TABLE OF CONTENTS

CHICAGO B128 USER'S GROUP -- INTERNATIONAL (CBUG, Inc.)
4102 N. Odell, Norridge, Il. 60634, U.S.A.

Bulk Rate
U.S. Postage
PAID
Desplaines, Il
60018
Permit # 296

!! ! ! ! ! ! ! ! ! ! ! ! ! ! !!
!! **POSTMASTER** !!
!! !!
!!**FORWARDING ADDRESS REQUESTED** !!
!! or !!
!! **FORM 3547 REQUESTED** !!
!! ! ! ! ! ! ! ! ! ! ! ! ! ! !!

DATED MATERIAL - DO NOT DELAY

THE CBUG ESCAPE

TWELTH ISSUE

Summer 1988

* *

THE CBUG ESCAPE is published 4 times a year*, more or less, by the Chicago B128 User's Group - International (CBUG, Inc.), an international membership organization in support of applications and usage of the Commodore B128 Computer. Note that some issues are combined in one publication.

AREA CODE CHANGE NOTICE: Effective in November 1989 (eleven months from now), our phone area code will become 708, the rest of the number will remain the same.

CHECK YOUR ADDRESS LABEL FOR EXPIRATION DATE. The expiration date is located at the top of the address label in the form of YYMM, eg. 8712 indicates an expiration at the end of the 1987 publishing year. CBUG will be unable to mail publications without renewal. **PLEASE KEEP YOUR MAILING ADDRESS CURRENT.** Renewal invoices will be mailed early Feb. 1989. PLEASE, PLEASE renew promptly!

CBUG is NOT affiliated or allied with any other organization, users' group, business or other entity of any kind, except in support of CBUG chapters.

Advertisements, articles and contents of disks are solely the responsibility of the individual authors. Their existence in a CBUG publication does not imply any endorsement by CBUG. Please report to CBUG exceptional performance, either pro or con.

Publishing address: CBUG, Inc. c/o Norman Deltzke, 4102 N. Odell, Norridge, Il. 60634 USA. 312-456-8720 7pm to 11pm CST.

Cover price this issue: $5.00. 1989 subscription rate: $14.00 (bulk rate, U.S. & possessions ONLY); $20.00 (first class, U.S. & possessions, Canada & Mexico); $21.00 (surface first class, all others); $35.00 (small packet rate air mail, any country).

* To be published 3 times per year beginning 1989.

* *

SCRATCH PAD

By: Norman Deltzke
Jan. 16, 1989

Here in Chicagoland, we have had several snows this year. Such weather for putting out a "Summer '88" issue. Oh well, atleast it is done. Along with this issue is a complete reiteration of normally available library disks -- a separate issue marked as Volume 12. With the Library directory now published, we will begin anew with library listings rather than to republish lengthy order forms every issue. Hang onto the directory!!

A significant issue before CBUG and all other groups is apathy. We are here as a service effort -- a mutual aid society. All things to all people is impossible. Like any community, all must contribute so that specific services are available. It is more important in our environment to ask for help than to receive it; else the helpers can not respond! It is not a matter of wondering when the group will do what you want, but what you can do to encourage the help you and your fellow members need.

I'd like to start a new "column" -- Requests from the Membership. You need not provide your name and phone if you prefer, or you can indicate that they are not for publication. BUT, allowing a direct response to your suggestion may get you a faster personal response as well as encourage a dialoge as to the question itself. Don't be bashful, even the simplist request is valuable in that it indicates areas of coverage both you and others likely need.

I submit that the only place consumer support in the world of computers is adequately available is thru groups such as ours; that with the inability of the manufacturers to counsel their customers as to newly released product they certainly can not assist customers with older product. As such, CBUG has become known the world over as a effective forum for successful group self aid!

The next issue will be dominated by all the hints and tips received by CBUG over the last year. If you have any discoveries, no matter how small or in hindsight just plain obvious, please send them in. Quickly if possible so we can get them in the next issue. The B128 and our software is fertile ground for discoveries as it still is one of the most powerful personal computers ever built; with limitless opportunities for expansion, adaptation and just plain creative use.

The issue of member discoveries is becoming one of greater importance. In this issue for example is an article from Gary Anderson regarding a technical bug. Infact, the bug is so elementary that those of us in the electronics engineering disciplines would, and often do, look for something more exotic. For hours, days, yea even months we pound our heads on the wall; then only to eventually find out that the problem had been previously identified by other members who simply did not publish their discoveries. **IMPORTANT - IMPORTANT - IMPORTANT:** Please everyone, if you discover something be it software or hardware, a bug or a solution -- or a suspicion or partial discovery. Please publish it! Submit to CBUG: paragraph sized items in typewritten form, larger on disk please in Superscript from. If you see someone else's quandry in The ESCAPE, contact them too!

The immediate future of THE ESCAPE. We are planning to adjust the 1989 publishing schedule to three issues for several reasons. Foremost is that I simply can not handle the work load in a timely fashion along with regular employment and other obligations. With rising costs the choice was to raise fees or reduce the schedule. While many of our authors have generously given hundreds of hours per year in articles and library materials, more is needed to properly produce at our current rate. Many issues have been delayed for want of adequate material. Another serious problem is that about 1/3 of renewals are belatedly sent in, often 4 to 6 months late. This creates a huge administrative problem in sending back issues, determining press runs, budgeting, etc. Please help us out. To the several hundred members who have renewed in advance for 1989, we thank you for your concern and anticipation.

In this issue Messrs. Jarvis and Springer are releasing the first of a series of revolutionary and first time published code. Much of these materials will be appearing in future issues of mainstream Commodore magazines such as Transactor. BUT, they were first here in THE CBUG ESCAPE. For those who are programmers, I'm sure you will be amazed beyond belief at these new materials.

Departing from our usual policy of not printing programming code, we have reproduced code from Faierson, Jarvis and Springer. All of these are in operating disk form on the current issues print file disk in the library section. It is probable that Faierson's Superbase program can be typed in from text as it does not appear to have any typesetting incompatable symbols; however the other two programs must be viewed on screen from disk for full accuracy.

Our friends in Cologne West Germany are making some significant progress. We hope to have substantial publishable materials from them in the upcoming months.

Progress is being made in the co-processor arena. Several members are reporting success in the efforts to more fully implement the 8088 co-processor board. With the advent of the Fast Bus software and hardware the transferance of those materials into the B128 is now practical. CP/M 86 is successfully functional; MSDOS may be just around the corner. Newer developments promise plain CP/M as well.

New compilers from Europe are in the pipeline along with Forth and Comal. Patience is the order of the day as everyone in this organization is a volunteer without compensation.

A word of good news for the few of our members interested: TPUG (The Toronto Pet User's Group) appears to have weathered their year long storm and is back functioning in a leaner tighter form. The Transactor magazine has changed ownership and management -- reportedly it is again publishing also.

<u>WORDS OF THANKS</u>

To our many contributors, helpers, mentors; to the wives, husbands and children who tollerate our long hours of devoted help to the unseen hoards of CBUG membership. Without your unstinting help CBUG could not exist.

A special thanks to my wife, Barbara and children, Bill and Jennifer who are volunteered on a near daily to handle to tasks of management and fulfilment within CBUG; maintaining the database, and duplicating and shipping library orders. Our children are even blessed with the joy of filing the blizard of paperwork generated by CBUG. Then too, grandparents and neighbor children who help with collating and preparing for mailing.

Lastly and most important those several members who contribute generously to nearly every issue.

IN MEMORIUM

EDISON L. RHYNER, January 2, 1989. Mr. Rhyner was a mainstay of the Chicago Area East local meeting and a frequent contributor to the CBUG effort. His expert telephone and programming help to numerous members will be sorely missed. He is survived by three children, seven grandchildren, three brothers and sisters and six nephews and cousins. Memorials to the American Cancer Society.

BASIC 4.0 (Extended) Tutorial

by: Warren. D. Swan

This is the last chapter of the abridged BASIC 4.0+ Tutorial. I hope you've learned how to more fully use your Commodore System. This final chapter deals with some of the more Commodore-specific features of your B, and some concepts that are needed for more advanced programming.

6 MISCELLANY

6.1 TIME$
[Miscellaneous 1]

One of the good features of Commodore computers is the ability to keep the time of day. The TIME$ variable (TI$ for short) is a predefined variable that can store the time. Unlike ST, DS, DS$, ER, and EL, TIME$ can be assigned a new value - as long as the value is a valid time. The TIME$ variable holds 7 characters as follows:

```
HHMMSST
```

The first 2 characters are the hours, the next 2 are the minutes, the next 2 are the seconds, and the final digit is the tenths of a second. When the computer is first turned on, the time is set to "0000000". Since it uses a 24 hour clock, we would set the time to 3:25 PM (1525 hours) with:

```
ti$ = "1525000
```

BASIC allows us to "be sloppy" about the tenths of a second, so we could leave them off:

```
ti$ = "152500
```

However, there must be at least 6 characters in the string. After performing the above assignment, we can view TI$ to see what time it is:

```
?ti$
```

and the time will have advanced accordingly. We can use TI$ to time portions of programs to see how long they take, to the nearest tenth of a second.

Be careful when referring to TI$ more than once. It may change between references. For example, this is sloppy:

```
t$ = left$(ti$,2)+":"+mid$(ti$,3,2)
```

The intent here was to get the hours and minutes from TI$ and assign them to T$ so as to make T$ in the form HH:MM. Most of the time this is OK. But what if TI$ was "2059599" (1 tenth second before 9 PM) when the LEFT$ was performed and then clicked over to "2100000" before the MID$ was performed. The result would be 20:00 or 8 PM! A whole hour off! The proper way to do this is:

```
t$ = ti$: t$ = left$(t$,2)+":"+mid$(t$,3,2)
```

Now given the same set of circumstances, T$ will be set to 20:59, or 1 minute to 9 PM. It is only a minute off, and we could not have helped that because T$ is only being set to the "current" minute. It is much better to ASSIGN TI$ TO SOME OTHER VARIABLE BEFORE SPLITTING UP THE TIME STRING.

When timing things it often becomes necessary to calculate elapsed time. Since TI$ is in the sexagesimal system (HHMMSS and tenths), we cannot simply subtract times to obtain elapsed time. The tutorial disk mentions the older Commodore systems' TI variable that was not used in the B series. It also has a subroutine to convert the TI$ variable into a TI variable containing elapsed time in 10ths of a second, which can then be used to time things.

6.2 Function Keys
[Miscellaneous 2]

B Series function keys are a great feature. When you first turn on the machine the function keys F1 to F10 have predefined values. However, they and the function keys F11 to F20 (shift F1 to shift F10) can be given new values with the KEY instruction. First of all, to see what the function keys are currently set to, just type the simple instruction:

```
key
```

Since you gave it no operands, it will simply list the values of all function keys defined. This may look like:

```
key 1,"print"
key 2,"list"
  ...
```

This means that when you press F1 it is as if you had typed print (without the quotes), and so on. If a function key contains a quote or a control character, BASIC will list it using the CHR$ function for these characters:

```
key 3,"dload"+chr$(34)
```

This means F3 will act as if you typed dload followed by a quote (CHR$(34)).

To change what a function key will do, simply type a KEY statement that looks like one of these listed by KEY:

```
key 20,"<home><24 cursor downs>
```

Of course, <home> means the HOME key, and <24 cursor downs> means you press the cursor down key 24 times. Now whenever you press F20 (shift F10), the cursor will move to the lower left corner of the screen. If you have to put a quote, ESCape, RETURN, INSert, DELete, or any other character that does not show in quotes mode (section 2.4), you must use the CHR$ function and the table in appendix D of the gray "Commodore B Series Users Guide".

The tutorial disk illustrates the setting of the function keys with an example that causes key 10 to behave like the SHIFT/RUN key.

It is BASIC's intention to list the function keys when you use KEY with no operands in such a way that you can modify a function key definition and press RETURN. Please note that BASIC sometimes gets confused listing function key definitions and ends up listing them in such a way that they cannot be redefined in this way. This is a BUG. It tends to happen whenever a function key definition (1) starts with something that must be CHR$ed, or (2) contains an ESCape (CHR$(27)) anywhere. These are not the only times it goofs, but they are most common.

6.3 Bit Manipulation with AND, OR, and NOT
[Handling Data 10]

This section is not for the casual BASIC programmer. An understanding of the binary number system is prerequisite.

The AND, OR, and NOT operators introduced in section 2.3 are actually bitwise logical operators. The operands to these operators must be able to be evaluated as signed 16-bit 2s complement numbers. That is, they must be able to be stored in integer variables. The number -1 has all bits on (1111111111111111). The number -32768 (lowest) is binary 1000000000000000, and 32767 (highest) is binary 0111111111111111. If non-integer numbers are used, the fractional part will be truncated. AND, OR and NOT actually operate on each bit of the 16 bit operands independently. Consider -25796 AND 13903, -25796 OR 13903, and NOT -25796:

```
 -25796:1001101100111100   -25796:1001101100111100
AND13903:0011011001001111  OR13903:0011011001001111 NOT-25796:1001101100111100
   4620:0001001000001100              -16513:1011111101111111
25795:0110010011000011
```

NOT is a 1s complement of the single operand. From a decimal standpoint, NOT A is equal to -1-A or -A-1.

BASIC's true value is -1 (all 1 bits) and its false value is 0 (all 0 bits). Be careful of mixing simple tests for non-zero without explicitly checking for <>0. In particular, (A AND 4) AND (B AND 8) will always produce 0, even if both parenthesized subexpressions evaluate to non-zero, because they will actually be 4 AND 8, which is 0. The correct test for non-zero here is:

```
(a and 4)<>0 and (b and 8)<>0
```

The <>0 will cause the bit tests to be "stretched" to all 1s or all 0s before doing the final AND. Notice that only one <>0 was actually needed here.

ASIDE: The ST variable is derived by taking the value in the operating system STATUS byte and sign extending it to 16 bits. That's why a "device not present" condition results in an ST value of -128, which is 1111111110000000.

6.4 Byte Manipulation with POKE, PEEK and BANK
[Miscellaneous 3]

BASIC allows us to "play with" bytes, as well as bits. The B machines are rather complex. They have 16 "banks" of memory, although not all banks actually have memory in them. The banks are numbered 0 to 15. In a raw B128 or B256, bank 0 is not used, nor are banks 5 to 14. Banks 3 and 4 are not used in a B128. Bank 15 contains: the ROM that contains the kernel and BASIC, input/output registers, the scratch pad RAM used by the kernel and BASIC, the system stack, and the screen RAM.

In order to access all these banks, BASIC maintains the concept of a "current bank". When the machine is first turned on, bank 15 is the current bank. You can change to a new bank with the BANK instruction, whose argument is an expression with a value from 0 to 15. To look at the current BASIC program, which resides in bank 1, use:

```
bank 1
```

To put something into the screen memory, use:

```
bank 15
```

Now that we can access banks, how do we actually access the bytes in those banks? To find out what is in a given byte location (once you've used the BANK instruction to get to the right bank), use the PEEK function:

```
bank 15: cc = peek(212): rem get cursor configuration into cc
```

The argument expression for PEEK must evaluate to a value from 0 to 65535.

To put a new value into a location in the current bank, use the POKE instruction:

```
poke 55296,10: poke 55297,peek(212): rem turn cursor on
```

The first operand expression must also evaluate to a value from 0 to 65535 and is the location which we wish to change. The second operand expression must evaluate to a value from 0 to 255 and is the new value to place in the location.

6.5 Tab Stops
[Miscellaneous 4]

The B machines allow you to set and clear tab stops for the screen. This can be used to aid inputting data, or it can be used to format data. The latter should be avoided because tab stops only work on the screen. If you later decide to print the same data on the printer, the tab stops will not work. Instead, the PRINT USING instruction should be used.

To set or clear a tab stop, press SHIFT TAB. If the column that the cursor is in was not a tab stop, this will make it a tab stop. If it was a tab stop, this will clear it. In other words, SHIFT TAB serves to toggle the tab stop status of the cursor's current position. There is no way to absolutely set or clear a tab. You can only toggle between set and cleared.

To move the cursor to the next set tab, just press the TAB key. If no tabs have been set or the cursor is on the last tab stop, the cursor will move to the rightmost column in the current line.

A quick procedure to clear all tab stops (useful when about to set new tab stops) is:

```
100 for x=1 to 80: print"<tab><shift tab>";: next: print
```

This will continually skip to the next tab stop and then clear it. Since we don't know how many tabs were set to begin with, we use the maximum, 80. If there were fewer than 80 tab stops, it will just keep the cursor at the right boundary, flipping its tab off and on. That's harmless since the right boundary has a permanently set tab stop regardless of SHIFT TAB there.

6.6 Recovering From Errors with TRAP and RESUME
[Program flow 9]

Usually when BASIC runs into a problem, it prints an error message and quits the running program. There are times when the programmer will want to handle the error condition without BASIC giving up. The TRAP instruction allows him to do that. The TRAP instruction can be used to turn on or off this ability for the program to handle errors. When the TRAP instruction is used with a line number, BASIC "remembers" the line number, and when an error does occur, it will transfer to this remembered line number instead of printing the error message or quitting. Here is an example:

```
100 trap 5000
```

The section of program at line 5000 that handles the error is called (logically enough) the ERROR HANDLER.

The programmer may only want to trap errors for a certain section of the main program. Outside of this section it may be desirable to let BASIC "handle" the error in its usual unforgiving way. To turn off the error handling, use the TRAP instruction with no line number:

```
300 trap
```

When "error trapping" is on, what can we do to handle the error? First we will want to know what caused the error. BASIC gives us two more special predefined variables, ER and EL. These variables only have meaning when used in the error handling routine. ER contains the error number as specified in Appendix J of the gray "Commodore B Series Users Guide" (pages 126 to 132). EL contains the line number where the error condition occurred.

In some cases we will only want to take care of certain errors and let others be treated as usual. For example, in an error handler at line 5000 we want to see if the error was caused by the statement:

```
1740 a = b/c
```

We will single out the problem caused by the value in C being 0 and will let A be set to 0, instead of the usual ratio of b to c. Here is the error handler:

```
5000 if (er=30 and el=1740) then a=0: resume next
5020 print err$(er)err$(13)el
5040 end: print err$(44): go to 5040
```

Line 5000 checks to see that the error was caused by a division by zero (error #30) and that it occurred at line 1740. If this is the case, then A is set to 0. The RESUME NEXT statement tells BASIC to resume with the statement

following the statement that caused the error (even if it's on the same line). BASIC's jumping to line 5000 when an error occurs is analogous to an unexpected GOSUB, and the RESUME instruction is analogous to a RETURN, with a few possibilities of where to return to.

If the error was not what we wanted to handle, it will continue to line 5020. The ERR$(x) function will give us the "error message" whose error number is the argument. ERR$(13) is a fancy way to print " in " (see Appendix K, ibid.), and EL is the line number where the error happened. So if the error was caused by executing a READ statement in line 1320 when there was no more DATA, line 5020 would print:

```
?out of data in  1320
```

and the program would quit at line 5040. If we try to continue with the CONT statement, line 5040 will also ensure that we get the

```
?cannot continue
```

message every time we try.

Of course the error handler can be quite complex if we wish. To keep it simple, it is better to have smaller error handlers and just use "TRAP line#" and "TRAP" pairs around small sections of program, with the line# being different each time. Then the error handlers don't have to be so complex. Incidentally, the line number operand for the TRAP instruction can be an expression, not just a constant.

We may not always want to resume with the statement following the error. In some cases we may want to retry the same statement with the conditions changed. In the above example, instead of simply setting A to 0, we may want to change C to a non-zero value (say 1000) and let A be the ratio of B to this new value of C. To do this we would change line 5000 to:

```
5000 if (er=30 and el=1740) then c=1000: resume
```

RESUME without the operand NEXT will cause BASIC to retry the statement that caused the error. Another prime use for retrying the same statement is if the error was error 0 (stop key detected) and BASIC was trying to do a BLOAD. Since it quit in the middle of the BLOAD, we would want to do it over.

Still another possibility is that we don't want to even resume in the same area of program where the error occurred. We may want to simply go back to a menu or something. In this case, we can NOT use a simple GOTO to resume at the desired location. Instead, we should use RESUME with a line number:

```
5100 resume 100 back to main menu
```

The line number here cannot be an expression, and it can be immediately followed by a comment (as in section 5.7). To summarize:

```
resume              retry statement with error
resume next         continue with statement after erroneous statement
resume line#        resume at a different part of the program
```

Two final notes about TRAP and RESUME:
(1) Not all errors can be trapped. For example, if a TRAP is in effect and the statement GOTO 1340 is executed and there is no line 1340, BASIC will terminate the program with the "undefined statement" error (#27) in spite of the TRAP. Some of the other errors cannot be TRAPped also. Try them before you try to TRAP them.
(2) RESUME NEXT has a bug. If the error occurred in the middle of a statement that changes the program flow (GOTO, IF/THEN/ELSE, etc.), RESUME NEXT will resume with the statement following the error (like it always does) RATHER THAN RESUMING WITH THE STATEMENT THAT WAS SUPPOSED TO BE EXECUTED NEXT. This usually only happens if the user presses the STOP key and the error handler decides to ignore it. In this case, RESUME (without NEXT) still isn't the proper answer because it may retry a statement that shouldn't be repeated (like a = a+1). Preferably you should turn off any TRAP in effect before any such program flow statements:

```
300 trap 9000: get c$: trap: if (c$="") goto 300
```

If you forget what the error numbers are, use:

```
1000 print err$(19)
1010 for e=0 to 18: print e,err$(e): next
1020 for e=20 to 44: print e,err$(e): next
```

(Don't let line 1000 give you a heart attack.)

6.7 DISPOSE
[Miscellaneous 5]

The DISPOSE instructions solves a problem that has always existed in Commodore BASIC (as well as other Microsoft BASICs). The DISPOSE instruction does one of two things, depending upon its operand.

DISPOSE FOR is used to toss away the currently active FOR/NEXT loop. If, for example, we want to find out how many elements of the array B are already in the array A we could use:

```
1000 for b=1 to nb: rem for each element in array b up to nb elements
1010 for a=1 to na: rem check through the na elements of array a
1020 if b(b)=a(a) then n = n + 1: go to 1040
1030 next
1040 next b
```

Line 1020 goes to line 1040 after counting the match, because we don't want to see HOW MANY TIMES an element in B is in A, but rather IF it is. When this happens, BASIC sees the NEXT B and decides that we must have finished the inner loop (FOR A/NEXT), and now want to do the next iteration of the FOR B/NEXT loop. Had we left off the B in this next statement, BASIC would have assumed that we were still continuing the inner loop.

In this case we had no trouble. However, there are times when leaving a loop unfinished can cause trouble later on. A better way to program the above is:

```
1000 for b=1 to nb
1010 for a=1 to na
1020 if b(b) = a(a) then n = n + 1: dispose for: go to 1040
1030 next
1040 next
```

The DISPOSE FOR tells BASIC explicitly that we want to leave that inner loop. You can use more than one DISPOSE FOR to exit several levels of FOR/NEXT loops. Notice that we don't need the B after the NEXT in line 1040, since BASIC now knows that we can only be talking about the FOR B loop when line 1040 is executed. Either the inner loop finished (element B(B) not found in array A), or the DISPOSE FOR "got rid of" the inner loop.

The tutorial disk discusses the other use of the DISPOSE command: DISPOSE GOSUB.

DISPOSE FOR is very valuable and recommended. Use of DISPOSE GOSUB is very handy in some situations, but can become obscure if abused

6.8 Loading & Saving Memory
[Miscellaneous 6]

There are some situations where we would like to load or save "stuff" in memory that is not normal BASIC programs. To save an area of memory that is not a BASIC program, use

the Binary SAVE, or Byte Save, command, BSAVE. This is a BASIC 4.0 disk instruction that will allow operands to be specified in any order. A typical BSAVE looks like:

```
bsave "filename",p1024 to p2048
```

Since we didn't specify which bank to work with, it will save a section of memory from the currently selected bank (see section 6.4). This BSAVE will save locations 1024 to 2047 (decimal) in the file "filename". Note that the last location it saves is ONE LESS THAN THAT SPECIFIED AFTER THE TO. The P parameters are Physical addresses. In this case, the entire

```
p1024 to p2048
```

is considered one operand, the ADDRESS RANGE. The individual components of the address range could not be specified in any order. However, the entire address range operand could have been placed before the file name:

```
bsave p1024 to p2048,"filename"
```

The bsave instruction can take the usual D and U operands for drive and disk unit. And like the DSAVE command, the filename must start with an @ if an existing file of the same name is to be replaced. Unlike the POKE and PEEK instructions, which have to use the current bank, the BSAVE instruction allows you to specify the bank to save from in the command:

```
bsave "@filename",b2,d1,p65000
```

Since the TO part of the address range isn't specified, it will save from location 65000 through the last byte (location 65535) in bank 2 (b2). This does NOT change the "current bank" to bank 2 - it will still be whatever the last BANK instruction set it to. The entire address range operand can be left off, but the result will be to create a file with no bytes saved. Since the P values in the address range operand can only be from 0 to 65535, the only way to include the last byte of the bank is to leave off the TO part. That is, "to p65535" will save up to location 65534, not 65535.

The BLOAD is the inverse of the BSAVE instruction. It loads in sections of memory saved with BSAVE. It too takes the usual file name, D and U operands and the B operand to indicate which bank to load into. Without the B operand it will load into the current bank.

The BSAVEd file will contain the location where the memory should be loaded into, in other words, what the first P parameter was in the BSAVE. Then when the file is BLOADed, it will be loaded into the same locations it was BSAVEd from. You can override this behavior and specify a different location for the BLOAD to load into using a STARTING ADDRESS P operand:

```
bload"/graph pak",b15,d1,p2048
```

The file will be loaded into locations 2048 on, in spite of where it was saved from.

6.9 More Output Formatting Trivia
[Miscellaneous 7]

The PUDEF instruction modifies how the PRINT USING instruction will behave. It was not presented with PRINT USING because it is more obscure.

The PUDEF instruction takes a single string operand, which of course may be a string expression. Only the first 4 characters of the string are used by PUDEF. Each character tells PRINT USING a character to print in a certain situation. If the operand has fewer than 4 characters, PUDEF will assume that we only wish to modify the characters that are specified.

PUDEF only changes which characters PRINT USING will OUTPUT in the situations given below. It in no way changes how a format string should be constructed.

(1) The first character tells PRINT USING which character to print when right justifying a number. This character is printed as the padding to the left and is the space if not PUDEFed otherwise. Example:

```
print using "#####.##"; 23.7
   23.70
pudef "!
print using "#####.##"; 23.7
!!!23.70
```

Strings will still be padded with spaces.

(2) The second character tells PRINT USING which character to print as a digit separator and is a comma if not PUDEFed otherwise:

```
print using "###,###.##"; 1234.567
  1,234.57
pudef " #
print using "###,###.##"; 1234.567
  1#234.57
```

(3) The third character tells PRINT USING which character to print as the decimal point and is a period if not PUDEFed otherwise:

```
pudef " .,
print using "###,###.##"; 1234.567
  1.234,57
```

This is the way numbers are printed in the UK, and (I suspect) is one of the prime purposes for the PUDEF instruction.

(4) The fourth character tells PRINT USING which character to print as the monetary symbol and is a dollar sign if not PUDEFed otherwise:

```
print using "#$####.##"; 23.98
   $23.98
pudef "*,.%
print using "#$####.##"; 23.98
***%23.98
```

Please note that if a $ is not preceded by at least one # in a format field, the $ is not treated as a special character, and indeed is not even considered a part of the format field. Such a dollar sign will print as a dollar sign regardless of PUDEF:

```
pudef " ,.@
print using "$####.#"; 23.98
$  24.0
```

The $ in the format string is NOT part of the ####.# format field because it does not have a # in front of it. The $ is treated as a non-special character and printed as is.

B Series users in England can use " .,<British pound>" (where British pound is the SHIFT BACK-ARROW character) to cause PRINT USING to print in familiar format.

The PUDEF modifications will stay in effect until the next PUDEF changes them, or until the machine is turned off then on.

6.10 Using Machine Language Routines
[Miscellaneous 8]

The SYS instruction and USR(x) function would be better covered in a separate machine language course. To summarize them, SYS causes BASIC to jump to a machine language routine, in the current bank, whose address is the value of the expression operand to SYS:

```
100 count = 2048: rem location of count m/l routine in bank 15
 ...
 20 bank 15: sys count
```

To use the USR(x) function, an address must be placed in bytes 3 and 4 of bank 15. When the USR(x) function is encountered in an expression, the argument expression is evaluated first, the result is placed in the "floating point accumulator" at locations 113 to 118 in bank 15, then a machine language JMP instruction is executed to the location whose address is in bytes 3 and 4. The user supplied routine must place its result (if any) in the floating point accumulator before executing an RTS to return to BASIC. BASIC will use the value in the floating point accumulator as the value for USR(expression).

6.11 Waiting For An Event
[Miscellaneous 9]

The BASIC WAIT instruction can be used to wait until some I/O operation or other event goes to a desired state. The WAIT instruction takes 2 or 3 numeric operands. The first is the memory location in the current bank to repeatedly check. The contents of this location will be ANDed (see section 6.3) with the value of the second operand expression, which must be from 0 to 255. If the result is 0, the ANDing is repeated until the result is non-zero, after which it will simply continue on to the next statement.

For example, location 209 in bank 15 counts the number of characters in the keyboard queue - an interrupt driven queue. We can wait until the user types a key with:

```
100 kc = 209
  ...
1040 wait kc,255
```

The value 255 ensures that if ANY bit in location 209 is set, there will be a 1 in the corresponding AND-result bit; thus causing a non-zero result and the wait will terminate.

If a third numeric operand expression is used with the WAIT instruction, it too must be a value from 0 to 255. This value is Exclusive ORed with the contents of the location BEFORE the AND with the second operand. For example, we could wait until there have been an even number of characters typed (including none) before continuing:

```
1040 wait kc,1,1
```

If an odd number of characters have been typed, say 3, the result of the Exclusive OR with 1 will have the least significant bit (LSB) cleared:

```
              00000011 = number of characters
Exclusive OR  00000001 = third operand of WAIT
              00000010 = result
```

When this is ANDed with the second operand, 1, the result is 0, so it will wait. As soon as an even number of characters have been typed (or none), the Exclusive OR will have a 1 bit as the LSB, which when ANDed with the 1 will result in a 1, so the WAIT will terminate.

* * *

This concludes the BASIC Tutorial for Commodore BASIC 4.0+ as found on the B Series Machines. Now, be sure to read the manuals for the various peripherals. Review the gray "Commodore B Series Users Guide." It describes other features about the B that are NOT limited to BASIC, such as the ESCape sequences. In other words, you should seek to discover the circumferential features of your machine that are not inherent features of BASIC and thus not covered in this tutorial.

Happy Computing!

NWM'S INVENTORY PROGRAM 2.0 UPDATE REVIEW

by: Bob Loeffler

WELL, HE DID IT AGAIN! - Bruce Faierson, of Northwest Music, decided to "massage" his excellent inventory program and has smoothed out some minor wrinkles and added more "bells & whistles".

The biggest improvement is dual indexing. With the former versions you could only access a record by the item description. Naturally, it was your choice if you elected to use descriptive nomenclature or a stock number in that field. But whatever you typed in, that would be the only way you could pull up a record for whatever the reason.

For a hardware store, using the stock number as the item description would be totally impractical. We must have a loose-leaf print-out of every item in inventory, with the item descriptions in alphabetized sections (using the long report function, which, incidently, now has greatly streamlined access procedures) so everyone working in the store can have immediate access to this information. Although we are affiliated with the OUR OWN Hardware Company out of Burnsville MN, we also use three other major wholesalers. Each company has their own stock number/catalog location scheme. In the past, if we needed to reference something by a part number, we might have to use all three catalog systems to match a part number with a description.

Now, with Bruce's update, when you enter a new record, the system will ask you for an item description and when completed, will ask you for a stock number and when completed, will bring up the remaining fields to fill in, such as retail price, cost, etc.. From that point on, whenever you have to call an item up (merchandise received, issued, etc.) you can use either the item description or the stock number.

New record entry has been improved. After entering the item description and stock number, the system checks them to see if there is an existing record and lets you know up front if there is one, instead of waiting until all information has been typed in and then tells you a duplicate exists when you try to store the record.

And, as in the prior version, if you are less than 100% accurate, the system will notify you that it can't find the item and will ask if you want it to search (for the closest match it can find). At this point something new has been added to version 2.0. When the search is completed, if the item it lands on is not the item you want and scrolling up or down in that area does not reveal it, and if, at that point, you realise that you had never entered the item into the system, instead of having to jump out of the Merchandise Issued/Received mode and go to the Enter function, you have the option to select the Enter function without leaving the mode you are currently in. You may also enter a new item from these modes without going through the search routine by typing /RET. This is a "one shot" function and /RET has to be typed for each new item entered.

That's the good news - now we have some bad news - (but not to worry, that bad news will result in more good news). The bad news is that whatever your disk drive record capacity is, it will be halved. So our 8050 record capacity dropped from 3000 to 1500 records. Bruce and I tossed this issue back and forth several times since I will eventually have close to 3000 records or more on disk and this would mean we would have to split up our records on to two or more data disks. We have close to 10,000 items in the store but are using the B128 strictly for warehouse (basement) inventory control - it would be impractical for us to use it as a point-of-sale device since it does not have cash drawer capabilities. In a high-turn, multiple purchase per customer situation, operating the B128 plus a cash register would be too time consuming. We have other methods of

controlling inventory in the sales floor area.

Anyway, Bruce informed me there was no practical solution to the problem and the more I thought about having to split my records onto multiple disks, the less I saw it as anything other than a minor inconvenience.

Now the good news. Operating with split disks (I'm currently using two, for items A thru M and N thru Z) has proven to be extremely beneficial.

#1. I find that when I do my re-order reports on Monday afternoons, I start with my A - M disk and in less than half the time (the new 2.0 is faster), compared to all records on a single disk, I have a printed report I can be working with, while the N - Z disk is printing. The same holds true when I print my alphabetized inventory updates using the long inventory report function - while the N - Z disk is printing, I can be replacing the A - M pages.

#2 - Several weeks ago, while working on inventory, we had half a dozen power drop-outs in 3 or 4 seconds and it trashed my A - M disk. Using our hand amended print-outs (we usually do not have time to run to the computer every time one or two items are taken out of warehouse stock) and the latest backup disk, I was able to bring it up to date in two hours instead of four hours if everything had been on one disk.

For someone wanting to upgrade from a prior NWM Inventory program to the current 2.0 version, NWM supplies all necessary software to covert your existing records to the new dual indexing system and also split your records onto multiple disks if necessary. I won't go into detail here since full documentation is supplied.

But here we go again with the good news/bad news stuff. The bad news is that this conversion process is extremely slow due to the length of time it takes the software to convert each item, so allow at least a good portion of a working day to complete the conversion if you have anywhere near 3000 records on a single disk.

The good news is that you will find it will clean up your records. Originally, as we entered new records we thought we were being very careful not to duplicate items of the same stock number due to entering them twice using different item descriptions. When the conversion software is run, it checks each item to see if an identical stock number exists. If so, it prompts you to enter a different number (you cannot delete a duplicate record at this point). Much to our amazement, it found 142 duplications and in some cases, triplications. It was a lot of work when the new software and data disks were up and running, to go in and delete these duplications, but now we know that our inventory data is absolutely "clean".

A second benefit resulted from cleaning up the duplicate stock numbers. We had been noticing that quite frequently, when our orders arrived every Thursday, we would receive items that we still had in stock. Because of duplicate stock numbers listed, one item would have been reduced to the re-order level, re-ordered and those items, by coincidence, entered as merchandise received under the other item description. On the next week's reorder report, the first item would again show at an ROL and would again be reordered. That problem is now eliminated, thanks to version 2.0.

Three new reports have been added:

#1. - Price List. This gives you general item information and the RETAIL price, not your cost, so you can show this one to your customers.

#2. - 12 Month Sales Report. If you run a monthly update, this report will allow you to track the monthly sales of each item. The advantage of this type of report is tremendous for intelligent buying.

#3. - Cost Summary. If you need to know your cost on a selective group of items via the short inventory report (the long report is not active for this report), this will do it without printing pages of unnecessary descriptive information.

It was not documented in my copy of NWM's 2.0 upgrade data sheets, but I discovered Bruce has added a VERY helpful feature to Merchandise Issued/Received functions. In the earlier versions, if you typed in an item description and hit RET, and then discovered you had the wrong item or were in Issued when you should have been in Received, you were committed to filling in all the fields before you could get out of the situation. With version 2.0 you simply enter a zero for quantity issued or received, followed by RET and it will dump the current description/part number and allow you to start over. This feature has saved me a tremendous amount of wasted keystrokes.

I will not detail the new, built-in Installation file except to say it is very easy to use since you no longer have to work directly with a Superbase program.

Over the past several months, I have worked with various stages of version 2.0 and I can't praise it enough. In addition to being faster than 1.3, the above features make it smoother and easier to use, in addition to providing more useful information. If you upgraded from the original inventory program to version 1.3, you sure will want to do it again. If you are still using the original program, you don't know what you're missing.

Bob Loeffler
Green Lake Hardware
511 Mill St. P.O. box 218
Green Lake WI 54941
414/294-6412

Notice

This space was left open due to an oversite on the part of the general membership who forget this is a volunteer organization and all members are expected to contribute to its operation.

SUPERBASE EXPORT OF SELECTED RECORDS

Recently Mr. C.G. Paterson of Arizona wrote me and asked about how to output selected records from one Superbase file to another file with . identical configuration. I have figured out ways to do this in the past, but had forgotten so I started on the path to rediscovery. Let me share my findings in hopes that someone else has a better way.

There are four ways I know to accomplish the feat. Two ways work with the OUTPUT function of Superbase. I will not burden you with how to output selected files. The Superbase Manual is very clear in how this is accomplished. The problem comes when you try to import to an identical file using the output file you have just created with output. The problem is caused from the output operation creating two extra trailing "fields" in the output operation. When you try to pick them up with export, the output record is too long to fit in the file which is identical to the one from which you output the record. You could simply backup the database to another disk and delete the records you do not want, and then use export to convey the remaining files to the desired database, but this is too much trouble to justify calling it a solution. Another way that it could be accomplished is to write a program to strip the two extra "trailing fields" from the output records and then import them without any problem. Not many have the expertise to do this.

The two ways I use are: (1) Just create two extra fields at the end of the file to which you wish to import. One space each is sufficient. Nothing will appear in them when you import, but the records will each be as you desire them. (2) The method I like best is one which I have demonstrated on my "RECORD COLLECTION DATABASE" which is a Superbase Application Program. It allows you to choose the output file, the output selection method, and the drive and file to which you wish to input. That's correct! You can choose to output selected files from drive 0 and input them to drive 1 or drive 0 and to a file of any name you choose. The way I accomplish this is to assign each field in the output record to a variable string, holding it in the computer's memory while switching to another file, another database and file, or even another drive or drive unit. Another file is selected with the FILE "filename" command. Another database is selected with the DATABASE "dbasename" command. If you wish to change drives/units, use the command DATABASE "dbasename",8,1 where 8 is the unit and 1 is the drive number. You could use DATABASE "dbasename",9,0 for the first drive on unit nine. Then the data is read into the new file by way of reversing the file/string routine.

To read fields into a string you simply pick a string name '$name' (only the first two letters count) for text information or a numeric name 'x' and then assign the info in the field content into the string as $name=[fieldname] or x=[numericname].

For beginners: A DATABASE is the "container" for a collection of files (like a file drawer). A FILE is like a folder in which you keep records of a similar nature. A RECORD is like a single document or a two or three page letter. A FIELD is one single item of information, a number, or a sentence. DATABASES contain files. FILES contain records. RECORDS contain fields. FIELDS contain information.

For a sample of a flexible program to accomplish selective output and input with Superbase, get a copy of the disk, RECORD COLLECTION DATABASE, a Superbase Application Program, offered in this issue of ESCAPE.

Happy Computing,
Clyde Northrop

Clyde Northrop is the Theater Army Chaplain for Third U.S. Army, Ft. McPherson, Georgia. He enjoys building and has recently built an office in his basement with an eleven foot long computer desk with a full-length printer and disk drive shelf above it. The desk has no legs so that there is free access the full length. It depends on the wall behind it for it's full support.

RECORD COLLECTION DATABASE
A Superbase Application Program
by Clyde Northrop

```
0 "by northrop      "    Designed to Catalog and Index a Record Collection
1    "RECORD"            Database in which files, and records are stored
8    "hDirectory"        An annotated DIRECTORY of this disk
15   "hread.import"      A "read-me" file with helpful program hints
4    "hbinsort"          Help file for using the binsort program
2    "hexport"           Help file for using export
10   "hhelp"             Help file for program information
8    "hinstruct"         Help file for program use instruction
1    "hlist"             File created automatically by FIND function
4    "hprint"            Help file with screen print instructions
4    "hregister"         Help file for registering your disk for support
1    "Album Pr.p"        An Album Price program
7    "bin sorter.p"      Program to sort album collection by bin numbers
1    "configure.p"       Program to run before running bin sorter program
13   "export.p"          Output program - Very Versatile (See Article)
4    "menu.p"            Menu program - Selects function of your choice
3    "input.p"           An input program to enter record information
2    "run.p"             Runs a record cost/price report
6    "report.p"          Runs a report of Database Contents
4    "new report.p"      Third report program in case you don't like above
6    "brief report.p"    Runs a shorter report of Database Contents
7    "start.p"           Sets up configuration and loads menu
6    "album"             Record ALBUM file
6    "orchestra"         Record ORCHESTRA file
6    "jazz"              Record JAZZ file
6    "groups"            Record GROUPS file
5    "classical"         Record CLASSICAL file
1828 blocks free.
```

Recovering Damaged Superbase Files

This article delves into a most unpleasant subject, a damaged Superbase file. Most anyone who has experimented, with Superbase I and on, has probably seen that repugnant message, data or index mismatch. It does not matter whether it was the system, the program or user error that caused the problem, the results are the same. You can not access part of your file and making lists, running searches or printing reports are impossible due to the damaged records.

So what do you do? Your first thoughts, upon realizing the magnitude of the situation, are probably to blow up all your equipment or commit suicide. After a few minutes of profound thought, you have discarded the first two solutions and are seriously planning a search and destroy mission on the software developer and the company that sold you the program. Realistically, there are several other solutions depending on the situation.

There is a Utility program that you can load from basic on the Superbase 2 disk. This program can help you copy and save both damaged database files and normal files. It also works between different drive units. To load this program type <dload "utility",d0> without the left and right brackets. This is loaded from the Superbase 2 program disk in drive 0 of unit 8. It should be noted that Superbase 2 has also fixed a number of bugs which may eliminate some of these problems.

There are a few problems with this program. It does not support hard drive databases, large files and it has not worked for me on any task that I have tried it on. It should be tried first, as it is the easiest method of recovery if your database is cooperative. Note that the program asks you for the drive that you are going to recover from. If the source drive is other than eight, then press the backspace key and enter the correct unit number. You can not change the default any other way.

If the error has been caused by the entry of the dreaded double quote, or <"> in a field, then read page 163 and 164 of Superbase the Book. They treat the recovery of your file in depth in this manual. If you do not have this book and use Superbase, it is like eating dinner without a main course. You are missing the most important information available on this system.

There is a third method of fixing your database but should be left only to the experts. This requires the use of a disk editor, such as Super Disk Doc, and assumes that you understand how Superbase stores its' data. The basic concept is to find the index file and check the pointers to the data file and make sure the data is stored with integrity. This is probably the fastest and cleanest way of repairing a simple problem but also the most dangerous.

The manner in which Superbase stores its' files is as follows. The main database file is stored on a track and sector with all the files and forward pointers to the start of each index for each file. Each file name begins on a multiple of sixteen or 10 hex and starts at location 10 of that sector. Since the files can be stored from location 10 to F0 you can see why there can be only fifteen files in one database.

Superbase uses only one sector to store the file name and index location. The index file location is located at the start of the file name plus hex A. If the filename starts at hex 10 then the index location is at hex 1A. The number of records in the file is stored at the filename plus hex E such as 1E.

The index file contains all the keys in CBM ascii order. Every time you add a new key it appears that the index is sorted and rewritten back to the disk. The actual location of the data remains the same. The index key is stored with the actual data location following it. It also indicates whether it is the 00 record in the sector or the 01 record if the record data is in 128 byte half blocks.

This method is only for those that have investigated the Superbase file structures. If you attempt this method of repair, it is recommended that you have several backup disks. Please archive them in a place where they are safe from your probing disk editor. In any case, proceed with care.

Another method to salvage your data is to create a Superbase program that will rewrite your data to a new database and disk. This is the most work and the slowest method of recovery but is probably the safest and most reliable, for the majority of problems. The key to this method is to find the records that are causing you problems. Unfortunately, these bad records are usually in the middle of the database and not at the beginning or the end.

Since the bad records are embedded in the file it requires a dual process. You have to select each record and store the data into an array and save the data onto a new database disk. You must use this procedure until you come to the damaged record or records. Then you must select the next record, after the last damaged record, and continue to select and store records until you reach the end of the file.

The following program will show the basic concepts of the type of program necessary for easy recovery. Please note that you will have to change the database, file and field names, as named in your own system, to make thi program usable.

```
line #   -    comment

 5     initialize variables and array
 10    find out if user is using hard drive.
 20    get hard drive unit number.
 30    insert bad data disk in floppy drive 1
 40    initialize default floppy drive.
 50    enter the first and last records to process
       this is not necessarily the first and last record
       of the database.
 60    clear screen
 70    select damaged database and file.
 80    goto line 200
 90    display processing current record and change database
       to drive 0 new disk.
100    comparison of current record to last record selected
       if true goes to line 500.
110    clears fields in current record so it can be used
       as new record:array equals fields in new file:stores
120    selects bad database:selects last and next record.
200    retrieve data and store in an array.
210    same as line 200
220    returns to calling line.
300    stores fields from damaged file to new file.
310    same as line 300
320    returns to calling line.
500    clear current record:gets data:stores record and
       thanks you for the privilege of serving you.
510    displays message.
520    scratches file and renames file
900    subroutine to get y or n input
910    same as line 900
920    goto 900 if input is other than y or n
930    message
940    message
990    message
999    message
1000   goto 999 if input is not a carriage return.
```

```
 5 c1$=chr$(13):c2$=chr$(63):c3$=chr$(147):dim a(14):sp$="
 10 display @0@25,3@+"NWM INVENTORY 1.3 FILE RECOVERY"@23,8@+" ARE YOU USING A HARD DISK ? (y/n) ":g
osub 900:if y=2then 30
 20 gosub 990:wait h:if h>8and h<16then display c3$:d=0:goto 50
 30 display @0@15,8@+"PLEASE PUT YOUR NWM INVENTORY DATA DISK IN DRIVE 1":gosub 999:gosub 930:gosub
900:if y=2then 30
 40 h=8:d=1:display c3$:goto 50
 50 ask @23,5"ENTER FIRST RECORD   ";a$:ask @23,8"ENTER LAST RECORD    ";b$:gosub 940:if y=1then 70
 60 display c3$:goto 50
 70 display c3$:database "accounting",h,d:file "inventory":select a$:a$(1)=a$
 80 gosub 200
 90 display @0@39,8sp$:display @0@27,8"Processing  ";@+a$(1):database "accounting",8,0:file "invento
ry"
 100 if a$(1)=b$then 500
 110 clear:gosub 300:store
 120 database "accounting",h,d:file "inventory":select a$(1):select n:goto 80
 200 a$(1)=[item]:a(1)=[stock]:a$(2)=[unit]:a(2)=[retail]:a(3)=[cost]:a(4)=[date]:a$(3)=[manufacture
r]:a$(4)=[cat/loc]:a(5)=[in-route]:a(6)=[reserved]
 210 a(8)=[leadtime]:a(9)=[issues]:a(10)=[year-issues]:a(11)=[eoq]:a(12)=[rol]:a(13)=[usage]:a(14)=[
times/year]:a$(5)=[number]:a$(6)=[st]
 220 return
 300 [item]=a$(1):[stock]=a(1):[unit]=a$(2):[retail]=a(2):[cost]=a(3):[date]=a(4):[manufacturer]=a$(
3):[cat/loc]=a$(4):[in-route]=a(5):[reserved]=a(6)
 310 [leadtime]=a(8):[issues]=a(9):[year-issues]=a(10):[eoq]=a(11):[rol]=a(12):[usage]=a(13):[times/
year]=a(14):[number]=a$(5):[st]=a$(6)
 320 return
 500 clear:gosub 300:store:gosub 520:display @0c3$@20,5@+" THANK YOU FOR LETTING ME SERVE YOU !!! "@
20,8@+" PLEASE PULL OUT DISKS AND PRESS RETURN "plus
 510 @20,11@+" YOUR NEW NWM DATA DISK IS IN DRIVE 0 ! ":gosub 999:menu
 520 maintaino"s0:start.p":maintaino"r0:start.p=0:ostart.p":return
 900 wait y$:if y$="y"or y$=c1$then y=1:return
 910 if y$="n"or y$=c2$then y=2:return
 920 goto 900
 930 display @0@22,5@+" IS THE CORRECT DISK INSERTED (y/n) ":return
 940 display @28,11@+" IS THIS CORRECT ? (y/n) ":gosub 900:return
 990 display @0c3$@22,5@+" ENTER YOUR HARD DRIVE UNIT # (9-15) ":return
 999 display @0@26,20@+" (PRESS RETURN TO CONTINUE) ":wait w$:if w$=c1$then display c3$:return
 1000 goto 999
```

FILE DESTRUCTION THE EASY WAY

by: Anthony Goceliak

Norm tells me he is about to put out a short lead time issue, but I wanted to weigh in with my two cents on a very important topic which is apparently under-emphasized. As usual I have received some letters from CBUGgers in trouble, two of which in particular prompt me to write this article.

Both of these members are having what they term 'intermittant troubles' when saving files to disk. A healthy 80xx drive absolutely NEVER turns the error led red, except of course when it is reset at power-on. Among their assorted troubles which are indeed real and should be considered fatal, since they prevent the proper loading of (programs/data/letters) are the real killers, Write errors!

Most of the time when a drive is sick it 'knows', and tries like heck to tell you too. The blinking error led or bumps when loading or saving are all signs of ill health (except when loading the original versions of Superscript II and Superbase I, where the programmers tried to prevent unauthorized duplication by intentional disk error).

If your drive has declined to the point of aborting loads or saves you Are in the soup, and you better do something about it. Read errors can range in severity from annoyance to critical loss of data, but I suppose that there is some comfort in knowing that although one particular (file - pgm - database) has become unuseable, at least in principle once your drive has been repaired, all will once again be ok.

THIS IS DEFINITELY NOT THE CASE WITH WRITE ERRORS!!!!

[Some clarification is necessary at this point, since a write error, once neglected, will be interpreted by dos as a read error the next time you wish to load the relevant file. What I call a read error is a dos error due perhaps to a mis-aligned drive, or electronic troubles within the drive, but which does not really exist on disk. (i.e. a 'good' drive would read the sector ok). Write errors are uniformly mis-read by all drives, since the sector on the disk has been mis-written.]

When you save (or esc (f)ile, or dsave, or copy, or header, or backup, or WHATEVER), your drive finds a suitable place to write, and then goes there and writes. It immediately reads back what it wrote, just to be sure that every byte checks out and may attempt re-writes as needed to correct any disacrepancies. If the re-write was ok, you see a blink or two but when all is finished, the error led is green, ds$ comfortingly says 00, ok, etc, and you have lucked out. The big, bigger, and all-out disasterous troubles happen when your re-tries have been exhausted, and for whatever reason, "what were writ ain't what were read back".

When the error led is red after a disk operation involving a write, panic is certainly justified, but first ask for ds$. From Superscript II, you get an automatic warning on the status line, but I think it was a major error not to REQUIRE some command to resume from a non-zero ds$ message. The average operator heard the drive whir, heard the 'ding' and at the next keystroke, the ds$ message disappears unread, and he or she goes blithly on, unaware that unless disk armegeddon has already happenned, every subsequent write tempts it even closer.

Why is this so? Whether from basic or SS, big trouble involves having saved only some of your letter, program or whatever, bigger trouble involves the resulting splat file which can corrupt subsequent files as they are saved to disk, and really major trouble occurs when the bam or directory has been trashed by the write error. Depending on exactly what happened, you may have lost not just the file you were working on, but the entire contents of the disk!

A disk drive (whether Commodore's or anyone elses) can malfunction, so given a write error, what should you do about it? No single answer can truly cover all contingencies, but I reccommend the following steps:

1. If you have a different Unit, re-save whatever you were working on on the alternate unit after having made a backup from your vault copy of the disk which showed the write error FIRST!

2. If you can't comply with #1, and you have a dual drive, take a DIFFERENT DISK, put it in the Other drive and re-save. Now try the backup from your vault copy to a new blank and once again re-save.

In either of these cases, skip now to reccommendation #6 and enjoy peace of mind.

What? You don't have a vault copy of every disk that is important? Do it Now!

3. Try to make a backup of the disk with the write error. Even if the backup succeeds, there is no guarantee that you are out of the woods, but you have now produced a copy of your work 'frozen in time', so that subsequent attempts to resuscitate the disk can begin with only the original error. Collect the disk you have just produced.

4. Especially if the backup failed, header a new disk and Copy all files from the old disk to the new. Dos 2.7 allows you to do this with one command, as in copyd0,"*" to d1,"*" (and return). Again, success of this command does not guarantee that all is well, but if individual inspection of each file shows nothing truncated and no 'chimeras' (the front half of one program or letter pasted, possibly violently askew, to the rear half of some other file, then you probably are ok.

5. Adopt the very wise policy of frequent backups, since disks cost well under a buck, but the time and value of what fills them is much more costly.

For cases 3 and 4, you may or may not be out of the woods, since some alterations can be pretty subtle, but at least you are in a position to track down the culprit to only 254 characters of one file unless the directory has been zonked, so peace of mind can probably be wrestled from the wreckage. Please take careful heed of points 5 and 6 BEFORE you find the option taken from you!

6. Get your malfunctioning drive looked at - NOW! This is realy where that alternate unit comes in handy, so your business, or correspondance, or whatever is not held captive until the drive comes back.

In any case, I further reccommend that the original disk be labelled (in red) with the exact ds$ from the write error and filed away in a section marked defective in your "vault". Just in case. Consider it a sixty-cent insurance policy. If the above resuscitation efforts turn out against your expectations to be less than 100% successful, the original disk would be invaluable to piecing humpty-dumpty together again, especially if you have to go to someone else and ask them to do it.

to the cynics: You don't have to buy any software, and you don't even have to buy your disks through CBUG to do this (although they are of uniformly high quality and at a good price), but remember Mr. Murphy and his "laws", they will eventually remember you! It helps when all you've lost is ten minutes to make an extra backup from your "vault"!

One more thing; Norman, being at the center of things often acts as a clearing-house for referral to reliable shops who know how to work on our equipment, and can help you avoid the sad experience of sending your drive, b, or printer to a shop which keeps it for months only to return it still malfunctioning, or even worse than when it went in. Tell him when a shop works wonders, and by all means tell him if you wonder when the shop works.

THE CBUG LIBRARY

Our contributors have become renown from their superior quality library submissions. This issue is no exception. Clyde Northrup has entered a disk for indexing records -- or for that matter useful for anything from inaugeral gowns to fishing lures. Belatedly released as I mis-placed it while preparing the last issue.

Likewise, Armand Carrier's upgrade was mis-laid by me. But fortunately so because the replacement disk had so much more on it. Lots of useful goodies.

Dennis Jarvis and Jim Springer are at it again. This time, the programs we are releasing are a FIRST. First to be developed, first time published. Developed specifically for the members of CBUG, they will soon be adapted and published all over the world. Several pubishers have waived their first and exclusive publishing demand to get access to these works, they are so important. AND, more is coming next issue! Read the articles in this issue.

Bruce Faierson of North West Music wrote a valuable article on salvaging Superbase disks, and the program is here in library. Must be a big surprise to find it here!

While Mathew Goldstein has the record for upgrades (The Checkbook Program), Jon Whatley is running for second place. Jon has put important improvements into his Super Teacher and Super Church programs. Upgrades are available to existing owners at half price. No matter what your endevors, Jon's programs will help you learn how to do all sorts of Superbase reports easily and quickly. Most of the programming can be transfered intact for similar applications.

Then comes Lt. Col. John Wright who as invested so many hundred hours to the benefit of the co-processor effort. He brings us a "C" language compiler and its sorce code, and a disk of valuable tools for those interested in the co-processor application. And a few other goodies too. These run under the 8088 coprocessor only!

Now for the news: For CBUG to be of maximum value to its members, all must contribute. Possibly the most important thing is to let our authors know what YOU, yes you, yes each individual member needs. Or what you think you need. Don't be shy; blurt it out. I go to meeting after user group meeting and the chair asks for suggestions -- silence. An instructor asks for questions of mis-understanding and one would be led to believe there is always 101% comprehension. Speak up, or you may have to forever hold your peace!

Then too, there are a number of members who have proudly called to explain the great programming they wrote, were using, or were developing. A program to calculate cut length and assembly proceedure for wooden shipping crates for machinery; a options tracking and trading program; interfacing programs; and much much more. Where are you folks? Where are the many more of you who've developed useful or transportable programs? We've had too many instances of people spending weeks, months even a year or two re-inventing a wheel which someone else already perfected. Let's get the spirit. Let's continue helping ourselves so that we, even with our orphan, continue to get even more usefullness (not to mention at a cost of pennies) out of our systems!

Without the questions, statement of need, fullfillment and outright unsolicited contributions, there will be little in the library. You've all got Superscript and a printer. I ask you, how hard is it to communicate? To ask of CBUG, to ask of the contributors, to suggest, and ---- maybe even to thank and author or two. You know, the Maytag man gets paid to listen to the clock tick, CBUG authors neither get paid nor do they wish to sit idle, they may just go tick, tick, tick .....

RECORD ALBUM **CBUG #96** **NEW RELEASE** **2910** **#13225**

A Superbase Application

By: Clyde Northrup

Mr. Northrup has been a frequent contributor to CBUG library, generally with Superbase applications. Here is another excellent set of easy to understand programs in a format which obviously can be adapted to keep track of most anything from clothing to a collection of antique cars. See also the article on page 8 of this issue -- Summer 1988 Vol. 13.

```
0 "by northrop     "    Designed to Catalog and Index a Record Collection
1    "RECORD"           Database in which files and records are stored
8    "hDirectory"       An annotated DIRECTORY of this disk
15   "hread.import"     A "read-me" file with helpful program hints
4    "hbinsort"         Help file for using the binsort program
2    "hexport"          Help file for using export
10   "hhelp"            Help file for program information
8    "hinstruct"        Help file for program use instruction
1    "hlist"            File created automatically by FIND function
4    "hprint"           Help file with screen print instructions
4    "hregister"        Help file for registering your disk for support
1    "Album Pr.p"       An Album Price program
7    "bin sorter.p"     Program to sort album collection by bin numbers
1    "configure.p"      Program to run before running bin sorter program
13   "export.p"         Output program - Very Versatile (See Article)
4    "menu.p"           Menu program - Selects function of your choice
3    "input.p"          An input program to enter record information
2    "run.p"            Runs a record cost/price report
6    "report.p"         Runs a report of Database Contents
4    "new report.p"     Third report program in case you don't like above
6    "brief report.p"   Runs a shorter report of Database Contents
7    "start.p"          Sets up configuration and loads menu
6    "album"            Record ALBUM file
6    "orchestra"        Record ORCHESTRA file
6    "jazz"             Record JAZZ file
6    "groups"           Record GROUPS file
5    "classical"        Record CLASSICAL file
1828 blocks free.
```

MISC & SUMMER 88 PRINT FILES **CBUG #95** **NEW RELEASE** **2911** **#13211**

Here is a real bargain! Not only some exceptionally important new programs, but even the current issue's print file all on one disk. Armand Carrier is upgraded and added to his earlier works. We've put Mr. Carrier's programs first so you can use the shift run menu -- however the menu program's menu in " pdirectory " only contains the files on Mr. Carrier's contributions at this time. We left it this way as a challenge to our members to update with the remainder of the programs on the disk. -- should be an easy task.

By: Armand Carrier

This program is called 'super function keys'. It is similar to 'keys' only there's more to it. I found that many things can be done using the disk directory and the funtion keys. Before I get into that, let me explain why the program is called 'super function keys' but is listed in the disk directory as '²'. That's so I can load & run it fast and with the fewest keysrokes possible. This is the first program I load into the computer on startup. Pressing F3 gives me 'dload" ' so I need only add '²",d0:' and press 'shift/runstop' and the program will load and automatically run. So, by pressing 7 or 8 keys my program has been loaded and run with a minimum of effort and I am ready to proceed with my computing. OK. Back to the Fkeys. Using this program and the Fkeys, you can by pressing 1 or 2 keys: load a program, load & run a program, scratch a file, rename a file, copy a file, or even concatenate files. You can do all of the above in direct mode without having to load and run a program to do it, and without disturbing the program currently in memory. You can even read sequential files with the short program that is contained in F16. No more typing in the wrong spelling of a file and having to do it over again when you load, scratch, copy or rename a file.

* * * * * Instructions on using Super Function Keys * * * * *

List the directory on the screen until the file you want to work with is listed. Stop the listing so that the file you want is on screen and then, cursor up or down to it and press the Fkey for what you want to do. Now let me explain what each of these keys will do for you.

F3 will load a program from drive 0. No need to carefully type in the name & make sure the spelling is exact in order to avoid the dreaded error message 'file not found' and then proceed to type in printds$ and type the name in again,etc. Just press F3. No problems. How nice!

F4 does the same job as F3 with one additional feature. F4 will not only load a program from drive 0 at the press of a key, it will also automatically run it for you. Now that's really nice! Load & run with one key!

F5 is a black hole type similar to other nameless machines to delete text.

F9 will scratch a file with the usual option: 'are you sure?'. Proceed as usual. Press return to scratch, 'n' plus return to cancel.

F12 Press this key, type in the new name, and the file is renamed.

F13 Press this key and the file in drive 0 is copied to drive 1. No typing at all.
F14 will add the sequential file in the directory to whatever file you name. This file that you name will be the 2 files combined as 1 under the name you have given it. The first file will be unchanged but the 2nd will be.
F16 is a sequential file lister. Type new, press return to clear things out. Press F16, enter the filename at the prompt, and that's it! Press the commodore key to stop scrolling, press shift key to continue or press stop key to stop, type 'c0' and press return to continue. If you end after pressing the stop key, press F6 to close the file before continuing. If you are not sure of the filename or it's spelling, press F15 to get a listing of all the sequential files on the disk.
F17 will set your printer (4023) to lowercase/uppercase print.
F18 is to be used after loading a program with F3. Cursor back up to the line that says f$='filename' and by pressing F18 the program will be listed to the printer with the 'filename' centered and printed in expanded print at the top of the listing. When the listing is completed, press F19 to close things.
F20 is simply a printer reset. Enjoy! a/c.
P.S. For a printout of the 'super function keys' instructions use Superscript II Load program called 'ssII Contents'. Happy computing, CBUGGERS. Thanks Norm, Barbara, Billy & Jennifer.

```
0 "SUM 88 & MISC   " 95 2c    1   "twinkling*stars"  prg   4   "profit"           prg   12  "utility audit"   prg
30   " menu"           prg    1   "my window"        prg   7   "temp conv"        prg   4   "slow printing"   prg
1    "ZZZZZZZZZZZZZZZZ" prg   5   "flash titles"    prg   1   "ave"              prg   7   "roman"           prg
8    "+cbug"           prg    1   "cursor demo"      prg   1   "ascii"            prg   1   "wait"            prg
1    "+ciout.f02a8"    prg    2   "cursor demo 2"    prg   2   "angles (sine of)" prg   1   "timer"           prg
4    "  pdirectory  "  seq    2   "cursor demo 4"    prg   1   "peek"             prg   8   "+frm"            prg
8    "+terminated"     prg    2   "cursor demo 3"    prg   7   "mem jogger"       prg   6   "msx+"            prg
8    "+goodbye"        prg    9   "hi-lite demo"     prg   3   "scrn format"      prg   1   "def fn"          prg
10   "1 pg dir printer" prg   4   "alpharound"       prg   1   "feminist's"       prg   2   "printat"         prg
2    "disk spinner"    prg    8   "book names"       prg   1   "waves"            prg   16  "cbm 4023"        prg
2    "printer on/off"  prg    1   "flashing screen2" prg   8   "patterns"         prg   2   "bug"             prg
2    "window demo"     prg    1   "flashing screen"  prg   3   "prime factors"    prg   4   "delay loop"      prg
1    "quilt"           prg    1   "screen dump"      prg   10  "keyboard matrix"  prg   8   "keys"            prg
8    "disk display"    prg    3   "speedy"           prg   2   "phone phun"       prg   4   "printat+"        prg
11   "disk demo"       prg    4   "stop & cont"      prg   53  "contents"         prg   5   "scrolling"       prg
4    "disk comm1"      prg    5   "chr$ codes"       prg   20  "dec-hex-bin"      prg   4   "jump search"     prg
3    "disk comm2"      prg    3   "for next hint"    prg   1   "vtab"            prg   1   "dynamic 1"       prg
4    "disk comm3"      prg    2   "get 2"           prg   1   "bubble sort"      prg   2   "dynamic formula" prg
5    "disk write"      prg    3   "get 4"           prg   1   "shell sort"       prg   3   "alpha slalom2"   prg
4    "disk read"       prg    3   "data debugger"   prg   2   "quicksort"        prg   3   "alpha slalom"    prg
3    "disk overlays"   prg    2   "asc finder"      prg   4   "nrm distribution" prg   2   "trap with sound" prg
5    "disk dir"        prg    2   "binary quiz"     prg   1   "underlining"      prg   3   "which key"      prg
5    "cbugx"           prg    2   "binary table"    prg   2   "centered prting"  prg   1   "trap+"           prg
                                                                                      4   "²"               prg
```

From Dennis Jarvis and Jim Springer we have their latest designs. These very important materials are being released for the very first time. I'm told that nothing like them has ever existed before for any computer -- and we of CBUG have another first. See the articles at pages 14 and 15 in this issue, Summer 1988 Vol. 13.

```
52   "make 2 sided.bas" prg        13   "verify.bas"       prg
```

From Bruce Faierson of Northwest Music is the program "nwmerror.p" which runs within Superbase. It is discussed in some detail at page 10 of this issue.
This program is found within the Library Print File section -- which follows here:

```
1    "-SUMMER 88 PRINT" seq   37   "loeffler"         seq   43   "make 2 sided.txt" seq   11   "front cover"      seq
30   "library 85-88 or" seq   31   "goceliak"         seq   58   "make 2 sided.lst" seq   10   "96"               seq
25   "lib line85-88 or" seq   15   "anderson"         seq   16   "verify.lst"       seq   38   "95"               seq
8    "nwmerror.p"       seq   1    "scratch"          seq   19   "summer 88 paste " seq   14   "library lead"     seq
29   "nwm error artic1" seq   25   "carrier."         seq   1    "numbered footers" seq   8    "order form"       seq
20   "want ads"         seq   6    "springer"         seq   30   "scratch pad"      seq   879 blocks free.
3    "ohaloran ad"      seq   24   "copy pgm.txt"     seq   7    "east"            seq
112  "tutorial"         seq   30   "verify.txt"       seq   10   "pub. statement"   seq
```

SUPER CHURCH, rev 1 **CBUG #97** **UPGRADE RELEASE** **2912** **#13230**

By: Jon Whatley

This is a part of our regular upgrade program. If you already have the original Super Church disk, you may upgrade to this version at half price -- just attach the label (or identifyable portion thereof) to your order form and extend the price less 50%. The titles of the various programs below are descriptive of the functions provided by this suite. Like all database applications there are obvious opportunities to directly adapt the application to other uses -- scouting, medical services, public service clubs, etc.

```
1 "student records "  0 2c    1   "print clear.p"    seq   2   "pastorvisits.p"   seq   2   "mt labels"        seq
5    "set printer.p"    seq   2   "clear added.p"    seq   9   "hvisitation.p"    seq   3   "hlabels"          seq
```

```
8    "chdirector"       seq  1    "visitation list." seq  3    "director's list." seq  2    "visitation"       seq
1    "VISITATION"       seq  3    "visitation updat" seq  10   "help.p"          seq  3    "list.p"           seq
7    "visitation.p"     seq  3    "contacts.p"       seq  2    "menu2.p"         seq  3    "clear files.p"    seq
5    "menu1.p"          seq  3    "teenvisits.p"     seq  1    "contact list.p"  seq  7    "start.p"          seq
3    "field poster.p"   seq  4    "additions.p"      seq  1    "hlist"           seq  3    "singleparents.p"  seq
30   "registration.p"   seq  2    "clear print.p"    seq  13   "makelabels.p"    seq  5    "start2.p"         seq
2    "membership.p"     seq  3    "singleparent.p"   seq  9    "labels.p"        seq  30   "registration cy." seq
2    "additions2.p"     seq  2    "singlesvisits.p"  seq  3    "chdirector.p"    seq  1848 blocks free.
```

SUPER TEACHER II **CBUG #98** **UPGRADE RELEASE** **2913-16** **#13244**

By: Jon Whatley

This is a part of our regular upgrade program. If you already have the original Super Teacher disk, you may upgrade to this version at half price -- just attach the label (or identifyable portion thereof) to your order form and extend the price less 50%. The upgrade does NOT come with a package of 10 blank disks included in the price. The titles of the various programs below are descriptive of the functions provided by this suite. This is by far the most elaborate Superbase CBUG has heard of. Its capabilities were outlined in great detail in the Library Section page L12 of the Winter 1988 Vol 10 of ThE ESCAPE, page L7 of the Fall 1987 Vol 9 issue, and page L3 of the Summer 1987 Vol 8 issue.

If you are ordering Super Teacher II for the first time (at listed price), the four program disks come with a package of 10 blank OPUS disks as a bonus. We do this both because of the necessity of backing up your program and data disks before useage.

Mr. Whatley writes:

Since my last submission of Super Teacher, I, of course like all good B128 members, have been improving, updating, and eliminating problems with my pet work, SuperTeacher. I have made so many changes that at this point I can't really remember what all I've done but here's at least a partial listing....

*Automatic on-screen file name verification where needed

*Specific make-up work posting provision in the Grade Posters

*Elimination of printer line counting problems when printing from more than one program on the same page and improved page to page line counting when changing programs

*25-PAGE TUTORIAL which may be viewed/printer from SuperBase or from Superscript

*Programmed entry method for the Quiz Data and Review Data Disks which now allows formatting of essay questions as well as multiple-choice and true-false all within the same quiz or review worksheet (this program revolutionizes the system and may be referred to as a quiz generator)

*Menu to select printer style for users of the 8023P (printer)

*Provision to print out all three grade periods across 8.5" wide paper using compressed print (8023P)

*More variables for general user manipulation ...

The user now enters the names of the grade catagories and weights when using the student file registration programs ... and they may be changed if desired.

The Progress Report programs allow the user to enter the number of daily work categories used (now 3 plus the quiz) so that computer generated comments regarding the student's grades are more accurate.

*SuperTeacher II has a grade capacity of 72 grades plus the final exam grade. There are 3 daily work categories in each six week period plus the quiz category (formerly there were 2 daily work catagories plus quizzes for a total of 54 grades plus the final exam).

*Automatic grade averaging of student lists so that the user may have class average data (appears in many programs where appropriate)

*Graph program now includes student averages as well as class average.

*... and many, many other miscellaneous changes including a more professional appearance of screen displays throughout.

P.S. If we had programming competition I would gladly enter this system in the SuperBase category! I can now say that I am proud of SuperTeacher!!!

The set of 4 Super Teacher Disks together with 10 OPUS blank disks is only $24.00. Upgrade set only $12.00 with label.

* * * * * * * * * * * * !!! GET YOURSELF A POWERHOUSE !!! * * * * * * * * * * * *

B-1024 1 MEGABYTE MEMORY EXPANSION CIRCUIT BOARD

* In the Low Profile B-128, Implements Banks 0 through 14 with 256K by 1, 150ns socketed Dynamic Memory. This increase provides 8 Documents with SuperScript II; 28,560 Input Values with Calc Result; and 14 8032s with the 8432 Emulator!

* Fully Assembled and Tested. Plugs internally onto pin fields in the low profile B-128. You can install in minutes at your site! The original configuration of a stock B-128 can be restored at any time by simply removing the board. This is a non-destructive upgrade!

* Includes Pin Fields for future I/O, RAM Socket for $0800 to $1FFF in Bank 15, Documentation for Installation, Parts Layout, Schematics, and a Machine Language DRAM Memory Test Utility on 8050 Diskette!

* Three Memory Density Options available to fit your applications! You can purchase the 41256-15 DRAM in the market place at your convienence to fully populate the board!

24K RAM/ROM CARTRIDGE For The B-128/CBM-256

* Increase Memory! Add Another 24K of Memory to your System in BANK 15 from $2000-$7FFF! Complete! Assembled and Tested! Plugs into the Cartridge Port on the back of your main system unit!

* Run Scott's B-MON, Serial Bus Software, Custom Machine Language Applications, Keytrix, JCL Work Shop, Harrison's Assembler, and More!

RAM/ROM SOCKET For $0800-$1FFF

* Implements the 6K memory area below the Cartridge Port in BANK 15! A small Circuit Board with Connector and Socket that will receive a 6264LP-15 SRAM or a 2764-25 UVPROM to customize your application!

COMMODORE SERIAL BUS INTERFACE for the B-128

* A Full Featured Hardware Interface for the B-128 Implementing the Commodore Serial Bus with the Functions of Controller, Listener/Talker, Slow Bus, Fast Bus, Attention Acknowledge, Power-On Serial Bus Reset, Manual Serial Bus Reset! Comes in Rugged Plastic case! $ 59.95

B-1024 1 MEGABYTE MEMORY EXPANSION BOARD for the Low Profile B-128!

	B-1024 with 1024K Memory Installed	$699.00
ANDERSON	B-1024 with 512K Memory Installed	$499.00
COMMUNICATIONS ENGINEERING	B-1024 with 0K Memory Installed	$289.00
2560 Glass Rd. NE.	24K RAM/ROM CARTRIDGE with Case	$ 84.95
Cedar Rapids, Iowa U.S.A. 52402	RAM/ROM Socket for $0800-$1FFF in BANK 15	$ 24.95

TERMS: FREE SHIPPING IN USA, USA FUNDS, IOWA ADD 4%, ALLOW 6-8 WEEKS

* *

CBUG WANT ADS

CBUG want ads (for sale or wanted) are restricted to the B128 and related products and periferals. They are available to any CBUG member at the rate of $10.00 per 120 character line. We reserve the right to reformat and abreviate as appropriate. Send payment together with LEGIBLE copy to CBUG. Ads received within one month of printing are sure to be included in the next issue.

1.) B128, 4023, 8050, SS & SB, cables, Zenith Monitor, Orig. Pkging. $425. 512 899 2641

2.) SFD-1001 Disk Drive w/ IEEE/IEEE cable $115. Buscard IEEE interface for C-64 Computer, $75. Ken. Magin 312 359 5159

3.) 6400 Printer, IEEE interface, 4 print wheels & 3 new ribbons. Excellent codition. $475. Dave Ritterbusch 602 878 4832

4.) B128 (needs repair), 8050, 4023, Zenith Monitor, SB, SS, CalcR, GL, AR, Ref Guide, Disks. --MAKE OFFER-- Joy Gambino. P.O. Box 53968, Lafayette, La 70505. Days: 318 233 2872, Evenings: 318 984 8383

5.) B128, 8050, Extron Monitor, 4023, O/E, CalcR, A/R;, G/L, Payroll, I/C, SB, SS, Modem & disc, Hardware like new, software never used. $650. Alliance Christian School, Steven DeGeorge 304 598 2929

6.) B128, 8050, 4023, cables & manuals. Best Offer, John Shaffer, 510 Lawndale Dr., Danville Il. 61832. 217 446 1260

7.) 9060 Hard Drive, 220v w/110v converter $250. Dave Porter, Box 127, Moylan, Pa. 19065-0127 215 566 6287, 8-11pm est.

8.) B256 Sys + 4040, 8023. Bo R. Murray. Rt 1 Box 205B, Bourbon, Mo. 65441. SASE. 314 732 4956

9.) Protecto B128 system w/ CalcR, SS, SB. Also 8032 computer. Extra 8050 drive & service lit. Make offer. Will Split. Glenn Klein, Rt 2 Box 238, Richland Mo. 65556 417 286 3756

10.) Equipment surplus to my needs. Excellent or NEW condition. 1 B128 $125; 1 12" Transtar & 1 Zenith 12" monitors - $50 each; 1 4023 printer $90; 1 8023 printer - $140; 1 8050 dual drive - $300; 1 Anderson 1 MB expansion board (factory pkg) $575 (list $769). UPS charges collect. J.E. O'Halloran, Rt. 2 Owl Creek Rd, Hiawassee, Ga. 30546

11.) B128 systems. One high profile B-256 system. One 8050 drive w/ Tycom installed speed regulation. One 8050 w/ defective drive 1. One B-128 that malfunctions after warm-up. Software. Make offer. Anxious to sell. 205 739 3354.

12.) Protecto B128/256 + more. $450. Includes shipping. C. Playdon, Box 421, Londonderry, N.H. 03053. 603 434 5734

13.) B128 system, 2 8050 dual drives, 4023 printer w/extra ribbons. Complete CABS package, CalcR, SB, SS. $500. Call collect: 702 782 2084 after 4pm PST.

14.) For Sale: Commodore hardware, software and some rare documentation. Complete Protecto B128 systems, separate 8050 and 4040 drives, 8032 and B128 computers, 4023 printers, CBM to Centronics and CBM to Epson printer interfaces. The hardware is guaranteed, the prices are low, the phone will be answered. Call or write Don Coombs, 861 Harold, Moscow, Id. 83843. After 6pm PSC, 208 882 8720. Send SASE for list of CBM documentation, books available.

15.) B128, 8050, 4023 printer, monitor, SS, SB, CalcR, etc. $450. R. Boyer, 2601 E. Airport Dr., Tucson, Az. 85706.

16.) B128 system complete hardware & solftware. $400. Send stamped envelope for list & details. Carl Myers, 134 Knollwood Lane, Greenville, S.C. 29607. 803 233 6931.

17.) Protecto B128 Package with software & manuals. B128, 8050 dual drive, 4023 printer & amber monitor. Excellent condition $450. Frank Meadows 304 822 5592

SERVICE CENTER:

TYCOM INC
503 East Street
Pittsfield, MA 01201
(413) 442-9771

Authorized Commodore service center, we repair B128 8032, 4023, 8023, 6400, 4040, 8050, 8250, 2031, C64, C-128, etc. Rates are $50/Hr. plus parts. We will diagnose & quote repair cost for $25. Normal turn-around time is 5 business days. We ship via UPS. Limited supply 8032's and 4023's (new) at $189 each

* *

APT.TAX FOR 1988 INCOME TAXES for B128/B1024

If you can move a cursor and enter figures, you can use this program. Calc & transfers are automatic. HELP SCREENS. Custom programs - $29 + $5 page - you pick forms. CPA program w/ 21 forms $169 (requires 1 MB). POST PAID. Paid call support. Also for IBM/comp w/123 or Quattro, 1.2K(5.25) or 720K(3.5). APTco, Rt. 2 Owl Creek Road, Hiawassee, Ga. 30546 404 896 4342.

* *

-- BEELINE V2.1 -- BEELINE V2.1 --

>> THE PROFESSIONAL TERMINAL PROGRAM FOR THE B-128 COMPUTER <<

LOAD/SAVE IN ASCII OR CBM FORMAT
SKIP BEGINNING OF LARGE FILES

VIEW MEMORY BUFFERS USING FORWARD AND BACKWARD CONTROLS
SET BLOCKS FOR SAVING & PRINTING
TRANSFER DATA BETWEEN BUFFERS
FIND STRINGS WITHIN BUFFER DATA

PRINT TO CBM OR ASCII PRINTERS
USE VARIOUS DEVICE NUMBERS

RECEIVE DIRECTLY TO DISK FILES
UP TO 500K USING CBM 8050 DRIVE
USE WITH/WITHOUT MEMORY BUFFERS

MESSAGE EDITING WITHIN BEELINE
USE 9 SEPERATE MESSAGES WITH SCREEN EDITING & ESC CODES
TRANSMIT & EDIT USING FKEYS
PREPARE MESSAGE FROM BUFFER PREVIOUS OR BLANK MESSAGE

XMODEM UPLOAD AND DOWNLOAD
EXCHANGE ANY DISK FILE
VISUALLY MONITOR TRANSFERS

TRANSMIT DATA FROM MESSAGES ANY DATA FILE OR BUFFER
DELAY AFTER CHAR. OR LINE
USE XON/XOFF FLOW CONTROL

AVAILABLE IMMEDIATELY FOR ONLY $40.00 - ORDER FROM CBUG TODAY!

LEMDATA SOFTWARE, P.O. BOX 175, DOVER, PA 17315

* *

Super Office
List
~~$149.00~~

SUPER OFFICE

SALE PRICE
$49.95* U.S.

WITH SUPERBASE 2

**ALL THE GREAT FEATURES
OF SUPERSCRIPT II AND SUPERBASE
INTEGRATED IN ONE FANTASTIC PACKAGE**

Store important data then print out personalized custom reports inserting totals and names in the proper places. Do invoices, past due notices and many other things with Super Office. With Super Office you can do anything a small business needs.

Super Office is the ultimate in integrated programs for the "B" series market. Some features include left and right margin settings, tabs, decimal alignment, right justification, underlining, bold print, page numbering and a whole lot more. Super Office is easy to use with simple function key commands and an easy to use manual. There is even a spelling checker to keep you from making embarrassing spelling mistakes. We challenge you to find an easier integrated program or one with more features than Super Office.

Includes the ultimate data base at an affordable price. You can do anything that requires data storage with Super Office. Simply set up your screens for easy entry of information (up to 4 screens). Then set up the way you'd like the information to be printed. Super Office does the rest! This program has full calculation capabilities and full data storage manipulation. You do not have to reload either program to access the other. Automated merging through the Super Office application generator.

(must have 256K-1 Meg. upgrade to operate)

*Purchase subject to signed disclaimer.

TAKE A LOOK AT THESE GREAT SUPER OFFICE FEATURES

- Help Screens
- Bibliographies
- Document Chaining
- Sales record
- Home Budget
- Built-in programming language
- Inventory
- Highlighting
- Up to 127 Fields per Record
- Calculation capabilities
- Formatting
- Up to 4 Data Entry Screens per File
- Form Letter Printout
- Better than dbase II
- Mailing list
- User definable screen formats
- Applications Generator
- Up to 15 files per Database
- Keyed Access
- Report Generator
- Macro Capability

Shipping and handling charges $3.95

IF YOU VALUE YOUR TIME, THEN IT'S TIME TO BUY SUPER OFFICE

NWM Rebate Plan

Super Office is shipped with the Superscript II, Superbase I and Super Office Insert

Buy one of our expanded B-256 computers and purchase Super Office for only:

$39.95 with SBI and SSII Manuals

$29.95 without SBI and SSII Manuals

Without B-256 Purchase:

$49.95 with SBI and SSII Manuals

$39.95 without SBI and SSII Manuals

NORTHWEST MUSIC CENTER INC.

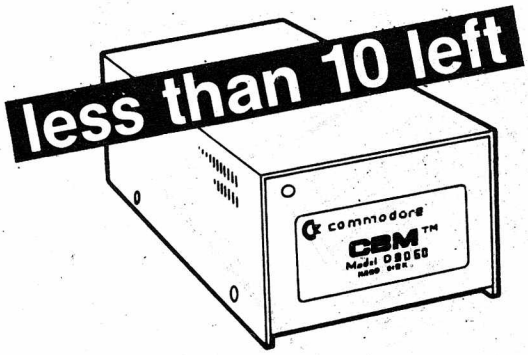
539 N. Wolf Rd. — Wheeling, IL 60090
Hours— (Voice) Phone (312) 520-2540
Mon.-Thur. 12:30-5:00, Sat. 12:00-4:00
(24 Hour Order Recorder)

CBM DISK DRIVE CORNER

8050 Dual Disk Drive

The model 8050 dual floppy disk unit uses a 100 Track Per Inch (TPI) single headed drive with a storage capacity of 533,248 bytes per drive. Each 8050 diskette has 77 tracks, and is read/write compatible with the model 8250 disk drive. This compatibility is limited to one side of the diskette.

NEW PRICED AT **$400.00** (U.S.)
USED or REHABS FROM **$200** *now available*
ADD $16.95 SHIPPING (U.S.)



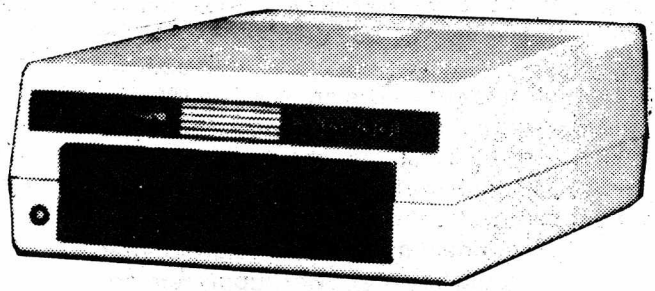
REHABS PRICED AT **$495.00** (U.S.)
ADD $15.95 SHIPPING (U.S.)

D9060 & D9090 Hard Disk Drive

The two models of hard disk units are the 5¼" single-drive non-removable "Winchester" technology storage devices. The D9060 and D9090 units feature two or three platters with recording surfaces on both sides and provide respectively 5.0 or 7.5 million characters of storage. A single random access file may occupy the entire capacity of either unit. An IEEE Interface connector is located on the back of the drive. Near the lower edge of the rear panel is a "slow blow" fuse, and an AC power cord.

SFD 1001 1 Megabyte Drive
double sided 8250 format
IEEE interface

N.W. Music now offers a constructive upgrade for the SFD-1001+ increasing its function and versatility. This modification allows the user to operate in a true single drive 8050 mode. This eliminates the previous problem of loading most 8050 software. This upgrade is only $19.95 over the base SFD price. This modification allows the user to select, by switch, which mode they desire to operate during power down. The user will stay in this mode until power down and switch reversal.



PRICED AT **$149.95** (U.S.)
ADD $10.45 SHIPPING (U.S.)
PRICED AT **$169.90** (U.S.) SFD with 8050 switch
ADD $10.45 SHIPPING (U.S.)

NORTHWEST MUSIC CENTER INC.

539 N. Wolf Rd. — Wheeling, IL 60090
Hours— (Voice) Phone (312) 520-2540
Mon.-Thur. 12:30-5:00, Sat. 12:00-4:00
(24 Hour Order Recorder)

CBM 128-80
w/8088 Co-processor

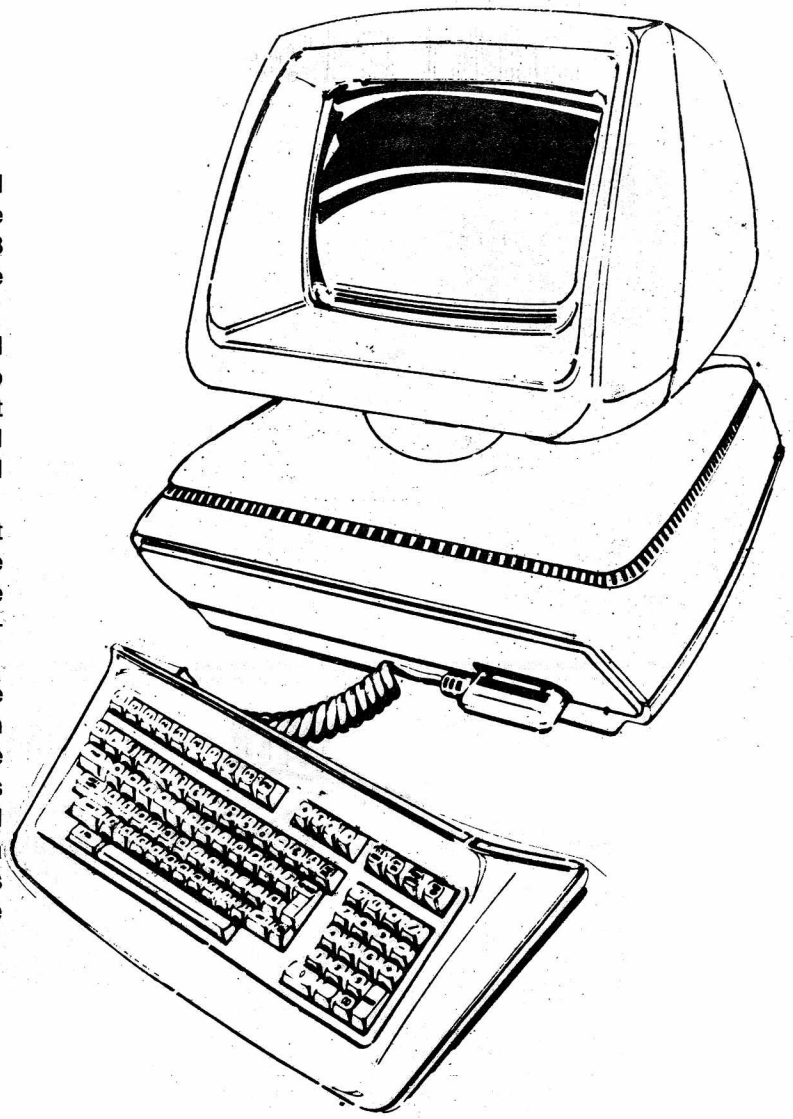
Call for Details
Very Limited Quantities

We are offering a limited number of used CBMX 128-80 computers for only $325. These come installed and tested with the original 8088 co-processor boards that until recently were hidden in the Commodore research labs.

The CP/M 86* operating system was implemented on the B system and has been tested and found to be very reliable. It had previously been stated that the co-processor would only work with the CBM 256-80. This is not true! The PLA that is installed on the hi-profile motherboard determines whether the operating system will run on that particular machine. We will also be offering the 8088 co-processor board for $80 providing the purchaser sends in their working hi-profile motherboard with the correct PLA. Call for details!

All the generic CP/M 86 software that we have tested will operate on the B series machine with an 8050 drive. The SFD and the 8250 can also be used with some restrictions. These same programs can also be run under the Digital Research CP/M 86 and Concurrent PC Dos operating system on an IBM. Therefore an investment in software is not wasted as it can be ported to other compatible computers.

*CP/M 86 is a trademark of DRI Inc.



SPECIAL-Co-processor only $80

In an effort to promote the CP/M-86 and Ms-Dos project, N.W. Music has decided to sell the remaining co-processor boards at our cost to help stimulate this effort. The price will be $80 and purchase will be subject to certain stipulations. It had been previously stated, that these boards would not work with the lo-profile computer. This does not seem to be the case. It has been suggested by a prominent member of the group that any problems could be only power supply related. We have a lo-profile model up and running. There are only a few boards remaining so call for details.

Co-processor $80

CBM 128 & 256 features:

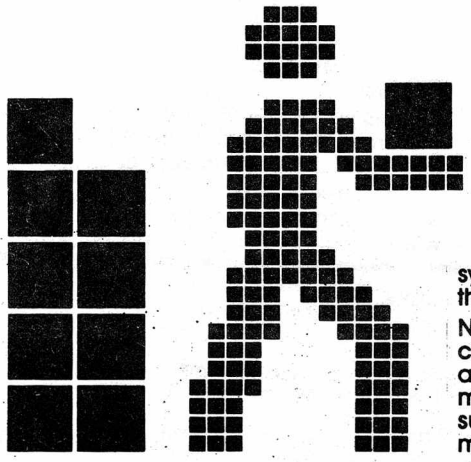
- SWIVEL MONITOR
- ADJUSTS HOR. and VERT.
- DETACHABLE KEYBOARD
- 9 x 14 PIXEL DISPLAY
- INCREDIBLE RESOLUTION
- DESIGNED FOR 2 INTERNAL DRIVES

Priced from $199 to $350 (US)
SHIPPING CHARGES EXTRA!

You really have to see the green phosphor display to believe it. This model has at least as good a display as the other company with those three big letters.

NORTHWEST MUSIC CENTER INC.

539 N. Wolf Rd. — Wheeling, IL 60090
Hours— (Voice) Phone (312) 520-2540
Mon.-Thur. 12:30-5:00, Sat. 12:00-4:00
(24 Hour Order Recorder)



NWM'S INVENTORY CONTROL SYSTEM* VERSION 1.3

This system was developed by a user for users. It seems that most systems are developed by computer engineers that never see or use the product when finished.

NWM, inc. chose the toughest software analyst we could find to give a complete and unbiased review. Bob Loefler is the gentleman that tore apart the CABS Accounting System and documented all the bugs in most of the modules. To have someone of this caliber make suggestions means better, more efficient software available for CBUG members.

- Loads program modules in less than **8 seconds** (superbase 2) to main menus in **3 seconds** or less
- On screen **pop-up calculator** in transaction modules
- Most data centered functions use the **calculator keypad**
- Versatile report features allow for **3 ways** to print the same report. User selects the fastest method
- Built in sophisticated export program allows for **complete packing** of the database
- Type ahead feature allowed
- You can display reports on screen
- Access to superbase menu for user developed applications
- Partial-match key search now allowed in transaction modules
- User can search forward and backward through file while in transaction modules
- Elimination of most tiresome prompts for faster movements between subfunctions and menus
- Simplification of entering prices and costs that have not changed when entering transactions
- User programmable calculator allows you to turn the calculator function on or off within the transaction modules. If the user does not need the calculator, it saves time by eliminating the prompt and bypasses the calculator function

NWM, inc. proudly presents Inventory Control 2.0 w/dual key indexing only $49.95

Listen to how NWM has improved this already great Inventory Control System...

- Dual indexing (allows user to select record by item or stock number)
- New records can be entered in issues, receipts and orders modules
- Price list report (w/o cost for customer viewing)
- Sales report (sales history by item per month for 12 months)
- Cost summary (calculates total cost for a category of items selected)
- Configuration module (allows user friendly configuration of printers, drives etc.)

Only a $19.95 upgrade charge for owners of previous versions of NWM Inventory.

WE SUPPORT OUR CUSTOMERS!

NWM Inventory Control System Version 1.3 Prices

B Version 1 8050 ..... **$39.95** C-128 Version 1 1571 ..... **$24.95**
B Version 2 8050 ..... **$39.95** B-128 Version 1&2 8050... **$44.95**

U.S. shipping and handling charge $3.95 (Prepaid)

NWM, inc. • 539 N. Wolf Rd. •Wheeling, IL 60090

(312) 520-2540 Hours— (Voice) **Mon.-Thur. 12:30-5:00, Sat. 12:00-4:00** (24 Hour Order Recorder)

Superbase is a registered trademark of Precidion Software.
B128 and C128 are registered trademarks of Commodore Business Machines Limited.

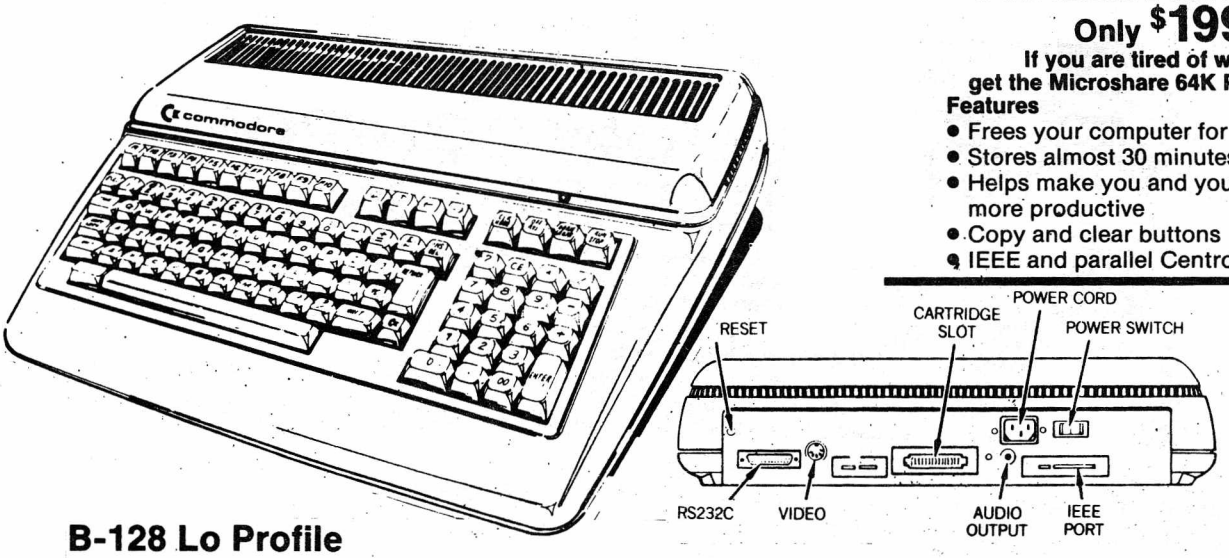
*Requires use of Superbase®

64K IEEE Print Buffer
Only $199

If you are tired of waiting, get the Microshare 64K Print Buffer.

Features

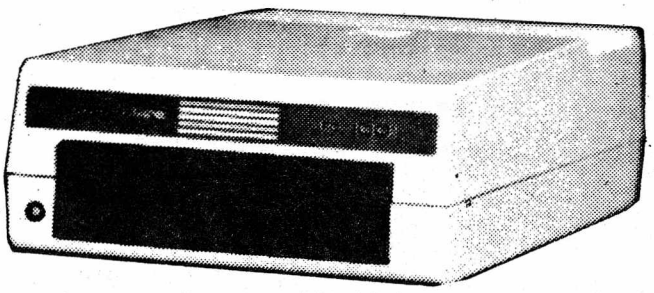
- Frees your computer for other tasks
- Stores almost 30 minutes of printing
- Helps make you and your employees more productive
- Copy and clear buttons
- IEEE and parallel Centronix output



B-128 Lo Profile

We have a very limited supply of new B-128-80 computers left. The available supply will last no longer than late October to early November if current sales hold-up. We also have a supply of Commodore rehabbed machines. If you desire a new machine, please order now! There are no new machines available once our supplies run out.

PRICED AT **$145** new
$99 rehabbed
ADD $11.45 SHIPPING (US)



SFD 1001 1 Megabyte Drive
double sided 8250 format
IEEE interface

N.W. Music now offers a constructive upgrade for the SFD-1001+ increasing its function and versatility. This modification allows the user to operate in a true single drive 8050 mode. This eliminates the previous problem of loading most 8050 software. This upgrade is only $19.95 over the base SFD price. This modification allows the user to select, by switch, which mode they desire to operate during power down. The user will stay in this mode until power down and switch reversal.

PRICED AT **$149.95** (US)
ADD $10.45 SHIPPING (US)

PRICED AT **$169.90** (US) SFD with 8050 switch
ADD $10.45 SHIPPING (US)

COMMODORE
8000-9000 SOFTWARE & MISC.

9000 Superpet	$199.95
64K exp for 8032	$110
BPI General Ledger	$25
BPI Accts Payable	$25
BPI Job Cost	$25
BPI Accts Receivable	$25
BPI Inventory	$25
Superscript 8032	$79
Superbase 8096	$79
OZZ Database	$25
Legal Time Acc.	$25
Dow Jones Program	$25
Info Designs 8032 Accounting System	$50
Superoffice 8096	$109.95
Calc Result 8032	$89
Petcom 1.3 Superpet comm program	$79.95

ORDER NOW WHILE STOCK LASTS!

Send or call your orders to: **Northwest Music Center, Inc. 539 N. Wolf Rd., Wheeling IL 60090. 312-520-2540** For prepaid orders add $25.50 for Superpet, $10.45 SFD 1001, $11.45 B-128, $10.45 4023p, $16.45 9090 and $5.45 64K memory expansion. For software add $3.50 for first and $2.00 for each additional book or program. Canadian shipping charges are double U.S. For C.O.D. orders add $2.20 per box shipped. All orders must be paid in U.S. funds. Include phone numbers with area codes. Do not use P.O. Box, only UPS shippable addresses. A 2 week hold will be imposed on all orders placed with a personal or business check. C.O.D. orders shipped in U.S. only and cash on delivery, no checks. 30 day warranty on all products from NWM, Inc. No manufacturer warranty. NWM reserves the right to limit quantities to stock on hand and adjust prices without notice!

All prices quoted in US dollars.

NORTHWEST MUSIC CENTER INC.

539 N. Wolf Rd. — Wheeling, IL 60090
Hours— (Voice) Phone (312) 520-2540
Mon.-Thur. 12:30-5:00, Sat. 12:00-4:00
(24 Hour Order Recorder)

USE PHOTOCOPIES FOR MULTIPLE ORDERS - ADDITIONAL ORDER FORMS WILL BE INCLUDED WITH SHIPMENT

CUT HERE

NAME _____

ADDRESS _____

CITY _____ STATE _____ ZIP _____

THIS IS YOUR SHIPPING LABEL -- PRINT OR TYPE NEATLY

ORDERS OVER 4 DISKS OR PHYSICAL EXAM SHIPPED BY BY UPS WITHIN CONTINENTAL U.S. YOU MUST USE STREET ADDRESSES FOR SUCH ORDERS -- UPS CAN NOT DELIVER TO POST OFFICE BOXES!

Your phone # _____

Remit to: CBUG, Inc., 4102 N. Odell, Norridge, Il. 60634 U.S.A. All payments must be in US funds.
IMPORTANT: If your shipping address is different than your membership address, please give your membership address!!

My Membership Address (if different) is: _____ (street) _____ (city) _____ (state) _____ (zip)

R ck mo ca pr S

Description	Quantity	Stock #	Price	Extension
COMMODORE IEEE to Centronics Adaptor Board (6400 type)	_____	11221	35.00	_____
CBUG connector type changer for above (changes output connector to standard Centronics "blue ribbon" D shell type	_____	11236	15.00	_____
*SERIAL BUS CARTRIDGE ADAPTOR (Anderson) (use with CBUG #84 Fast Bus programs)	_____	12608	65.00	_____
*24K RAM/ROM Cartridge (Anderson)	_____	12613	85.00	_____
BALANCE FWD FROM DISK & DATA CASE ORDER FORMS				_____
<North America addresses only!!!> Illinois residents add 7% Sales tax to above items only				_____
CBUG #84 FAST BUS Programs Disk (Jarvis/Springer)	_____	13033	35.00	_____
CBUG #95 MISC. & Summer 88 Print Files	_____	13211	9.00	_____
CBUG #96 Record Album	_____	13225	9.00	_____
CBUG #97 Super Church Rev. 1	_____	13230	9.00	_____
CBUG #98 Super Teacher II	_____	13244	24.00	_____
PR #19 CP/M 86 2.004	_____	13259	9.00	_____
PR #20 CP/M 86 2.005	_____	13263	9.00	_____
_____	_____	_____	_____	_____
_____	_____	_____	_____	_____
_____	_____	_____	_____	_____
KNIGHT's 8050 (DOS 2.7) COPY UTILITY	_____	12204	20.00	_____
PHYSICAL EXAM for the 8050 Disk Drive	_____	12219	35.00	_____
PHYSICAL EXAM for the 8250 and SFD-1001 Disk Drives	_____	12192	35.00	_____
BEELINE v2.1 Telecommunications Program	_____	12280	40.00	_____
Close tollerance Zener diodes & instructions for 8050 $6/pair	_____	11330	6.00	_____
Precision Reference diodes & instructions for 8050 repair /pair	_____	11344	8.50	_____
Recopy fee remittance	_____	12401	5.00	_____
BALANCE FORWARD FROM ADDITIONAL ORDER FORM			------->	_____
MERCHANDISE SHIPPING AND HANDLING CHARGE --- ALWAYS INCLUDED, NO EXCEPTIONS				$ 2.00
SUBTOTAL				$ _____
Free Will Contribution to CBUG		12416		_____
Extra Copy of this issue, Summer 1988 CBUG ESCAPE (this issue)	_____	13278	6.00	_____

BY CHECK ___ BY MONEY ORDER ___ TOTAL REMITTED $ _____

* These are special ordered by CBUG as a convenience for members. There is an extra $5.00 handling charge built in to these prices. Pricing and availability subject to change. Please allow extra time for trans-shipping. You may prefer to order directly from Mr. Anderson -- see ad page 2.

NOTE: We endeavor to ship against money order within 5 business days of receipt. Others delayed 14 to 18 days.

BLANK PAGE

USE PHOTOCOPIES FOR MULTIPLE ORDERS - ADDITIONAL ORDER FORMS WILL BE INCLUDED WITH SHIPMENT — Summer 88

CUT HERE

NAME _____

ADDRESS _____

CITY _____ STATE _____ ZIP _____

THIS IS YOUR SHIPPING LABEL -- PRINT OR TYPE NEATLY

ORDERS OVER 4 DISKS OR PHYSICAL EXAM SHIPPED BY BY UPS WITHIN CONTINENTAL U.S. YOU MUST USE STREET ADDRESSES FOR SUCH ORDERS -- UPS CAN NOT DELIVER TO POST OFFICE BOXES!

Your phone # _____

Remit to: CBUG, Inc., 4102 N. Odell, Norridge, Il. 60634 U.S.A. All payments must be in US funds.
IMPORTANT: If your shipping address is different than your membership address, please give your membership address!!

My Membership Address (if different) is: _____ (street) _____ (city) _____ (state) _____ (zip)

R ck mo ca pr S

Description	Quantity	Stock #	Price	Extension
COMMODORE IEEE to Centronics Adaptor Board (6400 type)	_____	11221	35.00	_____
CBUG connector type changer for above (changes output connector to standard Centronics "blue ribbon" D shell type	_____	11236	15.00	_____
*SERIAL BUS CARTRIDGE ADAPTOR (Anderson)	_____	12608	65.00	_____
*24K RAM/ROM Cartridge (Anderson) (use with CBUG #84 Fast Bus programs)	_____	12613	85.00	_____
BALANCE FWD FROM DISK & DATA CASE ORDER FORMS				_____
<North America addresses only!!!>				
Illinois residents add 7% Sales tax to above items only				_____
CBUG #84 FAST BUS Programs Disk (Jarvis/Springer)	_____	13033	35.00	_____
CBUG #95 MISC. & Summer 88 Print Files	_____	13211	9.00	_____
CBUG #96 Record Album	_____	13225	9.00	_____
CBUG #97 Super Church Rev. 1	_____	13230	9.00	_____
CBUG #98 Super Teacher II	_____	13244	24.00	_____
PR #19 CP/M 86 2.004	_____	13259	9.00	_____
PR #20 CP/M 86 2.005	_____	13263	9.00	_____
_____	_____	_____	_____	_____
_____	_____	_____	_____	_____
_____	_____	_____	_____	_____
KNIGHT's 8050 (DOS 2.7) COPY UTILITY	_____	12204	20.00	_____
PHYSICAL EXAM for the 8050 Disk Drive	_____	12219	35.00	_____
PHYSICAL EXAM for the 8250 and SFD-1001 Disk Drives	_____	12192	35.00	_____
BEELINE v2.1 Telecommunications Program	_____	12280	40.00	_____
Close tollerance Zener diodes & instructions for 8050 $6/pair	_____	11330	6.00	_____
Precision Reference diodes & instructions for 8050 repair /pair	_____	11344	8.50	_____
Recopy fee remittance	_____	12401	5.00	_____
BALANCE FORWARD FROM ADDITIONAL ORDER FORM ------->				_____
MERCHANDISE SHIPPING AND HANDLING CHARGE --- ALWAYS INCLUDED, NO EXCEPTIONS				$ 2.00
SUBTOTAL				$ _____
Free Will Contribution to CBUG		12416		_____
Extra Copy of this issue, Summer 1988 CBUG ESCAPE (this issue)	_____	13278	6.00	_____
BY CHECK ___ BY MONEY ORDER ___ TOTAL REMITTED				$ _____

* These are special ordered by CBUG as a convenience for members. There is an extra $5.00 handling charge built in to these prices. Pricing and availability subject to change. Please allow extra time for trans-shipping. You may prefer to order directly from Mr. Anderson -- see ad page 2.

NOTE: We endeavor to ship against money order within 5 business days of receipt. Others delayed 14 to 18 days.

CUT HERE

CUT HERE

CUT HERE

BLANK PAGE

Hacker's Corner

One of a Kind • Surplus • Monthly Special • Closeouts

Limited quantities to stock on hand

PRECISION SOFTWARE

Superbase 1	$9.95
Superscript 2	$19.95
Superbase 2	$39.95
Superscript 3	$39.95

HANDIC SOFTWARE

Calc Result	$79.95
Word Result (special order)	$79.95

B-128 GOODIES

B-128 Programmable Reference Guide was $29.95	$ 5.95
SFD 1001 1 meg drive	$149.95
Superbase The Book	$14.95
Superpet 9000	$150.00
Applied Calc Result	$14.95
The Power of Calc Result	$14.95
8050 rehabs from	$200-$300

Due to popular demand we are now handling the ribbons for the B series printers

MPP-1361 and 8023p ribbons	$5.50
4023p ribbons	$6.00
6400, 8300, Diablo 630 ribbons	$4.95
9 1/2 x 11 continuous form perf (2500 sheets) per box	$24.95

C.A.B.S. ACCOUNTING

General Ledger	$ 9.95
Accounts Receivable	$ 9.95
Accounts Payable	$ 9.95
Order Entry	$ 9.95
Payroll	$ 9.95

SUPER DISK DOC

It's your choice, you can get it now . . .

If you want control over your disks then you need Super Disk Doc!

operates from easy to use duck shoot menus
examines bytes on your disks
interprets in English, hex or ascii
modify and save disk data
recovers previously unreadable files

$24.95

Never before has there been such a powerful and easy to use disk utility like this for Commodore computers!

or . . . you can wait until its too late!

PRINTERS

Gemini 15 x 120 cps	$125.00
Smith Corona DM200 RS232 and parallel (160 cps w/near letter quality mode)	$150.00

8023P 160 CPS PRICED AT $179 US
ADD $16.95 SHIPPING US

ONE OF A KIND ITEMS

EPSON GENEVA PORTABLE. INCLUDES: SOFTWARE, MODEM, ACOUSTIC COUPLER, CABLES, PRINTER, CARRYING CASE. (Floor Model) $475.00

CP/M COMPUTER . . . . . .
EVEREX 2400 MODEM $245.00
MONOCHROME MONITOR $59.95
9090 7.5 meg HARD DRIVE, REHAB . . $495.00

ORDER NOW WHILE STOCK LASTS!

Send or call your orders to: **Northwest Music Center, Inc. 539 N. Wolf Rd., Wheeling IL 60090. 312-520-2540** For prepaid orders add $25.50 for Superpet, $10.45 SFD 1001, $11.45 B-128. $10.45 4023p, $16.45 9090 and $5.45 64K memory expansion. For software add $3.50 for first and $2.00 for each additional book or program. Canadian shipping charges are double U.S. For C.O.D. orders add $2.20 per box shipped. All orders must be paid in U.S. funds. Include phone numbers with area codes. Do not use P.O. Box, only UPS shippable addresses. A 2 week hold will be imposed on all orders placed with a personal or business check. C.O.D. orders shipped in U.S. only and cash on delivery, no checks. 30 day warranty on all products from NWM, Inc. No manufacturer warranty. NWM reserves the right to limit quantities to stock on hand and adjust prices without notice!

All prices quoted in US dollars.

NORTHWEST MUSIC CENTER INC.

539 N. Wolf Rd. — Wheeling, IL 60090
Hours— (Voice) Phone (312) 520-2540
Mon.-Thur. 12:30-5:00, Sat. 12:00-4:00
(24 Hour Order Recorder)

CUSTOM IEEE CABLES

CBUG has received numerous calls from members wishing to have special IEEE cables. Longer, extra plugs, etc.

My most complex system has a B128, 2 8050's, a 4040, plus printers -- 8300P, 8023P, a star NL-10 and daisywriter 2000. Some of the equipment is on the buss all the time, some others I'd like to have on all the time; and a few have to remain off buss if they are not turned on. NOTE: Connecting more than two un-powered devices will devistate the buss -- you will get data errors, erase wrong disks, etc. Thus, shutting down 2 of three printers as long as everything else on the line is powered is usually OK. One exception though, the 6400 printer and some early CBM dot matrix printers will lock up the buss in certain modes.

Right along with such a complex system is the length of cable runs -- often the 1 meter cables just don't reach. Plus it is expensive to have a cable for each device.

SOLUTION: multiple male (and female) connectors on a long flat cable. CBUG has acquired the equipment to prepare special IEEE cables to your sketch. Since all CBM IEEE periferals receive information via a female socket on the rear, normally there is no reason to have even one single female on the cable -- mere plug the first male into the back of your P to I cable at the first periferal, and string it along one device to the next.

To order make a sketch similar to that below:

```
M-----------M-----------M-------F-------M-------------------------M
      4'          3.5'        1'      2'            7.5'
```

The sketch does not need be to scale. Be sure to allow slack between each run for servicing your installation and to allow for turning plugs in the proper attitude device to device -- atleast 6" each side of a plug.

PRICING:		
	BASE CHARGE PER CORD	$15.00
	PER RUNNING FOOT OF CABLE	1.00
	PER CONNECTOR	5.00
	Ill. Residents add 7% sales tax	
	Shipping & Handling per order	2.00

NOTE: This is a flat cable crimp on connector system. Connectors are single faced, either Male or Female. Not both as on the regular round wire IEEE cables. We suggest you do not exceed 30' prox in overall cable on your system. Do not use flat cable in close proximity to your wife's favorite TV, etc.

Please order on a separate sheet of paper including your name, address and phone even if submitted with a regular CBUG order form. Cable orders will usually be shipped separately due to fabrication time -- allow extra week to 10 days.

This offer is for IEEE connectors on the cable ONLY. We do not do the Pet to IEEE end. You only need one of those -- for the computer in any event.

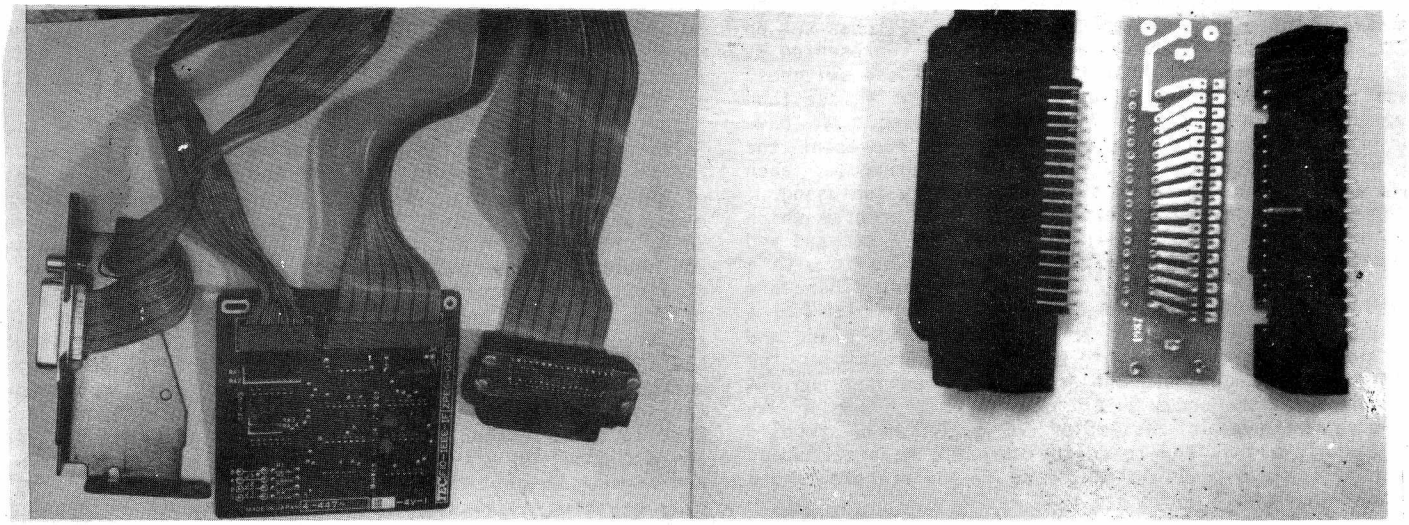
◆◆◆◆◆◆◆◆◆◆◆◆◆

PHYSICAL EXAM

NOW FOR THE 8250 & SFD 1001

$35.00 ea. from CBUG

8250 -- SFD 1001 order #12192; 8050 order #12219; 4040 order #12238
1541 order 12223; 1571 order #12242



IEEE to Centronics Adaptor Board
order #11221 $35.00

Connector type changer (comes assembled)
order #11236 $15.00

USE THE 6400 IEEE CONVERTER WITH OTHER PRINTERS

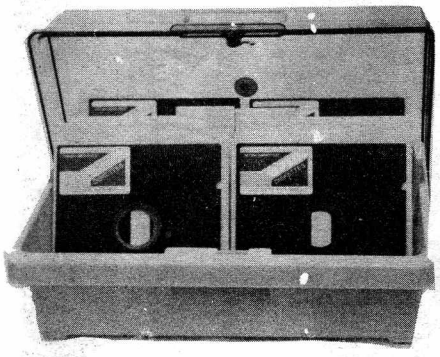
A couple of issues back Warren Kernaghan suggested that the 6400 converter board was a superior method of IEEE to Centronics interfacing. So why not rewire the adaptor board's flat cable to a standard 36 pin Centronics connector? Easy said, not so easy to do even for fairly good technicians. So, CBUG has made up a little circuit board and rounded up the necessary mating connectors to produce a type changer adaptor. Plug the type changer into the Centronics header on the IEEE adaptor, and the other side of the type changer into your Centronics ported printer. Walla, instant Centronics with all the added features Mr. Kernaghan wrote about.

For about a year, CBUG has been offering the IEEE internal converter designed for the CBM 6400 printer. It is a 4" square board with two long flat cables, one of which has a standard IEEE 488 connector, and the other has a 34 pin dual row 1/10" header to connect with the logic board in the 6400. Unfortunately the header with not mate with standard Contronics connectors. The circuit board is double sided plated thru, the connectors prime quality with gold plated contacts.

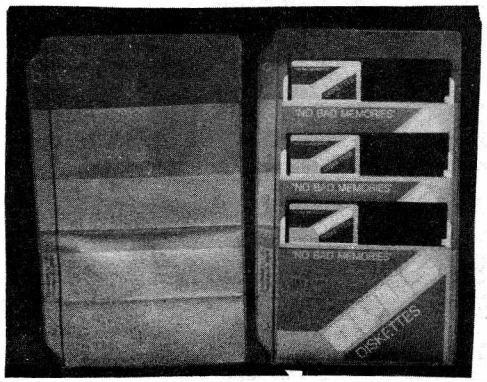
These adaptors are believed to work with most common printers, though they will not work with my large Daisywriter 2000. They do work with the Star and Cannon printers and many others which now follow the standards. If by chance the adaptor does not work with your printer, we'll refund the full purchase price.

The adaptors have jumper positions to select device numbers other than 4. They are not enclosed so you will want to mount or tape them securely out of the way or install them in a protective box. In some cases, the connector on the printer uses bailing clamps to keep the connectors latched together -- so you may need to use a short Centnronics extension cord compatable with the bailing clamps which are readily available locally or via mail order (see Matos's article).

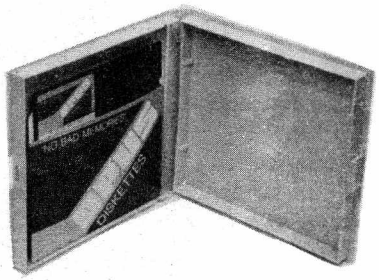
DATA CASES FOR 5 1/4" DISKS



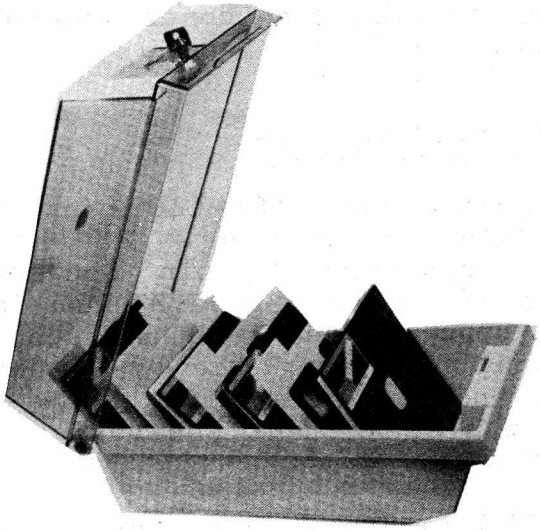
120.25L Clear top, putty body locking case for up to 120 5.25" disks. Also holds C.D. disks & spooling carts.



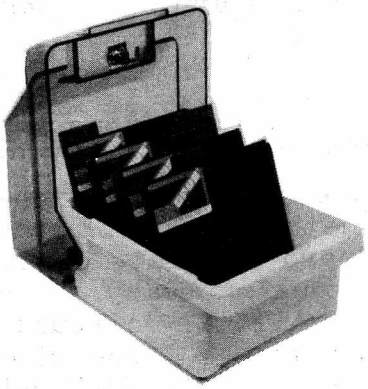
V3.15 Antistatic clear vinyl flexible sheet for standard 8.5 x 11" 3 ring binder. 3 pockets for 5.25" disks. Non-fragile.



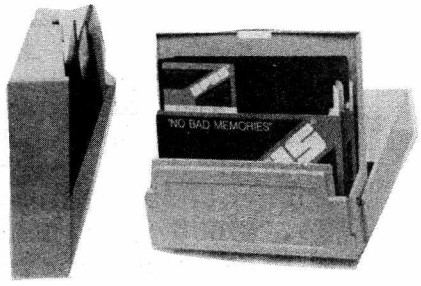
5.15 Rigid Vinyl type tote/mailer for 5 ea 5.25" disks. Pastel colors du jour. Non-fragile.



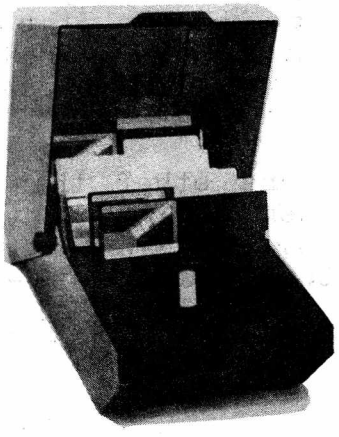
100.15L Clear top, putty body locking case for up to 100 5.25 disks. Our most popular 5.25" data case!



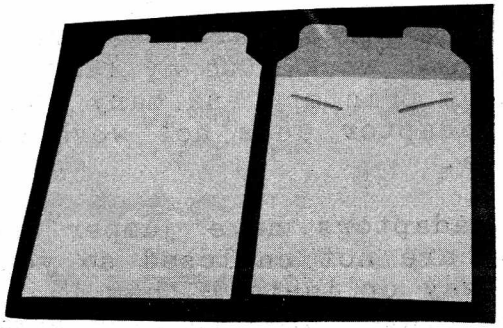
50.15L Clear top, putty body locking case for up to 50 5.25" disks.



10.15 Rigid Vinyl type case for up to 10 5.25" disks. Opens to set up display for easy access to disks. Pastel colors du jour. Non-fragile

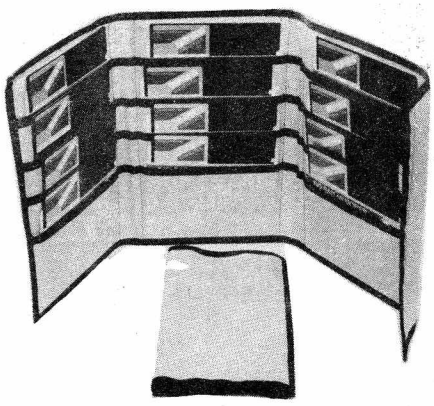


50.15 Smoked transparent top, darked smoked body. Non locking for up to 50 5.25" disks.

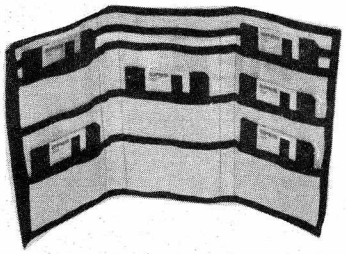


#M3.15 White chipboard mailing envelope. Holds 3+ 5.25" disks. 1.33 oz.

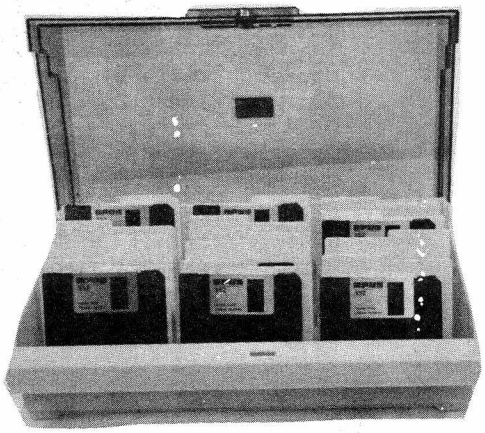
DATA CASES FOR 3 1/2" DISKS



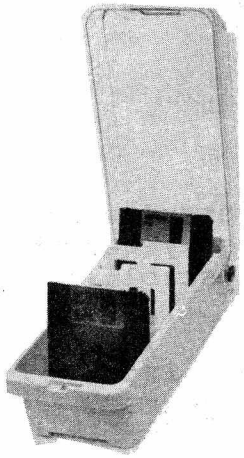
W12.35 Floppy Wallet. Anti-static Nylon with reinforcing rigid panels. Velcro closures Holds 12 5.25 disks. Can hold up to 24 disks. Colors du jour. Non-fragile.



W9.33 Floppy Wallet Anti-static Nylon fabric construction. Velcro closures. Holds 9 3.5" disks. Colors du jour. Suitable for pocket or purse. Panels not reinforced. Non-fragile.

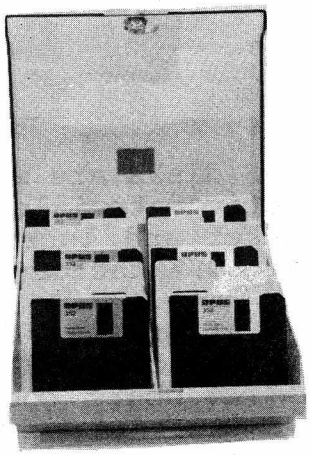


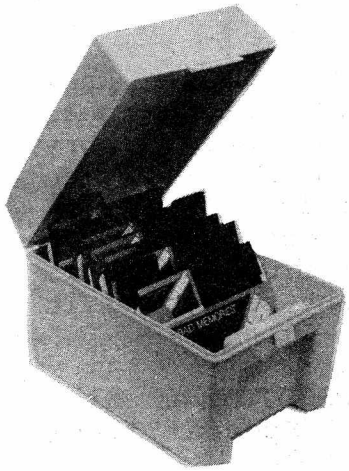
120.33L Lightly smoked top, putty body locking case for up to 120 3.5" disks. The largest 3.5" data storage made!

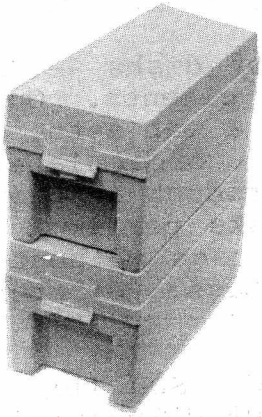


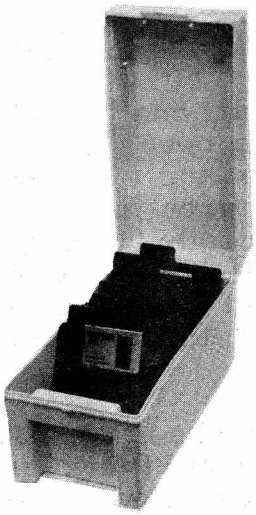
45.13 Clear top, putty body non-locking case for up to 45 3.5" disks.

80.23L Clear top, putty body, locking case for up to 80 3.5" disks.





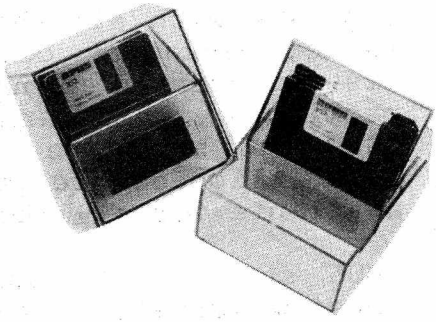




NESTING - STACKABLE

60.15 60 count 5.25" # 50.13 50 count 3.5"

These rugged poly-prop boxes are one piece construction with "living hinge" construction. Secure front latch allows ease of handling without risk of spill. Inexpensive and ideal for tote or archival usage as well as normal day to day uses. Non-fragile. Colors du jour. Putty shipped unless color req.



10.13 Clear transparant flip up storage and display case for up to 10 3.5" disks

Specifications & manufacturers subject to change. Substitutions allowable unless preferance marked on order form.

OPUS®

"NO BAD MEMORIES"

OPUS disks are without doubt the finest disks made. Mil spec formulas and exotic materials promise performance superior to all others! Heavy weight jacket and lubricated antistatic liner insure the ultimate in stability. The 100% factory lifetime warrantee is virtually un-necessary:

> After years of accelerated testing the numbers are finally in: Typical OPUS disk life is 38.5 million passes -- that's 12 years 24 hours continuous, or 36 years 8 hour/day business usage; far exceeding even the most demanding needs of any user. These extraordinary findings are well known by our clients.

CBUG is proud to offer just one brand of disks -- the only one we know will absolutely satisfy the needs of our clients. Our OPUS product comes complete with all the trimmings -- label & tab sets, sleeves, and factory sealed packaging complete with the OPUS corner label on each disk. No mis-represented no-name seconds. Most product is provided in full retail packing -- chipboard shelf boxes 10 disks per box. For economy, 5.25" SSDD and DSDD are provided in plastic bags without the chipboard box. 5.25" Flippies are packaged in 25 count bags. Of course any packaging is available for any item.

CBUG stocks all popular disks from 3.5" SSDD thru 8" DSDD, even high density formats. Immediate delivery is usually available. Special items such as hard sector disks are readily available thru CBUG.

OPUS is a full line manufacturer of all types of computer storage media. Let us provide you with superior Spooling tapes (backup tapes), Reel to reel and other computer needs.

CBUG, INC. NORRIDGE IL. 60634

OPUS QUALITY DISKS

Description	Stock #	$/pkg	Qty	Extension	Packaging	Cost ea.
5.25" SSDD	10017	$5.60	_____	_____	Poly Bag, 10 disks*	$ .56
5.25" DSDD	10021	5.90	_____	_____	Poly Bag, 10 disks*	.59
5.25" DSDD Flippy	10111	15.25	_____	_____	Poly Bag, 25 disks**	.61
5.25" DSQD Quad Density	10182	17.00	_____	_____	Retail Box, 10 disks	1.70
5.25" DSAT for AT etc.	10228	14.00	_____	_____	Retail Box, 10 disks	1.40
3.5" SSDD	10074	13.00	_____	_____	Retail Box, 10 disks	1.30
3.5" DSDD	10088	14.00	_____	_____	Retail Box, 10 disks	1.40
3.5" High Density (PC-2)	10125	50.00	_____	_____	Retail Box, 10 disks	5.00
8" SSDD	10509	21.60	_____	_____	Retail Box, 10 disks	2.16
8" DSDD	10547	23.70	_____	_____	Retail Box, 10 disks	2.37
5.25" Tyvek Sleeves	10318	1.25	_____	_____	Poly Bag, 25 sleeves	.05
5.25" Paper Sleeves	10303	1.00	_____	_____	Poly Bag, 25 sleeves	.04
Label & Tab sets. 5 ea labels & tabs per sheet black ink, white paper	10322	1.20	_____	_____	Poly Bag, 20 sheets	.06

All Opus disks are packaged with the appropriate complement of labels and tabs. Each disk carries the OPUS label. 5.25" and 8" disks are inserted in chipboard OPUS sleeves; 3.5" in individual poly bags. Retail packaging is a fully decorated and labeled OPUS box.

*Poly Bag packaging is exactly the same as retail box, but without the box. Bags are factory sealed. These items also available in full retail box packaging: $.30/box extra

**Retail packaging and other count packaging available on special request.

Call for quotations on special packaging, bulk and duplicator un-labeled product.

DATA CASES

Item #	Stock #	Price	Qty	Extension
5.25"				
120.25L	10960	$15.00	____	_____
100.15L	10975	12.00	____	_____
60.15	10994*	7.50	____	_____
50.15L	10922	10.50	____	_____
50.15	10834	9.00	____	_____
W12.35	10903***	12.50	____	_____
10.15	10941**	2.50	____	_____
5.15	10989**	1.50	____	_____
M3.15	10797	.40	____	_____
V3.15	10872	1.00	____	_____
3.5"				
120.33L	10782	18.00	____	_____
80.23L	10815	15.00	____	_____
50.13	10918*	7.00	____	_____
45.13	10829	7.00	____	_____
W9.33	10891***	8.50	____	_____
10.13	10778	3.00	____	_____
SUBTOTAL FROM OTHER SIDE			____	_____
				==========
ORDER SUBTOTAL				_____
7% Illinois Sales Tax				_____
Shipping & Handling (U.S. surface)				$3.00
				==========
TOTAL REMITTED			US $	_____

Prices & Specifications subject to change without notice. 18Nov88

Colors available. We will try to follow your requests when in stock. Circle your choice in the applicable catagory. Circle "ONLY" if you will not accept color du jour.

* Putty, Black, Slate Blue, Gray, Mauve, Burgendy ONLY

** Putty, Red, Yellow, Blue, Green, Gray, Black, ONLY

*** Gray, Maroon, Navy ONLY

Send order and make checks payable to:
CBUG, Inc., 4102 N. Odell, Norridge, Il. 60634
312 456 8720 (after Nov. '89, area code 708)

Your Phone Number _____

Terms: Cash item checks and money orders shipped promptly. Personal checks held two weeks.

This is your shipping label. Neatly PLEASE!!!

Name_____

Firm_____

Address_____

City_____ State_____

Zip Code _____

CBUG, INC.
4102 N. ODELL NORRIDGE IL. 60634

ADDITIONAL WANT ADS

1.) Two 8050 1-Meg Dual-Disk Drives w/ manuals. One has device number switch (8 or 9), One has an error tone / no error tone switch (when the switch is on, a disk error causes a soft tone until disk status is retrieved). Including IEEE-488 cable. $350 each or best offers.

4023 Dot-Matrix Tractor/Friction Printer w/ manual. $80 or best offer.

CALL or WRITE ***BEFORE*** ORDERING. Warren Swan, 1 N. 114 Woods Ave., Wheaton, Il. 60188. 312 665 1514 7pm to 9pm CST!!!

2.) Complete B-128 System; B-128, Dual Floppy Drive, Commodore 6400 Printer with Forms Tractor Feed, Magnavox Monitor 80, All software I possess. Original boxes, all cables. $500 plus shipping. James J. Wallace, 503 Main, Oconto, Wi. 54153.

Super Teacher Programs stock #13244.1

```
1 "superbase 700   " sb 2c    15  "grade poster.p"    seq  5   "hposter.p"          seq  2   "study list.p"      seq
1    "sb en 04apr86"    prg   16  "grade poster2.p"   seq  3   "features.p"         seq  2   "ltrgd ranked.p"    seq
14   "report.p"         seq   15  "grade poster3.p"   seq  5   "hprogress.p"        seq  6   "manual entry rec"  seq
4    "henter"           seq   7   "instructions2.p"   seq  6   "habsences.p"        seq  1   "hlist"            seq
5    "hselect"          seq   5   "absence record.p"  seq  6   "hlists.p"           seq  10  "labels2.p"        seq
8    "hfind"            seq   10  "registration.p"    seq  4   "hdaily records.p"   seq  2   "seating"          seq
5    "houtput"          seq   7   "menu1.p"           seq  5   "hfield poster.p"    seq  5   "set printer.p"    seq
5    "hcalc"            seq   6   "pd1 daily record"  seq  6   "hmissing work.p"    seq  13  "graph.p"          seq
5    "hreport"          seq   6   "pd2 daily record"  seq  4   "hmissing work li"   seq  4   "tutor.p"          seq
4    "hexecute"         seq   6   "pd3 daily record"  seq  12  "hposter tips.p"     seq  3   "file graph.p"     seq
5    "hhelp"            seq   9   "field poster.p"    seq  6   "hcalculations.p"    seq  3   "hgraph.p"        seq
5    "hfile"            seq   4   "menu2.p"           seq  11  "missing work3.p"    seq  126 "htutor"          seq
7    "hformat"          seq   15  "report1.p"         seq  5   "quiz format.p"     seq  5   "rename data.p"    seq
4    "hbatch"           seq   15  "report2.p"         seq  3   "key.p"             seq  6   "hlist.p"          seq
5    "hsort"            seq   15  "report3.p"         seq  8   "instructions.p"    seq  5   "address list.p"   seq
4    "hprog"            seq   4   "progress reports"  seq  3   "study guide.p"     seq  5   "set printergd.p"  seq
4    "hmaintain"        seq   3   "daily records.p"   seq  5   "menu6.p"          seq  5   "set printerrd.p"  seq
4    "hmemo"            seq   4   "grade posters.p"   seq  4   "start II.p"       seq  3   "certificate.p"   seq
3    "hcommands"        seq   4   "lists.p"           seq  5   "menu7.p"          seq  7   "display list.p"   seq
3    "himport"          seq   2   "absence data.p"    seq  4   "menu14.p"         seq  4   "student grader.p" seq
3    "hexport"          seq   9   "registration2.p"   seq  4   "menu15.p"         seq  110 "htutor II"       seq
7    "hmenu"            seq   3   "registrations.p"   seq  5   "hltrgd.p"         seq  16  "format quiz.p"    seq
12   "hlabels"          seq   8   "hregistration.p"   seq  13  "fexam calc.p"     seq  17  "format review.p"  seq
7    "grade work sheet" seq   3   "absence list2.p"   seq  4   "start III.p"      seq  17  "format review1.p" seq
13   "makelabels.p"     seq   7   "missing work.p"    seq  6   "review format.p"   seq  16  "format quiz1.p"   seq
4    "student list.p"   seq   7   "missing work lis"  seq  4   "menu2a.p"         seq  11  "format review 1." seq
9    "grade list.p"     seq   6   "missing work2.p"   seq  2   "hmanual entries."  seq  13  "quiz review.p"    seq
5    "ranked list.p"    seq   3   "aisd reports.p"    seq  3   "aisd report1.p"    seq  1031 blocks free.
19   "daily record.p"   seq   3   "sp prog rept lis"  seq  3   "aisd report2.p"    seq
6    "ltrgd.p"          seq   4   "ID list.p"         seq  3   "aisd report3.p"    seq
4    "absence list.p"   seq   2   "manual entries.p"  seq  3   "aisd report4.p"    seq
4    "absence poster.p" seq   4   "manual entry2.p"   seq  3   "aisd report5.p"    seq
5    "clear files.p"    seq   3   "hprocedure.p"      seq  3   "aisd report6.p"    seq
```

Super Teacher Student Data stock #13244.2

```
1 "student records " 01 2c    15  "period 2"          seq  15  "period 5"          seq  8   "start.p"          seq
1    "STUDENT"records   seq   15  "period 3"          seq  15  "period 6"          seq  4   "file"             seq
15   "period 1"         seq   15  "period 4"          seq  15  "period 7"          seq  1   "hlist"            seq
                                                                                          1910 blocks free.
```

Super Teacher Review Data stock #13244.3

```
1 "data bank      " 01 2c    3   "chapter 5"         seq  3   "chapter 11"         seq  3   "chapter 16"       seq
1    "REVIEW"i          seq   3   "chapter 6"         seq  3   "chapter 12"         seq  3   "chapter 17"       seq
3    "chapter 1"        seq   3   "chapter 7"         seq  3   "chapter 13"         seq  3   "chapter 18"       seq
3    "chapter 2"        seq   3   "chapter 8"         seq  3   "chapter 14"         seq  8   "start.p"          seq
3    "chapter 3"        seq   3   "chapter 9"         seq  3   "chapter 15"         seq  3   "chapter 19"       seq
3    "chapter 4"        seq   3   "chapter 10"        seq  1   "REVIEW"ii           seq  3   "chapter 20"       seq
                                                                                          1962 blocks free.
```

Super Teacher Quiz Data stock #13244.4

```
1 "databank       " 01 2c    3   "chapter 4"         seq  3   "chapter 11"         seq  3   "chapter 17"       seq
1    "QUIZ"data         seq   3   "chapter 5"         seq  3   "chapter 12"         seq  3   "chapter 18"       seq
8    "start.p"          seq   3   "chapter 6"         seq  3   "chapter 13"         seq  3   "chapter 19"       seq
3    "final"            seq   3   "chapter 7"         seq  3   "chapter 14"         seq  1   "hlist"            seq
3    "chapter 1"        seq   3   "chapter 8"         seq  1   "QUIZ"data ii        seq  3   "chapter 20"       seq
3    "chapter 2"        seq   3   "chapter 9"         seq  3   "chapter 15"         seq  1950 blocks free.
3    "chapter 3"        seq   3   "chapter 10"        seq  3   "chapter 16"         seq
```

CP/M 86 2.004 **PR #19** **NEW RELEASE** **2917** **#13259**

By: Lt. Col. John A. Wright

Here's disk four of CP/M-86 utilities. I have changed my normal format for this disk because it only contains a Small C compiler. In the past I have included a "DISK.LBR" file used to hold all the "information" type files. In this disk it was not necessary as the whole disk is needed for the compiler.

I have included the source code for all the modules needed to assemble the compiler so that someone smarter than me can modify the code and make it run faster. The program runs fine on the B-256. I can't guaratee how well it will run on the B-128. If you look at the DOC file, you will see that the program requires 128K of RAM. That may be pushing the

B-128 a little. The DOC file also states that it will run with less once compiled. In fact the C86N.CMD file is only about 30K, so I think it will work.

Granted CP/M is a somewhat old operating system, but from what I have seen CP/M and MSDOS are pretty close to being the same. In fact some of the newer machines are still using the 8088 CPU. I think that we have something here that no other PC has, especially the Commodore PC 10 series. We can run, with some modifications, all C-64/128 programs plus all CP/M-86, CP/M, (using the Z-80 emulator), and MSDOS 1.25 routines. What we need is the system code for MSDOS 2.0 or higher and we will have the best of all worlds. I remember an article in one of the first CBUG issues about emulating IMB computers. Well we can do it now with the 8088 all we need is the operating system.

-CATALOG OF CP/M-86 UTILITY DISK 004

PROVIDED BY:

LT COL JOHN A. WRIGHT
818 JUNIPER DR., PAPILLION, NE.
68046
Tele: 402-339-5728

```
**************************************************************************
********          NOTICE!!!:  All programs copyrited, and not      *********
********          Releasable outside CBUG, nor releaseable for    *********
********          for commercial purposes!!!                      *********
**************************************************************************

004.01      C86      .CMD   35K   ED 54  CP/M-86 Small C Compiler Source Code
004.02      C86N     .C      5K   B3 9D
004.03      C86N-0   .C      5K   AF B8
004.04      C86N-1   .C      7K   68 20
004.05      C86N-2   .C     10K   79 F9
004.06      C86N-3   .C      5K   32 F9
004.07      C86N-4   .C      6K   52 B3
004.08      C86N-5   .C      3K   0E 46
004.09      C86N-6   .C      4K   1D EC
004.10      C86N-7   .C      4K   4E 41
004.11      C86N-8   .C      5K   BE 29
004.12      C86N-9   .C      5K   35 94
004.13      C86LIB   .A86    17K   A2 5B
004.14      C86      .DOC    1K   FD D1  Doc on how to assemble above code.
004.15      C86N     .A86  212K   AD C4  Compiled version of above code
004.16      C86N     .CMD   35K   DD A9  CP/M-86 Small C compiler program.
004.17      C86NDEF  .C      4K   73 90  Misc Small C routines.
004.18      CDEMOA   .C      2K   03 6C
004.19      CDEMOB   .C      1K   92 12
004.20      CDEMOC   .C      2K   68 7C
004.21      CDETAB   .C      3K   72 72
004.22      CENTAB   .C      3K   73 38
004.23      CTESTC   .C      6K   88 85
004.24      CTESTDO  .C      1K   7B AD
004.25      CTESTID  .C      1K   C8 80
004.26      CTESTIN  .C      1K   A7 05
004.27      CTESTN   .C      1K   ED 4F
004.28      CUNCMT   .C      2K   81 1C
Software Tools RCPM - MISC Volume Number - 004, 28 Files cataloged.
```

CP/M 86 2.005 **PR #20** **NEW RELEASE** **2918** **#13263**

CBUG CP/M-86 DISK 005 HELP FILE

By: Lt. Col. John A. Wright

This is a new CP/M disk that may or may not be usefull to the members of CBUG. I have compiled a series of files here that are (a): either runnable on the B-128, or (b): should be runnable on the B-128.

This is, without a doubt, not a "NOVICE" disk. In fact, I am using this disk to ask for support in the CP/M work that we are doing here at CBUG.

On this disk you will find several files that start with the "+" symbol. These will run under CPM 80 emulation using the Z80 emulator supplied with this disk. For those of you that just want to use "runnable" files, this disk contains several that you can use. But for those of you who want to experiment with other capabilities (definately not the new CPM user) look at the other files. Those with the "" as their lead character are files that "should" run under 8080 emulation, but don't. I have included the "source" files with these files. So if you feel ambitious in you efforts, see what you can do. I have included an 8080 to 8086 translation routine to help with your efforts. Just run "+xlt86"

and let it convert the 8080 code ASM file to an A86 file. Then correct the OP-CODES that XLT can't handle.

Then there are the files that I would like to have. These are files that look like they would serve a purpose for us. Unfortunately they don't have source files associated with them. I have used the "#" symbol to differentiate these files from the others I have previously explained. They should run, but I keep getting the following error message "BRANCH OUTSIDE ADDRESS SPACE". If someone can figure out what we need to do to "fix" this problem, P L E A S E let me know! These don't have the source files with them so I think it may be a lost cause.

Lastly, I have started working the "BASIC" side of CP/M-86. I have included four files that I have either writen or converted to CB-86 BASIC. These are prefixed with the "@". I have included the BASIC SOURCE programs in the DISK.LBR so if you are interested in writing basic programs while in CP/M mode, you will have a starting place. I recently purchased an ACE computer and obtained a "MANUAL" on "CBASIC". I have used this exclusively to modify/write these programs. What I have found is that we can compile "CBASIC-2" programs and make them run on the B-128. Of course you must have CB-86.CMD and LINK-86.CMD progams to accomplish this process. I think both are available from Northwest Music. The interesting thing is that most CBM basic statements will work in CPM. (Who said Commodore didn't do their homework!
Of course there is always an exception. The "calc.com" series has no symobol in front. This is because this set of files requires the file names to be exact. Thus the files without any preceeding symbol will run under Z-80 emulation. Try thisone, it's a Hewlett Packard hand held computer program that runs on the B-128!

Here's A Summary:
"" - Files that should run but don't. (source file included)
"#" - Files that should run but don't. (source file not available
"+" - Files that run under the Z80 emulator.
"@" - Basic programs that run on the B-128.

You will also find a few ".cmd" files that will run under CP/M-86. These are files that you can use without any modification.

This disk set up is not the same as on my other disks. To be honest, I got a little carried away with what I wanted to send and the disk ran out of space before the programs. Instead of leaving some of the programs off, I have opted to "squeeze" the disk.lbr file. You can type "help+ disk" and still get a listing of the files that are in the disk.lbr file, but before you can access any of them you must unsqueeze the disk.lbr (disk.lqr) file. Sorry about the inconvenience. By the way, you will need to unsqueeze the disk.lqr file to another disk. Insert a blank disk in drive B (1), type control and c keys at the same time, place a disk with nusq.cmd on it in drive A (0) and type "nusq". When the screen prompt appears remove the disk from drive A and place this one in and follow the prompt.

We are in need of a lot of technical help in the CP/M world! If you have any experience in CP/M, please let us know. I have over 12MB of files that I can't get to run because I can't find any references on CP/M-86. If you have knowledge in this field, please let us know.

MSDOS is another field that we need help in. The 8088 CPU is still being used by many machines. It runs MSDOS 3.0 and higher, nice things here if you want them. But we are limited to MSDOS 1.25 with no capability to download from networks. Your assistance is needed to upgrade our systems to the new MSDOS. <<It is principally upgrading of software, not hardware shortfalls at this time.>>

HERE'S WHAT IS NEEDED:

1. UPGRADED MSDOS TO 2.0 OR HIGHER.
2. TELECOM PROGRAMS FOR CP/M-86 AND MSDOS.
3. MEMORY MAP OF THE 8088 CPU.
4. INDEPTH KNOWLEDGE OF HOW THE 8088 WORKS WITH THE 6509 CPU.
5. RS-232 INTERFACE COMMANDS.

Folks, it's to our advantage to get this CPU running with current software. Only you can do it! Let's all get together behind this one, there is no machine on the market today that can do what we can, but without your help, it's just another "orphan"!

The "bottom line" is, I've been working this side of the CBUG effort for about two years now along with Tony Goceliak, Dennis Jarvis, and Bruce Faierson with not too much support or many words of encouragement from any of the other members! If you think our time has been well spent, P L E A S E, drop us a line or give us a call and let us know. If you think the 128 is an orphan, you ought to think about the orphan running CPM with no moral or other type support.

You know, we don't even know how many 8088 co-processors are in use. Can anyone tell us who is running this operating system? If no-one is really interested in CPM or MSDOS (I think that would be a mistake), I will stop working this area and concentrate on basic CBM programs and utilities. Its up to you! Tell me if you like and use what we are doing and we'll continue, else, it's not worth the time and we'll do our CPM/MSDOS thing for our own use.

Think about it! DBII, Wordstar, Wordstar Professional, Supercalc, how about Lotus or Oracle, Symphony. Its all possible, all that's needed is a little effort from someone who is smarter than me.

ORPHAN <<John's handle on the Delphi network, home of CBUG's bulletin board service.>>

John A. Wright
Papillion, Ne.

P.S. This may be my last contribution for a few months. I am retireing from the USAF and must spend some time trying to find employment to support my ever so supportive family. I'll still be around (I think) so give me a call or drop a line.

-CATALOG OF CP/M-86 UTILITY DISK 005

PROVIDED BY:

LT COL JOHN A. WRIGHT
818 JUNIPER DR., PAPILLION, NE.
68046
TELE: 402-339-5728

```
********************************************************************************
********        NOTICE!!!:  ALL PROGRAMS COPYRITED, AND NOT       *********
********        RELEASABLE OUTSIDE CBUG, NOR RELEASEABLE FOR      *********
********        FOR COMMERCIAL PURPOSES!!!                        *********
********************************************************************************

005.01      INSTRUCT.HLP    5K   CA 89 DISK INSTRUCTIONS/HELP
005.02      HELP+   .CMD    6K   65 1E HELP PROGRAM
005.03      DISK    .HLP    1K   C4 6A DISK HELP FILE
005.04      +DEVICE .COM   10K   E7 E0 CHANGE DEVICE
005.05      +ALLOC3 .COM    2K   60 70 PRINT ALLOCATION BIT MAP
005.06      +COMP   .COM    2K   E6 39 COMPARE TWO FILES
005.07      +DCOMP  .COM    2K   B8 2B COMPARE TWO DISKS
005.08      +FILE21 .COM    1K   E7 D9 FIND A FILE
005.09      +LABEL  .COM    2K   73 BE LABEL A DISK AND PRINT A DISK LABEL
005.10      +NC     .COM    2K   01 D2 FILE COPY UTILITY
005.11      +SFILE13.COM    3K   81 B3 FIND A FILE
005.12      +TAIL   .COM    1K   6C 51 PRINT THE LAST XX LINES OF A FILE
005.13      +XZI    .COM    7K   24 41 Z80 TO 8080 CONVERT SOURCE CODE
005.14      +FMAP   .COM    3K   EE 36 FILE MAP
005.15      CALC    .COM   35K   A0 E6 HP CALCULATOR
005.16      EQN     .CLC    1K   D5 68   /
005.17      QUADRAT .CLC    1K   C4 C0  /
005.18      SIMPLE  .CLC    1K   C6 E2 /
005.19      SUM     .CLC    1K   AA D8 /
005.20      XLT86   .CMD    8K   D2 2E 8080 TO 8086 CONVERT SOURCE CODE
005.21      @TEST3  .CMD    9K   98 75 BASIC TESTS.
005.22      @TEST1  .CMD    6K   A6 06  /
005.23      @TEST2  .CMD    7K   D9 08 /
005.24      @FPRIME .CMD   10K   55 C1 PRINT OUT PRIME NUMBERS
005.25      ANYDISK.COM    10K   9A 09 EXAMINE/MODIFY DISK
005.26      #BDO4   .COM    6K   15 DC FIND AND LOCK OUT BAD SECTORS
005.27      #CERTIFY.COM   16K   71 77 ??????
005.28      #SHOBLK1.COM    3K   74 2C REPAIR/RECOVER FILES
005.29      #TIME11 .COM    5K   60 2B CLOCK
005.30      DISK    .LQR  192K   F4 CF SOURCE AND DOCUMENT FILES
005.31      Z8Q     .CMD    5K   D1 50 8080 EMULATOR

     SOFTWARE TOOLS RCPM - MISC VOLUME NUMBER - 005, 31 FILES CATALOGED.
```

<<This entire file was transfered using the Delphi telecommunications network as it was omitted from the disk in native mode!>>

HARDWARE BUG UNVEILED!

by: Gary L. Anderson

Vern Rotert, a member of our local chapter, the Silicon Corridor B-128 Users Group, first discovered and identified this sometimes intermittent hardware problem with his B-128. I have also seen its effects with the V-20 co-processor prototype installed in my B-128. I prefer to call it a hardware flaw on the B-128 main circuit board because the operating system software can be fooled by the hardware into thinking a particular memory configuration exists when in fact it does not!

The Symptoms:

The first problem that appeared for Vern was that none of the function keys were loaded with their info upon power up. Further investigation revealed that the top of DRAM memory pointer was wrong. A higher value was returned than what actually existed in the system. Since the function key data resides in the top DRAM bank and the particular top bank was not there as defined by the pointer it explained why the keys were not there.

Another problem that I saw was an intermittent co-processor crash during boot up. Upon furthur experimentation I discovered that the co-processor worked flawlessly when installed on the B-1024 1 Meg board but the intermittent boot reappeared when installed in a stock B by itself. Apparently the co-processor boot software also tests for memory, is it also getting confused with recognizing the amount of main board DRAM in the system? and why just main board DRAM? Hmmmmmmmm!

The Cause:

The initialization routines at power up test for the amount of DRAM memory in the system, Typically a memory location is written then immediately read to see if the write was sucessful. if it was then it does the same with the next location. It keeps doing this until it can't write sucessfully to a location then sets the top bank pointer accordingly. The problem is this: the write/read sequence mentioned above happens so quickly that the distributed capacitance on the unterminated DRAM data bus holds the charge from the write cycle well into the read cycle when no DRAM answers the address, so it reads back the same byte that was written. This fools the operating system software into thinking it sucessfully tested the memory location when in fact no DRAM resides there. An erroneous top of memory pointer gets set up and the function key info gets placed in a DRAM bank that does not exist.

This is no doubt the reason that so few of the Commodore co-processors have reportedly worked in the low profiles. Leakage currents from the DRAM chips and bus drivers, and distributed capacitance vary from machine to machine and board layouts differ from the low profile to the high profile making the situation complex.

The Solution:

Vern reports that the simplest way to fix this is to add a single 10K ohm resistor from one of the DRAM data bits, pick any one you want, to +5 volts. This effectively wipes out the effect of the distributed stray capacitance on one bit of the DRAM data bus and allows the operating system software to correctly determine the amount of DRAM memory. On both my B-1024 1 Meg board and now the V-20 co-processor prototype I terminate each bit of the DRAM data bus with a 10K ohm pull up to +5 volts and a 10K ohm pull down to ground. My V-20 co-processor prototype which didn't before terminate the DRAM data bus now runs flawlessly in a stock B-128 with the termination.

Gary L. Anderson
2560 Glass Road N.E.
Cedar Rapids, Iowa 52402

Super F Keys

by: Armand Carrier

You and everyone else at CBUG are doing a fantastic job. Congratulations and keep up the good work. Here's a little program you might enjoy (along with the rest of the CBUGGERS on this planet)!

If you're like me you must be sick and tired of typing filenames for everything you do, such as loading programs, loading & running programs, scratching files, renaming files, copying files.

Especially if you make a typing error!. The dreaded message '?file not found' and try again. If you're not sure of the spelling you can call up the directory to be sure of the spelling!. How clever!!. Since you have the directory on the screen why not just cursor up to the file and press a function key to do any of the above and let the computer do the work for you!!!. Wouldn't it be nice to be able to do all these things just by calling up the directory on the screen and cursoring up to a particular file in the directory and pressing a function key? This program is the answer!

The accompanying program will do just that for you with a minimum of keypresses. I save it to disk as the first file on the disk. Then, I just press shift/run-stop and the program loads and runs automatically. The directory of program files on drive 0 will be listed on the screen. I use this program all the time now instead of loading a menu program and then running it. (two for the price of one?). If you use this little program you will only have to do this once after powering up, after that a function key will do.

Now, you can load any program on the disk directory by moving the cursor up or down to the program you want and pressing function key 'F3'. This will load your program for you. That's all there is to it. If you want to load and automatically run it, just press function key 'F4'. That's it!

Those of you familiar with other computers, (nameless of course!), know that they have a delete key that works like a black hole. (everything disappears into it) We have a delete key. Same end result, just a different approach. Function key 'F5' will now act as a black hole for you.

Function key 'F6' gives you the option of viewing the directory of PROGRAM FILES only, available from either drive (assuming 8050).

Function key 'F7' will run the program currently in memory regardless of what is printed on the screen or where the cursor is located on the screen.

Function key 'F8' allows you the option of viewing the directory of all files on either drive (assuming 8050 drive).

Use the disk directory in the same way to scratch a file on drive 0. Just cursor up or down to it and press function key 'F9'. Again, if you want to copy a file from drive 0 to drive 1, just make sure the file you want to copy is on the disk in drive 0 and the disk you want to copy it to is in drive 1. Just cursor to the file and press function key 'F13' - (that's shift/F3).

I'm sorry but to rename a file you will have to type in the new name for your file. Otherwise follow the same procedure using the disk directory and cursor keys to rename a file. In this case use function key 'F12' - (that's shift/F2).

Function key 'F14' allows you to concatenate two sequential files. That is, you can add one file to another winding up with one larger file containing both of the previous two files. CAUTION!. I haven't had any problems with this to date but after using the above command, I usually wind up in the machime language monitor and typing 'x' to exit doesen't work! I usually get out by typing 'G 8000'. This gets me back to basic but I don't know if everything is right with the computer. (My machine language knowledge is slightly less than NOVICE!). Perhaps someone at CBUG can let us know if we can avoid this problem (if it is a problem) or if we are all right doing it this way.

Function key 'F15' gives you the option of viewing the directory of SEQUENTIAL files only, available from either drive. (assuming 8050).

Function key 'F16' allows you to read a sequential file. When using this key, just be sure the program in the computer does not have a line #5 or less. If it does and you then dsave the program, this program will be your new line #5. (This is actually a one line program). If you want to break out before reaching the end of the file, press stop. At this point, the file you were reading will still be opened. (the red light on your disk drive will be lit!) Now, you must type dC or dclose to close the file properly or your data in the file may be scrambled and useless. The disk drive light should now be out.

Fuction key 'F17' sets the printer to uc/lc print or you can change the printer settings by changing the sevens to whatever you need.

Function key 'F18' will give you a printout of all or part of your program listings with an enlarged print title centered at the top of your listing. I use this key in the following manner. Use 'F17' first to set the printer to lowercase/uppercase mode. Type on screen in direct mode f$="filename" where filename is the name of the program in the computer memory. Then, press 'F18' and press return for a printed listing of the program. (If you only want sections eg. lines 100 to 200 then just add this to the end of the line after the word 'list' 100-200.) That's it!

This may be the best plus of all. You can do all of these things so much easier now plus, it will not disturb what you are currently working on (or have loaded into the computer). There is no need to quit what you are working on and loading another program to do any of these things. Just list the directory do what you want and continue where you left off.

That's all for now Norm, keep up the good work everyone!:

Sincerely,
Armand Carrier
20 Cole St.
Torrington, CT. 06790

<<The program refered to on disk in this issue's CBUG Library.>>

B/128 Super Disk Sweep

By: James D. Springer

This program allows the users of our Fast Bus product to transfer files to and from several diskette formats. The disk formats that are currently supported are as follows:

1. Standard CBM
2. C/128 CP/M Single and Double sided (Must have a 1571 for Double sided)
3. B/128 CP/M 8050 Disk Drive
4. C/64 CP/M
5. MS-DOS 8 Sectors Single and Double sided (Must have a 1571 for MS-DOS)
6. MS-DOS 9 Sectors Single and Double sided

The program allows conversion of files to ASCII, PET ASCII or no conversion. If the transfer is being made from a CBM format to MS-DOS oor vice-versa the program automatically adds or removes linefeeds from the file. The program is easily operated by following the menus and prompts throughout the program. Transfers from single drives, dual drives or drives on different busses are fully supported.

<<CBUG is awaiting receipt of the master disk for this issue. We hope it will arrive in time for the Library preparation and printing -- otherwise next issue.>>

B SERIES DISK BACKUP

By: Dennis Jarvis

This program is geared toward those people that own any of the following disk drives.

2031
4040
1540
1541
1571

With this progrm you have a full disk backup program and a BAM copier all built in one. This program will enable you to transfer data back and forth between any of the aforementioned disk drives.

Currently this is the only program I know of that will allow you to copy a diskette from the 4040, and place it on a 1571 diskette using BURSTREAD, and BURSTWRITE during a full disk backup.

This program will allow you to copy your 4040 disk and place it onto the new 1571 diskette and still keep it a double sided disk. The program will ask you to ENTER your SOURCE DEVICE NUMBER: at this point you need to enter your disk drive's device number that the disk you want to copy is in. Next it will ask you to ENTER DESTINATION DEVICE NUMBER: This is asking you what the device number you wish to copy this disk to. If you only have one disk drive then it would be the same number, for example if you have a 2031 disk drive on the IEEE bus and 1571 on the serial bus then all you would have to do is to place the IEEE/SERIAL bus switch (on your fast bus board) in the IEEE position, then enter 8 as the source device number and 28 for the destination device number.

Next the program will inform you if the device can or cannot use fast bus, and what the device's model number is (the model number must be one of the above disk drives, the 8050/8250 devices are not supported.)<<See copy utilities from Goceliak and from Deal for 80xx/SFD programs>>. If the destination device number is 1571 then this program will give you a very powerful option. This option is the first of its kind.

Before I tell you any more about this option I need to tell you a little about how your disk drive operates. As each sector is written to your disk, the Diskdrive Operatiing System (DOS) will read that sector back in and make sure it was written correctly.

This is where the option comes into play, it allows you to turn off this operation so that the DOS does not verify each sector after it is written. I'm sorry but this option can only exist in the 1571 in 1571 or in 1541 mode.

In the next issue of CBUG as well as in a future issue of TRANSACTOR the source code for this option will be released to PUBLIC DOMAIN along with the theory on how it operates so stay tuned for more info.

There is one WARNING I must give you if you do use this option. In turning off the 'WRITE then VERIFY it' operation, you may be playing with fire. I have been using this option for 1 year and have not had any problems, but don't get me wrong it could give YOU problems. Why you and not me? Well as the old saying goes - a chain can only be as strong as it's weakest link - this saying holds true for any software package of this type because;

1) Your 1571 could be going bad

2) Your diskette was just blown a way.

3) A timer chip just blew up

Even though most of the aforementioned items are not likely, they are possible. If any of these were true then you would have encountered errors by now, but there is a possiblility that one of the above problems just occured after you selected the option.

With the WRITE VERIFY disable option installed it cannot be turned off with out doing one of the following operations to your disk drive.

1) Turn the disk drive off then on.
2) Send out one of these commands over the command channel (SA=15);

```
OPEN 15,DV,15,"U0>M0"

OPEN 15,DV,15,"U0>M1"

OPEN 15,DV,15,"U:"
```

This program will make several checks and print out any error messages or even ask you some questions from time to time. This questions are actually more options for you to select and are very straight forward. For example let's say you place a 1571 double sided disk in a 2031 disk drive. Since the 2031 disk drive can not read a double sided disk this program will inform you of the fact and ask you if you would like to copy it any way. Why give you this option? Simple, let's say you only have a couple of files on this disk and want to back it up. Since the 2031 (or 4040) is not capable of reading tracks 36 thru 70 it allows you to copy the disk if you want to (I would prefer the computer works for me insted of me working for the computer).

This program also takes advantage of your new FAST BUS ROUTINES Jim and I have developed, It uses our BURST READ/WRITE, BLOCK READ/WRITE routines. By using the BURST routines with the 1571 disk drive it will greatly speed up the coping process. Another advantage to this program is it takes full advantage of the different bus types when possible. For example if your source disk is a 1571 and your destination disk is a 4040 this program will start the formating process on the 4040 disk drive then go ahead and start reading the 1571 disk. On a single sided diskette the all of the track and sectors are read in from the 1571 BEFORE the formating process is completed on the 2031 or 4040 diskettes.

In talking with other members of our group, I found out that their are not many users that have a 2031 or 4040 disk drives. If you do have a 2031 or 4040 disk drive then let us know so that we will continue to support them in the future. For example how would you like a program that would allow you to format a disk in about 10 seconds in your 4040, or 2031 disk drive???

To start up this program all you do is have to load up the file called RUN ME then type RUN to start the program.

Well that's it for now, again use this program on some junk disk's until you get the feel for how it operates. Until next time...

Turning off the WRITE/VERIFY operation with the 1571 disk drive

By: Dennis Jarvis

Before I get into the actual operating system I would like to give some background on the 1571's operating system.

When COMMODRE released the 2031 disk drive it included Disk Operating System 2.6, which was a single drive unit. DOS 2.6 thus became the standard for single disk drive units. This drive is what the VIC-1540 was based on. The IEEE bus was reduced to a SERIAL bus, and the disk formating method was changed (GAP1 was increased by 1 on the 1540). This is why the 4040 and 2031 disk drives are READ compatable with the 15xx series of disk drives but are NOT WRITE COMPATABLE. For more information on this read/write problem consult the book INSIDE COMMODORE DOS by RICHARD IMMERS and GERALD G. NEUFELD on page 208.

When COMMODRE released the PLUS 4 computer they included a SERIAL bus for the existing serial bus devices, but they also included a new type of bus called the TED bus. This bus was a cross between the serial bus and an IEEE bus with 8 data lines and a few handshake lines. When COMMODORE released the 1551 disk drive (also known as the 488 disk drive) they (DAVID SIRACUSA) made several changes to the operating system, which included a FASTER GCR to BINARY conversion, a FAST FORMAT routine, and corrected some old bugs that were in the BLOCK READ, WRITE routines. Oddly enough this DOS was released as DOS 2.6 which is the same DOS version that is still in the 154x disk drives, even though their memory maps are no where near the same. COMMODORE's logic on numbering their DOS versions are still not very clear to me yet!

COMMODORE (DAVID SIRACUSA) based the new 1571 disk drive on the 154X's memory map. What DAVID did was to remove alot of the code he installed in the 1551 disk drive and splice it into the 1571 (such as the new GCR routines and the faster formating code etc.) I only wished COMMODORE had released the 1571 with a SERIAL bus and an IEEE or 1551 type of bus which would have increased the overall speed of the drive if the DOS was recoded corretly. To wrap up all of the above, the only drives that are both read and write compatable are the 4040 and 2031 disk drives, and the 15xx series of drives. Of cource you can't put a 1571 disk in a 15xx and read side 1 as you don't have the head to do it with. You can read disk's between the 15xx, 2030, and 4040 but to not write back to them ... or you may regret it.

One of the changes made was in the 1571 DOS was the method in which it handles the IRQ's. Instead of always jumping to a CONSTANT memory location (address), it jumps thru a vector the same way that your computer's KERNAL jumps thru the ICHROUT vector to output a character to the screen/printer etc. Since this is the first disk drive to have this ability, such programs as this were not possible before with out a major effort on a programer. With this new vector in your disk drive it will give you and me many options to persue -- such as the one were doing here.

Normaly when ever a sector is written back to a disk by all the current disk drives (2030 thru the current 1571), it will first write the sector to the disk then read that sector back in off the disk to verify it against what it has in RAM to insure it was written correctly. Normally you will never see an error occur unless you have a bad diskette or a hardware failure in the disk drive or serial bus. Normally a bad disk is detected during the format process; but disks do fail just by sitting in your disk box from time to time.

In either case you won't see the problem very often. What this program does when installed in the 1571 is to 'WEDGE' itself into the IRQ vector to allow my program to check to see if the DOS is about to do a VERIFY operation and if it is I replace it with a SEEK operaion. If you don't understand any of this don't worry about it as it is not required to know HOW this program works to use it.

This program could be considered RISKY to some degree and it probably is RISKY, but I've been using this program daily for 3 months with NO problems encounterd to date. Do I guarantee this program? NOT ON YOUR DISK I DON'T. Why you might ask? Well, to put it simply their are too many variables involved from your disk drive having the flu, to disks that are used as floor mats. It would only take one smudge to wipe a disk. On any account simply run

this program and give it a try by doing some testing on a JUNK DISK you know the one, it's the one you don't care if your dog or cat eats, or JR. uses it as a frizby! What i'm trying to say it that I've used this program for quite a while with NO problems on several diffrent versions of the 1571 Disk Operating System, but this choice is up to you to make not me.

This short program allows you to turn off the write verify operation in your 1571 disk drive even while it's in the 1541 mode. Currently there are several versions of this program runing around on public domain bulletin boards but several of them contain bugs because they, like mine sit some where in $0100 which is the disk drive's stack. Most of the ones I have seen to date use up too much room on the stack and apon the first DOS error - crash - the drive dies. The following program has been tested, tested again then tested more...

To use this program all you need is a 1571 disk drive and this program. This program runs on ANY COMMODORE computer form the VIC 20, to the C-128. To start this program just load it up and type run. The program will ask only ONE question, and that is simply ENTER THE DEVICE NUMBER OF YOUR 1571 DISK DRIVE: this value can be in the range of 5 thru 30, and would be the same number that you would enter for such a BASIC command as DLOAD"FILENAME",U(XX) or LOAD "FILENAME",XX where XX is the device number of the 1571 disk drive. If the program finds that you indeed have a 1571 disk drive at that device number then it will proced with it's assigned task. If there is NOT a 1571 online then it will display an error message. When the program does indeed turn off the write verify operation it will inform you by displaying "WRITE VERIFY OPERATION IS NOW TURNED OFF!"

During the course of this program I make several checks to ensure this is realy a 1571 disk drive. First I check the IRQ vector at $FFFE to ensure it contains an $FE67, this will ensure that it is a 154x or 157x type of drive. Next I check $8002 for the text 'S/' to ensure it is a 1571 disk drive. Finaly I check $02A9 which is the IRQ vector for the 1571 disk drive. When I check this vector I ensure the MSB of the address contains a value greater than $80 to ensure the te IRQ is going somewhere into ROM, and not to an applications program such as mine. If all of the aforementioned are true then I down load the M/L into the drive and execute it.

<<The program refered to is reproduced in printed form at the end of this issue, and is available on disk, both program and sequential files, in this issue's CBUG Library.>>

MAKE 2 SIDED

By: Dennis J. Jarvis

This program was written to solve two problems I was having with the 1571 disk drive. They were:

1) Converting a single sided diskette to a double sided one;

2) Doing a COLLECT on a 1571 disk while it was in 1541 mode, causing it to be reverted back to a single sided disk. (NOTE: This problem has been corrected in the newer 1571 disk drives).

The first problem can be a pain in the keyboard. What was required was that I use the DOS SHELL or other such program and copy the FILES over to a newly formated 1571 diskette. I could not use any full disk copiers because they would copy track 18 and 0 from the single sided disk to the double sided disk reverting it back to a single sided disk (as far as the 1571 disk drive was concerned that is). Well that got to be a time consuming operation. So I sat down to find an easier way. Using my 1571 INTERNALS book from ABACUS SOFTWARE, I began to scan through the FORMAT routines. As it turned out when DAVID SIRACUSA wrote the format routine, he set them up as a two pass operation. First, he formats all of SIDE 0, then returns to format SIDE 1.

Before I go into any further details on how to use the format routine I would like to tell you how the 1571 checks to see if it's got a double sided diskette in the drive or not. When you perform most disk functions, the Disk Operating System (DOS) reads track 18 sector 0 into the disk drives RAM. Then checks the third byte over (with the first byte starting with 0) and checks to see if it has bit 7 set or not. If it is not set then, It is a single sided disk. If bit 7 is set it will read in track 53 sector 0 to read the rest of the disk's Block Availability Map (BAM). This information is used to find out if a sector, on a specific track, is in use or if it is free on side 1 of the disk.

As you can see, before the DOS will allow you to read a track beyond track 35 using the normal DOS, it needs to see bit 7 set in the third byte of track 18 sector 0. In my program it will always set bit 7 in the third byte of track 18 sector 0. After this has been done I send out an I0: which does a couple of things. First, it forces the disk drive to reload track 18 sector 0 into its ram (the BAM for side 0) and set the internal single/ double sided disk flag. Secondly it will force the DOS to read in this disk's formating ID's. Whenever you format a disk such as "N0:COMMODORE 1571,DJ" it will make the disk's name COMMDORE 1571, and place the formating ID's in each header preceeding each sector. In this case the formating ID's would be DJ. These ID bytes are placed along with the name on track 18 sector 0. These can be changed by various software programs, sucn as HEADER CHANGE which is on you 1571 test/demo disk --- which was given to you when you bought your 1571. If you change the NAME of the disk or the DISK ID's on track 18 sector 0, you are just changing the information that you would see when you load/read in the DIRECTORY. The actual formating ID bytes can not be changed with out reformating the disk from scratch.

Once the disk has been set up as a double sided disk, and has read in the disk ID bytes, we're ready to go to work. Next I set the read attempts to 1 to prevent the disk from searching too long so it does not waste your time, then I attempt to read track 53 sector 0. Track 53 sector 0 could have been any track on the second side of the disk. If I'm able to read in this track and sector, then I know that the disk was previously formated as double sided disk, but was reverted back to a single sided disk due to problem 2 stated above, and all I have to do is to perfrom a collect command on the disk to reconstruct a valid (correct) BAM. If I'm unable to read track 53, sector 0 then I know that side 1 was never formated, and I need to format it.

Note that this program does NOT convert flipies. Flipies are those disk's people HOLE puch, or notch the FLIP side of the disk so they can FLIP it over and use the other side!! In order to format side 1 only, all I have to do is to enter the formating routine at $A445 in the 1571 disk drive. Once this is done the disk drive will go on it's way formating side 1 ONLY of this disk. If no errors have occured all I need to do is to perform a collect on the disk so that it will create a BAM for the second side of the disk. Once this has been completed you now have a double sided disk with SIDE 1 empty of all files.

Even though this program may sound straight forward, IT IS NOT! You must allow my program to complete it's task or it will cause you problems in the future. Say for example, you have a single sided disk that you want to convert to a double sided disk, you get all the way up to the point where

the disk was just about to be formated (SIDE 1 only of course), and you changed your mind and pulled the disk out of the disk drive. If you do not rerun this program and select no, when it asks you if you want to format the disk, you and your disk now have a big problem on you hands. Since my program has already set the double sided flag so that I could read side 1 of the disk, so the next time you do anything to that disk such as saving or loading a program the 1571 sees the double sided flag set and tries to read track 53, sector 0 to read the rest of the bam in and bang, a read error because there is NO TRACK 53, SECTOR 0 because it was never formated!

To use this program all you need to do is to type it in or obtain a program disk, then answer the questions. They are very straight forward questions such as inputing the device number of you 1571. This program has been tested and operated with out any problems on a VIC-20, C-64, PLUS4, B-128 (with our fast serial bus installed of course), and the C-128 line of computers. I made sure that I only used BASIC 2.0 command to allow everyone that owns a 1571 a chance to use this program. This program has been tested, tested, tested, then tested some more, this program has already been release via the CBUG user's group for the B-128 computer with no complaints thus far.

As with any peice of software there is room for change and growth, but this is a functional program with out any bells and flags. If you wish to change this software, make sure you know EXACTLY what you are doing or you could erase the wrong side of the diskette or find your self with a new dent in your 1571's outer case when you told you disk drives head to take a quick trip to MARS and back.

When this program was sent in it was over 4 pages long mostly because of single line statments, and massive amount of comments. Probably the comments will be removed, and the program shortened due to type space. A couple last comments about track 18 sector 0. If you look at the last few bytes starting at byte $DD you will see the number of sectors free on each track on SIDE 1. I'm not sure but in my opinion, DAVID did this to allow us (programers) a way of seeing if any tracks are in use on the other side of the disk. When I wrote a full disk copier for CBUG, I used this method to see quickly if any sectors were in use on SIDE 1 of the disk. This would give them the option of using a 154x, 2031, or 4040 disk drives as the source disk and the 1571 disk drive as the copy disk. The only draw back I can see and have been affected by is that my normal method of reading in a disk's BAM, and directory no longer works. If you open the directory up for READING (not load"$",8 etc) you would be able to read in the disk's bam at the same time. On the 1571 it WILL NOT SEND you the BAM for SIDE 1, you have to go get it your self with a U1 command. One last comment about the format ID byte in the bam also. This byte (byte 2 starting with byte 0) is a $41 on the 4040, 2031, and 15xx drives. This byte has been confused over the years as the format type byte. In my OPINION this tells you the number of sectors per zone and is layed out as follows;

DRIVE	ZONE 1	2	3	4	FORMAT TYPE
2040	20	19	17	16	$31 or $A0
3040	"	"	"	"	"
1540	20	18	17	16	$41
1541	"	"	"	"	"
1551	"	"	"	"	"
1571	"	"	"	"	"
2031	"	"	"	"	"
4040	"	"	"	"	"
2030	(DO YOU KNOW?)				$42
8050	28	26	24	22	$43
8250	"	"	"	"	"

All of the above table is of my own creation and opinion. Take the FORMAT TYPE $41 for example. On the 154xx and 1551 they are all read, write compatable and are identical in about every way and are currently formated exactly the same. On the 4040, and 2031 drives, their is an extra GAP1 byte in the format process as compared to the 15xx drives. Due to this extra byte they are not write compatable with the 15xx. On the 1571 it is double sided drive with a diffrent BAM sceem when compared to the the 2031, 4040 and 154x, and 1551 type drives. Another words this byte DOES not indicate the number of tracks on a disk, nor the formating method used to format the disk as your are led to believe in the disk drives manual. One interesting point, on all of the COMMODORE disk drives, the DIRECTORY routine defaults back to the 2040 disk drive which was VERSION 1 as the format type, on the 4040 if you change the 2A on track 18 sector 0 to 2 followed by and $A0 it will show it as 21, after the disk name! This holds true for all on the COMMODORE disk drives, including the 8050, 8250 drives as well!! (change the 8050, 8250 form 2C to 2, followed by an $A0).

If anyone has any input on my opinion as stated above please send it into CBUG, I would like to hear from such people as FRED BOWEN, JIM BUTTERFIELD, LIZ DEAL, JESSI KNIGHT, ANTHONY GOCELIAK, DAVID SIRACUSA or from anyone wanting to make a comment! All of the aforementioned programers have been programming on various COMMODORE disk drives for many years, and have far more hands on time than I do. I'm still just a beginer just like must of us.

Hopefully in the next month two I will have obtained the time to clean up my comments, in my source code for my FAST FORMAT for the 154x, and 1571 (in 1571, or 1541 mode), and will be able to get them off to CBUG and to you. Until then keep those disk drive's spinning.

<<The program refered to is reproduced in printed form at the end of this issue, and is available on disk, both program and sequential files, in this issue's CBUG Library.>>

By: Dennis J. Jarvis
Kissimmee, Florida

Notice

This space was left open due to an oversite on the part of the general membership who forget this is a volunteer organization and all members are expected to contribute to its operation.

MAKE 2 SIDED

By: Dennis J. Jarvis

```
 10 rem ************************************************************************
 20 rem *                                                                      *
 30 rem *     this program is used to convert a double sided diskette that     *
 40 rem * was collected on the 1571 disk drive while it was in the 1541 mode   *
 50 rem * which will automaticaly convert a diskette back to a single sided    *
 60 rem * diskette. this progam will also convert a diskette that was formated*
 70 rem * on a 1541 to a double sided 1571 diskette.                           *
 80 rem * to uses this program just place the diskette into the 1571 drive and*
 90 rem * this has happened to in the drive and run this program, and if this *
100 rem * problem holds true then this program will correct the problem.       *
110 rem *                                                                      *
120 rem * warning: if for any reason any errors occur or if during the        *
130 rem *          execution of this program you should change your mind      *
140 rem *          just answer (n)o to the questions, if for some reason      *
150 rem *          (such as a write error or a power failure) this program    *
160 rem *          does not finish it's task you must rerun this program or   *
170 rem *          serious problems could occur with your diskette!.          *
180 rem *                                                                      *
190 rem *  (c) 1988, 1989    by dennis j. jarvis  publications rights reserved*
200 rem *                                                                      *
210 rem ************************************************************************
220 a=40 :rem  change this to your computers screen size 40, 80 etc.
230 sc=a/2:printsc
240 print"ssS":rem clear the screen and delete any current windows (if any)
250 dv=8
260 p$="enter the disk drives device number":rem string to be printed
270 row=12             :rem row to print the string on
280 gosub1820          :rem print it to the center of the screen
290 printdv;"]]";      :rem display the default drive number
300 open5,0            :rem prevent question mark from being printed
310 input#5,a$         :rem allow the user to change the current device number
320 close5             :rem allow proper screen prompting
330 dv=abs(val(a$))    :rem make the device number positive
340 ifdv = 0 then print"S":end:rem their done now
350 ifdv>5  and   dv<31 then 430: rem if the device number is in range branch
360 print"S"         :rem if it's not the clear the screen and print the message
370 p$="illegal device number " :rem to the screen
380 rv=1               :rem set the reverse field on flag
390 row=12             :rem print it in the center of the screen
400 gosub1820          :rem print it to the screen
410 gosub1700          :rem wait for the user to acknoledge the error
420 goto250            :rem restart the input loop
430 open1,dv,15,"u:" :rem soft reset the disk drive to obtain model number
440 forx=0to2000:next:rem give the drive time to finish it's reset process
450 gosub2050          :rem read in and save disk drives model number
460 print#1,"u0>m1":rem just incase were running on a non fast bus computer
470 print#1,"u0>r"+chr$(1)
480 gosub2080          :rem close down the command channel to the disk drive
490 ifleft$(right$(e$,9),1)="7"then o=1:goto510
500 o=0
510 ifothen570         :rem if it's a 1571 online then bypass the error message
520 print"S":p$= "sorry for the 1571 disk drive only":rem string to be printed
530 row=12             :rem print it on this row
540 rv=1               :rem print it in reversed field
550 gosub1820          :rem print it to the center of the screen
560 gosub2080:forx=0to3000:next:run
570 gosub1660          :rem open the proper channels to the disk drive
580 fl=128:gosub1230 :rem set the flag for double sided disk
```

```
590 print#1,"u0>r"+chr$(1):rem set read attempts to 1
600 print#1,"u1:2 0 53 0" :rem attempt to read second side of the disk
610 gosub2050:d=e         :rem preserve the error channel
620 print#1,"u0>r"+chr$(5):rem reset number of read attempts
630 if d = 66 then 810    :rem if unable to read the second side of the disk
640 rem                        then branch to see if they want to format it
650 rem                        d will equal 66 ( illegal track and sector if
660 rem                        it's unable to read the second side of the disk)
670 gosub2080             :rem close all open channels to the disk drive
680 gosub2090             :rem make the drive reconize that this disk 2 sided
                           * disk and have it read in the disk right$str$'s
690 print"S
700 p$="disk has now been restored to a   double sided diskette"
710 row=11:gosub1820
720 p$="i am now performing a collect on  this disk please wait..."
730 row=13:rv=1:gosub1820:gosub2080:gosub2090:open15,8,15,"v0:":close15:print
    "S":end
740 rem *-------------------------------------------------------------------*
750 rem *  if we are unable to read the second side of the diskette then we  *
760 rem * give the user the option of formating only the second side of the  *
770 rem * disk, if they do then we will format only the second side of the   *
780 rem * disk, or if they do not then we restore the single sided flag on   *
790 rem * the disk and terminate this routine.                               *
800 rem *-------------------------------------------------------------------*
810 print"S
820 gosub2080
830 p$="sorry this diskette was not formatted as a double sided diskette."
840 row=10:gosub1820:p$=" would you like to make it one? (y/n) ?":row=12:rv=1:
    gosub1820:gosub1920:
850 print"S
860 p$= "are your sure! this will earse the second side of your disk? (y/n)"
870 row=13
880 gosub1820
890 gosub1920     :rem get their response if it's 'y' then return
900 gosub1410     :rem format the second side of the diskette
910 goto690       :rem inform the user and perform a collect on the diskette
920 gosub2080     :rem close down all open files
930 gosub1660     :rem open the proper channels to the disk drive
940 fl=0         :rem set the flag for a single sided diskette
950 gosub1230    :rem reset single sided diskette flag
960 return
970 rem *-------------------------------------------------------------------*
980 rem *   this routine is the error handler. if any disk errors have       *
990 rem * occured ,other than the drive being off ,the routine outputs the   *
1000rem * error message and resets the double sided/ single sided diskette  *
1010rem * back to a single sided diskette. note that this is very important!! *
1020rem *-------------------------------------------------------------------*
1030 if (st and 128) then1130 :rem check for the disk drive being turned off
1040 gosub2050:ife<2 then return:rem if theirs no disk drive errors then return
1050 print"S":open5,0
1060 p$= "disk error has occured - ":row=11:r=0:gosub1820:gosub2050:p$=e$
     :row=13:r=1:gosub1820:print:goto920
1070rem *-------------------------------------------------------------------*
1080rem *   this routine is used by the trap command to detect such errors   *
1090rem * as syntax, or the disk drive being turned off, if the disk drive   *
1100rem * is turned off then it prints a warning to the screen and restarts  *
1110rem * the program. if any other error occurs then the program is aborted. *
1120rem *-------------------------------------------------------------------*
1130 p$=" turn on your disk drive ":rv=1:row=12:gosub1820:gosub1700:run
1140rem *-------------------------------------------------------------------*
1150rem *  read in the diskettes bam into the buffer and set the flag to     *
1160rem * indicate this disk is double sided to enable us the read the second *
```

```
1170rem * side of the disk, and if we are able to read it then this diskette  *
1180rem * was formatted as a double sided disk, but if we are not able to     *
1190rem * read the second side of the diskette then it was formatted as a     *
1200rem * single sided diskette so reset the flag back to indicate that this  *
1210rem * is a single sided diskette and end this program.                     *
1220rem *-----------------------------------------------------------------------*
1230 print#1,"u1:2 0 18 0" :rem read in the diskettes bam
1240 gosub1040                  check for read errors
1250 print#1,"b-p 2 3"     :rem set the pointer to the double side indicator
1260 print#2,chr$(f1);:rem set the flag to the value for single or double sided
1270 print#1,"u2:2 0 18 0" :rem write the new bam back to the diskette
1280 gosub1040                  check for write/read errors
1290 print#1,"i0:"             :rem force drive to read it's new bam in
1300 gosub2080                 :rem close down our channels (1571 just did!)
1310 gosub1660                  reopen the channels
1320 return
1330 :
1340rem *     this routine is used to format the second side of the disk      *
1350rem * that is currently formatted as a single sided disk. to do this a     *
1360rem * routine that is in the 1571 disk drive is used, which is in the      *
1370rem * normall formatting process, to use this routine we place the max.    *
1380rem * number of tracks on this disk (71), into $02ac, and jump to the     *
1390rem * format routine ($a445) to format the second side of the disk         *
1400rem *-----------------------------------------------------------------------*
1410 a = 284        :rem maximum track (located in the 1571's memory at $02ac)
1420 b=71           :rem maximum track number-1 to format too
1430 open1,dv,15    :rem open the command channel to the disk drive
1440 gosub 1570     :rem check for any errors
1450 print#1,"m-w"+chr$(a and 255)chr$(a/255)chr$(1)chr$(b):rem set max trk
1460 print"S
1470 p$="the formatting process has now being performed on side 1 of
       your diskette"
1480 row=11:gosub1820:print#1,"m-e"chr$(69)chr$(164):gosub1570:close15.
     :gosub2080:return
1490rem *-----------------------------------------------------------------------*
1500rem *   this rutine will check to ensure that their are no errors during  *
1510rem * the format process, such as the user 'poping' the diskette out of    *
1520rem * the disk drive, or any write errors, etc. if any occur then this     *
1530rem * subrouting will terminate the formatting process and reset the       *
1540rem * single double sided flag back to a single sided diskette.            *
1550rem * note: read the warning in the first set of rem statements.           *
1560rem *-----------------------------------------------------------------------*
1570 gosub2050:ife<20then return :rem if no disk drive errors have
       occured then return
1580 print"S":close5:p$="a disk drive error has occured ":row=9:rv=0:gosub1820
       :p$=e$:row=11:rv=1:gosub1820:p$="please check your disk drive "
1590 row=13:gosub1820:gosub1700:gosub920:goto810
1600rem *-----------------------------------------------------------------------*
1610rem *   this routine  will open the command channel to the disk drive     *
1620rem * whose number is specified in dv, and this routine will also reserve  *
1630rem * a buffer for our use inside of the disk drive to allow us to read    *
1640rem * and write the bam to/from the disk drive                             *
1650rem *-----------------------------------------------------------------------*
1660 close2:close1 :rem ensure channels are closed
1670 open2,dv,2,"#":rem allocate a buffer for our use in the disk drive
1680 open1,dv,15   :rem open the command channel to the disk drive
1690 goto 1570     :rem check for any disk errors then return
1700 p$= " press any key to continue ":rv=1:row=row+2:gosub1820:gosub2040
1710 geta$:ifa$=""then1710:dsaveprint"S":return
1720rem *-----------------------------------------------------------------------*
1730rem *   this routing is used to print our text centered on the 40 or 80    *
1740rem * column text screens.                                                 *
```

```
1750rem *                                                               *
1760rem *  to use this routine you must predefine the following variables.  *
1770rem *                                                               *
1780rem *  p$  = the string you wish to print centered onto the screen   *
1790rem *  r   = reverse feild on (1), or off (0)                       *
1800rem *  row = row number to  print the text on                       *
1810rem *-------------------------------------------------------------*
1820 nl = len(p$)/2 :rem find the true length of the string
1830 print"s":forx=2torow:print:next
1840 fori=1toabs(sc-len(p$)/2):print"]";:next:ifrv=1thenprintchr$(18);
1850 printp$;:rv=0:return
1860 rem*-------------------------------------------------------------*
1870rem *   clear the keyboard buffer of any key presses and wait for the  *
1880rem * user to press a key and if it's the key 'y' then return to the  *
1890rem * calling routine, else reset the flag in the bam to indicate this *
1900rem * disk is a single sided disk, not a double sided diskette then end *
1910rem *-------------------------------------------------------------*
1920 gosub2040   :rem purge the key board buffer of any outstanding key presses
1930 getz$:ifz$=""then1930  :rem wait for the user to press a key
1940 ifz$="y"then return
1950 gosub920       :rem set the flag for a single sided diskette
1960 gosub2080      :rem close any open disk drive files
1970 print"ssS"     :rem clear the screen and terminate the window
1980 end            :terminate this program
1990rem *-------------------------------------------------------------*
2000rem *   generic purge routine for all cbm computers. this subroutine will *
2010rem * remove all characters entered up to this point.               *
2020rem *-------------------------------------------------------------*
2030 getzz$:ifzz$<>""then2030   :rem any more keys in the buffer? if so loop
2040 return                     :rem buffer now purged
2050 e$=""
2060 ifstthen e=val(left$(e$,2)):return
2070 get#1,a$:e$=e$+a$:goto2060
2080 close2:close1:return
2090 open1,dv,15,"i0:":close1:return
          THE END       copyright by Dennis J. Jarvis
```

VERIFY DISABLE

By: Dennis J. Jarvis

```
 10 rem publication rights reserved
 20 printchr$(142)"ssSqqqqqqeturn off write verify operation for"
 30 print"     commodore's 1571e disk drive
 40 print"q    (c)e 1988, 1989
 50 print"q             by
 60 printchr$(15)"q    dennis j. jarvis"chr$(143)
 70 print"q    kissimmee florida"
 80 printchr$(7)chr$(15)"qq  Zpress any key to continue"chr$(142)
 90 geta$:ifa$<>""then90:rem purge buffer of any previous key strikes
100 geta$:ifa$= ""then100:rem wait for a key to be pressed now
110 print"S"
120 input"ewhat is the 1571's device number  8 ]]]]";a$:dv=val(a$)
130 ifdv<5ordv>30then120
140 print"S"
150 open15,dv,15
160 close15:ifstthenprint"Sqqqq]]]] device number:"str$(dv)" is not turned on"
      :end
170 open15,dv,15
```

```
180 lsb=254:msb=255:gosub350:rem read the rom irq vector ($fffc)
190 iflsb=103andmsb=254then220
200 print"Sqqqqq]]]]]]sorry this program will only work on the 1571 disk drive"
210 close15:end
220 lsb=002:msb=128:gosub350:rem check text at $8002 's/w - david g. sir...etc
230 iflsb<>83ormsb<>47then200
240 lsb=169:msb=002:gosub350:rem make sure irq is pointing to a routine in rom
250 ifmsb<128thenprint"Sqqqqq sorry some one is using the irq vector":goto210
260 fori=2to50:read h$:a$=left$(h$,1):b$=right$(h$,1)
270 gosub300:b=a*16:a$=b$:gosub300:c=a+b
280 print#15,"m-w"chr$(i)chr$(1)chr$(1)chr$(c):next
290 print#15,"m-e"chr$(23)chr$(1):print"Sqqqqq]]]]] write verify now turned
      off":end
300 ifa$<":"anda$>"/"thens=48:goto330
310 ifa$>"@"anda$<"g"thens=55:goto330
320 print "invalid hex byte -"h$":stop
330 a=asc(a$)-s
340 return
350 print#15,"m-r"chr$(lsb)chr$(msb)chr$(2):get#15,a$,b$:lsb=asc(a$+chr$(0))
     :msb=asc(b$+chr$(0)):return
360 data  48:                 :rem pha          save acc.
370 data  8a:                 :rem txa
380 data  48:                 :rem pha          save x reg. onto the stack
390 data  a2,05:              :rem ldx #5       number of job queue's to check
400 data  b5,00:              :rem lda $00,y    get current job out of queue
410 data  c9,a0:              :rem cmp #$a0     is it a verify command?
420 data  d0,04:              :rem bne *+15     no then branch
430 data  09,10:              :rem ora #$10     replace the verify function
440 data  95,00:              :rem sta $00,y    with a seek command
450 data  ca:                 :rem dex          move down to next queue
460 data  10,f3:              :rem bpl *-11     if any more to do branch
470 data  4c,fe,ff:           :rem jmp $0000    see text for this one
480 :
490 :rem this is where the program is executed when it's in the disk drive
500 :222222222222222222222222222222
510 data  78:                 :rem sei          stop background jobs
520 data  ad,a9,02:           :rem lda $02a9    get the address of the
530 data  18:                 :rem clc          current addres in the irq
540 data  69,03:              :rem adc #3       vector and add 3 to is then
550 data  8d,15,01:           :rem sta $0115    save it for us to jump to,
560 data  ad,aa,02:           :rem lda $02aa    this was done to save code
570 data  8d,16,01:           :rem sta $0116    (place at jmp $xxxx
580 data  a9,02:              :rem lda #2       place the address of our
590 data  8d,a9,02:           :rem sta $02a9    new irq routine into the
600 data  a9,01:              :rem lda #1       the 1571's irq vector
610 data  8d,aa,02:           :rem sta $02aa    so were called each irq.
620 data  58:                 :rem cli          restart the background jobs
630 data  60:                 :rem rts          and were done
        THE END    copyright by: Dennis J. Jarvis
```

IMPORTANT NOTE

BECAUSE OF NUMEROUS PRINTER TYPE FONT DIFFERENCES, THE PROGRAM LISTINGS IN THIS ISSUE ARE ONLY FOR CASUAL READING! YOU MUST USE THE DISK VERSIONS FOR ACCURATE REPRESENTATIONS.

CBUG ARCHIVE ORDER FORM
AND
NON-DISCLOSURE AGREEMENT

As a condition of receiving copies of CBUG Archive documents I agree and represent as follows without limitation or qualification:

1.) I am a member in good standing of the Chicago B128 Users Group;

2.) I intend to use the materials requested to enhance my and/or other CBUG members' usage of the B128 series of computers and accessories;

3.) I will not disclose, divulge, nor copy the contents of these documents to non-members of CBUG, except as they may be acting for me or on my behalf in the usage defined in #2 supra;

4.) I will cause a signed copy of this agreement to be forwarded to CBUG upon any further disclosure or release of these documents as permitted supra

5.) I understand that many of the documents covered hereunder are either copyrighted by or proprietary to Commodore Business Machines and/or their vendors; that the sole purpose of the provision of these informations by CBM to the CBUG membership is to enhance the usefulness of the CBM products purchased by CBUG members; and that there is no warrantee or assurance of accuracy, completeness or other representation by CBM, CBUG or any other. Any other disclosure is expressly prohibited, and I will not engage in such prohibited disclosures.

6.) I have read the limited no cost license (as printed on the reverse side hereof) extended to CBUG by CBM, and agree to be bound by those same said terms as if I were the named licensee.

Name_____ Please print or type

Address_____ Phone_____

City_____ State_____ Zip _____

Country_____

X_____ Date_____
Signature

Qty	Cat #	Archive	Description	Price	Extension
_____	12370	001	8088 Schematics w/ CCPM listings (printed)(40 pgs prox) <Add $5.00 air mail if not USA or N. America>	$25.00	_____
_____	12365	002	Software Devel. Information (microfiche)(302 images/pages prox)	9.00	_____
_____	12351	003	CBMII Programmer's Reference material (3 microfiche)(798 images/pages)	27.00	_____
_____	12327		from pg _____ to pg _____ of Archive # _____		
			from pg _____ to pg _____ of Archive # _____		
AGREEMENT MUST BE COMPLETED AND SIGNED BEFORE WE CAN SHIP ARCHIVE MATERIALS.			from pg _____ to pg _____ of Archive # _____		
			Custom produced photocopy of Archive documents.. Allow 45 days for delivery. Add $.05 per page for air mail if not USA or North America. $10.00/order minimum photocopy order	.08	_____

- -

TOTAL ORDER THIS PAGE $_____

CBM II PROGRAMMER'S REFERENCE MATERIALS

CBM II PROGRAMMING SUPPORT DOCUMENTATION

COPYRIGHT:

DISCLAIMER:

THE INFORMATION CONTAINED IN THIS DOCUMENT IS PRELIMINARY IN NATURE AND IS SUBJECT TO CHANGE WITHOUT NOTICE. THE INFORMATION CONTAINED HEREIN IS SUPPLIED IN AN "AS IS" BASIS, AND THE ENTIRE RISK AS TO ITS QUALITY AND ACCURACY IS SOLELY WITH THE BUYER. COMMODORE BUSINESS MACHINES, INC. ("CBM") MAKES NO WARRANTIES EITHER EXPRESS OR IMPLIED, WITH RESPECT TO THE ACCURACY, COMPLETNESS, QUALITY, MERCHANTABILITY OR FITNESS FOR ANY PARTICULAR PURPOSE OF THE INFORMATION HEREIN CONTAINED. IN NO EVENT SHALL CBM BE HELD LIABLE FOR DIRECT, INDIRECT, INCIDENTAL OR CONSEQUENTIAL DAMAGES RESULTING FROM ANY USE OF THIS DOCUMENT OR ANY INFORMATION CONTAINED HEREIN, EVEN IF CBM HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. SOME LAWS DO NOT ALLOW THE EXCLUSION OR LIMITATION OF IMPLIED WARRANTIES OR LIABILITIES FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES, SO THE ABOVE LIMITATION OR EXCLUSION MAY NOT APPLY.

HDS:98/mdc

commodore
business machines, inc.

Executive Offices
1200 Wilson Drive
West Chester, PA 19380
(215) 431-9100

February 10, 1986

Mr. Norman Deletzke
Chicago B128 Users Group-International
4102 North Odelle
Norridge, IL 60634

Re: Commodore B128 PM Trademark Computer Documentation

Dear Mr. Deletzke:

Effective February 10, 1986, Commodore Business Machines, Inc. ("CBM") hereby grants to the Chicago B128 Users Group-International ("Licensee") a limited, no cost, non-transferable, non-exclusive license (the "License") to use the CBM owned CBM II Programers Reference Material and the CBM II Programming Support Documentation with updated memory maps (the "Products") for the sole purpose of reproduction of the products by Licensee for distribution to its members as reference material. This License is granted on the express conditions that the Products be held by Licensee and the members of Licensee as reference materials, for their individual and personal use only, and not for any commercial purpose. The term of this License shall be for as long as Licensee continues to provide the products solely for reference purposes and its members continue to hold the products for their individual and personal use and shall automatically terminate when this use ceases, at which time the products and all copies thereof shall be returned to CBM.

In consideration of the granting of this License, Licensee hereby agrees to include with each copy made of the Products, a copy of the applicable introduction sheet attached herewith as Exhibit A which acknowledges Commodore Electronics Limited's ownership of the products and provides CBM's disclaimer as to the products.

IN WITNESS WHEREOF, the parties hereto have executed this Agreement as of the date set forth above.

CBUG, INC
(CHICAGO B128 USERS GROUP-INTERNATIONAL)

BY: _____

Title: Pres CBUG, INC

COMMODORE BUSINESS MACHINES, INC.

BY: _____

Title: Vice President

HDS:97/mdc

CBUG EAST MEETING SCHEDULE

CBUG East meets at 2:00 pm, on the fourth Sunday of each month except December, although occasional rescheduling is necessary. In case of rescheduling, all on the local mailing list will be notified. We meet at Bethlehem Lutheran Church, Wesley Avenue and Greenwood Street, in Evanston. Greenwood Street is one block north of Dempster. Wesley Avenue is a block west of Asbury (a continuation of Chicago's Western Avenue). Enter on the north side of the building. Parking on the street and behind the church. If you need more details, call Marilyn Gardner at 866-9159.

Due to the untimely passing of Ed Rhyner, the subject matter for the next two meetings is being adjusted. With the merger of the West group into the East Group, **it is important that all local members appear at the Feb. 26, 1989 meeting.** If you can not come, **please call Marilyn Gardner** to discuss how the meeting can be of greater service to you -- or you to the other members attending. Meeting Dates:

Feb. 26, 1989

March 26, 1989

CBUG WEST

CBUG West efforts are being merged into CBUG East. There are no more meetings scheduled for the Monday night West Dundee site. See you all in Evanston. (The Fox Valley Commodore Group remains alive and well for those of you who attended both).