NORTHWEST COMPUTER NEWS

Dedicated to Orphaned Computers

404 Inverrary | Deerfield, Il. 60015 | (708)808-7000

THE JCL WORKSHOP *A Further Look* by Bruce Faierson

The JCL Workshop is a development, utility and basic extension tool that has been available since the B-128 was first introduced. It was a descendant of the 8000 series Workshop. Though this package has been reviewed at least once, I feel it deserves more accolades than it has previously received. To really give this package a fair review would require a substantial part of this issue. Therefore, I will break this review up into several articles to be published in the future, if the interest is warranted.

In this first article, I will cover two areas that will be most beneficial to the whole group. These areas are the dos support and programming aids sections. All of the information presented is covered in detail in the printable manual.

$0 or $1 - allows you to display the directory much faster and in a two column format. This has to be seen to be appreciated.

$0:*=p or $1:*=p - displays a directory of only program files. If you substitute an s for the p, it will display all sequential files. You can also substitute the ? or * for wildcard searches such as $0:start* or $0:st??t. For further information on wildcard searches see your 8050 manual.

i0 or i1 - Initialize the specified drive.

f8 or f9 - This will change the default drive unit addressed by the dos support commands.

u: - this will reset the disk drive unit.

d1=0 - this is the backup command and you can reverse it to d0=1.

(continued on page 3)

Table of Contents

Hint and Tip

Question: I adjusted the speed of my drive with the Cardinal Speed and Alignment disk. Why is the speed off when I use my own brand of disk?

Answer: Since each brand of disk has a certain amount of friction between the magnetic media and covering material, the resulting speed difference can be critical. Therefore, load the Cardinal program but use a formatted, blank disk of the brand that you normally use. Now the speed test will reflect the speed measured on your disk, not theirs.

Those Who Squeak Might Get

by Tony Goceliak 14 June 89

Another issue of the News is going to press, and the B still lives. I have noticed a few new authors coming to the fore in the last year or so, and it is good to see. Congratulations to those of you who are willing to spend your time sharing your hard-won knowledge.

In this article I'd like to answer publicly a few questions which have either been asked of me by more than one individual or which seem of general interest.

Number one, many, no let's say most problems which are amenable to solution on a small computer do not require machine language in order to produce answers. Machine language can be very useful for speeding up the results, and in certain cases circumventing arbitrary limitations imposed upon us by basic, but results can usually be obtained without resorting to anything beyond normal basic.

I say this because there are a good deal more experienced Basic-language programmers in our group than m/l fanatics, so if you have a problem which you would like to solve using your B, open your yap! Write a letter detailing as much as possible what you need done, and send a copy of the Superscript file to Bruce. Of course there are no guarantees, but you might find some programmers in the group willing to help you out, or even if your project tickles their fancy, write the whole shebang for you. M/l routines are tougher to write and debug than basic, but there could even be help there as well. The point is until you ask, you'll never know. It's exactly like grumbling to your television set when the anchorman reads the latest numbskull decision from Washington D. C. instead of grumbling to your Congressman. Maybe the Congressman can't or won't help, but good golly, just maybe he Can!

There are several broad classes of problems which will probably not draw any solutions, but here are a few of my all-time favorites:

1.) Can you add a "foo-bar" command to Superbase, Superscript, Calc-result... for me?

2.) I have hard-wired a Hashafisti model xxx music synthesizer by putting the green wire on U67 pin #.....etc.....and cannot program a foo-bar command. Can you do it for me?

3.) After replacing the standard ROM chips of the B with eproms of my own devising, yep you guessed it, we want a "foo-bar".

The common thread here is either requests for modification to intentionally obfuscated (as well as copyrighted) programs, or Grossly non-

(continued on page 7)

BPI FOR THE SUPERPET *by Bruce Faierson*

Several months ago we had an interesting problem with a customer trying to run an 8032 BPI program that required a rom on a Superpet. It was our understanding that the two machines were totally compatible in the 8032 mode. We were very wrong! The Superpet uses the rom area at 9000 for the bank switched ram in the Superpet. Is this where the name Superpet 9000 originated? The only rom address available is at A000.

Since the software is still under copyright, we had to find a method that the user could implement without sending him a modified program disk. The procedure we developed required that we insert the rom at the A000 rom location, save the rom contents to disk, change the load address, modify the program disk to load the rom file to the 9000 ram area and then run the program. We came up with a solution which follows.

Instructions for BPI Install on SuperPET

1.) Make sure the SuperPET switches are on 6502 and read/write.

2.) Put BPI rom on top board of SuperPET at location U46.

3.) Put in a brand new disk into drive 0 of your 8050. Type the following observing lower and upper case.

header"bpi disk",d0,i01

Press return.

When the disk has finished formatting and assuming that there is no error, type the following to enter the monitor.

sys 54386

Press return.

Type the following at the dot prompt, there will be a number of registers displayed above it.

s"0:bpiload",08,a000,b000

Press return.

After the drive has stopped, type x at the dot prompt to exit the monitor.

Type the following program:

```
10 open 15,8,15
20 open 2,8,2,"#2"
30 print#15,"b-r"2;0;38;1
40 print#15,"m-w"chr$(3)chr$(19)chr$(1)chr$(144)
50 print#15,"u2"2;0;38;1
60 close 2
70 close 15
```

Assuming there are no error lights, put in a working copy (not the original) of your BPI disk in drive one with no write protect tab on it.

Type the following.

copy d0,"bpiload" to d1,"bpiload"

Press return

Remove the disk from drive 0 after the drive stops running and put your BPI working disk in drive 0. Type the following.

load"bpi",8

Press return and type the following.

0 open 2,8,2,"0:bpiload":close 2

Press return and type the following

dsave"@bpi"

This concept is useful for the B series or any other computer that requires a rom in a certain memory location that the user has put ram into for more versatility. This way you don't have to switch roms or cartridges to load other software. You just follow the above procedure and load the code into the specified area for the program desired.

EDITORIAL *by Bruce Faierson*

Firstly, we are sorry for the delay in publishing this first issue of the Northwest Computer News. Due to the magnitude of trying to communicate with all the old CBUGers, this issue was delayed. We had to go through the CBUG lists and cross reference it with our own to keep duplicate mailings to a minimum. We didn't complete this project til mid June.

As of this issue, we will be publishing every two months in order to achieve our goal of four issues this year. The readership is growing but we will be publishing at a loss until we get more subscribers. Please keep those articles coming and don't assume that the littlest discovery you have made is useless to the readership! Everyone has something useful to contribute.

After carefully examining our relationship with the readership in the dual role of publisher and direct marketer, we have decided to remove our advertising from the publication. Therefore, to avoid a conflict of interest our advertising will be mailed out separately. The only advertising appearing in the publication will reference items of special interest the readership should be aware of.

The library is currently in our possesion but we are trying to reach all the contributors to obtain a release for their work. If there are any questions please call us at 708-808-7000. Most disks are in the public domain but there are several royalty disks.

Lastly, please note our normal voice hours are listed in our advertising but due to other committments we aren't always available. If you are placing an order or have questions regarding merchandise please leave your name and number. If you are calling for help or to sell equipment please call back.

Northwest Computer News
Editor Bruce Faierson
Publisher Bruce Faierson
Contributors Tony Goochiak, Edwin Bowerman, Fred Peterson

Northwest Music Center,Inc.
President Bruce Faierson

Business Office
404 Inverrary Lane Deerfield Il. 60015
Advertising Office
404 Inverrary Lane Deerfield Il. 60015
Phone:
(708) 808-7000

Vice-President Kathy Faierson

SPECIAL CHARACTERS FOR SSII *by Edwin Bowerman*

A neat procedure that I have been using for including two or more special characters on the same line when using Superscript II with the 4023 printer occurred to me after reading Lowell Breunig's "Underline" note in the ninth issue of CBUG Escape.

1.) Type the text with substitute symbols wherever special characters will be needed. You can use @,#,% for example.

2.) Output to screen and note the last words above and at the line where the special characters, SC, will be needed. ESCape to the edit mode and insert a RETurn after both last words. (It's a good idea to save the file beforehand in case any text is lost because not enough spaces were inserted after a word to avoid erasing to the end of the line when the RETurn is added.)

3.) INSert a line above the SC line with the definition of the first SC.

4.) On the SC line, replace the first SC with a reverse $ (ESC-SHIFT-$).

5.) Below the SC line, INSert a line and align a reverse $ with the second SC. Repeat the insertions for the third, etc SCs. End each with RETurn ().

6.) As in step 3, precede each $ line with the definition of its' SC.

7.) Return to the SC line and replace the second, third, etc substitute symbols with spaces. Step 5 kept the alignment of the SCs.

8.) INSert *sa6,32:sa6,0; - at the beginning of the SC line.

9.) INSert *sa6,36; - at the beginning of the last line of step 5.

This procedure simplifies the alignment of the special characters and the use of the format command with secondary address of *sa6,0; results in no linefeed. (Bruenig's *sa6,1; actually feeds 1/8 of a character.)

SCs are best defined using the secondary address of 5 with 8 arguments. For example, I use a right arrow defined as *sa5,56,56,56,254,124,56,16,0 instead of the SSII *ch with 6 arguments in order to avoid garbage in the last two columns of the 8x8 character matrix. See the 4023 printer manual for the appropriate arguments for *sa5. It does not say so, but 255 can't be used and a format error type c will hold up the printer.

SC line example. **Delta, %,# Sigma, @.**

At end of procedure:

```
*sa5,2,6,10,18,34,18,10,6
*sa6,32:sa6,0;Delta, $,  Sigma, .
*sa5,56,56,56,254,124,56,16,0
      $
*sa5,130,198,170,146,130,198,0,0
*sa6,36;
```

Prints: **Delta, Δ, ➡ Sigma, ∑**

Edwin Bowerman
47 Parsonage Lane
Topsfield, Ma. 01983

(JCL continued from page 1)

c0:name=1:name - this is a much easier to remember syntax for the many quoted copy command out of Basic.

/0:program - this will load the program but not run it.

!1:program - this will load and run the program.

These commands are much easier than trying to implement them from standard Basic. After all, the purpose of using a computer is to simplify access to data and make the use of desired utility routines easier. The programming utility commands that follow, take the B computer into the world that most have known in the CPM and MS-Dos system from the beginning.

type - The type command may be the most beneficial for B-128 users. This command allows you to format and print text to the screen or a printer. You can specify a left margin, header+footer and a page length. With this command you don't have to load a word processor to view or print a text file. A text file in memory isn't disturbed by displaying another file to the screen or printing.

auto - for basic programmers who desire to use a line numbering scheme that increments automatically by the specified number.

number - you can renumber a program by specifying new start, the increment and the old start.

find - you can search text or for text within a program by a specified range of line numbers.

join - allows you to concatenate a program file with the one currently in memory, although it will not resolve line numbering problems.

merge - allows a text file specified to be read and executed as if it were being typed at the keyboard.

edit - allows you to edit with a standard line numbered system.

kill - exits the edit mode to normal Basic text mode.

put - creates a sequential file of all or a specified range that is acceptable by the CBM Assembler.

fetch - reads a sequential file into ram for editing.

gocom - converts an object file produced to the CBM Assembler format to a program file.

drive0 or drive1 - sets a drive default so you don't have to specify it.

As you can see, just the superficial commands the JCL Workshop allows are more than enough to justify the purchase. The Workshop also includes a quality assembler, a complete set of extensions to the Basic language and the ability to add your own extensions. This set of utilities is a must for the serious or casual B-128 user.

There is a catch of course. To allow for the maximum available memory in the B with the workshop loaded and to simplify access to their routines they use a bank 15 cartridge. Gary Anderson makes the best cartridge though you can modify a Calc Result cartridge for the necessary 24k of ram necessary. Contact NWM Inc. for further information on cartridge availability and pricing. We will try to organize a bulk purchase of the cartridges if we get enough interest.

BASIC-MACHINE LANGUAGE *By Tony Goceliak*

Once again a new year, and the B machine has still not lost its ability to amaze. A number of my friends have long since "upgraded" to 16, and in one case a 32 bit machine, but through the magic of our distributed smarts housed not only in the B, but our drives as well, I just won a steak dinner from a friendly bet by coming up with a program to solve a given task, beating every one of the new-tech machines.

Fortunately the rules, laid out by the rest of the group, not me, were heavily favoring our beloved B, even though the rest of them didn't know it. We had to start out from power-off, and could configure our systems in any 'legitimate' way. What I did was hook up four 8050's, each addressed differently of course, and shift/run a specialized program IN BASIC! Talk about rubbing their noses in it, for those of you who are priviledged Not to know any professional programmers, Basic is virtually uniformly panned, especially interpreted Basic, the kind that is the B's native tongue. Truth be known, there isn't really much wrong with Basic, although the speed can leave much to be desired. The "pro's" mostly shun it because then everyone else can see how easy it really is to get 'the computer' to do something, and lets face it, obfuscation is their bread and butter.

Anyway, back to my saga. The Basic program manipulated each of the other three 8050's into shift/running ampersand files (yes those were machine language) from within the drives, and by running in essence six 6502 processors more or less in parallel, I was able to beat all the others. The B really didn't do a heck of a lot, except co-ordinating things, and of course trumpeting the solution!

The foregoing anecdote doesn't really prove anything, but I sure enjoyed that steak!

Running M/l Easily in the B

The simplest way to write and run machine language programming on the B is of course in bank 15. The unadorned B has a meager 1028 bytes of contiguous freely usable ram in bank 15 to play with, and of course there is an additional 2000 bytes available if you don't mind your screen looking like a crazy quilt. Additionally there is the potential to upgrade as much as 24,000 bytes or so by purchasing one of Mr. Anderson's bank 15 cartridges, and I highly reccommend that you do so. Quite a bit of programming has been written requiring one of these cartridges to accommodate the machine code.

However, sooner or later that yawning void of bank 1 beckons. Good grief, 64,000 bytes of memory available, and 254 bytes of zero-page, the special place where a 650x can run as much as 30% faster. There are any number of ways to allow bank 1 to run machine code, but here are the objectives which I set forth for this system of activating machine code.

1. dload NOT bload the code into a stock B.

(once again, with apologies to Ms. Deal, no keytrix please) (this time however, the trouble is in the start of Basic, which Ms. Deal moves, but I don't)

Note: While the proceedure is easily re-written to accommodate an altered start of Basic, an unwary user who just powered up his B machine and dloaded the similarly configured code would get unexpected results unless the code to move +btxfer.fe48 were expanded. I feel that a user sophisticated enough to have altered the start of Basic vector in his machine is more equipped to figure out why a pseudo-Basic machine code program crashed than those of us who leave the machine alone. Consequently the code presented here is for a completely stock B machine, just as it is when the power switch is turned on, allowing someone to 'dload' and 'run' a program without even knowing whether the program was Basic or machine language!

2. run the m/l by typing 'run', instead of 'sys ????'.

3. Allow full use of kernal (and even ROM) calls, after all why re-invent the wheel.

4. Cleanly return to the Basic environment when the m/l has run.

5. Allow re-running the m/l program without any special treatment.

(after all, you can run a Basic program more than once without re-dloading)

6. Leave over 200 bytes of zero page completely at the programmer's disposal.

Good grief, not a bad little list, machine language programs which run like that become MORE convenient than those for the 8032, or c64 or whatever, and may even stimulate some more machine language programming from our group.

My Solution

The solution is not particularly elegant, neither is it wholly mine, but by gum, it works, and satisfies each and every point on the list from above. There is a very minor time penalty involved, roughly 0.1 seconds, and only the first time the m/l is run, since we are doing cartwheels to set things up.

The heart of the solution is the extremely clever transfer of execution code presented to CBUG on Liz Deals disks, from #2 onward. spoke directly to Ms. deal recently and she assures everyone that the transfer-of execution file, although not hers, is not encumbered by copyright. The transfer-of-execution code, remind me to ask Ms. Deal for a better explanation of it sometime, is real cute code. Once installed in bank 1, machine code running in that bank can simply call any of the kernal jumps precisely as if the machine code were in bank 15. In other words, once that page and a half of code has been loaded, you are running what I like to call a 'clean machine'.

The problem of course was to load the transfer of execution file. Three ways spring to mind, all will work, but all require significant penalties regarding ease of use, flexiblility, efficient disk utilization or time to load. Oh well, the shotgun blasted my list pretty well.

1. make the m/l file 64 k long, and paste the transfer of execution at the end.

(by the time your program loads, it should have been run and done!)

2. make a pseudo-Basic program which first 'bloads' the transfer of execution file and then runs the main code.

(promising, but awkward. What about using unit 9, or drive #1 to try bloading? Besides a two-step process invites trouble, as in

a. dload the program.

b. remove the disk.

c. oops!

3. run a 'menu' type program which will set things up and then dload the m/l.

(I hate menus, and once again item b. above lurks)

The Final Kludge

What I opted to do is to adopt a bit of each of the three solutions above, just enough to make it work, but not enough to bog things down.

A pseudo-Basic program goes in front of your m/l beginning at $1 0003, exactly the same address as Basic begins normally at power-up. The pseudo-Basic runs a bit of pseudo m/l, returns ever so briefly to direct mode and completely automatically zooms off to execute your m/l when you just type 'run'.

The transfer of execution code is pasted anywhere conveniently behind your m/l and is included exactly as written because it will be re-located to the appropriate spot automatically before it is ever called.

Five free bytes of bank 15 are temporarily used in order to allow the grunt-work machine code to move transfer of execution while leaving the B happily content with nary a bit of its system mussed.

Tell your assembler to begin at address $0003, and using whatever syntax is appropriate to your own assembler, include the following bytes exactly as detailed below:

***= $0003**

bytes

0a 00 00 00 89 32 00 1c 00 01 00 dc 31

3a 9e 35 31 32 3a dc 31 35 3a 80 00 6c 00 02 00

dc 31 35 3a 97 32 33 31 2c 31 36 39 3a 97 32 33

32 2c 31 3a 97 32 33 33 2c 31 33 33 3a 97 32 33

34 2c 30 3a 97 32 33 35 2c 39 36 3a 99 22 93 11

11 11 52 55 4e 13 22 3a 97 39 33 39 2c 31 33 3a

97 32 30 39 2c 31 3a 9e 32 33 31 00 ae 00 03 00

8f 20 50 47 4d 20 49 4e 43 4c 55 44 45 53 20 2b

42 54 58 46 45 52 2e 46 45 34 38 20 53 55 50 50

4c 49 45 44 20 42 59 20 4c 49 5a 20 44 45 41 4c

20 43 42 55 47 20 44 49 53 4b 20 23 32 00 00 00

end bytes

What this is is a Basic program as listed below:

0 goto2

1 bank1:sys512:bank15:end

2 bank15:poke 231,169:poke 232,1:poke 233,133:poke 234,0:poke 235,96:print"(clear) (down) (down) (down) run (home)":poke 939,13:poke 209,1:sys231

3 rem pgm includes +btxfer.fe48 supplied by liz deal cbug disk #2

Line #3 is of course not necessary to this system, but acknowledgement is nice.

So far, page zero is gummed up all the way to $00b0, and the creature still can't fly. Patience.

Next tell your assembler to go to $00e7 and include the following source for assembly:

***= $00e7**

lda #0f

sta $00

ldx #00

lda $0700,x; or whatever address lies safely behind your code

sta $fe00,x

lda $0800,x; do I have to mention this is one page further

sta $ff00,x

inx

bne #f1

dec $08; or why you can't move start-of-Basic!

bne #e7; go back to the beginning, and exit to bank15

Finally, tell your assembler to goto $0200 and begin to assemble your own code. Why 0200? I'll just bet that you don't want your code trashed by the action on the stack. From this address all the way to $fe47 is ALL YOURS! Miles and miles of empty bytes. And that isn't the end of the good news!

Zero page is also yours with only two exceptions. $1 0000 to 001d, which is the remnant of the Basic program needed to allow running the code more than once, and $1 00ac and 00ad, which are used by the transfer of execution code. By modifying the file +btxfer.fe48 slightly, the locations could easily be changed to $001e and $001f, which will leave 223 contiguous free bytes on page zero! The code above at $1 00e7 to 00ff will never be executed again, and neither will the five bytes in bank 15 represented by the pokes.

Good programming practice should dictate pre-setting any storage space to a definite value, (mostly 00), instead of assuming that some value should have been there, so feel free to wipe page zero clean from $001c to 00ff (wiping location $001c will result in a clean result if a user attempts to 'list' the program after you have stolen some or all of the space assigned to the 'rem' line.)

The program is executed by typing 'run'. Line zero steers the program away from the main action, since transfer-of-execution isn't set up properly yet. Line 2 makes a very simple brute-force transfer-of execution to the code at bank1 $00e7. If you never intend to use a kernal call, this is really all you need, it works. However, opening files, getin from file or keyboard, printing to screen with all the bells and whistles and all the rest sure are handy.

The code at $00e7 merely relocates the transfer-of-execution code to the correct addresses and then shuffles back to

bank15, where the rts returns us to Basic. Since line 2 is finished, and line 3 says 'rem', and that is the last line, we ever so briefly drop back to direct mode, where, thanks to the pokes to 939 and 209, the B system finds one character in the keyboard buffer, (even though your fingers didn't walk), and the cursor right on top of the word run.

The character is carriage-return, and when Basic makes sense of that, the program is run for a second time.

This time, however, the good old instruction 'dec $08' has mutated line #0, causing it to read as follows:

0 goto1

The point!

Transfer-of-execution allows the B to properly handle everything, including interrupts, ti$, IEEE, keyboard, you name it just like your code was in bank 15, except OH MY! those wide open vistas of space!

T-o-e is so smart that your code purring along in bank1 can gracefully exit to Basic with a simple rts, precisely as if it were purring in the system bank!

Now, whether you erase line #2 and 3 or not, the Basic program reads and performs the following; be it the first time or the seventieth time you 'run' the program:

0 goto1

1 bank1:sys512:bank15:end

Location $08 is never dec'ced again, the pokes are never done again, and your m/l beginning at $0200 (decimal 512) runs like a dream.

I have included a demonstration program for you of more use than the free-steak program, using exactly this system. The machine language program is handled exactly as if it were Basic, except of course that Basic has no knowledge of what went on between the sys512 and the bank15 commands, which includes the number input by you or the numbers put out by the program.

The program asks for a positive integer to be typed in whose length is between 1 and 64 digits, and when you type return to signal 'thats all folks', the computer will respond with a list of the prime factors of the number.

The length of the program did not dictate use of bank1, but my desire to use LOTS of zero page did.

One interesting point - the arithmetic is done in bcd mode, just because I never did that before. I suspect that many 650x m/l programmers aren't really sure that their own assembler correctly handles the instruction to set the decimal mode, because almost no one ever does.

If you enter non-numeric characters the program will discard all previous input and begin with a fresh slate. The prime factors of 'tony' are hard to derive. As usual, there is an exception, which is the comma. If entered, a comma will be echoed to screen but ignored by the program. While testing, one of my benchmarks was a particular seventeen digit number, and I became convinced of the desireability of commas to allow me to look at what I typed without going crazy.

Entering zero as the input, or trying to exceed 64 characters will result in short messages and a return to Basic direct mode, ready to be 'run' again.

Have fun with the demo, there are even practical uses for factoring programs, and by all means use the 'run' system to make your own m/l experiences much less painful on the B.

Tony Gocekiak
RFD#2 Box 433 Wesson Road
Lancaster, N.H. 03584

COPYRIGHTS *by Fred Peterson*

Please carefully read both this notice and all other copyright & freeware notices which may be on this disk.

Some programs are copyrighted by their creators/contributors and permission must be obtained from those authors prior to certain types of usage as may be specified. Other programs are submitted on a FREEWARE basis -- i.e. if you use them, you are morally obliged to make at least a token payment to the author/contributor. Many CBUG disks bear the notice "This is a royalty paid copy" -- as such the Freeware consideration has been waived for that copy only by the inclusion of a nominal royalty remitted to the author/contributor. Without acts of FREEWARE gratuity, many contributors may not continue to go to the extreme effort exhibited herein.

NWM INC., and I hope I speak for all members, has no interest in duplicating materials not in the public domain. Anyone caught submitting non-public domain materials will be summarily aborted from membership.

I'm no expert in copyright matters, but as I understand it, many persons routinely put copyright notices on freeware and public domain disks to discourage profiteering. Thus the mere existence of a copyright notice is not conclusive. As a small group with limited means, I hope everyone uses their very best judgement in trying to make these determinations.

Generally, programming printed in magazines may only be acquired by subscribers to those magazines. Sometimes affidavits of subscription are used -- but it is haphazard at best. On the other hand, magazines and programming published by Commodore are intended for their customers, so for our purposes Commodore written articles and utilities are unrestricted. Obviously programs purchased by Commodore from 3rd party vendors (such as Precision Software and Info Designs) and bearing dual copyright notices on their commercialized programming are likely to be restricted. Similarly certain obviously commercialized publications of Commodore itself are not intended for public domain usage. Similarly, the entire Commodore education library is for copying (available thru TPUG and most local clubs). Much of the Commodore public domain library is in pure basic -- and runable on any Commodore machine.

If you discover something on any of our disks which you believe to be violative of copyright laws, kindly write me yesterday. Identify the disk, program and why you believe it is illegal -- i.e. you know the same program is for sale at the local computer store under another name (state name and publisher, address if possible).

Unfortunately, time does not allow us to correspond with members as liberally as we gab on the phone. BUT, your letters and notes are read and usually passed on to our

Question and Answer man. You just might hear from him and more likely your comment may seed a paragraph or two in our publications.

Thank you for your cooperation and assistance in advance.

Fair Use Doctrine.

United States Copyright Laws provide for what is commonly known as fair use copying (fair use doctrine). Simply put, a purchaser of copyrighted materials is generally free to make copies for his own use without violating Federal law. However, to make copies of copyrighted materials contrary to the copyright owners' intention for the benefit of third parties or possibly even multiple places of use (such as branch offices/stores/departments) may not be permissible. Certain programming is sold under contracts far more restrictive than the copyright laws, and such contractual agreements are enforceable under applicable civil laws of contract.

Certain states, notably Louisiana and recently (but not final) Illinois have enacted laws codifying "shrink wrap contracts". This is a device wherein the copyright owner is permitted to bind the purchaser to a contract contained within the sealed package even though the purchaser has not had an opportunity to read this agreement. Obviously such a "unique" legal concept is being hotly contested. Not only is a contract being imposed, but often the software is stated as being licensed, not sold! -- i.e. the purchaser owns no product, only a permission to use. Moreover some states, Illinois notably, is attempting to permit the software producers to disclaim all warranties including even loading or operating at all! The issue of State's rights to impose laws relating to copyright (particularly as these laws conflict with Federal law in many areas) is new and uncharted territory. Take care to follow your local press on this subject.

CBUG has incurred considerable expense in preparing its' library. None of our disks are copy protected and, other than as stated supra, no copying restrictions are imposed regarding use of these materials by NWM readers in good standing. It is, however, requested that if additional copies are distributed by a member to others (preferably members), that a copy fee of $5.00 per disk so duplicated be remitted to us. This also should be regarded as a FREEWARE obligation. Please remit to Bruce Faierson as below and indicate that the sum is for copy fee(s) regarding CBUG library materials.

Materials have been submitted to CBUG for the use of its members, and no permission is granted by CBUG, myself or any other party to distribute copies beyond the membership of CBUG.

CBUG library disks are provided in several forms, this one is of the official release or reviewed release (CBUG # or RR #) variety. If the disk is labeled with a piggyback label (easily detachable and having "PEEL HERE" printed on the right hand end thereof), the disk is subject to future update. You may acquire such updated disks when announced at 1/2 price (including 50% off on royalties if any). The detachable label serves as proof of ownership and must be returned with such updated disk orders. The updated disks will carry special stock numbers and can only be ordered using those numbers as announced. We suggest you use Scotch tape to better affix the label to the present disk. **(Editors note: We may or may not use piggyback labels as described above.)**

Bruce Faierson, President
Northwest Music Center, Inc. (NWM)
404 Inverrary Lane,
Deerfield, IL 60015
Telephone: 708/808-7000 12pm to 7pm CST

P.S. All of our disks have been duplicated in a manner that should assure they are as the authors intended. Should your equipment be slightly out of adjustment in an opposite direction (i.e. one fast, one slow), you may have difficulty in backing up the disk. Please try to back up both d1=0 and d0=1. If that fails, send us a note within 30 days and we will replace the disk promptly without charge.

LIBRARY DISKS REVIEWED

Goceliak 1990: Does this guy ever stop amazing us with his new discoveries. If you want to learn what the B machine is capable of, Tony Goceliaks many disks are the way. His understanding of our system is astonishing. One has to wonder whether he sleeps at night or just has his B-128 do it for him. This disk is a must for Goceliak collectors. This disk includes the following.
Price $14.00 Royalty paid. Shipping $2 any quantity

1.) **One disk - two directories**
2.) **Alphabatize your directory**
3.) **Unscratch programs leaving memory intact.**
4.) **File locking program - prevents scratching.**
5.) **Obliterate a file so no one can resurrect it.**
6.) **80xx spinning program for cleaning drives.**
7.) **Prime Factoring program and much, much more.**

Goldcoast Gambit: Fred Peterson one of the many fine contributors to the CBUG library has done it again. After you shift/run your eyes are treated to an interesting video creation for the B. Fred includes not only a complete description of all the programs on the disk but a menuing system to access them. His disk includes his SSII Financial Spreadsheet. Fred states this a satisfactory though simple method of keeping a record of all receipts and disbursments for a small business. This disk has many different types of programs from utilities to games to science and mathmatical challenges. Fred really spent some time putting this one together. This disk includes the following.
Price $9.00 Freeware. Shipping $2 any quanitiy.

1.) **SSII Financial Spreadsheet.**
2.) **Normopoly - a properties buying game runs on 8432e.**
3.) **Backgammon - 8432 emulator.**
4.) **Chessmate - 8432 emulator.**
5.) **Games - Utilities - Disassembler etc.**

(Squeak continued from page 1)

standard hardware. Just because you have installed an a-b switch on your rs-232 port and sometimes operate a printer and sometimes a modem does not put you in this category, but remember the further your system strays from the "standard" Protecto package, the fewer potential gurus there will be to aid you. Asking for help in any case won't hurt, the worst that can happen is no one will answer.

Remember too, that there can be more than one way to skin a cat, so that "foo-bar" that does not seem likely to be added to Superscript just might work as a basic program which operates on a Superscript file. If you ask one person, you may get an honest answer to the effect of 'no way!', but if you ask the whole group, you are automatically asking several THOUSAND individuals. So don't just read the Escape, write!

Mr. Anthony J. Goceliak
32 Cottage Street
Jersey City N. J. 07306

YELL FOR HELP!

The people listed below have graciously offered their expertise and time to help fellow CBM computer users. I applaud their generosity and want to thank them for their willingness to share their knowledge.

1.) Please call them only during the hours listed and don't call collect.

2.) Please don't abuse this privilege and only call them when their help is genuinely needed.

NAME	AREA OF EXPERTISE	TIMES AVAILABLE	PHONE OR ADDRESS
Louis Black	Calc Result - Paperclip	7:30-11:30 PM EST	416-728-3244
J. Boyle Electronics	Forth - 65xx ASM	Sat. 7-10 EST	904-539-0506
Alan Bouvier	SSII,ASM,Basic	M-F 6-10 S-S 12-8 CST	504-649-5772
Edwin Bowerman	Basic - SSII	Mail Only w/S.A.S.E.	47 Parsonage Ln Topsfield,Ma. 01983
Art Chick	Basic,SSII,,SBII	T-Th 7-10PM PST	916-674-7006
Vern Kempfer	General Info	Evening & Sat-Sun CST	608-244-3353
Bob Loeffler	CABS GL-AR-AP	T-W CST	414-294-6412
Fred Lovejoy	SSII	M&W 7:00-8:30 PM	602-946-0202
Dan Mikesell	Basic	7-10 PM EST	616-842-4205
Carter Pawlus	Calc Result	9-9 PM CST	414-457-6100
Fred Peterson	SSII,SBI,SBII	M-F 7-5 PM PST	805-492-0066 **Will call collect if you leave a message**
Robert Walther	SSII,SSIII	Mail Only w/S.A.S.E. **Phone in future.**	20209 150th Dr. Sun City West, Az. 85375. **Just moved here so I'm not sure.**
John Wright	CP/M-MS Dos for B-128	M,T,TH,F 7-10PM CST	402-339-5728

Anderson Communications Engineering

2560 Glass Rd. NE.
Cedar Rapids, Iowa 52402
USA

This is a plug for Gary Anderson. This man has single handedly developed most of the publicized add on boards for the B-128. Among his many accomplishents are the 1 meg add on board, 24k memory cartridge, the alternate operating system board and worked on developing a prototype v-2 or suped up 8088 board. He will continue manufacturing some of these boards until October 1st, 1991. Please support this man as he is the only source for any expansion products.
The following products will be available.
24k ram cartridges @$75.00
B-1024 Expansion boards @$329.00
Gary may be willing to manufacture the 24k cartridges at a lower price if we buy in volume. Call NWM Inc. for details. **707-808-7000.**

Hot Info: *March 1991*

Jesse Knight of Knight's Computers has indicated an interest in doing an upgrade of his immortal Knight's Copy Utility. The upgrade would be unprotected and would include a faster copy utility. The price would be lower to registered owners than first time buyers. Please notify NWM,Inc. if you are interested so we can let Jesse know if there is adequate demand for it!

Fred King of King Communications has announced that he will still perform1 meg upgrades on B-128 & B-256 computers. Contact him directly at **715-341-1149.**

Northwest Music Center Inc. still does repairs on B-128 computers, 8050 drives, 4023p and 8023p printers. Call for details. **708-808-7000**