CHICAGO B128 USER'S GROUP -- INTERNATIONAL
CBUG, Inc.
4102 N. Odell -- Norridge, Il. 60634 USA

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

FORM 3547
REQUESTED

DATED MATERIAL
DO NOT DELAY!!

Return the ENTIRE Page -- including the mailing label. **Make** corrections as necessary
Please help us out. Print very legibly or type

---

* THE CBUG ESCAPE *

Welcome to THE COLOGNE B128 SELF HELP USERS's GROUP, OUR NEW AFFILIATE!

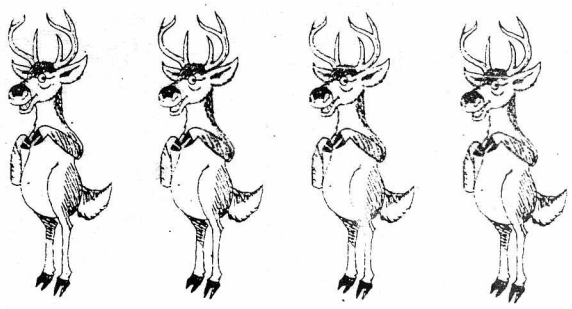
Together we are nearly 3000 strong!

* NEW * NEW * NEW *

* B128 SOURCE CODE *

* THE NEW TESTAMENT *

* MULTITASKING 8432! *



Cover Price: Four Bucks (U.S.)

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	_____
!!See pages A11 & A12 for disks & data cases!! MERCHANDISE SUB TOTAL				_____
Illinois residents add 7% Sales tax to above items only				_____
CBUG #56 Harrison's Assembler, revised, #5.5 v8 (If upgrading your prior copy, attach label and extend at $5.)	_____	11959	35.00	_____
CBUG #60 LIZ's Summer 87 Upgrade *	_____	11659	16.00	_____
CBUG #61 Super Teacher, a Superbase application *	_____	11682	9.00	_____
CBUG #62 Super Church, a Superbase application*	_____	11697	9.00	_____
CBUG #63 The King James New Testament, set of two disks . . .	_____	11709	30.00	_____
CBUG #64 Sermons #2	_____	11714	9.00	_____
CBUG #65 Sermons #3	_____	11728	9.00	_____
CBUG #71 Precision Church, a Superbase application	_____	11593	25.00	_____
SET the above 5 releases, 6 disks total	_____	11547	57.00	_____
CBUG #66 The NEW 8432 Emulator v.g & more	_____	11733	11.00	_____
Pre Release #8 CP/M 86 Info & Pgms #1	_____	11611	9.00	_____
Pre Release #9 CP/M 86 Info & Pgms #2	_____	11625	9.00	_____
Pre Release #10 CP/M 86 Info & Pgms #3	_____	11630	9.00	_____
***CBUG #67 B128 Kernal/editor Source Code	_____	11747	9.00	_____
***CBUG #68 Basic Source Code B128 & Others	_____	11752	9.00	_____
CBUG #69 Basic Source Code under study	_____	11766	9.00	_____
PROGRAMMER'S CHALLENGE SET of the above 6 releases, 6 disks	_____	11588	35.00	_____
CBUG #70 Summer 1987 Print Files	_____	11606	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	_____
9060/9090 Service Manual - photocopy (allow extra week) . . .	_____	12295	25.00	_____
SFD-1001 Schematics by photocopy of stat reductions, 11" x 17"	_____	12308	5.00	_____
Recopy fee remittance	_____	12401	5.00	_____
BALANCE FORWARD FROM PAGE 2 RIGHT HAND SIDE OF FORM -------->				_____
BALANCE FWD FROM DISK & DATA CASE ORDER FORMS, page A12 --------> <North America addresses only!!!>				_____
MERCHANDISE SHIPPING AND HANDLING CHARGE --- ALWAYS INCLUDED, NO EXCEPTIONS				$ 2.00
SUBTOTAL			$	_____
Free Will Contribution to CBUG.		12416		_____
	_____			=========
BY CHECK ___ BY MONEY ORDER ___ TOTAL REMITTED			$	_____

NOTE: We endeavor to ship against money order within 3 business days of receipt. Others delayed 14 to 18 days
* This disk is subject to the 50% discount upgrade policy -- see detail listing in the Library section.
*** You must submit a signed copy of the agreement on the back of this issue to obtain these disks.

SEI LAST MINUTE UPDATE

While the main body of this issue is being collated, this last minute insert is being printed. We've held this till the last possible moment hoping we would have better news to report -- although it's not that bad either.

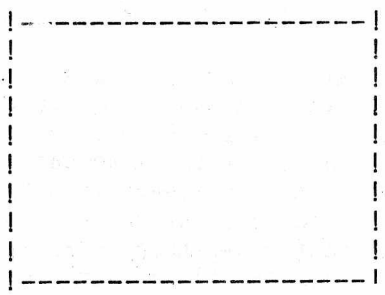
As of 10pm CST Oct. 22, 1987 CBUG has received reports of five upgraded 8050's being delivered. Florida, North Carolina, Michigan, Indiana, and California. With one exception these machines were delivered with some relationship to the order received. Additionally one unit was attempted to be delivered but was refused for reason of a substantial COD amount resulting from additional work done without prior approval.

Yesterday CBUG made a 85% prox telephone survey of all members subscribing to the SEI offer to ascertain the above data. SEI/Mitchell Cotter provided us with a list of machines shipped one or more weeks previously totaling in excess of 16 units. The delivered units were all from that list; and no accounting of the others has been made available. There seems to be no purpose in repeating delivery promises.

In each instance the delivered machines were reported working to specification and each customer was absolutely pleased with the performance and workmanship. There has been only one reported failure -- which is <u>not</u> reasonably attributable to the repair work, merely a coincidence.

While this is encouraging news, complacency is NOT the appropriate proceedure.

If a revised delivery number is received up to the moment ink hits this paper, a correction number will be "gouged" into the printing plate below.



CBUG LOCAL NEWS - CHICAGO MEETINGS

Sept. 19, 1987

Chicago Area B128/256 Users' Group -
International (CBUG, Inc.)

CBUG EAST MEETING SCHEDULE

September 27 - CABS Software -
Forum and Demonstration

Dan Schoger will demonstrate some of the uses to which he puts the business programs such as general ledger, accounts receivable, accounts payable, payroll, inventory, and order entry. A number of CBUG members have purchased this accounting suite of programs and have yet to use them to their potential. Dan uses these programs daily and knows them well. Bring your questions and be prepared to learn a lot about what your software can do.

NOTE THAT CABS SOFTWARE IS STILL AVAILABLE - all programs except inventory are available from Northwest Music (312) 520-2540 for $9.95 each or $24.95 for five. If you own the CABS order entry program, you can get the CABS inventory program from CBUG (312) 456-8720.

CBUG East meets at 2:00 pm, on the fourth Sunday of each month except December, though occasional rescheduling is necessary. In case of rescheduling, all on the local mailing list will be notified by one of these mailings. There are no dues for these meetings. No one even checks to see if you are a member of CBUG! There is a sign up sheet, to make sure that you will get these local mailings. 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 (312) 866-9159.

This is a meeting you can't afford to miss: Accounting suites are generally very similar one to the next, much like data base programs. Learn one and all others are far more easily understood and mastered. It is believed that the CABS suite and computerized accounting on the B128 are amongst the least understood therefore least used of any of the commercial software available for

Continued at rear insert

THE CBUG ESCAPE

SEVENTH ISSUE

Summer 1987

* * *

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.

CHECK YOUR ADDRESS LABEL FOR EXPIRATION DATE. The expiration date is to the right of the postal code in the form of YYMM, eg. 8712 indicates an expiration at the end of December 1987. CBUG will be unable to mail publications without renewal. PLEASE KEEP YOUR MAILING ADDRESS CURRENT. Renewal invoices will be mailed late December.

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

Cover price this issue: $4.00, 1987 subscription rate: $14.00 (bulk rate, U.S. & possessions ONLY); $20.00 (first class, U.S. & possessions, Canada & Mexico); $21.00 (surface <boat> first class, all others); $35.00 (small packet rate air mail, any country).

* * *

DON'T FORGET DELPHI!

Take a look at the several previous articles about the Delphi Network and CBUG's own private area there. While traffic has been light during the summer months, come ahead this fall, join, ask, learn. Lots of our best and brightest there to serve you. All you need is a modem and a terminal program. Call 800 544 4005 and ask for the local access phone number for your area. At the prompt use the username "joincbug", and the temporary password "newmem".

* * *

CBUG Chicago Area Meetings

East: 4th Sunday each month save holliday conflicts. 2pm, Bethlehem Lutheran Church, 1334 Wesley, Evanston, Il. For current information contact: Marilyn Gardner 312 866 9159 early evening hours
West: 2nd Monday each month. 7:30 pm, 1st Congregational Church, 5th & Main, West Dundee, Il. For current information contact: Warren Swan 312 665 1514 early evening hours. For weather cancellations: Herb Gross 312 695 1316.

* * *

TABLE OF CONTENTS

CBUG SCRATCH PAD

By: Norman Deltzke

Oct. 3, 1987

Thanks for bearing with us -- waiting for the Summer issue which now ought be called "injun summer." I still have hopes of putting out one more issue before the end of 1987 -- contingent on receiving atleast a few articles and disks to offer. Thanks to the many of you who did get things in by late August for this issue. As is our policy, we are publishing everything on hand. If by chance we missed someone's article or disk, let us know in case it got lost.

To our amazement, CBUG is continuing to grow. Yes, we lost quite a few members at renewal time last January, however, there is a constant parade of new member applications consequent to inquiries of Protecto, Commodore and many other groups and companies world wide who know of CBUG's remarkable performance. The most unexpected of membership applications come from people who received the Fall 1985 mailing to all of Protecto's B128 customers. Two years later -- I guess I must yield the title of procrastinator.

We are now exchanging limited materials with the Cologne 600/700B Self Help Group. Their roster is approaching 1000 members. From conversations and input received, they appear to be extremely technically oriented. The wizards of our two groups together will certainly exceed the productivity of the two independantly. We expect early results which will be of great benefit to lay members at large. In short order most library and documentary material will be mutually available. CBUG will need several more translators -- German to English, for all types of work, technical thru more conventional writtings. Rev. Mark Schwarzbauer is administering the German/CBUG exchange project. If you can assist, please contact Mark.

As a result of Mark's heavy load with the interchange with the Cologne group and several other time consuming responsibilities, the job of Librarian is being taken over by Marilyn Gardner. In the next months Marilyn will be forming up the review and repair functions. Concurrently, a much simpler project has been designed by Warren Swan to "rate" the quality of software appearing in the CBUG library.

In the future we will become a bit more efficient. The storage, shipping, --all but the computer/office operations have been removed from my home to a nearby office and storage area.

In this issue we are releasing a large body of source code for the B128 and several other related products which CBM kindly gave us. This material is for CBUG use only. Further disemination is illegal. Take a gander at the new 8432 emulator: Multitasking has come to the B128! And there is plenty more in this library section

****

*** ***

*** LEGAL NOTICE ***

*** As consequence of our cross publishing understanding, ***

*** all authors and contributors of any kind are hereby ***

*** noticed that unless they conspicuously instruct to ***

*** the contrary in writing, all materials published by ***

*** CBUG will be made available to The Cologne 600/700B ***

*** Self Help Group as if it was a division of CBUG, Inc ***

*** ***

****

PUBLISHING NOTICE

The CBUG ESCAPE is published by CBUG, Inc., c/o Norman Deltzke, 4102 N. Odell, Norridge, Il. 60634. 312 456 8720 7pm to 10pm CST Wed thru Sat. The opinions, averments, statements and representations of authors and also those of advertisers are the sole responsibility of those authors and advertisers. CBUG assumes no obligation to apply editorial prerogative. Evaluation of materials appearing in the CBUG ESCAPE is left solely to the reading member.

THE SEI PROBLEM

By way of background, in the spring of 1986 and again in the fall of 1986 SEI SERVICE CENTER provided an article and placed two advertisments for an important improvement and also repair service for 8050 drives. Unfortunately it did not come to gether. Members sent in drives beginning in about April 1986. By June of 1987 the list of drives as provided by SEI to CBUG and to The U.S. Postal Inspection Office totaled 32. To date, one drive was returned without the upgrade but otherwise repaired with fair charges for the service performed - and reported performed quite professionally. A second drive was delivered COD for additional charges incured, and a third drive was shipped but refused due to the COD. A fourth was picked up in person, done "promptly" as quoted -- very happy client.

Since June 1987 Mr. Mitchell Cotter who represents himself as in charge of the operation has been in aproximately weekly contact with CBUG giving continuing reports of machines shipped or being shipped. Excepting as above, none have been confirmed. After the fact admissions and appologies were tendered and new promises made. As this is being prepared, yet another set of promises have been made. We will insert a last moment delivery status report in this issue under the closure staple.

If you are a customer herein, it is strongly advised that you protect your interest by writing the vendor via Return Receipted Letter demanding return of your drive. Copy said letter along with a formal complaint to atleast:

Mr. Wayne Laicanno, U.S. Postal Inspector, U.S. Post Office, Charlotte, N.C. 28228-3000;

Katheryn Wilson, esq., North Carolina Attorney General's Office, P.O. Box 629, Raleigh, N.C. 27602.

Write frequently to keep the fire lit. CBUG has the new phone number for Mr. Cotter -- call if you need it.

FROM THE BEGINNING

By: Bruce Faierson

This is a new section initiated by the constructive criticism of a CBUG member. He felt that the articles presented, in general, were way too difficult for the average CBUG member. After reviewing the situation, it certainly seems that this criticism has merit. Many of the groups members have neither the time nor the patience to go through the reference and technical manuals. So keep those suggestions coming and lets see more input and articles from everyone.

A basic course, in fundamental computer use and setup tips, has not been attempted before this time. It should be stated that, Warren Swan did an incredible job with his course on Basic. His course should be read by anyone who

has an interest in this column. This column will be an attempt to supplement the fine work that Warren has done.

It was late in 1984 when I learned of Protecto, a booming new mail order business. Within a few months of using the C-64, I realized it was way too slow to use for business applications. The printer was the pits. It was so slow, you could go out for lunch and 18 holes of golf and it would still be printing a small inventory when you got back. Assuming, of course that the tractor had not jammed up with paper. So I was in the market for a printer, if not a complete replacement for the 64.

I went out to Protecto, which at that time was just opening up a customer showroom. It was at this time that I saw the deal of 1984. This was the B-128 system! If you compared it to other systems such as IBM or Tandy you were getting almost the same functionality for 25% of the price. Needless to say I bought the system. I'd increased our efficiency by at least four times over the C-64.

Now came the time to hook up the whole system. With the C-64 it was easy. All you had to do was hook up your serial cables and soon we were computing. With the CBM B-128 we had a whole new language of cables and connectors. We have a p-i, i-i, rs-232c, and a monitor cable besides all of the internal connector ports.

The B-128 has an IEEE-488 interface that uses different cabling and connectors than most computer systems. The edge of the mother board was used for one end of the cable and became known as the pet end or "p" for short. The other end was the ieee connector known as "i". Thus we have developed the shorthand name for the computer to drive cable, p-i. It follows that the i-i cable, from drive to printer or drive to drive, has two ieee connectors

Now we had to connect them all together. This may sound easier to you, than it may have been for some others. To attach the Pet end into the computer seemed to be just a matter of inserting the polarized connector into the computer. Technically, the Commodore label should be facing up. Unfortunately, there was a batch of cables that were made upside down and a batch that had poor soldering.

If you ever had a cable that did not work of the Commodore variety p-i, you might try removing the two little separators and reversing the cable so the Commodore label is down. The typical effect of an upside down p-i cable is the computer hanging up while trying to load a program. If the cable is removed at this time, the computer will come back with an error message.

The i-i cable goes on one way and should be screwed tightly to the back of the p-i cable. The "i" end of the p-i cable is attached directly to the disk drive unit. You will also notice that the Commodore cables are shielded and have grounding lugs. These cables should be connected to the chassis grounds of the computer and drive.

This leads to an important topic. A ground is used for several reasons. Firstly, if some component shorted internally to the chassis, it would take the chassis to ground or 0 potential. This will occur if the chassis is connected to a good earth ground. Just because you have three prong outlets does not mean that they wired correctly. The round plug should be connected to a metal pipe, with at least ten feet of surface in contact with the ground.

Some contractors may cut costs by eliminating this important safety feature. The user should make every reasonable effort to insure that earth ground is connected. If your outlets have only two prongs, the hot wire and the return, you should get a three prong adapter with a pigtail lead. This should be connected to the screw that holds on the outlet cover. It should be ascertained whether this is connected to the conduit which should be connected to an earth ground. If not, you will have no effective earth ground.

The shield should only be connected on one end. This explains why there is no lug on the generic CBM printers for the IEEE shield. All of the other devices should be connected to their ground lugs. The RS-232 cable follows the same principle. The shield or pin one should only be connected on one end. Remember that this is a shield, not a ground reference for the signal leads. Pin seven is your signal ground.

When the RS-232 cable is run a distance to another area of a building other problems may occur. Assume the wiring in a building was reversed in one circuit versus another. If your equipment was connected to both circuits for some reason, you could have a difference of potential on your ground leads. This voltage difference between the shield and ground could cause erratic signal transmission. Since the RS-232 has two grounds, under the right circumstances this can cause havoc with transmission and receiving of data. So care must be taken. Remember that the shield is designed to protect the conductors from stray electrical interference and not to be a ground reference.

The B series computer can have up to five dual drives daisy chained together. The unit numbers normally used would range from eight to twelve. The Commodore IEEE bus allows use of the numbers through thirty one for unit number. The lowest default number drive is eight. The author has programmed drives from six to thirty. However, using the jumpers available in the 8050 or SFD the maximum hardware change is from eight to fifteen. It is recommended that you use only eight through fifteen for drive unit numbers.

If you have more than one drive, there are several methods for changing unit numbers. Instructions are given for hardware and software changes in your disk drive manual. There is also a method of using an spst switch in series with a unit change line. We will cover this in a future issue.

The unit hardware change is accomplished by the lifting of pin 22, 23 or 24 on UE1 which is located on the digital board. It is recommended that you lift the pins rather than cutting the traces to the pin. This facilitates easy changing of unit numbers. Software changes are also possible but can be a painful method if you have to reset your computer. Every time you reset your computer, your drive is reset to its default number. Check the Liz Deal Utility disk for a soft unit change program.

It is recommended that the first device connected to the computer be the lowest and hopefully default drive. Drive unit eight is always the default drive. It is the only drive that can auto load programs and carry out basic DOS functions without specifying a unit number. DOS means disk operating system. The DOS coordinates all activities for the drive that are commanded by the computer CPU, or central processing unit. More on DOS in future issues.

The final item to connect on the IEEE bus is the printer. Always place the printer as the last item in the chain. The printer can be unit four or five. It is of course possible for you to have two printers on line with different unit numbers. You could, for example, have a daisy wheel and a dot matrix hooked up for different projects. One final note is to only place one cable piggybacked ontop of each other cable.

The only item remaining before we can start up our system is to hook up our monitor. This is a matter of connecting a five pin din connector to the computer and the appropriate video cable to the monitor. It is also interesting to note, the B-128 lo profile has a TTL output if the appropriate changes are made to the mother board. It also appears that by substituting the hi profile character generator that we can have the same screen display as the higher resolution hi-profile. More on these developments later as we investigate more thoroughly.

At this time we are getting excited. It would appear that we should start everything up and run our first program. This is what most beginners would do. The first item on your checklist should always be to backup your disks. I did say backup, not copy. For some reason, the word backup and copy have become confused with each other.

For the record, backup is defined as the operation of making an exact duplicate of your disk. Copy means to copy the files usually found in the directory. The difference here is slight but deadly. If the original disk has data written in tracks and sectors that are not in the directory, then your copy will not have that data on the disk. This may be information that the program needs to operate or data that the program needs to access for a database for instance. By the way, you can not copy a random file though you can back it up. So backup all your disks that can be backed up.

For those that can not be backed up, get a program that will make an exact copy. There is only one program that can do this on the B-128. This is the Jesse Knight copy utility. Rumor has it that Jesse will introduce a new and better version. Send or request information from CBUG on this fine utility. If you do not backup your program disks or data disks, then you only have yourself to blame.

Your data disks should be backed up in a manner that a damaged disk can not destroy all of your good disks. The only way to do this is to rotate several disks to be backed up. Using this method, a bad disk will only damage some of your disks if it is backed up. This means you will always have a good disk. A good disk that is not as current, is better than your most updated disk that is unreadable.

Once you have made sufficient backups, it would be a good idea to cross check the directories. By displaying the directories you can make sure the disks were backed up correctly. The original command structure was to type, without the brackets, <load "$",8>, wait and type <list>. Any basic command or statement can be reduced to two or three letter abbreviations. Therefore this command can be simplified. The same results will occur if you type <lO "$",8> wait and type <lI>. A complete list of Basic abbreviations is found between pages 111 and 113 of the silver B-128 users guide.

The longer you work with computers the more shortcuts you find. I prefer to type the function key <F8>. This command automatically loads the directory and displays it screen by screen. The previous command would display the whole directory without stopping, unless you abort the routine with the stop key. The same results can be obtained by typing <directory> or <diR> for short.

Another check to make sure that your disk was backed up correctly is to type <?ds$> without the brackets. This command will check the disk status of your drive right after you have finished backing up a disk. The properresponse should be <00,ok,00,00,0>. If you do not receive this response then you know there are system problems. It would probably be best to seek professional help if this occurs.

The next step in using your system is to header some disks. Most programs require the se of pre-formatted disks to operate. The header command formats your disk with tracks and sectors. At this time it also puts a directory and bam on the disk. The bam is the block availability map. This area on the disk stores the allocated and free track and sectors on the disk. The bam is updated regularly when additions or changes are made by the system.

Though the application program usually handles these operations, in the background, the user should beware. If there is a power failure, the user turns off the computer before the bam is updated or some other error occurs, the file might not be closed properly. An improperly closed file has an asterick beside it when the directory is displayed.

If files are not closed properly the bam is not updated and your data might be lost. It also appears that the data is written to the disk after the bam has been referenced for free blocks. The bam is not updated unless a dsave or dclose command is issued or periodically when the bam in dos memory is updated. Thus if some calamity occurs, though your data may be on disk, the bam does not know where to find it since it was not updated.

You can use the COLLECT command to delete these open files. Beware that this command also deletes any user - type files. The syntax is as follows - < collect dOonu8 >. Only use lower case letters without the brackets. The small <d1> refers to the drive number and the <onu8> refers to the unit number. The unit number is not necessary if you have only one drive.

The header command erases any information on the disk with the following limitation. If the disk was previously formatted by another disk drive of the same format, and that drive was out of alignment, it is possible that not all of the data was erased. Since this can cause interesting problems, it is best to bulk erase a previously used disk. DO NOT HEADER A DISK THAT HAS YOUR IMPORTANT DATA OR PROGRAMS ON IT. All of us have done this item at least once. So think twice and put labels on all your disks so that you do not inadvertently erase your disk. The header command is permanent, no program or utility can retrieve the data.

The header command has the following syntax but without the brackets, < header "name",dO,iOlonu8 >. The name can be anything length up to 16 characters. The drive number is specified by <d0> or <d1>. The identification <iO1> should be unique for every disk you own. The <i> is required but the following two identifiers can be any characters. The unit number <onu8> is not required unless the unit is not eight.

Well this is all for this issue. We will try to keep this as a quarterly column and cover as much basic material as we can. See you in the next issue.

FOR THOSE WHO KNOW NOTHING ABOUT COMPUTING

By: Mark Schwarzbauer

When my B-128 first arrived I was the proud owner of my first computer. I had no previous computer experience so I started off just like many of you. It took me a couple of weeks to go through the Superscript II manual and learn how to do word processing. I began to wonder if I was ever going to save any time. When I went to tackle Superbase I was even more challenged. I gave up on Calc result because it didn't seem worth my time to learn for my purposes.

I was a member of the original B-128 group from Lompoc California. When I got their newsletter I read it cover to cover and understood about a tenth of it. I was a bit frustrated because it was so far above my understanding and experience. Liz Deal who wrote our Superscript patch first released it through that group. She was disappointed when Marlin turned her simple 4 line machine language patch into a couple of pages of basic programming. But, Marlin did it at my request to make it "KISS" (keep it simple stupid). Sorry Liz but I quess I am partly to blame. But at the same time, it was typing in that program and seeing how it worked that started me really learning how to use this machine.

Of course, all I really needed to do was read my blue and gray manual and I would have learned how to get a directory listing and load and run programs. I know it seems a little bit silly after all this time has gone by but have you read

your computer manual yet? I just read mine for the first time.

I have been preparing the PRECISION CHURCH MANAGEMENT 700 series. In the process of this I have run into a number of frustrating programming problems that have been plaguing me for weeks. But guess what? I read my silly manual and it had the answer! My, my, who ever would have thought it could be there? I had one more problem with Superbase not able to do searches on key fields from a user written program. Well, I was reading through my old bug issues and found a three year old article that repaired my problem in 5 minutes.

So what has this world come to? Do we really have to read? No you don't have to. There are disks in our library, like the GAMES AND EDUCATION DISK and SUPERPRINT that just shift run like Superscript and give you a menu to choose from and most return back to the main menu to make things very simple and easy to use. But if you want to use some of the other disks in the library, yes, you will have to learn to read, arrrghhhh! Oh Well, that's the breaks. If you are sincerely finding the manual tough to understand We would be happy to print more detailed "simple" to use instructions here in the Escape. If this would help you then give me a call and I will respond with an article. But we need to know that you want to know.

Pastor Mark Schwarzbauer
414-743-4151 Central time before 10 pm please.

Consistency in Programming or Loyalty to Those Who RUN a Program

By: Anthony J. Goceliak

This [hopefully short] article was prompted by a letter from a CBUG Member who requested my endorsement of a major re-working of one of my programs. While the members' revisions were on the whole quite sound, I cannot endorse his variation of my program. Why? On the face of it, the new version has much to offer, several added features, and one particularly appealing idea with regards to data input. The problem is that the output [and input] of the program will be incompatible with previous extensive program user input.

While I encouraged him to release his modification to the CBUG membership as a 'new' offering, it is not fair to void all prior input to a data file. Forgive an old man his crustiness, but I write programs with the user (Myself!) in mind, and feel honor bound to maintain compatibility with previously published versions. Apparently not very many of us do, as (merely an example, I'm not picking on SS) a Superscript II user, trying to handle Without external aid, a Superscript III file can attest. It should be incumbant on a programmer to maintain consistency with previously released versions of a program, unless there is both:

1. No external files associated with the program. [i.e. a 'space-zappers v2' instead of 'space-zappers v1']

2. A really COMPELLING reason to change the way things are done. [Changed keys, not extras!]

This does not mean I'm voting against change. What I mean is for instance, if a program used to use key 'y' for yes, instead of ADDING a key 'l' to scroll leftward, there better be a REALLY GOOD reason for telling the user in version number 2 that the 'y' no longer means yes, but instead now signifies 'yaw to the left', scrolling the screen away from him!

I have indeed bumped from time to time into self imposed roadblocks, but if the program was sent out, well, the moving finger wrote, and that's that. Very few of the roadblocks are indeed permanent, and although they may preclude using an 'elegant' solution, it is better to have a long, imperfect program rather than the promise of a short 'gem' in just a few years, or a plethora of incompatible versions buzzing about.

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

THE PENNSYLVANIA CONNECTION

***** ERASING MANY FILES *****

By: Liz Deal

Commodore's file erasing command is called SCRATCH in BASIC 4. As with all pattern matching, you can scratch several files at a time,
SCRATCH U9,D1,"+TEST*"
will scratch all files beginning with a "+" sign, having "TEST" in the second to the fifth position, and followed by anything. However, if you plan to do extensive disk maintenance, you will be calling on filenames with many unique names. In this case typing SCRATCH and all that other stuff begins to be a pain, even if you have defined a function key to do the job. So this tiny program is meant to make the job easier:

```
10 INPUT"FILENAME";F$:IF F$="*"THEN END
    :REM INPUT * TO QUIT
20 SCRATCH U9,D1,(F$):PRINT DS$:GOTO 10
```

U9 stands for disk unit 9, use 8 if that's what you're scratching. D1 stands for drive 1, use 0 if you wish. In some of their documentation CBM uses words like "ON U9", which boggles my mind. You can use ON, of course, but commas suit me just fine.

***** COPYING MANY FILES *****

By the same token, you can copy a batch of files for the prince of inputting just the filenames using this:

```
10 INPUT"FILENAME";F$:IF F$="*"THEN END
    :REM INPUT * TO QUIT
20 COPY U9,D0,(F$) TO D1,"*":PRINT DS$:GOTO 10
```

Bear in mind that unless you use COPY-ALL you cannot copy across devices. And if you plan to rename the files as you copy them, you can write yourself another INPUT statement to pick up destination filename, and then use:

```
20 COPY U9,D0,(F$) TO D1,(DF$)
```

***** LIST/CONVERTER BUG *****

On my UTILITIES#2 disk is a program that lists and optionally converts BASIC programs from one machine to another. It's ironic for us B-users, that there is a bug in there which afftects only B-machines. The bug is that when you ask for screen output, a one-column wide window gets set, and the entire listing flys by in that skinny thing. Rather useless. Fortunately, the program is just fine when disk or printer output is specified. In any case, I've fixed this bug. Any of you who are using this program and wish to have the fix should send me any floppy, a label from that disk, cardboard, stamps and a self-addressed envelope, and I'll pop the fixed version in the mail.

***** SCREEN INPUT BUG *****

Bing Hart reports there is a bug in the Screen Input routine on UTILITIES#2 disk. I use screen input in the NOTEMAKER

program, and it never fails. I use it in several other programs and it never fails. Yet the bug can be demonstrated by, for example, running the Screen Input program twice - garbage gets picked up. It turns out that in all programs in which I use Screen input I set a window (escape-T) and sometimes clear the screen. There is no bug. Yet without setting a window the bug is there and I am puzzled, and will try to report if and when I fix it. If any of you are playing with this routine and come up with a solution, let us all know, please.

***** UTILITIES FOR B128 and B256 *****.

It is Norman's desire that B256 people get rid of their ROMs and put in B128 ROMs into their machines. Good idea. But there will always be some people who will not convert. So for your benefit I decided to make sure that my most important utilities function on the B256. If you get the disk which I'm now submitting to Norman, ignore "B128" in filenames. Chances are good that programs will work in the Big B, so long as the ROMs are the same as mine.

SUPERMON and KEYTRIX (up to V7) always worked on the B256 until I introduced split screen (window setting using function keys 8 and 18) in version X1. It turns out that the keyboard's SHIFT-key is decoded slightly differently in the 256, so the new feature did not work. I've re-written the code to accomodate B256.

EXECUTING commands from a file failed to do the job for the same reason, it is now fixed.

When I was converting Jim Butterfield's COPY-ALL program to run on the B, I chose to ignore B256 altogether. Without the machine, I just could not cope with the silly scattering of BASIC variables all over the memory that CBM coded. I finally bit the bullet, version 7 (v7) can now handle B256 machines.

B-KIT 8/87 disk now becomes the current disk of UTILITIES for both machines.

***** REAL TIME PROGRAMMING
by Caxton C. Foster
Addison-Wesley Publishing Co.
ISBN 0-201-01937-X $9.95 *****

Tiny book review: This is a beautifully written book about the logic of interfacing a computer to the real-world. It is aimed at users of the 6502-family of computers (CBM, Apple, Atari) as well as the Z80 bunch. It is general enough, and contains enough pseudo-code to show how to go about programming such things without getting into the nitty-gritties of a particular computer.

Newcomers will appreciate this book for its clarity. They can then code their own little routines using other books as references. More experienced people will appreciate elegant algorithms and will be able to move on and elaborate on the Foster suggestions.

If you have ever seen Foster's "Programming a Microcomputer", and if yoy liked it, you'll find a similar down-to-earth, clear explanations and suggestions. Real Time Programing to me is a very enjoyable book.

***** THE ELEMENTS OF GRAPHING DATA
by William S. Cleveland
Wadsworth Advanced Books and Software

No, nobody pays me for these reviews. I got the books, 'cause I needed them badly at work. This one was advertised in SCIENCE (a nifty, weekly magazine). The part of the ad that caught my eye was a statement that many people are really good at analysing data, but are unable to COMMUNICATE it to others in form of good graphs. Graphs was what I was doing at work, so I decided to take a shot at this book. It's good. You'll find basics of data presentation as well as some elegant algorithms developed by John Tukey (Bell Labs), Brian Kernighan, and other big figures in statistics.

You will not find 65xx code in this book. You will not find any code at all. But you will find lots of graphs and ideas how to make them better. The book helped me cope with High-resolution graphics on the C128 computer I was using at work. It may help you in whatever you're doing.

***** THANKS *****

Thank you to those who have written about B's POWER SUPPLY in the recent CBUG issues. One day mine got cooked and the articles confirmed that that's what was wrong. North-West Music Ctr replaced in in a jiffy and I'm in business.

***** INTERMITTENT FAILURES *****

When you get a new computer shipped to your house, don't be surprised if it fails to work. My B256 was intermittent. One day I'd turn it on at it was just fine, then on another day it couldn't wake up. The screen was on, the cursor was flashing, but after a long while the computer was braking into the monitor. A tip-off was that if I left it alone turned on, and then switched it off and on again it would work - the familiar "Commodore Basic 4.0" sign was there. Heat? Most likely. This suggested loose chips. I opened the computer up and pushed all the socketed chips harder into the board. I haven't had any trouble since. Clearly the chips worked themselves loose in transit, and the expansion under heat was necessary for them to function.

***** CURSOR, WHERE ARE YOU? *****

The Hi-Profile B256 computers that have an attached swivel-screen have a really weird and annoying feature: the blinking is not disabled during cursos's travels about the screen. Now you see it, now you don't; mostly you don't. Here are two ways to overcome some of this madness:

1. Define a function key to set tabs every 10 positions. Each time you press the key, it will toggle TABs in those places. The cursor then hops and you can see it until the final adjustments are needed.

2. Type escape-E to make the cursor non-flashing. The problem here is that on reverse field you cannot find it. So a solution:

3. Strip upper 2 lines of the cursor by these commands: BANK 15:POKE 212,2. But this also causes some problems: in a program that has INPUT statements there is suddenly no flashing cursor. So it's not a bad idea to issue escape-F at the beginning of such programs in order to enable flashing.

This phenomenon explains a strange piece of code I once found in Superscript 2 and 3: Mr. Tranmer explicitely handles location $D4 (212) in his software. That was puzzling to me using a properly functioning B128, but now I see that his code takes into account the bugs in the system.

THE GERMAN CONNECTION

By: Mark Schwarzbauer

Progress is taking place with our German group. Software arrangments are being made to release products both directions. Cooperation in our hardware area is also speeding up. Our German group is planning on manufacturing the 8088 co-processor board. Of course with CPM 86 up and

running it is great to have. We have 125 Meg of CPM 86 programs coming (thats over 300 8050 disks). This will be the largest CPM 86 library I know of held by a users group. With literally thousands of programs the Co-processor board will give you a whole new computer!

But, lets be honest, my friend Anthony Gociliack thinks we are crazy (he may very well be right) but we want MSDOS and IBM compatability too. When I first got my computer and saw that it was supposed to be MSDOS compatible with a co-processor board I called CBM and tried to get one. But alas, they were not available. Now that I seldom use it because I don't have the ability yet to port over meaningful software for my use and we don't have the 300 disks of cpm 86 yet. (Bruce Faierson at Northwest Music will port over software for people with legal copies such as D-base 2 and Wordstar). Yes we have MSDOS now but it is only version 1.25. We really need to have at least 2.1. However, things may have greatly changed in this area.

Commodore has been so helpful to us. Their giving us support in our effort has made it possible to get as far as we have. Their support will help us achieve even greater accomplishments. We have the personel within our group and help from former CBM engineers and technicians to make this possible. We expect to produce an updated version of a compataible MSDOS vesrion 2.1 or even 3.2! Soon the B-128 will be the total computing marvel it was designed to be. Our friends in Germany are also making great contributions in this direction.

In the next issue we should have a time table put together for the release of much of this material. Stay tuned!

An Uncommon DOS Sort

By: Anthony J. Goceliak

A while back a member wrote to ask if there was a way to allow for directory displays of similarly ending, instead of similarly beginning files.

To briefly clarify, typing the following will yield a directory display of all files on d1 beginning with the letter 'a':

```
directory d1,"a*"
```

but there is no straightforward way to convince DOS to sort for files with an arbitrary suffix, in the member's case a requested distinction between '.src' and '.obj'. Granted, a directory display requesting the following:

```
directory d1,"??????.obj"
```

will distinguish between those filenames which have six letters before the .obj or .src, but what about those with five or seven letters? Typing twelve directory requests is not my idea of efficiency, and frankly it can be improved upon.

There are a number of characters not acceptable to DOS as a part of a filename, and most of these are unacceptable because they are used by DOS to flag some special activity. Enter the equal sign. Perhaps some of you know what this is used for, but the books I got with my b and my 8050 are silent. Rummaging through the roms yields this interesting tidbit.

Several DOS commands, among them the directory command, will accept the proper inclusion of this marvelous goodie, and DOS wedge type programs such as Superscript II's disk mode also honor it. Ok, so what? The single letter after the equal sign is parsed by DOS [that means looked at and interpreted] to indicate a filetype!

Try this on one of your CBUG utility disks which have a variety of files of varying type:

```
directory d1,"*=s"
```

Shazaam! Only seq files will be displayed! Additional selection can be included to display for instance only usr files whose names are three letters long, or prg files with 'c' as the second letter, etc, etc.

The following letters are allowed, s p u r for seq, prg, usr, or rel, and if using only one letter bothers you, go ahead and type all three. DOS won't even care if you mis-spell, as long as the first letter is OK. If you foul up even that, DOS won't even hiccup, but instead simply ignores the extended pattern matching for filetype, and displays all filenames matching the name specification in your command. Typing:

```
directory d1,"a*=g"
```

is therefore the same as typing:

```
directory d1,"a*"
```

since no filetype begins with the letter 'g'.

Now to nuts and bolts. My assembler uses seq files for source code, generates prg files as object code, and when code is to be executed by the disk drive, the code comes out with the added bytes for checksums and so forth as a usr file. Searching a disk for all source files or object files or disk drive programs is therefore a snap, and best of all, no DOS utility needs to be written for this purpose.

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

CBUG GOLDWARE

By: Warren Swan

CBUG GOLDWARE is a refinement of the CBUG Library. The CBUG library contains as many disks with as much B Series software as can be made or found. CBUG GOLDWARE is not a proliferation of a lot of software, but a collection of disks approved by a group of users as being useful and usable. The stress on the library is quantity. The stress on Goldware is quality.

What is Goldware? Goldware disks contain useful, usable programs. You don't have to be a hacker to understand and use Goldware disks. Each disk in CBUG Goldware has to meet certain criteria:

1. The disk should be able to be placed in drive 0 and RUN/STOP pressed to get the first program.

2. This first program must be one of the following:
   - A menu that allows all other programs to be run
   - A start up program that tells what the other files on the disk are, or at least identifies which files are to be run and which ones are to be looked at with a word processor
   - The main program (such as Superscript or Superbase)

3. Each program on the disk must have usage instructions, either in a separate word-processable file, or built into the program. For those programs with separate instructions, it must be obvious to the user which files are instructions and how to view them.

4. Each disk is rated as to how many main programs there are. This is to identify which disks are someone's slap together of 2 or three good programs and which disks are actually chocked full of stuff. If the disk is primarily a data disk (perhaps the Bible in Superscript files with no programs at all) some attempt will be made to indicate the number of such major data files.

5. Every program and associated data when applicable will

be of top notch quality. Programs that are incompletely converted from old 40 or 22 column screens will not be counted. Programs that try to use color or a poke to change the screen mode to graphics or normal will not be counted. Programs that use all upper case but no graphics, or programs that use all lower case for no apparent reasons (except lazy, sloppy conversions from older Commodore systems) will not be counted. Programs must have the "feel" that they were written for the B even if they were actually converted from other systems.

6. Disks of high quality but trivial programs (like the one offered commercially by "Intelligent" Software [quotes mine]) will not be accepted as Goldware.

7. Goldware disks will be selected out of the library, thus insuring that they are not copyrighted or illegally obtained. Disks will not be submitted directly to Goldware. They must first be submitted to the CBUG Library and then approved.

8. Goldware disks are approved by members of the CBUG West local group. This group will retain a listing of disks so approved. No disk is a Goldware disk unless it is registered in CBUG West's listing of Goldware disks. If you're not sure, call Warren Swan: (312) 665-1514 evenings from 6 to 9 Central Time (please no later).

Each month at the CBUG West meetings we will test more of the library disks to be certified as Goldware in the next issue of the CBUG Escape. Members of CBUG West may take home CBUG Library disks that don't conform to the Goldware standard and try to conform them for certification. Disks submitted to the library on a royalty basis that do not meet Goldware standards can only be made to do so by the author of the disk.

Look for the CBUG Goldware stamp of approval and rating in upcoming issues of the CBUG Escape.

USE 1 RAM CARTRIDGE IN PLACE OF YOUR ROM CARTRIDGES

By: Warren Swan

Perhaps you have contemplated buying the RAM cartridge that adds 24K RAM to bank 15. Or perhaps you've already bought it. But now you don't want to have to keep switching between the RAM cartridge, the Calc Result ROM cartridge, and the Word Result ROM cartridge. Those of you who have bought Word Result know that the Calc Result cartridge can be used with it, in place of the cartridge that came with Word Result.

You'll always have the RAM cartridge in the computer when you need it for other programs. Harrison's assembler, for one, uses the RAM cartridge. Liz's keytrix also uses the RAM cartridge if you have it. Now you won't have to keep switching cartridges. Instead, we change the Word Result and Calc Result loader programs to load the bank 15 machine code (normally already in their ROM cartridges) into the bank 15 RAM cartridge.

First we have to save the machine code from the Calc or Word Result ROM cartridge onto a file on the Calc Result and Word Result disks (whichever you have). If you have both of these packages, it is preferable that you use the Calc Result ROM cartridge for both disks. This will cause the Word Result disk to load both Word and Calc Result. If you save the Word Result cartridge code, the Word Result disk will only load in Word Result. Got it?

This is all simple to do. Just follow these steps:

1. Turn off the computer and place the Calc Result cartridge (preferable) or Word Result cartridge into the cartridge port, then turn the computer back on. If you don't have Calc Result, skip to step 7.

2. Place the Calc Result program disk in drive 0. This is not the original Calc Result disk that you received in the package, but the program disk created in the procedure in section 1.6 of the Calc Result manual - your everyday Calc Result program disk. Every time you follow that procedure to create a new 'original' disk, you will have to perform the following steps on the new disk.

3. Now to save the cartridge code onto the disk, type:
   bsave "/calc bank 15",b15,p24576 to p32768

4. Load the Calc Result start up program by typing:
   dload"*

5. List the program (it will fit entirely on the screen) and modify line 20 from:
   20 bank15
   to:
   20 bank15:bload"/calc bank 15
   and press RETURN.

6. Save the program back on the disk by typing:
   dsave"@loader
   Calc Result is now ready to use the RAM cartridge for its ROM code.

7. If you don't have Word Result, skip to step 12; otherwise place the Word Result program disk in drive 0. This is not the original Word Result disk that you received in the package, but the program disk created in the procedure in section 1.5 of the Word Result manual - your everyday Word Result program disk. Every time you follow that procedure to create a new 'program disk', you will have to perform the following steps on the new disk.

8. Now save the cartridge code onto the disk by typing:
   bsave "/calc bank 15",b15,p24576 to p32768

9. Load the Word Result start up program by typing:
   dload"*

10. List the program (it will fit entirely on the screen) and modify line 5 from:
    5 a=0:bank15
    to:
    5 a=0:bank15:bload"/calc bank 15
    and press RETURN.

11. Save the program back on the disk by typing:
    dsave"@booter
    Word Result is now ready to use the RAM cartridge for its ROM code.

12. Now power off the disk and replace the ROM cartridge with your RAM cartridge. You should be able to place either of the program disks into drive 0, press SHIFT RUN/STOP, and away you go.

CBM 8432

Another new CBM-Computer?

Translation by: Jurgen Billhoffer

NO, but a new operating system for the 700 series.

It is called 8432, because
- the 700 is now almost comparable to the 8032
- the 720 can simulate now up to four 8032 simultaneously

<<- the 1Meg upgrade likewise up to 14 8032's>>

The 8432 presents the following advantages to programmers, software producers and the user:

1. Almost all existing programs for the 8032 (incl. machine language and compiled programs) will run without a change. Modifications, if needed at all will be minute.

2. Up to four 8032 programs can be loaded on a single 720 = B128 (two on a 710 = B256 or 256 expanded B128)(14 on a 1Meg expansion). Using only two keys one can switch immediately, at all times between programs, even while the program is running. Returning to a program restarts this at the precise point of interruption. This provides a wide range of possibilities for all programmers, unknown by any computer in its class.

3. The processing speed on 8032 programs will be increased by 30% - 40%. These programs run even 20% faster as if they where converted to the 700 operating system! Besides larger memory requirements there is no valid reason to convert these programs anymore.

4. A few more out of the many advantages should be mentioned here:

   - The keyboard layout is freely variable

   - Changes on the operating system can be realized without changing the EPROMs

   - Toolkits and expansion modules for the 8032 will run under 8432 control. In some cases minor adaptions especially in address 0 and 1 might be necessarily.

   - Under 8432 control there is about 8k more room for machine language, eliminating some space problems in applications.

The 8432 operating system is complemented by a new monitor program with powerfull search and link commands and a integrated assembler / disassembler, witch allows multisegment work.

A program in any of the four segments can, if programmed that way, make use of the other segments as additional storage area. Shortly CBM will release a data control system (CBM software catalog " allegro 84 ") that under 8432 control will allow direct access to about 20,000 data sets with variable length and a access time < 2sec. Such data quantities can only be handled by a harddrive, but a 8250 floppy is still able to handle about 4,000 data sets with an average of 250 digits. <<No word at this time as to the fate of this feature>>.

USING THE 8432

A. Start and loading of programs

The system is started, using the "RUN" key.

The system displays:

> Anzahl der Maschinen? (Number of units ?)

On a 710 the answer is 1 or 2, on a 720 between 1 and 4.

The system displays:

> Welche Tastaturtabelle? (Which keyboard table ?)

As long, as there is no special keyboard table prepared, pressing "RETURN" selects the default table (Ref. E).

The system is now setting up the preselected number of " virtual " units. After completion of the setup procedure the usual 8032 screen appears with the addition of "virtual maschine". The unit is now ready to function like a normal 8032 in loading and running any application program.
Most programs, including machine language and compiled programs, will run without problem. Most exceptions and problems are discussed in section "B" and require very minor modifications.

B. Problem areas

1. Using memory location $0000 and $0001.

These locations are used by the 6509 and can not be used by programs.
Recommendable locations are $0002 and $0094 or $0094 and $0095 (the locations $0094 and $0095 are reserved for the NMI-vector witch is not used).

2. The "USR" - function

This function needs to be differently accessed, since $0001 and $0002 can not be used.
Recommendable locations are $03fe and $03ff, then everything should work.

3. Direct I/O port addressing

If the user program needs direct access to the IEEE - bus, the cassette port or the user port are modifications necessarily. Reasons are:

3.1 These ports are on the 700 in bank 15
3.2 They require a different code
3.3 The pin configuration is different

For detail information refer to the known literature (i.e. "ROM-RAM-I/O Assembler listing " by R. Schineis and O.-M Braun). Then the modification should not present a problem anymore, using the command expansion of the 8432 (section "D") and the routines for accessing bank 15. The routines used in BASIC-4 and assembler programming to access the IEEE port and the cassette port (with plug-in module) are fully supported and function identical as in the 8032.

C. Multitasking

While working on any of the "8032s" one can at any time, regardless of its status, leave this unit and transfer to any other "8032" or the 700 monitor.
By pressing "SHIFT/CBM" the unit displays this menu:

```
Bank 0 = frei (free)
Bank 1 = 8432
Bank 2 = .....
Bank 3 = .....
......
Bank F = CBM-700 (TIM)
```

Any bank showing "8432" represents a "8032" witch will be activated by typing the appropriate bank no. If one of these units was activated before the break, it is displayed in reversed video (the unit in bank 1 is automatically activated after loading).
Selecting this unit again will continue to run this program restarting at the exact breakpoint and the screen with every detail will be restored.
Selecting a different bank (unit) will initialize this "8032"
This feature of switching between different units

at any time without loosing the working parameters of this particular program enables the programmer to work in a very efficient way. I.e. load in

Bank 1 = Editor
Bank 2 = Assembler
Bank 3 = Loader

This saves substantial time in loading and reloading the different modules.

Additional features.

a. Monitor access

Using from the menu "w1" instead of "1" will access the monitor of unit 1. This is very useful to stop a program, where the "STOP" key was disabled.

b. Restart of a unit.

To reinitialize a unit (cold start) a "#1" is entered instead "1" for unit 1.

c. Exit the 8432

To exit the 8432 and jump into the 700 monitor from the main menu (READY status) one types "f" or "x". To restart the system 8432 without loss of programs or datas one types "sys1024".

D. Command expansion.

To communicate between the multiple units and to access every memory location in the 700 directly the commands PEEK and POKE are expanded.

1. peek(a)#i returns the content of address "a" in bank "i"
2. poke#i,a,b writes the value "b" into addr "a" in bank "i"

A number, a variable or an expression can be used to define "a", "b" and "i". If "i" is not used, then the value in Byte 59647 = $e8ff is the default. At start-up "i" = peek(0) that means the actual bank.
If one writes "pokea,b", the present bank is used, but the missing "i" in peek would cause to use the bank in $e8ff.
Should a compiler like the DTL compiler be used, one has to change the commands as follows:

1. For PEEK: poke59647,i:b=peek(a)
   ----
2. For POKE: sys59648#i,a,b
                        --

The underlined parts are only used, if the bank is different from the value in 59647.
The incoherence in the command structure could not be avoided for several reasons (i.e. for the interpreter is POKE a command and PEEK a function, witch makes some difference internally).
The basic goal was to achieve compiler compatibility and a simple, space saving syntax for easy all day usage.

On assembler level there are several routines, easy to use:
For the machine commands "lda" and "sta" in absolute addressing mode there are routines available to access Bytes from other banks.

adr = adrl/adrh = absolute address
seg = bank no. to be accessed
mseg = $e8fe; additional storage

This procedures enable the user to write the value "a" into "adr" and transfer the content of "adr" to "a".

```
sta adr                          lda adr
seg=any        seg=mseg          seg=any        seg=mseg
-------------------------------------------------------
ldx adrh       ldx adrh          ldx adrh       ldx adrh
stx $12        stx $12           lda adrl       lda adrl
ldx adrl       ldx adrl          sta $11        jsr $e90f
stx $11        jsr $e90c         lda seg
ldx seg                          jsr $e909
jsr $e906
```

The y-register stays untouched during this operation and the status register contains the correct values after "lda", the c-flag is maintained during "sta". If during multiple access the address high byte in the x-register remains the same and the bank no. was written to "mseg" then there are very few commands required.

E. Keyboard layout

The keyboard layout is independent from the normal keyboard, it is stored in tables. These tables are 96 byte each and start at $8800 (normal keys), $8860 (SHIFTed - keys), $88c0 (CTRL - keys), $8920 (SHIFT+CTRL - keys) and $8980 (ESC - keys). All of these tables can be changed using the monitor. The position of one particular key is calculated the following way:
Using the "internal number" of a key (i.e. SPACE = 41 or $29) and add this number to the start of the table.
I.e. SPACE in normal mode is located in $8800 + $29 = $8829, Space in SHIFT - mode is located in $8860 + $29 = $8889.

After changing positions in these tables using the monitor, one can save the modified tables with the command:
.s"tx",08,8800,89e0 (x = any identifier)

When restarting the system type the selected identifier at the question " Welche Tastaturtabelle ?" (witch keyboard table ?). Please be aware, that all changes are effective immediately upon entering.
The "8432" system is delivered with the tables "t1" (ASCII) and "td" (DIN).

The keyboard

Please read the following tables using this legend.

1. line: Unshifted key imprint (ASCII normal)
2. line: SHIFT values (for letters only = normal ASCII + 128)
3. line: ASCII value of this key
4. line: INTERNAL NUMBER

The main key array

```
 F1  F2  F3  F4  F5  F6  F7  F8  F9 F10  CD  CU  CL
  CR
234 235 236 237 238 239 240 241 242 243
224 225 226 227 228 229 230 231 232 233  17 145 157
  29
  0   6  12  18  24  30  36  42  48  54  60  66  67
  68

ESC   1   2   3   4   5   6   7   8   9   0   -   =  PND
DEL
     33  64  35  36  37  94  38  43  40  41      43 222
148
```

27	49	50	51	52	53	54	55	56	57	58	45	61	95	20
1	7	13	19	25	31	32	37	43	49	55	56	61	62	69
TAB	q	w	e	r	t	y	u	i	o	p	[	]	RET	
209	215	197	210	212	217	213	201	207	208	141				
9	81	87	69	82	94	89	85	73	79	80	91	93	13	
2	8	14	20	26	27	33	38	44	50	57	58	63	64	
SHL	a	s	d	f	g	h	j	k	l	;	'	²		
193	211	196	198	199	200	202	203	204	58	34				
65	83	68	70	71	72	74	75	76	59	39	255			
9	15	21	22	28	34	39	45	51	52	59	65			
SHT	z	x	c	v	b	n	m	,	.	/	SHT	CBM		
218	216	195	214	194	206	207	60	62	63		130			
90	88	67	86	66	78	77	44	46	47		2			
10	16	17	23	29	35	40	46	47	53		70			

CTR SPACE
160
32
41

The number array

HOME	RVS	GRPH	STOP
147	146	142	131
19	18	14	3
72	78	84	90
?	CE	*	/
	132		
63	4	42	47
73	79	85	91
7	8	9	-
26		15	
55	56	57	45
74	80	86	92
4	5	6	+
25	153	143	
52	53	54	43
75	81	87	93
0	.	00	
176	22		
48	46		

F. Special features.

1. Screen scrolling and stop.

Pressing the "CBM" - key during scrolling causes an immediate stop of the scrolling. Pressing any other key will resume scrolling. While pressing the "CBM" - key limited scrolling is possible, by pressing any other key at the same time. Scrolling will stop again, if this other key is released.

2. Disable bank switching.

To disable the bank switching for any application program by means of " SHIFT-CBM " set the command:

poke 34982,0

To enable the bank switching again set the command:

poke 34982,130

3. Processing time.

The run time of any program can be reduced by appr. 3.5% if the " bank mirror " is switched off. The " bank mirror " displays the proceeding of the " virtual unit ". If switched off, all displays created by ones own assembler routines, written directly to the display will not be visible. If the program does not include such routines, the program will run normally.
This switch can be changed by program or using the keyboard at any time:

	program	keyboard
switch off	poke58734,128	SHIFT+CE
switch on	poke58743,0	SHIFT+CE

4. Program crash

If a program crashes, there are two possibilities to regain operation.

a. SHIFT+CBM is still working:
Leave the program and try to find out what happen by going in the monitor of the crashed unit, or by using a different unit and the expanded PEEK and POKE commands. then correct the error and resume operation.

b. SHIFT+CBM is not working anymore:
Press the reset button on the rear of the unit and reenter the system with SYS1024. Then proceed the same way as in "a".

5. Internal clock.

Unlike on the 8032, if the STOP - key is disabled using " poke144,88 " the clock will continue to run.
Every " virtual unit " has its own clock, witch is running only, while this unit is activated.
Each clock will also stop, while the 700 works in bank 15 (i.e. disk operations) or the program uses subroutines in other segments.

COMMAND EXPANSION.

1. COPY

The command COPY of the BASIC - 4 copies datas on the disc drives. The expanded command COPY# of the 8432 copies memory content between different segments.
The segment witch issued the COPY# command is at all times the destination for the datas. Datas will therefore be written from another bank into the working segment.

SYNTAX copy#xi,a,b,c

PARAMETER
- x mode: **#, dim, & or nothing**
- i source - **segment (i = 1 to 15)**
- a start **address of the source field**
- b end **address of the source field +** 1
- c destination start address

If "a" is used, "x" can not be used. Parameter are not mandatory, but may be left out from right to left. To express "i", "a", "b" and "c" numbers, numerical variables and expressions may be used. If "a", "b" and "c" are left out, the BASIC variables or part of it will be copied:

1.1 COPY# without "a", "b" and "c": copying of BASIC variables.

The following four command structures are possible:

-COPY# copy all BASIC variables from another unit.
This is only used in subroutines, operating in other segments and accessed with GOSUB# (ref. 2).

Operation: all Basic variables will be written from the accessed unit and are available for the program. (In memory is this the area between the pointers 42/43 and 47/48 as well as between the pointers 48/49 and 52/53).

Exceptions: strings defined within the BASIC text in the calling program are not accessible after the transfer.

Correction: i.e. instead a$="xyzabc" type a$="xyzabc"+"".

Note: The number of the accessed bank during GOSUB# is in byte 1823=$071f in bank 15.

-COPY#i is the same function as COPY# .
This command is different only by the segment accessed. The segment represented by "i" (i = 1 to 14) will be accessed. Use caution not to select a segment without a program!
COPY#i is used after GOSUB# to provide the main program with the datas to continue processing or to look at the variables in "i" from a different unit.

Programs using these two commands need to be smaller as the program in the called unit, to avoid writing over its end. After return from GOSUB# are the program margins corrected and COPY#i can be used without precaution.

Correction: The main program can be enlarged using poke42,x:poke43,y:clr before the program starts. This is similar to the " overlay " programming technic as outlined in the CBM programming hand book chapter 4.

The other two command structures are not affected by this problem but can not be used to transfer strings:

- copy#x copies selected groups of variables:
For x are three selections

the simple number variables are copied, not "dim" variables or strings.
The area between (42/43) and (44/45) of the "calling" unit will be copied onto the starting address (42/43) of the "called" unit.

dim only the "dim" variables are copied.
The area between (44/45) and (46/47) of the "calling" unit will be copied starting at (44/45) on the "called" unit.

& the two transfers take place, one after the other.

- copy#xi only used after a gosub#i call.
This is almost the same function, but only copying the variables that existed prior to the copy#xi call in the subprogram. This is very important, since the subprogram in bank i might use "local" variables witch are not used by the main program. Every variable name that is not on the main program variable list will not be copied using gosub#xi.
More details under section 2.

1.2 copy#i,a,b,c (do not use x while using a, b or c)

If "i" is missing, the unit takes the value stored in $e8fb (59643). Using poke59643,i one can preselect the bank accessed.

If "c" is missing, the unit assumes c=a. That means, if the destination-address = source-address, "c" can be left out.

If "b" and "c" are missing and "a" is <32768, "b" is assumed to be $8000 = 32768. If "a" is >$8000 the "b"=0

Examples:

copy#i,32768,34768
Will display the screen of unit "i" on the calling unit.

copy#i,32768,32848,34688
Will display the top line of the screen from unit "i" on the bottom line of the calling screen.

copy#i,0,256,32768
Will display the zeropage of unit "i" on the screen.

copy#i,1024:copy#i
Will transfer the BASIC program and the variables from unit "i". Following this command one has a momentary picture of the program running in unit "i" and is able to investigate its status and its variables without stopping the running program.

1.3 Assembler version of copy#i,a,b,c

If the bank number "i" and the addresses "a, b, c" where defined within the assembler code the routine can be written in assembler:

```
dest  = $e8f0 ; buffer for destination address
mcopy = $e915 ; jump table
; ....
 lda #<a ; low byte source start
 pha
 lda #>a ; high byte source start
 pha
 lda #<b ; low byte source end+1
 pha
 lda #>b ; high byte source end+1
 pha
 lda #<c ; low byte destination start
 sta dest
 lda #>c ; high byte destination start
 sta dest+1
 lda i ; bank number
 jsr mcopy ; copy
```

1.4 Compiler version of copy#

If the DTL-compiler should be used, replace copy with sys(59666).

2. GOSUB#

This command expansion enables the user to avoid the limitations on BASIC program size (normally 32k) and gain processing speed compared to an equal sized large program running under the 700 operating system.
The task on hand should be divided into the main program (normally in unit1) and several larger subprogramme located into banks 2 to 4. From the main program are then the subprogramme accessed using gosub#, instead of gosub.

SYNTAX: gosub#i,n or gosub#i;n

PARAMETER: n line number in the subroutine

i bank number for the subprogram

; instead of comma
used in cases where the subprogram should perform screen output

The value for "i" can also be a variable or an expression, the value for "n" has to be a number.

To perform continuous operations the variables from the calling program need to be moved as well using the copy# command.

If the comma between "i" and "n" is replaced by a semicolon the screen will also switch to the subprogram, if the comma is used the screen will stay in the calling program. The semicolon is therefore used in cases, where the subprogram provides screen output or user input is required. Following a return, the screen will switch back to the calling program.

EXAMPLE for a call into bank 2:

```
Main program              subprogram in unit 2

.......                   .......
200 gosub#2,3500          3500 copy#
210 copy#2                3510 ... text ...
                          3990 return
```

The subprogram can contain one ore more gosub and gosub# commands without problems. The system will remember what kind of gosub was used last and act accordingly at the return command.

If a "syntax error" occurs in a subprogram, the unit then stays active until a direct return command. This enables the programmer to inspect the error, then issue a direct return command and continue the calling program the same way, as if the error had not existed. This feature can be used to induce breaks into the subprogram to inspect ore change variables before returning to the main program.
The copy#ix commands can be used, if strings are avoided.

If the DTL compiler should be used, replace gosub with sys(59672).

8432/e : Command expansion and multitasking.
==========

After the introduction of the new 8432 operating system in 1984 during the Hannover fare by commodore, experts realized soon there where far more possibilities in this system not fully explored in the original released product. Within a short period of time, the following features where developed:

BASIC command expansion as discussed above

Real multitasking with two programmes running at the same time

A new comfortable monitor including an assembler and disassembler effective throughout the whole memory

These new additions to the operating system made the 700 series more valuable for programmers and users alike.

The programmer can:

stop a running unit at any time and and "look" at it from another unit, using the copy# command, or using the monitor to check every byte in memory.

manipulate the "live" computer, since the operating system is located in RAM.

use the 8032 programming aids and implement BASIC extensions.

save time during system programming, by having the assembler, the editor, the loader and the test program available in memory at the same time.

The user can:

without reloading switch between multiple programmes.

share the 700 with other users without unloading his program. The only limitation is that a disk with an open file can not be changed.

The following description addresses especially the multitasking feature. The monitor is a tool for the system programmer and must be ordered separate including its manual. The monitor is a stand alone program functioning without the 8432.

CBM HARD DRIVES

By: Bruce Faierson

NWM Inc. has recently received many inquiries regarding the little known, though highly regarded, Commodore hard drives. For those that are not aware, a hard drive is a fixed magnetic, non-removable storage medium. The 9060 and 9090 type drives are referred to as "Winchester" technology. This type of drive was developed by IBM several years ago.

There are several reasons why these drives are not as well known as the 8050. Like the 8250, there were not many of these drives made. These drives originally sold in the range of $2500 - $3200. The cost made it prohibitive to purchase except for medium to large companies. Finally these drives were released around the time of the Superpet 9000 series. Since Commodore appeared to have lost interest

in the 8000-9000 series during the B series development, the drives were never marketed heavily. The rarity of these particular drives have given them a loyal following.

There are several advantages to owning a hard drive. The primary advantage is for database access. The CBM drives rotate at a rate of 3600 rpms. Since the 8050 rotates at 300 rpms, it can be seen that these hard drives are able to access data at least twelve times as fast as a floppy. As a matter of fact, the 9060 and 9090 have an internal transfer rate of 5 megabytes a second. With Superbase, the access to an indexed record is virtually instantaneous. Unfortunately we are limited by the 1200 cps IEEE buss or we would have incredible speed in loading longer files.

The 9060 and 9090 operate under DOS 3.0. This operating system allows for unlimited file entries, replacement mapping of bad sectors, unlimited relative file lengths and a self locating BAM. The preceding is limited to available disk space. Bad sectors are subtracted from the available number of free blocks. There are very few, if any, perfect drives. Since the bad sectors are not available to write to, there are a few less available blocks on the disk. These drives excel for applications requiring large amounts of random or relative data storage.

To our knowledge there are no programs to backup two hard drives together. If there are we would certainly like to know about them. The only way to copy data to another drive is through a good copy utility. The Jim Butterfield copy utility is perfect for this. Files other than sequential or program must be exported to a sequential format to copy easily.

The 9060 drive stores 5 megabytes on two double sided platters. The 9090 drive stores 7.5 megabytes on three double sided platters. These drive sizes may sound relatively small compared to the average size used for an IBM system. The CBM system stores data more efficiently than the IBM system so you can have more data storage in the same storage medium.

An interesting bit of information on these drives is that while these units have the standard CBM IEEE interface, they in turn are interfaced to a SASI interface. This in turn is interfaced to the ST-225 type drives. This leads to some interesting possibilities for the engineering types out there.

For further information regarding these drives call NWM Inc. and we will try to answer any questions.

BASIC 4.0+ Tutorial (Abridged)

by Warren D. Swan

2 BASIC DATA MANIPULATION - continued from last issue

2.7 Deleting Lines (NEW, DELETE) [Programming 4]

Up until now we have been constructing some very simple little programs. One thing we have noticed is that if a program is already in memory, if you type in more lines, they just get put right into the program in the computer's memory. What if we want to start from scratch and "wipe out" the program in memory? We can do this with the NEW instruction:

```
new
```

The NEW instruction is also the NEW statement, since NEW does not need (or allow) any operands. NEW always does the same thing; it wipes out the program and all variables.

We may decide that we no longer need some of the lines in our BASIC program. For example, earlier we used a PRINT statement to print a prompt (it was a line numbered 50), then found out that the INPUT statement itself can have a prompt. To delete just line 50 all we have to do is type 50 and press return:

```
50
```

When BASIC sees nothing after the line number it knows that you just wanted to delete that line. This will also cause BASIC to wipe out all the variables.

If you want to delete a bunch of lines in a row, you can use the DELETE instruction. The DELETE instruction takes the same kind of operands that the LIST instruction takes, namely a line range. Using DELETE without an operand is the same as using NEW:

```
delete
```

It will delete all lines and wipe out the variables. You can also delete ranges of lines, such as:

```
delete 100-200
delete 5000-
delete -10
```

Remember that changing a program in any way, including adding or deleting lines, always has 2 side effects: a. All variables get wiped out, and b. The program is no longer able to be continued with the CONT command. Later we will learn a third side effect that is caused by changing a program.

So far we have learned some basic programming rules, and some basic ways to manipulate data. The rest of this chapter will delve into techniques that will allow us to do some serious programming.

2.8 IF condition GOTO line# [Program Flow 2]

Up until now the only program flow we have learned is the GOTO (or GO TO) instruction, which allowed us some simple loops that could only be stopped with the STOP key. One of the concepts that separate simple programs like the ones we've been doing from serious programs is the concept of "decision making". And one of the instructions that BASIC has that allows our programs to make decisions is the IF/GOTO instruction. Actually there are 2 parts to this instruction. The first part consists of the keyword IF and its operand, an expression. The second part consists of the keyword GOTO and its operand, a line number. (In this case, the word GOTO cannot be split up into GO TO.) An example of an IF/GOTO is:

```
100 IF a>3 GOTO 200
```

The expression a>3, also called a condition here, must evaluate to TRUE in order for BASIC to go to line 200. If the expression a>3 turns out to be FALSE at this point, the GOTO 200 is ignored. A better example might be:

```
10 rem print the numbers 1 to 20
20 a = 1
30 print a
40 a = a+1: if a<21 goto 30
```

Line 20 sets A to 1, which then gets printed in line 30. Line 40 changes A to 2; and since 2<21, it goes back to line 30 and prints the 2. This is repeated for each of the numbers up to A=20. Then in line 40, when A is incremented to 21, the condition 21<21 is false, so the program just "continues on." In this little example there is nothing to continue on to, and the program simply ends. We could have used the condition A<=20 in the if statement. Then it would be easier to see that A will go up to 20.

In reality, the "condition" can be any expression. BASIC only checks to see if the result of evaluating the expression is zero or non-zero. A zero result means FALSE, and any non-zero result means TRUE. Remember that the expression A<=20 results in the true value of -1 if A is (say) 2. Since -1 is non-zero, the consequence is that

BASIC will execute the GOTO part.

The reason we mention this here is that this feature can come in very handy in BASIC programming. A common usage is to set a variable to any non-zero value if you want something to be executed that normally isn't (or vice versa). For example, when debugging our program, we might want it to stop periodically so we can check some variables:

```
10 rem print the numbers 1 to 20.  set db to any non-zero number to debug.
20 a = 1
30 if db goto 1000: rem if debug is on, stop.
40 print a
50 a = a+1: if a<=20 goto 20
60 end
1000 stop: go to 40
```

Now to debug the program, all we have to do is insert a line such as:

```
15 db = 1: rem turn on debug
```

Everytime it gets to line 30 it will evaluate the simple expression "db" and will find out that the result is 1. It will then transfer to line 1000 and stop. After you have checked out the variables, you type:

```
cont
```

and the program will continue with the go to 40, where it will go back into the loop. Doing this we find that for some reason the variable A keeps getting set to 1 every time through the loop. A little study shows us that we have a mistake in line 50, which should read:

```
50 a = a+1: if a<=20 goto 30
```

not goto 20. You may wonder why we even used db if it is only checked for once. Normally we wouldn't, but in a very large program we could have many checks for db, some of them causing the program to stop, and others perhaps just causing the program to print out the values of some key variables.

Expressions can be very complex in an IF/GOTO:

```
10 input"What is your friend's name"; name$
20 if name$="stop it" goto 1000
30 print "Oh yes, Good ol' ";name$;"!
40 print "And what was ";name$;"'s ";
50 input"friend's name"; name$: go to 20
1000 end
```

This nosy little program will keep asking for names until you type in

```
And what was Fred's friend's name? stop it
```

The condition expression in line 20 checks to see if the typed in name$ was "stop it" and if so the result is true (-1), thus executing the goto 1000 and ending the program.

Other examples of complex expressions include:

```
100 if a<5 and b>100 goto 1000
200 if exempt or not (name$="Fred" and tax>.2) goto 2000
```

When used as such, the variable exempt should be set to either -1 (true) or 0 (false) and no other value. This can be done by a statement like:

```
180 exempt = salary <= 3000
```

If salary is less than or equal to 3000, exempt will be set to -1 (true), otherwise it will be set to 0 (false).

As with GOTO, it does not make sense to put another statement on the line after an IF/GOTO:

```
200 if a<20 goto 100: a = a+1
```

If a is less than 20 it will go immediately to 100 and ignore the a = a+1. If a is greater than or equal to 20, it will skip to the next LINE (NOT the next STATEMENT). So either way it will never execute the second statement on line 200.

2.9 IF condition THEN statements [:ELSE statements] [Program Flow 3]

A more flexible statement than IF/GOTO is IF/THEN/ELSE. Instead of just allowing a line number to transfer to if true (although it does allow that), it also allows you to use one or more statements to be executed if the condition is true. Optionally, it also allows you to use one or more statements to be executed if the condition is false. This is a rewrite of a previous example:

```
10 rem print the numbers 1 to 20.  set db to any non-zero number to debug.
20 a = 1
30 if db then stop: rem if debug is on, stop.
40 print a
50 if a<20 then a = a+1: go to 30
```

Line 30 is an example of a very simple example of an IF/THEN. It is much easier to read than the earlier example that used IF/GOTO. It works the same way too. Just type CONT to continue after each time it stops.

Notice that we rewrote line 50. The IF/GOTO example was perfectly OK, but we rewrote this to remind you that several statements can follow the THEN. Please note that when the condition (a<20) is FALSE, BASIC will skip to the next LINE, not the next STATEMENT, just as did the IF/GOTO. In this case there is no next statement, so it ends.

IF/THEN can behave just like IF/GOTO because the "statement" after the THEN can actually be just a line number:

```
50 a = a+1: if a<=20 then 30
```

In this case it is identical to IF/GOTO.

IF/THEN also allows an ELSE part which tells BASIC what to do if the condition is false:

```
10 rem find the largest of 2 numbers:
20 input "What are the 2 numbers you want tested"; n1, n2
30 if n1=n2 then print "Both numbers are the same.": end
40 if n1>n2 then max=n1: else max=n2
50 print "The larger is "; max
```

Notice that the ELSE part must be separated from the THEN statements by a colon. We could have used more than one statement after either or both the THEN and ELSE part. The "statement" after the ELSE could instead be a line number also, just as with the THEN:

```
1000 if tax>income then 666: else 2000: rem do i have any money left?
```

Any BASIC statements can be used after a THEN or ELSE, even another IF/THEN/ELSE or IF/GOTO. The tutorial disk explains why trying to nest an IF/THEN/ELSE in the THEN part of another IF/THEN[/ELSE] doesn't work in BASIC 4.0+. While the following will not do what you might have intended:

```
200 if a<b then if a<c then l=a: else l=c: else l=b
```

the statement can be rewritten so that it does work, as follows:

```
200 if not a<b then l=b: else if a<c then l=a: else l=c
```

The reason for this is explained on the tutorial disk.

The tutorial disk also suggests some "good programming practices" at this point. If followed they will make your programs easier to read and understand when you or someone else tries to modify them later.

2.10 Looping: FOR/NEXT [Program Flow 4]

Earlier we learned a simple techniques to make the computer "loop" back to execute statements over and over:

```
10 print "I are a genius!"
20 go to 10
```

Which must be stopped with the STOP key (or turn the power off!). Then we got a little more sophisticated:

```
10 a = 1
20 print "I are a real genius";a;"times!
30 a = a+1: if a<238 goto 20
```

But BASIC has a pair of statements that make programming loops very simple and more readable. A FOR statement is placed at the top of the lines that are to be executed in a loop, and a NEXT statement is placed after them:

```
10 for a=1 to 237
20 print "I are a real genius";a;"times!
30 next a
```

This example does the same thing as the previous example. The statements that are to be repeated (here the print) are called the "body of the loop". The FOR statement tells BASIC to set the "looping variable" (here A) to 1 for the first time through the loop. The NEXT statement tells BASIC to increment A and check to see if it is beyond the limit given in the TO clause of the corresponding FOR statement (here 237). If the looping variable A is less than or equal to the upper limit, it will loop back and execute the body of the loop again. It will continue to do this until the looping variable oversteps its limit. In this case, it will execute the print statement 237 times.

The looping variable may be used inside the loop just like any other variable, but be careful about changing it. If we were to add this line to our example:

```
25 a = 3
```

the loop would never end (until STOP keyed). A would never be allowed to get up to 237.

Commodore BASIC does not require that you use the looping variable name in the NEXT statement in most cases. For example:

```
100 for i=1 to 12: print: next: rem skip 12 lines
```

It knows that you meant next i here. In fact, this will execute a smidgeon faster than if we had said next i!

The FOR statement can cause the looping variable to go from any number to any number:

```
100 for zz = 105 to 267 step 13
110 print zz;
120 next: print
```

The STEP clause can be added to a FOR statement to cause it to increment (or even decrement) the loop variable by something other than 1. If the STEP value is negative, the loop will continue until the looping variable has gone below the limit value:

```
100 for zz = 105 to -28 step -13
110 print zz;
120 next: print
```

This will print:

```
 105  92  79  66  53  40  27  14  1 -12 -25
```

Since -25 minus 13 is -38, which is below -28, it will stop at -25. The step value does not have to bring the looping variable to exactly the value specified after the TO (the limit value).

The starting value, limit value and step value may all be variables if desired:

```
50 n = 71: aside = 11
100 ford = 1 ton step aside
110 print "How do you like them apples?"
120 next
```

This will loop the looping variable D (cute, eh?) from 1 to whatever the value of variable N is in steps of the variable ASIDE (or AS). Sorry about the poor punctuation there.

Loops can be "nested" inside of each other:

```
100 for x=2 to 12
110 for y=2 to 12
120 print x;"*";y;"=";x*y
140 next y
150 next x
```

Notice that the FOR Y/NEXT Y loop has to be entirely nested INSIDE the FOR X/NEXT X loop. We call the Y loop the inner loop, and the X loop the outer loop. Several loops can be nested inside each other, but things can get more obscure than U.S./Soviet relations in a little while if not kept simple.

In this last example the Y and X did not have to be specified on the NEXT statements, but there are times when the looping variable must be included. This program finds the sum of three sets of positive numbers. Each set may have up to 10 numbers. To terminate a set early, just enter a negative number:

```
100 for set=1 to 3
110 print "This is set"; set
120 print "Enter up to 10 positive numbers.  A negative number ends the set.
130 sum = 0
140 for number = 1 to 10
150 input x: if x<0 goto 180
160 sum = sum + x
170 next
180 print "Sum ="; sum
190 next set
```

Line 190 MUST say next set rather than just next. The reason for this is that the inner (number) loop may be aborted by a negative value. If we didn't use next set it would think that this was still the next for the number variable. BASIC cannot "see" the program as we do to see that there is an inner and outer next here. It only thinks it has encountered the outer loop's NEXT if (1) the inner loop was finished (had executed 10 times), or (2) if the looping variable is used with the next. In the latter case it then knows to "throw away" the information it had been remembering about the inner loop when it encounters the next set. Moral of the story: WHEN DOUBTFUL, USE THE LOOPING VARIABLE ON THE NEXT.

The tutorial disk discusses another feature of the NEXT instruction that allows you to end several FOR loops with one NEXT statement. It also gives 4 reasons why not to use that feature though.

2.11 Arrays; Subscripts; DIM [Handling Data 5]

In school we learned to use variables to refer to arrays (or matrices) or numbers using subscripts. Thus, one variable name could be used to represent a whole set of numbers, arranged in columns and rows. An array could contain just one column and several rows, or one row and several columns, or several rows and columns.

BASIC allows us to work with arrays of numbers that can be ANY number of dimensions. For example, we could have several planes of rows and columns, resulting in 3 dimensions. We will start by showing the simple case of a one dimensional array:

```
100 a(0) = 10
200 a(1) = 30
300 a(2) = 700
400 sum = 0
500 for i=0 to 2
600 sum = sum+a(i)
700 next
800 print sum
```

As you can see, to give a subscript to an array variable, we

THE CBUG LIBRARY

The character of the CBUG library this issue shows some response to the plea of many members for nontechnical materal. At the same time there are several disks for the highly advanced technician which appear in the latter third of the listings.

As previously published, CBUG has been honored to receive a great deal of important source code which we are now releasing to our members. Remember that these disks are not to be released outside of the CBUG organization without written permission. Lest the non-technical members feel they are being given short shrift, we must all remember that with this information, major contributors such as Kernagan, Swan, Deal and miriads of othes who have not published as extensively if at all, will be able to make rapid and important advances in programming for use by the rest of us lay B128 owners. The objective is not to turn our general membership into computer scientists, but rather to allow those scientists amongst us to more readily provide user friendly, non-technical applications. The rest of us ought study what is published by CBUG which is more appropriate to our expertise, interest, and available time.

This issue has a number of religious disks, attributable to Mark Schwarzbauer and Jon Whatley.

One of the most valuable entries is a A very important new work is the radically revised 8432 emulator. It is reported to allow multitasking (running several programs simultaneously), machine language, and other needed features. It is designed to operate with 8032 software, though many shorter B128 programs can be easily modified to run on the B128. Some such programs are already in the CBUG library.

Dave Wacks, a member out of touch with the group for almost two years has come forth with what looks like an interesting disk. Most certainly his method of presentation and main menu are quite different -- and worth making comment on in your future communications with CBUG, librarian, etc.

I appologize to the few of you who received defective disks from a period of about March to July this year -- and there may be a few left in inventory. It seems that two of our duplication drives developed a habit of putting in (or taking out) their own two cent opinions about certain bits -- at random of course. We finally caught them red-handed (or bitted rather) and imposed a long term sentence of sitting in our favored repair shop for fix and several weeks of continuous performance testing. The jailer (Mike Shartiag) did an outstanding job re-educating these two electronic levitators. Even then, they will spend a few months in the halfway house (my computer room) before returning to duplication duty.

TO OUR CONTRIBUTORS: We would like to get the Fall 1987 ESCAPE into the mail before Christmas -- I mean Christmas 1987! That's a tall order I know, but worse yet, we would appreciate having materials in by early November at the latest. If you find yourself past the closing date, phone us -- maybe we can hold room, etc.

IMPORTANT LEGAL NOTICE TO OUR CONTRIBUTORS. Please read elsewhere in this issue about our cross publishing agreement being firmed up with The Cologne 600/700B User group. In essence we are agreeing to exchange master disks without costs with the sister group. Said disks may not be altered except as to language and possible idiosyncrnisms between our respective machines. Author royalties will be added to base prices charged, then remitted in bulk to the author's group, thence distributed domestically. If you have any problems with this proposition or, if you are a contributor, with any or all of your materials being distributed in western Europe etc., let CBUG know no later than Yesterday!

This is an upgrade of Liz's #2 utilities. There are several new entries on this disk as well. Note, however, the 8432 emulator is not the latest version found on CBUG #66 in this issue. For members who already purchased earlier versions of Liz's utility disk(s), you may attach the label from your existing disk and write in stock #11663 for the half price up grade upgrade disk.

B128-kit August-87 Liz Deal. All programs on this disk also run in a b256 I have, But I haven't tested the pet.emulator section.

```
1  "<----(c)---->"    seq
?? "disk contents"    prg load,list
6  "important mesg"   prg run & read
1  "<---- b ---->"    prg
9 >> "tsk"           >> prg run to configure the machine
                         and to load
2  "+btxfer.fe48"     prg transfer sequences, supermon,
1  "<---- 1 ---->"    prg         (t)          (s)
                                          keytrix
                                            (k)
16 "+smb v8.0400"     prg supermon obj. files. $400
                          version loads
16 "+smb v8.e000"     prg unless you modify tsk loader
16 "+smb v8.0400+"    prg
16 "+smb v8.e000+"    prg
16 "+smb v8.0400++"   prg
16 "+smb v8.e000++"   prg
17 "supermonb v8.ins" prg load/run to list/print
1  "<---- 2 ---->"    prg
17 "+b1 kxb128.x2"    prg keytrix (basic utilities) obj.
                          files
4  "+b15kxb128.x2"    prg
2  "+b15kxb128.5c0"   prg -special short version (see
4  "+b15kxb128.800"   prg    important message)
4  "+b15kxb128.x2+"   prg
4  "+b15kxb128.x2++"  prg
17 "+b1 kxb256.x2"    prg
4  "+b15kxb256.x2"    prg
2  "+b15kxb256.5c0"   prg
4  "+b15kxb256.x2+"   prg
4  "+b15kxb256.x2++"  prg
60 "keytrix x2.ins"   prg load/run to list/print lots of
                          instruct.
1  "<---- 8 ---->"    prg
21 "dcl b128.pal"     prg b128 declares for pal
8  "dcl b128.sym"     seq b symbol table
2  "mykeys s8kx2"  prg if you clobber my f-keys, run this
2  "+4keywords"       prg load to $1f000 to list +4 prg
                          via kytrx
1  "<---- k ---->"    prg
4  "change dev#"      prg various disk utilities converted
26 copy all-b v7" pgm  from jim butterfield's pet versions
3  "+copyb.21c00"     prg
8  "disk view.b128"   prg
11 "disk logger.b128" prg
1  "+disklog.400"     prg
7  "disk mod.b128"    prg
15 "disk check"       prg
5  "stringthing b128" prg
4  "big rel file"     prg how to set up small/big relative
                          file
17 "reader/writer.3"  prg decodes cabs files to screen/
                          printer/dsk
1  "<---- i ---->"    prg
7  "execute.v1"       prg executes commands from seq file -
1  "e.demo"           seq    load & run to see how it works
2  "+xec b128.f0400"  prg
2  "+xec b128.f5a00"  prg
2  "+xec b128.f7a00"  prg
2  "+xec b256.f0400"  prg
2  "+xec b256.f5a00"  prg
2  "+xec b256.f7a00"  prg
34 "xec.pal"          prg pal source
3  "e.xall"         seq example of prg modification in exec
1  "<---- t ---->"    prg
18 "list/conv v4c"    prg lists & converts basic programs
10 "+pet2.klst v4"    prg
5  "+cross.b128"      prg
2  "ramtestb.v1"      prg tests memory in b - list &
                          modify to
1  "+ramtestb.v1"     prg test any ram
15 "ramtestb.v1.pal"  prg
1  "<---- c ---->"    prg
6  "pre-superscript"  prg keybounce killer for ss-ii
15 "b128 irq rate"    prg logic behind it
2  "seqfilesplitter"  prg files too long? edit & run this
2  "q j at stripper"  prg files too messy? strip the
                          garbage
4  "scrn input b128"  prg permits screen input without
                          errors
5  "notemaker.v5"     prg makes seq files from what's on
                          screen
12 "notemaker.ins"    prg load/list for instructions
7  "spool b128"       prg spooling disk to printer
10 "w.b128 spool"     seq explained
9  "alarm thought"    prg just playing with the tod clock
7  "3tod clock.pal"   prg
1  "<---- d ---->"    prg
6  "xcall"            prg how to call non-kernel routines
1  "+xcall.fe00"      prg   from non-system bank
80 "w.b128 xcalls"    seq lotsa explaining here
                          (ss-ii file)
29 "windows.pal"      prg pal source for windowing
                          routine
2  "datamaker"        prg makes data statements fror
                          mc bytes
4  "verifizerb128"    prg see transactor
27 "w.exp15"          seq how to expand bank 15 in the
                          cartridge
1  "<---- f ---->"    prg
23 "grungy towers"    prg bunch of jim butterfield's
15 "diary"            prg   programs
2  "events"           seq
12 "twin bagels"      prg
3  "waitline"         prg
1  "<---- j ---->"    prg
6  "areacomputer"     prg computes area within set of
                          points
2  "egcff"            prg finds gratest common factor
5  "tab"              prg tab values into categories
4  "sm sort"          prg shell-metzner pointer sort
8  "words match"      prg template for teachers
11 "squares"          prg game
16 "rotate"           prg game
3  "primes"          prg transactor's prime number finder
1  "<--8432 utl->"    prg pet 4.0 emulator program - see
3  "Start8432.21"     prg instruction file
2  "t1"               prg
2  "td.alt"           prg
2  "t2"               prg
2  "td"               prg
81 "8432.21"          prg
4  "bank F.20  '1024" prg
5  "instructions8432" prg
9  "supermon4.rel"    prg supermon for emulator. load & run
5  "aid4.6b00"        prg basic aid. load & sys the given
5  "aid4.7b00"        prg    address
6  "cross"            prg crossreferences basic in
                          emulator
1  "<---- e ---->"    prg
24 "b128<->am2.b+pal" prg b-amiga file transfer program.
2  "+b128<>am.2"    prg  Use null modem (cross lines 2/3)
1  "<---- h ---->"    prg
7  "4023 scr dump"    prg normal resolution slow but
```

```
10 "+pet4.klst v4"    prg
10 "+c64 .klst v4"    prg
10 "+p14 .klst v4"    prg
10 "+c16 .klst v4"    prg
10 "+b128.klst v4"    prg b needs cartridge memory to run
10 "+b256.klst v4"    prg b needs cartridge memory to run
10 "+c128.klst v4"    prg
24 "list all"         prg jim b's lister
5  "c128 draw"        prg couple example programs to play
                          with
9  "c64e.prog"        prg
5  "ted prg.draw"     prg
1  "<---- a ---->"    prg
4  "cross.b128-1.1"   prg jim b - crossref maker for basic prgs

8  "p.dump ins"       prg   exact basic screen dump
5  "p.cats"           prg
1  "<---- m ---->"    prg
17 "4023 hr dumps"    prg high-resolution dumps to 2 printers
18 "6400 hr dumps"    prg -load c64/c128/+4 screens into b at
1  "+b128/4023.10220" prg $1a000, bload the +files and run
1  "+b128/6400.10220" prg 5000. explanations of parameters
32 "px0.pixpic3"      prg after line 5000.
1  "<--the end-->"    prg
35 "cbug 8/87         seq article summer 87 issue
6  "imp mess seq"     seq   seq file same as above =title
31 "disk conts seq.   seq   seq file same as above =title
 697 blocks free.
```

!!!!!SUPER TEACHER!!!!! CBUG #61 NEW RELEASE #11682

From Jon Whatley
A Superbase Application (SB I, will self adapt to SB II)

This is the successor to CBUG #60. If you plan to use this suite, take heed. If you teach more than a few courses, such as in a small school, you may need to split the databases off of one disk onto several -- or make several copies of this disk and divide up your work load. You'll need the extra disk space. If, however, you are in a more customary environment, there should be little risk of running out of memory on one disk even if using all the features.

You may upgrade your existing #50 Educational Records disk at half price by returning the label off of your #50 disk and writing in stock #11644 on the order form

Jon writes:

"I'm submitting a drastically updated version of CBUG #50 Educational Records disk. This release is a result of extensive use of the hi-tech programming covered in SUPERBASE THE BOOK plus realized additional needs of good classroom management for a really super teacher. Thus, I have named this new version !!!!!SUPER TEACHER!!!!!

Additional features to those previously listed in Educational Records CBUG #50 are:

1.) Completely menu driven, including database menus as well as numerous program menus.
2.) All on-screen field name selections (i.e., no program listing necessary to insert field names, etc.)
3.) A very helpful registration program to set up student files.
4.) A much improved method of tracking absenteeism (if needed by the teacher).
5.) A manual entry print-out which includes field data to keep the user from posting grades to wrong data fields.
6.) Much improved grade posting programs with accompanying print-outs.
7.) Much improved progress report print-outs plus a special progress reporot for sudents participating in extra-curricula activities. Prompts for teacher comment and comment not only printns out in the progress report but also is stored in the student file along with the date of the comment/report.
8.) HELP SCREENS FOR JUST ABOUT EVERY FUNCTION
9.) A system for keeping tract of delinquent student work (work not handed in due to absenteeism, etc.)...
   a.) Automatic annotatin of the file of the name of the missing work when a zero (0) is posted during regular grade posting routines.
   b.) The grade posting print-out indicates the reason for a zero and the ranked listing indicates an "Incomplete" statement when a zero is averaged in due to a delinquent assignment or work not handed in on time.
   c.) A "delinquent work" posting mode which deletes delinquent work annotations from the file when the grade is finally posted.
   d.) An auditing mode which allows annotated file checking before a ranked grade listing is printed out.
   e.) A print-out of the list of students and the names of their delinquent work (suitable for bulletin board) which can be printed out daily or immediately after initial grade posting routines.
   f.) A print-out of individual student notices which includes the name of the student and the name of the delinquent work (description).

I have used the system for two complete semesters and I believe all the bugs have been found and removed. The students really appreciate seeing their grade average and rank in the class on a daily basis. Many of them actually compete with each other to jump on another on the rank listing --- a new learning motivator created thru the use of a computer? (I post each grade posting print-out of each class on my bulletin board.)

The delinquent work tracking feature is really a SUPER FEATURE! I love it! The one thing that drives teachers up the wall is keeping up with all the delinquent work. Almost every day in every class at least one student is absent who misses an assignment and keeping track of this really can mount up!

By the way, at North Texas State University, I was given a demonstration of about six different classroom managment systems off the shelf by the School of Education. Many of them had many different features but primarily in the area of statistical data manipulation out-put and some included beautiful full color bar and pie graph capabilities, etc. Now these were all self-contained software systems (not reliant on a database) and were very well written for what they could do. BUT, they did not seem to fulfill my every practical need for everyday classroom management. The system I have

written does fulfill this need and at this point I believe I have exhausted the capabilities of SUPERBASE for this purpose as far as my own personal knowledge of SUPERBASE is concerned.

What I have not been able to include is an automatic form feed code at the end of print-outs. I REALLY WANTED THIS BAD but no one seems to know it. <<HEY CBUGgers! lets get some one to answer that one so we can print it next issue!>> Both SUPERBASE THE BOOK and the users manual mention use of "pring@0,12" as a form feed if the printer code is 12. But on our B-128 SUPERBASE, I have found no evidence that this can be anything but a syntax error or illegal parameter problem. Can anyone help me? I would still like to add a form feed code to the end of many of the programs in SUPERTEACHER for greater work efficiency.

```
1 "edu records     " 01 2c     6    "manual entry rec" seq    2    "final.p"          seq    3    "quiz 2"           seq
4   "start.p"            seq    3    "quiz 9"           seq    1    "QUIZ"data ii      seq    3    "quiz 3"           seq
4   "henter"             seq    4    "pd2 daily record" seq    4    "quiz format.p"    seq    3    "quiz 4"           seq
5   "hselect"            seq    4    "pd1 daily record" seq    2    "key.p"            seq    3    "quiz 5"           seq
8   "hfind"              seq    4    "pd3 daily record" seq    7    "instructions.p"   seq    3    "quiz 6"           seq
5   "houtput"            seq    4    "field poster.p"   seq    4    "study guide.p"    seq    3    "quiz 7"           seq
5   "hcalc"              seq    4    "menu2.p"          seq    5    "menu6.p"          seq    3    "quiz 8"           seq
5   "hreport"            seq    3    "graph.p"          seq    1    "QUIZ"data iii     seq    3    "chapter 9"        seq
4   "hexecute"           seq   13    "report3.p"        seq    1    "QUIZ"data iv      seq    3    "quiz 10"          seq
5   "hhelp"              seq   13    "report2.p"        seq    1    "QUIZ"data v       seq    3    "quiz 11"          seq
5   "hfile"              seq   13    "report1.p"        seq    1    "QUIZ"data vi      seq    3    "quiz 12"          seq
7   "hformat"            seq    3    "progress reports" seq    1    "QUIZ"data vii     seq    3    "quiz 13"          seq
4   "hbatch"             seq    3    "daily records.p"  seq    1    "QUIZ"data viii    seq    3    "quiz 14"          seq
5   "hsort"              seq    4    "grade posters.p"  seq    6    "start II.p"       seq    3    "quiz 15"          seq
4   "hprog"              seq    3    "lists.p"          seq    5    "menu7.p"          seq    3    "quiz 16"          seq
4   "hmaintain"          seq    2    "absence data.p"   seq    5    "menu8.p"          seq    3    "quiz 17"          seq
3   "hmemo"              seq    4    "registration2.p"  seq    5    "menu9.p"          seq    3    "quiz 18"          seq
7   "hmenu"              seq    3    "registrations.p"  seq    5    "menu10.p"         seq    3    "final"            seq
3   "hcommands"          seq    5    "hregistration.p"  seq    5    "menu11.p"         seq    3    "chapter 16"       seq
3   "himport"            seq    3    "absence list2.p"  seq    5    "menu12.p"         seq    3    "chapter 17"       seq
3   "hexport"            seq    6    "missing work.p"   seq    5    "menu13.p"         seq    3    "chapter 18"       seq
7   "hlabels"            seq    3    "missing work lis" seq    5    "hltrgd.p"         seq    3    "chapter 2"        seq
2   "cust.rec"           seq    3    "missing work2.p"  seq    5    "fexam calc.p"     seq    3    "chapter 3"        seq
2   "cust.inv"           seq    3    "notices.p"        seq    6    "start III.p"      seq    3    "chapter 4"        seq
1   "TRAINING"           seq    3    "sp prog rept lis" seq    1    "REVIEW"i          seq    3    "chapter 5"        seq
1   "STUDENT"records     seq    3    "ID list.p"        seq    4    "menu14.p"         seq    3    "chapter 6"        seq
12  "period 1"           seq    2    "manual entries.p" seq    4    "menu15.p"         seq    3    "chapter 7"        seq
12  "report.p"           seq    3    "manual entry2.p"  seq    4    "menu17.p"         seq    3    "chapter 8"        seq
3   "student list.p"     seq    3    "hprocedure.p"     seq    4    "menu18.p"         seq    3    "chapter 10"       seq
4   "grade list.p"       seq    5    "hposter.p"        seq    4    "menu19.p"         seq    3    "chapter 11"       seq
4   "ranked list.p"      seq    3    "features.p"       seq    4    "menu20.p"         seq    3    "chapter 12"       seq
7   "daily record.p"     seq    5    "hprogress.p"      seq    4    "menu21.p"         seq    3    "chapter 13"       seq
3   "ltrgd.p"            seq    6    "absences.p"       seq    4    "menu22.p"         seq    3    "chapter 14"       seq
3   "absence list.p"     seq    6    "hlists.p"         seq    1    "REVIEW"ii         seq    3    "chapter 15"       seq
3   "absence poster.p"   seq    4    "hdaily records.p" seq    1    "REVIEW"iii        seq   12    "period 2"         seq
2   "clear files.p"      seq    5    "hfield poster.p"  seq    1    "REVIEW"iv         seq   12    "period 3"         seq
8   "grade poster.p"     seq    5    "hmissing work.p"  seq    1    "REVIEW"v          seq   12    "period 4"         seq
8   "grade poster2.p"    seq    4    "hmissing work li" seq    1    "REVIEW"vi         seq   12    "period 5"         seq
8   "grade poster3.p"    seq    5    "hposter tips.p"   seq    1    "REVIEW"viii       seq   12    "period 6"         seq
3   "absence record.p"   seq    6    "hcalculations.p"  seq    1    "REVIEW"vii        seq   12    "period 7"         seq
5   "registration.p"     seq    7    "missing work3.p"  seq    5    "review format.p"  seq    1    "hlist"            seq
4   "menu1.p"            seq    1    "QUIZ"data         seq    3    "quiz 1"           seq 1185 blocks free.
```

SUPER CHURCH CBUG #62 NEW RELEASE #11697

From Jon Whatley
A Superbase Application (SB I, will self adapt to SB II)

The object of this database appliction is obvious. However there are certainly many other applications it can be applied to from medicine to counseling. CBUG would like to hear from you as to what other applications members can devise!

```
1 "visitation      " 01 2c     3    "field poster.p"   seq    1    "visitation list." seq    2    "singleparent.p"   seq
8   "chdirector"         seq   26    "registration.p"   seq    2    "visitation updat" seq    2    "singlesvisits.p"  seq
1   "VISITATION"         seq    1    "contact list.p"   seq    3    "contacts.p"       seq    2    "pastorvisits.p"   seq
6   "start.p"            seq    2    "membership.p"     seq    2    "teenvisits.p"     seq    9    "hvisitation.p"    seq
7   "visitation.p"       seq    2    "additions2.p"     seq    1    "hlist"            seq    2    "director's list." seq
5   "menu1.p"            seq    1    "print clear.p"    seq    3    "additions.p"      seq    9    "help.p"           seq
1   "menu2.p"            seq    1    "clear added.p"    seq    1    "clear print.p"    seq 1932 blocks free.
```

THE KING JAMES NEW TESTAMENT CBUG #63 NEW RELEASE #11709

THE NEW TESTAMENT ON DISK! In a two disk set we are releasing the NEW KING JAMES NEW TESTAMENT. The N.T. is saved as

individual chapters which are all linked together in SUPERSCRIPT format so you can search for individual words or sentences. Hundreds of hours have gone into preparing this in B-128 format. Pastors, teachers, and laymen will enjoy these great disks.

```
1 "nkjnt1          " ms 2c
12   "matt 1"          seq
14   "matt 2"          seq
9    "matt 3"          seq
13   "matt 4"          seq
24   "matt 5"          seq
18   "matt 6"          seq
13   "matt 7"          seq
17   "matt 8"          seq
19   "matt 9"          seq
20   "matt 10"         seq
15   "matt 11"         seq
25   "matt 12"         seq
30   "matt 13"         seq
16   "matt 14"         seq
18   "matt 15"         seq
15   "matt 16"         seq
14   "matt 17"         seq
19   "matt 18"         seq
16   "matt 19"         seq
17   "matt 20"         seq
25   "matt 21"         seq
19   "matt 22          seq
19   "matt 23"         seq
23   "matt 24"         seq
22   "matt 25"         seq
35   "matt 26"         seq
31   "matt 27"         seq
10   "matt 28"         seq
21   "mark 1"          seq
16   "mark 2"          seq
15   "mark 3"          seq
20   "mark 4"          seq
22   "mark 5"          seq
28   "mark 6"          seq
18   "mark 7"          seq
19   "mark 8"          seq
25   "mark 9"          seq
26   "mark 10"         seq
17   "mark 11"         seq
24   "mark 12"         seq
18   "mark 13"         seq
34   "mark 14"         seq
21   "mark 15"         seq
10   "mark 16"         seq
37   "luke 1"          seq
25   "luke 2"          seq
19   "luke 3"          seq
23   "luke 4"          seq
21   "luke 5"          seq
27   "luke 6"          seq
26   "luke 7"          seq
31   "luke 8"          seq
31   "luke 9"          seq
21   "luke 10"         seq
29   "luke 11"         seq
30   "luke 12"         seq
19   "luke 13"         seq
18   "luke 14"         seq
16   "luke 15"         seq
17   "luke 16"         seq
17   "luke 17"         seq
19   "luke 18"         seq
23   "luke 19"         seq
22   "luke 20"         seq
18   "luke 21"         seq
31   "luke 22"         seq
26   "luke 23"         seq
23   "luke 24"         seq
22   "john 1"          seq
12   "john 2"          seq
17   "john 3"          seq
24   "john 4"          seq
22   "john 5"          seq
32   "john 6"          seq
22   "john 7"          seq
28   "john 8"          seq
19   "john 9"          seq
18   "john 10"         seq
25   "john 11"         seq
24   "john 12"         seq
15   "john 13"         seq
15   "john 14"         seq
13   "john 15"         seq
17   "john 16"         seq
13   "john 17"         seq
22   "john 18"         seq
23   "john 19"         seq
17   "john 20"         seq
15   "john 21"         seq
16   "acts 1"          seq
24   "acts 2"          seq
14   "acts 3"          seq
20   "acts 4"          seq
22   "acts 5"          seq
9    "acts 6"          seq
31   "acts 7"          seq
20   "acts 8"          seq
23   "acts 9"          seq
24   "acts 10"         seq
3 blocks free.
-----directory disk two-----
1 "nkjnt2          " ms 2c
15   "acts 11"         seq
15   "acts 12"         seq
28   "acts 13"         seq
15   "acts 14"         seq
23   "acts 15"         seq
21   "acts 16"         seq
20   "acts 17"         seq
16   "acts 18"         seq
23   "acts 19"         seq
20   "acts 20"         seq
24   "acts 21"         seq
17   "acts 22"         seq
21   "acts 23"         seq
15   "acts 24"         seq
16   "acts 25"         seq
18   "acts 26"         seq
23   "acts 27"         seq
19   "acts 28"         seq
17   "romans 1"        seq
15   "romans 2"        seq
14   "romans 3"        seq
13   "romans 4"        seq
12   "romans 5"        seq
12   "romans 6"        seq
13   "romans 7"        seq
20   "romans 8"        seq
17   "romans 9"        seq
11   "romans 10"       seq
19   "romans 11"       seq
10   "romans 12"       seq
9    "romans 13"       seq
12   "romans 14"       seq
17   "romans 15"       seq
12   "romans 16"       seq
15   "1 cor 1"         seq
9    "1 cor 2"         seq
10   "1 cor 3"         seq
11   "1 cor 4"         seq
8    "1 cor 5"         seq
11   "1 cor 6"         seq
21   "1 cor 7"         seq
7    "1 cor 8"         seq
14   "1 cor 9"         seq
15   "1 cor 10"        seq
15   "1 cor 11"        seq
14   "1 cor 12"        seq
6    "1 cor 13"        seq
20   "1 cor 14"        seq
25   "1 cor 15"        seq
10   "1 cor 16"        seq
13   "2 cor 1"         seq
9    "2 cor 2"         seq
9    "2 cor 3"         seq
10   "2 cor 4"         seq
11   "2 cor 5"         seq
9    "2 cor 6"         seq
10   "2 cor 7"         seq
13   "2 cor 8"         seq
9    "2 cor 9"         seq
10   "2 cor 10"        seq
17   "2 cor 11"        seq
13   "2 cor 12"        seq
7    "2 cor 13"        seq
11   "gal 1"           seq
13   "gal 2"           seq
15   "gal 3"           seq
14   "gal 4"           seq
11   "gal 5"           seq
8    "gal 6"           seq
12   "eph 1"           seq
11   "eph 2"           seq
10   "eph 3"           seq
15   "eph 4"           seq
13   "eph 5"           seq
11   "eph 6"           seq
14   "phi 1"           seq
14   "phi 2"           seq
11   "phi 3"           seq
11   "phi 4"           seq
15   "col 1"           seq
12   "col 2"           seq
11   "col 3"           seq
9    "col 4"           seq
6    "1 thess 1"       seq
10   "1 thess 2"       seq
7    "1 thess 3"       seq
9    "1 thess 4"       seq
10   "1 thess 5"       seq
7    "2 thess 1"       seq
9    "2 thess 2"       seq
8    "2 thess 3"       seq
11   "1 tim 1"         seq
6    "1 tim 2"         seq
8    "1 tim 3"         seq
8    "1 tim 4"         seq
12   "1 tim 5"         seq
12   "1 tim 6"         seq
10   "2 tim 1"         seq
12   "2 tim 2"         seq
8    "2 tim 3"         seq
10   "2 tim 4"         seq
9    "titus 1"         seq
7    "titus 2"         seq
7    "titus 3"         seq
10   "philemon"        seq
9    "heb 1"           seq
11   "heb 2"           seq
9    "heb 3"           seq
9    "heb 4"           seq
8    "heb 5"           seq
10   "heb 6"           seq
15   "heb 7"           seq
9    "heb 8"           seq
16   "heb 9"           seq
19   "heb 10"          seq
21   "heb 11"          seq
16   "heb 12"          seq
12   "heb 13"          seq
12   "james 1"         seq
12   "james 2"         seq
9    "james 3"         seq
9    "james 4"         seq
11   "james 5"         seq
15   "1 peter 1"       seq
14   "1 peter 2"       seq
13   "1 peter 3"       seq
11   "1 peter 4"       seq
7    "1 peter 5"       seq
11   "2 peter 1"       seq
14   "2 peter 2"       seq
12   "2 peter 3"       seq
6    "1 john 1"        seq
16   "1 john 2"        seq
12   "1 john 3"        seq
10   "1 john 4"        seq
11   "1 john 5"        seq
7    "2 john"          seq
7    "3 john"          seq
16   "jude"            seq
13   "rev 1"           seq
18   "rev 2"           seq
14   "rev 3"           seq
8    "rev 4"           seq
10   "rev 5"           seq
11   "rev 6"           seq
12   "rev 7"           seq
8    "rev 8"           seq
12   "rev 9"           seq
8    "rev 10"          seq
13   "rev 11"          seq
11   "rev 12"          seq
11   "rev 13"          seq
14   "rev 14"          seq
6    "rev 15"          seq
13   "rev 16"          seq
12   "rev 17"          seq
16   "rev 18"          seq
14   "rev 19"          seq
11   "rev 20"          seq
16   "rev 21"          seq
12   "rev 22"          seq
12   "INSTRUCTIONS"    seq
18 blocks free.
```

SERMONS 2 CBUG #64 NEW RELEASE #11714

SERMONS TWO is a compiled disk of sermons by Pastor Mark Schwarzbauer, the author of SERMONS ONE and Senior Pastor of

Family Worship in Sturgeon Bay Wisconsin. All the files all linked together so you can run searches and more.

```
1 "sermons 2       " ms 2c     26  "the cross 13+"     seq   17  "laboring+"          seq   22  "miracles 6+"        seq
15   "cross 7"          seq     16  "the cross pt14+"   seq   21  "the cross 22+"      seq   31  "revelation 5+"      seq
25   "prayer 7"         seq     11  "gifts ho2"         seq    9  "dating"            seq   17  "miracles 7+"        seq
23   "show your faith+" seq     18  "gifts pt3"         seq    9  "dating ho"         seq   18  "revelation 6+"      seq
21   "share your faith" seq     17  "the cross 15+"     seq   21  "three witnesses"    seq   23  "relying on God+"    seq
12   "cross 8+"         seq      9  "gifts ho3"         seq   17  "the cross 23+"      seq   20  "building for God"   seq
13   "cross 9+"         seq     23  "gifts pt4"         seq   15  "superspell.tm"      seq   26  "miracles 8+"        seq
21   "his cause"        seq     26  "what is man"       seq   17  "the cross 24+"      seq   24  "independence+"      seq
6    "cross 10+"        seq     11  "gifts ho4"         seq    7  "hormones"          seq   20  "miracles 9+"        seq
12   "cross 11+"        seq     18  "the cross 16+"     seq   19  "hope"              seq   28  "missions+"          seq
12   "syf thru vision"  seq     24  "what's happening"  seq   21  "resurrection"      seq   18  "miracles 10+"       seq
25   "syf by knowing i" seq     20  "the cross 17+"     seq   18  "heaven"            seq   39  "miracles 11+"       seq
19   "thanks alot"      seq     31  "worship+"          seq   26  "hell+"             seq   27  "missions cont+"     seq
24   "the cross 12+"    seq     30  "trials"           seq   30  "miracles intro+"    seq   36  "harvest time+"      seq
14   "let's be thankfu" seq      4  "sing"             seq   17  "miracles pt2"      seq   30  "miracles 12+"       seq
12   "work work work+"  seq     10  "worship ho"       seq   22  "hurdles to heave"   seq   25  "lift him up+"       seq
18   "the tribulation+" seq     37  "the cross 18+"     seq   21  "revelation pt1"    seq   26  "excavation faith"   seq
19   "the lord's retur" seq     19  "body building"     seq   14  "prayer"           seq   28  "miracles 13+"       seq
31   "the last days"    seq     21  "temptation"        seq   20  "miracles 3+"      seq   26  "God's not finish"   seq
5    "radio messages"   seq     20  "the cross 19+"     seq   22  "revelation pt2"    seq   23  "faith+"             seq
21   "prepared for 198" seq     19  "the cross 20+"     seq   20  "miracles pt4+"    seq   20  "miracles 14+"       seq
15   "gifts pt1"        seq     29  "stare down+"       seq   21  "revelation 3+"    seq    8  "faith ho"          seq
14   "gifts pt2"        seq     18  "the cross 21+"     seq   20  "miracles 5+"      seq  215 blocks free.
7    "gifts ho1"        seq     20  "because he lives"  seq   17  "revelation 4+"    seq
```

SERMONS 3 CBUG #65 NEW RELEASE #11728

SERMONS THREE is a compiled disk of sermons by Pastor Mark Schwarzbauer, the author of SERMONS ONE and Senior Pastor of Family Worship in Sturgeon Bay Wisconsin. All the files all linked together so you can run searches and more.

```
1 "sermons 3       " ac 2c     26  "miracles 23+"      seq   22  "don't step+"        seq    8  "harvesting"        seq
14   "miracles 15+"     seq     32  "miracles 24+"      seq   26  "miracles 32+"       seq   36  "unequally yoked+"  seq
3    "NOTE TO USER"     seq     35  "miracles 25+"      seq   26  "stumbling+"         seq   25  "faith"             seq
28   "miracles 16+"     seq     28  "desert 2+"         seq   24  "press towards+"     seq   15  "confidence in..."  seq
22   "God cares+"       seq     31  "preparing+"        seq   18  "miracles 33+"       seq    5  "expectation"       seq
33   "God cares2+"      seq     21  "what child+"       seq   20  "commitment+"        seq   21  "state of church+"  seq
25   "miracles 17+"     seq     30  "miracles 26+"      seq   19  "miracles 34+"       seq   15  "memorial day"      seq
25   "God cares 3+"     seq     25  "begin again+"      seq   24  "miracles 35+"       seq    6  "sound off+"        seq
32   "miracles 18+"     seq     32  "miracles 27+"      seq   22  "don't trip+"        seq   22  "retaining+"        seq
19   "breaking thru+"   seq     21  "prepared for 87"   seq   21  "miracles 36+"       seq   26  "compelled+"        seq
21   "God cares4+"      seq     38  "miracles 28+"      seq   25  "the prize+"         seq   24  "restoration"       seq
24   "miracles 19+"     seq     28  "born again+"       seq   12  "nature of God+"     seq   21  "restoration 2+"    seq
33   "miracles 20+"     seq     28  "source+"           seq   19  "endurance+"         seq    5  "communion"         seq
22   "attacks+"         seq     31  "miracles 29+"      seq   16  "nature of man+"     seq    7  "anointing+"        seq
17   "victory on pm+"   seq     15  "quart low+"        seq   21  "gods are judged"    seq   31  "carry the cross+"  seq
32   "oxymoron+"        seq     24  "his voice+"        seq   28  "easter authority"   seq   27  "perfect father+"  seq
22   "miracles 21+"     seq     10  "gifts"            seq   25  "easter power"       seq   21  "restoration 3+"    seq
26   "yielding+"        seq     25  "miracles 30+"      seq   25  "inheritance+"       seq   37  "God is for me+"    seq
23   "miracles 22+"     seq     13  "touching"         seq   13  "good friday+"       seq   28  "high cost+"        seq
32   "desert+"          seq     32  "miracles 31+"      seq   20  "family+"           seq   39 blocks free.
```

PRECISION CHURCH (A SB Applicatioin) CBUG #71 NEW RELEASE #11593

by Mark Schwarzbauer

PRECISON CHURCH 700 is a multi-use church program that also lends itself to other non-profit organizations and groups. It contains some highly refined special programs such as the contributions program that keep track of peoples giving and has complete mailmerge recall of these files as well as other print out programs. In addition to the contributions program there are other programs that keep track of birthdays, and anniversaries, and print out selected lists, there is a membership program that keeps track of various types of membership as well as printing out selected reports. PRECISION CHURCH 700 comes with 14 pages of instructions and can be immediately put to work even without knowledge of SUPERBASE II as all the programs come with user friendly drop down menus where you just select what you want. This program is designed for use with SUPEROFFICE which contains SUPERBASE II. While you can use SUPERBASE II itself, you will be able to better utilize the full benefits of PRECISION CHURCH 700 if you use it with SUPEROFFICE. This series is copyrighted by the author and may not be reproduced without permission except for backup for personal use.

```
0 "precision church" ms 2c     1   "giving list.p"     seq    9  "membership.p"      seq    3  "recheck.p"         seq
2    "churches"         seq     10  "giving.p"          seq    3  "order"             seq    5  "reports.p"         seq
3    "days"             seq     79  "instructions"      seq    3  "quarter 1 ms"      seq   10  "start.p"           seq
```

WANT ADS

RATES: $10.00 per nominal line of 120 characters. CBUG reserves the right to alter printing layouts and spacing.

1.) Std. B128 System, SSII + much software in orig. pkg. Excellent condition. $500/bst offer. Edgar Peara, 214 5th Wilmette, Il. 60091, 312-256-4642

2.) B128, 8050, 4023, Amber Mon, Software. $400 Tom Taylor, Box 1176, Laurie, Mo. 65038, 314 372 2248.

3.) DOS 2.7 chip set for 8050 drives $35, IEEE/Epson interface $35, IEEE cables $15 and many more B128 goodies. Mark Hughes, 317 356 8436 after 6pm.

4.) B128 w/ fan, SFD 1001, 4023, 8023P, IEEE/Cent interface, internal IEEE-Cent interface, SS II, SS III, Knight's. $650. Lowell Nissen, 501-267-3608.

5.) B128 Computer + Hardwaare & Software. Any reasonable offer + shiping. Wm. Bender, 201 832 2933

6.) B128, 8050, 4023, SS II, SB, Knight's & more. $600 plus shipping. Paul Hennes, 206-282-1471.

7.) B128, 8050, SS, SB plus 6 other program disks. Will split. $400. L Strout 4917 Banewell Ave., Covina Ca. 91722, 818-332-8623.

8.) Standard B128 system, modem, software. Excellent condition. $650, shipping included. Nick Dittiger, 1961 Cedarwood Drive, Melbourne, Fl. 32935, 305 259 0687

9.) B128, 4023, 8050, SS, SB. R. Alexander, 806-835-2997 Evenings. P.O. Box 326, Lefors, Tx 79054.

10.) B128, 8050, mint, condition, cables, CBUG Utilities, inst. booklets, Superscript, Superbase, $400 plus shipping. H&G Engineering, Stockton Ca. 209 931 3815, 8am to 5pm.

11.) Complete Protecto B128 package, assorted business software, cabling, etc. $600. Roger Wehner, 321 Lake Ave. Fairmont, Mn. 56031. 507 235 5481 after 5pm.

12.) Complete B128 system. $700.00. Doug Freeman, 919-455-1756

13.) Complete Protecto B-128 system in original packaging. $500 plus shipping. Ken Schultz. 313-429-4792 eves, 313-231-2727 days.

14.) Standard B-128 low profile $65, B-128 low profile ready for 256K, Zenith monochrome monitor $65, Commodore 8023P Dot Matrix printer $100. Commodore 8050 Dual Disk Drive $350 -- Note I will not sell the Disk drive by itself. Pet 2001 calc. Keyboard 8K pc $50. Total for all $705.00; Discount if purchased as a package $-105.00; Net package total $600.00. Larry Cooper, 312-490-9431

15.) 2 B128 computers, 2 8050 disk drives, 2 USI P3 monitors, 1 4023 printer. Wayne Kenyon, City of Ionia Mi., 114 N. Kidd St., Ionia, Mi. 48846. 616-527-0370.

16.) 3 B-128 computers, 3 8050 drives, 2 4023 printers, 2 amber monitors, all cables, SS, SB, CABS <GL, AR, AP, PR, INV, POS>, Knights & 4 other utility disks. Will sell all or part. Ed Repic, 714-520-0778

17.) B128 w/ board mounted 256K $80, 8050 disk drive w/recent alignment receipt $275, 4040 dual disk drive $125, Pet to IEEE cable $25 (free if a complete system is purchased). Original copies SS & SB w/ original documentation $5. Original manuals and boxes provided with B128 and 8050. Gene Lambert, 322 S. Clark, Orange, Ca. 92668, 714-771-3593 anytime (answering machine).

18.) B-128 low profile w/ 256K $100, Orange swivel monitor & cable $75, Green monitor w/ cable $60, 5 SS II packages $6.50ea, 4 SB packages $6.50ea. Warren Swan, 1 N. 114 Woods Ave, Wheaton, Il., 60188, 312-665-1514.

RIBBONS - MEMORY KITS - FANS - POWER SUPPLIES

			12+	1-11
CBM	4023	Multistrike.....ea.	$5.50	$6.00
CBM	8023	Nylon Fabric.....ea.	$5.00	$5.50
CBM	1525	Nylon Fabric.....ea.	$5.75	$6.25
CBM	1526	Multistrike (Carbon).....ea.	$5.50	$6.00
CBM	MP801	Nylon Fabric.....ea.	$6.50	$7.15
CBM	PET	Nylon Spool.....ea.	$2.90	$3.20
CBM	PET	Multistrike.....ea.	$4.05	$4.40
CBM	6400	Nylon Fabric.....ea.	$4.45	$4.95
CBM	6400	Multistrike.....ea.	$4.45	$4.95

Minimum Postage: $2.00 plus $0.25 each ribbon
Please give Street Address for UPS shipments -- Foreign Add 35%

"IF IT PRINTS, I HAVE IT."

Memory Expansion Kits (each)	($3.00 UPS)	$75.00
Memory Upgrade (In Shop)		$100.00
Fan Kits	($5.00 UPS)	$25.00
B128-80 Power Supplies	($5.00 UPS)	$50.00
SuperPet	($25.00UPS)	$375.00

Will accept COD's

S & W Supply Company
Jim L. White
5308 Timberline Trail
Rapid City, South Dakota 57702
(605) 348-3696

***

CompuServe #70007,1147
[Prices Subject to change without notice.]

C-64™ • VIC™ • SX-64™ • C-128™ • Plus 4™ • C-16™ • B-128™ • PET™ • CBM™ • LCD™

One disk, 25 business programs, $19.95

The **Intelligent Software Package** is the one product for your Commodore that can take care of all your data processing needs.
Customers write: *". . . accolades for the authors. This is as slick a deal as I have seen and more than adequate for all except fancy presentations. The best thing is the ease of use . . "*
"I have come to consider these programs among the most valuable pieces of software I own."
There are no hidden fees for shipping or documentation, and no clubs to join. The package is not public domain software, and is sold only direct to customers by mail: it supports all available printers, and will run on any Commodore computer (except Amiga) with a minimum of 10k RAM, including the C-128 in C-128 mode.
What you get when you order the Package:

Database—A complete database manager. All fields completely user-definable. Can be used for any number of tasks, including accounting, checkbook and tax records, mailing lists, inventory control, catalog maintenance, or as an electronic rolodex. A customer writes: *"I am especially impressed with Database, and have used it to replace a half-dozen other 'database'-type programs I had been using."*
Word Processor—A full-featured menu-driven word processor. Allows full control over margins, spacing, paging, indentation, and justification. *"Highly recommended."*—Midnite Software Gazette. *"Provides good basic features."*—Compute's Gazette.
Copycalc—An electronic spreadsheet. *"Excellent program for budgeting, estimating, or any math-oriented use . . . well worth the money. Highly recommended."*—Midnite Software Gazette.
ReportGen—creates form letters, mailing lables, etc.
ReportMerge—creates statements invoices.
Baseball Stats—compiles team batting statistics.
Index—indexes W/P's text files.
Wordcount—counts words in a text file.
WPConvert—converts files to other WP formats.
DBMerge—facilitates relational D/B applications.
DBStat, DBStat2—analyze D/B files.
ASCII—converts text files into program files.
Checkbook—reconciles checkbook.
Inventory—Maintains inventory records.
Paper Route—A/R for paper route.
Loan Analysis—computes finance terms, prints schedules.
Breakeven—computes breakeven analysis.
Depreciation—creates depreciation schedules.
Labeler—creates labels.
File Copier—copies sequential, program files.
Correlation—calculates statistical correlation.
Also other Database and Word Processor utilities.

To order, send name, address, and $19.95 to address below. Please specify regular [1541/1571/2040/4040/2031] disk, 8050 disk, or cassette [cassette not available for Plus 4 or C-16]. Add $3 for credit card or COD orders; Calif. residents add 6%. No personal checks from outside USA. A sampling of program output is available for $1. **Tear this ad out and keep it handy!**

Intelligent Software
Quality Software since 1982

Box A Dept. M-9
San Anselmo, CA 94960
(415) 457-6153

SERVICE CENTER:

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

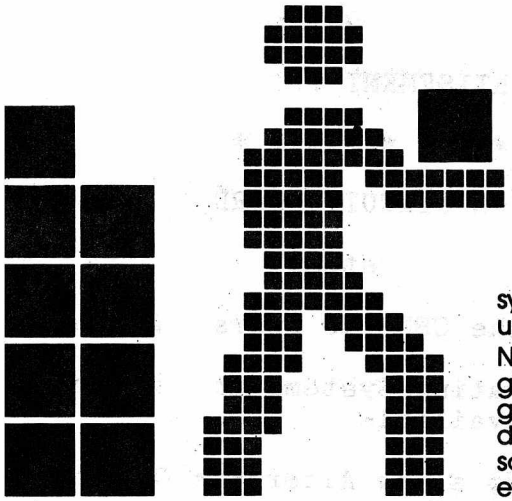
Authorized Commodore service center, we repair B128, 8032, 4023, 8023, 6400, 4040, 8050, 8250, 2031, C-64, C-128, etc. Rates are $50/Hr plus parts. We will diagnose & quote repair cost for $25. Normal turn-around time is 5 business days. We ship via UPS. Limited supply 8032's & 4023's (new) at $189 each.

B-128 FUNCTION KEY OVERLAYS - CLOSEOUT SALE

Keep track of your custom programmed function keys. Blank overlays, 140# stock ruled on both sides to provide 2 sets of functions for each overlay. First Class postage paid. 5 for $2.00 - 10 for $3.50. Iowa residents add 4% tax. Cash, check, or money order to:
OFFICE ASSISTANCE - 4218 PINE VIEW DR. N.E. - CEDAR RAPIDS, IA 52402.
Pre-printed overlays for quantity users available. Advise details for quote.

INCOME TAX PROGRAMS FOR 1987 TAX YEAR ON B128/8050

They will be ready for shipment soon after IRS forms are available for incorporating changes (Dec or early Jan.) Standard PROGRAM has 1040, A, B, C, F, and 4797 for only $65. Order custom program for $20 PLUS $5 per page. A CUSTOM PROGRAM for CPA's at same price may contain as many pages as desired with 1 MEG expansion. Programs operate within CALC RESULT ONLY. Calculations and transfers on forms is automatic. Start up HELP SHEET with HELP SCREENS cover your needs. Form PRINTOUT (your records) instructions included. Paid call support. Jim O'Halloran, Rt 2 Owl Creek Rd., Hiawassee, Ga. 30546.



NWM's INVENTORY CONTROL SYSTEM*

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

Now it's even better!

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

- **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.**

Only a $10 upgrade charge for owners of previous versions of NWM Inventory.

WE SUPPORT OUR CUSTOMERS!

And listen to our prices!

B Version 1 8050 ..... $49.95 **C-128 Version 1 1571 ..... $49.95**
B Version 2 8050 ..... $49.95 **B-128 Version 1&2 8050... $54.95**

U.S. shipping and handling charge $3.95

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

(312) 520-2540

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

*Requires use of Superbase®

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

NEW 128K USER INSTALLABLE MEMORY EXPANSION! INTRODUCTORY PRICE OF ONLY $125.

NWM Inc. has contracted to have a user installable 128k memory expansion board designed. This non-destructive upgrade will turn a B-128 puppy into a full grown 256k monster machine! The board will install on the 40 pin co-processor and 60 pin expansion bus.

This will give you the capability to run the following programs.

- SuperOffice
- Superscript w/3 banks.
- Calc Result w/12,047 cells
- 8432 emulator w/4 banks.
- CP/M 86 extended programs*
- Word & Calc Result integ.

If you were afraid to send your machine away or did not have the ability to self install the upgrade, then this is for you. This is your opportunity to enjoy expanded memory and what other B-128 users have found it can do for you!

*W/8088 co-processor

Shipping and handling charges $3.95

NORTHWEST MUSIC CENTER INC.

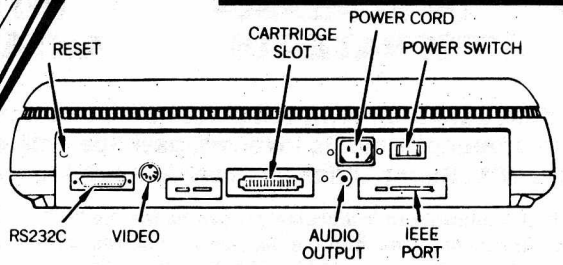
539 N. Wolf Rd. — Wheeling, IL 60090
Hours — Phone (312) 520-2540
Mon.-Thurs. 12:30-8:30, Sat. 9:00-4:00

64K IEEE Print Buffer
Only $199

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

Features

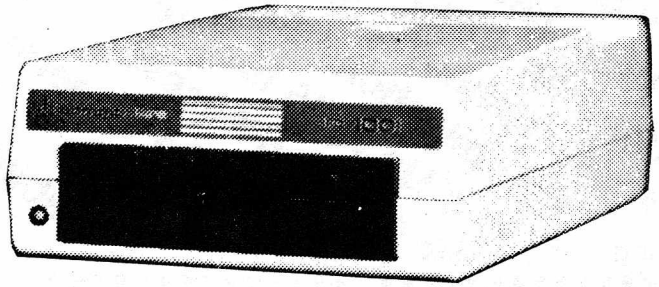
- Frees your computer for other tasks
- Stores almost 30 minutes of printing
- Helps make you and your employees more productive
- Copy and clear buttons
- IEEE and parallel Centronix output



B-128 Lo Profile

We have just purchased the last of the lo and hi profile B-128 computers from Commodore. We have both new and used models of both of these computers. If you would like to have a backup computer for your system then this is it! When these are gone there are no more! At this time it is certainly very prudent to have a backup computer. If your computer goes down and has to be repaired, you will certainly feel more secure knowing you have a replacement if it takes time to find parts for the repair!

PRICED AT **$145** US
ADD $10.95 SHIPPING US



SFD 1001 1 Megabyte Drive
double sided 8250 format
IEEE interface

PRICED AT **$169.95** US
ADD $9.95 SHIPPING US

COMMODORE
8000-9000 SOFTWARE & MISC.

9000 Superpet	$279.95
64K exp for 8032	$125
Pet Switch	$199
Pet Daughters	$105
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	$149
Calc Result 8032	$89

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 16.90 for 8023p, $25 Superpet, 9.95 SFD 1001, 10.95 B-128. 9.95 4023p, 15.95 9090 and 4.95 64K memory expansion. For software add $3.00 for first and $1.50 for each additional book or program. Canadian shipping charges are double U.S. International orders, call for shipping. For C.O.D. orders add 1.90 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.

International Orders Call For Shipping Charges

NORTHWEST MUSIC CENTER INC.

539 N. Wolf Rd. — Wheeling, IL 60090
Hours — Phone (312) 520-2540
Mon.-Thurs. 12:30-8:30, Sat. 9:00-4:00

CBM 256-80
w/8088 Co-processor

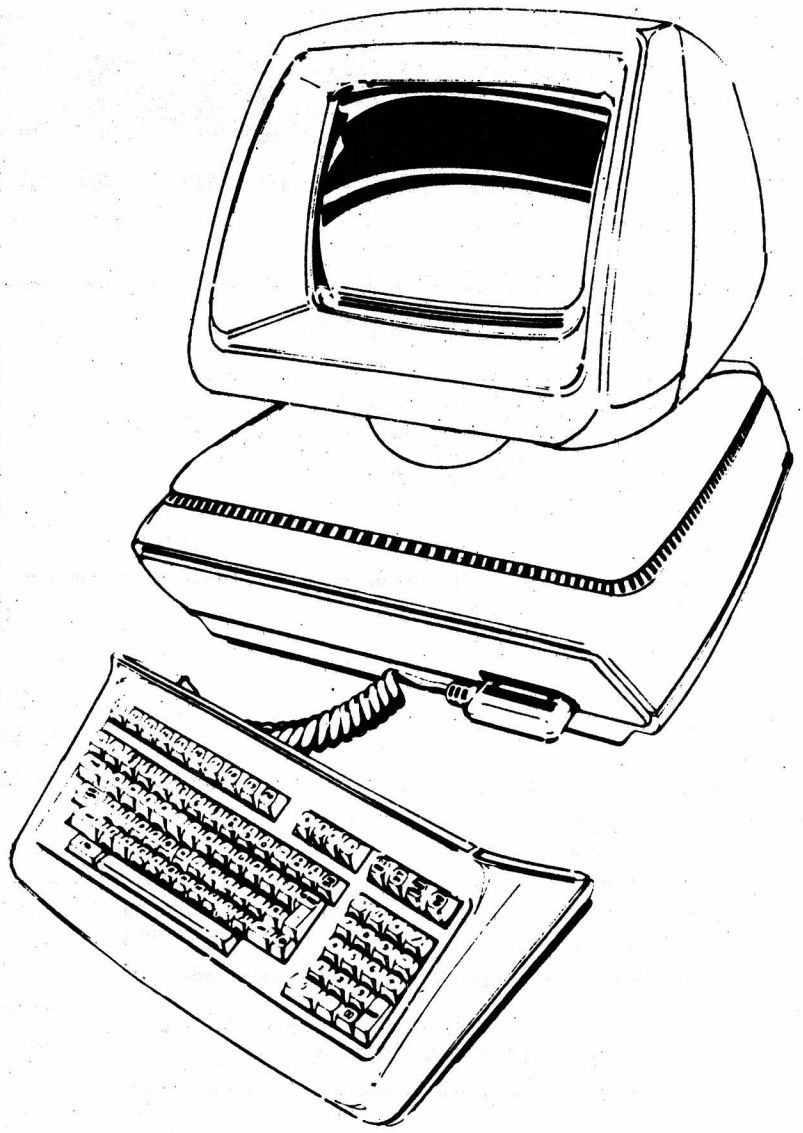
Call for Details
Very Limited Quantities

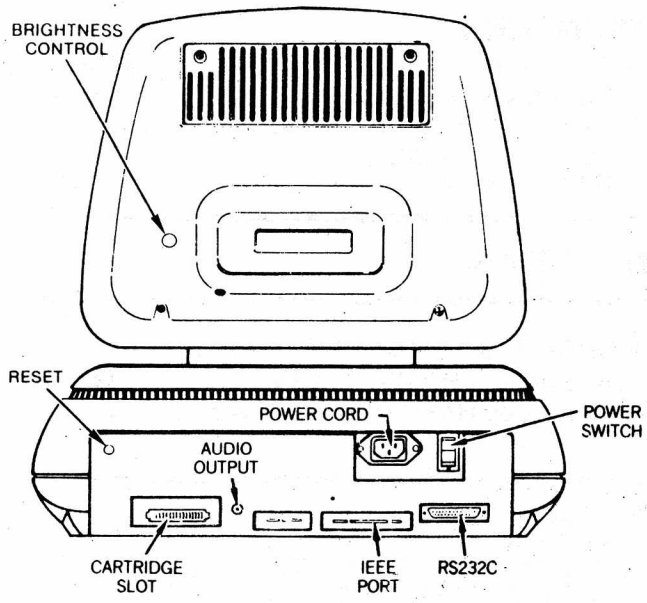
We are offering a limited number of used CBMX 256-80 computers for only $400. These come installed and tested with the original 8088 co-processor boards that until recently were hidden in the Commodore research labs.

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 $150 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.





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 $225 to $400 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 — Phone (312) 520-2540
Mon.-Thurs. 12:30-8:30, Sat. 9:00-4:00

Hacker's Corner

One of a Kind • Surplus • Monthly Special • Closeouts

Limited quantities to stock on hand

PRECISION SOFTWARE

Superbase	$ 9.95
Superscript	$19.95

HANDIC SOFTWARE

Calc Result	$89.95
Word Result	$59.95

C.A.B.S. ACCOUNTING

General Ledger	$ 9.95
Accounts Receivable	$ 9.95
Accounts Payable	$ 9.95
Order Entry	$ 9.95
Payroll	$ 9.95
Buy 5 for only	$24.95

B-128 GOODIES

B-128 Programmable Reference Guide was $29.95	$ 5.95
P-I	$24.95
I-I	$29.95
SFD 1001 1 meg drive	$169.95
Superbase The Book	$14.95
Superpet 9000	$279.95
Pet Switch	$199.00
Pet daughters	$105.00
8050 rehabs from	$250-$400

CALL FOR INFORMATION ON 8000 & 9000 SERIES SOFTWARE AND MISC.

SUPER DISK DOC

ONLY $24.95 WHILE SUPPLIES LAST!

CBM PROFESSIONAL USERS GUIDE

This is a reference manual on most of CBM'S Business Machines.

An excellent source for looking up useful information not found in most of the standard doc we have seen with most computers! ONLY $17.95 WHILE SUPPLIES LAST!

PRINTERS

8023p 150 cps	$179.95
4023p 100 cps rehab	$99.00
Gemini 15 x 120 cps	$159.95
Smith Corona DM200 RS232 and parallel	$179.95

ONE OF A KIND ITEMS

MJ-10 COLOR MONITOR	$125.00
AVATEX 1200 MODEM	$89.95
INTERFACE SERIAL to IEEE	$49.95
INTERFACE IEEE to PARALLEL	$89.95
MONOCHROME MONITOR	$79.95
9060 5 meg HARD DRIVE, NEW	$495.00
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 16.90 for 8023p, $25 Superpet, 9.95 SFD 1001, 10.95 B-128. 9.95 4023p, 15.95 9090 and 4.95 64K memory expansion. For software add $3.00 for first and $1.50 for each additional book or program. Canadian shipping charges are double U.S. International orders, call for shipping. For C.O.D. orders add 1.90 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 — Phone (312) 520-2540

Mon.-Thurs. 12:30-8:30, Sat. 9:00-4:00

A NEW DATABASE FOR THE B-128

Mike Konshak's dfile DBMS for C-128 is now available for the B-128

Readers of RUN magazine will know Mike as a regular contributor, especially for the "Datafile" series of programs published in several installments beginning in November, 1984. In the nearly two years since then, it has evolved from a comparatively simple program to a full featured database for the C-128 computer. Now, with appropriate changes, it is released for the B-128 with a licensing agreement to Chicago B128 User's Group - International. Distribution initially is limited to the U.S. only.

The newest version, now sold commercially, is "dfile128". This program is the result of input from thousands of users of Mike's earlier programs. It is highly recommended in the Midnite Software Gazette.

Being memory-based, rather than keeping all the files on disk for processing, operation of this database is much faster than Superbase and similar databases. All processing takes place in memory, without waiting for disk access for each record. When entries or changes are completed, data is stored on disk in sequential files. This speed does have a trade-off: the amount of data that can be in memory is limited by memory size. However, most folks don't have an enormous number of records, and do desire speed and ease of use in preference to capacity and complexity. dfile, as an example, will allow an inventory with 8 columns to have 740 records. If that isn't enough, a program awaiting conversion for later issue can combine up to 30 compatible files into one large record file for printing reports, labels, etc.

dfile DBMS main menu options:

Create new file	Add record to current file
Print records	Modify record in current file
View file on screen	Delete record in current file
Sort records by field	Read old file from disk
Disk commands	Write new file to disk
Print records	

dfile for B-128 includes provision for IEEE interfaced printers as well as RS-232-C printers at both 300 and 1200 bps. Escape codes can be sent to printers to control modes and styles. Output can be unformatted, report format (set up as desired), mailing labels, or calculated reports that offer column totalling, averaging, and calculated fields such as in a spreadsheet.

The complete instruction manual is included on the disk. A commercially published booklet (for the C-128 version) is available from Mike Konshak for $5. Printer options are the most noticeable difference in manuals.

A companion disk of utility programs is expected to be offered. These programs allow manipulation and modification of your record files. They were created because they were requested by users of dfile128.

One program will merge files for those requiring files larger than can be stored in memory. It can merge up to 30 compatible record files into one huge record file for printing with a print file utility.

Clone file will clone or copy the data from one record file into a new record file with a totally different structure. You can even combine the da a from several old fields into one of the new. The ability to change the structure as the need arises, without re-entering all the data, makes expansion or modification much more feasible.

Xport file will strip all non-essential structure data found in a dfile128 record file and leave just the records so that the files can be transferred to other databases or wordprocessors for form letters.

Conversion effort applied to these already existing utilities will be proportional to CBUG member reception of dfile.

ORDER FROM THE CBUG LIBRARY, CBUG #47, stock #11856

THE PETSPEED USER GUIDE
for THE B-128 !!

The Petspeed Basic Compiler for the B-128 has been a long time in coming, it's power and ease of use is legendary, but it suffered from the lack of a deserving and credible user manual. The Petspeed User Guide changes all that. Written by the most published and referenced Petspeed author, the 1st edition was originally produced for the 8000 and c64 series of Petspeed. The 2nd edition B-128 version carries over the full introduction and Petspeed facts section, full breakdown and discussion of Petspeed commands and procedures and full details on how and where Petspeed stores variables and other components of the compiler program. It includes many usable programs designed for Petspeed's abilities. A must have for anyone who is serious about using petspeed!

67pp, 8x11 spiral bound, isbn 0-931259-02-9, $19.95 + $2.10 United Parcel
Check or money order only.
CSM, 4734 East 26th Street, Tucson Az 85711. info only: (602) 790-6333 (9am-8pm mst)

◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆

	NEW	UPGRADE EXCHANGE
SUPERSCRIPT III	$60.00	$45.00
SUPERBASE II	60.00	25.00

Ask for Steve Spring -- the regular order takers are not briefed on B128 products

PROGRESSIVE PERIPHERALS & SOFTWARE
464 Kalamath Street
Denver, Colorado 80204
303/825-4144

◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆

A HIGH-RESOLUTION BOARD is available FOR THE B128

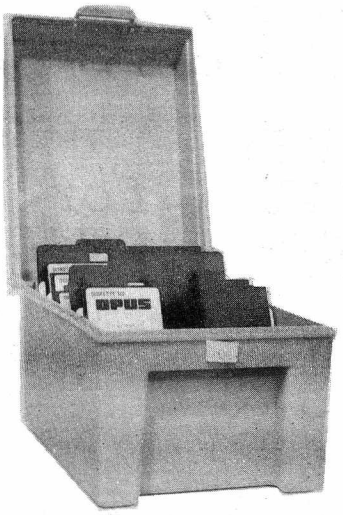
The board allows bit-mapped graphics. The HR picures can coexist with the normal text screen. The resolution is 1024 across by 512 down; to that area the B's screen is a window. Any size window for HR grphics can be created and it can be moved independently of the text screen. The boaard includes 64K of RAM to support the bit graphicis. Installation is simple: a six by five inch board goes inside the B, and one line needs to be cut. The board was designed in cooperation with the people who are building 1meg expansions, in order to be compatible with that project.

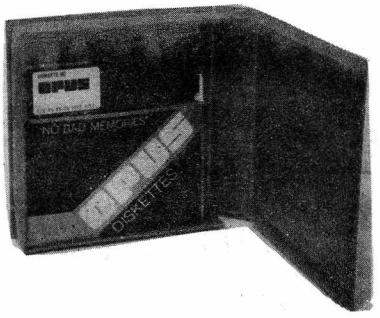
Software is in Basic at the present timem. Machine code routines will be developed as an independent project, as soon as possible.

HI RES TECHNOLOGIES
16 English Ivy Way
Toronto, Ont. CANADA M2H 3M4
The price is $199.00 U.S.

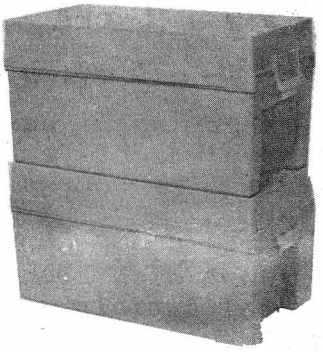
DATA CASE ORDER FORM

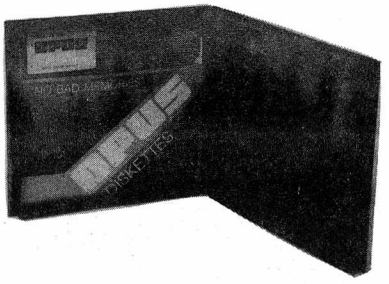
DD120L Holds 120 5¼ Disks
Locking Clear Top, Beige Body
10 Dividers w/ Labels
Order #10960





Library Shelf Box, Holds 10 5¼ Disks
Living Hinge Design. Pastel colors
- we'll ship all same in an order
DD10 Order #10989

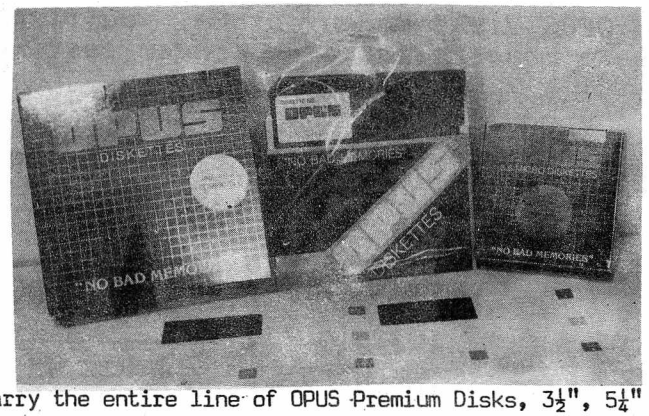




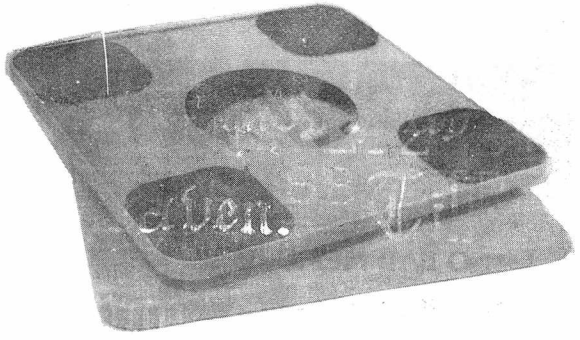
DD5 Mailer/Tote Box, Holds 5 5¼ Disks
Living Hinge Design. Pastel colors
- we'll ship all same in an order
Order #10941

DS100L Holds 100 5¼ Disks
Locking Clear Top, Beige Body
Useable Hinged or Lid Nested Underneath
8 Dividers w/ Labels Order #10975

Heavy Duty, Compact, Stackable, Non-Breakable.
One Piece Living Hinge Design. Six Dividers
Factory Installed. Beige body and cover.
D55 Holds 60 5¼ Disks Order #10994
D33 Holds 40 3½ Disks Order #10918



NOTE: Due to shipping and Packaging costs, our data case products are not available outside N. America!



Extra Heavy Duty Monitor Swivel Stand
Holds Monitors up to 12" on Center Feet
Can be used locked in positon or Adjustable
Beige MS1212 Order #10937

We carry the entire line of OPUS Premium Disks, 3½", 5¼" & 8" Single and Double sided, Single & Double Density; PLUS, quad and High Density AT compatable 5¼ disks.

Item	Description	Quanty	Price	Extension
10960	DD120L locking 5¼ case	_____	$16.00	_____
10975	DS100L locking 5¼ case	_____	12.50	_____
10922	DD50L same as 10975; holds 50	_____	8.00	_____
10994	D55 stackable 5¼ case	_____	6.45	_____
10918	D33 stackable 3½ case	_____	5.00	_____
10989	DD10 5¼ library case	_____	1.50	_____
10641	DD5 5¼ mailer/tote case	_____	1.00	_____
10937	MS1212 Monitor Stand	_____	16.00	_____

Name _____ Zip Code _____ TOTAL _____
Prices and specifications subject to change without notice.

DISK SALE

5.25"	Each	SAVE!		Each	SAVE!
SSDD	$.57	34%	for IBM AT	$1.70	43%
DSDD	.60	35%	DS4D	1.58	15%
			flippies	.61	27%
3.5"			8"		
SSDD	1.50	37%	SSDD	2.60	19%
DSDD	1.70	48%	DSDD	2.80	21%

OPUS QUALITY DISKS!

OPUS disks are without doubt the finest disks made. Mil spec forumulas and exotic materials, promise above spec performance even after 5,000,000 revolutions on each track! Heavy weight jacket and lubricated antistatic liner insure the ultimate in stability. Of course there is a 100% factory warranty backed by CBUG as well.

We are proud of our product. Each disk bears the OPUS corner label; you can be confident you are buying the real article -- no misrepresented no-name seconds. Each disk is inserted in its sleeve; labels + write protect tabs factory sealed in each package. SSDD and DSDD products come 10 in a poly bag; flippies are 25 in a bag. All others are 10 disks in a sealed retail chipboard or plastic package.

This is a special offer by CBUG, Inc. (The Chicago B128 User's Group), one of the world's largest user's groups. Our volume exceeds that of most computer stores and retailers. We pass our good fortune on to you, our members. If you can find OPUS disks in any store, you'll pay several times our price! AND, no one makes a more reliable or longer lasting disk!

----------Oct. 15, 1987----------

Desc	Qty	Stock#	/Pkg	Extension
5.25"				
SSDD	. .	10017	$ 5.70	$_____
DSDD	. .	10021	6.00	_____
DSDD Flippy	. .	10111	15.25	_____
DSQD	. .	10182	15.80	_____
For IBM AT	. .	10228	17.00	_____
3.5"				
SSDD	. .	10074	15.00	_____
DSDD	. .	10088	17.00	_____
8"				
SSDD	. .	10509	26.00	_____
DSDD	. .	10547	28.00	_____
Pkg 25 Tyvek sleeves	. .	10318	1.00	_____
TOTAL from other side this form				_____
TOTAL ORDER				_____
Ill. Residents add 7% sales tax				_____
COPY TOTAL TO MAIN ORDER FORM				$_____

PLEASE ENTER NAME & ZIP BELOW
IN CASE SEPARATED FROM MAIN ORDER

Name _____
Zip Code _____

OPUS®

"NO BAD MEMORIES"

THRU CBUG, PAY THE LEAST!

```
7   "days.p"          seq   7   "labels.p"        seq   3   "quarter 2 ms"    seq   3   "total.p"          seq
1   "depart"          seq   4   "mailmerge"       seq   4   "quarter 3 ms"    seq   1   "CHURCH"           seq
4   "giving"          seq   2   "membership"      seq   4   "quarter 4 ms"    seq   1867 blocks free.
```

HIS WORK — A SET OF NEW RELEASES — #11547

CBUG is back in the Baker's half dozen tradition again -- the above 5 CBUG releases (6 disks) are offered as a package thru December 25, 1987 at about 1/3 off their individual prices. A truly miraculous value!

THE NEW 8432 EMULATOR v.g & MORE CBUG #66 NEW RELEASE #11733

--PLUS: Goceliak, O'Halloran, Chick

This is the first of the major software received, in transit, and/or expected from our sister group in Cologne West Germany. Many of us have used or experimented with the original 8432. Well, other than the name, not much else is the same any more. Read the abridged instructions for this new program in the article section of this issue of THE CBUG ESCAPE. True multitasking -- it will run programs simultaneously in up to 14 banks if you have a 1 meg upgrade, or four banks in a 256K machine, etc. And those are potentially all different programs yet there is the ability to draw upon results from a program in any other bank to assist the execution in the current bank -- all under program control. It would be wild to have a monitor for each bank, as they are all doing their thing at the same time!

These are the things the B-128 was designed to do. As we move along our and our sister group's guru's will be devising more miricles for us lowly consumers to chortle over. Give the genius's some room and time, don't begrudge them the rest of the library this issue for it is of these things that such programs as the 8432 v.g are developed.

A word about the effort at CBUG's end of this rainbow. We received the program and fortunately extensive instructions, BUT, it was all in Deutsch. One of our fine members, a native German, Jurgen Billhofer, translated the text to English resulting in the 133 blocks of SUPERSCRIPT (sequential) files on the disk. Jurgen is quite a computer expert himself -- a professional programmer, with a collection including IBM AT, and the current top of the line Tandy. As a side note, Jurgen has run time/performance tests between the several machines --finding that the B128 can wip the other boys (on comparable programs such as Calc Result by 20% to 50%!. Similar report are comeing in regarding use of the co-processors with CPM-86 vs. IBM, etc. We are now just seeing the B begin to shine!

After Jurgen finished his translation, Mark Schwarzbauer dug into it to bring spellings, grammar, etc. into conventional ideomatic convention. We all owe both Jurgen and Mark a huge debt of gratitude for these efforts. Anyone with German English skills, whether technically oriented or simply a average citizen is urgently requested to register with Mark for future needs in the translation

```
1 "8432 emulator   " aa 2c    5   "tabler"          prg   32  "assembler700"    prg   26  "assfile.bsp"     prg
1   " loader"           prg   3   "t1"              prg   4   "loader-low"      prg   1   "edit"             prg
8   " bootscrn"         prg   3   "t2"              prg   4   "loader-high"     prg   12  "8432.e442.a"      prg
8   " instscrn"         prg   3   "td"              prg   18  "superbasic.a"    prg   3   "src.textout"      prg
91  "8432-1"            seq   3   "tw"              prg   1   "s10"            prg   4   "src.interrupt"    prg
42  "8432-2"            seq   5   "bank F.23.14"    prg   21  "editor700.9"    prg   17  "softfloppy8032"   prg
3   "boot 8432.24"      prg   18  "toolkit.9"       prg   4   "monitor"        prg   9   "renumber 8432"    prg
1   "edit.start"        prg   16  "tom.7"           prg   5   "assembler"      prg   1546 blocks free.
115 "8432.24"           prg   17  "editor700.7"     prg   3   "editor"         prg
```

* *

Additional materials added to this disk are from:

From ANTHONY GOCELIAK

What can I say! Mr. Goceliak just keep on putting out miracles.

```
1 "july 87 ajg     " 50 2c    12  "uncommon DOSsort" seq   24  "better b screen" seq   3   "goto demo"       prg
15  "cbug for/next"     seq   16  "computed goto"    seq   1   "----------"      seq   4   "display select"  prg
18  "preserving disks"  seq   26  "special renaming" seq   0   "demo pgms follow" del   5   "screen configure" prg
15  "short invective"   seq   10  "consistency thik" seq   1   "--------------"  seq       50 additional blocks
```

From JAMES OHALLORAN

See the "article" in this issue to see what this one is about.

```
1 "O'Hallorans Util" 01 2c    8   "load me first"   seq   7   "XTRA symbols"    seq       15 additional blocks
```

From ART CHICK

Art has provided what he believes to be a substantial improvement over amortization programs currently available for the B128. Wallcover calculation deals with the quickly computing the amount of standard double rolls of decorative wall paper much be purchased to do an installation. Pulley size does the math of pully ratios, etc.

```
1 "cbug mort pgm.  " 01 2c     13   "amort.cbm"       prg    23   "Wallcover calc"   prg    12   "--instructions--" seq
7    "amort. loan"       prg    13   "amort. sched"    prg    15   "pulley size"      prg         83 additional blocks
```

CP/M 86 INFO & PGMS /1 Pre Release #8 NEW RELEASE #11611

This is material for use with the 8088 co-processor and is intended for advanced programmers only.

John Wright has available the largest repository of CP/M 86 software in the world. Nearly all of which is public domain! Not a small amount either by many many hundreds of megabytes of it. The whole thing placed on B128/8050 style disks will be an incredible pile..... Alas, however, John is undertaking to get some of the more important information and necessary utility programs out to those CBUG members who've acqired the co-processor board. Admitted these are very very few, but their success with this information and the following three disks will enhance the practicality of a 8088 co-processor equiped B128, which inturn will encourage manufacture of more co-processors, and emprovements on them. We owe a huge vote of thanks to John for undertaking this Herclean task.

While these disks appear as a standard 8050 disk, which infact is what they are, the data on them will need to be ported out of the B128 into a suitable compatable or clone, then back into the B128 under 8088 control. Yes, the 8050 can store data in a form which looks to the software as standard CP/M 86 and/or IBM. However at that point the directory doesn't exist in B128 mode. A few of the files on each disk are in Superscript form for immediate perusal.

There are three disks being released this issue and Mr. Wright intends to continue providing more on a regular basis.

```
1 "cp/m-86        " 01 2c     526  "catalog.doc.1"   seq    74   "lu86.cmd.1"       prg    92   "ffynde.asm.1"     seq
3    "to-norm.sii.1"     seq    63   "apc-dump.lbr.1"  prg    156  "lu86-2.lbr.1"     prg    18   "ffynde.cmd.1"     prg
7    "guide.sii.1"       seq    44   "bidump86.lbr.1"  prg    57   "print86.lbr.1"    prg    16   "ffynde.com.1"     prg
5    "directory.sii.1"   seq    6    "bishow86.cmd.1"  prg    27   "typesq86.lbr.1"   prg    162  "cnvadv.hlp.1"     seq
52   "delbr.com.1"       prg    46   "bishow86.lbr.1"  prg    457  "80t86.lbr.1"      prg    72   "usq.cmd.1"        prg
8    "usq.com.1"         prg    37   "ltype86.lbr.1"   prg    101  "ffynde.a86.1"     seq    16   "sq.cmd.1"         prg
                                                                                              7 blocks free.
```

CP/M 86 INFO & PGMS /2 Pre Release #9 NEW RELEASE #11625

See above. This is material for use with 8088 Coprocessors and is intended for advanced programmers only.

```
1 "cp/m-86        " 02 2c     36   "siosys.a86.1"    seq    102  "mdm840.hq6.1"     prg    21   "cbaddr.z80.1"     seq
2    "to-norm.sii.2"     seq    100  "main.a86.1"      seq    6    "mdm840.nqt.1"     prg    63   "code8086.z80.1"   seq
10   "guide.sii.2"       seq    9    "finish.a86.1"    seq    72   "m8lib.aq6"        prg    1    "codecb.z80.1"     seq
7    "directory.sii.2"   seq    44   "files.a86.1"     seq    14   "m8lib.cmd.1"      prg    19   "codeed.z80.1"     seq
38   "modem-86.doc.1"    seq    8    "deff.a86.1"      seq    3    "m8lib.doc.1"      seq    8    "edaddr.z80.1"     seq
163  "modem9xx.doc.1"    prg    47   "console.a86.1"   seq    16   "m8nm-40.aq6.1"    prg    21   "jumpaddr.z80.1"   seq
100  "sendrecv.a86.1"    seq    34   "8251sys.a86.1"   seq    4    "f47-test.a86.1"   seq    223  "cp4ker.doc.1"     seq
62   "term.a86.1"        seq    33   "2651sys.a86.1"   seq    3    "f47-test.cmd"     prg    55   "cp4ker.com.1"     prg
10   "modem.a86.1"       seq    500  "mdm840.aq6.1"    prg    39   "z80.a86.1"        seq    29 blocks free.
24   "start.a86.1"       seq    105  "mdm840.dqc"      prg    21   "z80.cmd.1"        prg
```

CP/M 86 INFO & PGMS /3 Pre Release #10 NEW RELEASE #11630

See above. This is material for use with 8088 Coprocessors and is intended for advanced programmers only.

```
1 "cp/m-86        " 03 2c     78   "qsort.lbr.1"     prg    333  "fastvf86.lbr.1"   prg    6    "bishow86.cmd.1"   prg
2    "to-norm.sii.3"     seq    251  "ty.lbr.1"        prg    39   "frag86.lbr.1"     prg    46   "bishow86.lbr.1"   prg
10   "guide.sii.2"       seq    78   "apc-caln.lbr.1"  prg    96   "grab86.lbr.1"     prg    37   "ltype86.lbr.1"    prg
6    "directory.sii.3"   seq    74   "apc-date.lbr.1"  prg    31   "save0-86.lbr.1"   prg    57   "print86.lbr.1"    prg
317  "encol.lbr.1"       prg    64   "apcserio.lbr.1"  prg    36   "wc86.lbr.1"       prg    27   "typesq86.lbr.1"   prg
235  "hjelp.lbr.1"       prg    85   "bytype86.lbr.1"  prg    44   "bidump86.lbr.1"   prg    100 blocks free.
```

B128 KERNAL/EDITOR SOURCE CODE CBUG #67 NEW RELEASE #11747

This material comes from CBM U.K. from Weldon Corby in England before they moved to Maidenhead with CBM International. Mark Schwarzbauer was given these materials for CBUG to distribute to it's members. Much of this material reflects

development on the B system and the accuracy is not verifed. This material is released ONLY TO CBUG MEMBERS pursuant to our license agreement with CBM. CBUG does not warrant the materials herein as to either their content or accuracy. We have enclosed both the Kernal and Editor on this one disk. Of course CBM has no responsibility of any kind. Any questions MUST be confined with the CBUG and our Cologne affiliate without exception!

REQUIRED AGREEMENT: No one shall be permitted a copy of any portion or all of this disk CBUG #67 or CBUG #68 without having first submitted an executed copy of the agreement on the back

CBUG members are indeed fortunate in having these materials. So far as is known never has any end user of a consumer computer product been allowed access to hardware source code. This type of information is provided normally only under rigid non-disclosure agreements to necessary inside and outside programmers.

```
1 "ed/kernal       " aa 2c     8   "vectors"          seq    3   "hiloader"         prg    10  "ed."              prg
1   "KERNAL rev.4"     seq     1   "nolist"           seq   33   "kp7/12.bin"       prg    32  "asx4.0."          prg
1   "&&&&&&&&&&&&&&"   seq     8   "save"             seq    2   "txtest2"          seq     3  "ld."              prg
4   "run-prog"         prg     6   "ktemp"            seq   20   "txram"            seq     7  "xref4.0"          prg
10  "ed"               prg    18   "load"             seq   40   "eprombox"          prg    24  "bwinit"           seq
32  "asx4.0"           prg    15   "irq"              seq    1   "txinit1.obj"      seq     8  "bwdisp"           seq
3   "ld"               prg     4   "newkernalp"       seq    3   "txinit2"          seq    30  "bwfun1"           seq
7   "xref"             prg     2   "bup"              prg    3   "txtest1"          seq    20  "bwsubs.old"       seq
8   "unit to unit"     prg    12   "crossover"        seq    2   "txinit1"          seq    17  "bwkybd"           seq
16  "newtapefile"      seq     3   "xtest"            seq    5   "txtest1.obj"      seq    32  "bwtabs"           seq
17  "newtapecontrol"   seq    23   "ipcom"            seq    1   "txinit2.obj"      seq    30  "bwfun2"           seq
3   "kernal"           seq    24   "z80com"           seq    5   "txtest2.obj"      seq     1  "nolist."          seq
2   "scnlist"          seq    49   "monitor"          seq   23   "xrll0000"          seq     6  "disclaimer."      seq
7   "disclaimer"       seq     1   "etest1"           prg   33   "kp3/11.bin"       prg    27  "zdeclare"         seq
33  "declare"          seq     2   "data i/o"         prg   38   "xrff0000"          seq    20  "equate."          seq
22  "equate"           seq     1   "newnolistp"       seq   33   "c64k.bin3csum"     prg    13  "escape"           seq
44  "newread"          seq     1   "# device number"  prg    3   "etestc2"          prg    33  "bwrun"            prg
18  "newwrite"         seq     3   "ipcgo"            seq   33   "kp7/15.bin"       prg     1  "allasm"           seq
4   "newkernal"        seq     1   "nolistp"          seq   33   "c64k2/28.bin3"    prg    17  "test1"            prg
1   "newnolist"        seq     3   "kernalp"          seq   33   "kp7/18.bin"       prg     1  "sced.lib"         seq
6   "time"             seq     1   "nolists"          seq   33   "901227-03c"       prg     1  "newasm"           seq
26  "channelio"        seq    46   "k.obj"            seq   33   "901227-03d"       prg     1  "x-rtest@$400"     prg
3   "messages"         seq    10   "oldtransx"        seq    1   "nlxdlair"         seq     2  "data i/o."        prg
20  "ieee"             seq     3   "finalkernal"      seq   33   "1526.bin.06b"     prg    20  "declare."         seq
19  "rs232"            seq     1   "finalnolist"      seq   33   "901244-04a"       prg    21  "bwsubs"           seq
15  "openchannel"      seq    16   "open"             seq   33   "901244-04b"       prg    17  "xrll0000."        seq
13  "close"            seq    46   "kp.obj"           seq    9   "CBUG INFO"        seq    25  "xrff0000."        seq
7   "clall"            seq     6   "croscbm"          prg    1   "cbmiihang4/25"    prg     1  "xrff0001"         seq
34  "init"             seq    10   "transx"           seq    1   ".&&&&&&&&&&&"     seq     1  "xrff0002"         seq
3   "oldtime"          seq    32   "assmcbm3"         prg    1   "EDITOR rev. 4"    seq   349 blocks free.
8   "errorhandler"     seq    10   "ed7"              prg    4   "run-prog."        prg
```

BASIC SOURCE CODE B128 & OTHERS CBUG #68 NEW RELEASE #11752

NOTE: This material is intended for advanced programmers only.

This material comes from CBM U.K. from Weldon Corby in England before they moved to Maidenhead with CBM international. Mark Schwarzbauer was given these materials for CBUG to distribute to it's members. Much of this material reflects development on the B system and the accuracy is not verifed. This material is released ONLY TO CBUG MEMBERS pursuiant to our license agreement with CBM it is not legal to pass this material to anyone who is not a CBUG member. This material may only be used in conjunction with the B series machines. Neither CBUG nor CBM warrants the materials herein as to either their content or accuracy.

The contents on this disk are the B series Basics including the pet 64 to the 256. There are also notes on errors like the last program "MEMO.29SEPT83.WP".

REQUIRED AGREEMENT: No one shall be permitted a copy of any portion or all of this disk CBUG #67 or CBUG #68 without having first submitted an executed copy of the agreement on the rear of this issue.

CBUG members are indeed fortunate in having these materials. So far as is known never has any end user of a consumer computer product been allowed access to hardware source code. This type of information is provided normally only under rigid non-disclosure agreements to necessary inside and outside programmers.

A passing note; during the engineering stages of the B128/700B series, the name of the machine was PET III or PET 128, etc. This is truly the mother lode for any serious programmer.

```
1 "b-series basics " p4 2c    41   "math4"            seq    1   "pet64no"          seq    61  "cbm128no.mod"     prg
4   "run-prog"         prg    39   "strng1"           seq    1   "pet128no"         seq     2  "compare"          prg
10  "ed"               prg    62   "strng2"           seq    1   "pet128"           seq     1  "cbm256no"         seq
32  "asm"              prg    46   "butes1"           seq    3   "bup"              prg     1  "pet256no"         seq
3   "ld"               prg    37   "butes2"           seq    8   "unit to unit"     prg     1  "err.pu.b"         prg
7   "cref"             prg    29   "renumber"         seq    7   "delete"           seq    65  "cbm256.bin1"      prg
```

```
18  "basic"          seq   1   "cbm128"         seq   18  "init"           seq   4   "lowload"          prg
61  "basdoc"         seq   4   "syscal"         seq   6   "sysvec"         seq   61  "cbm256no.mod"     prg
40  "bdefine"        seq   5   "mover2"         prg   1   "cbm192"         seq   6   "memo.29sept83.b"  prg
38  "tokens"         seq   1   "nolist"         seq   1   "pet192"         seq   5   "patches"          seq
59  "contrl"         seq   1   "nolist2"        seq   1   "cbm128no"       seq   3   "change 8050"      prg
50  "bverbs1"        seq   1   "nolist3"        seq   1   "cbm256"         seq   33  "bas-hi 128v4"     prg
46  "bverbs2"        seq   1   "nolist4"        seq   1   "cbm192no"       seq   65  "cbm128.29sep83.b" prg
39  "bverbs3"        seq   11  "indir"          seq   1   "pet256"         seq   12  "memo.29sept83.wp" prg
50  "math1"          seq   43  "using"          seq   1   "cbm64"          seq   4   "CBUG INFO"         seq
51  "math2"          seq   1   "cbm64no"        seq   1   "pet64"          seq   783 blocks free.
20  "math3"          seq   40  "box"            seq   1   "pet192no"       seq
```

BASIC SOURCE CODE UNDER STUDY CBUG #69 NEW RELEASE #11766

NOTE: This material is intended for advanced programmers only.

This material is independantly derived from the same source disk as is CBUG #68. John Berezinski who obviusly is an outstanding programmer has left members at the CBUG West Chicago area meeting spellbound with the discoveries he is making and the revolutionary programming soon to be released. It appears he has done some considerable work in with the Source Code below which may save you a good deal of study time. John writes as follows:

This is the source code for basic as supplied by Commodore with the exception of a few seq. files that were to long for Superscript to load. These few were edited with a disk doctor and resaved as two parts. The first part under the original name and the second with a 'b' added. Many of the PRGs I have not looked at. I suspect some are incomplete or are meant to be run with a symbolic assembler or other program. Programs suffixed with .mod are probably compiled object code (machine language). Programs suffixed with .b or .wp were done with some type of word proccessor. Some will load as basic, others won't.

```
1 "fourone       " 41 2c   41  "math4"          seq   1   "cbm128"         seq   1   "pet256"          seq
3   "blurb"          prg   39  "strng1"         seq   5   "mover2"         prg   1   "cbm64"           seq
50  "bverbs1b"       seq   62  "strng2"         seq   1   "nolist"         seq   1   "pet64"           seq
59  "contrlb"        seq   46  "butes1"         seq   1   "nolist2"        seq   1   "pet192no"         seq
51  "math2b"         seq   37  "butes2"         seq   1   "nolist3"        seq   61  "cbm128no.mod"    prg
62  "strng2b"        seq   29  "renumber"       seq   1   "nolist4"        seq   2   "compare"          prg
37  "butes2b"        seq   4   "syscal"         seq   1   "cbm64no"        seq   1   "cbm256no"         seq
18  "basic"          seq   11  "indir"          seq   40  "box"            prg   1   "pet256no"         seq
61  "basdoc"         seq   43  "using"          seq   1   "pet64no"        seq   1   "err.pu.b"         prg
40  "bdefine"        seq   7   "delete"         seq   1   "pet128no"       seq   65  "cbm256.bin1"     prg
38  "tokens"         seq   18  "init"           seq   1   "pet128"         seq   4   "lowload"          prg
59  "contrl"         seq   6   "sysvec"         seq   3   "bup"            prg   61  "cbm256no.mod"    prg
50  "bverbs1"        seq   5   "patches"        seq   8   "unit to unit"   prg   6   "memo.29sept83.b" prg
46  "bverbs2"        seq   4   "run-prog"       prg   1   "cbm192"         seq   3   "change 8050"      prg
39  "bverbs3"        seq   10  "ed"             prg   1   "pet192"         seq   33  "bas-hi 128v4"     prg
50  "math1"          seq   32  "asm"            prg   1   "cbm128no"       seq   65  "cbm128.29sep83.b" prg
51  "math2"          seq   3   "ld"             prg   1   "cbm256"         seq   12  "memo.29sept83.wp" prg
20  "math3"          seq   7   "cref"           prg   1   "cbm192no"       seq   525 blocks free.
```

PROGRAMMER'S CHALLENGE SET A SET OF NEW RELEASE #11588

Another bargain just in time for your B128's Christmas stocking. The above 6 disks are available as a set at a total cost of just $35.00 -- BUT, only if ordered by December 25, 1987

SPECIAL BONUS TO ALL MEMBERS

To make sure you know why one should read in every nook and cranney of THE ESCAPE, the rest of the membership can participate in the Season's spirit: CBUG will send without any charge a package of 10 OPUS premium SSDD disks with any order of 6 or more library disks (sets such as 11588 and 11593 count as only as one qualifying item for the purpose of this offer. Just write in on your qualifying order form 1 ea of stock #11077 OPUS disk bonus. You may order a package of 11077 for each group of 6 qualifying library disks ordered on the same order form set. Merchandise does not qualify as a library item.

PRINT FILES, SUMMER 1987 CBUG #70 NEW RELEASE #11606

Due to time constraints, the Library portion of THE ESCAPE is going to the printer before the entire print files are completed. The finished disk will contain all print files for this issue, articles, and so long as there is room on the disk, advertising and order forms.

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.

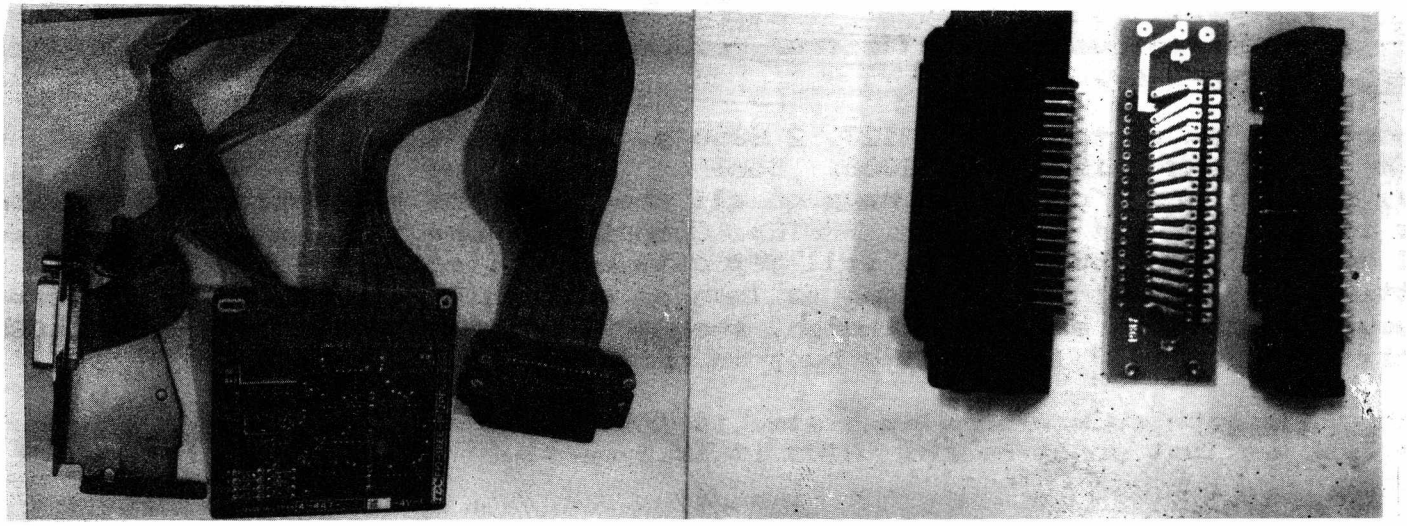
PRICING:		
	BASE CHARGE PER CORD	$15.00
	PER RUNNING FOOT OF CABLE	1.00
	PER CONNECTOR	5.00
	Ill. Residents add 7% sales tax	
	Shipping & Handling per order	2.00

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.

* *
* *



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).

simple follow the variable name with a parenthesized expression. In lines 100 to 300 we refer to the ELEMENTS of the array directly using numeric constants. In line 600 we use the looping variable I as the INDEX into the array A. The first time through the loop it will add a(0) to the sum, the second time a(1), and finally a(2).

Since we did not tell BASIC otherwise, it assumed that our A array could have up to 11 elements, numbered 0 to 10. We could have told it how big the array is using the DIM statement:

```
10 dim a(2)
```

This tells BASIC that array a can have 3 elements, 0 to 2. The DIM statement MUST be used if we are going to use more than 10 elements:

```
10 dim array(100)
```

It doesn't take many elements for BASIC to really eat up the memory. The above DIM will cost us over 500 bytes of variable memory. Notice that array names can be any valid variable name.

Commodore BASIC automatically distinguishes between array and non-array variables that use the same name:

```
10 dim x(20)
20 x = 14
30 input x(x-1)
```

This will INPUT the 14th element of array X [X(13)]. The single variable X is a totally different entity from the array X.

Many programmers ignore the zeroeth element of their arrays. This is perfectly OK. Many algorithms are easier to follow if we simply ignore the 0 element. It's up to you.

Please, DO NOT confuse the element number of an array with the contents of an element. The 13 element of an array YY is referenced or changed using YY(13), but there is nothing that says that YY(13) must contain the value 13. We can think of YY(13) as being like any single variable. If the array YY has 21 elements, 0 to 20, we can think of it as actually being 21 variables, YY(0) through YY(20). Or we can think of the array YY as having 21 pigeon holes into which we can place anything we wish. It is common for beginners to confuse the subscript with the contents of an array. How can you keep them straight? Ask yourself, "Just because I live at house number 114 [or whatever], doesn't mean that I am 114, or that my mail must always consist of 114 letters." Got it?

So, if we RUN the above three line program and respond:

```
? 456
```

We know that X(13) STORES 456, just as any other variable might store 456.

Basic doesn't stop here though. It allows us to have arrays that are multi-dimensional:

```
100 for x=1 to 10
110 for y=1 to 10
120 mult(x,y) = x*y
130 next
140 next
```

This simple program stores a multiplication table in the array MULT. Since we didn't use the DIM statement, BASIC will assume that the array has 121 elements MULT(0,0) to MULT(10,10) (11 by 11). We chose not to use 21 of the elements here, those with either subscript 0.

You may be asking which is the column and which is the row. Well that's entirely up to you to decide; it depends upon how you think about it and how you output the results. Most programmers choose to use the first subscript to select the row (Xth row above), and the second subscript to select the column (Yth column above).

As we said earlier, arrays may be multi-dimensional, but they have to be DIMed if they use more than 2 dimensions:

```
100 dim e(2,3,4), f(3,4,5,6)
```

Notice that a single DIM statement can be used to dimension any number of arrays. As stated, arrays can be real memory eaters. The E array will require 3*4*5*(5 bytes/number) = 300 bytes and some more for overhead, while the F array will require 4*5*6*7*(5 bytes/number) = 4200 bytes and some more for overhead.

Of course, BASIC will also allow you to have arrays of integers or strings:

```
200 dim x%(100), y%(20,30), st$(35), lk$(3,13)
```

Integer and string arrays can also be multi-dimensional. They also don't have to be DIMed if you are only going to use up to element 10 (one dimensional) or up to element 10,10 (two dimensional). Please notice that the dimensions used on these strings st$ and lk$ don't have anything to do with how long each string can be. Each of the elements of the string arrays can still hold the maximum of 255 characters.

Back in section 2.2 it was mentioned that the tutorial disk included a discussion on "the relative merits of using integer variables", at that point. At this point in this section the tutorial disk continues that discussion by discussing the relative merits of using integer arrays, which can save a lot of memory when needed. It also discusses the amount of memory needed for string arrays.

Just as there may be 3 variables with the same name but different types (v, v%, v$), so too there may be 3 array variables with the same name but different types:

```
10 dim v(100), v%(250), v$(2,2,8)
```

These 3 arrays can coexist with the three single variables v, v% and v$.

The DIM statement can be allowed to use variables in the subscript limits:

```
10 input"How many pupils, and how many tests"; np, nt
15 if np<1 or nt<1 then print"Try again!": go to 10
20 rem one name per pupil, one score per pupil per test:
30 dim name$(np), sc%(np,nt)
40 print"Enter each pupil's name:
50 for p=1 to np
60 input name$(p)
70 next
80 for p=1 to np
90 print"Enter the";nt;"scores for ";name$(p);":
100 for sc=1 to nt: print"Score";sc;
110 input sc%(p,sc)
120 next: next
```

This incomplete program offers several examples of concepts discussed in this chapter:

(1) The IF/THEN statement with several statements after THEN (line 15)
(2) The DIM statement, with variable number of array elements and various numbers of dimensions (line 30)
(3) FOR/NEXT loops (lines 50, 70, 80, 100, and 120)
(4) Nested FOR/NEXT loops (lines 80, 100 and 120)
(5) The INPUT statement (lines 10, 60 and 110)
(6) Using PRINT to supply a prompt for the INPUT statement (line 100)
(7) Reals, integers, and strings
(8) Coexistence of single and array variables with same name (lines 30 and 100)

The next issue will continue with Chapter 3.

Any questions arising from this tutorial should be sent directly to the author, whose address is given below. Also, you may obtain disks containing the entire tutorial directly from the author. It comes in either a dot-matrix (4023, 4022, 2022, etc.) version, or a letter quality version (6400, 8023, etc.). Each version (1 disk) costs thirteen dollars, or you can obtain both versions for nineteen

column 33 do not print

dollars (both disks). Handling is included. Write to:

Warren D. Swan
1 N 114 Woods Avenue
Wheaton, IL 60188

COMPUTED GOTO ON THE B128

By: Anthony J. Goceliak

Other varieties of basic have the ability to goto or gosub to a line number which is not spelled out, but computed. By this I mean that a line such as:

```
100 goto x
```

or

```
250 gosub v+q
```

is allowed, where the variable value is used as the line number and program execution proceeds to whatever line number x, or (v+q) happened to be.

Emulating this function with our version of basic is not too difficult.

There are several methods, the simplest being the on/goto or on/gosub. Where the value of the expression can assume only a few numbers, this is the most direct way to go about things, but I recently ran into a situation where the program should fan out to one of 255 lines! The on/goto in this case becomes quite strained. with multiple lines of on/gotos separated by lines of adjustment of x. The culprit here is the line length limitation, since 255 different line numbers cannot fit within a 160 character line.

There had to be a better way, and of course there is. Make the first line of the program read exactly like this:

```
0 goto 00010
```

The space between goto and the number, and the five digits of the number are both critical, and must not be deviated from unless you fully understand the operation we are going to undertake.

The line number of this line itself is not important, but the number 0 is highly reccommended to ensure that it will always remain the first line of the program, which IS critical.

Obviously, the line number 'gone to' from line 0 as it is typed in should be adjusted to be the first line of the program, but the leading zeroes must not be removed. What we will do is compute the target line number and poke it into memory right over the digits that were there before and THEN goto 0. It will now act like a springboard and send execution wherever we wish, in this case to the line# (x*100). Substitute any appropriate machinations involving x for the [*100] in line 10000, but beware of unexpected line numbers! My 'goto demo' program on this disk has a trap to eliminate the value of x=100, I hope you can see why.

To implement the computed goto -=- at the appropriate time goto 10000 [the following piece of basic], which may be renumbered to be anywhere in your program.

To emulate the computed gosub, just gosub 10000 instead of goto 10000 at the appropriate time, which will allow fanning to one of an arbitrary number of different subroutines.

```
10000 bank15:if x>640 then end:else
x=x*100:zz=peek(45)+256*peek(46):bank1
10010 yy=int(x/10000):poke zz+6,yy+48:x=x-10000*yy
10020 yy=int(x/1000):poke zz+7,yy+48:x=x-1000*yy
10030 yy=int(x/100):poke zz+8,yy+48:x=x-100*yy
10040 yy=int(x/10):poke zz+9,yy+48:poke zz+10,(x-10*yy+48)
10050 bank15:goto 0
```

If you are sure that the computed line number will never exceed 64000, the maximum our basic allows, then the test in line 10000 may be eliminated. The test should also be modified if you are going to goto some different function of x than x*100.

Similarly, if the range of x is restricted to 3 active digits as in my case, only those pokes which are needed must be done. Therefore line 10040 is not used in my own program.

Finally, if you are absolutely 100% sure that you will never move the start of basic away from the b's normal value, line 10000 may be further simplified as follows:

```
10000 bank1:x=x*100:zz=3
```

or the value of zz may be eliminated entirely, as long as the pokes in lines 10010 to 10040 are each moved appropriately to 9,10,11,12, and 13.

Finally, the whole thing can be coded in a loop, although the demonstration isn't (for completeness and clarity), requiring but one line to perform the entire feat of legerdemain!

FOR / NEXT IN BASIC

By: Anthony J. Goceliak

A few days ago, I received a letter from a non-b computer user who wanted my help in de-bugging his basic program. He had written a program which included a seemingly logical, but non-fatal flaw.

To briefly digress, the fatal flaws are easy to spot, if not fix. At least they stop the program in its tracks. The non-fatal flaws are 'invisible', so to speak. They merely cause the program to do something other than what was intended, but since they do not stop the program, sometimes are very tricky to find.

What he had done is summarized in the following 'skeleton' basic program.

```
100 w=4
110 for x=1 to w
120 print"Pass #";x
130 w=w+1
140 next x
500 print"Do NOT bet that this line will never print!"
510 end
```

Looking at the listing, it is obvious that since w is set to 4 in line #100, when it comes time for line #110 to execute, there will be a for/next loop of four passes set up. That is absolutely correct. However it is NOT the way that my friend thought it would work. He believed that since the value of w is altered in line #130, the number of executions of the for/next loop would be similarly controlled.

The way that Commodore's Basic is written, a for/next loop is 'engraved in stone' the moment it is set up. When the variable steps up [down] to exceed [drop below] the value which was specified in the line <u>at the time it was executed</u> the loop ends.

Perhaps a better way to state the point is that when setting up a 'for/next loop', you have a 'for' variable, but as far as the loop is concerned, there are 'to' and 'step' <u>Constants</u>.

A for/next loop can be prematurely exited by two methods:

1. artificially change the value of the 'for' variable to equal or go beyond the 'to' value. If the value of the 'for' variable has to be preserved, just set some other (otherwise unused) variable to that value.

```
135 xx=x:x=4
```

And, if you can't live without having the 'premature' value of x, (instead of merely continuing to use xx), add this also:

```
145 x=xx
```

2. use the b-128 command 'dispose for'. In the above program, if the following lines are inserted, the loop will exit.

```
135 if w>5 then goto 1000
1000 dispose for:rem this straightens out the stack
1010 print"Now you can bet!":end
```

Remember to 'goto' somewhere outside the now dismembered for/next loop, or you will see an error message! The lines can be made 'all in one' if desired or convienient:

```
135 if w>5 then dispose for:print"You can't make ME print line #500!":goto 510
```

As far as extending an already set up for/next loop is concerned, there is no straightforward way to do it in general, but for the case in which the 'for variable' is NOT used in general computation within the loop, but only as a counter, the following method works. Please note that any condition may be tested for, not merely a keypress. (i.e. 135 if score >= record then x=1 or 135 if name$ < "a" then x=3)

```
135 geta$:if a$<>chr$(13)then x=2
```

This merely adjusts the value of x so that when the instruction in line 140 is encountered (adding the value of the step [default of 1] to x), the new value of x will still be 'within the loop' and around we go again. If you press return, the variable x will no longer be reset and the loop can end.

The mention of 'straightforward' serves to exclude the tricky (and DEFINITELY risky) methods of finding the for/next entries and poking new values there, as well as similar machinations. Frankly with as many variables at our disposal as the b provides, there is little need to use the 'for' variable as anything but a loop counter.

B256 SCREEN DISPLAY ON B128'S

By: Anthony J. Goceliak

The b is a nice computer, except when you try to read the screen display. Need I elaborate on the screen display?

People have told me how nice the high profile screen is to read, although I haven't seen one myself, but the thought occurred to me that since the kernal is identical between the two machines, and since it IS the kernal which is responsible for screen output, the high profile display code must be present, but un-invoked within a b-128. A short search through the wake-up code yields the code which accompanies this task. It allows, under user control, the selection of normal b-128, high profile, or european variations of the screen display. One drawback - you will need a really tolerant monitor to utilize the 'straight' b-256 display organization. CBM, having built in their own monitor, had no need to keep the horizontal scan rate to NTSC standards. The horizontal rate is important because most monitors use a flyback transformer driven by the horizontal oscillator to generate the very high voltage necessary for the CRT. Undervoltage, due to off-resonance operation can lead to problems with focusing, low brightness of the display, and if an exceptionally marginal (read cheap) circuit was used in the video monitor, even possible circuit damage to the monitor.

Additionally, the display will probably not be capable of being sync'ed in the horizontal direction. Either a very sophisticated monitor, or an over-designed one are necessary to use the b-256 display. Oddly, besides one 'super-duper' monitor which I borrowed to determine how good the display quality is [very good indeed], my trusty Samsung $69.00 special, needing no modifications, albeit with all controls at their extreme ends, can yield a stable display just like the b-256!

One monitor which will definitely NOT be capable of this is the Zenith (the one in the Protecto special package). When I first got my b, the display s l o w l y wiggled in the horizontal direction. To test your own monitor, place your fingernail directly on the screen with your finger pointing left or right, covering just exactly a letter 'l'. Wait perhaps 10 seconds. If the 'l' disappeared, and then re-appeared, you've got the 'zenith wiggles'.

It is perhaps unfair to call this the zenith wiggles, because, as I said, when my b package arrived, I saw this and wanted to eliminate it. Being a good technician, the first thing I did was substitute monitors, temporarily(!) stealing my 1802 computer's monitor. Aha! no wiggle. But just to confirm the trouble, I put the zenith to work on the 1802 computer. Aha! no wiggle either! So that was that, and I never investigated further to see why the display wiggled only when the b and zenith were together, but since several other people have remarked upon this phenomenon, it is probably not an isolated case.

So far, so good, but what can be done to make the highboy display work with something other than a monitor which cost more than the b, or a Samsung?

Enter the second program, which is written in basic only [somehow it is satisfying to manipulate the innermost recesses of the computer operating system from a 'high level language']. By manipulating the control registers of the video display controller chip, it is possible to create a 'pseudo b-256' display, which will give a nicely spaced display, with minimal tweaking of the horizontal hold control, but making up for the many more lines of display [12 raster lines per line of text instead of 8] by discarding the vertical sync rate of 60 cps. Except in the case of the wiggling monitor, where the horizontal gyrations will no longer take a leisurly pace, the vertical sync rate is not critical to the proper operation of a monitor. The 'displayable' part of the screen will grow in the vertical direction, and therefore the height and vertical linearity will need adjustment, but the increase in legibility will make up for your one-time investment of perhaps 5 minutes to 'play' with the knobs on your monitor.

The program is pure basic, and should be 'dloaded' and run. To assist you in adjusting the vertical and horizontal sync, and then the height and linearity, the program temporarily reverses the screen. Do not give up on the display until you have adjusted everything to yield a display which shows the ENTIRE reversed area of the screen, surrounded by at least some black. The display will not be noticeably better at the start.

Wibbly-wobbly displays will be intolerable to watch since the vertical rate is no longer near 60 cps, but otherwise the display is quite remarkable. The individual characters are a bit smaller, but I can read them from further away, and with no strain. You must judge for yourself.

Once the screen has been re-configured in this way, Superscript II or any other basic or m/l program which does not alter the screen configuration may be loaded and run. I am not aware of any programs which do alter the screen configuration, but if something unexpected happens on screen, you probably found one.

This does not affect the number of lines of text (still 25), print statements or poke addresses to the screen, (bank15:poke 53248,01 still does what it used to), so essentially any program that used to run, still should.

The program is of course a soft change, and must be re-run (although the monitor adjustmants will not need to be repeated) whenever you power-up or press the little red reset button on the right rear of the b. In my own experience, I have found that following these adjustments, if the b was powered down and up again to disconnect the

screen re-configuration, a stable, though shrunken display is generated even before re-configuring, so I don't have to blindly trust my spelling of 'dload"screen configure"'!

SPECIAL FILE RENAMING ON THE B

By: Anthony J. Goceliak

If you have ever looked carefully at a directory display on your b, you will have noticed that all the columns line up neatly whether the name of the file is one or sixteen [or any number between] characters in length. That suggests to me at least, that there is room for a bit of manipulation with regards the filename, and it turns out to be true.

Consider the following hypothetical directory:

```
0 "whatever disk"    54  2c
20    "geeky"             prg
3     "more geeky"        prg
5     "geeky inst."       seq
50    "data"              rel
1984 blocks free
```

The demonstration above is so short that missing the instruction file is unlikely, but imagine one of those miscellaneous disks from CBUG with perhaps 150 or so files on it, especially when the instruction files are separated from the actual program they relate to.

What I'm getting at is wouldn't it be nice if the files could somehow be flagged to indicate coupling, but NOT at the expense of the programmer. What this means is that if the CBUG librarian, or you, wish to monkey with filenames using this system, it will be ok, EVEN IF ONE OF MY PROGRAMS CALLS ANOTHER FROM INSIDE ITSELF!

Perhaps that isn't very clear. Many times if there is a need for a data file, it will be called from within the execution of the main program, and if you renamed that data file without making the relevant change(s) in the main program, too bad! Instruction files fall into this category too sometimes, although the consequences of missing the 'dclose' on a read file are not nearly as severe as the equivalent when the file was used for writing! In any case it calls for work, and exacting work at that. Line number 2345 for instance may build f$, and f$ will be the filename for the data file. Line 5432 may re-build it, due to changed circumstances, or the program user having selected a different option, anyway looking for all references to a given filename within a basic program is not nearly as easy as it sounds. And this doesn't even begin to scratch the surface of short m/l utilities bloaded and then sys'ed to.

Ok, what is this magic which will allow us to visibly couple files, but not affect their 'official' filename? Strange and Exotic disk programming? Stranger and more Exotic M/l run on the b? Superbase application code executed by the JCL cartridge? Wires cut on the IEEE cable with a simple 200 IC board installed? No, it is the rename command.

Rename? Yes, rename. Of course there is a minor trick to this, or you will indeed rename the file.

The inevitable brief diversion

Inspection of a disk directory with a disk editor shows that Commodore uses the shifted space [or the filling of the allotted 16 characters] to indicate the termination of a filename. That means DOS considers only those characters which preceed the first shifted space to be the filename, or if no shifted space was encountered, the first sixteen characters.

Obviously, the insertion of a shifted space before the end of the allotted 16 characters causes DOS, and remarkably the b also to respond perfectly! Anything entered after the shifted space, but before the 16th character shows up on the directory, but PAST the second quote mark (Does anybody in the whole world besides Commodore call that an inverted double comma?).

End of detour

Try this on for size. From basic, in direct mode (no program running and blinking cusor and ready on screen), place one of your disks in drive #0, [obviously altering the filename to agree with one of the existing filenames on the disk].

rename d0,"geeky" to "geeky"+chr$(160)+"a1" (and press return!)

Or to get a bit more fancy:

```
f$="geeky"
renamed0,(f$)to(f$)+chr$(160)+left$("
",13-len(f$))+"a1"
```

where what is between the quotes is 13 spaces. This will right-justify the 'a1', or for non-superscript oriented b owners, put all the added characters in a nice column under the disk id in a directory display, without regard for the length of the 'official' name.

The command works with either drive as well, by changing the d0 to d1. As long as the new string of characters does not exceed 16 (i.e. the old name was 13 or fewer characters) the rename will proceed and the next time you list your directory you will see the following:

```
0 "whatever disk"    54  2c
20    "geeky"a1           prg
3     "more geeky"        prg
5     "geeky inst."       seq
50    "data"              rel
1984 blocks free
```

or if you did the 'fancy' command, and proceeded to do all the files:

```
0 "whatever disk"    54  2c
20    "geeky"         a1  prg
3     "more geeky"    a2  prg
5     "geeky inst."   a1  seq
50    "data"          c7  rel
1984 blocks free
```

If programmer 'a' is flagged in this way, 35 file groups can be rapidly associated with his (her) efforts, tieing together the main program, its instruction file, data file (s), and any m/l or screen bload files with the same designation [a1]. This uses no special keys or characters, only the alphabet and the numbers from 1 to 9. I suggest using the zero [a0] for a ubiquitous utility used by more than one file, the back arrow to flag bloadable screen displays, and the percent sign (%) to indicate machine language routines, but the actual implementation of the system is left up to YOU, the user, where it should be left.

dloading "geeky" (if the file was originally designed to dload in the first place) will still be successful, since the 'official' filename has not been changed. Now however, there is a trailer, which can be used to associate all 'connected' files! Use whatever suits your fancy, decimal or hexadecimal numbers, program author's initials, etc, etc. And the program author, who is frankly no more likely to remember all the line references to files than you, can breathe more easily.

Obviously this system allows flagging those bloadable files differently from the dloadable, eliminating the (not too successful) attempt to dload and run m/l utilities, or screen pictures without need to alter the official filename.

Now if we can only make this work without having to shrink the filename from 16 to 13 characters.....

AVOIDING NEEDLESS DISK DESTRUCTION

By: Anthony J. Goceliak

This article was prompted by two letters from CBUG members asking for help in diagnosing 'mysterious and intermittant' corruption of disks by 8050's even though write protect tabs were installed on one set of them. One member lost several Superscript II dictionary disks and ultimately the Superscript II disk itself, while another lost a disk with Superbase data on it. Both members were in the habit of leaving their disks in place all the time, namely while powering up and powering down.

May I briefly refer you to the b-128 silver User's guide, page #43, the warning printed in bold type.

Commodore has taken pains to eliminate the possibility of the drive powering up in the write mode, and while the circuitry looks good to prevent disk destruction when going up, the sad fact is that on the way down, you are watching a race between power supplies, and maybe/maybe not a false write pulse can be created on disk.

The state of current flow through the read/write head is [in quaint technical terms] 'undefined' during power down. What this means is that Commodore does not guarantee that disaster will NOT strike during this period, and disaster quite often can!

Hardware oriented members are invited to repeat an experiment to demonstrate this. Take a blank, unformatted disk, [bulk erasing an expendable disk is also ok], place it in the active drive of your (already powered up) 8050, (the active drive is d0 on power-up), and with an oscilloscope connected to the analog board, watch what is displayed when you spin the drive by opening and closing the drive door. Not much. Now disconnect the test equipment [to prove that the following is not just due to the extra equipment hung on to the drive circuitry] and with the disk in place power down. Power up, tease the door to spin the disk and repeat a few times. Reconnect the O-scope, and watch again. Believe me, those who perform this experiment will never leave a vital disk in the drive when powering down ever again.

The reason for a totally blank disk is of course to help you find the extraneous pulses. With no 'forest' of normal pulses, the 'trees' really stand out.

Two things work for you, and perversely one 'cuts two ways' in preventing a disk from becoming a frisbee instead of a program repository.

The sophisticated hardware pulse rejection circuitry which is common to the entire 8050 drive family can be VERY effective at coping with these extraneous pulses, but they do exist, and should one be (un)fortunately placed, one bit will be trash on your disk. Big deal, one measily bit out of over five million? WRONG!! One bad bit can make an entire program fail to load!

The other thing in your favor can be the 'vast wasteland' of unused spaces on a disk, (spaces between the places where the drive read clock expects to see a bit, as well as header and tail gaps, but also currently unused blocks as well), so that an extraneous pulse will often simply be where your files ain't.

The dual edged sword is the prediliction of DOS for the directory track. Should the extraneous bit be located at an unfortunate location within one of the header blocks on the directory track, well, ask the above-mentioned member what happens then. However, the file itself still exists and with a block by block disk copying program and a disk editor to repair a defective directory on a new disk, it should be possible for you to re-create your disk in perhaps 30 minutes to an hour. That should be bad enough, but consider the poor soul with a garbaged out data block. There is no way to tell WHICH bit on that particular disk block is in error, and with rnd file blocks like Superbase data, good luck at figuring out which file or database the block even belongs to. It is not necessarily impossible to repair, but you had better be prepared to spend more than a bit of time doing it! Damage to something like Superscript, Superbase, or even Basic programs instead of data, can be even more bewildering. In ALL cases of disk repair, it is important to remember that repair is art, not science.

Summarizing, while it is possible to Sometimes repair the damage caused by powering up or down with a disk in the drive, it is a whole lot easier to avoid in the first place.

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

WAR DECLARED AGAINST MIND SLOTH

By: Anthony J. Goceliak

<<See footnote below>>

About two weeks ago, our fearless leader contacted me, and in a quite diplomatic manner attempted to persuade me to write articles which are 'not so esoteric'. Apparently some number of you are complaining to Norman that CBUG's resources are not being deployed to the maximum benefit of the membership. The reason that my reply has taken two weeks to formulate is the fear of jeopardizing CBUG's mailing privileges should he have printed my initial reaction. Post 1 regulations are quite unforgiving when dealing with that sort of invective.

First, I'm not a club resource! No pay, so you get no say in what I work on.

What do you people out there want? Articles on how to use Superscript? READ the book. Articles on basic? What's that you say? You don't even USE basic? Too hard? Gee, now you have two choices. READ your user's guides, or for a far more comprehensive and lucid presentation buy Mr. Swan's BASIC TUTORIAL and read IT.

Now, having read the available information, and still puzzled, try READing the 'Yell for Help' section of the Escape. Although actively discouraging telephone calls (a letter doesn't interrupt me at some time which you felt was convienient for yourself), even I have been known to respond to a letter that demonstrates the writer has first made some effort at sorting out his troubles.

The keyword in each of these suggestions is READ. Having failed to avail yourself of the extensive amount of help already extant concerning each of these topics, what reason do I have to believe that my time would be fruitfully used in an attempt to 're-explain the wheel' to people who use their documentation to balance the short leg of the table? Someone once told me that it was a shame that there was no such job as public relations director for the plague, because he knew the perfect man for it. This is probably true, but I will not let those who refuse to learn attempt to direct the course of CBUG without typing my two cents worth.

CBUG sent one of our ablest diplomats on a business trip to Great Britain, on an airline so luxurious that it has since gone belly up, who had to fly through Newark N.J. the airport that loses Planes, not baggage, who returned with priceless information that would now simply be non-existant, since it was being tossed out with the whole of CBM's european operation. If you think that was such a waste of your dues, I'm going to tell you how to get your money back perhaps ten thousand-fold. Just write a book on how to go to Europe for under $1. Not $1. per day, just under $1. period. That's considerably more than your share of CBUG's lavish european expenses.

What do YOU want to do with CBUG? Turn it into a mutual crying society where we can all moan about how lonely it is to own an orphan computer? I'm not interested. Maybe instead we can show everybody what fools they were for passing up our bargain b machines!

Norman is only human. If you think CBUG is doing a good job, why not tell him. Remember the adage about the squeaky wheel. If you think I'm doing a lousy job, also tell Norman. Considering how much I get for writing for the Escape, it won't take much pleading to get me to stop. However be prepared to write something to fill this space on a regular basis. CBUG, exactly like any other organization, is just what the membership makes it.

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

Footnote from CBUG:

You may have just read a series of preceeding articles by Mr. Goceliak. You may have read some of his prior articles. How long did it take just to type them? How many times more to write them. And how long to do the research. I'd not even attempt to estimate the research hours and money expended for equipment and sundries to do what Tony has done for our group. His reputation, those of his contemporaries has earned them, as well as you and CBUG by association the respect of the Commodore world. Our orphaned pig's ear has long been turned into a silk purse, now we continue to shoot for the moon and beyond.

Tony, like anyone, has his own preferences, habits, likes and dislikes. I did not publish, nor he write, this letter to offend those in need, but rather because it states a very subtle but important position. I've learned time and time again that what he is underlining is the only way -- just like in school. Read, read again, backwards and forwards, then upside down. Each time you will learn a bit more, the meaning of a mystery word, the location of a solution you needed two weeks ago, etc. Some of the Yell for Help group will hold hands and give mini-courses on the phone, others can't.

I was told of an unfortunate situation some months ago where a B128 owner called up one of our guru's and DEMANDED that he be provided with a working co-processor that would run all of the IBM software in the market, or, HE would sell his machine! Of course our guru is unpaid, and has no allegence to any commercial party. Such gaul, to threaten a volunteer with such a childish threat as well. Worse yet, there is not an IBM computer which will run ALL IBM software, etc., etc. How to destroy the mental continuity for the evening!!

The gist of my message is that Tony has done an excellent job of writing the sharp stick communication, something most everyone else has been afraid to tackle for fear of offending the very people we want most to help. Please understand that there is more than the "member helping contact" that makes these wheels turn. No development, no CBUG. Behind the stage is never the fun of infront of the curtin.

Thanks Tony for saying it straight.

YVT, Norm Deltzke

P.S. While CBUG is often terribly late in responding to letters, it is not that we are selective in answering (a privledge of any other member), but that we simply do not have time to do correspondence regularly.

P.P.S. While Tony suggests writing CBUG, please only to tell us were there was a failure breakdown of other detail which needs attention. Send your letters of gratitude to our publishing authors and contributors. It is they who make this users' group function. I just co-ordinate as best I can in the 37 hours per day He left me after my regular job.

SERIAL BUS INTERFACE

By: Gary L. Anderson

In the last issue a few comments were noticed regarding the development of a serial bus interface for the B-128. Let me procede to describe what the Commodore Serial Bus is and what is presently happening regarding this subject.

GENERAL OVERVIEW: The earlier Commodore computers such as the PET 40XX, 80XX, and Super PET used the industry standard parallel IEEE-488 bus to interconect all the peripherals to the computer. Commodore needed a less expensive way to connect peripherals as the IEEE-488 cables and connectors are rather expensive for the mass home market. They came out with a serial bus that daisy chained like the IEEE-488, had only 6 pins in the connector as compared to 24, and used similar terminology in describing the lines namely ATN (attention), DATA, SRQ (service request), CLK (clock), RESET, and GND (ground). The original serial bus as used with the VIC-20 and C-64 was decided to be rather slow so with the C-128 came a dual speed serial bus and a new disk drive, the 1571, to take advantage of the higher speed designed into these newer devices. The main interest in a serial bus interface for the B-128 comes from the C-128's ability to run CP/M. It is envisioned that the B-128 could load in CP/M software from the 1571 disk drive solving what would otherwise be a messy porting problem. Also, other serial bus peripherals and computers could be made to talk to their big brother the B-128.

THE CONCEPT: Presently the hardware is being analyzed and it appears that the best solution is a small external box with 6-pin DIN serial connectors connecting via ribbon cable to the B's user connector pin field. This way both low speed and hi speed modes can be implemented to take full advantage of the high speed 1571 disk drive. Both Dennis Jarvis and James Springer, who will be writing the software, and myself, building the hardware, have come to the same conclusion independently that the cassette port can't handle high speed mode along with the inability of the cassette port to generate a reset on the serial bus. The user connector can handle high speed and a reset circuit with a push button can be built into the external box. The reset switch would only effect the serial bus and not the B-128. The hardware must be able to implement the "controller" function to command the bus for any and all serial peripherals low and high speed. The hardware will also be designed so that software could turn the "B" into a listener/talker. This would enable a C-64 or a C-128 to talk to the "B" via the serial bus. A couple of future possibilities include turning the B-128 into a serial bus analyzer monitoring commands and data, and displaying and recording the traffic on the bus. Another idea would be to have the "B" act as a smart serial bus to IEEE-488 converter letting the C-64 and C-128 etc. gain access to the IEEE-488 peripherals like the 4040, 8050 disk drives and printers. So that is it for now, the serial bus interface is being defined and is presently under development. Write me at the above address with your comments.

By: Gary L. Anderson
2560 Glass Road NE.
Cedar Rapids, Ia. 52402, USA

A REVIEW OF HOWARD HARRISON'S ASSEMBLER/EDITOR

By: Dave Porter

PREFACE

I would like to applaud those people who have made CBUG what it is. Those names that appear in every issue... Warren Swan, Liz Deal, and Angel Matos to name a very few... and those names that don't appear... Norm's family and goodness only knows who else. I can't say that I'm a part of that group. I can say that I have done something to help keep CBUG up and running... I've written the following review for the ESCAPE. I hope it inspires someone else out there... someone like me, to do the same. I am not a writer by a longshot. English has always been my worst subject, but the ESCAPE is worth writing for. The CBUG ESCAPE is the ONLY publication I truly find worth reading cover to cover.

THE REVIEW

I have written this review for one reason. I wrote it to INFORM fellow members about a product I found worthwhile. An assembler is of no use to a non-programmer, so this

review is written with the assumption that the reader is a programmer and familiar with the normal workings of an assembler. This article does not address what an assembler is, or how to program in assembly language. Perhaps there is someone in the group who has the talents (and patience) required to cover this topic, much the same way Warren Swan is covering BASIC.

Background: I'm an electrical engineer with a broad computer programming background. For the past two years I've been programming in assembly language day-in and day-out at work. The code I write at work is used in a graphic digitizer. We control the digitizer with a cpu much the same way Commodore controls the 8050 disk drives with 6502s (two of them inside the 8050 running the show). To program in assembly on the B-128 I needed an assembler. By mere chance (and with a little help from our fearless leader Norm) I got connected to Howard. That was spring 1986, before ASSEMBLER5.5 was officially released. I became Howard's beta-test-site.

I had used several assemblers on the C-64. None impressed me. I was getting an assembler written by an individual I didn't even know, and expected no more than what I had seen for the C-64. Howard's assembler turned out to be far better than I had expected, and since then it has been improved even more.

General: ASSEMBLER5.5 is two programs in one. It is an assembler and a text editor. The editor allows you create/modify source code to be assembled. The assembler turns that source code (assembly language) into object code (machine code). The two programs are loaded and run as one. There is a basic loader for those of you who want to do anything on start-up like redefine the function keys. I just define F11 as the SYS to re-enter the editor. The program does require a ram cartridge, but since the program resides in the cartridge space (it requires an 8k block of ram in bank 15) it doesn't interfere with BASIC, nor do BASIC programs disturb it. This, in my case, is a real blessing since the program I have been working on I want loaded at the start of BASIC. (The start of my program is data which is a basic program which does a "sys" to my main routines, a common trick used on the C-64.) I can exit the assembler/editor, "run" my program and, when I am done, press F11 and I am back in the editor. No re-loading of the assembler/editor or my source-code is necessary unless my program crashes the machine. (This happens even to the best of us once in a while.)

About the editor: The text editor is a screen editor designed with the editing of assembly language source code in mind. It doesn't use line numbers. It may or may not tokenize (crunch) your source code, whichever you prefer. Tokenized source-code requires less memory space and assembles faster, but un-tokenized is more transportable (incase you want to port source-code to or from another assembler.) My source takes up 131 blocks when crunched, and 244 when its not. From within the editor any crunched file can always be un-crunched, and vice-versa. You can scroll or page (forward and backward), "hunt" for a specified string, or "goto" a specific label in the file. Once you get there just edit away. Insert or delete characters, lines, or blocks of lines, or just type over what is there to start with. The editor processes lines the same way the regular BASIC screen-editor does, it doesn't change anything on a line until you press <return>, so if you screw-up a line accidentially you can press a <shifted-return>, or just cursor to a different line and press <return>. The "load" and "save" commands allow you to load or save to drive 0 or 1, of units 8-15. Although it comes configured with unit 8, drive 0 as the default, you can make the default whatever you desire. This is an especially nice feature for anyone with more than one drive unit or a defunct drive 0. You can also save just a "block" of text (a group of lines) as a file. Just mark the beginning and end of the text block, and save it to disk. This is the easy way to make library files from existing source code. The editor also allows you to insert one file into the middle of another. Just load in one file, place the cursor where you want to insert the other, and load it in. The editor doesn't clear memory before a load like BASIC. Last but not least is the ability to hold more than one file in memory at once. You have two "text buffers" for each bank of RAM in your machine. Each bank has a "regular" and an "alternate" text buffer. All the regular buffers are one size and all the alternates are another. There size can be set as you desire but their sum (reg+alt) cannot exceed 56k. This means you cannot edit any single file which exceeds this size. The default locations of the text buffers are: bank 1 locations 1000-d000 and d000-f000. But, again, the editor can be configured to initialize them elsewhere. These are the features I found most useful in the editor.

There are a few things in the editor that I did find annoying:
1) Redrawing the screen. Every time you press <return> on a line it redraws the entire screen. [This has been improved in the latest release.] You don't see it doing it (unless you changed a line and cursored away without pressing <return> as described above) but you notice it if you repeatedly press <return> to move down the screen. If you do this, you notice a significant time (maybe 1/2 sec per <return>) before the cursor re-appears. I must give it some credit though, at least it doesn't wipe-out the text like superscript does.
2) No uppercase permitted. Shifted letters are reserved for editor commands ("I" is insert line, "D" is delete line, "L" is load file, etc.). Again I must defend it though. I complained at first, and suggested that it might use the control key instead of the shift, but then I got a version to use on my 8032... The 8032 has no control key! This brings up another point. ASSEMBLER5.5 was originally developed on a VIC-20, and can run on the VIC, C-64, or B-128.

About the assembler: The assembler is invoked with the "A"(ssemble) command from within the editor (unless you don't use the editor, then you can configure it to power-up in the assembler). If the current text buffer isn't empty you are given the option to save (or replace) before assembling, then assembly is attempted on the contents of the current text buffer. If the current text buffer is empty, you are prompted for the name of the file to be assembled. Note: If you assemble from the text buffer, the assembler reads the source code right from memory. If you assemble from a file, the assembler reads the source line-by-line from the drive. Assembling from the drive has NO file size restrictions (except it must fit on the disk) but is slower than doing it from memory. While you can't use this editor to create a source file larger than 56k, if you find another way to create one you can use this assembler to assemble it. I assemble from memory, but include some LIB(rary) files to get me of the source from disk. This allows me to assemble more than 56k because me of it is on disk, but it assembles fast because the bulk of it is in memory.

The assembler always assembles to memory. If you tell it to write to disk it first assembles to memory, then saves memory to disk. Why do I mention this? Because if you aren't careful you can assemble right over your text buffer! Not to worry too much, I have never had a problem. I configured my system to initialize with my text buffer in bank 2, while I always assemble in bank 1 (because part of my code is BASIC as I mentioned earlier.)

The assembler has two unusual pseudo-ops. They are "BNK" and "AS@". BNK allows you to specify what bank you want to assemble to. It defaults to the current text bank (the bank your source code is in!) The other instruction, AS@ can be used to assemble the object code at an address other than that specified by the ORG. For example, if you want to write an assembly language program that will run in bank 15 in the cartridge space (address $2000) you can't assemble it there because the assembler is there. You can:
A) Assemble into the current text bank, making sure that your text is not at address $2000. This may mean using the

alternate text buffer.
B) Assemble into a different bank. Note: this is not an option on the VIC or C-64
C) Save your source to, and then assemble from, disk.
D) Use the AS@ instruction to assemble your code elsewhere in the current text bank.
If you select option D then you ORG at $2000 and AS@ $D000 (for example.) This will assemble the code in memory at $D000 but it will be assembled to run at $2000. When using an AS@ you almost always have the assembler save the object code to disk for you (it is saved as a PGM file with the correct load address of $2000.) I have not found assembling to memory to be a problem except when I use the assembler on one of the other machines (C-64 or VIC) where memory space isn't so plentiful. There my source barely fits, and to assemble I must select option C above (painfully slow with a 1541 on the serial bus.) Note that there is little need for AS@ on the B-128. It is used more on the VIC and C-64.

Let me get back on track once again. All assemblers for the 65xx series microprocessors are the same when it comes to regular op-codes (LDA, STA, etc.) The differences are mostly in pseudo-ops. I just mentioned the two that I find unusual. Here's a complete list: AS@, BNK, BYT, EQU, LIB, LST, OPT, ORG, REL, RES, SAV, SYM, TXT, WOR.

BYT, TXT, WRD and RES allow you to put data (bytes or words of your selection) into your program or to REServe a number of bytes. Note: Text is supported in the TXT and BYT statments, but it must be lower-case). ORG is required to specify the start address of your code. EQU allows you to assign values to labels. OPT allows you to select the following options: 1)turn the listing on and off. 2)automatic insertion of BRK instructions at every label (for use with the debugger "BIRQ").

LIB, LST, REL, SAV and SYM all deal with files. LIB allows you to "include" a LIBrary file as part of your source code. LST is used to save a listing of your assembled source. REL is used to save a ".rel" file for relocating object code (if you desire). SAV is used to save the object code. SYM is used to save the symbol table (an ASCII list of labels and their values.) All of these file pseudo-ops and the (L)oad, (S)ave, and (P)rint commands can communicate with any device. Theoretically you can r/w on devices 0-7, secondary addresses 0-15, and on devices 8-15, drive 0 or 1. This means you can direct output to a disk file, a printer (or any other IEEE or RS232 device), cassette or the screen, and receive input from a disk or other IEEE device, cassette, or the keyboard. I say theoretically because: 1) When accepting input from the KB there is no way to indicate end-of-file. 2) RS232 communication is limited to: Transmission only, No CBM to ASCII conversion, 7 data, odd or no parity, and 1 stop bit (any baud rate.) This is because the filename must be part of your text and thus the character values are limited to 32-95, and because the secondary address cannot exceed 15. Note to users: RS232 filename must be 4 characters long (last two aren't used but must be there.) 3) Cassette communication: only one file can be opened at a time (for obvious reasons), and is not supported at all on the B-128 because the cassette port is not supported by the kernal. I mention cassette only because someone might have a use for it on a VIC or C-64. I've never actually tried it.

I've already been too verbose... It's time I bring all of this to a close. Everyone has different tastes. I like this assembler because: 1) I can customize it to suit me (and my hardware environment.) 2) It assembles fast. and 3) I found it easy to learn and use.

WHERE ARE OUR MEMBERS?

We thought the membership would be interested in the locations of all our members. Following is the postal code listing (foreign entries are strangly sorted due to the fact that we can not use a foreign postal code beginning with a number as it will be sorted into the domestic bulk mail. In some cases we use the foreign postal code, in other cases the country. If you recall your mailing label, there is a delimiter character in the 21st column of the postal code - which is our key name file. The delimiter avoides duplicate key problems.

The NNN.. codes are the postal break points for presorting as to post office distribution points.

We do not provide Dramamine if you try to read this!

006..	01830	027..	03060	05401	06611	07506	08318
00612	01854	027..	03060	05446	06776	076..	08323
00619	01887	02719	03060	05486	06790	07643	08330
00657	01906	02739	03060	05674	06790	07644	08332
00731	01915	02740	03061	05674	06790	07666	08344
00732	01945	02742	03063	05847	06798	07666	08501
008..	01966	02745	03103	060..	06798	07675	08501
009..	01983	02762	03106	06010	06820	07719	08618
00905	02021	02771	03240	06027	06906	07724	08619
00908	02021	02780	03264	06040	070..	07830	08648
00921	02062	028..	03301	06066	07003	07840	08753
00927	021..	02812	03576	06096	07024	07960	088..
010..	02108	02813	03801	06110	07032	080..	08807
01002	02124	02814	03878	06112	07067	080..	08807
01012	02131	02865	039..	06340	07071	08009	08833
01201	02136	02878	03901	064..	07072	08010	08852
01201	02139	02903	03904	06410	07080	08012	08854
01440	02151	030..	04009	06417	07080	08015	090..
01520	02151	030..	04041	06470	07104	08037	09012
01604	02152	03001	04092	06473	07104	08037	09114
01605	02159	03036	04096	06475	07108	08043	09159
01720	02173	03051	04441	06483	07111	08060	09194
01752	02193	03053	04762	06489	07306	08081	09524
01757	02205	03053	04862	06497	07420	08085	09701
01757	02359	03054	050..	06514	07451	08088	
01801	02543	03054	05401	06610	07470	08097	

NY	11203	11754	13357	14305	15238	17018	19083
10003	11203	11757	13357	14432	15243	17073	19120
10010	11213	11782	13493	14437	15260	17078	19128
10010	11221	11783	13493	14467	15282	17105	19138
10019	11229	11787	13502	14470	15317	17315	193..
10022	11234	11788	13601	14513	15327	17331	19355
10027	11369	11788	13634	14521	15736	17579	19380
10040	11413	11788	13685	14559	15801	17601	19380
10128	11421	11795	13827	14608	15801	17603	19380
10305	11423	11967	13838	14612	15823	17815	19380
10308	11507	12065	13865	14617	15846	18042	19380
10314	11542	12070	13865	14777	15905	18049	19380
104..	11552	12078	13903	150..	15905	18056	19382
10457	11561	12114	13903	15005	15934	18102	19403
10457	11565	12414	14008	15024	15945	18411	19403
10458	11566	12501	14028	15101	15963	18452	19406
10463	11691	12531	14048	15106	16046	18901	19406
10467	117..	12534	14059	15132	16101	18917	19446
10469	11701	12550	14075	15146	16117	18969	19460
10475	11704	12550	14092	152..	16148	19002	19473
10530	11705	12754	14120	15206	16502	19004	19605
10538	11706	12801	14174	15216	16507	19018	19606
10548	11717	12853	14210	15217	16508	19047	19606
10566	11726	12870	14211	15218	16602	19065	197..
10940	11729	130..	14218	15227	16652	19067	
11040	11729	13045	14221	15234	169..	19073	
112..	11735	13135	14221	15236	17013	19073	

200..	21014	21643	22405	24061	27249	28390	28779
20001	21014	21769	22554	24174	27278	28392	28804
20001	21031	21801	22554	24201	27285	28403	290..
20008	21044	21801	22554	24219	27288	28405	29161
20008	21053	21842	22601	24333	27292	28512	29201
20036	21061	21842	22601	24540	27320	28540	29210
20036	21074	21842	22835	24614	27403	28557	29304
20037	21090	220..	22980	24641	27403	28560	29451
20044	21114	22041	23233	247..	27415	28605	29526
206..	21136	22060	23452	24966	27502	28605	29576
20619	21146	22102	23456	25213	27593	28607	296..
20650	21146	22102	23464	25414	27602	28607	29646

20715	212..	22115	23517	25701	27602	28645	29657
20740	21206	22170	23518	25801	27804	28658	29657
20743	21215	22170	23606	25801	27834	28681	29678
20748	21222	22191	23669	26035	27943	287..	29687
20755	21227	22192	23690	26505	28043	28704	29710
20770	21228	22205	23805	26506	28054	28704	29715
20772	21234	22205	23823	26555	28115	28712	29730
20783	21234	22303	23824	270..	28150	28712	29812
20854	21236	22304	23832	27030	28228	28723	29831
20878	21401	22309	23834	27030	28304	28726	29842
20904	21620	22309	23834	27106	28307	28734	29903
21001	21629	22312	23875	27109	28379	28763	

300..	30605	32405	32780	33324	33595	35810	38108
30001	30650	32446	32796	33405	33603	35988	38116
30038	31039	325..	32804	33405	33613	36107	38134
30046	31076	32503	32812	33406	33615	36109	38163
30060	31082	32504	32899	33415	33625	36116	38242
30061	31093	32505	329..	33428	33703	36301	38320
30066	31401	32506	32902	33428	33704	36302	386..
30084	31404	32570	32903	33430	33801	36507	38803
30130	31513	32570	32904	33432	33803	36541	38827
30136	31520	32571	32927	33436	33813	36609	38834
30141	31524	32578	32931	33442	33830	370..	38863
30214	31558	326..	32935	33452	33910	37042	38930
30236	31705	32601	32937	33466	33940	37064	38930
30246	31707	32601	32952	33467	342..	37210	39056
30247	31707	32608	32953	33482	34228	37321	39083
30253	31707	32611	32960	33497	34230	37334	39150
30317	31770	32615	33055	335..	34239	37343	39208
30339	31771	32630	33060	33515	34248	37421	39305
30340	320..	32640	33114	33522	34647	37421	39466
30341	32068	32670	33125	33526	34668	37601	39501
30342	32097	32678	33126	33528	350..	37643	39532
30345	32216	32678	33126	33533	35051	37664	39564
30420	32219	32724	33127	33546	35055	37690	39567
30501	32221	32741	33166	33552	35055	37760	39648
30525	32244	32748	33166	33563	35136	37814	
30527	32303	32751	33312	33563	35630	37922	
30542	32303	32751	33315	33581	35802	38001	
30546	32303	32751	33320	33595	35810	38104	

400..	43725	44646	45822	46383	480..	48176	48879
40033	43749	44663	45840	46383	480..	48182	48888
40059	43832	44708	45849	46408	48024	48202	48910
40299	43915	44709	45865	46410	48030	48214	48911
40324	43920	44824	460..	46411	48030	48220	49007
40324	43946	44830	46011	46517	48035	48221	49009
40356	440..	44839	46013	46572	48043	48227	49022
40769	44004	44870	46060	46580	48048	48235	49030
41101	44004	44870	46122	46580	48057	48423	49085
41183	44012	44890	46140	46601	48060	48439	49229
41311	44041	44902	46142	46805	48064	48442	49202
42044	44053	45157	46158	46902	48066	48446	49229
42078	44070	45202	46181	46970	48066	48450	49230
42223	44092	45202	462..	47130	48075	48451	49230
42409	44092	45206	46201	47203	48075	48503	49264
430..	44104	45208	46201	47355	48077	48506	49269
430..	44106	45331	46203	47362	48084	48509	49283
43055	44109	45385	46205	47374	48087	48602	49325
43055	44114	454..	46208	47374	48089	48616	49412
43055	44118	45403	46208	47401	48092	48640	49417
43068	44120	45404	46219	47424	48094	48640	49431
43068	44126	45406	46226	47441	481..	48703	49442
43076	44202	45408	46227	47501	48101	48708	49442
43081	44221	45409	46231	47512	48104	488..	49453
43081	44232	45415	463..	47542	48121	48809	49456
43130	44266	45419	46303	47591	48122	48822	49460
43147	44270	45419	46307	47609	48123	48827	49503
43213	44287	45426	46307	47712	48128	48846	49505
43220	44301	45440	46307	47807	48138	48850	49509
43342	44310	45459	46321	47834	48145	48858	49707
43512	44314	45628	46350	47885	48150	48867	49735
43537	44502	45638	46360	47905	48150	48867	49878
43567	44502	45648	46360	47905	48154	48876	49887
43623	44512	45805	46360	47933	48164	48879	
43725	44615	45805	46360	47970	48167	48879	

500..	52001	52404	53146	54235	54941	570..	58301
50134	52060	52405	53150	54304	550..	57004	58355
50158	52240	52405	53172	54440	55104	57005	58401
50201	52240	52405	53184	54446	55303	57022	58545
50208	52253	52577	53206	54467	55432	57028	590..
50309	52302	52601	53220	54467	55455	57078	59068
50310	52314	52654	53566	54467	55455	57501	59230
50311	52332	52722	53583	54501	55720	57501	59501
50316	524..	530..	53703	54702	55746	57528	59624
50546	52402	53029	53704	54703	55760	57625	59701
50616	52402	53081	53705	54806	56152	57701	59715
50616	52402	53081	53705	54843	56301	57702	59801
50662	52402	531..	53714	54848	56301	57709	59872
51031	52402	53105	54016	54888	56312	57718	59872
51103	52402	53105	54130	54901	56482	57783	59901
51237	52403	53115	54220	54935	56601	580..	

600..	60062	60153	60516	60643	61542	640..	66208
600..	60062	60159	60516	60643	61571	64001	66413
60008	60064	60172	60521	60645	61604	64050	66436
60010	60064	60172	60521	60646	61604	64055	66502
60010	60067	60174	60521	60647	61606	64055	66502
60010	60068	60174	60525	60647	61607	64063	66502
60010	60068	60185	60525	60649	61739	64093	66535
60010	60074	60185	60532	60649	61774	64116	66617
60010	60076	60185	60532	60649	61801	64131	67042
60010	60077	60185	60540	60656	61820	64132	67204
60010	60084	60185	60540	60659	61820	64141	67207
60010	60084	60187	60555	60914	61832	64479	67208
60010	60085	60187	606..	60919	61920	64499	67220
60010	60088	60188	60601	60942	62025	64676	67337
60013	60088	60190	60601	610..	62049	64683	67402
60013	60088	60194	60602	61015	62207	648..	67501
60013	60089	60194	60607	61032	62248	64801	67578
60014	60090	60195	60611	61032	62321	64801	67672
60015	60090	60195	60612	61032	62656	64801	67701
60015	60090	60201	60614	61042	62656	65038	67876
60016	60091	60202	60615	61068	62821	65265	67880
60016	60098	60202	60618	61068	62832	65265	680..
60016	60098	60202	60618	61072	62859	653..	68025
60016	601..	604..	60618	61074	62864	65327	68046
60018	60102	60402	60620	611..	62901	654..	68106
60025	60102	60402	60625	61103	630..	65441	68123
60025	60103	60402	60625	61108	63017	65712	68505
60025	60103	60423	60625	61108	63026	65717	68508
60025	60103	60439	60625	61108	63033	65723	68701
60030	60110	60442	60628	61111	63044	660..	68730
60047	60110	60453	60629	61111	63103	66042	68788
60047	60115	60466	60630	61254	63114	66044	68803
60047	60120	60470	60630	61278	63119	66044	68803
60048	60120	60477	60631	61335	63129	66046	68823
60050	60120	605..	60634	61401	63130	66048	68832
60050	60123	60505	60638	61455	63134	66061	68901
60050	60123	60505	60640	61455	63144	66102	68959
60050	60131	60506	60641	61455	63628	66106	68978
60053	60134	60507	60641	61473	63645	66202	69101
60056	60151	60510	60641	61523	63801	66202	61068
60060	60152	60515	60642	61529	63901	66206	

700..	71112	72901	75040	76103	77092	78130	78763
70005	71112	72902	75077	76135	77096	78130	790..
70117	71201	730..	75080	76206	77263	78133	79007
70123	71201	73066	75080	76301	77304	78148	79035
70123	71291	73112	75152	76310	77422	78155	79106
70125	71291	73120	75217	76541	77459	78213	79108
70126	716..	73139	75238	76655	77479	78221	79109
70127	71901	73160	75238	76689	77520	78233	79109
70301	72003	73533	75247	770..	77551	78238	79124
70433	72031	73632	75251	**77042**	**77571**	**78245**	**79408**
70458	72032	74037	75473	**77044**	77590	**78293**	**79412**
70458	72032	74074	75935	**77055**	**77590**	**78336**	**79416**
70501	72301	74074	75943	**77056**	**77627**	**78363**	**79423**
70503	72354	74105	760..	77060	77803	78363	79510
70510	72360	74126	76004	77063	77840	78404	79601
70510	72543	74149	76012	77072	77843	7843-	79601
70570	72641	74429	76014	77072	77901	78501	79607
70809	72653	74134	76031	77075	77954	78520	79701
70810	72653	750..	76048	77076	77954	78577	79703
70816	72663	75004	76048	77079	78016	78596	79831
71006	72701	75020	76051	77080	78073	78597	79902

71105	72801	75023	76067	77087	78130	78741	79912
800..@	80231	82301	84067	85031	85607	85901	89101
80010	80233	82327	84074	85040	85617	85937	89106
80020	80537	82426	84109	85063	85620	86001	89107
80030	80831	82520	84118	85068	85632	86301	89119
80033	80840	82601	84119	85202	857..	86336	89121
80104	80916	82714	84123	85203	85705	86444	89121
80110	80917	832..	84501	85219	85706	870..	89121
80110	81003	83402	84526	85252	85710	87015	89410
80111	81301	83536	84603	85258	85711	87108	89419
80113	81401	83605	84660	85301	85711	87401	89423
801xx	81435	83605	84754	85306	85711	87501	89503
80203	820..	83676	850..	85339	85712	87544	
80204	82001	83702	850..	85345	85715	88101	
80204	82070	83843	85008	85365	85718	88435	
80206	82070	840..	85010	85533	85719	890..	
80229	82082	84006	85021	85534	85748	89030	
80230	82301	84015	85023	85607	85749	891..	
900..	91326	92123	93032	94086	95127	96825	98221
900..	91326	92131	93033	94086	95129	969..	98221
90021	91331	92254	93033	94122	95150	970..	98225
90033	91344	92277	93060	94303	95152	97004	98226
90038	91350	92284	93101	94402	95204	97023	98248
90044	91360	92307	93105	945..	95205	97027	98250
90044	91362	92311	93111	94501	95240	97030	98277
90045	91504	92316	93130	94544	95301	97058	98312
90231	91601	92335	93221	94546	95315	97124	98335
90254	91604	92345	93230	94550	95318	97201	98362
90272	917..	92373	93257	94558	95336	97201	98373
90280	91701	92504	93301	94566	95338	97202	98501
90280	91701	92509	934..	94577	95338	97352	98502
90291	91709	926..	93402	94578	95338	97396	98684
90304	91722	92621	93422	94589	95422	97402	98684
90501	91732	92626	93422	94602	95451	97448	98837
90620	91733	92635	93423	94606	95453	97470	98837
90631	91764	92640	93436	94612	95482	97470	98860
90638	91786	92641	93436	94621	95492	97479	98901
90638	920..	92644	93436	94703	95501	97520	98902
90706	92010	92646	93437	94707	95521	97520	98902
90745	92011	92647	93437	94806	95554	97523	98902
90805	92024	92647	93437	94925	95628	97527	98908
90807	92026	92649	93446	94960	95670	97701	98944
90808	92041	92651	93446	950..	95688	97835	99205
90814	92041	92658	93447	95003	95696	97838	99205
91006	92050	92665	93449	95005	95821	980..	99336
91011	92054	9268	93455	95005	95825	98002	99337
91016	92055	92683	93536	95019	95827	98012	99352
91016	92056	92686	93555	95023	95842	98023	995..
91020	92064	92688	93561	95030	95991	98037	99577
91101	92069	92691	93561	95050	96052	98055	99619
91208	92071	92705	936..	95051	96555	98111	99801
913..	92071	92706	93612	95054	96555	98119	
91303	92077	92708	93702	95065	967..	98125	
91306	92106	92708	93928	95076	96714	98126	
91307	92108	92806	93928	95076	96789	98146	
91320	92117	92806	93955	95112	96789	98166	
91325	92118	93010	94014	95112	96813	98198	
91326	92122	93023	94080	95123	96822	982..	

Afric	H4P 2	L3Y 2	M9C 4	ROK 0	VOK 1
Afric	IG50Q	L4C 1	ME14	R610	V2L 4
AL7 1	Israe	L6L 3	mex..	RG2 9	V2L3N
AUSTR	Israe	L9G 3	MEXIC	SON 2	V3R 8
AUSTR	JOL 1	L9T 3	MEXIC	S4X 2	V6A 1
Austr	KOA 3	M2H 3	NOR 1	S6H 3	V6R 1
Austr	KOJ 1	M2J 3	N3S 3	SE11	V6Z 2
B2G 1	K1R 7	M2N 5	N6A 5	SL6 7	V7R 4
BT52	K2L 1	M2R 3	N6K 1	TA1 4	V8X 3
COSTA	LOC 1	M3C 2	N7 0A	TN4 0	V9Y 1
ENGLA	LOK 1	M3M 2	New Z	Unite	W. GE
H2W 1	LOS 1	M4M 2	OOV 3	Unite	W12 0
H2W 1	L2V 4	M8W 1	POM 2	VOJ 2	W5 4L
					West

TRANSFERRING DATA FROM SUPERBASE TO MS-DOS DATABASES

By: Warren Kernaghan

Perhaps you've been told by one of the MS-DOS bunch that it's impossible to move the information so tediously entered into Superbase from there to the IBM/MS-DOS world. They probably said that the only way to go from Commodore to the real world was to start from scratch and enter it all over again. Well, that isn't true, and here's how to do it:

While in Superbase, use the EXPORT command to create an export file.
Process the exported data into an acceptable delimited file.
Transfer the file from the B128 to the MS-DOS computer.
Prepare a new database format appropriate for the data.
Import the data into the MS-DOS database.

Now to detail each step. There appears to be no way to create an export file directly from Superbase that meets the most common MS-DOS database requirements. (If someone knows how, please let me know). Most require a format as follows: "field1","field2","field3","field4","etc"CRLF Each field is surrounded by quotes, and separated by commas. Each record ends with a carriage return and line feed. Page R-73 in the Superbase manual discusses EXPORT.

Superbase EXPORT command allows a choice of separators and record endings, but I've found no way to output three separators (",") or to get a line feed at the end. The choice was made to create an export file as close as possible to what was wanted, then to process the resultant file.

The command 'EXPORT "filename",","' will create an export file that has fields separated by a space and a comma, with records separated by a space and a carriage return.

Before quitting Superbase, get a printout of the format and status unless one is already at hand. One way to print the format is to select ENTER on menu 1. When the ENTER screen is displayed, press SHIFT/RUN to print it. ESC Q will exit to the menu again. Status provides a list of each field, with types, lengths and order of fields. To print this list while at one of the main menus, type 'print:maintain' then press '1'. When printing is finished, type 'display', and quit Superbase.

What we need now is to add quotes to the beginning of the file, replace the space and comma between fields with "," and replace the space and carriage return with carriage return and line feed. If dealing with a small database, all of this but the linefeeds can be done in Superscript II. (Maybe someone knows how?)

MultiSearch, a program by Bing Hart, and available from him at 816 373-5523, allows search and replacement of text characters as well as control characters, such as carriage returns, quotes, and line feeds. It will read a text file from drive 0 and process it as a new file is written to drive 1.

MultiSearch must be told exactly how to process the file. This means telling it to search for each occurrence of 'space,' replacing it with 'chr$(34),chr$(34)' then to search for 'space chr$(13)' replacing that with 'chr$(34)chr$(13)chr$(10)chr$(34). The program then writes the desired delimited file. At present, a separate small routine is used to prefix the file with quotes. The extra quotes at the end of the file have caused no problems with any of the databases tried. Should an extra record be created, it should be deleted.

The next step is to get the delimited file from the B128 to the MS-DOS computer. The disks may be identical, but MS-DOS surely won't read a disk with the 8050 format. Using the RS-232-C ports seems the easiest way. IBM calls it a serial port, an option not all MS-DOS computers have.

An interconnecting cable is required, which is almost a standard null modem cable except for pins 8 and 11 at the B128 end. Here's the connection table, assuming the short pigtail cable from the MS-DOS computer's 9-pin serial port is used:

MS-DOS DB-25	B128 DB-25
2	3
3	2
4	5
5	4
6	20
7	7
20	6
	8-11

Note that pins 8 and 11 on the B128 connector are jumpered, and that only a 7-conductor cable is needed. A female connector is used at the MS-DOS end, while a male is used at the B128.

Actual data transfer is next. Any terminal program that will transfer a text file should serve. Xmodem protocol isn't needed. My choices are BeeLine for the B128 and Procomm for the MS-DOS computer. BeeLine has a maximum transfer rate of 9600 bps, so use that for both ends. Other settings suggested are 8 bits, no parity, and 1 stop bit, and local echo for BeeLine. Procomm 2.4.2 is an excellent terminal program for MS-DOS computers and is available on many bulletin boards as 'shareware' at $25 for registration if you find it useful. When the MS-DOS terminal program is setup to store the data as 'filename.txt' as the new filename, proceed with the transfer.

The next step is to prepare the new database so that the data can be imported. This is where the Superbase status printout is indispensable. The new format in dBASE III Plus must be made so that each field is the same length and in the same order as it was in Superbase. This sounds terribly difficult, but isn't. dBase will be used as a brief example as it requires field length setting. Not all databases do.

Load dBASE and get to the dot prompt. Type 'create' and you'll see 'Enter the name of the new file:'. Enter the name desired. A screen will be displayed with highlighted blocks for the entry of field names, types of fields, field widths, and the number of decimal places. Using the Superbase status printout (and the format for clues, if single digits were used for field names) go down the list entering field names, etc, making sure the field width/length is the same as in Superbase. If you don't like the order of fields or don't want all of them, this is NOT the time to attempt change. When all of the fields have been entered, press RETURN on an empty field, then again to confirm completion. The format will be stored on disk. Press 'n' to decline data entry. That's all there is to making a simple format.

To import the data previously transferred, make sure it is on the same disk, then type 'append from filename.txt delimited' using whatever filename was assigned. dBASE will display an account of records added. The total should equal that previously shown when Superbase was in use.

To examine the data in the new database, enter 'go 1', then 'list'. The entire database will be displayed. Several fields can be displayed for a neater check by entering 'list fieldx, fieldy, fieldz'. Refer to the database instructions for more details.

Note: for some reason not yet investigated, date fields from Superbase are incorrectly transferred. They must be re-entered manually.

If in viewing the data you find that something went wrong, what to do? Well, going back to the beginning, the export file from Superbase and the processed file from Multisearch can both be read with SuperScript II, at least until it runs out of memory. A sequential file reader may be a good choice. The exported file should have fields separated by a space and a comma; the records should end with a space and carriage return. MultiSearch output files should appear as "field","field","etc"CR then (with SuperScript II at least) a 'j' should appear first on the next line, depicting a line feed.

If all looks well so far, load a wordprocessor into the MS-DOS computer, and load 'filename.txt' from the database disk. If all looks as it should, with the appropriate quotes and commas (you won't see the carriage returns and linefeeds on most wordprocessor programs) the problem is likely the new format and it's structure not matching the data.

Anyone wanting a database transferred from Superbase to a dBASE III Plus or PFS: Professional File format can send me a duplicate (NOT your only) disk. I'll transfer the data into a simple new format with the same order of fields for $40 for the first format. Any changes, revisions, calculations, applications programming, etc. are not included. This offer is for reasonably simple databases. Should a very complex, difficult, or seemingly impossible database arrive, I'll contact you. If it isn't transferable for any reason, the check will be returned, but not the disk. Address: 901 E. 108th. Street, Kansas City, MO 64131. 816 942-3615.

Superbase data transfers from Commodore C-64 or C-128 computers may also be possible, but hasn't been tried yet.

WS CP/M-86 vs SS2
Part 1 of 2

By: Bruce Faierson

This column will be a continuing attempt to educate the B user to other software and operating systems on the market. This column will only attempt to explore products known to work on the B series computer and will not cover the same products or more current products for other systems. The author does not represent himself to be an expert on the B-series, CP/M-86 or MS-DOS programs reviewed, though the author does have sufficient competence and experience with all the programs. It should also be stated that most programs offer more features than most any normal user will ever use. Therefore, the only true evaluation can be by the user and for the features that the user requires.

I have recently found an old computer review that I found buried among mounds of books. This article was about the new 16 bit computers coming to the marketplace. Since I have been working with pseudo 16 bit computers for some time, I thought it would be worth reading. This article was written around 1983 and went into an in-depth discussion of all the new models and their features. Just as I was about to put the magazine down, I read about the brand new $3000 Commodore entry into the field.

Wow, I wondered what computer this could be. The most interesting statement was that **this computer was setting** a new pricing trend in the marketplace. **I was beginning** to wonder, what kind of marketplace **had Commodore machines** competing on the low end at $3,000. **I rapidly paged through** the article to the specifications page.

There it was, in a non-Commodore publication, and being touted highly was the incredible BX-256-80 with dual floppy disk drives with dma access. What a machine! Not only did it have the competitors beat on price, but can you imagine that this machine had four operating systems designed for it. The best the other machines had were two and did not

come with half of the expansion ports. All I can say is, "Commodore, what happened to you?".

This article will cover some of the features of both Wordstar and Superscript II. Even though there are other good word processors available, including Wordcraft and Word Result, it is obvious that SS2 is the favored word processor for the B-128. Superscript III has good points and some weak points but is not as well used as SS2. A little history follows.

Superscript was the highly developed and polished version of a user group word processor. Simon Tranmer had written the program for his user group and found it to be commercially viable. The user interface resembled other popular word processors.

Wordstar was the program that brought CPM-80 of age in the late seventies. This program has set more standards, helped establish more operating systems and sold more computers than any other program in computer history. Don't let anyone fool you, this program has millions of users in an established base, and developed a user interface that is still one of the best today. A user interface is the command structure of the program. Many famous programs such as Dbase and Supercalc use this command structure. It sure makes life easier when moving from one program to another to use the same keys to perform commands.

Most good word processors will do the same basic functions. The differences are in the ease of use, speed, macro ability and user friendliness to name a few. There is no one word processor that will be the best for all users. Each one has certain strengths and weaknesses that will appeal or possibly repel users. As we go through some of these program features, remember that these programs were written several years ago. At the time they were among the first and the best. As with anything, the longer a product is on the market the more features it will have.

The first item you notice when you boot up SS2 is the configuration menu. While this menu is convenient for changing page width, device number and printer it would seem better to have it put in a configuration file. By doing it this way, the user would not have to always answer these prompts every time you enter the program. It is a good idea, however, to have this menu readily available for quick changes from within a document.

Wordstar has an installation program where you can pick a particular terminal configuration and change your default formatting commands. As an example, if all your documents were going to have a left margin of 0 and a right margin of 65, you would set those as defaults in the installation program. If you wanted to change them, you would either reinstall the program or use commands within the program to change them. All of Wordstar's editing commands are either alterable through the menu structure or through dot commands. The dot commands are similar to the SS2 formatting commands.

Wordstar boots up into what is known as the Opening Menu. Through this menu you can change drives, display a directory, print a file, load a file, set the help level and do most DOS functions. All the commands from this menu are single key commands. This means that you only have to touch one key to activate the command. This menu is very useful as it allows rapid access to many normally used functions. You can also direct Wordstar to load a particular file upon booting the program. You can do this by specifying the name of the file in the execution command.

With SuperOffice, integrated SS2 and Superbase 2, you can easily program an opening menu like this for SS2. An applications generator can be very useful for setting up structures of programs that the user likes. You can not affect the operation of SS2 from the programming mode once you have entered SS2.

Wordstar has seven menus. These are the opening, main, block, onscreen, quick, print and help menus. Except for the main menu they are self explanatory. The main menu allows for basic text editing and cursor movement. Wordstar has four levels of help that vary from all commands displayed to none. These are very useful and can be changed quickly from within the program.

Superscript has four menus and with Superoffice it is unlimited as determined by the application. The menus are the configuration, editing, dos support and the spell check menus. Superscript does not have a built in help feature. If you have more than 128k, you can use an extra bank to store the SS2 reference file. This will give you instant access to the command structure in another bank. In Super Office you can set up a help feature in either a database format or a memo format.

Upon entering the Wordstar editing mode, there are several noticeable differences compared to SS2. Firstly, Wordstar was the first program of the WYSIWYG type word processor. This stands for "What You See is What You Get!". Therefore the format you see on a Wordstar screen is the format that you will print. In other words, you do not have to reformat the document to see what it will look like because it is always the same. This is very convenient because you always know the structure of the document. No surprises!

The second item that will be very noticeable is the user interface. Unlike SS2 and most other major programs, Wordstar does not use the first letter of the command to simplify editing. This is a touchy topic with most serious users. Either you love it or you hate it. The interesting thing about this approach is the method which they have used to substitute for so called easy to remember command letters.

The Wordstar approach is to use a diamond configuration of keys above the left hand. These keys have nothing to do with the actual commands. The letters, for the most part, do not correspond to anything logical. The advantage of this feature is the user can edit by feel, not by looking for the correct letter designation. Hence, most commands can be entered by the left hand rather readily. What I have found interesting is that I could not tell you the commands without really thinking about them. When I get to the keyboard they are second nature.

To illustrate this structure, the prime keys for movement around the keyboard are e,s,d,x. To move up a line you type <ctrl e>, down a line <ctrl x>, right a letter <ctrl d> and left a letter <ctrl s>. If you look at your keyboard, you will see the logic and speed that this method can develop over most other methods. Most software developers have a knack for finding a different word for the same command. It can be difficult to remember if you are using many programs.

Several of the heavy weight programs such as Dbase II, Dbase III, and Supercalc use this type of command structure. Most other good programs have macro capability to imitate the interface. This type of compatibility is a blessing in the crazy and inconsistent world of computers.

Document handling is done very differently between Wordstar and SS2. SS2 handles the whole document in ram. This of course makes the access to any particular part of the document quick. SS2 also allows up to nine text banks to work with. Unfortunately you can not swap material back and forth between banks <<except by ranging a block and temporarily saving to disk>>. SS2 does allow you to link files to make larger documents. Having that many document banks open allows for a lot of interesting concepts. With Superoffice you could have mail merging in some banks, text editing in others or even direct printing to measured or labeled blocks in an other.

If you added a print buffer, you could do all of the

above at the same time. This would depend on how much data was dumped to the buffer at a time. The print buffer allows data to be sent to the buffer and stores it until it reaches the capacity of the unit. This is a real time-saver for short to medium letters or reports. As soon as the buffer has downloaded the text or report, it releases the computer for other tasks.

Wordstar is disk intensive when working with files. It loads only a small amount of text and saves it as it is edited. This of course is slower than a ram based program like SS2. The only advantage this method has, is the document length is virtually unlimited. A ram based program, is limited to not only the amount of available ram but also to power failures and accidental erasure. Wordstar will not allow you to erase or abandon a file, without sufficient effort. Wordstar also makes a copy of your original file when you edit that file again.

This ends part one of this product comparison. It can be seen that each product has certain features that are desirable and some that are not. What it really comes down to are the features that are best for the application desired. Only the user can determine this.

CP/M-86 SOFTWARE REPORT

Note: the following is duplicate of some information found in the library section this issue. However the comments and other information, since this is the first exposure of what is being done to bring CP/M 86 to CBUG, I thought ought to be in the main body of THE ESCAPE. These are excerpts from Lt. Col. Wright's cover letters. This information also appears on the disks Pre 8, 9, and 10.

Introduction to CP/M-86 Files

By: Lt. Col. John A. Wright

The files on this disk are in many forms. First the file naming convention that I have used. It is standard throughout the libraries that I have accessed. My files are set up as follows:

File name.type file.generation.

For example 80t86.lbr.1 is the cpm80 to cmp 86 translator. The "lbr" extension means that it is a library type file (several files stored as one). The "1" is the generation number of this file meaning that it is the first of this file. As newer versions become available I will change the generation number.

Now for some "file type" definitions:

LBR	Library Files as described above. (use delbr.com to break apart)
COM	Executable program files
CMD	A command file of some sort. Usually used by another program but not always.
ASM	An assembler file
A86	8086 assembler (when available I will include these along with the COM files.
HLP	Text files that should be readable with SSII. Some are "verry" long and you may have to read with a seq reader like Liz Deal's.
XQX	The "Q" in the middle means the file has been "squeezed" to save disk space. (Use usq.com to "un-squeeze" these)
DOC	A file with instructions for running/loading a file.
SII	Superscript files.
Z80	These appear to be assembled files. You cannot read them with SS2 but they can be read with a seq reader like the one found in Kernigan's Utilities from CBUG.

That's all there is. Sorry I can't guarantee any of these will run. I know they will run on other machines. If I can be of further assistance, let me know. I can provide de-librariied and unsqueezed files if you need them.

This is just the first of many disks. I will not include the dlbr and usq on future disks unless they change or you need them.

NOTES:

1.) Although all the seq files were stored as 7 bit files (ascii/seq) files on the main frame, they cannot all be read with SS2. They force SS2 into the spell checker. Use a seq reader like the one found in Kernaghan's Utilities from CBUG.

2.) <<It is reported that these programs will require one additional level of transformation before they can actually be run on a co-processor equiped B128 or B256 (latter preferable). The files now readable in the standard B's must be ported in Commodore mode from the B to an IBM or clone, then back to the B with the coprocesser installed and running in CPM mode. The destination disk carries a 1 block loader program to put the co-processer in charge. The 6502 then runs as a slave to the coprocessor. Any disks produced in this manner, will show a directory of only one, one block program and zero blocks free. The actual directory will be available only under CP/M 86, etc.>>

```
0 "cp/m-86 Pre 1     " 01 2c
3    "to-norm.sii.1"    seq     Ltr to CBUG
7    "guide.sii.1"      seq     Disk explaination
5    "directory.sii.1"  seq     This file
52   "delbr.com.1"      prg     De-lbr routine
8    "usq.com.1"        prg     Unsqueeze routine
526  "catalog.doc.1"    seq     Other CP/M avalibility
63   "apc-dump.lbr.1"   prg     NEC APC dump pgm
44   "bidump86.lbr.1"   prg     Bidirectional dump
6    "bishow86.cmd.1"   prg     Bidirectional show
46   "bishow86.lbr.1"   prg           /
37   "ltype86.lbr.1"    prg     LTYPE for CP/M 86
74   "lu86.cmd.1"       prg     Library utility
156  "lu86-2.lbr.1"     prg           /
57   "print86.lbr.1"    prg     Print utility
27   "typesq86.lbr.1"   prg     Typesqueeze
457  "80t86.lbr.1"      prg     Convert CPM80 to CPM86
101  "ffynde.a86.1"     seq     FIND - with the
92   "ffynde.asm.1"     seq     capability of gener-
18   "ffynde.cmd.1"     prg     ating a cross ref.
16   "ffynde.com.1"     prg     listing.
162  "cnvadv.hlp.1"     seq     Adv help file
72   "usq.cmd.1"        prg     Unsqueeze squeeze files
16   "sq.cmd.1"         prg     Squeeze files
7 blocks free
```

```
0 "cp/m-86  PRE 2    " 02 2c
2    "to-norm.sii.2"    seq   Ltr to CBUG
9    "guide.sii.2"      seq   Guide for this disk
5    "directory.sii.2"  seq   This file
38   "modem-86.doc.1"   seq   Section of pgms to
163  "modem9xx.doc.1"   prg   implement the term MODEM
100  "sendrecv.a86.1"   seq   on CPM-86 machines.
62   "term.a86.1"       seq                 /
10   "modem.a86.1"      seq                /
24   "start.a86.1"      seq               /
36   "siosys.a86.1"     seq              /
100  "main.a86.1"       seq             /
9    "finish.a86.1"     seq            /
44   "files.a86.1"      seq           /
8    "deff.a86.1"       seq          /
47   "console.a86.1"    seq         /
34   "8251sys.a86.1"    seq        /
33   "2651sys.a86.1"    seq       /
500  "mdm840.aq6.1"     prg   MODEM(840) Section
105  "mdm840.dqc"       prg   Documentation File
102  "mdm840.hq6.1"     prg   Hex file
6    "mdm840.nqt.1"     prg   Translator's Notes
```

```
72   "m81ib.aq6"          prg    Auto Dialer
14   "m81ib.cmd.1"        prg         /
3    "m81ib.doc.1"        seq        /
16   "m8nm-40.aq6.1"      prg       /
4    "f47-test.a86.1"     seq    DEMO of CPM86 BDOS
3    "f47-test.cmd"       prg    Function 47
39   "z80.a86.1"          seq    Working CPM80 emulator
21   "z80.cmd.1"          prg    for CPM86
21   "cbaddr.z80.1"       seq          /
63   "code8086.z80.1"     seq         /
1    "codecb.z80.1"       seq        /
19   "codeed.z80.1"       seq       /
8    "edaddr.z80.1"       seq      /
21   "jumpaddr.z80.1"     seq     /
223  "cp4ker.doc.1"       seq    Kermit DOC for CPM 3.0
55   "cp4ker.com.1"       prg    Kermit pgm for CPM 3.0
15 blocks free.

0 "cp/m-86  Pre 3  " 03 2c
2    "to-norm.sii.3"      seq
10   "guide.sii.2"        seq
7    "directory.sii.3"    seq
317  "encol.lbr.1"        prg    multiple column listing
235  "hjelp.lbr.1"        prg    help files, even squeezed,
                                   within a library
78   "qsort.lbr.1"        prg    quick sort to order the
                                   lines of a file
251  "ty.lbr.1"           prg    type, whether binary,
                                   squeezed or library
78   "apc-caln.lbr.1"     prg    prints calendars - APC &
                                   MX printers
74   "apc-date.lbr.1"     prg    screens calendars on APC
64   "apcserio.lbr.1"     prg    interrupt driven I/O for APC
85   "bytype86.lbr.1"     prg    DIR sort by types to con,
                                   lst or disk
333  "fastvf86.lbr.1"     prg    cleaner faster VFILER. CRC
                                   optional
39   "frag86.lbr.1"       prg    file-ext cleaner, faster,
                                   sorts by files
96   "grab86.lbr.1"       prg    finds paragraphs
31   "save0-86.lbr.1"     prg    emulates CP/M 2.2 SACE 0
                                   command
36   "wc86.lbr.1"         prg    word count, text & WS files
                                   to 8 digits
44   "bidump86.lbr.1"     prg    CP/M 86 bidirectional dump
6    "bishow86.cmd.1"     prg    Improved CP/M 86
                                   bidirectional SHOW
46   "bishow86.lbr.1"     prg          /
37   "ltype86.lbr.1"      prg    LTYPE for CP/M 86
57   "print86.lbr.1"      prg    CP/M 86 print utility
27   "typesq86.lbr.1"     prg    CP/M 86 typesqueeze
99 blocks free.
```

OF THE PROGRAMS FROM ART CHICK

By: Arthur T. Chick

This miscellaneous group of programs have been developed over the past 5 years when I ran into a problem that needed to be solved. Generally I was too lazy to do it the hard way (pencil & paper) more than once, so I worked up these programs.

Program #1 Is an amortization program. It was designed to give a quick answer for the payments on a potential home purchase. The program will ask for the purchase price, down payment (as a % of the purchase price), the interest rate, and the length of the loan. It will then calculate the balance to be amortized, the payment, and the amount on principal and interest. You may also have a hard copy if you want. This program is formatted for the Epson MX100 printer

Program #2 Will calculate calculate the monthly payments on a fully amortized loan. Display/print the name of the lender, the interest rate and the payment, and then display/print an annual schedule of payments and balances for a full year, total amount applied to the principal, and the total amount of interest paid. The program will ask if you want the next year. This program is formatted for an Epsom MX100 printer.

Program #3 is the identical program formatted for a Commodore 8023P printer. --

Program #4 came about when my wife decided on a BIG redecorating project involving painting and wallpapering. I got tired of doing all the calculations by hand and wrote this program to help me find out how many gallons of paint, and rolls of wallpaper I would need. Unfortunately, it will NOT measure your rooms for you, but it will take your measurements in feet and inches.

Program #5 I work with mechanical equipment and need, from time to time to calculate how big a pulley would have to be to turn a pump at a given speed with a fixed motor speed, or the other option if any three of the four variables are known. It will also determine the belt size if needed.

Random comments... Since these programs were originally written on a CBM 8032 feeding an Epson printer, I did not have the luxury of printusing commands so I worked up my own decimal align program based on several ideas found in Compute magazine and other areas. I combined those bits and pieces with some of my own trial and error work and came up with a decimal align system that works. It will handle trailing zero's and will round up to the same degree of accuracy that my bank does on my mortgage payment.

If you find any of these programs of value, send a contribution to CBUG for me. Norman & CBUG need all the help and support we can give.

If you have any question, or find any bugs, please let me know, or call me.

Arthur T. Chick
P.O. Box 626
Freedom, CA 95019
(408) 722-9114

PS: The disk is formatted to print on the Epson printer. The above programs are on CBUG #66

SPECIAL CHARACTERS ON 4023 & 8023 PRINTERS

A COMMENTARY ON THE WORK OF COL. J.E. O'HALLORAN

The next page shows a self explanatory printout of what one can do if they really do read, and then read some more. The disk itself is a must (CBUG #66) as we can not reproduced all the symbols available on a screen. With everything already coded, one need only "borrow" the escape and secondary codes for any other project.

What is being shown here is the alternative to the "format" command which wasn't in Superscript II. The process is identical in using other printers such as Epson, Star, etc. Only the codes change. And, incidentally, these commands continue to be operable in the 8023 condensed mode!

Passing comment. There must be atleast one character sent to the printer BEFORE escape codes are sent. It is wise therefore to put a lone Carriage Return near the very top of the page, else the codes will not be invoked till the second copy run.... Escape codes can be changed at will during a text, so there is no limit as to how many can be used.

COL. J.E.O'HALLORAN
HORSAO FARMS
RT.2 OWL CREEK ROAD
HIAWASSEE,GA. 30546

30 June 1987

Dear Norm,

Are there any B128 users who would NOT care to be able to create the following symbols on their CBM Dot Matrix Printers?

[£] ↑ ← ¯ _ | |

Check YOUR answer: I would like ✓ I would NOT like ___

DON'T BE RIDICULOUS ! OF COURSE YOU WOULD.

Here are ten symbols which we have been told by the printer manuals cannot be reproduced on the 4023 or the 8023P BUT I am including a disk formatted in SUPERSCRIPT II which, if used in formatting a document, will print them.

There are more that may be used when set up in the same manner using the correct CHR$ codes but these are the ones I was interested in so I stopped with them.

The 'utility' on the disk is called XTRA symbols.

Load it by that name at the top of your format then add your other desired formatting commands and SAVE under a file name of your choice for future use. It can be in every format you use as it takes up space ONLY on the screen and is inert until called upon by keying in ESC#.

The 'keys' to which is which symbol are as follows:
1=lb=[; 2=bs=£; 3=rb=]; 4=ua=↑; 5=la=←; 6=ul or tn=¯ ,underline or top horizontal; 7=lh=_; 8=lv=| ; 9=rv= |; 0=ck=✓.

To insert into text, press ESC # at the cursor position where the symbol is desired. The reversed # will be displayed on the screen BUT the symbol will be printed.

Here is to more satisfactory documents with you B128s/4023s or 8023Ps.

Jim O'Halloran

After you have read the information on the lines below---place the cursor on the line BELOW the COMMANDS and press ESC E R to erase all from cursor down
Then enter YOUR format commands in place of those in top two lines.

Here are ten symbols which we have been told by the printer manuals cannot be reproduced on the 4023 or the 8023P BUT I am including a disk formatted in SUPERSCRIPT II which, if used in formatting a document, will print them.

There are more that may be used when set up in the same manner using the correct CHR$ codes but these are the ones I was interested in so I stopped with them.

The 'utility' on the disk is called XTRA symbols.

Load it by that name at the top of your format then add your other desired formatting commands and SAVE under a file name of your choice for future use. It can be in every format you use as it takes up space ONLY on the screen and is inert until called upon by keying in ESC#.

The 'keys' to which is which symbol are as follows:
1=lb=[; 2=bs=£; 3=rb=]; 4=ua=↑; 5=la=←; 6=ul or tn=¯ ,underline or top horizontal; 7=lh=_; 8=lv=| ; 9=rv= |; 0=ck=✓.

To insert into text, press ESC # at the cursor position where the symbol is desired. The reversed # will be displayed on the screen BUT the symbol will be printed.

Here is to more satisfactory documents with you B128s/4023s or 8023Ps.

Jim O'Halloran

THE CBUG YELL FOR HELP DIRECTORY

Following is a listing of members who have volunteered to give of their valuable time to aid other membmers in specific areas of B128 applications. Threre are a few rules which need to be followed carefully:

1.) Thou shalt not call collect;
2.) Thou shalt not ask helpers to call back unless you instruct them to call you collect;
3.) Thou shalt follow time restrictions listed, and when not, use exceptional judgement, making sure of time zones FIRST.
4.) Thou shalt have all materials at hand when calling, and if possible be infront of your machine if applicable.

NOTE: To our helpers, please double check your listing and advise of errors. Unfortunately we lost the file of unentered updates for this listing. Anyone else wishing to join the Yell for Help Group is invited to send a note with name, address, phone(s) hours and expertise. Please mark the envelope in the lower corner "Yell For Help."

zip	expertise	member name	main phone & hours		additional phone number & hours
03264	sbp	Glen Van Valkenburg	603 536 1025	7pm to 10pm	
06790	sbuP	Armand Carrier	203 489 0184	1pm to 2am	
08043	bp	Michael F. Gullo	609 768 4789	9pm to 11:30pm	
10010	P	Benson Greene	212 683 6906	Noon to 10pm	
10128	S	Anthony Liversidge	212 534 7371		
10467	p	Angel Matos	212 231 6028		
11229	s	John Francis	718 376 9269	7:00 to 10:00 pm	
11729	SBPml	George Kowalski	516 667 7076	7pm to 10pm	
12065	pd	Mathew Goldstein	518 383 0067	8:00 to 9:30pm EST	
13045	B	Paul J. Comfort, Jr.	607 753 8433		
17315	sPmt	John Lemkelde	717 292 4933	5pm to 10pm	
17331	sIpmt	Steven J. Greenaway	717 637 2453	7:00 to 10:30pm	301 890 3878 7:00 to 10:30pm
21801	SUPmt	Leonard (Len) Kloft	301 742 2373	7pm to 10pm Th/F	
28533	sbpmTl	Rick DeGraffenreid	919 223 5765	8:00 to 11:00pm	
31313	sBp	Clyde M. Northrop	919 368 6712	7pm - 10pm	
32405	sbp	Bill Hammack	904 763 8808	8am to 10pm	
35136	PS	Allyn Uptain	205 377 4476		
44106	pml	Allen E. Tracht (='tract')	216 932 2754	9pm to 10pm	
45805	ui	Ron W. Burkholder	419 331 1719	6:00 to 12:00pm	
45849	sp	Dale K. Elston	419 587 3804	7pm to 11pm	
46013	sI	Dan R. Schoger	317 649 8364	reasonable hours	
48827	p	Albert Meinke, III, M.D.	517 663 4994	7pm - 11pm	
50662	u	Randy Wilbur	319 283 1422	8am to 5pm	
53081	U	Carter S. Pawlus	414 457 6100		
54235	SBup	Mark Schwarzbauer	414 743 4757		
54304	sBP	Mickey Crittenden	414 494 3142	5pm to 9pm	
55108	PMv	Charles A. MaCarthy	612 645 6867	7pm to 10pm	
59872	p	Dick Wilkinson	406 822 4989	5pm to 11pm	
60077	SBIP	Roy Sherman	312 673 5094	9am to 9pm	
60188	XMPSTCbhu	Warren D. Swan	312 665 1514	7:00 to 10:00pm	
60202	sPxEh	Marilyn Gardner	312 866 9159	7pm to 9pm	
60634	Sit	Norman Deltzke	312 456 8720	7pm to 10pm	
60643	sbut	Eric L. Watkins	312 734 0312	6pm to 10pm	
61015	bUp	Gerald Beck	815 732 7387	7pm to 10pm	
64055	PL COMPILERS	Bing Hart	816 373 5523	3:00 to 7:00pm	
64683	bipv	Marvin V. Pinnick	816 359 2138	9:00am to 9:00pm	
67337	bp	Tex Davis	316 251 5623	6:00 to 10:00pm	
68046	sbpl	Lt. Col. John Wright	402 339 5728	2:00 to 7:00pm	
68788	i	Ron Meyer	402 372 5785	7pm to 10pm	
74126	?	S. Ray Lohman	918 425 0669		918 425 3660
75040	bPMX	Richard H. Wood	214 530 2595		
77063	sPtl	Mauricio J. De La Torre	713 953 9249	5pm to 10pm	
78210	b	Francis Martin	512 534 1400	noon to 9:30pm	
78504	pml	Robert Hargrove	512 686 5219	noon to 10pm	
83536	Sbp	Rodney Jay Lillibridge	208 935 2962	5pm - 8pm	
83605	SbPM	David L. Evans	208 459 3279	5:30 to 7pm	208 454 8421 BBS 7pm to 10pm
89121	SuPMl	Ron W. Hardy	702 459 4964	Fri 10pm to Sun 4p	
91733	PmX	Jerry Bailey	818 448 8351	7pm to midnight	213 598 7661 9am to 4pm
92056	spv	Lt. Col. G. A. Carlson	619 726 3219		
93423	sBuPm	W.K. Vance	805 466 5123	10:00am to 4:00pm	
98032	st	Dan Gayman	206 878 8783	7pm to 10pm	
98198	st	Dan Gayman	206 878 8783	7pm to 10pm	
B2G 1C	sbptL	Michael Steinitz	902 867 3909	9am to 6pm AST	
V8X 3P	sPMt	Russ Beinder	604 479 8510	7pm to 11pm PST	
V9Y 1X	cP	Steve Pearson	604 123 3231	9am - 10pm	

Expertise codes are as follows. Lower casase indicates the helper has given themself a rating of "pretty good", Capitals indicate an Expert rating:

a=all	p=basic language programming	c=general business	h=math
s=Superscript	m=machine language programming	h=accountant/ing	w=law
b=Superbase	x=expert programmer	d=tech/elect. eng.	f=financing
i=CABS/Info Designs	l=laboratory instrumentation	z=repair service	e=education
u=Calc/Result	v=interfacing, hardware, etc.	t=telecommunications	n=CMS/Southern Solutions

the B128. The CABS suite is a traditional double entry cross footed accounting system -- a typical example of small business accounting suites. Don't miss this meeting!

Dan Schoger is a CPA practicing in Anderson Indiana, some six hours drive south of Chicago. The attendance and interest in Dan's presentation at this meeting will determine the further frequency and content of accounting matter for CBUG East meetings -- and his sojourns to Evanston.

If for any reason you are unable to attend but would attend future meetings on accounting programs, please contact Marilyn Gardner with your rsvp.

==========

CBUG WEST SCHEDULE

CBUG West meets at the First Congregational Church of West Dundee, 5th and Main Streets (S.E. Corner), W. Dundee, Il. at 7:30pm on the 2nd Monday of each month. Questions regarding the meetings should be addressed to Warren Swan, 312 665 1514 or Mike Wodrich 312 931 5293. Weather cancellation info only also try Herb Gross 312 695 1316. Dundee is the first town north of Elgin Ill.

A NEW DISK REVIEW PROCESS

Marilyn Gardiner has generously taken on a new assignment of librarian in as much as Mark Schwarzbauer has become badly overloaded with our overseas and some other matters. At the same time, Warren Swan suggested that many of our local members could do an admirable job of reviewing (and occasionally repairing), organizing, etc. of the incoming and existing library materials. Therefor, subject to discussion and atlarge endorsement the next several CBUG WEST meetings will begin the process of getting existing library materials to volunteering members within and without of the local groups, etc., this appears to be a most interesting opportunity for some of the local members. One of the benefits is that attendees at a meeting are entitled to purchase the day's topic disk(s) for $1.00 plus applicable royalties, and those members taking on review work would get it along with necessary blanks free. Our Sept. 14 meeting under this format was a roaring success.

INTERESTING FACETS OF 'HELPING'

Nearly two years ago I received a plaintive call from a lady who just could not get her new B-128 to cooperate in writing a book. A book about herself and her pet. Such a pet; such a lady in distress. And residing way out in rural Oregon where shoes to be lost had not yet been invented; there just were no other members in reasonable distance to give hands on help.

She'd purchased a typewriter printer along with her B system and they were as cooperative as untaimed big cats. Well, many conversations later and far more effort on the lady's part than ours, pagination was had.

The other day I received a note from Dee, with a portion of the above picture adorning the stationary entitled "Epistle from BC and Me". Folks, helping others get un-ensnarled is really most rewarding, often most interesting such as in the case of Dee and her pet tigress!

That's what the Yell for Help section is all about. If you have some skill you can share, join up. With the large number of members in Yell for Help now, the load stays lightly spread around. It's really most rewarding.

P.S. Dee has raised BC as a house pet since she (BC) was 11 weeks old. BC is now 15 1/2. Can you comprehend the magnitude of of such a project?

Please repeat your name and zip code per chance your order sheets are separated:

Shipping Name: _____ Shipping Zip Code: _____

Description	Quantity	Stock #	Price	Extension
RR1 Norm's Utility v1.2		12862	9.00	
CBUG #3 Swan's Utility #1		12881	14.00	
CBUG #6 CBUG/TPUG #1		12913	9.00	
CBUG #47 dFile database pgm--Available ONLY to US members		11856	16.00	
CBUG #51 JCL Work Shop & Assembler -- 2 disk set		11894	29.00	
CBUG #7 Northrup's **Superbase** Applications		12932	9.00	
CBUG #13 **Superbase** tutorial pgms & Leighfield aids texts		12787	9.00	
CBUG #31 **Superbase** Corner & Hints		12538	9.00	
CBUG #15 Friendfam (**Superbase** application pgm)		12716	14.00	
CBUG #33 Medical Accounting (**Superbase** application)		12540	9.00	
CBUG #49 Medical Finance #2 (**Superbase** application)		11875	9.00	
CBUG #50 Educational Records (**Superbase** application)		11889	9.00	
CBUG #8 Sermons		12946	9.00	
CBUG #9 CABS GL pro forma #1		12951	9.00	
CBUG #11 **Terminal** Pgms w/ BTerm		12257	14.00	
CBUG #11a **Terminal** Pgms w/o BTerm		12261	9.00	
CBUG #12 Scott's B-Mon		12984	14.00	
CBUG #16 Swan's Basic Course		12773	19.00	
CBUG #17a Liz's Utility v1.2a		12670	16.00	
CBUG #18 Games and Education		12792	10.00	
CBUG #19 Old BUG texts and programs		12805	9.00	
CBUG M20 CBUG Utilities etc #2		12768	9.00	
CBUG M45 CBUG Utilities & Misc. #3 (mislabeled #2 fall 87)		12837	9.00	
CBUG M54 CBUG Misc. #4		11930	9.00	
CBUG #58 Dittinger's Utilities +		11925	12.00	
CBUG #21 Retail News Distribution pgm		12699	9.00	
CBUG #22 Math Education Programs		12701	9.00	
CBUG #23 Bible Games		12735	15.00	
CBUG #24 8432 Emulator Disassembled		12720	9.00	
CBUG #27 Goceliaks Gold Mine - disk utilities/engineering		12492	9.00	
CBUG #57 Goceliak Strikes Again		11963	9.00	
CBUG #28 Casey's Scrubber		12504	19.00	
CBUG #29 CBUG TPUG P1 & P2		12519	9.00	
CBUG #56 Harrison's Assembler 5.5 v8 (revised)		11959	35.00	
CBUG #32 Kernaghan's Utilities v3		11536	10.00	
CBUG #36 London Sampler		12561	9.00	
CBUG #37 SUPERPRINT		12576	19.00	
CBUG #40 Public Domain Math A		11771	10.00	
CBUG #41 Public Domain English A		11785	10.00	
CBUG #42 Public Domain GHBT		11790	10.00	
CBUG #43 Public Domain Science A		11803	10.00	
CBUG #44 Public Domain Science B		11818	10.00	
SET of 5 Public Domain CBUG #40 thr #44 inclusive		12822	45.00	
Physical Exam for the 1541		12223	35.00	
Physical Exam for the 4040		12238	35.00	
Physical Exam for the 1571		12242	35.00	
CBUG #48 CBM Diagnostics adapted for the B128		11860	9.00	
CBUG #M55 ML Programming Information		11944	9.00	
PR1 Pre Release #1		12824	9.00	
PR2 Pre Release #2		12839	9.00	
PR3 Pre Release #3		12843	9.00	
PR4 Pre Release #4		12749	9.00	
PR5 Pre Release #5		12542	9.00	
PR6p Pre Release #6 partial		12557	6.00	
CBUG #10 Fall 1985 ESCAPE and prior files - disk		12965	9.00	
CBUG #25 Winter/Spring 1986 ESCAPE print files - disk		12665	9.00	
CBUG #26 Jan. 1986 Telecom issue and CBUG #25 overflow		12651	9.00	
CBUG #38 Summer part 1 1986 ESCAPE print files - disk		12580	9.00	
CBUG #52 Summer part 2 1986 ESCAPE print files - disk		11906	9.00	
CBUG #53 Fall 1986 ESCAPE print files - disk		11911	9.00	
CBUG #59 Winter/Spring 1987 ESCAPE print files - disk		11978	9.00	
NOTE: Fall '85 ESCAPE and Jan '86 Telecom issues are out of print. Substitute print file disks.				
Winter/Spring 1986 ESCAPE, copy of publication		12449	6.00	
Summer 1986 ESCAPE Part 1, copy of publication		12468	4.00	
Summer 1986 ESCAPE Part 2, copy of publication		12473	3.00	
Fall 1986 ESCAPE, copy of publication		12346	5.00	
TOTAL THIS PAGE ---- Extend to main order form				

NON-DISCLOSURE AGREEMENT AND LICENSE

re: Proprietary Code

As a condition of receiving copies of CBUG distributed code such as Source code or any product containing portions thereof (such as stock #'s 11747, 11752, 11766, 11588 etc. but not limited thereto currently nor in the future), I agree and represent as follows without limitation or qualification:

1.) I am a member in good standing of the Chicago B128 Users Group International (CBUG, Inc.);

2.) I will use the materials requested to enhance my and/or other CBUG members' usage of the B128 series of computers and accessories, and for no other purpose without exception;

3.) I will not disclose, divulge, copy, or communicate any portion of the contents of any CBUG product identified or which I have reason to suspect contains code made available to CBUG by CBM or others which is not intended for general distribution; such disclosure being prohibited in any medium used including but not limited to: speach; accoustic, optical, electronic or magnetic transfer; recording by any method; written or printed; without limitation;

4.) I will cause a signed copy of this agreement to be delivered to CBUG prior to any further disclosure or release of these informations as permitted supra;

5.) I understand that the subject documents, code, programs and related information may be copyrighted and or held as trade secrets by or proprietary to Commodore Business Machines and/or their vendors; that the sole purpose of the provision of these informations by CBM to CBUG for benefit of its 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 party. I agree that there are no exceptions to these rights and waivers.

6.) I shall not disclose information from, regarding, and or resulting from my work product regarding the subject materials except to and thru CBUG and those members authorized to receive such informations

7.) I understand that violations of laws protecting intellectual properties such as those properties refered to herein hereto may result in civil and/or criminal prosecution; that should I breach this agreement I will hold CBUG, its officers, members, agents and assigns harmless from any damages direct or consequential resulting from my action.

8.) I understand and agree that all materials acquiried by me under this no cost license and do not constitute a property right or right of possession of any kind.

Name _____ Please print or type

Address _____ Phone _____ _____

City _____ State _____

Postal Code _____ Country _____ _____

X _____ Date _____

Signature