OS-9 Newsletter

Volume II No.9 <<< BELLINGHAM OS9 USERS GROUP >>> September 30, 1991

Basic09 Part-I

A Tutorial by Scott Honaker and Rodger Alexander

IN THIS ISSUE

For those of us who were raised on our coco using Extended Basic (referred to as "Spaghetti Basic" by professional programers), we find ourselves a little intimidated by BASIC09 or worse, PASCAL. The structured format looks so formal, so "outlined", and Oh My Gosh...NO LINE NUMBERS! Yet, it is possible to run most standard Extended Basic programs with little or no modification.

I suppose the most offending thing about BASIC09 is that every variable needs to be, or should be defined and/or dimensioned. Other than that (and the way the printer is addressed), BASIC09 is not so different from Extended Basic. If we can just overcome and understand the way BASIC09 handles variables differently then what we are accustomed to, then we should be able to MOVE UP to BASIC09.

DATA TYPES:

In EXTENDED BASIC we have only two variables to work with: STRING variables, where we can store words (A$ = "HELLO"), and REAL NUMBER variables where we store numbers (A = 10).

How many articles have you read about different ways to write variable commands to save memory? After all, REAL NUMBERS may be super accurate, but they are also major memory wasters. In EXTENDED BASIC a simple integer is stored as a mantissa and an exponent (remember your algebra?). For accuracy purpose, the number "1" is stored out to 38 places (1 x 10 to the 38th power or 1 = 1.00000000000000000000000000000000000000). Get the picture? Although EXTENDED BASIC can store a REAL number into only 5 bytes of memory, the number "1" is only one bit of information that could easily be stored in only one byte.

BASIC09, on the other hand, is much more efficient in it's method of data storage by giving you 5 different types of data storage:

Simple Type	Allowable Values	Memory
BYTE	Whole numbers (0-255)	One byte
INTEGER	Whole numbers (-32768 to 32768)	Two bytes
REAL	Floating point (±1*10^±38)	Five bytes
STRING	Letters, digits, punctuation	One byte/char.
BOOLEAN	TRUE or FALSE	One byte

REAL is the default data type if none is specified.

SUBSCRIPTION INFORMATION

The OS-9 Newsletter is compOiled and printed monthly by the Bellingham OS-9 Users Group. Subscription rates are $5 for 6 months or $10 for 12 months. Mail your subscription check to:

Rodger Alexander
3404 Illinois Lane
Bellingham, WA. 98226

BENEFITS TO SUBSCRIPTION MEMBERS

As a subscribing member of the Bellingham OS9 Users Group, you enjoy the following benefits:

1. Monthly Newsletter
2. Public Domain Library (6-Megs)
3. Technical assistance (734-5806)
4. FREE Classified Ads
5. On-Line OS-9 Conference (676-5787)

Complex Data Types:

In addition to the five standard data types above, you can create your own data type(s) using the TYPE command. In the example below we want to define a collection of simple data types to be combined to form database records.

Name	Length	Contents	Type
Name	25	Employee name	String
Street	20	Street address	String
City	10	City of address	String
Zip	--	Address Zip Code	Integer
Sex	--	True = male, False = female	Boolean
Age	--	Employee age	Byte

You can combine all these variables into one complex data type by defining the variables with a TYPE command line (See below). This will create a new BASIC09 data type called "EMPLOYEE"

```
TYPE Employee=Name:STRING[25]; Street:STRING[20];
City:STRING[10]; Zip:REAL; Sex:BOOLEAN; Age:BYTE
```

Now we can define variables to use "EMPLOYEE". The following program line defines "EMPLOYEE" as a 10 element array named "Worker".

```
DIM Worker(10):Employee

Worker(1) .Name="Scott Honaker"
Worker(1) .Street="12740 NE 10th PL #E205"
Worker(1) .City="Bellevue"
Worker(1) .Zip=98005
Worker(1) .Sex=TRUE
Worker(1) .Age=27

Worker(2) .Name="Barbara Alexander"
Worker(2) .Street="3404 Illinois Lane"
Worker(2) .City="Bellingham"
Worker(2) .Zip="98226"
Worker(2) .Sex=FALSE
Worker(2) .Age=42
```

PASSING DATA VARIABLES:

A really neat thing about BASIC09 is that a program can consist of a bunch of "mini" programs or procedures, but data variables are local to each procedure (mini-program) in which they are defined. A variable defined in one procedure has no meaning in another procedure unless you use the RUN with PARAM statements to pass the variable when you call another procedure. Check out the example below:

```
PROCEDURE Demo
PARAM value:STRING[10]
DIM length:INTEGER
length=LEN(value)
RUN Parse(value, length)
PRINT "The result is: ";value
END

PROCEDURE Parse
PARAM temp(10):BYTE; length:INTEGER
DIM index:INTEGER
FOR index=1 to length
        temp(index)=temp(index)+1
NEXT index
END
```

From OS-9 type: **RUN DEMO("HAL")** The program should print: "The result is:IBM"

Next month we will take the "EMPLOYEE" data type that we defined above to create a simple database. It's easier than you might think. MUCH, MUCH EASIER THAN EXTENDED (SPAGHETTI) BASIC!

NOTE: For further information about BASIC09 data types, read chapter 6 in the BASIC09 section of your OS-9 Level Two Manual.

Simple Switch for you Hi-Res Interface

by T.Warren/Seattle 68xxxMUG

Isn't it aggravating every time you have to switch between "HI" and "LOW" resolution joystick. It sure would make computer life easier if programs gave you a choice of either LOW or HI resolution joystick operation (Actually some programs/games do!). Better yet....a simple switch to access either HI or LOW resolution operation without disabling the casette port.

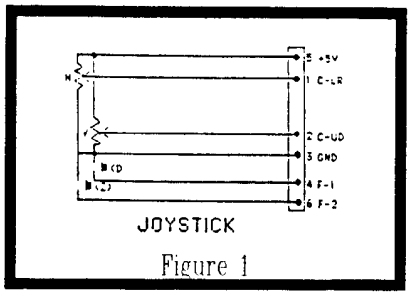

Figure 1

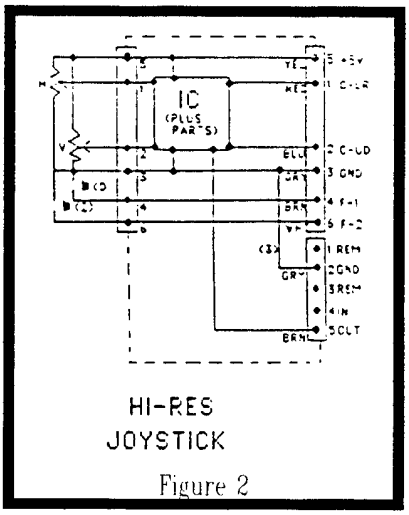

Figure 2

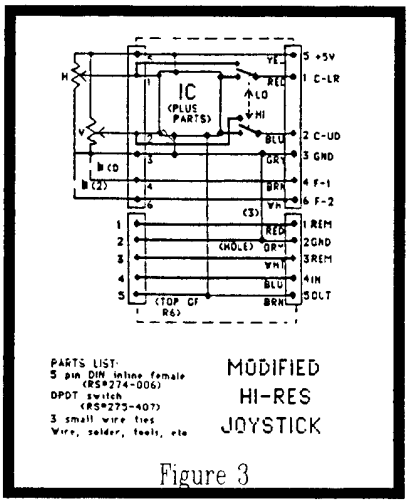

Figure 3

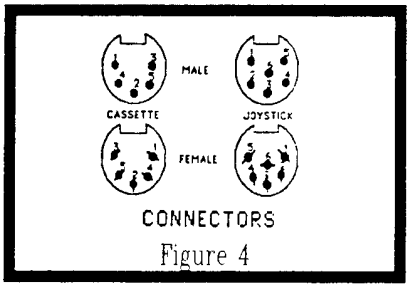

Figure 4

MODIFCATION THEORY:

Take a look at figure 1 to see the standard joystick wiring to the color computer and compare it to figure 2, which is the High Resolution Joystick connection to the color computer. Notice that pin 1 and pin 2 are interrupted by the integrated circuit (IC) in the HI-RES Interface. By simply switching the IC in and out of the circuit it's possible to have both LOW and HI resolution operation by the flip of a switch.

Also notice in figure 2 that the 5-pin jack is the casette port and that pin 5 is monitored by the IC in the hi-res interface. It's not necessary to disconnect the casette tape recorder during hi-res operation.

Simply put, we want to modify the Hi-Resolution Joystick Interface to switch between LO and HI resolution operation. This can be accomplished by using a double-pole, double-throw mini switch (Radio Shack Cat.# 275-407) mounted on the top of the interface box. The switch will route pin 1 (C-LR) and pin 2 (C-UD) of the joystick port around the Hi-Res joystick integrated circuit (IC). Also note that it is possible to continue the casette lines through the interface box with a 5 pin DIN inline female jack (Radio Shack Cat.# 274-006).

MODIFICATION INSTRUCTION:

1. Open the Hi-Res Interface box by removing the four retaining screws.
2. Cut the two plastic wire tie straps to give you easier access to the individual wires.
3. Unsolder the red wire (C-LR) from the circuit board and solder it to one of the center connectors on the mini slide switch.
4. Solder a 2 to 3 inch length of insulated wire to the same hole in the circuit board from which you removed the red wire in step 3 above. Solder the other end of the wire to one of the end terminals on the mini slide switch on the same side that the red wire is connected to. (See figure 5)
5. Unsolder the blue wire (C-UD) from the circuit board and solder it to the other center connector on the mini slide switch.
6. Solder another 2 to 3 inch length of insulated wire to the same hole in the circuit board from which you removed the blue wire in step 5 above. Solder the other end of the wire to the vacant end terminal on the mini slide switch next to the other wire in step 4 above. (See figure 5)
7. Using an ohm meter or continuity checker, find the pin on the integrated circuit (IC) that is connected to pin 1 on the joystick socket mounted on the circuit board. Solder a 2 to 3 inch length of insulated wire to the identified pin on the IC, then solder the other end of the wire to the remaining end terminal on the mini slide switch on the same side as the red wire referred to in step 2 above. (See figure 5)
8. Using an ohm meter or continuity checker, find the pin on the IC that is connected to pin 2 on the joystick socket mounted on the circuit board. Solder a 2 to 3 inch length of insulated wire to the identified pin on the IC, then solder the other end of the wire to the remaining end terminal on the mini slide

switch on the same side as the blue wire referred to in step 3 above. (See figure 5)

9. Cut an appropriate length of 5 conductor cable and solder one end of the cable to a 5 pin DIN inline female jack. File an appropriate size whole in the Hi-Res Interface box 1/4 inch to either the left or right side of the large joystick jack hole. (See figure 7)

10. Using an ohm meter or continuity checker identify where pin 1 of the casette jack is soldered to the circuit board then solder the pin 1 wire from your female DIN jack to the same point on the circuit board.

11. Repeat the procedure in step 10 above to connect the 4 remaining wires from your female DIN jack to the corresponding connection points on the circuit board so that you have continuity from the male casette jack that plugs into the color computer and the female DIN jack.

12. Secure the original casette and joystick cables with two small wire ties in their original holes. Drill a third tie down whole to line up your new casette cable with the hole you filed in the box in step 9 above. Secure your new casette cable with a small wire tie down strap. (See figure 6)

13. Cut and drill appropriate size wholes in the top of the Hi-Resolution Interface box to accomodate the mini slide switch. Mount the switch and close up the box securing it with it's 4 retaining screws. (See figure 7)

THAT'S IT!

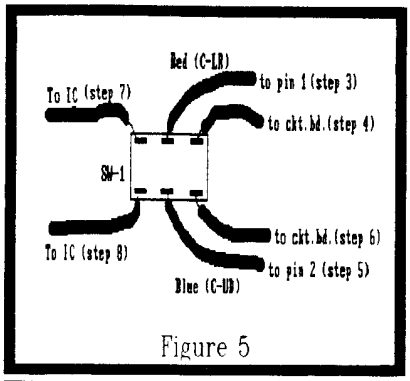

Figure 5

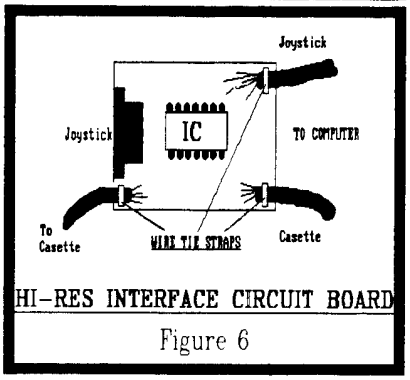

Figure 6

SOFTWARE SWITCHING:

It is possible to switch between HI and LOW resolution via software. This can be achieved by sampling the casette out (pin-5) and feed the signal to a two stage operation amplifier (OP AMP), then filter out the alternation current (audio) component. The resultant DC volatage (5VDC) can be used to turn on a relay to function as the switch. In fact it is possible to mount a "mini" 5 volt D.C. relay (low current) inside the interface box. See figure 8 for a schematic diagram of the software switch.

The software switch works because programs/games requiring the hi-resolution interface send out an audio signal through the casette port. When the audio signal is present, the relay closes switching in the hi-res IC. When the audio signal stops, the relay opens and the joystick is returned to low-resolution.

PARTS LIST

5 pin DIN inline female plug
(Radio Shack Cat.# 274-006)
DPTD mini slide switch
(Radio Shack Cat.# 275-407)
3 small wire ties
Wire, Solder, Tools, etc.

NOTE: If you plan to build the software switch, specific parts are listed on the schematic diagram.

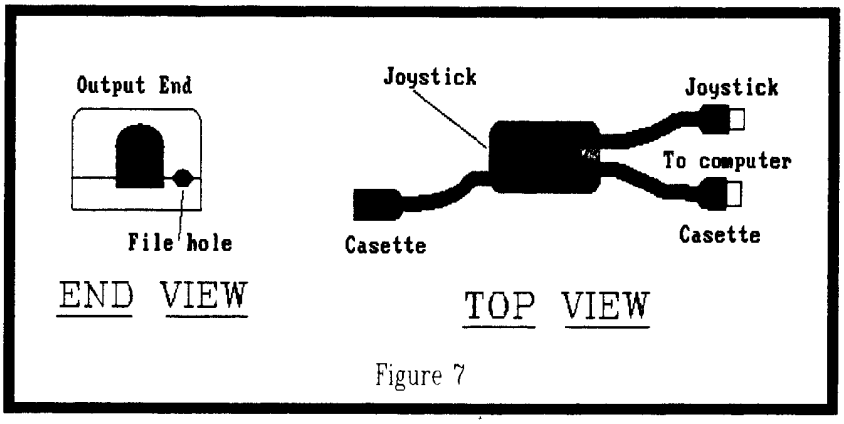

Figure 7

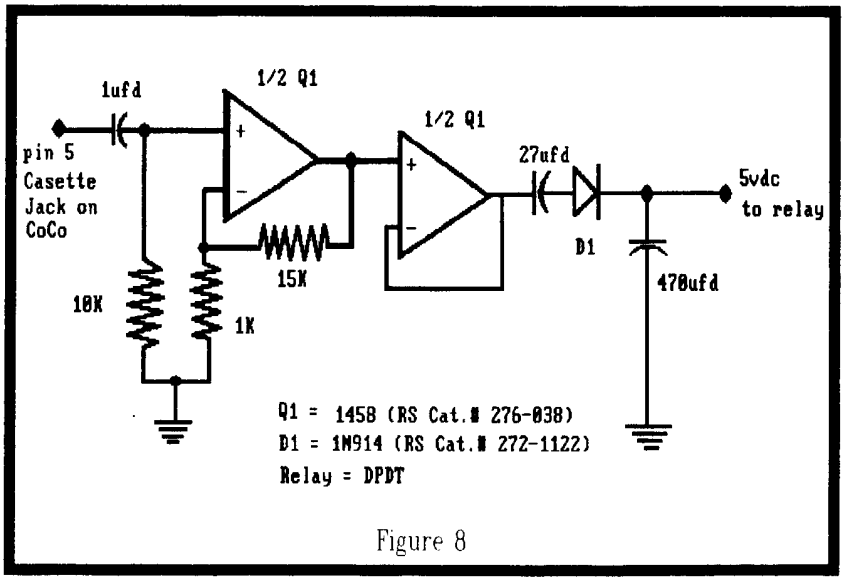

Figure 8

Using MODPATCH to "patch" OS-9

Modifying executable modules is not nearly as difficult as you might think. OS-9 Level Two comes with a command program called **MODPATCH** which simply reads a text file that you create with any text editor or even the lowly **BUILD** command. Your text file will tell **MODPATCH** to link to an executable module that is loaded into memory, go to specified locations, check the data at those specified locations and changes them to the desired data value, and finally verify the CRC (Check-sum) of the file. You can accomplish the same thing with the **DEBUG** utility but this seems to be less threatening.

Below are two Patches that you might consider for your OS-9 Level Two system. The first patch improves the performance of **DELDIR**, and the second patch will permit you to burn an eprom that will automatically boot OS-9 when you turn on your computer. Although you may not want to use the eprom idea, the patch to the **REL** module is suggested to correct some minor bugs.

DELDIR:

The following patch modifies **DELDIR** so that the utility will not perform a "DIR E" (extended directory) when it reads the directory. Since it doesn't display the directory, there is no point in performing an extended directory when a standard "DIR" will do just as well thank you. It still deletes multiple sub directories just as before but now it will do it faster.

Use the BUILD utility (or any text editor) to create the following **MODPATCH** file:

```
OS9: build deldirpatch
? l deldir
? c 0143 45 20
? v
? [ENTER]
```

To run the modpatch file, type in the following command lines:

```
load /dd/cmds/deldir
modpatch deldirpatch
del /dd/cmds/deldir
save /dd/cmds/deldir deldir
```

NOTE: If you don't have the **SAVE** command utility from the Level Two Development System Package, use the one supplied with OS-9 Level One.

BOOT OS-9 IN ROM by Mike Sweet

Putting OS-9 in ROM is fairly simple. Basically, there are two changes that need to be made. The first change fixes a bug in the REL module. That change can be made by simply running modpatch with the included script ('rel.pat.') Before you make the change, though, type:

```
ident -m rel
```

The output should look like this:

```
Header for: REL
Module size: $012A #298
Module CRC: $6FD34C (good)
Header parity: $D3
Edition: $05 #5
Ty/La At/Rv: $C1 $81
System mod, 6809 obj, re-en, R/O
```

If the output doesn't match this, DON'T TRY TO PATCH REL! Boot-up with another disk (you'll have to reset twice to start from RS-DOS again), and try ident again. If all else fails, boot-up with your backup of the original OS-9 system disk. That definitely should match.

Here's the **MODPATCH** file:

```
1 rel
00dd 30 33
c 00e0 fc 19
c 00e1 8c 11
c 00e2 ed 83
c 00e3 00 ed
c 00e4 24 00
c 00e5 1b 24
c 00e6 ce 1a
c 00e7 26 12
c 00e8 00 12
v
```

Once you have patched REL, cobbler to a new disk. Now, reset twice to return to RS-DOS and type in the following program:

```
10 CLEAR 500,&H3FFF
20 INPUT "INSERT OS-9 DISKETTE IN DRIVE 0 AND PRESS <ENTER>";A$
30 FOR A = 1 TO 18
40 DSKI$ 0 , 34 , A , A$ , B$
50 FOR M = 1 TO 128:POKE &H3EFF+A*256+M,ASC(MID$(A$,M,1))
60 POKE &H3F7F+A*256+M,ASC(MID$(B$,M,1))
70 NEXT M , A
80 POKE &H4000,68:POKE &H4001,75
90 INPUT "INSERT RS-DOS DISKETTE INTO DRIVE 0 AND PRESS <ENTER>";A$
100 SAVEM "OS9BOOT/BIN",&H4000,&H51FF,&H4002
```

Double check your typing, and then run the program. When done you will have a ML file on your diskette that can be loaded and burned onto an EPROM. In case you couldn't figure it out from the listing, the code resides in memory locations $4000 to $51FF (or &H4000 to &H51FF in BASIC.)

With the resulting EPROM installed in your disk controller, OS-9 will automatically boot from drive 0. If you want to boot directly from your hard drive, you will have to replace the floppy BOOT module with one that will read from your hard drive.

OS-9 Level Two "UPGRADE" *The FINAL Chapter !?*

by Kevin Darling

BACKGROUND

Apr-Sep 88 - prehistory, private work

That April, Kent Meyers and myself got wind that an update was coming and we began passing along any bugs we found to the CoCo guys at Tandy. Partly because Micro Ware was getting out of the 6809 by then, I got the job.

At the end of September, Micro Ware flew me up to Des Moines to get info needed for two projects at the same time: the Level-II update, and a port of something to the Atari ST. Both were supposed to be done by the end of that year; the plan was to first do all the known L-II bugfixes, and also add any enhancements possible within a few weeks. And that was all, at first.

In October, the ST project got cancelled. Otherwise, the L-II update would've been out way back then, and yet you'd have spent $20-30 for nothing (in comparison to what you've gotten free off the nets since).

Actually we'd be worse off than that, as most serious work would've stopped and there'd have been no fast **grfdrv**, new **gfx2**, **clock** patches, acia speedups or a dozen other neat giveaways which came along... much, much later on.

NEW ORIGINAL SCHEDULE

Oct-Dec 88 - new timeframe for official upgrade. Now we suddenly had some more time for L-II work (tho not much more), and both Micro Ware and Tandy gave carte blanche to add whatever we wanted. (Applause!) Knowing that this update would also become the new stock L-II release, I wanted a lot in there <grin>, but needed some help doing so.

From that arose a special private section on the Compu Serve OS-9 forum donated by Wayne Day, wherein about two dozen volunteers wrote, donated and tested tons of new code (names later - remind me!!)... again with MicroWare and Tandy's blessing. This included rival third parties who worked together for the common good, modifying their own drivers to come into upgrade compliance. To say that all this was unique in the history of OS's, would be an obvious understatement!

Now, although no one at Tandy and Micro Ware ever said "Let's stop here" (and I did so a few weeks too late), there were times that a version could've been released.

END OF FIRST WORK

Jan 89 - original clean up work done. Okay, we could've stopped in January 1989. But that still would've left tons of possible stuff untouched. By this time we were also writing demo programs that went beyond what anyone else had tried, and discovered (as others have recently) that **Windint** had major problems with multiple overlays and menus.

I should've collected the money and run right then, but <sigh> I wouldn't have been able to face myself later if I had done so. Dumb? Maybe. Hmm. But remember, this was going to be the new stock L-II...

GOING BEYOND

Feb-Mar 89 - rewrite **Windint**, numerous major and minor fixes.

Apr-May 89 - **grfdrv** crunch and speedups applied, CLEAR problem solved. So we all continued, and I rewrote much of Windint. In the meantime, Kent was carving out huge chunks of room for me in **Grfdrv**, and that let me create those **Grfdrv** speedups you have now. We also were able to find a lot of new and really obscure/rare bugs, and fix them. Best of all, I also worked out a simple and effective fix for the irq stoppage when changing Multi-Vue screens (and you already have that too! included in Bruce Isted's IBM mouse patch).

Thus the end of May 1989 was the next best stopping point, and in fact that's pretty much the version I myself still run today. The volunteer work was still ongoing tho... and something else happened: re-

member the RainbowFest when CerComp's "Window Master - you don't have to be an OS-9 rocket scientist to have overlapping windows" RSDOS program came out? (Kinda miffed me.) "HA!", I gritted my teeth, "you wanna see easy-to-use overlapping windows?!"

GOING TOO FAR

Jun-Aug 89 - overlapping, moveable, resizeable windows, new **gfx2**. From June to the end of August, "super-windows" was written. After enormous work... 3-D look, moveable, resizeable, non-stopping, overlapping windows were crammed into the original 8K **Grfdrv** space. Still proud of that! You could even pick up windows and move them to other screens. Too neat.

Too neat. And useless. Great for demos and fun to play with (Dale Puckett loved 'em), but they took up slightly too much RAM, slowed down the covered windows, and compared to the full-screen apps we were used to: just useless. Overlapping windows make a heckuva lot more sense on a larger display, yah? Also, they were impossible on a 128K system, which was part of the specs. So I think they'll remain a curiousity piece forever. Or probably go in OSK. Anyway, I decided to yank those and put back in other, more needed, stuff. But y'all got much of the new **GFX2** from that part of the project, too!

SUDDEN DEATH

Sep 89 - coco cancelled: Without any warning (which would've caused us to immediately wrap up), the CoCo got cancelled... just as everyone involved was 99% done (I'm sure even the remaining Tandy CoCo guys were caught totally off guard).

Naturally, that officially killed the upgrade... and the first Catch-22 situation arose: Tandy would consider selling it through EOS, if it was first submitted to them completely done, doc'd, and packaged ready-to-sell.

But no one had the money to do that. (Cancelled project = no payment) And to do all that, without knowing for sure if it would be accepted? No way.

Plus, at exactly that moment, the new 68K machines were being born, which meant all our efforts had to turn in that future direction instead. Not long after, the CoCo EOS program was stopped. So the new upgrade got shelved. Well, not totally... bits and pieces have been posted the whole time since, and I think we all expect that sooner or later, it'll fully come out.

OTHER COMMENTS

We honed that sucker to a fare thee well! Absolutely the sweetest OS-9 version around... smooth, fast, 99.999% bugfree, crunched down code. Honestly, OSK sometimes feels clunky in comparison... I think others agree. (Note: we'd change some things nowadays, because the 128K limit is gone)

Again, you already have more stuff posted than the original spec called for. Anything after that has been gravy. Free gravy, too.

THE PEOPLE, AND WHAT WE PROVED

Even tho the upgrade got abruptly cancelled, I believe we easily proved that there's a lot of talent out here that MW can and should make use of... not only by farming out paying projects, but even down to using volunteers to check over any of their current code (it's amazing what a third person can find and fix, or at least optimize).

[*Incredibly, many of the volunteers are normally well paid professionals (at least two coders were people Micro Ware had failed to entice to Des Moines :-) I'd guesstimate at least .25 million dollars worth of time was donated willingly, just to help out the project. I love these guys!!*]

Another good point, Who better to do an update, than the people who have to write programs and drivers for and/or use that system daily for personal use?

I must also note the unbelievable secrecy, which was held for well over a year (until Tandy said it was okay to talk) by everyone. Sure, all involved signed nondisclosure agreements, but still it was unprecedented. I didn't even see secrets held that well when I did NSA-related work.

Another thing that was proved beyond a shadow of a doubt: telecommuting is viable! You couldn't find a more scattered group of people on this continent, yet the project progressed smoothly and in coordination (in some part due to the Compu Serve forum space and lots of phone calls). The most astonishing thing was that even while they were being independently innovative, not one person ever balked at my telling them that a certain method was unsuitable... instead they just used such rules as a baseline to do even more superlative work.

So whether the "full" upgrade comes out or not, all the volunteers deserve the utmost praise and high marks for their work.

How to get Deskmate 3 to work on a Hard Drive-

TOOLS:

First, let me say that certain tools are necessary for a person to do any job, be it building a bird house, digging a ditch or modifying a program. The tools you need for this job are **dEd** or some other equally good disk editor with the capability to <F>ind a byte or ascii string and then find the <N>ext occurence of the string you want to change. Also, the disk editor must be able to <V>erify the module to give it a new CRC. Another tool you will need will be Burke&Burke's **EZGEN**. Buy it. Now. It has saved me about ten year's worth of hair loss! It will be one of the best investments you will make. The version of **dEd** to download is the one archived with the .ar extention not the .PAK extention. The text editor I use is **VEd** by Bob van der Poel. It is an excellent editor and takes up no more memory than the <list> utility when <list> is loaded by its self.

Directories:

In your root directory on your hard drive (which should be named /dd) you will need a CMDS directory and a DESK directory. I also have a directory called WORKROOM where I do such things as modify files, de-arc or un-PAK files, write Basic-09 and "C" programs, etc. and then move them to the proper directory later.

Copy all of the files on the DESKMATE disk to your /dd/WORK directory. After modifying them they will be placed in your /dd/CMDS directory or /dd/DESK directory.

VDG Window

Deskmate will ONLY work in a VDG WINDOW!!!! Make sure you have **vdgint** in your os9boot on your boot disk or DESKMATE will not work. If **vdgint** isn't in your bootfile then use **EZGEN** to put it there. Follow Chris Burke's directions. **EZGEN** is quite well documented. Also, there are some VDG device window discriptors available on some of the bulletin boards. I've got /v1, /v2, /v3, & /v4 in my boot file along with fifteen windint type window discriptors. If you can't find a VDG type window on a BBS or at a local CoCo or OS-9 club meeting then make one with the **wcreate** command or use the **xmode** command to make a type=1 pag=16 window.

To run DESKMATE:

After you have modified the files to run on a hard drive the order in which to type the commands to run DESKMATE in a previously defined /w4 would be:

```
deiniz /w4
xmode /w4 type=1 pag=16
iniz w4
shell i=/w4&

-OR-

iniz /v1
shell i=/v1&
```

and then use the <clear> key to get to /w4 (or /v1). When you get to the VDG window, enter the commands:

```
chd /dd/desk
desk
```

****

Modifying DESKMATE:

Some folks use a directory called SCRATCH, I call mine WORKROOM because I was taught that it is not polite to SCRATCH but it is always honorable to WORK. (My father

was a Republican!) In any case it is always best to do the modifications to files in as small a directory as possible. It should contain no distracting files, - ONLY the files you wish to modify. This way you don't accidentally miss a file and have your system crash and leave you pulling out already thinning hair! If you don't want the directory to remain on your disk that's fine. That's why MicroWare wrote the **DelDir** command.

OK. Now you have a /dd/DESK directory, a /dd/CMDS directory, a /dd/WORK directory (containing all the DESKMATE files), and have made provision for a VDG window by adding **vdgint** to your bootfile and EITHER adding the proper wcreate or xmode commands to your startup file OR by adding the /v1, /v2, /v3, & /v4 discriptors to your bootfile with **EZGEN**. You have **dEd** (or YOUR favorite disk editor) in your CMDS directory. You are now ready to go to work.

If you have 512K you may want to open several windows, - (1) to run **dEd** for the actual modification of the several modules of DESKMATE, (2) to keep a DIR of the directory you are using to modify the files, (3) for copying modified files to the /dd/DESK or /dd/CMDS directories, (4) for verifying that the files actually arrived at the correct directory, and (5) to use **VEd** or **MORE** to follow along in this file as you work. Don't you just love OS-9 L2's windows? Thank you Radio Shack and Microware!

You might as well load **dEd** into memory. If you have **VEd**, **MORE**, or another small text editor you may want to load it into memory for listing, scanning, or reading this document.

```
In window /w1 (80 col) type:
        chd /dd/WORK
        load dEd (* or other disk editor *)

In window /w2 (80 col) type:
        chd /dd/DOCS
        ved  hddesk.doc

In window /w3 (40 or 80 col) type:
        chd /dd/work
        dir

In window /w4 (40 or 80 col) type:
        tmode -pause
        chd /dd/cmds
        dir

In window /w5 type:
        chd /dd/DESK
```

Now we are ready to go. Look at the directory of DESKMATE files and find CONFIG.DESK. Go to /w1 and type:

```
        ded config.desk
```

When **dEd** loads the file your first command will be F for <F>ind then hit the <BREAK> key to shift to ascii mode and type /d0. **DEd** will then find /d0 and you hit the <E> key to enter the edit mode. Check to see that you are still in the ascii mode and type over the d0 with a **dd**. Use the arrow keys to treat all the /d0s like that and then, before leaving that screen, hit the W key to write the modification to disk. Now hit the enter key to get out of edit mode and hit the N key for the <N>ext occurance of /d0. Repeat the above procedure. When all occurrences of /d0 have been changed to /dd in the config.desk file you hit V whereupon **dEd** will automatically update the CRC. Your last command is Q .

```
Shift to /w5 (/dd/DESK) and type:
    copy /dd/work/config.desk config.desk
    copy config.desk config.desk.bak
    copy /dd/work/sample.FIL sample.FIL
    copy /dd/work/fish.DOC fish.DOC
    copy /dd/work/calendar.CAL calendar.CAL
    copy /dd/work/termstat.dmc termstat.dmc
    dir
```

This will give you a backup copy of your <config.desk> module. You will most likely trash the config.desk module once or twice in the process of learning how futile it is to try to run DESKMATE from anything other than a VDG window so it's a good idea to always keep a backup of your config.desk module around, just

in case. Now you have verified that all the necessary files are in the DESK directory and we can concentrate on the CMDS directory.

Back in /w1 type:

```
ded desk
```

When the module named DESK is loaded proceed as with config.desk. Just hit <F> for find, <break> to toggle hexidecimal/ascii mode (if you needed this document to help you, you will most likely need to use the ascii mode like I do). Change all occurrences of /D0 to /dd and don't forget to <W>rite the changes to disk before leaving the screen to look for more occurrences of /d0. Also, don't forget to <V>erify the CRC before <Q>uiting **dEd**.

Check with /w3 for the name of the next module to modify. Shift to /w5 and type:

```
chd /dd/CMDS
copy /dd/work/desk desk
```

Shift to /w4 and type:

```
<CTRL><A> (* repeats the DIR *)
```

Shift to /w3 to get the name of the next module to modify and repeat the last few steps. Continue until all modules (Except maybe DMDSKINI and DMBACKUP) have been modified. Decide for yourself if you want it to be that easy for someone to re-format your hard drive for you!

BE CAREFUL:

Be sure you are in ascii mode in **dEd** when <F>inding or <E>diting. Be sure you check all modules and help files. Some need changes and some don't. DON'T modify DMDSKINI or DMBACKUP by accident. If you really want the capability, go ahead, but be warned!

Read the right hand (ascii) portion around where you are editing (in **dEd**) to be sure what you are changing is actually a place where the program is calling /d0. There are places where there are ascii representations of machine code that look like d0 or D0 without the slash. Only change the place where there is a /d0 and if you decide not to change something, write it down and if there is an error displayed in DESKMATE as you use it then you'll know where to look. The places that need changes are actually easy to spot. And they are easy to find, too, with **dEd**.

One of the nice things about how OS-9 and Windows work together is when you get the hang of shifting to another window to use the <CTRL><A> combination to repeat a command very quickly. I would use this for verifying the copying of a file and also in /w5 to repeat the copy line. Then hold down the left arrow key to back over the name of the last module and type in the name of the current module to be copied. It takes less time to do it than it takes to tell about it. Any time you want to check yourself, you can shift over to /w2 to read the documentation.

When all modules are copied to the correct directories you may now run DESKMATE. The speed from a hard drive is miles ahead of running from floppies. DESKMATE gives the only affordable alternative to DYNACALC as a spreadsheet for OS-9. DESKMATE's LEDGER spreadsheet is a flavor of MicroSoft's MultiPlan so there are a good many books available on how to optimize it's features.

Now that you know how to patch DESKMATE, what else do you have that might yield to the same treatment?

MOTIVATION:

"Because I have known the torment of thirst, I would dig my well where others may drink." - Ernest Thompson Seton

CREDITS:

Most of the work in figuring out how to run DESKMATE on a hard drive was done by members of the 68xxxMUG of Seattle and the Bellingham OS-9 Users Group.

You may contact Bernie Besherse by writing to P.O. Box 9381, Ketchikan, AK 99901 or calling him direct at (907) 225-1324.

Club Activities

Mt. Rainier CoCo Club

John Schliep oversaw September's meeting at beautify facilities of Parkland Library. There were two major presentations this month. Alan Johnson took us back to an application that has been around for years. But it was not a walk down memory lane. Rather is was an illustration of a fine integrated program that can still fill many needs for the average user. The VIP Library for the CoCo III. Alan talked about two uses in particular. First the data base. It does most of the complicated work for you. In return you have to accept a few restrictions. The application creates the display screen and the field definition at the same time. This advantage is balanced by the disadvantage of not being able to add any fields after the definition process is completed. The advantage of being able to sort by any field is countered by needing one disk for each data base. (The size of each data base file is considerable.)

Alan had handouts of each stage of the process to show just how easy and yet sophisticated VIP Data can be. There were lots of questions and complete answers as the group toured though the screens and processes.

Then Alan illustrated how easy it is to create a merge in the work processing side of the package. His example was generating an invoice for shipment. Not only could it print out all the info from the data base, but also it did all the math for extensions, subtotal, sales tax, and total. It was impressive. Alan's clear presentation showed that VIP Library deserves lots of use by today's users.

The second presentation was by Randy XX. He has wanted to learn more about machine programing. His first attempt was no small task. He wanted to create a screen that looks amazingly like a word processor. It looked like Max 10. Now, this screen doesn't do anything . . . yet. But we have to take into account that doing it was a learning processes for Randy. And he probably will continue his project. You see, Randy is a professional programmer in that other environment sometime referred to as "messydos." Well, it's no wonder Randy gets confused at times!

Next month's meeting is October 8th at 7 p.m.

-- Donald Zimmerman --

Port O'CoCo Club

Our meetings are getting larger and more exciting. This month we had 15 people, three running systems and a great time! Our snacks were provided by our members and a LARGE supply of popcorn was contributed by Sound Sound Cinema 6. Since there were several new people we went around the room introducing ourselves, our systems, and our interest areas. New people were Mike Torell of Olympia, Don Anderson of Lacey, Vivian Jensen of Port Orchard, and Mark Kulien of Seattle. Clell Harmon is a recent addition to the group. He told us the new number for BBS Cloud's Corner 377-4290. This BBS is supporting CoCo RS-DOS and OS-9 for our area.

During the Kent Computer Swap Meet Gary Nelson of Federal Way stopped by and said he had several bags of CoCo "stuff" given to him by one of the stores he frequents. He had no need for the software so he gave it to Port O' CoCo to help it raise some funds. We sold the games and applications for $1 each. Such a Deal!

The Computer Swap Meet had been two days before. Again, it was well worth our time and effort. Terry Laraway brought his whole system to go along with my system. Tom Brooks and Chris Johnson handled many of the questions when either Terry or I wanted to take a break.

Back to the Monday meeting, after requesting a donation of $1 and adding in the sales of the software Tom Brooks gave us a financial report of about $315 in the cookie jar. Lastly in the area of publicity, we have delivered a supply of flyers of the computer groups in Kitsap County to the Kitsap Greeter Service. They will give this information, which also contains a partial list of BBSs in the area, to each new resident they meet. These flyers are already in the various Chambers of Commerce around the county.

We thanked Phyllis Young for all her efforts in getting notices in the newspapers and phoning everyone on our list. She also went beyond that and delivered notice sheets to several of the Radio Shack Stores in the area. She will be adding radio announcements to that list of notices for next month. Tom Brooks placed an ad in the "Little Nichel." We agreed to run the notice of our meeting for three months to see what the results are. Then we will evaluate again.

Chris Johnson recently found out that Microsoft grants $1,000 to any computer related group in the Puget Sound area. The catch for us is that the group needs to be incorporated. We discussed the idea of incorporating the four groups (Seattle, Tacoma, Bellingham, and Port Orchard) into one. Since our meeting I have talked with a paralegal who is willing to loan to us samples of all the paperwork needed. Filing with the State is $30 with an annual fee of $5.

With a need $$$ for a banner and incorporation, we need to find a way to raise some funds. I proposed selling a safety light home unit.

Called the 911 Lite, the small fixture screws into any light socket. When the light is turned on all is normal, but when the switch is turned on twice quickly the light flashes on and off. These units could be handled by the club and only 25-40 need be sold to pay for the banner and the incorporation. After a demonstration of the unit there was a great deal of interest.

Next month's meeting is October 21. Mark King will continue with his tutorial on C and Alan Johnson of the Tacoma group will talk about the VIP Library.

-- Donald Zimmerman --

Seattle 68xxxMUG

The September 3rd meeting featured T.Warren's hardware modification to the Hi-Res Joystick Interface. The modification consisty of a simple double-pole double-thro mini slide switch and a casette extension cable. (See his article in this issue).

The other half of the meeting was the 2nd part of Scott Honaker's Basic09 Tutorial in which the group, with Scott's guidance, created a Basic09 database.

-- Rodger Alexander --

Washington State BBS List

FAR POINT BBS - Seattle
RiBBS (Fido NET)
(206) 285-8335

COLUMBIA HTS. BBS - Longvie/Kelso
RiBBS (Fido NET)
(206) 425-5804

DATA WAREHOUSE BBS - Spokane
RiBBS (Fido NET)
(509) 325-6787

BARBEQUED RIBBS - Bellingham
PCBOARD (PC-NET)
(206) 676-5787 Conference #5

OS-9 Newsletter
3404 Illinois Lane
Bellingham, WA 98226