PROGRAMMING TRICKS

CONSOLE CONFIGURATOR

Dave Land (Computer Center, 5815 Johnson Dr., Mission KS 66202) sent us a few ideas he uses in his custom programming. The first one is a method for adapting programs to different consoles. It will work nicely for the library, if I get a chance to incorporate it into all the programs. Besides being a universal configurator, it shows a few neat things you can do that I hadn't thought of. The subroutine is listed on page 2.

To use, change the REM line below your console to a RETURN and then MERGE it to your program. You can MERGE it into several user programs at once by using DATASMITH's MULTIPLE MERGE. For example, to set up for a Hazeltine 1500, change line 30065 to RETURN. Left as is, it would configure for a SOL. From your program, execute a GOSUB 30000 once per program. Now, to execute "Home and Clear Screen":

```
PRINT O$(4);
```

To print on line number 10:

```
PRINT O$(1)+REPEAT$(O$(2),10);
```

In your next letter, Dave, tell me why you don't have a LENGTH or a RIGHT. If you have it, I would like to receive the same type routine for printers. I'd also like the members to send me the control characters for their consoles, if they aren't listed here.

ROUNDING WITH FMT

Also given was a method for "rounding" on reports. Dave defers rounding until the very last step, to preserve accuracy. He uses the format statement:

```
FMT(N+.005,"$$$,$$$V.99")
```

For samples of N, the printed results would be:

N	Output	
34556.455	$34,556.46	(Rounded Up)
12354.464	$12,354.46	(Rounded Down)
.985	$.99	
.064	$.06	

If you need the result to make pennies add up, you can get it with:

```
VAL(FMT(N+.005,"$$$,$$$V.99"))
```

SOFTWARE

REAL ESTATE PROGRAMS

Investment Analysis Systems is offering a comprehensive Real Estate Software package for residential, income and commercial property management and analysis. The package consists of three modules; PROPERTY MANAGEMENT SYSTEM II, PROPERTY ANALYSIS SYSTEM, and BUSINESS SUPPORT SOFTWARE.

(text continued on page 3)

```
30000     DIM O$(8,4):! CURSOR CONTROLS
30005     ! CONSOLE CONFIGURATIONS - 9/15/80
30010     ! CURSOR CONTROLS
30015     ! FOR HAZELTINE 1500
30020     O$(0)=CHAR$(7):! BELL
30025     O$(1)=CHAR$(126)+CHAR$(18):! HOME
30030     O$(2)=CHAR$(126)+CHAR$(11):! DOWN
30035     O$(3)=CHAR$(8):! LEFT
30040     O$(4)=CHAR$(126)+CHAR$(28):! HOME & CLR SCRN
30045     O$(5)="":! UP
30050     O$(6)=CHAR$(20):! DIM
30055     O$(7)=CHAR$(20):! BRIGHT
30060     O$(8)="79":! WIDTH
30065     REM
30070     ! FOR DYNABYTE NAKED TERMINAL
30075     O$(0)=CHAR$(7):! BELL
30080     O$(1)=CHAR$(27)+CHAR$(72):! HOME
30085     O$(2)=CHAR$(10):! DOWN
30090     O$(3)=CHAR$(8):! LEFT
30095     O$(4)=CHAR$(12):! HOME & CLR SCRN
30100     O$(5)="":! UP
30105     O$(6)="":! DIM
30110     O$(7)="":! BRIGHT
30115     O$(8)="79":! WIDTH
30120     REM
30125     ! FOR EXIDY SORCERER
30130     O$(0)="":! BELL
30135     O$(1)=CHAR$(17):! HOME
30140     O$(2)=CHAR$(26):! DOWN
30145     O$(3)=CHAR$(01):! LEFT
30150     O$(4)=CHAR$(126)+CHAR$(28):! HOME & CLR SCRN
30155     O$(5)="":! UP
30160     O$(6)="":! DIM
30165     O$(7)="":! BRIGHT
30170     O$(8)="63":! WIDTH
30175     REM
30180     ! FOR VECTOR MINDLESS TERMINAL
30185     O$(0)="":! BELL
30190     O$(1)=CHAR$(2):! HOME
30195     O$(2)=CHAR$(10):! DOWN
30200     O$(3)=CHAR$(8):! LEFT
30205     O$(4)=CHAR$(4):! HOME & CLR SCRN
30210     O$(5)="":! UP
30215     O$(6)=CHAR$(20):! DIM
30220     O$(7)=CHAR$(20):! BRIGHT
30225     O$(8)="79":! WIDTH
30230     REM
30235     ! FOR PROC TECH VDM (SOL)
30240     O$(0)="":! BELL
30245     O$(1)=CHAR$(14):! HOME
30250     O$(2)=CHAR$(26):! DOWN
30255     O$(3)=CHAR$(1):! LEFT
30260     O$(4)=CHAR$(11):! HOME & CLR SCRN
30265     O$(5)=CHAR$(23):! UP
30270     O$(6)="":! DIM
30275     O$(7)="":! BRIGHT
30280     O$(8)="63":! WIDTH
30285 <*  RETURN
```

The software was designed by real estate professionals and is menu driven and fully prompted for ease of operation. The software is available for S-100 computer systems with two 5 1/4" Micropolis drives, a CRT with cursor controls and 56K of memory.

Property Management System II (PMS II) automates most of the accounting required to manage and control all types of income property. The software has the flexibility to allow users to closely duplicate their methods of operation, but with better control, speed and accuracy.

PMS II consists of a complete General Ledger, A/R, and A/P optimized for property management and a Data Base Management system for tenant, vendor, and owner data files.

PMS II will manage up to eight properties per disk and provides complete formated reports; cash receipts, dispersments, balance sheets, budget analysis, operationg statements, chart of accounts, and tenant activity, information, and mailing labels.

PROPERTY ANALYSIS SYSTEM (PAS) analyzes all types of income properties for cash flow, tax benefit, return on investment and equity. All economic variables are considered; inflation, interest, income, vacancy factor, operating expenses, tax bracket, depreciation, etc., providing net operating income, net cash flow, equity growth, tax benefit, and return on investment and equity before and after taxes for 9 years. All variables can be changed, instantly showing the user its economic effect and then print at user's option. This program is ideal for modeling existing or future investments and making sales presentations.

BUSINESS SUPPORT SOFTWARE (BSS) consist of many useful programs that apply to both real estate and business; depreciation, loan amortization, future value of investment, etc.

Each software system can be run separately and is supplied in CBASIC2 object code on CP/M. The cost of each software package including manual is as follows:

PMS II	ver. 6.0	$695.00	($486 to MUG)
BSS	ver. 6.0	$65.00	($45 to MUG)
PAS	ver. 2.11	$245.00	($171 to MUG)

The PAS ver. 2.11 is also available in MDOS Basic for $245 ($171 to MUG).

A Manual and Demonstration disk is available for PMS II at $40.00 ($28/MUG) and for PAS at $35.00 ($25/MUG). Send $5.00 for Brochures and Sample Printouts, or the appropriate amount for each Demonstration or System module(s) desired. Calif. residents add 6% Sales Tax. Available from Investment Analysis Systems, P.O. Box 282, Palos Verdes CA. 90274 (213) 375-7784

REACT - REMINDER/ACTIVITY RECORDING SYSTEM

Did you forget to renew a license, pay a bill, clip a coupon, collect a debt or keep an appointment? Can you use some help in planning a project, scheduling your activities, balancing your workload and otherwise using time and dates to your advantage? Do you recall when you last had some maintenance work done on your car, when that washing machine was purchased, when the house was painted, the last time you visited your most important account and what you discussed?

The REACT system can help you in both these areas. It allows you to enter dates, activites, and reminders into a file and retrieve the data in many ways.

The system will permit a reminder or activity to be coded for a specific individual and retrieval of only that person's information can be requested. A classification

code can also be used so only data pertaining to a specific group of reminders or events can be requested. Both of these controls and a date, or range of dates, permit you to recall only the information you desire.

Automatic deletion of groups of data records, automatic file compression, machine-level sorting, complete prompting and error-trapping, and a complete instruction manual give the user a valuable software system.

The REACT system is designed to work in a minimum system of 48K with dual-drives. REACT can be used in single-drive systems at reduced capacity. A printer is useful but is not required.

REACT is supplied in Micropolis BASIC source code on MOD II media for $40.00 ($30.00 to MUG members). Available from Bonjoel Enterprises, P.O. Box 2180, Des Plains, IL 60018 (312) 297-2921

LETTERS
=======

ASSEMBLY LANGUAGE

MUGSY,
Your "gentle start" in Assembly Language (MUG newsletter #5) is excellent. We were gathering energy to make the plunge, and you've provided the right vehicle.

Our machine didn't like your line 220. Your input list calls for a "DT", although it's output as "DB". Your message line 200 differs from input to output, too, and, since we typed our input from your output, our count (line 50) was off. We're proud that we found those problems, or even tried. Congratulations.

For the initial learning process, we found "8080/8085 Software Design", by Titus, Rony, Larsen, and Titus, presents the instruction set in a helpful sequence. Then, for reference, we've been using "8080A/8085 Assembly Language Programming" by Leventhal. Both books have helpful additional material.

CP/M

We appreciate your reference to various program packages available for Micropolis BASIC. But, we keep hearing comments that CP/M is the common base language of the future. And, isn't Micropolis about to support CP/M. If that's the future, where does that lead MUG?

Jerry Factor
709 No. Palm Drive, Beverly Hills CA 90210

Jerry:
The best laid plans ... and all that. I checked and double-checked - and still missed the errors. The reason (but not an excuse) is that I re-typed both the LINEEDIT and ASSM output. "DB" (Define Byte) is correct for the numbers, while "DT" (Define Text) is for characters. Right about the record count, also. The LINEEDIT listing was correct. The ASSM listing is either missing an "L" in "EX-CEL-LENT" or else line 50 should read MVI C,61. The way to avoid the counting error (who likes to count characters anyway) is to make these changes:

```
050         MVI C,MESSEND-MESS
230 MESSEND END
```

Now the computer does the counting.

CP/M (Control Program for Microprocessors) isn't a language, as are BASIC, PASCAL, FORTRAN, COBAL, etc. It's an operating system - a structure and the code to support that structure. MDOS (Micropolis Disk Operating System) is equivalent in concept to CP/M, while Micropolis BASIC (which needs parts of MDOS) is equivalent in concept to CBASIC2 or BASIC-80 (which need CP/M). NO! ABSOLUTELY NO! Micropolis won't support CP/M or MPM. They will continue to support MDOS, and also OSM, which is the operating system for the Rigid Disk system. MDOS and Micropolis BASIC are upwards compatible to OSM and its BASIC. OSM is a multi-tasking, multi-user system - equivalent in concept to MPM.

I've said before and I'll say again, the Micropolis systems are technically superior to the CP/M-MPM systems. What CP/M has going for it is relatively universal acceptance. There are more languages and programs available for it, and for many different computer configurations and disk systems.

Where does that leave MUG? In pretty fair shape I'd say. If you want to run CP/M, you can. MUG doesn't intend to ignore it. But as we learn about MDOS, as we continue to converse with, and hopefully influence, software developers, you'll find that the MDOS based software will outperform CP/M software - especially in disk access.

To really go out on a limb, I'll predict that CP/M is about at its peak. As users become more sophisticated, CP/M's deficiencies become more evident. There will be a new "standard", perhaps based on the Bell Labs' UNIX system.

Even if a better operating system exists, I think one should always consider whether it is preferable (more productive) to be a novice with a potentially more powerful tool, or an expert with a less powerful tool. The object of the MUG is to make us all experts on the Micropolis system.

CP/M FILE SIZE

Buzz,
Concerning the CP/M "RUMORS" (MUG newsletter #6): I am no expert on CP/M (yet), but I did do some research concerning whether or not it knows how long a file is. According to the "File Control Block" format documentation, CP/M does know how long a file is, within certain limitations. Files are divided into "extents" of 128 records of 128 bytes each, and there can be a maximum of 16 "extents". Each "extent" knows how long it is, but it seems you must access the last "extent", find out how long it is and do some math to find out how long the whole file is. At the operation system level, there is nothing like the @RFILEINF routine in MDOS that tells you everything you need to know about a file. Also, MICROSOFT BASIC-80 has a function that returns the number of records in the current "extent" of a file being accessed, but nothing to tell you how many records total are in the file. As a practical matter, you must hunt for the end of file using CP/M.

It is interesting to note that the largest file supported under CP/M 1.4 is 256K bytes, which is less than a Micropolis Mod II disk capacity.

Another interesting tidbit is that IBM System/34 BASIC (the system/34 is a medium sized multi-user machine) doesn't know how long a disk file is either. The operating system knows, but IBM in their infinite wisdom does not provide this information to users of their BASIC.

LIBRARY ADDITIONS

I am enclosing a disk with a couple of programs to add to your library. One is a loan amortization schedule generator, useful to Mr. Guralnick, and the other is a tictactoe game. The tictactoe game requires a terminal with cursor addressing to

generate the X and O graphics. It is set up for a Vector Mindless Terminal, but if you examine it you will also find code for a Hazeltine 1500 included as comments. Tictactoe is one of the first programs I wrote for my TI-59 calculator, and later converted to Micropolis Basic. It is just dumb enough to let you beat it now and then.

VARIABLE ALLOCATION

The following is some information that may prove useful to anyone who does programming and has limited memory: As most of us know, Micropolis allows the use of the variable names A% to Z% for integer variable, A$ to Z$ for string variables, and A to Z plus A0 through Z9 for floating-point (real) variables. What Micropolis does not document is the way memory is allocated for the use of these variables. At first glance, it would appear that you could use any appropriate variable name you choose, but using some combinations of variables makes you pay a heavy price in memory used.

Memory allocation for variables works as follows: As soon as the interpreter sees its first variable, it allocates a whole alphabet worth of variables for the type encountered, 26 in all. There are 13 different alphabets possible: The integers, the strings, single letters, and ten alphabets for letters followed by a numeric digit. Therefore, in the program:

```
10 A=1
20 PRINT A
```

or in the program:

```
10 A=1
20 B=2
30 X=3
40 PRINT A,B,X
```

the exact same overhead is used for variable storage. When BASIC saw the first variable, A, it allocated memory for the variables A through Z so that all 26 letters could be used as variables without adding any additional storage in memory. No more memory was used when it encountered the B and X in the second example.

However, in the program:

```
10 A1=1
20 A2=2
30 A3=3
40 PRINT A1,A2,A3
```

the exact same function is performed as in example two above, but three times the memory is used. When BASIC saw the A1, it allocated space of A1 through Z1, when it saw A2, it allocated space for A2 through Z2, etc. We have allocated space for 78 variables (3*26), but we are only using three variables. Therefore, it is a clear advantage in memory used to allocate variables an alphabet at a time or to use dimensioned arrays for storage whenever possible. In the case of arrays, the only memory used is that required for the array. The program:

```
10 DIM A(2)
20 A(0)=1
30 A(1)=2
40 A(2)=3
50 PRINT A(0),A(1),A(2)
```

You can verify the memory used for all the above programs by entering them, typing RUN and then typing PRINT SPACELEFT. The one with the largest SPACELEFT uses the least memory.

FLOATING-POINT ROUTINES

If anyone has them, I would like to see Assembly-Language subroutines for floating-point math (+ - * /) that uses numbers stored in the Micropolis convention of type, mantissa, and characteristic in binary coded decimal.

Burks A. Smith
DATASMITH, P.O. Box 8036, Shawnee Mission KS 66208

Editor's Note-
This allocation of variables is what I was referring to last month in the discussion of Ed Burkhardt's GENSORT program. Ed used L0, L1, L2, L3 and L5, which generated an additional allocation of 130 variables. If you change them to A, B, D, E & F (not presently used), you save 650 words of memory (5*26*5wds/variable). Another 78 words can be saved if I% and O% are changed to real variables. I haven't been able to prove that there's any increase in speed when using integers, or any problem in using real numbers in an integer situation, such as loop variables.

Buzz,
As a new member of MUG, I enjoyed reading the back issues of your newsletter. The wide variety of discussions and reviews is certainly worth the $12/year membership fee.

I have a few comments and questions that I would like to share with you:

FMT FOR LEFT-FILL ZERO

First, I was very happy to read your review of Systemation's SORT/A in Newsletter #3. I had been thinking about purchasing this software package, and you convinced me that it is indeed a worthwhile investment. May I offer one suggestion? On page 4 of that newsletter you present the three lines of code:

```
B$=MID$(STR$(I),2,LEN(STR$(B))-2)
IF LEN (B$)<4 B$=REPEAT$("0",4-LEN(B$))+B$
Z$(I)=MID$(W$,72,5)+B$
```

This can be simplified to:

```
Z$(I)=MID$(W$,72,5)+FMT(I,"9999")
```

which will execute much faster. (Ed. Note - Indeed it does, Thanks.)

CRUNCH

Second, in Newsletter #4 you mentioned Systemation's CRUNCH utility almost in passing as you discussed the TX program. I think it deserves some special comments. CRUNCH is invaluable to the programmer who likes to heavily document (comment) his programs, but who is hindered by core size limitations and execution speed considerations. The abundant use of REM statements is important for good programming, if only to prevent hours of head scratching when a "bug" must be found in a program that was developed six months earlier. With CRUNCH a programmer can document all variables used in a program, indent FOR-NEXT loops, keep a running commentary at the right hand side of his listing, and generally code in a most shamefully core-wasting manner. His masterpiece can then be CRUNCHed into a miniscule amount of very efficiently-executed

code. (The last program I wrote was CRUNCHed from 32,695 bytes down to 6707 bytes.) Hats off to Systemation!

SAVE ACROSS PROGRAM LOADS

Finally, a question that I would like to throw open to other MUG members. Has anyone discovered a fast way to read a large quantity of values from a disk record directly into a vector? Specifically, consider the problem of trying to read 50 numbers from a disk record into vector V.

The direct method:

```
GET 3 V(1),V(2),V(3), ... ,V(49),V(50)
```

will not work because this statement will not fit within a 250-character program line. The disk record can be read in as a string and then parsed, but the Micropolis string functions needed for parsing are extremely slow to excute. I have had to resort to a solution of the form:

```
GET 3 A1,A2,A3, ... ,E9,E0
V(1)=A1:V(2)=A2:V(3)=A3: ...
V(49)=E9:V(50)=E0
```

This works, but it is cumbersome and uses up a lot of variables. Does anyone have a better idea?

Berill Mitchell
5524 Old Salt Lane, Agoura CA 01301

Alright, you guys. Wake up. The answer to the most asked question is about to be written. Lots of people have asked how to save across program loads. I really hadn't considered the possibility. As you can see from my numbers on the memory save, I don't understand all the details. But it works.

In general, the idea is to partition memory as though you were going to chain programs. I tried to set the start of the variables at B000H. The fourth variable in the SIZES statement is fussy, though, and it wouldn't accept what I wanted to put in. So, it turns out, for this program, that the T-array starts at BB40, the A-to-Z array starts at BD40, with I and R, two variables used in the program, at BD68 and BD95. All this shows if you RESET to your monitor and dump memory from B000.

I also accomplished the same idea without SIZEing. In that case I had to inspect memory for the start of the T-array. The start location will change, however, if you change anything in the program.

```
0010 SIZES (5,3,40,24000)
0020 dim t(99)
0030 PRINT "ENTER 1 FOR SET-UP"
0040 PRINT "ENTER 2 FOR LIST"
0050 PRINT "ENTER 3 FOR SAVE"
0060 PRINT "ENTER 4 FOR READ"
0070 INPUT R
0080 IF R=1 GOSUB 140
0090 IF R=2 GOSUB 180
0100 IF R=3 GOSUB 220
0110 IF R=4 GOSUB 240
0120 IF R=5 END
0130 GOTO 30
0140 FOR I=0 TO 99
0150 T(I)=I+100
0160 NEXT I
0170 RETURN
0180 FOR I=0 TO 99
0190 PRINT I;T(I)
0200 NEXT I
0210 RETURN
0220 SAVE "MEMB000" 16RB000,16RBF00
0230 RETURN
0240 LOAD "MEMB000"
0250 RETURN
```

There is no restriction, as far as I can see, on types or volume of data which can be saved in this manner. I'll play with it for a month and see if I, or some of you, can come up with some rules.

COPYRIGHTS

Buzz,
I have some observations about copyrighted programs. I can understand your reservation about us illegally sharing programs; I share that concern. On the other hand, many programs are available which require extensive adaptation to be used on our machines. I, for one, would be more than happy to buy the book (or whatever) from the copyrighted source so that the author could get his money. After that though, it seems we could share the programs on disk, thereby saving each individual having to adapt and type in the program. I have KADATH (an Adventure game) adapted on disk. The program comes from University Software's Fun & Games Vol. 1. It took me a week to chop up the program into manageable size files and enter it. Why couldn't someone else buy the book, then send to me for the adapted version? Everybody gains that way.

I also have COMBAT from the same book and some games from More Basic Computer Games. Tell me if this is "cricket."

I would be very interested in a strong chess program for 8080. Why hasn't someone adapted SARGON for 8080? I have one chess program, but it is weak.

KEYBOARD PORT

When I use IN(1) to scan the keyboard, how do I clear the port. If I am in a loop, when I come back to scan again it always reads the last data if nothing new has been pressed. I tried OUT, but no luck. What do I do?

VERSION 4.0 PDS

What's this Joe Callaway says about version 4.0 being shared? I still have version 3.0. I didn't have the extra $75 for the new, just after getting up and running. I would love to have a copy if it is legit to share it. I keep thinking they will come out with version 5, so why buy 4.0 now?

DOUBLE-SIDED DISKS

I guess you noticed that I use both sides of my disks on Mod I by cutting appropriate holes so that I can turn it over and use the back side. It works great for me.

Micheal Anders
P.O. Box 324, Clarendon AR 72029

Michael:
I have written several publishers requesting permission to reproduce their printed code in both the newsletter and the library. Phone conversations indicate that we will receive that permission. We must do it for each publisher, however, and have a letter on file that specifically defines the agreement. Members who are contemplating conversions such as you have done should send me the name of the publication and the publisher's name and address. I'll write them and print a list of "approved" sources in the newsletter.

Some I/O boards will let you clear the keyboard data port. Yours evidently isn't wired that way. Basic doesn't work fast enough for you to look at the status port.

Micropolis will NOT approve reproduction and distribution of their programs, in either load or text versions. You will have to buy Version 4.0 from Micropolis. The disk lists for $25, the manual for $50. There are no plans for a Version 5, although there is a 4.1 for the new double-sided drives. You might be able to get by without the manual if you have a little help by phone in configuring the system.

Micropolis also prefers that we not discuss the mechanisms of RES, MDOS and MpBasic, except for specific I/O configurations. Their justification is that people tend to alter the system, get it so it won't run, and then call Micropolis for help, which they obviously can't give. With due respect to Micropolis, I'm going to ignore their preferences. One of the primary aims of the MUG is to document the system. We should all remember, though, that anything appearing in these pages that doesn't appear in the Micropolis Manual, is just the MUG interpretation, not necessarily the facts. If the newsletter happens to print an article that says that such-and-such a portion of code does an operation in a particular way, don't go calling Micropolis to ask them why they did it that way. Their response will be "We didn't say that code did that function. We discuss only the code that is printed in the manual." I don't like the situation, but we'll live with it.

FIRST CLASS MAIL
===== ===== ====

FIRST CLASS MAIL
===== ===== ====

Published monthly by the MUG
Subscription (August through July) rates:
U.S., Canada, Mexico; $12/year: Other, $25/year
Mid-year subscribers receive current year's back issues.

- -

MICROPOLIS USERS GROUP
Buzz Rudow, Editor
604 Springwood Circle
Huntsville AL 35803
(205) 883-2621

FIRST CLASS MAIL
===== ===== ====

FIRST CLASS MAIL
===== ===== ====