting)
*
```

```
Records equ $02 * Number of Records (2)
```

```
DataStrt fcb Records
```

```
UserRecs
REPT Records * Set up User Records Now

fcb $00 * Entry Usage Flag
fdb $0000 * User ID
rzb 32 * RMA's RZB Statement is Used Here to Reserve a 32 Byte
* * User Name Buffer
rzb 80 * This Will Create an 80 Byte Text Buffer
```

```
ENDR
endsect
```

<EOF>

Advertise in the
"OS9 Underground" Magazine

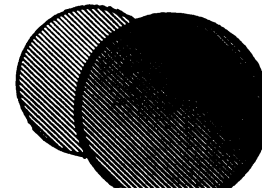
Reasonable Rates

Call or Write for our Free Ad Rate Card
"The OS9 Underground Magazine"

Fat Cat Publications
4650 Cahuenga Blvd. Ste #7
Toluca Lake, Ca 91602
(818) 761-4135

ATTENTION: Tomcat 70 and MM/1 Owners!

Get the Best of Both Worlds...



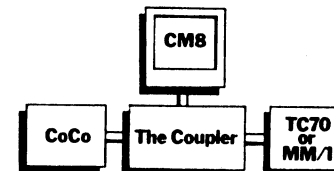
THE COUPLER

...Now you can use your CM8 on both your CoCo and your *New Age Machine* without switching cables.

The COUPLER was designed with you in in mind. Now all you need to do is flip a switch and you change which computer your monitor is hooked up to.

Here's how it works:

The COUPLER contains gold contacts, assuring you of the best possible connections for your computers and monitor. Also with the custom cable from the COUPLER to your computer you will be in effect adding an "extension cord" to your system and thus not have to "cram" everything so close to your monitor as the CM8 come with so short a cable.



Tomcat 70 Owners: The COUPLER may also be used as a switch between GWindows and your modem port, so you won't have to be constantly switching the port cabling.

The COUPLER:
Priced at \$75.00

OS9 TOOLKIT

The OS9 TOOLKIT is now available for your OSK machine. The TOOLKIT adds the following commands to your system:

Cls, Convert, Cpy, Date, Display, Locate, Mkdir, Nocntrl, Read and Size.

Priced at \$49.95

SIRIUS
SOFTWARE and HARDWARE

16750 Parthenia Ste#234
North Hills, Ca 91343

Call: (818) 894-0012



by Mike Orloff

Around February 22nd, 1992, a gentleman in Tasmania, an OS9er inquired of the readers of the InterNet (the global hi-speed computer link where many in the OS9/OSK community get their updates and entertainment) as to the availability of microprocessors compatible with and faster than our beloved Motorola 6809. In response were the expected deluge of messages suggesting the 68000 family, but a cryptic note from a student at Hiroshima University caught the attention of Kevin Darling and many others currently involved with or using OS/9 in various 6809-based systems.

The student, one Hirotsugu Kakugawa, had posted his experiences and discovery that the Hitachi product, the HD63C09, had certain features designed into it that were compatible with the existing 6809 architecture and provided expanded capabilities and faster operation over the 6809. Apparently his discoveries were quite by accident, as these features (discussed in detail in subsequent articles) were not mentioned in the Hitachi specifications for the 6309. According to Mr Kakugawa, the features were discussed in the Japanese OS/9 magazine Oh!FM in the April 1988 issue (4 years ago!! Are we asleep or what?!) and verified by independent Japanese OS/9 hackers. In response to Mr. Kakugawa's original message, an incredulous-sounding Kevin Darling requested more info, and got it.

Following the discussions between Mr. Darling and Mr. Kakugawa, I and a good many others became excited. Was this the rebirth of the Color Computer? Could ancient GIMIX and Peripheral Tech boxes be reincarnated as platforms to rival the power of the 68000-based machines? Were there chips made to complement this additional power and would the peripheral chips in the current 6809-based CPUs require updating to take advantage of the faster up? A "bazillion" questions, and patience and careful observation of discussions on the BITNet and UseNet proved to be prudent.

It would seem Kevin Darling has been using a 6309 for a good many years in his Color Computer, without using nor

Mr. Darlings' associates has been using the 6309 reliably at a clock speed of 5 mHz, an unsupported speed, (the 63C09 being rated to only 3 mHz in the specs) also without realizing there were additional undocumented features. In response to the excitement regarding the potential new applications of the 6309, one Takeshi Miyazaki posted a warning, that the extensive and powerful undocumented features were not only undocumented and unsupported, but untested as well.

According to Net discussions revolving around the reason such features existed, it seems that Motorola wanted Hitachi to produce a simple, straightforward CMOS clone of the 6809. In the process of developing such a beast, Hitachi added additional capabilities to the basic 6809 architecture without Motorolas' direct involvement. These capabilities were to run in a new mode, the 6309 "native" mode, while it retained an ability to emulate the 6809 in the "emulation" mode. It appears this upset Motorola and they informed Hitachi that this violated certain agreements and patents, so Hitachi simply produced and documented the emulation mode, leaving the new features engineered onto the production mask. Hitachi's production technique resulted in these features being fully functional on a majority of the chips, yet they have yet to be tested, by other 6809 users, in any statistically significant number.

THE SEARCH BEGINS...

Beginning April 10th, 1992, I began calling various suppliers mentioned by posters in the UseNet newsarea comp.sys.m6809. The first one I tried being Marshall Electronics in \$XXX\$, Texas. During that first call, I spoke to a Dan Pence, who informed me they were out of stock at the time but were expecting more a few weeks later. I followed this up with a call on Apr. 27th, when another conversation with Mr Pence concluded with him encouraging me to contact another company handling Marshalls' marketing, as they didn't accept orders under \$200.00 US. Mr Pence was extremely helpful, understanding and sympathetic in referring me to this other company, insisting I call back if I encountered any problems.

Following another lead, supplied by paulba@lab.tek.com, to the readers of the BITNet CoCo listserver, I contacted Western Micro, Hitachi's sales and distribution representative in the U.S., where a conversation with a gentleman by the name of Mike O'Brian led me to a local lead, Quad Rep \$whew!\$. A brief discussion with a very nice sales rep there resulted in a sample quantity of HD63C09EPs being sent out to Animajik. As we go to press, I have yet to receive the sample

```

/*
** printstats() - Prints Information On Data Module
*/

printstats()
{
    int counter; /* General Purpose Counter */

    printf("Stats on Data Module '%s':\n\n",MODNAME);
    printf("Module Linked At: %u ($%X)\n",modptr,modptr);
    printf("Offset to Data : %u ($%X)\n",modptr->m_data,modptr->m_data);
    printf("Data Begins At : %u ($%X)\n",mod,mod);
    printf("\nContents of Data Module:\n\n");
    printf("This Module Contains %d User Entries.\n",mod->NumOfRecs);
    /* Print Each Record */
    for (counter=0;counter<mod->NumOfRecs;counter++)
    {
        printf("\nStats For User Record #&d:\n",counter);
        printf("This record currently is%s in use.\n",
            mod->UI[counter].UsedFlag==EMPTY?" not":"");
        printf("User's User ID: %u ($%X)\n",mod->UI[counter].UserID,
            mod->UI[counter].UserID);
        printf("User's Name : %s\n",mod->UI[counter].UserName);
        printf("User's Buffer : %s\n",mod->UI[counter].TextBuf);
    }
    printf("\nEnd of Data Module\n");
}

/*
** Chat.h - Copyright (C) 1992 by Andy DePue.
** Header file for Chat.c
*/

/* Include header files */

#include <stdio.h>
#include <module.h> /* Includes structures for OS-9 Memory Modules */

/* Setup Constants */

#define MODNAME "ChatMod"
#define USED 0xFF /* $FF=Entry in Use */
#define EMPTY 0x00 /* $00=Entry Empty */
#define DTYPE 04 /* Module Type for a Data Module */
#define LTYPE 00 /* Language Type */
#define DEFRECS 02 /* Default Number of Records */

/* Setup Data Module Header Structure (Returned From Link) */
mod_data *modlink(),*modload(),*modptr;

/* Setup Data Module Structures (or "Templates") */
/* Structure for a User Entry */
typedef struct {
    char UsedFlag; /* This Entry in Use? */
    int UserID; /* User ID for This Entry */
    char UserName[32], /* User's Chat Name */
        TextBuf[80]; /* Text Buffer for This User */
} USER_ENTRY;

/* Structure for Entire Data Module */
typedef struct {
    char NumOfRecs; /* Number of User Records (Entries) in This Module */
    USER_ENTRY UI[DEFRECS]; /* Setup Two User Records (UI for User Info) */
} DATA_MODULE;

/* Now Setup "Template" for Data Module */
DATA_MODULE *mod; /* Data Module Structure */

```



Linking to a data module is very simple; just execute the OS-9 F\$Link system call with the proper parameters. If an error occurs, it would be best to then execute the OS-9 F\$Load system call to check for the data module in the CMDS directory, if it is found then it will automatically be linked. If an error still occurs, then act accordingly. Once finished with the data module, execute the OS-9 F\$Unlink system call. Once the data module has been linked, the next step is to calculate the address of the data from the header information returned. To do this, add the offset of the data to the address of the header. The offset can be found in the header itself, as defined in 'module.h'. Since we are going to be writing this program in C, it is best to create a "template" for the data module. This can be accomplished with 'typedef struct'. Since this structure must be a pointer, simply take the address of the data and literally point your template in the right direction. The data module can then be accessed just

as if it were any other global variable. We can now begin to lay the groundwork for the 'Chat' program by creating the proper structures for the data module and putting them in a header file for 'Chat' along with other useful information. See program listing 2 for the source of this header file. Because of space limitations, I won't be able to give you the 'Chat' program this month, but I will leave you with a small program which uses the Chat header file (program listing 3). It simply links to the module, prints info about it, and unlinks so as to demonstrate how it is actually done. This program will, of course, show an empty data module for now, but when we write the Chat program it will be very useful to the System Operator for seeing who all is using the data module at the moment. The routine 'getmodule' in this program will also be used in Chat as well. Well, that's all for this month. In the next article of this series we will begin work on the Chat program itself. Until then, happy programming!

```

/*
** LinkModule.c - Copyright (C) 1992 by Andy DePue
** A demonstration of linking to a data module (in this case 'ChatMod')
** and setting up the proper structures for it. This program simply links
** to the module, prints information on it's memory location, prints
** information on it's contents, unlinks, and exits
*/

#include "Chat.h" /* Header file for 'Chat' program, used here as well */

/* main() - Links, Prints Stats, Unlinks, and Exits */

main()
{
    if(getmodule(MODNAME)==-1) /* Attempt Linking to ChatMod */
    {
        /* If Attempt Fails, Inform User and Exit */
        fprintf(stderr, "LinkModule, Error: Cannot Link to '%s'\n", MODNAME);
        exit(errno);
    }

    printstats(); /* Print Stats on Module */
    munlink(modptr); /* Unlink Module */
    exit(0); /* And Exit */
}

/*
** getmodule(name) - Attempts to link to data module. If it fails, it attempts
** to load data module. If it still fails, it returns an
** error.
*/

getmodule(name)
char *name;
{
    char *databegin;

    if((modptr=modlink(name, DTYPE, LTYPE))== -1) /* Attempt Link */
    {
        /* If Link Failed, Attempt Load */
        if((modptr=modload(name, DTYPE, LTYPE))== -1)
            /* If Load Failed, Return Error */
            return(-1);
    }

    databegin=modptr; /* Assign address of module header */
    databegin+=modptr->m_data; /* Calculate address of data from offset */
    mod=databegin;
    return(0); /* Return Non Error */
}

```

quantity requested (no fault of QuadRep I'm sure, whaddaya want fer nothin', o'ernite delivery?) however, the editor of the Underground received the Hitachi microprocessor databook on May 8th, like 24 hours after requesting it from QuadRep. Quoting from the Data Book, much of which is virtually identical to the specsheet on the 6809, except...

"The HD63C09E is the highest 8-bit microprocessor of the HMCS6800 family, which is just compatible with the conventional HD6809E".

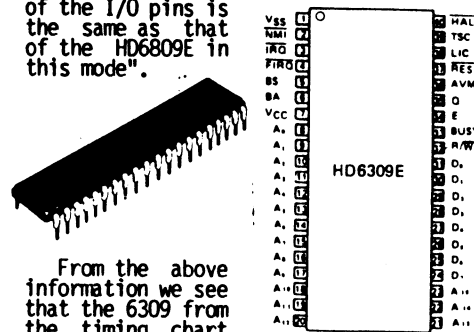
"The HD6309E has hardware and software features that make it an ideal higher level language execution or standard controller applications. External clock inputs are provided to allow synchronization with peripherals, systems or other MPUS".

"The HD6309E is a complete CMOS device and power dissipation is extremely low. Moreover, the SYNC and CWAJ instructions makes low power applications possible".

FEATURES:

- o Hardware-Interface with ALL HMCS6800 Peripherals
- o Software-Object code compatible with the HD6809E
- o Low Power Consumption Mode during SYNC state of SYNC instruction and during the WAIT state of CWAJ instruction. (Sleep Mode)
- o External Clock Inputs, E and Q, Allow Synchronization
- o Wide operation range
f = 0.5 to 3MHZ (Vcc = 5v +/-10%)

Sleep Mode:
"During the interrupt wait period in SYNC instruction (the SYNC state) and that period in the CWAJ instruction (the WAIT state), MPU operation is halted and goes to the sleep mode. However, the state of the I/O pins is the same as that of the HD6809E in this mode".



From the above information we see that the 6309 from the timing chart in fig. 2 (page 8) and the illustration in fig.1 above, that this chip is pin-for-pin compatible with our beloved Motorola 6809. Convenient! I couldn't see any difference in functionality between the

two chips either, aside from this slight difference in bus timing. (hey, will this difference help with the "sparklies"? Inside joke, guys) and this most unusual documented feature, the "sleep mode". What use this feature may be, is up to the experimenters among us to determine. A Laptop OS/9 box, maybe? With the extended instruction set, companion hi-speed CMOS peripheral chips?

As of press time, this is the only firsthand information I have for this article, although monitoring DELPHI, FIDO and BITNET leads me to conclude many individuals are already running the Hitachi chip with no problems, in the default emulation mode. Next month, with luck and a good tailwind, I should be able to report my initial findings in testing the features of the chips I receive. In any case, I'll be adding the "reference card" created by Kevin Darling, originally from Mr Kakugawas' raw info, in sections, in upcoming articles. As well, the next article will include some simple instructions for those familiar with EDTASM and the OS/9 assembler under Level I so all of you to see for yourself how the new features work...

If, indeed, they do!

WORK TO BE DONE

As this is to be an ongoing series, documenting my own discoveries as well as that of other users, watch for future articles, some copied from the BITNet, some from UseNet and others from Fido as well as StGNet. In order to provide you with firsthand information about the 6309, and knowing that the 6309 is pin compatible with the 6809, I decided I would drop a 6309 in my CoCo 2 and test it under OS9 Level 1 (the 6809 in my CoCo III ain't dead yet, and so isn't socketed) as well as Landys' DECB. I plan to check ALL the chips I receive, and in articles to come, let you, the readers, know of my results. As these features are unsupported and untested by the manufacturer, it's up to us, the OS9 user community, to see for ourselves like what percentage of these chips actually have the additional features implemented reliably. Seems if like 100 of us test, and report our findings, that would be an adequate sampling to begin inferring that maybe, just MAYBE, we have a future in OS9-6809 after all. Arriving on So, with Kevin Darlings' permission, we commit the thorough documentation that Kevin posted to the BitNet, to print, for your consternation and enlightenment. Challenges, flames, kudos and whatever may be directed to the editor here at the OS9 UnderGround or to boobie@mcws.fidonet.org.

<EOF>

NOTES:

by Andy DePue

I have been asked by several people to explain the practical use of a "Data Module", and have also had requests for information on how to allow several programs to communicate with each other. Since the two go along with each other so well, I have decided to write a series of articles about them. These articles will be focused on data modules, how they relate with "inter-process communications", and how to use them with the C language.

WHAT A DATA MODULE IS:

A data module is essentially an OS-9 memory module that contains data similar to executable OS-9 program modules. However, instead of containing program code, it contains data that is meaningful to a program. Because of this, a data module can be loaded and linked just like any other OS-9 memory module. This makes it useful for inter-process communications since any number of programs can link to the data module, and each program that is linked to the data module is actually sharing the same physical RAM. For further information on OS-9 memory modules see the OS-9 Technical Reference, chapter three.

USES FOR A DATA MODULE

Data modules can be used for a variety of things such as a fast and efficient method of storing data (which can be loaded and saved as well), a permanent storage for a program's global variables or configuration, a method for inter-process communications, etc... I will be explaining the latter in this article.

HOW TO MAKE A DATA MODULE

There are basically two ways to make a data module. Each method has a good side and a bad side. I will have a short explanation of one method first then go into the second method in detail. The first method is to write a program (in the language of your choice) that will output the proper header information for a data module, write

the actual data, and then calculate the CRC of the final product. Once you have written this program, you could easily have it output any data module size you want. This method is good if you want to quickly make a data module with the same initialized value throughout the entire module, but it doesn't allow much control over the data within the data module. Thus; if you had a program that required, say, a value of \$00 at the end of every used "record" and a value of \$FF at the end of every non-used "record", you lose out since you would not be able to pre-define each record. The second method is to use an assembler such as "ASM" or "RMA". Simply set up a module header for a data module and, using Pseudo-Instructions such as FCB and FDB, fill it in with any data you want. This is good if you want full control over the data module and the data within it, but not everyone has access to an assembler. Since these articles are written for programmers who wish to interface with data modules via C then they must have an assembler (either C.ASM or RMA), so I will use the second method. This method also gives a much clearer idea about the inner workings of a data module. Before you start to build a data module you will most likely want to sit down and make a rough outline of what you want it to do and how you want it organized. Once you have the basic idea outlined, expand it to include more detail so you can begin to build it with an assembler. This will make it alot easier to build and then interface with the data module.

CREATING OUR OUR DATA MODULE

With all the technical information out of the way, we can now build our own data module. First, a purpose. What should this data module be designed for? Well, this article is about getting programs to communicate with each other, so that should be a good place to start. But, what kind of programs? How about if we design a simple "chat" between two people? As in: if a user who is logged into a terminal (/T2) wants to chat with a user who is logged into the computer console (say, /Term), then all he has to do is type something like "chat /term" and the user on /term would be paged. The user on /term would then type "chat" and the two would be linked together with a means to communicate. Thus, the data module would be allowing two programs (in this case, "Chat" on /T2 and "Chat" on /Term) to communicate with each other. Now that we know what the data module should do, we decide how it can be done. The data module should have at least two "entries"; one for each user, or program, that is going to be linked

ends. Another very fundamental concept of structured programming is the use of comment or remark statement to help document the internals of the program logic. All programmers should make liberal use of the REM statement in BASIC. BASIC09 allows the use of the Pascal-style remark symbols: (* This is a remark statement *). Place remarks at the beginning of each procedure and each subroutine, and remark each program statement that is not 100% self-documenting. For example, no remarks are needed after the print statement in the following code, but remarks after the calculations help to document the program:

```
PRINT "This Sample program written
      by Jim Vesta."
s := a(n) + b(n) + c(n) ½
(* Calculate the sum of the entries *
n := n + 1 ½ (* increment row number *
```

COMING NEXT MONTH

The next installment of this column will concentrate on the rules of using line numbers, subroutines, and goto statements within a structured BASIC09 program. (Editor's Note: You can reach Jim for comment or questions at one of the following addresses: Delphi :JIMVESTAL, StG Net:SysOp@Narnia or US Mail to Jim c/o this magazine.)

<EOF>

DOING THE SPLIT - SPLIT.B09

By Wayne Campbell

```
PROCEDURE Split
(* *****
* Split - Text File Splitter Program
* by: Wayne Campbell - April 1992
*
DIM path,path1,cls,char1,char2:BYTE
DIM lnum,count:INTEGER
DIM ext1,ext2:STRING[1]
DIM lnum:STRING[3]
DIM file1,file2:STRING[40]
DIM line(100):STRING[80]

cls:=12
ext1:="a"
ext2:="a"

(* Clear the screen
PUT #1,cls
PRINT
(* Input filename
INPUT " File to Split: ",file1
(* Output filename
INPUT " Sub-File Name: ",file2
IF file2="" THEN

(* Default output filename
(* w/ extension
file2:="x.a"
ELSE
(* User specified output
(* filename w/extension
file2:=file2+".aa"
ENDIF

(* # of lines to write per output file
INPUT " Lines per Sub-File: ",lnum

IF lnum="" THEN
(* Default # of lines if no input
lnum:=100
ENDIF
lnum:=VAL(lnum)
IF lnum=0 THEN
(* Default # of lines if input is 0
lnum:=100
ENDIF
PRINT
ON ERROR GOTO 2
OPEN #path,file1:READ
(* Read lnum lines from input file
FOR count:=1 TO lnum
READ #path,line(count)
NEXT count
PRINT file2,
(* Create output file
CREATE #path1,file2:WRITE
(* Write lines to output file and
(* close file
FOR count:=1 TO lnum
WRITE #path1,line(count)
NEXT count
CLOSE #path1

(* Update output filename and repeat
(* process
GOSUB 3
GOTO 1

2
(* When EOF is reached, create output
(* file, write lines, and close file
CLOSE #path

lnum:=count
PRINT file2,
CREATE #path1,file2:WRITE
FOR count:=1 TO lnum
WRITE #path1,line(count)
NEXT count
CLOSE #path1
PRINT
END

3
(* Change extension identifier on
(* output file
char2:=ASC(ext2)
char2:=char2+1,ext2:=CHR$(char2)
IF ext2="2" THEN
ext2:="a"
char1:=ASC(ext1)
char1:=char1+1
ext1:=CHR$(char1)
ENDIF
file2:=LEFT$(file2,LEN(file2)-2)
file2:=file2+ext1+ext2
RETURN
```

SOFTWARE ENGINEERING

By Leonard Cassady

Welcome to the premier issue of OS9 Underground.

I've encountered many coding examples over the years, some well written. In the first part of this column, we'll explore the basic elements of software engineering relating to program design, programming style, hopefully avoiding the pitfalls usually encountered in poorly written program code.

The guidelines presented here may be applied to any programming language, although the focus of this column will be on the C language.

The second half of this column will be devoted to programming questions and problems submitted by readers.

The following procedures are similar to those used to design and produce commercial software. The guidelines presented have proven to be efficient and productive.

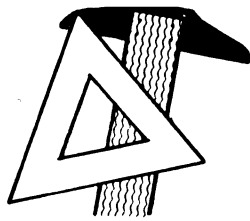
PROGRAM SPECIFICATION

A program specification should describe exactly how the program is expected to behave. It should describe how the final program will appear to the users. Vague language such as "fast response" or "easy to use" and the like should be avoided as these are relative terms.

"Consistency" and "simplicity" should be cardinal rules. A few primitive operations that meet all the user requirements is ideal.

Sometimes it is not clear what the set of primitive operations should be when designing the program. A suggested outline could be:

- 1) Abstract of project. The intent and goal of program operation and behavior.
- 2) Input format. User input, data input, or command selection method. This includes mouse-driven menus, command-line arguments, file formats, etc.
- 3) Screen design. The user interface should be concise easy to use and understand. This includes prompt messages, and the program bell and whistles.
- 4) Output format. Data file output, "in progress" reporting, etc.
- 5) Error messages. Diagnostic error messages.
- 6) Future expansions. Enhancements or increased functionalities.



Some or most of the suggestions may be excluded, depending on the particular program in design.

SOFTWARE DESIGN

Once the program is specified, the processing phases and major data structures need to be identified. Magazines, books, source code listings are excellent references, and should not be overlooked when difficult programming problems arise.

There are no physical parameters to measure against. The only guides are previous attempts at solving similar problems.

Important to consider is the speed, memory usage, and flexibility when choosing efficient data structures and algorithms. No amount of trickery can compensate for a slow, inefficient algorithm.

A common pitfall of software design is known as the NIH (Not Invented Here) syndrome. It is the tendency to feel that if the problem solution wasn't invented by the programmer, it can't be any good.

"CONSISTANCY" AND "SIMPLICITY" SHOULD BE CARDINAL RULES

An effective design method called "stepwise refinement" is often used to divide up a program at this level of abstraction into smaller subproblems, each less abstract, until each part can be easily implemented. Dividing a program in this manner clearly defines sections, or "modules" that are not strongly interrelated. The same method can then be used to decide the functions within each module.

The three basic phases of stepwise refinement are:

- 1) Identify major divisions of functionality. Define what goes into each program module or subsection.
- 2) Identify major data structures shared by each program module. Program modules identified in step 1
- 3) Create an additional module for each data structure. Data structures identified in step 2

The fewer module inter-dependencies, the easier the program will be to read and maintain.

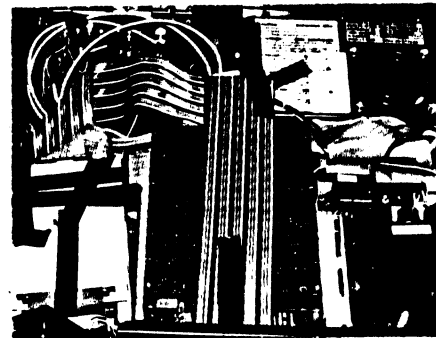
I enlisted the help of some friends to build a wide expansion slot cover out of sheet metal with 3 large holes punched in each for installing the DIN connectors to the back of the case. (Photo Below)



This proved to be one of the neatest features of my CoCo/PC! Now I have ALL the CoCo ports (and then some) neatly arranged along the back expansion slots just like the clowns, ops, I mean clones (g). It's amazing how fast the little parts all add up (\$\$\$)!

I learned something funny while wiring the audio... if you don't wire the ground properly (the Radio Shack salesman didn't even know how to) you can listen to Country & Western music! Of course if you live in California you may wind up listening to Heavy Metal!

Alas, the RGB connection was a chore. I unfortunately didn't find out in time that you can (supposedly) by a 10 pin plug which could have been installed in back of my case. I wound up cutting the existing RGB cable, lengthened it and installed a DB-9 connector since the CoCo cable had one hole plugged on it's connector, it obviously wasn't used. It took some trial and error to get the wires hooked up right, but the results were a cable that is 5 feet longer and will fasten securely to the case with thumb-screws.

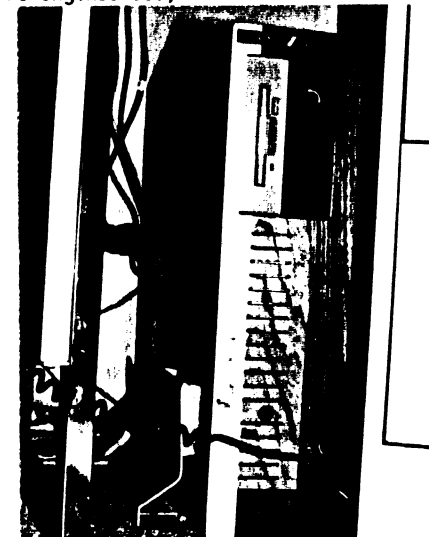


(CoCo PC with "innards" exposed)

The power supply has a fan built in, but to be sure the 512K upgrade and all the other components didn't get too hot, I added a 12 volt DC fan to the front of the case. It's a bit noisier this way, but I hope it will insure the

longest possible life for my CoCo/PC.

The question of what to do with the keyboard interface, and the controller (and later the RS232 pak) attached now with the home grown Y-cable was a very important one which would have a great deal to do with how successful the conversion would be when it's used, moved and jostled about in the months and years (hopefully) to come! I first installed a metal shelf covered in plastic attached to the side of the drive bay. However, after a run in with a bad disk (it's soft hub was misaligned causing the drive and it's neighbors to shake violently) caused problems with my CoCo's sound (likely due to a momentary shorting to ground), I replaced it with a plastic pencil case (properly mutilated, I mean re-engineered!)



The really unique part about my CoCo/PC is the front... it's wood! My other hobby is woodworking. I laminated Southern Oak, Honduras Mahogany, Walnut and Hard Rock Maple together, had it surface planed to 1/2" thick and cut out the opening for the floppy drives. Before gluing it up, however, I cut saw kerfs in the center board for ventilation and added a hole for my extended reset button to protrude through. It turned out great and without a doubt the only CoCo of it's kind!

I got a great deal on a used keyboard (108 key) and interface so the whole project cost about \$160 give or take a few dollars, and took several weeks to complete. The interface is easy to hook up and I have come to LOVE the bigger keyboard! I strongly recommend the conversion to all but the hopelessly timid or those unskilled hackers that don't have a friend to assist and/or troubleshoot.

<EOF>

My CoCo PC

by Caroyl Johnson

Last fall I was overcome by the desire to put my CoCo 3, hard drive and floppy drives into a PC case. I wanted to clear some space on my small desk and also have the ability to easily gather up my computer and take it along to club meetings or when I go on two weeks active duty for Naval Reserve training. Price was an important factor in the project, but not the most important one. I felt that a conversion worth the time, effort and cost of doing was well worth doing well.

First, I looked around for a suitable case. If purchased new, it would cost \$40.00 or more. I located a used XT case that was missing its front face piece. I checked out a friend's CoCo that had been installed in a similarly sized case for ideas to make it fit. I decided to buy the used case (for \$5) and build a wooden face piece. I also picked up a used 150 watt power supply (for \$35). The case came with 8 expansion slots and several slot covers. At first I planned to use the plastic box I'd built earlier to provide multiple bitbanger ports and joystick ports and mount it to the back of the case, but later decided to take a different approach.

The afternoon I brought home the case was the same day that my November 1991 Rainbow came in the mail. This was fortunate because there was a letter in the CoCo Consultations column (by Marty Goodman) which dealt with switching power supplies. I soon figured out that I'd have to find another way to hook up the motherboard (m/b) to the power supply as I'm not a "skilled" hacker and shuddered at the thought of cutting my beloved CoCo. I spoke with our club president, Tim, about an idea he had once mentioned on the subject. Tim advised that I tap into the switch using a normal extension cord (the female end) to gain 100 volts and leave the wimpy CoCo p/s inside the PC case with the m/b. This worked well and along with a few modifications to the m/b, fit my needs and skill level.

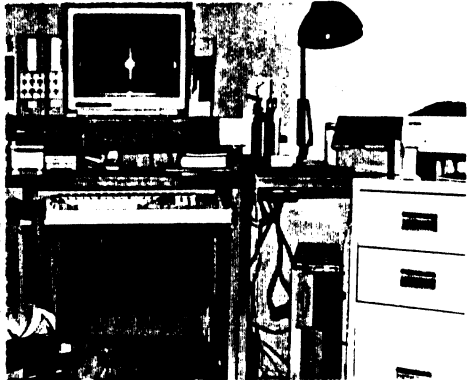
The case had hard drive bays for 2 full height or 4 half-height drives. I have a 40 track DDS drive and an 80 track 3.5" drive as well as a full height 20 meg hard drive. I picked up a spare CoCo 3 at the local Radio Shack Outlet Store just days before it closed (which came as a big surprise to everyone, including the manager) in case I messed up big-time during the conversion. As it turned out, I didn't need it, by my courage was strengthened just by knowing I had a CoCo (ace) in the hole! I learned something about these switching power supplies... they (at least mine did) require a minimum load be applied before they will work. The floppy drives will not draw enough power to do the job, but the hard drive had no trouble in meeting the requirement.

I planned on using Y-cable to hook up my Disto Super Controller and to leave room for a converted DC modem/RS232 Pak. When I checked with COCOPRO, Dave Meyer's said he had none in stock but could special order one.

I decided that price wouldn't mesh well with my limited budget so I took another approach. A friend from our local CoCo Cub, Gene, had a cost-effective solution and helped out by soldering a 40 wire ribbon cable to the bottom of the CoCo bus to which we added two female 40 pin edge card connectors. I had to file the edges of my controller's edge card so the ground tabs would fit around the female connectors I bought. (gee, a Y-cable would have been so much easier...but \$30 more).

Fitting all the components inside the case was the most difficult part (no surprises there, huh). My friend Gene also came to the rescue by ripping out the support brackets (with brute force and a pair of tin snips) in the bottom of the case which were creating a space problem. This freed up enough room to fit the corner of the m/b (that holds the RF modulator) under the leftmost drive bay. We removed the heat sink, drilled a new hole and mounted it lying on its side to allow for the cables crossing above it (be sure to use heat sink compound if you do this).

I made shielded cables to connect all the ports from the back of the CoCo to the back of the case. I took several of the slot covers and drilled them out to hold the smaller ports such as the TV, audio and video ports.



DEBUGGING

Not even the best programmers write bug-free code on the first try. A large part of the development time for the program will be spent finding and fixing program bugs. This is also known as "debugging". There are three laws pertaining to debugging:

- 1) All complex software has bugs.
- 2) The bug is probably caused by the last thing you touched.
- 3) If the bug isn't where you're looking, it's somewhere else.

Algorithms have the nature to grow more complex as they gain added functionality, becoming more difficult to read and maintain. The first step is to isolate erroneous behavior. Vague comments such as "it doesn't run", need to be defined clearly as, "After I type 'run', the program hangs".

Programming requires painful attention to detail, and humans are prone to err. We have the tendency to stick to first impressions. What may be obvious to a dispassionate person, may not be clear to the frustrated programmer. After an exhaustive, unsuccessful search for the bug where you know it has to be, try looking elsewhere.

Another phenomenon that affects software is the tendency to add too many bells and whistles. Adding new features usually means the program will run slower, require more system resources, and certainly introduce additional bugs.

TESTING

The two major test phases a program should be subjected to are "Alpha" and "Beta" testing.

The alpha testing should be done by the programmer for the express purpose of purging all bugs from the software. Once in this phase, no new features or specification changes are made to the software.

The beta test is best done by a dispassionate person before the software is released to the general public. This phase of testing insures the program will function properly in "real life".

<EOF>



OS9: the Q&A by Paul Pollock

FIRST EDITION

The OS9 Underground Magazine (or OS9UG) is finally off the presses. And this is the first of what I hope to be many inclusions in what surely is a much needed resource in the OS9 community.

Alan Sheltra, the Editor-In-Chief of the OS9UG, asked me some time ago, if I'd be willing to host this column. And after some thought, I heartily agreed. The rules here are fairly simple. This is a place for all users of OS9 (of all types); to ask questions, pose problems, and air difficulties.

You'll find separate columns (in the OS9UG) for specific programming languages.

This column is intended to answer those OS9 operating system questions, and programming algorithm methods, or hardware specific questions; which just don't seem to 'fit the nitch' anywhere else. To do this, just send your questions in one of several ways:

- 1) You can write directly, or call directly, the Editor (Alan Sheltra). His address and phone number is in the opening pages of OS9UG.
- 2) Contact the Editor, or I, via the StG-Net. This can be done either via a mailgram (public messages are OK if you want the world to read your submission (grin)), to:

OS9UNDER@ZOG

A dummy account specifically intended for submissions of this type.

SYSOP@ZOG

Another name for the Editor-In-Chief (or as we are ordered to refer to him, "Fearless Leader").

PAULBELL@ZOG

That's me. I'm extremely easy to reach.

It's a very good idea to place in the SUBJECT line, the name of this column: "OS9: Questions & Answers" (or just Q&A). This will route all proper traffic to me.

If I know the answer, you'll not only get an accurate and thorough answer, but you'll be adding to an important text-based resource for many other users. I promise!

If I don't know the answer, I'll seek out the finest minds in the galaxy, to reach an answer. If that becomes inconclusive, I won't waste you time by trying to sound pretentious, and noncommittal. I'll just tell you "I don't know". Those of you that have read me in the past, know that I can be gritty and devilish one minute, and inciteful and informative the next. So without further ado, let's get on with the show!

ASKING GOOD QUESTIONS AN EDITORIAL APPROACH

As some of you know, I own a shortwave radio. Some time ago, I was listening to these old guys talking to each other. This is a long-established practice known as 'rag-chewing' among friends.

Anyway, after some minutes, one of these guys asked an interesting question. But before he did, he explained the situation.

He told them that for many years, he was a committed soda-pop drinker. Even to the point of leaving a glass of soda on the bed-stand at night.

Well, recently he had gone to the hospital for a checkup. And they prompted him to discontinue the habit of drinking soda-pop, and drink water instead.

Finally he got to the point. Soda-pop always tasted the same, no-matter how long it had been left standing (except for the loss of carbonation). But water, tasting fine in the evening, consistently tasted awful, the following morning. How come?

As luck would have it, everyone party to the question, had his own idea of the answer.

Maybe it was the iron, rusting in the water? No, iron makes water taste delicious. The cleaner water is, the blander it tastes, yech!

Maybe it was germs in the water, growing over time? No, can't be any worse than the germs already in his mouth.

Maybe it was just his imagination? No, his imagination was as creative as ever (grin), no soap.

So after all of that, what was the answer? Well, here we can bring the problem into clear focus by asking the fellow a second 'fact finding' question.

"What did you eat, the night before?"

Sure enough, that makes the solution easy. The fact is, that during the waking day, we are constantly eating or drinking something. So anything that might be in our mouths which might otherwise alter the taste of things, is constantly being flushed out of our mouths.

But in sleep, whatever was last in our mouths, stays there for many hours. And it becomes stagnant, and 'flavorful'.

Since water rarely has much flavor, anything that's in your mouth already, tends to flavor the water.

In this case, morning breath is a distinct possibility; and is solved by the judicious use of a toothbrush.

Yea, but what's the point?

This is your Q&A area. And it is yours to use as you see fit.

But anything used badly, becomes broken and discarded.

If you have a simple question, which should have a simple answer, feel free to make your question; in the simplest form possible.

But if you think your question may require more complex or detailed data than what a couple of lines might satisfy; please feel free to submit as detailed and explained question as you feel it deserves.

And let me have your phone number and address (in private mail ONLY), so that I can give you the feedback your question might require.

On long or detailed questions, I might abridge your submission at press-time. But rest assured, every effort will be made to answer your question in the spirit in which it was submitted.

Questions to Mr. Paul Pollock may be

addressed to: "The OS9 Underground"

OS9: The Q&A

4650 Cahuenga Blvd. Ste. #7

Toluca Lake, Ca. 91602

<EOF>

BANNERS

and Posters for
Shows and Conventions &
Sales Promotions

- o Reasonable Rates
- o Fast Service
- o We Ship Anywhere

Call for estimate:

(818) 761-4135

(213) 460-2968 (Fax)

```
/* error(err,msg1,msg2,flag) - prints error message, exits if flag set */
```

```
error(err,msg1,msg2,flag)
int err; /* Error Code */
char *msg1,*msg2; /* Error Messages */
int flag; /* Exit Flag */
{
    fprintf(stderr,"ConvC, Error: %s%s\n",msg1,msg2);
    if(flag)
        exit(err);
    else if(err>0)
        prerr(fileno(stderr),err);
}
```

```
/* getdig(ptr) - gets the next digit in the string and returns the value. */
/* also updates the ptr. */
```

```
getdig(ptr)
char **ptr;
{
    int digi; /* Return digit value */

    /* Scan line until a digit is encountered */
    while(**ptr && !isdigit(**ptr)==0) *ptr=*ptr+1;
    /* Convert ASCII to INTEGER */
    digi=atoi(*ptr);
    /* Skip past ASCII digits */
    do
        *ptr=*ptr+1;
    while(**ptr && !isdigit(**ptr)==1);
    if>(*ptr+1) *ptr=*ptr-1;

    /* Go to end of ASCII digits */
    return digi;
}
```

<EOF>

After months of preparation, the OS-9 Users Group is now accepting applications for membership!

The OS-9 USERS GROUP "Dedicated to Excellence in OS-9 Computing"

All members of the OS-9 Users Group receive:

- o A one year quarterly subscription to the MOTD Newsletter, a quarterly periodical filled with insights and quality information on the OS-9/6809, OS-9/68K and OS-9000.
- o A complimentary utility diskette for your OS-9 system (First-time members only).

Becoming part of the OS-9 Users Group is easy. Just complete the form below, and mail it along with your check or money order of \$25.00 for a one-year membership.

- o Access to hundreds of quality OS-9 software titles from the OS-9 User Group Library.

- o More to come!

OS-9 Users Group
P.O. Box 434
Farmington, Utah 84025

[Please Print]

Name: _____

Company: _____

Mailing Address: _____

City: _____ State: _____ ZIP Code: _____

Home Phone: (____) _____

Business Phone: (____) _____

OS-9 specific interests: _____

```

/*
** convfile(in,out,linecount,tabcount) - This routine converts the text file
**                                         (Stripping LFs...etc..)
*/
convfile(in,out,linecount,tabcount)
FILE *in; /* In (Orig.) file */
FILE *out; /* Out (New) file */
int *linecount,*tabcount;
{
    int inchar;
    int curpos,curtab,counter;

    /* Init Variables to NULL */
    *linecount=*tabcount=curpos=curtab=NULL;

    /* Loop through entire file character by character and convert it */
    while((inchar=getc(in))!=EOF)
    {
        if(inchar!=lfchar)
        {
            if(inchar==TABCHAR && tabflag)
            {
                /* Get Next Tab Position */
                curtab=tabsize-(curpos % tabsize);
                for(counter=0;counter<curtab;counter++)
                    putc(' ',out);
                /* Update Current Line Position */
                curpos+=curtab;
                /* Update Number of Tabs Converted */
                *tabcount=*tabcount+1;
            }
            else /* If not a TAB, write out and update curpos */
            {
                putc(inchar,out);
                if(isprint(inchar))
                    curpos++;
                else if(inchar=='\n')
                    curpos=0;
            }
        }
        else /* If char is a LF, convert it */
        {
            if(lfchar>0)
            {
                if(newlf>0)
                    putc(newlf,out);
                *linecount=*linecount+1;
            }
            curpos=0;
        }
    }
}

docopy(realfile,tempfile)
char *realfile; /* Orig. File */
char *tempfile; /* Temp. File */
{
    int waitstats;
    char renstring[160]; /* Rename Command Line Option String */

    unlink(realfile); /* Delete Orig. File */
    strcpy(renstring,tempfile);
    strcat(renstring," ");
    strcat(renstring,realfile);
    strcat(renstring,"\n");
    if(os9fork("Rename",strlen(renstring),renstring,1,1,0)==-1)
    {
        fprintf(stderr,"Cannot rename temp file for '%s'!\n",realfile);
        fprintf(stderr,"Temp File: ");
        error(errno,tempfile," is still intact",0);
    }
    else
    {
        wait(&waitstats);
        if(waitstats>0)
            error(waitstats,"Rename Returned With Error";"",0);
    }
}

```

ConVC

TEXT FILE UTILITY

by Andy DePue

Here's a powerful utility to add to your OS9 bag of tools. While ConVC was written with the idea to convert C programs from other platforms (Unix, MSDos) to OS9 convention, it will also come in handy to for text file applications. ConVC allows you to strip linefeeds, convert TABS characters to spaces and may also be used in a pipe as a filter.

ConVC has built-in help, just type: "convc -?" to better explain it's options.

ConVC was written by Andy DePue who writes our "C Notes" column.

```

/*
** Convc v2.0 - A utility to convert tabs and/or strip line feeds from a file
** Copyright (C) by Andy DePue - April 23, 1992
*/

#include <stdio.h>
#include <ctype.h>

#define TABCHAR 9 /* Tab Character */
#define DEFLF 0x0A /* Default Line Feed Character */
#define DEFTABSZ 4 /* Default Tab Size */

/* Define Globals */
int lfchar; /* Line feed character = 0 for no line feed stripping */
int newlf; /* If lfchar>0 and newlf>0 then the program will replace */
/* lfchar with newlf */
int tabsize; /* Size (in spaces) of a tab */
int tabflag; /* Convert Tabs? */

/* help() - prints help screen and exits */
help()
{
    printf("Convc v2.0 - Copyright (C) 1992, by Andy DePue\n");
    printf("Description:\n");
    printf(" Convc is a general purpose ASCII text conversion utility\n");
    printf(" for OS-9. It will convert tabs and/or linefeeds. Convc\n");
    printf(" stands for \"Convert C\" because the main reason I\n");
    printf(" programmed this was to convert ASCII C files from other\n");
    printf(" platforms to OS-9, but it can convert any ASCII file just\n");
    printf(" as well.\n\n");
    printf("Usage:\n");
    printf(" Convc <Opts> [Opts... ..]\n");
    printf("Where \"Opts\" can be any of the following:\n");
    printf(" -? This help screen.\n");
    printf(" -N Do not convert Line Feeds.\n");
    printf(" -C Do not convert Tabs.\n");
    printf(" -L=### This will cause the program to treat ASCII\n");
    printf(" character '###' as the Line Feed. Where\n");
    printf(" '###' is a decimal number representing any\n");
    printf(" ASCII character. Default is 10 ($0A)\n");
    printf(" -T=### This is sets the tab size. Default is 4\n");
    printf(" -R=### This will replace the line feed character\n");
    printf(" with '###' instead of stripping the LF out\n");
    printf(" (### being a Decimal number representing\n");
    printf(" any ASCII character)\n");
    printf(" -F This will cause Convc to act as a filter\n");
    printf(" (converting from stdout to stdin) so that\n");
    printf(" you can use pipes..etc.. No reporting is\n");
    printf(" done with this option and program is exited\n");
    printf(" once finished.\n");
    printf(" filename where 'filename' is the name of a file to be\n");
    printf(" converted.\n");
    printf("\nExample:\n");
    printf(" Convc -t=5r=13 Test1.TXT Test2.TXT -r=0 Test3.TXT\n");
    printf("\n This will convert the files 'Test1.TXT' and 'Test2.TXT'\n");
    printf(" with a tab setting of 5 and it will replace all linefeeds\n");
    printf(" with a carriage return (#13). It will then convert\n");
    printf(" 'Test3.TXT' by stripping out all line feeds (r=0).\n");
    exit(0);
}

```

```

/*
** Main(argc,argv) - This routine reads the parameter line and sets up
** the proper global flags.
*/

main(argc,argv)
int argc;
char *argv[];
{
    int linecount; /* Number of LFs program stripped/replaced */
    int tabcount; /* Number of TABS program converted */
    int count; /* General purpose counter */
    int filecount; /* Number of files processed */
    char *ptr; /* Command line pointer */
    char tmpfile[80]; /* Temp filename */
    char tstring[2]; /* Temp string */
    FILE *path1; /* Orig. File */
    FILE *path2; /* New File */

    lfchar=DEFLF; /* Setup LF to default value */
    tabsize=DEFTABSZ; /* Setup tabsize to default */
    tabflag=1;
    newlf=filecount=0;

    tstring[1]='\0';

    /* Check for no parameters */
    if(argc<2)
        help(); /* If no parameters, display help screen */

    /* Parse Command Line */
    for(count=1;count<argc;count++)
        if(argv[count][0]!='-') /* Check for no option */
        {
            strcpy(tmpfile,"temp."); /* Setup temp file */
            strcat(tmpfile,argv[count]);
            /* Attempt to open original file */
            if((path1=fopen(argv[count],"r"))==NULL)
            {
                error(errno,"Cannot Open File: ",argv[count],0);
                continue;
            }
            /* Attempt to create temp file - Check for existence first */
            else if((path2=fopen(tmpfile,"r"))==NULL)
            {
                /* Temp does not exist, so create it */
                if((path2=fopen(tmpfile,"w"))==NULL)
                {
                    fclose(path1);
                    error(errno,"Cannot Create Temp File: ",tmpfile,0);
                    continue;
                }
            }
            /* If temp file exists, skip conversion for this file */
            else
            {
                fclose(path1);
                error(errno,"Temp File Already Exists! Skipping file: ",
                    argv[count],0);
                continue;
            }
            convfile(path1,path2,&linecount,&tabcount);
            fclose(path1);
            fclose(path2);
            docopy(argv[count],tmpfile); /* Delete Orig. and Rename New */
            /* Print Info on File */
            printf("Replaced %d tabs",tabcount);
            if(lfchar!=NULL)
            {
                if(newlf!=NULL)
                    printf(" and replaced ");
                else
                    printf(" and removed ");
                printf("%d linefeeds",linecount);
            }
            printf(" from file '%s'...\n",argv[count]);
            filecount++;
        }
}

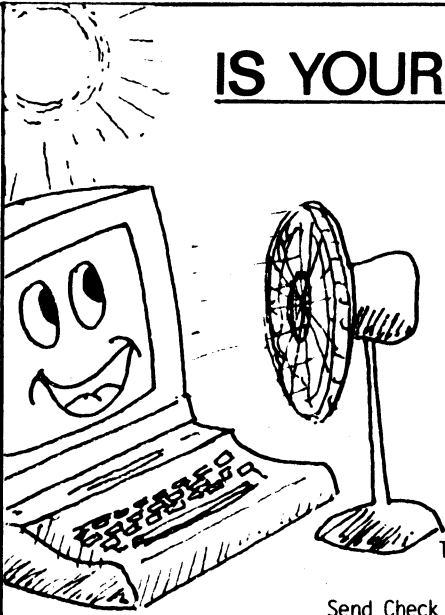
```

```

else /* If line does begin with '-' then process as option */
{
    ptr=argv[count]; /* Point to current parameter */
    ptr++; /* Skip past '-' */
    if(!*ptr) help(); /* If no option, show help */
    do
    {
        switch(toupper(*ptr)) /* Now process option */
        {
            case 'N':
                lfchar=NULL;
                break;
            case 'L':
                lfchar=getdig(&ptr);
                break;
            case 'R':
                newlf=getdig(&ptr);
                break;
            case 'T':
                tabsize=getdig(&ptr);
                break;
            case 'C':
                tabflag=0;
                break;
            case 'P':
                convfile(stdin,stdout,&linecount,&tabcount);
                exit(0);
            case '?':
                help();
                break;
            default:
                tstring[0]=*ptr;
                error(0,"Unknown Option: ",tstring,0);
                break;
        }
    } while(++ptr); /* Continue Processing Options */

    printf("Done.\n");
    printf("Processed %d file%s\n",filecount,filecount==1?"":"s");
}
}

```



IS YOUR CoCo Cool?

Did you know that there's 's about 120-150 DEGREES coming out of the top vents of your Coco on a 75 DEGREE day!! Now that summer is just around the corner you can keep your COCO COOL!

Installs internally to do the most efficient cooling possible. NO cutting, NO hacking, NO soldering! fan installs in less than 15 minutes! Runs off the CoCo's power supply with no harm to your Computer. The Extra-Small fan is very quiet. Good for all CoCo's 1-2-3 4K-1MEG.

To get your CoCoCool Fan for just \$29.95 (Plus \$3.00 S&H, \$5.00 Can)

Send Check or MO to: Dan Allen
7560 Woodman Pl. Ste# G-14
Van Nuys, CA 91405

For more info call: (818) 781-6573

AWESOME BOOTFILE EDITOR!

KWIKGEN v1.01

Still using OS9Gen, Cobbler, or Config? Now create boot disks in MUCH less time!

EzGen v1.09 *vs.* **KwikGen v1.01**
5 minutes 40 secs. **44 SECONDS!***

*Identical operations on identical fragmented boot disks
— 2 deletes and 1 insert performed by both utilities

- Editing done in memory
- Load boot from disk or memory
- Patch modules
- Change order of modules in seconds
- Make multiple boot disks in one session
- Edit existing boot files in place easily
- Load kernel from disk or mem and write to disk

You'll Experience GALE FORCE Speed!



Send check or money order to:
GALE FORCE ENTERPRISES

P.O. Box 66036, Stn. F, Vancouver,
B.C., Canada V5N 5L4

Checks: allow 4-6 weeks for delivery.
Money orders: processed immediately for
KWIK delivery

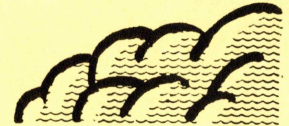


•Introductory price:
\$19.95 U.S.
(+\$4.00 shipping and handling)

(604) 522-6922

AniMajik Productions

NEW!
CLOUD_09 - by Albert P. Marsh



The BEST Graphics/Animation Editor for the CoCo! Tools include: Line, Box, Ellipse, Fill, Pencil, Brush, Flow, Spray, Text, FatBlox, Palette. Work with up to 8 animation pages. Copy one page to another. Complete control of animation speed. Edit/Save/Load VEF picture files.

Req: OS9 Level 2, Multi-Vue and 512K 34.95

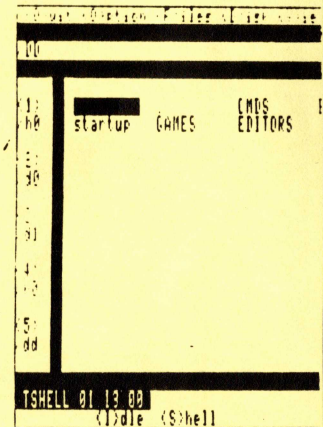
TShell V3.13.02 - by Paul Pollock

A revolutionary New Program... "TShell" does most of what Multi-Vue does at up to 5 times the speed! TShell will run most programs with one keypress and use standard MV AIF files. Delete, Copy, Rename files all with 1 or 2 keystrokes! Many utilities included.

WINDINT and Multi-Vue NOT required!
Req: OS9 Level 2 512K

TShell (with printed manual) \$39.95

TShell (with "printer-ready" manual, TDoc included to make the job easy!)
You print it... you save! \$29.95



SunDialer "WarGames" Dialer - by John Powers
(Includes versions for ACIA and SACIA)
Req: OS9 Level 2 and 512K \$19.95

DCom - Basic09 Decompiler - by Wayne Campbell
Req: OS9 Level 2, 512K \$24.95

Coming Soon!

TAKENOTE
COCO TYCOON V2.0
COCODEX
MEMMATCH
StG Net V4

(Prices Subject to Change without Notice)

Send Checks or M.O.'s
4650 Cahuenga Blvd. Ste #7
Toluca Lake, Ca. 91602

(818)
761-4135

Best Wishes CoCoFest!

Long Live OS9!

From our advertisers...

delmar Co.

Gale Force Enterprises

Sub-Etha Software

DCS Banners and Signs

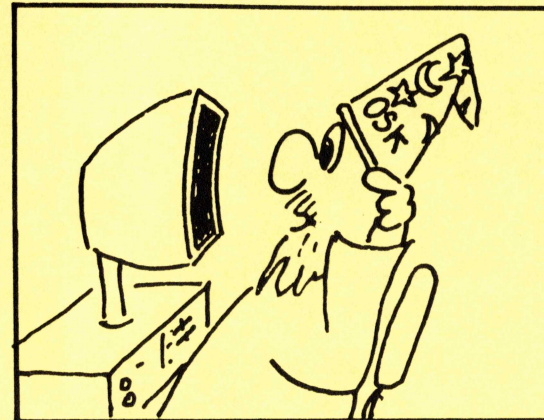
Sirius Software

AniMajik Productions

OS-9 Users Group
and the

OS9 Underground

WIZARD OF OSK® by Alan Sheltra



⌘ we're sorry ...
All available
memory is in
use right now...
Please execute
your program
later!

MS.

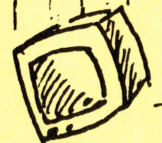
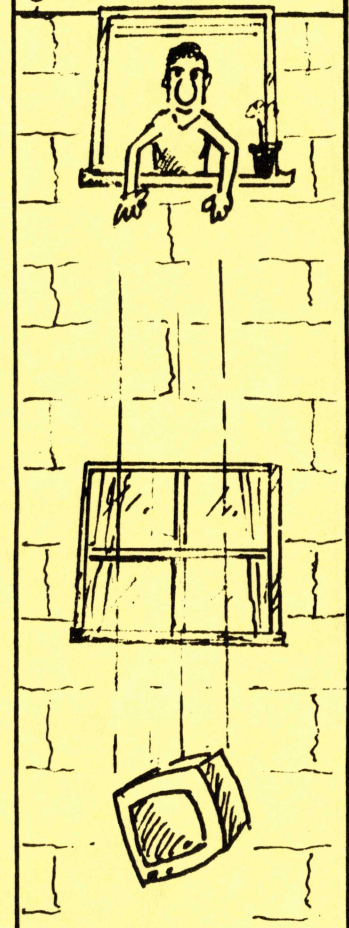
SHELL GAME by Alan Sheltra



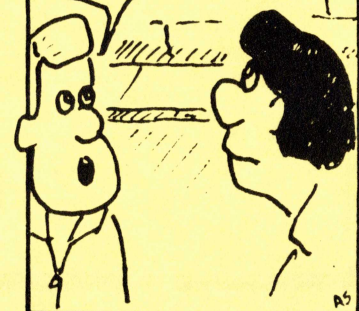
S92
AS

Smedley still doesn't quite grasp the
idea of forking a process.

QUIK BITS



NO BIG DEAL...
..JACK'S JUST DOIN'
ANOTHER ONE OF
HIS SCREEN DUMPS.



AS