

The Unofficial OSI Users Journal

P.O. Box 347 Owings Mills, Md. 21117 (301) 363-3268 \$1.75 MARCH 1984 VOL.5, NO.3

INSIDE

OLD'FOR OSI BASIC-IN-ROM 2 INSTALL NON-OSI 40 TRK DISK DRIVE 3 EXPAND CIP/SBII PART 3 XREF: BASIC FILE CROSS REF. GEN. 8 CIP CORNER 12 NEW CHALLENGER PERS. COMP. 16

Column One

This column has lately begun to look like the Isotron, Inc. PR department. Please be assured this will not long be the case. However, since we are all very curious as to the future, survival, plans and prospects of OSI/Isotron, for the nonce we will continue reporting what we hear. Herewith this month's batch of announcements and information:

During the month of March, two new machines will be announced as a part of the national advertising campaign which will soon start: each will be a 3user machine, one running OS-65U level 3, the other a multiprocessor Turbodos machine. Old Peekers will remember that we very much like the idea of multiprocessing since it means that each user has his/her own CPU, sharing only the expensive stuff like hard disks and printers.

Both these machines will run DMS. There will also be other software bundled as is the practice of the industry today. However, these machines will go beyond the usual practice by bundling hardware as well as software. The very reasonable prices will include one terminal and a letter quality printer as well as the software. The prices are not finalized yet, but should be very competitive.

In case the national ad campaign isn't enough to stir up some interest, we are told Isotron will be at Comdex/ Spring with a 35 ft (1) booth. That should get some attention.

. . . . · ·

So what, you ask, does all this have to do with your ClP? Simply that if the company had folded, you would be left to your own devices (and those of PEEK(65)) for support and encouragement. With the company still in business, there is at least some support and assistance from headquarters.

Now to the stuff I really like to write about, what is in this month's issue, and what it means.

If there was ever any doubt that PEEK(65) is the hackers' bible, this issue should put it to rest. There is hardly a line here that other magazines would publish, with a few notable exceptions. "Too technical," the editors would say:"Too limited in audience." Translate that to mean that if 300,000 mindless video-game players don't understand it at a quick scan, it won't sell enough copies of the magazine to print it.

Here at PEEK(65) we feel differently. We can read various other 2.7 lb. computer magazines for non-technical information about computers we don't own (and do read them!), but our particular corner of the information business is to spread technical information among users/enthusiasts with OSI computers. We have fun doing it, and make our living at other things... It has not always been so. For a long time there, we tried very hard to follow the lead of the company and become a business-computer magazine. However, you our readers told us by your response to our calls for articles and your letters that you were more interested in the technical stuff. So be it. We will publish what you want to read!

Of course, that means if you change your collective minds and decide you want more articles on business systems, we will print them. In short, we are running a service here, and will print what you want to read. This month, obviously, the interest is in hardware articles and program listings, the kind of stuff you just won't find anywhere but good ole PEEK(65).

If you disagree with this mix of articles, write to us. We are at your service, and will print what you want to read.

al.

'OLD' FOR OSI BASIC-IN-ROM

By: L. Z. Jankowski Otaio RDI Timaru New Zealand

'OLD' enables the recovery of BASIC programs which have been inadvertently NEWed, or lost as a result of a crash of zero-page. 'OLD' can be placed in BASIC 4, or alternatively can be loaded and run in RAM.

Ever typed NEW and wished you hadn't? Or, POKEed into RAM and wiped zero page? No need to despair, use 'OLD'!

'OLD' is a machine language program that can be placed either in RAM, or in EPROM in a new BASIC 4. (See listing). If the former choice is taken, routines STORE and CHECK are not required.

BASIC 4 code is full of superfluous messages and contains code for running a non-exis-tent serial port! Available RAM that could be put to better use ranges from \$BE39 to \$BF2C. Some of this RAM space is required for the Cold Start messages which if shortened, add to available RAM. Over 200 bytes are freed! See PEEK(65), Aug. 82 issue, for a fuller explanation.

'OLD' is simple to use. Tf recovery from NEW is required, enter the Monitor and run the code from the point at which the routine RUNOLD begins. If zero-page has crashed and 'OLD' is in BASIC 4 then do the following: COLD start and type OLD in response to MEMORY SIZE. Answer TERMINAL WIDTH as desired and the BASIC program is ready for LISTing.

If the code for 'OLD' has been MEMORY SIZE with a number: 8192 for 8K of RAM, 16384 for 16K of RAM, etc.. Then, enter the Monitor and run the code which begins with the routine

Copyright @1984/by PEEK (65) Inc. All Rights Reserved. published monthly Editor - Al Peabody Technical Editor - Brian Hartson Circulation & Advertising Mgr. - Karin Q. Gieske Production Dept. - A. Fusselbaugh, Ginny Mays Subscription Rates US (surface) Canada & Mexico (1st class) \$23 \$35 So. & Cen. America (Air) Europe (Air) Other Foreign (Air) \$35 \$40 All subscriptions are for 1 year and are payable in advance in US Dollars. For back issues, subscriptions, change of address or other information, write to: PEEK (65) P.O. Box 347 Owings Mills, MD 21117

Mention of products by trade name in editorial material or advertisements contained herein in no way constitutes endorsements of the product or products by this magazine or the publisher.

20,0000 by L.Z. JANKOWSKI. 30 0000 40 0000 50 0000 70 0000 80 0000 90 0000 100 BEA4 110 BEA4 120 BEA4 130 BEA4 140 BEA4 85F0 150 BEA6 A900 160 BEAB A080 170 BEAA 4CBABD 180 BEAD 190 BEAD ASFO 200 BEAE C94E 210 BEB1 D035 220 BEB3 230 BEB3 A900 240 BEB5 85F3 250 BEB7 6903 260 BEB7 B5F4 270 BEBB A005 280 BEBD 290 BEBD 20EBBE 300 BEC0 F002 LOOP 310 BEC2 DOF9 320 BEC4 330 BEC4 CB ÁDD 340 BEC5 BC0103 350 BEC8 A5F4 360 BECA BD0203 370 BECD . 3BO BECD BR 390 BECE 88 400 BECF 20EBBE LOOP2 410 BED2 DOFB 420 BED4 20EBRE 430 BED7 DOF6 440 BED9 20EBBE 450 BEDC DOF1 460 BEDE 470 BEDE 98 FOLIND 480 BEDF 18 490 BEE0 6902 500 BEE2 8578 510 BEE4 ASF4 520 BEE6 857C 530 BEEB 4C74A2 WARM 540 BEEB 550 BEEB CB ŻERD 560 BEEC CO00 570 BEEE DOO2 580 BEF0 E4F4 590 BEE2 BIE3 ONE

10 0000

; Destruction of zero-page contents assumed. ; 32K RAM system assumed. If not, change ; contents of \$85 & \$86 to suit. ; Alternatively, change lines 150 and 160. * = \$BEA4 ZP = \$F0 WSTART = \$A274 STORE STA ZP Jump here from \$BDOA - 'O' hit. LDA #\$00 LDY #\$80 JMP \$808A Set RAM size to 32K, and bypass RAM size check. CHECK LDA ZP Jump here from \$BE36 CMP #\$4F BNE WARM 'OLD' required? No. RUNDED LDA #\$00 Yes, 'OLD' required. STA ZP+3 LDA #\$03 STA ZP+4 LDY #\$05 JSR ZERO Search for first null. BEQ ADD Found it? BNE LOOP Yes. Lo-byte of pointer to next line of BASIC. Hi-byte goes here. INY STY \$0301 LDA ZP+4 STA \$0302 DEY DEV Search for end of BASIC. JSR ZERO signified by 3 nulls. BNE LOOP2 BNE LOOP2 BNE LOOP2 TYA Reset pointers. CLC ADC ##2 STA \$78 LDA ZP+4 End of BASIC pointer. STA \$7C Pointer hi-hyte. WSTART WARM start jump. JMP INY CPY #\$0 SNE ONE INC ZP+4

LDA (ZP+3).Y

RTS

; 'OLD' routine for OHIO BASIC 4.

RUNOLD. Jump to WARM start is automatic. If PRINT FRE(X) is now required, enter CLEAR first to speed up garbage col-

600 BEF4 60

lection.

Placing the program in BASIC 4 requires three changes to be made to BASIC 4 code.

(1) Contents of BD83 are changed from 41 (A for Author) changed from the to AF (O for OLD). If the 'OLD' option is taken, the 'OLD' option is taken, This code branches to BDOA. is where the second change is made.

(2) At BDOA, change A9 4E A0 to 4C A4 BE. Address BEA4 is the entry to STORE. Finally,

(3) At BE36, change 6C 01 00 to 4C AD BE. Address BEAD is the entry to CHECK which, if completed successfully, leads to RUNOLD and complete recovery of the lost BASIC program.

The change in (1) intercepts COLD start. The second change enables the jump to STORE that sets the OLD flag on zero page. The jump from STORE to

\$BDBA bypasses the RAM test which would normally destroy any BASIC in RAM.

The third change is the crucial one. When BASIC arrives, at \$BE36, it has reset all pointers as if there was no BASIC program in RAM. The jump at \$BE36, originally to WARM start, is now intercepted and forced to jump to CHECK. If CHECK discovers that the flag on zero-page has not been set then the jump to WARM start is taken immediately.

If the flag is set, then RUNOLD takes over. The first search made, is for the null marking the end of the first line of BASIC. The second search is for the three nulls marking the end of the BASIC program. When this search is successful, the appropriate pointers are calculated and placed in their addresses on zero page. The BASIC program has now been recovered and the jump to WARM start is made.

'OLD' can only recover 'undamaged' programs. If, on



LIST, you see a screen full of garbage, then you have been successful in not only destroying zero-page contents, but have also run amok in RAM!

INSTALLING A NON-OSI SINGLE SIDED 40 TRK DISK DRIVE ON THE C1P

By: David L. Kuhn 109 Shaw Avenue Lewistown, PA 17044

There are two issues of the now defunct Aardvark Journal, that contain articles on installing non-OSI disk drives on your ClP. For those of you that have all the back issues, refer to the February 1982 and the October 1981 issues. I am writing this article for those who already have an OSI 610 expansion board, or the equivalent. If you do not, buy or steal the February 1982 issue of the 'Journal'. In that issue, there are plans for building a disk controller.

For the last three years, I had already been running a disk drive on my ClP. About a month ago my standard MPI drive started to 'go West' on me. I tried cleaning the heads and correcting the disk rotational speed. That helped but I still ended up with disk errors creeping up on me. 1 figured the drive needed to be realigned. A friend said that he could probably fix it, but until I got the drive to him, I wanted to experiment with other drives. So I picked up an almost new Siemens FDD 100-5B that was left over from a Heath H89 computer when it was upgraded with better drives. The price was right (<\$100) and it is a 40 track drive that is almost OSI-MPI compatible.

The two differences, and they are not small, is that the Siemens drive doesn't have a DATA SEPARATOR and the tracktrack step speed is slower. The latter difference can be handled by changing the software. The first one though, I thought, was a tough one. I remembered that I read an article in one of the issues of the 'Journal' that dealt with adding a 35 track Shugart SA400 drive. My final work is a combination of those two issues and careful planning. These two problems were overcome.

My solution to the DATA SEPARATOR is really independent of what kind of drive we are dealing with. I used a circuit similar to the one shown in the 'Journal'. The two I-C's used are a 74LS221 and a 74H00. The 74LS221 is a dual monostable multivibrator. The circuit uses only 1/2 of the chip. You can use the 74LS121 if you can get it. I didn't have immediate access to one, so I used the dual chip version, which for me was easier to get. NOTE: The pin outs of the chips are different, but function similarly. I used the 'H' version of the 7400 for the same reason.

To build the DATA SEPARATOR, you can wire it together on a piece of small perfboard using wire-wrap wire or some small hookup wire (I used wire split out of a scrapped 40 wire flat computer jumper cable). Miniature push-in terminal pins that you can buy at a Radio Shack store can act as a hookup spots when you are ready to install the board. Double check all your wiring and then you are ready to hook it up!

Find a place fairly close to the Jl connector on the disk drive and mount the separator. Wire the ground and +5v lines to the separator by soldering your wires to the power connector on your drive. Watch out! There is ground, +5 volts and +12 volts at this connector. Pin 4 of the power connector is usually +5v and pin 3 is usually ground, but please double check with your drive manual and drive power supply. Next, follow the copper land coming from PIN 32 of J1. On the Siemens FDD 100-5B, it doesn't go anywhere, but if it does on your drive, cut it. Follow the land coming from PIN 30 of Jl back about 1/2 inch from the connector, and cut it. Scrape a little of the paint insulation off each side of the cut land. On the connector side of the cut, solder a wire COMING from the RECEIVE CLOCK OUTPUT of the data separator to this land. On the side of the cut coming from the drive circuitry, solder a wire from this point TO the composite data input of the separator. On the land coming from PIN 20 of Jl, solder a wire GOING to the STEP NOT INPUT of the data separator. Solder the wire FROM the RECEIVE DATA OUTPUT of the data separator to the land coming off of PIN 34 of Jl. PIN 34 of Jl is the spare pin that OSI uses. The land from PIN 34 of Jl should not go anywhere on your drive. Hardware modification is completel

Once the circuit was built and

installed, I adjusted it by putting a write protected disk in the drive, and then tried booting the system while adjusting the pot on the DATA SEPARATOR. There is a wide range adjustment of the pot that will allow the separator to work. Center it in the middle of that range.

Ahh, but how could the Siemens drive boot a disk when it has a slow step speed? I found that for a short period it will step beyond its rated speed. I wouldn't trust it to do it all the time, but for me it did work at the faster OSI step speed long enough to boot OS65D V3.3. When it did boot the first time, OS65D3.3 went directly to the KERNEL. It shouldn't have done that! Amazingly enough, the KERNEL commands still worked!!! tried booting it again, this time it booted to and the BEXEC*. You should then imme-diately EXIT BASIC to the KERNET. Give the Give the command: KERNEL. 'EM'. This will put you into the Extended Monitor. If the Extended Monitor doesn't load, keep trying until it does. On my system it worked the first time, but with some slower drives it may take a couple of may take _____ Immediately after 'EXIT' tries. entering the EM, type 'EXIT' to return to the KERNEL. Type command: 'CA 0200=06,4' if you are using OS65D V3.3 or type command: 'CA 0200=13,1' ONLY if you are using OS65D V3.1. Then type 'GO 0200'. The Track Zero/Copy Utility will appear. If you are using V3.1 a menu will be the first item displayed, select option 2. Version 3.3 of the DOS doesn't have this menu and goes directly to the Track Zero utility. When you are at the Track Zero Utility, type: R4200. The drive will hum. Exit to the KERNEL. Type: 'RE EM' (You are once again in the extended monitor). Type: extended monitor). Type: '046A3' (The '0' sign is the 'shift-P'). The monitor sho-uld then respond with: 46A3 08. Type: 20 then a <CR> (carriage return). Type: @467B <CR>. The monitor should respond with: 467B 31. It may also respond with: 467B C7. If C7 isn't in location 467B, put it there by typing: C7 and then <CR>. Type 'EXIT'. You will then again be at the KERNEL. ' CA Type: 0200=06,4' (CA 0200=13,1 for V3.1). Get to the Track Zero Utility and type: 'W4200/ 2200,8' to store the newly modified track zero on your disk. Do this with a back-up disk if possible and make sure it IS NOT write protected. That should give you a modi-fied OS65D disk. Use this disk to modify your other ones by using the Track Zero Copy Utility. With this disk you could even modify a PICO-DOS disk! It seems that PICO-DOS uses the same boot track that OS65D does. Keep in mind that you still need one of the versions of OS65D to modify PICO-DOS as the simpler DOS does not have a KERNEL or Extended Monitor.

If you can't get your disk to boot due to the drive not stepping fast enough, you will either have to get together with a friend that owns a ClPMF or make sure that the drive you are buying has a fast step speed (somewhere around 6ms). The faster drives are becoming more popular and less expensive. If you have a friend that owns a ClPMF then you will not have any problems modifying your diskettes on his/her system.

SCHEMATIC OF DATA SEPARATOR



INTERFACE SIGNALS

SIEMEN	NS MODEL	, OSI	610	BOARD
EDD]	L00-5B	EQU:	IVALE	NT DISK
DRIVE		CONT	TROLL	ER
J1-2		SPA	ARE	
J1-4		SPA	ARE	
J1-6	DRIVE S	ELECT	0	
J1-8	INDEX/S	ECTOR		J3-17
J1-10	DRIVE S	ELECT	1	J3-3
J1-12	DRIVE S	ELECT	2	J3-18
J1-14	DRIVE S	ELECT	3	J3-
J1-16	MOTOR C	N		J3-
J1-18	DIRECTI	ON SEL	LECT	J3-6
J1-20	STEP			J3-5
J1-22	COMPOSI	TE WRI	[TE	J3-9
	DATA			

J1-24	WRITE GATE	J3-8
J1-26	TRACK 00	J3-23
J1-28	WRITE PROTECT	J3-19
J1-30	RECEIVE CLOCK*	J3-10
J1-32	SIDE SELECT**	
J1-34	RECEIVE DATA*	J3-11

ALL ODD PINS ARE GROUND

* - AFTER MODIFICATION.

**- CUT, NOT USED!

\star

EXPANDING THE CLP/SBII

PART 3

By: David Tasker lll Bass Highway Tasmania, Australia 7303

Adding the 1st 8K continued

8K STATIC RAM BOARD PARTS LIST

INTEGRATED CIRCUIT SOCKETS 2 x 14 pin. 7 x 16 pin. 16 x 18 pin.

INTEGRATED CIRCUITS I.C. Number- 1.2,3.. 74LS367 or 74LS365 or 8T97 or 8097. I.C. Number- 4.. 74LS138. I.C. Number- 5.. 74155. I.C. Number- 6... 7400. I.C. Number- 7.. I.C. Number- 8.9.. 7412. 8T28. (8T26 may be used if also fitted on ClP) (8T26 for C4P) I.C. Number- 10 thru 25.. 2114 Static RAM. 450ns or faster. RAMS increment in pairs e.g. 10 and 11.

CAPACITORS Disk Ceramic preferred for physical size. Polyester may be used.

Capacitors. Cl-C6,C9 all 0.047 or 0.luf Disc. Capacitors C7. 3.3uf Tantalum. Capacitor C8. 47 uf 16v. Electrolytic.

RESISTORS R1-R4. All 1Kohm, 1/4 watt.

The printed circuit board comes coated with an antitarnish coating which acts as a soldering flux aid. It is not necessary to clean the board. Except that when the board is fully assembled you will have to clean the copper area which plugs into the Motherboard edge connector. To clean, hold the board in such a way that the copper is facing up and the component side of the board is supported only underneath the edge connector area, e.g., place the board at the edge of a table or workbench. A non metallic scourer pad is preferable to steel wool to clean the edge connector area as steel wool tends to scratch heavily but. more importantly, also leaves fine particles of steel which must be carefully cleaned away from both sides of the board.

Avoid touching the cleaned area with fingers as this will cause tarnishing. Once testing of the board is complete, give this area a final rub over with your cleaning pad and once plugged into the Motherboard try to avoid too many insertions and removals.

ASSEMBLY INSTRUCTIONS

Begin by inserting as many straps on the board as possible. Do not yet insert the 7 straps which run zig-zag between the RAM Integrated circuits (I.C.'s 10 to 25) as these are best put in after the 18 pin RAM sockets are inserted. If you work from the edge connector end of the board, left to right, you will have the board orientated to the diagrams. There are two component placement sheets but one of them does not show any sockets at all. This sheet is straps, resistors, and capacitors only.

You can use bare wire for all straps if you like but it is a good idea to alternate bare with insulated wire where there are many straps running along side one another.

Make certain the straps underneath the integrated circuits are in.

Insert the 4 Resistors.

Do not insert the capacitors until AFTER the sockets as these will forever be in the way each time you turn the board over to solder.

Once you have inserted all the sockets, check carefully that there are no solder bridges between tracks. If you have a multi-meter you could check for shorts between tracks particularly in the RAM area where tracks and soldering are very close together. You could also check continuity of the finer tracks with your meter.

BEFORE PLUGGING IN INTEGRATED CIRCUITS

If you have a multi-meter or logic probe you could insert the board onto the bus and check that +5 volts (1) and



ISOTRON, INC.

INCREASES SOFTWARE SUPPORT

ISOTRON has opened two regional OSI Software Support Centers, increasing access time to its Support Staff.

Our skilled people are anxious to answer questions regarding our Ohio Scientific software products including: OS-65U and TurboDos operating systems, KeyWord, KeySort, KeyBasic and utility programs.

FOR ASSISTANCE CALL:

From USA, Eastern and Central time zones, Europe, Africa, South America, Central America, the Caribbean and Eastern Canada:

ISOTRON SUPPORT EAST (616) 451-8435 (8:30 A.M. - 5:00 P.M., EST, Monday through Friday)

From USA, Mountain and Pacific time zones, Australia, New Zealand, Asia, Far East and Western Canada:

ISOTRON SUPPORT WEST (503) 796-1018 (9:00 A.M. - 5:00 P.M., PST, Monday through Friday)

THIS SERVICE IS ANOTHER INDICATION THAT ISOTRON IS COMING THROUGH FOR YOU.



140 SHERMAN ST. FAIRFIELD, CT 06430 (203) 255-7443 TLX-756436 earth (0) are connected to the respective supply pins of the I.C. sockets.

INSERT CAPACITORS

Insert all capacitors and pay attention to the polarity markings of C7 and C8. Tantalum capacitors are usually marked with one lead positive (+), however, if not, then the longest lead is positive. Electrolytic capacitors have the negative lead marked.

ZIG - ZAG STRAPS

Insert the seven straps that run between the 18 pin sockets. You can do this just after the insertion of the sockets. Insert and solder the strap at one end, then using the sockets as corner posts, run the strap to the other hole. Do not pull the strap too tight as they may cut in under the socket and could cut into one or more of the socket pins.

TESTING

Check once more for any shorts on the board. Insert all I.C.s at this point. It is a good idea to only insert the first pair of RAM chips to start.

BOUNDARY SELECTION

Each 8K memory board can be divided up into two 4K memory blocks. These two memory blocks may be anywhere within the lower 32K of your computer's memory. If you need the RAM in the upper 32K, then refer to the circuit diagram and the note regarding address line Al5.

From the strapping sheet which shows the RAM selection table, select the straps which suit your application. In most cases, the 4K blocks will follow each other. For example, if you own an Ohio ClP or 4P, then these computers have provision on their main boards for the 1st 8K of memory. If this memory board is the first one used for expansion. then both 4K blocks will follow on from the computer's memory and thus will provide the second 8K of RAM (total of 16K).

RAM TEST

Make sure all power is turned off. Insert the memory board onto the bus. If you have a memory board which has the 40pin expansion socket. then make sure you refer to the assembly sheets for that section of the board. You will



Reset line is optional and is a hardware reset for input/output board.

For power on reset, connect this line as:



either be connecting this $4\emptyset$ pin socket directly into the computer or cutting the expansion off and plugging both into a motherboard.

Turn on the computer. If you hit the BREAK key, the computer should respond with the usual D.C.W.M?. If it does not then you have a fault on your new board. There are three most likely areas for trouble at this point and during the subsequent testing stages.

1. Address lines open, shorted, or buffers reversed.

2. Data lines open, shorted, or buffers reversed.

3. Control lines are faulty, e.g. RW or Ø2 wrong. DD (data direction) reversed. Check that the buffers are correctly inserted. With a logic probe or oscilloscope, check that the address lines and data line are changing from \emptyset to l at a fast rate. Check that DD is high (1 means write to memory).

If you have D.C.W.M.?, then proceed with a cold start in BASIC. If all is well and you have only one pair of RAMS inserted, you should have 8447 BYTES FREE. For the complete 8K RAM inserted you should get 15615 BYTES FREE.

Next month, the Motherboard.

More schematics on page 7



8K - 4K BOUNDARY RAM CARD, 2 mhz STATIC RAM BOARD * 7412 for OSI Boards with 1K pull up Resistors R1,2,3 capacitors C1-C6 .047 pf OR .lpf (l00uf) as required.

40 PIN - RIBBON EXPANSION - Clp/SBII - TASKER MOTHERBOARD

PC PATTERN AS VIEWED FROM COMPONENT SIDE

 \hat{t}_{i}

7



THIS CARD IS NORMALLY PROVIDED AS AN INTACT PART OF THE 1st USED 8K - 2114 RAM CARD, i.e., it is electrically & physically joined at the 36 way edge connector. This enables an 8K MEM card to be used on the ClP/SBII via the 40 pin expansion socket without the need for a motherboard.

WHEN additional boards are required, i.e., a motherboard to be added, then a cut is made through the edge connector to separate the 40 pin socket section as shown.

NOTE: When used on a motherboard - all component sides of boards face towards the address buffer ICs on the M/Bd. This card plugs into Slot "A" only.

XREF: BASIC FILE CROSS REFERENCE GENERATOR PART TWO

by:Rick Trethewey 8 Duran Court Pacifica, CA 94044

10;	XREP:	BASIC	FILE CROSS	S REFERENCE GENERATOR	460		LDA	SOF+1	GET OFFSET MSB
201	FRA 4	-			400		ALC	TBUFFERY 200	ADD BUFFER ADDRESS MEB
307	DEND	1 103 4	enn		400		SIA	TATPIR+1	GIVE TO TXTPIR
4U 60	READ				490		STA	SUP+1	SAVE FOR PASS 2
20		DIA A			500		LUA CITA	PUTPIR	SAVE LAST FREE ENTRY AS
00			BUFFER 256	LUAD SUFFER FSB	510		STA	REFBOT	REFERENCE TABLE BOTTOM
/0		STA A	ADRHX	GIVE TO 65D	520		LDA	FUTPIR+1	HANDLE MSB TOO
80			rşui		530		STA	REPBOT+1	
.90		STAE	SECT	SET 650 TO SECTOR #1	540		LDA	#\$00	INIZ
100		JSR L	JOAD	LOAD HEAD	550		STA	TXTPIR	CLEAR TXTPIR
110		JSR C	XILIX	READ SECTOR	560		JMP	XREF	JUMP TO HANDLING CODE
120		JMP U	INLUAD	UNLOAD HEAD AND OUT	570	÷			
1307					580	GETCHR	LDY	INDEX	PETCH BUFFER PAGE INDEX
140	XHQ	LDAS	SPIK	GET FILE START TRACK	590		LDA	(TXTPTR),Y	FETCH CHARACTER
120		SIA 1	NAKA	GIVE TO 65D	600		INY		BUMP INDEX
100		JSR S	WAP	* DOS CONTEXT *	610		BEQ	GEIC1	PAGE? ==> GETC1
1/0		JSR 5	SEEKX	MOVE HEAD TO TRACK	620		STY	INDEX	NO: SAVE INDEX
180		JSR	CAD	READ IN IST ROACK OF FILE	630		RIS		AND QUIT
190		JSR 5	WAP	* LANGUAGE CONTEXT *	640	GEIC1	INC	TXTPIR+1	BUMP TATPIR MSB ON PAGING
200		LUAE	SOFFER	FEICH FILE START LSB	650		STY	INDEX	RESET BUFFER INDEX
210		SEC			660		LDY	TXTPIR+1	Petch New MSB
220		SBC 1	SRLSTR	SUBTRACT SOURCE START	670		CPY	BPENPG	AT END OF BUFFER?
230		STA S	OF .	SAVE HEADER OPPSET LSB	680		BEQ	GETC2	YES1 ==> GETC2
240		LDA B	UFFER+1	HANDLE MSB TOO	690	•	RIS		AND QUIT
250		SBC #	SRCSTR/256-	-9	700	GETC2	PHA		SAVE FETCHED CHARACTER
260		STA S	SOF+1	NOTE 3.3 FIX ABOVE	710		TXA		FUT X IN ACC.
270		BEQX	URQ3	OPPSETINGE = 0 ?? ==>	720		PHA		SAVE IT TOO
280		SEC			730		LDA	BUFFER	PETCH BUFF. ADDR. LSB
290	XRQ1	LDA S	SOP+1	FETCH OPPSET MSB	740		STA	TATPIR	RESET TXTPTR
300		SBC 5	SRCSIZ	SUBTRACT ONE TRACK	750		LDA	#BUFFER/256	PETCH BUPP. ADDR. MSB
310		BCC X	CRQ2	LESS THAN 1 TRACK ? ==>	760		STA	TXIPIR+1	RESET TATPTR MSB
320		STA S	50F+1	NOI SAVE RESULTI	770		JSR	SWAP	* DOS CONTEXT *
330		INC 3	IRAKX	SHOW MOVE TO NEXT TRACK	780		LDA	TRAKX	PETCH CURRENT TRACK #
340		BNE >	GQ1	AND LOOP!	790		QIP	ENDIK	E.O.F. ?
350	XRQ2	JSR 5	MAP	* DOS CONTEXT *	800		BEQ	GETERR	YESI ERRORI ==>
360		JSR 8	SEEKX	SEEK PROPER TRACK	810		INC	TRAKX	NO, BUMP IT ONE
370		JSR F	READ	READ SECTOR INTO BUFFER	820		JSR	SEEKX	MOVE HEAD TO TRACK
380		JSR 8	SWAP	* LANGUAGE CONTEXT *	830		JSR	READ	READ IN TRACK
390		LDA 1	IRAKX	GET RESULTANT TRACK #	840		JSR	SWAP	* LANGUAGE CONTEXT *
400		STA 8	STIK	SAVE AS NEW STIK	850		PLA		RETRIEVE X
410	XRQ3	LDA 8	SOP	GET OFFSET LSB	860		TAX		PUT IT BACK
420		ac			870		PLA		RETRIEVE TEXT CHARACTER
430		ADC	BUFFER	ADD BUFFER ADDRESS LSB	880		RTS		AND QUIT
440		STA I	NDEX	GIVE TO INDEX	890	1			=
450		STA S	SOF	SAVE FOR PASS 2	900	GETERR	JSR	STROUT	SHOW ERROR TO USER!
									Continued

From Gander Software

A New Standard of Excellence

FINANCIAL PLANNER

Get "What If" answers for up to 10 displayed problems in:

- Loan/Annuity Analysis
- Annuity 'Due' Analysis
- Present/Future Value Analysis
- Sinking Fund Analysis
- Amortization Schedules
- Interest Conversions

HARDWARE REQUIREMENTS: 48K OSI, 8" floppy, serial terminal system, OS-65U v. 1.2 or later.

FEATURES: package allows configuration to almost all non-ANSI terminals, AND user specification of printer port.

PRICE: \$300.00 (User Manual, \$25.00, credited toward Planner purchase). Michigan residents add 4% sales tax.

DEALERS: This program, of great value to lawyers, bankers, insurance people, and real estate people, will help you sell hardware! Inquiries invited.

A POWERFUL TOOL FOR EVALUATING ALTERNATIVES!

The first four programs all: allow you to solve a named variable after changing another variable, let you net the difference between any displayed problems, provide selective saves to disk, give you very informative printouts based on the problems solved, and much, much more.

The "Amortization Schedules" program provides more flexibility than any other schedule known to GANDER. It lets you deal with balloon payments, early pay-offs, annual payment increases (by percentages or dollars), keeps a running total of your entire transaction to pay off, schedules payments by both month and year, and reports YTD totals based on user selected calendar OR fiscal years.

"Interest Conversions" lets you key in any nominal rate and reports the true effective rate for compounding semi-annually, quarterly, monthly, daily, and continuously, and allows the print out of interest tables (your choice of rate and increments). It also includes a simple calculator, which can be used without disturbing other problems displayed, and which contains three separate user addressable memories.

Finally, to aid planning, the Menu program will generate a calendar for any month/year between 1901 and 2399, and accurately accounts for leap years!

GANDER SOFTWARE 4

3223 Bross Road "The Ponds" Hastings, MI 49058 (616) 945-2821



910 920	.B) .B)	TE CR,LP TE 'END OP I	PILE ERROR', CR. LF.0
930	LDA	DEFAUL+1	GET CONSCLE DV #
940	JME	VUPLAG WARM	RETURN TO MENU
960;			
970 GE 980	STLIN JSP STA	NLAL	GET CHAR. FROM FILE SAVE AS NEXT LINE LSB
990	JSF	GETCHR	GET CHAR. FROM FILE
1000	STA	A NLAH	SAVE AS NEXT LINE MSB
1020	BNE	GEILIL	NO, CONTINUE ==> GETLI
1030	LDA	NLAL	YESI CHECK LSB
1040 1050 GE	arii 122	GETCHR	SUU ALSO 7 ==> E.O.F.I GET CHARACTER FROM FILE
1060	STA	LNLO	SAVE AS LINE # LSB
1070	JSE	GETCHR	GET CHARACTER FROM FILE
1090	LDS	#\$00	INIZ TXBUFF INDEX
1100 G	ETLI2 JSE	GETCHR	GET CHARACTER FROM FILE
1120	BR	GETT.T2	WAS IT A (SPACE) ? VESI ICNOREL max GEW.12
1130	ST	TXBUPP,X	NO, SAVE IN TRBUFF
1140	IND		BUMP TXBUFF INDEX
1160	BNE	GETL12	NOT \$00? ==> GETLI2
1170	512	ONFLAG	YESI CLEAR "ON" FLAG
1180 GE	SILI3 RIS	5	AND QUIT
1200 XF	NET JSF	GETLIN	GET LINE OF TEXT
1210	LDA	NLAH	CHECK FOR E.O.F.
1230	LDA	NLAL	MAYBE, CHECK LSB
1240	ENE	XREP1	NO ===>
1250 1260 XI	JME REPI LOV	DISLIN	YES! GO TO DISPLAY
1270 X	UEP2 LDA	TXBUPP, Y	LOOK AT CHARACTER
1280	BEC	XREP	E.O.L.? => LOOPI
1290	BEC	INSERT	"GOIO"? YESI INSERT LINE #
1310	CM	GOSBIK	"GOSUB"?
1320	BEC	INSERT	YES! ==>
1340	BEC	INSERT	YESI =>
1350	CM	TONIK	"ON" ?
1360	BNE	XREP3	NO! ===> VESI SHOW "ON" TN DROCEDESS
1380 X	REP3 IN		BUMP POINTER INDEX
1390	BNE	XREP2	AND LOOP!
1410 1	SERT IN	•	RUND DOTNTOT 1
1420	1 774		
	1.00	\$\$00	INIZ
1430	LDA	\$\$00 TXBUFF,Y	INIZ YESI LOOK AT NEXT CHARACTER CHECK FOR LINE &
1430 1440 1450	LDA LDA CME BCC	\$\$00 TXBUFF,Y \$'0 XREF2	INIZ YESI LOOK AT NEXT CHARACTER CHECK FOR LINE # NO! => RETURN
1430 1440 1450 1460		: #\$00 TXBUFF,Y #*0 : XREF2 #*9+1	VEST LOOK AT NEXT CHARACTER CHECK FOR LINE # NO! => RETURN
1430 1440 1450 1460 1470 1480 IR	LLA LDA CME BCC CME BCC NSL LDA	<pre>\$ #\$00 A TXBUPF,Y # * 0 C XREP2 # * 9+1 S XREP2 A TXBUFF,Y</pre>	POINTER T INIZ YES! LOCK AT NEXT CHARACTER CHECK FOR LINE # NOI => RETURN PETCH CHARACTER
1430 1440 1450 1460 1470 1480 IP 1490	LLA LDA OME BOX OME BOX STL STL STL STL STL STL STL	(#\$00 TXBUPF,Y #'0 XREF2 #'9+1 XREF2 TXBUFF,Y INBUF,X	PETCH CHARACTER TRANSFER TO INBUP
1430 1440 1450 1460 1470 1480 IP 1490 1500 1510		: #\$00 A TXBUPF,Y #*0 : XREF2 #*9+1 : XREF2 A TXBUFF,Y A INBUF,X #*0 * TNS2	PETCH CHARACTER PETCH CHARACTER TRANSFER TO INBUP LOOK FOR END OF NUMBER
1430 1440 1450 1460 1470 1480 IP 1490 1500 1510 1520	LLA LLA OVE BCC STA STA STA STA STA STA STA STA STA STA	: #\$00 A TXBUFF,Y #'0 : XREF2 A TXBUFF,Y A TXBUFF,Y A TXBUFF,Y A TXBUFF,Y C INS2 P #'9+1	DOWN FORMAL I INIZ YESI LOOK AT NEXT CHARACTER CHECK FOR LINE # NOI => RETURN PETCH CHARACTER TRANSFER TO INSUF LOOK FOR END OF NUMBER
1430 1440 1450 1460 1470 1480 1490 1500 1510 1520 1530		: #\$00 TXBUFF,Y #'0 : XREF2 #'9+1 : XREF2 : TXBUFF,Y INBUF,X #'0 : INS2 : INS2 : INS2	DENTER FORMAL FOR THE STANDARD
1430 1440 1450 1460 1470 1480 P 1490 1500 1510 1520 1530 1550	ILIA CME BCC OME BCC OME STI STI CME BCC CME BCC INI INI INI	\$\$00 TXBUFF,Y \$10 XREF2 \$19+1 XREF2 TXBUFF,Y TXBUFF,Y TXBUFF,Y \$10 UNSU \$19+1 SINS2	DIATE TO ANTIAL T THIZ YES! LOOK AT NEXT CHARACTER CHECK FOR LINE ♦ NO! => RETURN PETCH CHARACTER TRANSFER TO INBUP LOOK FOR END OF NUMBER BUMP TRBUFF INDEX BUMP TRBUFF INDEX
1430 1440 1450 1460 1470 1480 1500 1510 1520 1530 1540 1550 1560	ILA OME BCC OME BCC OME BCC STE STE STE STE STE STE STE STE STE STE	: \$\$00 T260FF,Y \$*0 : XREF2 **9+1 : XREF2 **9+1 : XREF2 **9+1 : TNSU **0 : TNS2 : TNS2 : INS1	DUAL TOTAL T INIZ YES! LOCK AT NEXT CHARACTER CHECK FOR LINE # NDI => RETURN PETCH CHARACTER TRANSFER TO INBUP LOCK FOR END OF NUMBER BUMP TXBUFF INDEX BUMP TXBUFF INDEX BUMP TXBUFF INDEX LOCP 'TILL DONE
1430 1440 1450 1460 1470 1480 R 1490 1500 1510 1520 1530 1540 1550 1550 1550 1550	LDA DA BCX ONU BCX DA BCX DA BCX BCX DA BCX DA BCX DA BCX DA BCX DA BCX DA BCX DA BCX DA BCX DA BCX DA BCX DA DA DA DA DA DA DA DA DA DA DA DA DA	<pre>\$\$00 T260FF,Y \$*0 XREF2 \$*19+1 XREF2 \$*19+1 XREF2 \$*19+1 INBUFF,Y INBUFF,Y INBUFF,Y INBUFF,Y INBUFF,Y INBUFF,Y INBUFF,Y \$*19+1 INS2 \$*19+1 INS2 INS1 INS1 INS1</pre>	DUAL TOURING T INIZ YESI LOOK AT NEXT CHARACTER CHECK FOR LINE # NDI => RETURN PETCH CHARACTER TRANSFER TO INBUF LOOK FOR END OF NUMBER BURP TXBUFF INDEX BURP TXBUFF INDEX BURP TXBUFF INDEX BURP TXBUFF INDEX CONVICT LOOK
1430 1440 1450 1450 1470 1480 1500 1510 1520 1530 1540 1550 1560 1570 1580	LIA COL BCC BCC BCC BCC STI COL BCC COL BCC COL BCC STI INN BCC STI INN SZ STI JSI JSI JSI	(\$500 TKBUFF,Y *10 TKBUFF,Y *10 TKBUFP,Y TKBUFP,Y 108UFP,Y 108UFP,Y 108UFP,X *19+1 1082 1	DUAL TO BUT A T INIZ YES! LOGK AT NEXT CHARACTER CHECK FOR LINE & NO! => RETURN PETCH CHARACTER TRANSFER TO INBUP LOCK FOR END OF NUMBER BUMP TXBUFF INDEX BUMP TXBUFF INDEX B
1430 1440 1450 1450 1470 1490 1500 1510 1520 1550 1550 1550 1550 155	LIA CHE BCC CHE BCC CHE BCC CHE BCC CHE BCC CHE BCC CHE BCC CHE BCC CHE BCC CHE BCC CHE BCC CHE BCC CHE BCC CHE BCC CHE CHE CHE CHE CHE CHE CHE CHE CHE C	(\$500 TXBUFF,Y *10 XREF2 *19 XREF2 *19+1 XREF2 TXBUFF,Y 108UFF,Y 108UFF,Y 108UFF,Y 108UFF,X *19+1 1082 *19+1 1082 *19+1 1082 *19+1 1082 *19+1 1082 *19+1 1082 *19+1 1082 *19+1 1082 *19+1 *1082 *1097 *1082 *1097 *1007 *100	DUAL TOINTIA T INIZ YESI LOOK AT NEXT CHARACTER CHECK FOR LINE # NOI => RETURN PETCH CHARACTER TRANSFER TO INSUP LOOK FOR END OF NUMBER BUMP INSUP INDEX BUMP INSUP INDEX BUMP INSUP INDEX BUMP INSUP INDEX CONVERT LINE # TO BINARY INSERT LINE # REFERENCE RETHIEVE LINE # REFERENCE
1430 1440 1450 1450 1470 1490 1500 1510 1520 1550 1550 1550 1550 155	LIA CHE BCC CHE BCC CHE BCC CHE BCC CHE BCC CHE BCC CHE BCC CHE BCC CHE BCC CHE BCC CHE BCC CHE BCC CHE BCC CHE CHE CHE CHE CHE CHE CHE CHE CHE C	(\$500 TXBUFF,Y *10 TXBUFF,Y *10 XREF2 *19+1 XREF2 *19+1 XREF2 *19+1 XREF2 *19+1 INSL *10+1 INSL *10+1 INSL *10+1 INSL *10+1 INSL *10+1 INSL *10+1 INSL I	DUAL FORMANY I INIZ YESI LOOK AT NEXT CHARACTER CHECK FOR LINE # NOI => RETURN PETCH CHARACTER TRANSFER TO INSUF LOOK FOR END OF NUMBER BUMP TABUFF INDEX BUMP TABUFF INDEX BUMP TABUFF INDEX BUMP TABUFF INDEX BUMP TABUFF INDEX CONVERT LINE # TO BINARY INSERT LINE # TO BINARY INSERT LINE # TO BINARY INSERT LINE # INDEX CHECK FOR "ON"
1430 1440 1450 1470 1470 1500 1510 1520 1530 1530 1540 1550 1550 1560 1590 1590 1600 1610 1630	LIA CHE BCC CHE BCC CHE BCC CHE BCC BCC BCC BCC BCC BCC BCC BCC BCC BC	(\$500 TXBUFF,Y *10 TXBUFF,Y *10 *10 *19+1 XREF2 *19+1 XREF2 *10 *19+1 XREF2 *10 *10 *10 *10 *10 *10 *10 *10	DUM FORMATIN T INIZ YESI LOCK AT NEXT CHARACTER CHECK FOR LINE ↓ NOI => RETURN PETCH CHARACTER TRANSFER TO INBUP LOCK FOR END OF NUMBER BUMP TXBUFF INDEX BUMP TXBUFF INDEX BUMP TXBUFF INDEX BUMP TXBUFF INDEX BUMP TXBUFF INDEX COMMERT LINE ↓ TO BUNARY INSERT LINE ↓ TO BUNARY INSERT LINE ↓ TO BUNARY INSERT LINE ↓ TO BUNARY CHECK FOR "ON" YESI ==> INS3 NOI RETURN TO MAIN LOOPI
1430 1440 1450 1470 1470 1500 1510 1520 1530 1540 1550 1560 1550 1560 1560 1610 1620 1630 1630 1640 1630	LIA CAR CAR CAR CAR CAR CAR CAR CAR CAR CA	(\$00 TOBUFF,Y **0 TOBUFF,Y **0 **9+1 XREF2 **9+1 XREF2 **9+1 INS2 **0 **0 INS2 **0 **0 INS2 INS1 TMPPTR DDCHEX FNDLIN TMPPTR DDCHEX FNDLIN TMPPTR OKELAG INS3 XREF2 XREF2 TOBUFF,Y TOBUFF,Y	DUAL FORMATIN T INIZ YESI LOCK AT NEXT CHARACTER CHECK FOR LINE # NOI => RETURN PETCH CHARACTER TRANSFER TO INBUP LOCK FOR END OF NUMBER BUMP TRBUF INDEX BUMP TRBUF INDEX LOCM 'TIL DONE SAVE CURRENT TOBUFF INDEX CONVERT LINE # INDEF SAVE CURRENT TOBUFF INDEX CONVERT LINE # INDEX CHECK FOR "ON" YESI ==> INS3 NOI RETURN TO MAIN LOOPI
1430 1440 1450 1470 1480 H 1490 1500 1520 1520 1550 1550 1550 1550 155	LIA LIA BCC BCC BCC BCC BCC BCC BCC BC	(4500 TXBUFF,Y 4*0 TXBUFF,Y 4*0 TXBUFF,Y TXBUFF,Y 108UFF,Y 108UF,X 4*10 108UFF,Y 108UF,X 4*10 108UFF,Y	DUAL FORMATION T INIZ YESI LOCK AT NEXT CHARACTER CHECK FOR LINE ↓ NOI => RETURN PETCH CHARACTER TRANSFER TO INBUP LOCK FOR END OF NUMBER BUMP TXBUFF INDEX BUMP TXBUFF INDEX LOCP 'TIL DONE SAVE CURRENT TXBUFF INDEX LOCP 'TIL DONE SAVE CURRENT TXBUFF INDEX COMMERT LINE ↓ INDEX CHECK FOR "ON" YESI ==> INE3 NOI RETURN TO MAIN LOOPI FFTCH LAST CHARACTER SEEN COMMA ?
1430 1440 1450 1470 1490 1500 1520 1520 1520 1550 1550 1550 155	LIA BCC OFF BCC OFF BCC OFF BCC OFF BCC BCC BCC BCC BCC BCC BCC BCC BCC B	(\$500 TKBUFF,Y \$'0 TKBUFF,Y \$'0 TKBUFF,Y TKBUFF,Y TKBUFF,Y \$'0 TKBUFF,Y \$'1 TKS1 TMPPTR DECHEX CMF1AG TKS1 TKS1 TKS1 TKS1 TKS1 TKS1 TKS1 TKS1 TKS1 TKS1 TKS3 TK	DIATIN T INIZ VESI LOGK AT NEXT CHARACTER CHECK FOR LINE 4 NOI ⇒> RETURN PETCH CHARACTER TRANSFER TO INBUP LOCK FOR END OF NUMBER BUMP TXBUFF INDEX BUMP TXBUFF INDEX BUMP TXBUFF INDEX BUMP TXBUFF INDEX BUMP TXBUFF INDEX BUMP TXBUFF INDEX BUMP TXBUFF INDEX CONVERT LINE 4 REFERENCE RETURE 4 REFERENCE RETURE 4 REFERENCE RETURE 4 REFERENCE RETURE 4 TO BUNARY VESI ==> INS3 NOI RETURN TO MAIN LOOPI FETCH LAST CHARACTER SEEN COMMA ? NDI GO BACKI
1440 1440 1440 1470 1470 150 1510 1510 1520 1530 1550 1550 1550 1550 1550 1550 155	LIA EDA BCC OFF BCC OFF BCC OFF BCC OFF BCC OFF BCC DF BCC BCC DF DF BCC DF BCC DF BCC DF BCC DF DF DF DF DF DF DF DF DF DF DF DF DF	(\$500 TKBUFF,Y *10 TKBUFF,Y *10 TKBUFF,Y *19+1 TKBUFF,Y *10 TKBUFF,X *19+1 TKS2 *1957 TKS2 *1957 TKS2 *1957 TKS2 *1957 TKS2 *1957 TKS2 *1957 TKS2 *1957 TKS2 *1957 TKS2 *1957 TKS2 *1957 TKS2 *1957 TKS3 *1957 TKS3 *1957 TKS3 *1957 TKS3 *1957 TKS3 *1957 TKS3 *1957 TKS3 *1957 TKS3 *1957 TKS3 *1957 TKS3 *1957 TKS3 *1957 TKS3 *1957 TKS3 *1957 TKS3 *1957 TKS3 *1957 TKS3 *1957	DUAL TOTAL T INIZ VESI LOGK AT NEXT CHARACTER CHECK FOR LINE 4 NOI => RETURN PETCH CHARACTER ITAANSPER TO INBUF LOCK FOR END OF NUMBER BURP INSUF INDEX BURP INSUF INDEX BURP INSUF INDEX CONVERT LINE 4 TO BUNARY INSERT LINE 4 REPERACE RETRIEVE LINE 4 INDEX CHECK FOR "ON" YESI INSI INSI NOI RETURN TO MAIN LOOPI FTICH LAST CHARACTER SEEN COMMA ? NOI GO BACKI VESI INIZ XFFER INDEX BURP FECH FORMER BURP FECH FORMER
1440 1440 1450 1470 1470 150 1500 1510 1520 1530 1540 1550 1550 1550 1550 1560 1660 1620 1630 1640 1650 1650 1650 1650 1650 1650	LIA BOX BOX BOX BOX BOX BOX BOX BOX BOX BOX	(\$500 TKBUFF,Y *10 TKBUFF,Y *10 KREF2 *19+1 KREF2 TKBUFF,Y *19+1 TKS2 *1957 TKS2 *1957 TKS2 *1957 TKS2 *1957 TKS2 *1957 TKS2 *1957 TKS2 *1957 TKS2 *1957 TKS2 *1957 TKS2 *1957 TKS2 *1957 TKS2 *1957 TKS2 *1957 TKS2 *1957 TKS2 *1957 TKS2 *1957 TKS3 *1957 TKS3 *1957 TKS3 *1957 TKS3 *1957	DUAL FORMATION T INIZ VESI LOCK AT NEXT CHARACTER CHECK FOR LINE # NOI => RETURN PETCH CHARACTER TRANSPER TO INBUP LOCK FOR END OF NUMBER BUMP TXBUPF INDEX BUMP INBUP INDEX BUMP INBUP INDEX LOCP 'TIL DONE SAVE CURRENT TXBUPF INDEX CONVERT LINE # TO BINARY INSERT LINE # REFERENCE RETHLEVE LINE # INDEX CHECK FOR "ON" YESI INIZ XFER INDEX DOI GO BACKI YESI INIZ XFER INDEX BUMP FETCH FOINTER AND LOOPI
1440 1440 1450 1470 1480 H 1470 1500 1510 1520 1530 1540 1550 1550 1550 1560 1630 1640 H 1630 1640 H 1650 1650 1650 1650 1650 1650 1650 1650	III III BCC BCC BCC BCC BCC BCC BCC BCC	(\$500 TXBUFF,Y *10 TXBUFF,Y *10 XREF2 *19+1 XREF2 *19+1 XREF2 *19+1 INSU *1000 *19+1 INSU *1000 *1	DUAL FORMAL T INIZ YESI LOCK AT NEXT CHARACTER CHECK FOR LINE # NOI => RETURN FFITCH CHARACTER TRANSFER TO INSUP LOOK FOR END OF NUMBER BUMP TXBUFF INDEX BUMP TXBUFF INDEX BUMP TXBUFF INDEX BUMP TXBUFF INDEX CONVERT LINE # TO BINARY INSUET LINE # TO BINARY INSUET LINE # TO BINARY INSUET LINE # TO BINARY INSUET LINE # TO BINARY YESI ==> INS3 NOI RETURN TO MAIN LOOPI FFITCH LAST CHARACTER SEEN COMMA ? NOI GO BACKI YESI INIZ XFER INDEX BUMP FETCH ROINTER AND LOOPI
1440 1440 1440 1450 1470 1480 150 1510 1520 1530 1540 1550 1550 1550 1550 1550 1550 155	LIA LIA BCC BCC CHE BCC CHE BCC BCC BCC BCC BCC BCC BCC BC	(\$500 TXBUFF,Y \$*0 TXBUFF,Y \$*10 XREF2 \$*19+1 XREF2 TXBUFF,Y \$*19+1 XREF2 TXBUFF,Y \$*19+1 INS2 *19+1 INS3 XREF2 INS3 *REF2 *105 *1	DIATE TO TABLE TOP STANDARD TO TABLE TOP STANDARD TO TABLE TOP STANDARD TO THE STANDARD STANDARD TO THE STANDARD STANDARD TO THE STANDARD STANDARD TO THE STANDARD STANDARD TO STANDARD SAVE CURRENT TABLE TOP STANDARD TO STANDARD TO TABLE TOP STANDARD TO TABLE TOP STANDARD TO TABLE TOP
1440 1440 1440 1450 1470 1480 150 1500 1520 1520 1520 1520 1550 1550	LIA LIA BCC OM BCC OM BCC OM BCC OM BCC STR STR STR STR STR STR STR STR	(4500 TKBUFF,Y 4*0 TKBUFF,Y 4*0 TKBUFF,Y TKBUFF,Y TKBUFF,Y 4*10 TKBUFF,Y 4*19 1 NSU2 1 NSU	DIATIN TO TABLE TOP THIZ YESI LOCK AT NEXT CHARACTER CHECK FOR LINE ↓ NOI => RETURN PETCH CHARACTER TRANSFER TO INBUP LOCK FOR END OF NUMBER BUMP TXBUFF INDEX BUMP TXBUFF INDEX BUMP TXBUFF INDEX BUMP TXBUFF INDEX COMMERT LINE ↓ TO BINARY INSERT LINE ↓ INDEX COMMERT LINE ↓ INDEX COMMERT LINE ↓ TO BINARY INSI ==> INS3 NOI RETURN TO MAIN LOOPI FETCH LAST CHARACTER SEEN COMMA ? NOI GO BACKI YESI INIZ XFFR INDEX BUMP FETCH FOINTER AND LOOPI INIZ SET TO TABLE TOP
1440 1440 1440 1460 1460 1460 1500 1500 1510 1520 1530 1550 1550 1550 1550 1550 1550 155	LIA BCC OME BCC OME BCC OME BCC OME BCC BCC BCC BCC BCC BCC BCC BCC BCC BC	(\$500 TKBUFF,Y *10 TKBUFF,Y *10 TKBUFF,Y *19 TKBUFF,Y *19 *19 *19 *19 *19 *19 *19 *19	DUAL FORMATINE T INIZ VESI LOOK AT NEXT CHARACTER CHECK FOR LINE # NOI => RETURN PETCH CHARACTER TRANSFER TO INBUP LOOK FOR END OF NUMBER BUMP TXBUFF INDEX BUMP TXBUFF INDEX BUMP TXBUFF INDEX BUMP TXBUFF INDEX BUMP TXBUFF INDEX BUMP TXBUFF INDEX CONVERT LINE # OTO BUARY INSERT LINE # PETERNCE RETRIEVE LINE # INDEX CHECK FOR "ON" VESI ==> INS3 NOI RETURN TO MAIN LOOPI FTCH LAST CHARACTER SEEN COMMA ? NOI GO BACKI VESI INIZ XFER INDEX BUMP FETCH BOINTER AND LOOPI INIZ SET TO TABLE TOP 5
1440 1440 1440 1450 1470 1400 1500 1500 1520 1530 1540 1550 1550 1550 1550 1560 1550 1560 1600 1610 1620 1630 1640 1650 1640 1650 1660 1670 1660 1670 1660 1670 170 170 170 1740 1750 1740 1750 1770	LDA LDA BCC OM BCC OM BCC OM BCC STR STR JSI JSI JSI JSI JSI JSI JSI JSI JSI JSI	(\$500 TKBUFF,Y *10 TKBUFF,Y *10 TKBUFF,Y *19 TKBUFF,Y *19 TKBUFF,X *19 *19 TKBUFF,X *19 *19 TKSU *19 *19 *11 TKSU *19 *19 *11 TKSU *19 *11 *18 *18 *19 *11 *18 *18 *18 *18 *18 *18 *18	DUAL TO TABLE TO INIZ YESI LOGK AT NEXT CHARACTER CHECK FOR LINE # NOI => RETURN PETCH CHARACTER TRANSPER TO INBUF LOCK FOR END OF NUMBER BURP TXBUFF INDEX BURP TXBUFF INDEX BURP TXBUFF INDEX BURP TXBUFF INDEX BURP TXBUFF INDEX BURP TXBUFF INDEX CONVERT LINE # TO BUNARY INSERT LINE # A REFERENCE RETRIEVE LINE # INDEX CHECK FOR "ON" YESI INIZ NOI GO BACKI YESI INIZ SET TO TABLE TOP 5 INIZ CLEAR TEST FLAG LOCK AT LINE # LSB
1440 1440 1440 1450 1470 1400 1500 1510 1520 1530 1540 1550 1550 1560 1550 1560 1570 1600 1610 1620 1630 1640 1620 1630 1640 1640 1670 1660 1670 1660 1670 1660 1770 177	LIA LIA BCC COL COL COL COL COL COL COL COL COL C	(\$500 TKBUFF,Y *10 TKBUFF,Y *10 TKBUFF,Y *19 TKBUFF,Y *19 1 NBUFF,Y *19 1 NBUFF,X *19 1 NBUFF,X *19 1 NBUFF,X *19 1 NBUFF,Y *10 TKPPTR 1 NEPTR 1 NETR 1	DIATE TO TABLE TOP DIATEST FLAG LINE SAME AS REFERENCE?
1440 1440 1440 1450 1470 1400 1500 1510 1520 1530 1540 1550 1550 1560 1550 1560 1660 1650 1660 1620 1620 1620 1620 1620 1620 162	NELIN LOY BCC OFFICE BCC DC BCC	(\$500 TKBUFF,Y *10 TKBUFF,Y *10 KREF2 *19+1 KREF2 *19+1 KREF2 *19+1 TKBUFF,Y *19+1 TKS2 *19+1 TKS2 *19+1 TKS2 *19+1 TKS2 *19+1 TKS2 *19+1 TKS2 *19+1 TKS2 *19+1 TKS2 *19+1 TKS2 *19+1 TKS2 *19+1 *10 *10 *10 *10 *10 *10 *10 *1	DUAL TO THE ALL TO THE ALL ALL ALL ALL ALL ALL ALL ALL ALL AL
1440 1440 1440 1450 1470 1480 150 1500 1510 1520 1530 1540 1550 1550 1550 1550 1550 1550 1600 1610 1620 1630 1640 1650 1660 1650 1660 1650 1660 1650 1660 1670; 1710 1740 1750 1740 1750 1750 1750 1770 1780 1780 1780 1780 1810	LIA LIA BCC BCC CHE BCC CHE BCC CHE BCC BCC CHE BCC CHE BCC BCC BCC BCC BCC BCC BCC BC	(\$500 TXBUFF,Y *10 TXBUFF,Y *10 XREF2 *19+1 XREF2 1080UFF,Y *19+1 XREF2 *19+1 INSU *19+1	DIATE TO TABLE TOP STATEST FLAG COMMARY 2 DIATEST FLAG LOCK AT NEXT CHARACTER CHECK FOR LINE # NOI => RETURN FETCH CHARACTER TRANSFER TO INSUP TRANSFER TO INSUP INSUP TOBUPF INDEX BUMP TOBUPF INDEX BUMP TOBUPF INDEX BUMP TOBUPF INDEX BUMP TOBUPF INDEX CONVERT LINE # TO BINARY INSUET INIZ SET TO TABLE TOP INIZ CLEAR TEST FLAG LOCK AT LINE # LSB SAME AS REFERENCE? NOI => FNDLIO YEST CHECK NEB TOOI
1440 1440 1440 1440 1460 1470 1480 150 150 1510 1520 1530 1540 1550 1550 1550 1550 1550 1550 1660 1620 1630 1640 1650 1660 1660 1660 1660 1660 1660 166	LIA BCC BCC BCC BCC BCC BCC BCC BCC BCC BC	(4500 TXBUFF,Y 4*0 TXBUFF,Y 4*0 XREF2 4*19 XREF2 1080UFF,Y 4*19 1080UFF,X 4*19 1080UFF,X 4*19 1080UFF,X 4*19 1082 1	DIATE TO TABLE TO DIATE TO TABLE TO SAME AS REFERENCE NOI => RETURN PETCH CHARACTER TRANSFER TO INSUF LOOK FOR END OF NUMBER BUMP TABUFT INDEX BUMP TABUFT INDEX BUMP TABUFT INDEX BUMP TABUFT INDEX BUMP TABUFT INDEX BUMP TABUFT INDEX COVERT LINE # TO BINARY ONVERT LINE # TO BINARY CONVERT LINE # TO BINARY TINISET LINE # TO BINARY TO RETURN TO MAIN LOOPI FTICH LAST CHARACTER SEEN COMMA ? NOI GO BACKI YESI INIZ XFER INDEX BUMP FECT HORY SET TO TABLE TOP 5 INIZ CLEAR TEST FLAG LOOK AT LINE # LSB SAME AS REFERENCE? NOI => FNULIO YESI CHECK NSB TOOI PETCH NSB COMPARE TO REFERENCE
1440 1440 1440 1450 1470 1480 150 1510 1520 1520 1520 1550 1550 1550	LIA LIA BCC BCC BCC BCC BCC BCC BCC BC	(4500 TKBUFF,Y 4'0 TKBUFF,Y 4'0 TKBUFF,Y TKBUFF,Y 4'9+1 TKBUFF,Y 4'9+1 TKBUFF,Y 4'9+1 TKS2 (TMPDTR DECHEX EFMILIN TMPTR DECHEX EFMILIN TMPTR 45RCSTR PUTPTR 45RCSTR/255 (FVTLIN 45R00 TKBUFF,Y 4', 1 TKBUFF,Y 4', 4', 4', 4', 4', 4', 4', 4',	DUAL TOTAL TOTAL INIZ VEST LOCK AT NEXT CHARACTER CHECK FOR LINE # NOI => RETURN PETCH CHARACTER TRANSFER TO INSUF LOCK FOR END OF NUMBER BUMP TABUFF INDEX BUMP TABUFF INDEX BUMP TABUFF INDEX BUMP TABUFF INDEX COVERT LINE # TO BUNARY INSERT LINE # TO BUNARY CONVERT LINE # TO BUNARY INSERT LINE # TO BUNARY CONVERT LINE # TO BUNARY INSERT LINE # TO BUNARY CONVERT LINE # TO BUNARY CONVERT LINE # TO BUNARY CONVERT LINE # TO BUNARY INSERT LINE # TO BUNARY CONVERT LINE # SEEN CONVERT TO TABLE TOP 5 INIZ CLEAR TEST FLAG LOCK AT LINE # LSB SAME AS REFERENCE? NOI => FURLIO VEST CHECK NSB TOOI PETCH MEB COMPARE TO REFERENCE NO MATCH ===>
1440 1440 1440 1460 1460 150 1500 1510 1520 1530 1550 1550 1550 1550 1550 1550 155	LID BCC BCC BCC BCC BCC BCC BCC BCC BCC BC	(4500 TKBUFF,Y 4*0 TKBUFF,Y 4*0 TKBUFF,Y 5*0 5*19+1 TKBUFF,Y 4*0 1*0 TKBUFF,Y 4*9+1 TKSU 1*19+1 TKSU 1*19+1 TKSU 1*19+1 1*10 1*19+1 1*10 1*1	DUAL TO THE AT T
1430 1440 1450 1460 1470 1480 1500 1500 1520 1530 1540 1550 1550 1550 1560 1550 1560 1570 1580 1600 1610 1620 1630 1640 1650 1660 1650 1660 1660 1670 1710 1740 1750 1740 1750 1740 1750 1740 1750 1750 1770 1760 1770 1760 1770 1780 1770 1780 1830 1830 1830 1830 1830 1850 1850 1850 1850 1850 1850 1850 185	LDA COM BCC COM BCC COM BCC COM BCC STR COM BCC BCC BCC BCC BCC BCC BCC BCC BCC BC	(4500 TKBUFF,Y 4*0 TKBUFF,Y 4*0 KEF2 DKBUFF,Y DKBUFF,Y 4*0 TKBUFF,Y 4*0 TKBUFF,Y 4*10 TKFLAG TKFLAG TKFLAG TKFLAG TKSI 4*500 TELAG 4*500 TELAG 4*500 TELAG 4*500 TELAG 4*500 TELAG 4*00 TELAG 4*00 TELAG 4*00 TELAG 4*00 TELAG 4*00 TELAG 4*00 TELAG 4*00 TELAG 4*00 TELAG 4*00 TELAG 4*00 TELAG 4*00 TELAG 4*00 TELAG 4*00 TELAG 2*00 TELAG 4*00	DUAL FORMATINE T INIZ VESI LOCK AT NEXT CHARACTER CHECK FOR LINE 4 NOI => RETURN PETCH CHARACTER TRANSPER TO INBUP LOCK FOR END OF NUMBER BURP INBUF INDEX BURP INBUF INDEX BURP INBUF INDEX BURP INBUF INDEX BURP INBUF INDEX CONVERT LINE 4 TO BUNARY INSERT LINE 4 LINE 4 BURP FETCH FOINTER NOI GO BACKI VESI INIZ SET TO TABLE TOP 5 INIZ CLEAR TEST FLAG LOCK AT LINE 4 LISB SAME AS REFERENCE? NOI => FNILIO VESI COMPARE TO REFERENCE NO MATCH ==> SHOW MATCH ==> SHOW MATCH ==> SHOW FOR TO A OF REFERENCES COMPARE TO REFERENCES
1440 1440 1440 1450 1470 1460 150 1500 1510 1520 1530 1550 1550 1550 1550 1550 1550 155	LDA COM BCC COM BCC COM BCC COM BCC STR BCC BCC BCC BCC BCC BCC BCC BCC BCC BC	(\$500 TKBUFF,Y *10 TKBUFF,Y *10 TKBUFF,Y *10 TKBUFF,Y *19+1 TKBUFF,X *19+1 TKBUFF,X *19+1 TKS2 *19+1 *1	DUAL TO TABLE TO DIAL THIL TENIZ TENIZ TENIZ TENISTER TO INDUF LOOK FOR END OF NUMBER BUMP TOBUF INDEX BUMP TOBUF INDEX CONVERT LINE & TO BUNARY INSET LINE & LEB STOT TABLE TOP INIZ CLEAR TEST FLAG LOOK AT LINE & LEB SAME AS REFERENCE? NO MATCH -POINT: TO REFERENCE IN MATCH -POINT: TO & CREFERENCE SAVE IT GET CURRENT FOINTER
1430 1440 1450 1460 1470 1480 150 1500 1510 1520 1530 1540 1550 1550 1560 1550 1560 1660 1650 1660 1620 1630 1640 1630 1640 1650 1660 1650 1660 1670 1770 1770 1770 1770 1770 177	LIA LIA BCC BCC BCC BCC BCC BCC BCC BC	(4500 TABUFF,Y ************************************	DUAL TO TABLE TO DIAL THIL TEST LOOK AT NEXT CHARACTER CHECK FOR LINE & NOI => RETURN PETCH CHARACTER TRANSFER TO INDUF LOOK FOR END OF NUMBER BUMP INBUF INDEX BUMP INBUF INDEX BUMP INBUF INDEX BUMP INBUF INDEX CONVERT LINE & TO BINARY INSET LINE & TO BINARY INSECT LINE & TO BINARY INSECT LINE & REFERENCE RETHLEVE LINE & TO BINARY INSECT LINE & REFERENCE RETHLEVE LINE & TO BINARY INSECT LINE & REFERENCE NOI RETURN TO MAIN LOOPI FFICH LAST CHARACTER SEEN COMMA ? NOI GO BACKI YESI INIZ XFFR INDEX BUMP FETCH FOINTER AND LOOPI INIZ SET TO TABLE TOP 5 INIZ CLEAR TEST FLAG LOOK AT LINE & LSB SAME AS REFERENCE? NOI => FNDLIO YESI CHECK MEB TOOI FFICH MEB COMPARE TO REFERENCE NO MATCH => OR REFERENCES SAVE IT GET CURRENT FOINTER SAVE IT
1430 1440 1450 1460 1470 1480 150 1500 1510 1520 1530 1540 1550 1550 1550 1550 1550 1550 155		(4500 TXBUFF,Y **0 TXBUFF,Y **0 **0 **0 **19+1 *XEEF2 **19+1 *XEEF2 **19+1 **10 **10 **19+1 **10 **	DIAL TO TABLE TO SITUATION TO A CONTRACT OF A STATE STANSFER TO INSUP LOOK FOR END OF NUMBER BUMP TOBUF INDEX BUMP TOBUF INDEX BUMP TOBUF INDEX BUMP TOBUF INDEX BUMP TOBUF INDEX BUMP TOBUF INDEX BUMP TOBUF INDEX CONVERT LINE # TO BINARY INSUET LINE # TO BINARY CONVERT LINE # TO BINARY INSUET INIZ SET TO TABLE TOP INIZ SET TO TABLE TOP INIZ CLEAR TEST FLAG LOOMARE TO REFERENCE? NOI => FNDLIO INST COMPARE TO REFERENCE NO MATCHI = FOILT & OF REFERENCES SAVE IT SAVE IT
1440 1440 1440 1450 1470 1480 150 1500 1510 1520 1530 1540 1550 1550 1550 1550 1550 1550 155	LIA LIA BCC BCC BCC BCC BCC BCC BCC BC	(4500 TABUFF,Y 4*0 TABUFF,Y 4*0 TABUFF,Y 4*0 TABUFF,Y TABUFF,Y 4*19+1 TABUFF,Y 4*19+1 TABUFF,Y 4*19+1 TABUFF,Y 4*19+1 TABUFF,Y 4*19+1 TABUFF,Y 4*19+1 TABUFF,Y 4*19+1 TABUFF,Y 4*19+1 TABUFF,Y 4*19+1 TABUFF,Y 4*10 TABUFF,Y 4*00	DUAL TO TABLE TO INIZ VEST LOCK AT NEXT CHARACTER CHECK FOR LINE # NOI => RETURN PETCH CHARACTER TRANSFER TO INSUF LOOK FOR END OF NUMBER BUMP TOBUF INDEX BUMP TOBUF INDEX BUMP TOBUF INDEX BUMP TOBUF INDEX CONVERT LINE # TO BINARY CONVERT LINE # TO BINARY INSERT LINE # TO BINARY CONVERT LINE # TO BINARY NOI RETURN TO MAIN LOOP! FFICH LAST CHARACTER SEEN COMMAR ? NOI GO BACK! INIZ SET TO TABLE TOP 5 INIZ CLEAR TEST FLAG LOOK AT LINE # LSB SAME AS REFERENCE? NOI => FNILIO VESI CHECK MSB TOO! FFICH # SB COMPARE TO REFERENCE NO MAICH ==> SHOW MAICH = -FOURT TO # OF REFERENCES SAVE IT BUMP FOINTER
1440 1440 1440 1460 1460 1460 1500 1500 1510 1520 1530 1550 1550 1550 1550 1550 1550 155	LIA LIA BCC BCC BCC BCC BCC BCC BCC BC	(4500 TKBUFF,Y 4*0 TKBUFF,Y 4*0 TKBUFF,Y 2*0 1*0 TKBUFF,Y 4*0 1*0 1*0 1*0 1*0 1*0 1*0 1*0 1	DUAL TO THE ALSO AND
1430 1440 1450 1460 1470 1460 1500 1500 1520 1530 1540 1550 1550 1560 1550 1560 1550 1560 1660 16	LIA BCC BCC BCC BCC BCC BCC BCC BCC BCC BC	(4500 TKBUFF,Y) 4*0 TKBUFF,Y) 4*0 TKBUFF,Y) 4*0 TKBUFF,Y 4*0 TKBUFF,Y 4*10 TKBUFF,Y 4*19+1 TKBUFF,X 4*19+1 TKSU 1*152 1*1552 1	DIAL POINTER 1 INIZ VEST LOCK AT NEXT CHARACTER CHECK FOR LINE 4 NOI => RETURN PETCH CHARACTER TRANSPER TO INBUF LOCK FOR END OF NUMBER BURP TXBUFF INDEX BURP TXBUFF INDEX BURP INBUF INDEX BURP INBUF INDEX CONVERT LINE 4 TO BURARY INSERT LINE 4 TO BURARY INIZ CLEAR TEST FLAG LOCK AT LINE 4 LSB SAME AS REFERENCE? NOI => FIDILIO INIZ CLEAR TEST FLAG LOCK AT LINE 4 LSB SAME AS REFERENCE? NO MATCH ==> SHOW MATCH ==> SHOW MATCH I FOUNTION 4 OF REFERENCE NO MATCH IT BURP FOINTER BURP FOINTER BURP FOINTER BURP FOINTER BURP FOINTER BURP FOINTER ADDRESS
1440 1440 1440 1450 1470 1460 150 1500 1510 1520 1530 1550 1550 1550 1560 1550 1560 1570 1580 1600 1610 1620 1630 1640 1630 1640 1650 1660 1670 1660 1670 1660 1670 1660 1670 1720 1770 1770 1770 1770 1770 1770 17	LIA BCC OME BCC OME BCC OME BCC STE BCC BCC BCC BCC BCC BCC BCC BCC BCC BC	(4500 TKBUFF,Y 4*0 TKBUFF,Y 4*0 KEF2 5000FF,Y 500FF,Y 500FF,Y 4*0 100FF,X 4*19 100FF,X 4*19 100FF,X 4*19 100FF,X 4*19 100FF,X 4*19 100FF,X 100FF,X 100FF,X 100FF,X 100FF,Y 100FF	DUAL FORMATING T INIZ VEST LOCK AT NEXT CHARACTER CHECK FOR LINE # NOI => RETURN PETCH CHARACTER TRANSPER TO INBUP LOCK FOR END OF NUMBER BURP TXBUFF INDEX BURP TXBUFF INDEX BURP TXBUFF INDEX BURP TXBUFF INDEX BURP TXBUFF INDEX CONVERT LINE # TO BUNARY INSET LINE # TO BUNARY INSET LINE # TO BUNARY INSET LINE # REFERENCE RETHLEVE LINE # REFERENCE RETHLEVE LINE # INDEX COMMAR ? NOI RETURN TO MAIN LOOPI FETCH LAST CHARACTER SEEN COMMA ? NOI RETURN TO MAIN LOOPI FETCH LAST CHARACTER SEEN COMMA ? NOI GO BACKI VESI INIZ XFFR INDEX BURP FETCH FOINTER ND LOOPI INIZ SET TO TABLE TOP SET TO TABLE TOP INIZ CLEAR TEST FLAG LOCK AT LINE # LSB SAME AS REFERENCE? NO MATCH => FINLIO FETCH MSB TOOI PETCH MSB TOOI PETCH # OF REFERENCE SAVE IT BURP FOINTER FOT IT IN ACC. ADD TO FOINTER ADDRESS SAVE IT

1990	FNDL12	LDA NR	PETCH # OF REFERENCES
2000 2010		ASL A STA T2	* 2 SAVE IT
2020		LDA \$\$00	
2040		STA T2+1	HANDLE MSB
2050		LDA 12 CLC	FETCH LSB MUST CLEAR CARRY 1ST!
2070		ADC PUTPIR	ADD POINTER LSB
2080		LDA PUTPIR+1	FETCH MSB
2100		ADC T2+1 STA PUTPTR+1	ADD ANY CARRY SAVE MSB
2120		LDA TFLAG	CHECK MATCH FLAG
2130		JSR MOVEUP	MAKE ROOM FOR ENTRY
2150 2160	FNDL15	LDY #\$00 LDA LNLO	INIZ GET REF. LINE # LSB
2170		STA (PUTPTR),Y	PUT IN TABLE
2190		LDA LNHI	GET MSB
2200 2210		STA (PUTPIR),Y	HAVE IT TOO
2220		LDA (OLDPTR),Y	GET OLD NR
2240		ADC #\$01	+1 1
2250 2260		RTS	AND QUIT
2270	PATT 16	LDA PERBOT	CET TABLE BOTTOM ADDRESS
2290	1144010	OMP PUTPIR	SAME AS PUTPIR?
2300 2310		ENE PNDL17 LDA REFBOT+1	MAYBE, CHECK MSB
2320		CMP PUTPTR+1	SAME?
2340	FNDL17	JMP FNDLI1	CONTINUE: ==>
2350 2360	FNDERR	JSR STROUT BYTE CR, LF	
2370		BYTE '?US ERRO	R #',\$00
2390		JSR STROUT	
2400 2410		LDA LNLO	,\$00
2420		STA RESLO LDA LNHI	
2440		STA RESHI	
2450 2460		JMP CRLP	
2470	, DISLIN	LDA #SRCSTR	RESET TO TABLE TOP
2490		STA PUTPTR	
2500		STA PUTPTR+1	
2520 2530		JSR STROUT BYTE 'LINE NUM	BERS', CR. LF. LF. \$00
2540			
2550	DISL12	LDY \$\$00 STY COUNT	CLEAR & LINES
2550 2560	DISL12	LDY \$\$00 STY COUNT LDA (PUTPTR),Y	CLEAR # LINES GET LINE # LSB
2550 2560 2570 2580	DISL12	LDY \$\$00 STY COUNT LDA (PUTPIR),Y STA RESLO INY	CLEAR # LINES GET LINE # LSB SAVE IT
2550 2560 2570 2580 2590 2590	DISL12	LDY \$\$00 STY COUNT LDA (PUTPTR),Y STA RESLO INY LDA (PUTPTR),Y STA RESHI	CLEAR # LINES GET LINE # LSB SAVE IT GET MSB SAVE IT
2550 2550 2560 2570 2580 2590 2600 2610	DISL12	LDY \$\$00 STY COUNT LDA (PUTPTR),Y STA RESLO INY LDA (PUTPTR),Y STA RESHI INY	CLEAR # LINES GET LINE # LSB SAVE IT GET MSB SAVE IT
2550 2550 2560 2570 2580 2590 2600 2610 2620 2630	DISL12	LDY \$\$00 STY COUNT LDA (PUTPTR),Y STA RESLO INY LDA (PUTPTR),Y STA RESHI INY LDA (PUTPTR),Y STA NR	CLEAR # LINES GET LINE # LSB SAVE IT GET # SB SAVE IT GET # OF REFERENCES SAVE IT
2550 2550 2560 2570 2580 2590 2600 2610 2620 2630 2630 2640 2650	DISL12	LDY #\$00 STY COUNT LDA (PUTPTR),Y STA RESLO INY LDA (PUTPTR),Y STA RESHI INY STA RESHI LDA (PUTPTR),Y STA NR LDA PUTPTR LDA PUTPTR CLC	CLEAR # LINES GET LINE # LSB SAVE IT GET MSB SAVE IT GET # OF REFERENCES SAVE IT GET CURRENT FOINTER
2550 2550 2560 2570 2580 2600 2610 2620 2630 2630 2640 2650 2660 2650	DISL12	LDY 4500 STY COUNT LDA (PUTPTR),Y STA RESLO INY STA RESLO LDA (PUTPTR),Y STA RESHI INY STA NR LDA (PUTPTR),Y STA NR LDA (PUTPTR) LDA (PUTPTR) CLC ADC 4503	CLEAR # LINES GET LINE # LSB SAVE IT GET MSB SAVE IT GET & OF REPERENCES SAVE IT GET CURRENT POINTER ADD HEADER OFFSET
2550 2550 2570 2580 2590 2600 2610 2620 2630 2630 2640 2650 2660 2650 2660 2670 2680	DISL12	LDY 4500 STY COUNT LDA (FUTPTR),Y STA RESLO INY STA RESLO INY STA RESHI INY STA RESHI LDA (FUTPTR),Y STA NR LDA FUTPTR CLC ADC 4503 STA FUTPTR LDA FUTPTR+1	CLEAR # LINES GET LINE # LSB SAVE IT GET MSB SAVE IT GET & OF REPERENCES SAVE IT GET CURRENT FOINTER ADD HEADER OFFSET SAVE IT BACK OUT GET MSB
2550 2560 2570 2580 2600 2610 2620 2620 2630 2640 2650 2660 2660 2660 2660 2670 2680 2670 2670	DISLIZ	LDY 4500 STY COUNT LDA (FUTPTR),Y STA RESLO INY STA RESLO INY STA RESHI INY STA RESHI LDA FUTPTR LDA FUTPTR CLC ADC 4503 STA FUTPTR+1 ADC 4500 STA FUTPTR+1	CLEAR # LINES GET LINE # LSB SAVE IT GET MSB SAVE IT GET 4 OF REFERENCES SAVE IT GET CURRENT FOINTER ADD HEADER OFFSET SAVE IT BACK OUT GET MSB ADD ANY CARRY SAVE TT BACK OUT
2550 2560 2570 2580 2590 2600 2610 2620 2640 2660 2660 2660 2660 2660 266	DISLIZ	LDY 4500 STY COUNT LDA (FUTPTR),Y STA RESLO INY LDA (PUTPTR),Y STA RESHI INY LDA FUTPTR LDA FUTPTR CLC ADC 4503 STA FUTPTR LDA FUTPTR LDA FUTPTR+1 LDA RUTPTR+1 LDA NR BEO DICT 16	CLEAR # LINES GET LINE # LSB SAVE IT GET MSB SAVE IT GET # OF REFERENCES SAVE IT GET CURRENT FOINTER ADD HEADER OFFSET SAVE IT BACK OUT GET MSB ADD ANY CARRY SAVE IT BACK OUT GET # OF REFS.
2550 2560 2570 2580 2610 2620 2620 2620 2620 2620 2620 262	DISLIZ	LDY 4500 STY COUNT LDA (FUTPTR),Y STA RESLO INY LDA (PUTPTR),Y STA RESHI INY LDA (PUTPTR),Y STA NR LDA FUTPTR LDA NR BED DISLI6 JSR STROUT	CLEAR # LINES GET LINE # LSB SAVE IT GET MSB SAVE IT GET (OF REFERENCES SAVE IT GET CURRENT FOINTER ADD HEADER OFFSET SAVE IT BACK OUT GET MSB ADD ANY CARRY SAVE IT BACK OUT GET # OF REFS. 0 ? => DISL16 DISPLAY TEXT
2550 2560 2570 2580 2600 2610 2620 2630 2640 2650 2640 2650 2660 2650 2660 2690 2710 2720 2710 2720 2730 2740 2750	DISLIZ	LDY 4500 STY COUNT LDA (FUTPTR),Y STA RESLO INY LDA (PUTPTR),Y STA RESHI LDA (PUTPTR),Y STA NR LDA (PUTPTR) LDA (PUTPTR) LDA PUTPTR LDA PUTPTR LDA PUTPTR LDA PUTPTR+1 LDA NR BED DISLI6 JSR NIPOUT	CLEAR # LINES GET LINE # LSB SAVE IT GET MSB SAVE IT GET # OF REFERENCES SAVE IT GET CURRENT FOINTER ADD HEADER OFFSET SAVE IT BACK OUT GET MSB ADD ANY CARRY SAVE IT BACK OUT GET # OF REFS. 0 ? ==> DISL16 DISPLAY TEXT ,500 DISPLAY LINE #
2550 2550 2570 2580 2590 2600 2610 2620 2640 2660 2660 2660 2660 2660 266	DISLIZ	LDY 4500 STY COUNT LDA (FUTPTR),Y STA RESLO INY LDA (PUTPTR),Y STA RESHI INY LDA (PUTPTR),Y STA NR LDA (PUTPTR) LDA (PUTPTR) LDA PUTPTR LDA PUTPTR LDA PUTPTR+1 LDA C 4500 STA PUTPTR+1 LDA RUTPTR+1 LDA NR BEQ DISLI6 JSR SIROUT JSR CHLP STA PUTP	CLEAR # LINES GET LINE # LSB SAVE IT GET MSB SAVE IT GET MSB SAVE IT GET CURRENT FOINTER ADD HEADER OPFSET SAVE IT BACK OUT GET MSB ADD ANY CARRY SAVE IT BACK OUT GET 4 OF REFS. 0 ? => DISL16 DISPLAY TEXT ,000 DISPLAY LINE # DO XCR> (LP> PAIR
2550 2550 2550 2580 2590 2690 2610 2620 2630 2630 2630 2630 2630 2630 263	DISLI2	LDY 4500 STY COUNT LDA (FUTPTR),Y STA RESLO INY LDA (PUTPTR),Y STA RESHI INY LDA (FUTPTR),Y STA NR LDA (FUTPTR) LDA (FUTPTR) LDA F033 STA FUTPTR LDA F030 STA FUTPTR+1 LDA C4500 STA FUTPTR+1 LDA R BEQ DISL16 JSR NOMOUT JSR CRLF LDY 4500	CLEAR & LINES GET LINE & LSB SAVE IT GET MSB SAVE IT GET MSB SAVE IT GET CREAT FOINTER ADD HEADER OFFSET SAVE IT BACK OUT GET CREAT FOINTER ADD ANY CARRY SAVE IT BACK OUT GET 4 OF REFS. 0 7 ==> DISLIG DISPLAY TEXT ,500 DISPLAY LINE & DO CRO-CLP> PAIR DO ANOTHER INIZ
2550 2550 2550 2590 2590 2690 2610 2630 2630 2660 2660 2660 2700 2710 2740 2710 2710 2710 2710 2750 2750 2750 2750 2750 2750 2750 275	DISL12	LDY 4500 STY COUNT LDA (FUTPTR),Y STA RESLO INY LDA (FUTPTR),Y STA RESHI INY LDA (FUTPTR),Y STA NR LDA (FUTPTR) LDA (FUTPTR) LDA (FUTPTR) LDA F500 STA FUTPTR+1 LDA F500 STA FUTPTR+1 LDA F500 STA FUTPTR+1 LDA R BEQ DISLI6 JSR NOMOUT JSR CRLF LDY 4500 LDA (FUTTR),Y STA RESLO	CLEAR & LINES GET LINE & LSB SAVE IT GET MSB SAVE IT GET MSB SAVE IT GET CREAT FOINTER ADD HEADER OFFSET SAVE IT BACK OUT GET CREAT FOINTER ADD HEADER OFFSET SAVE IT BACK OUT GET # OF REFS. 0 7 ==> DISLIG DISPLAY TEXT ,500 DISPLAY LINE # DO ANOTHER IN12 FETCH REF. LINE # LSB SAVE IT
2550 2550 2550 2550 2559 2600 2610 2620 2640 2650 2650 2650 2650 2650 2650 2650 265	DISL12	LDY 4500 STY COUNT LDA (PUTPTR),Y STA RESLO INY LDA (PUTPTR),Y STA RESHI INY LDA (PUTPTR),Y STA NR LDA (PUTPTR),Y STA NR LDA (PUTPTR) LDA (PUTPTR) LDA PUTPTR+1 LDA PUTPTR+1 LDA PUTPTR+1 LDA PUTPTR+1 LDA PUTPTR+1 LDA PUTPTR+1 LDA STROUT .BYTE 'LLNE &'' JSR NUMOUT JSR CRLP LDY 4500 LDA (PUTPTR),Y STA RESLO INY (PUTPTR),Y	CLEAR & LINES GET LINE & LSB SAVE IT GET MSB SAVE IT GET MSB SAVE IT GET CREAT FOINTER ADD HEADER OFFSET SAVE IT BACK OUT GET MSB ADD ANY CARRY SAVE IT BACK OUT GET # OF REFS. 0 7 ==> DISLIG DISPLAY TEXT ,500 DISPLAY LINE & DO CANCHER INIZ FETCH REF. LINE & LSB SAVE IT SUMP INDEX DET H DEY LINE & MSB
2550 2550 2550 2550 2559 2600 2610 2620 2630 2640 2650 2650 2670 2650 2670 2750 2770 2770 2770 2770 2770 2770 27	DISL12	LDY 4500 STY COUNT LDA (PUTPTR),Y STA RESLO INY LDA (PUTPTR),Y STA RESHI LDA (PUTPTR),Y STA NR LDA (PUTPTR),Y STA NR LDA (PUTPTR),Y STA NR LDA PUTPTR+1 LDA PUTPTR+1 LDA PUTPTR+1 LDA PUTPTR+1 LDA PUTPTR+1 LDA PUTPTR+1 LDA STA OL JSR STROUT JSR CTLP LDY 4500 LDA (PUTPTR),Y STA RESLI INY LDA (PUTPTR),Y STA RESLI	CLEAR & LINES GET LINE & LSB SAVE IT GET MSB SAVE IT GET MSB SAVE IT GET CREAT FOINTER ADD HEADER OFFSET SAVE IT BACK OUT GET MSB ADD ANY CARRY SAVE IT BACK OUT GET # OF REFS. 0 7 ==> DISLIG DISPLAY LINE & DISPLAY LINE & DO CANCHER INIZ PETCH REF. LINE & LSB SAVE IT SAVE IT SAV
2550 2550 2550 2550 2559 2600 2610 2620 2630 2640 2650 2650 2650 2650 2650 2650 2650 265	DISL12	LDY 4500 STY COUNT LDA (PUTPTR),Y STA RESLO INY LDA (PUTPTR),Y STA RESHI INY LDA (PUTPTR),Y STA NR LDA (PUTPTR),Y STA NR LDA PUTPTR LDA PUTPTR LDA PUTPTR+1 LDA PUTPTR+1 LDA UTPTR+1 LDA UTPTR+1 LDA UTPTR+1 JSR NUMOUT JSR CRLP LDY 4500 STA PUTPTR),Y STA RESHI JSR NUMOUT LDA (PUTPTR),Y STA RESHI JSR NUMOUT LDA (PUTPTR),Y STA RESHI JSR NUMOUT	CLEAR & LINES GET LINE & LSB SAVE IT GET MSB SAVE IT GET MSB SAVE IT GET CREAT FOINTER ADD HEADER OFFSET SAVE IT BACK OUT GET MSB ADD ANY CARRY SAVE IT BACK OUT GET # OF REFS. 0 ? =>> DISLIG DISPLAY LINE & DO ANOTHER DUIZ PETCH REF. LINE & LSB SAVE IT BUMP INDEX FETCH REF. LINE & MSB SAVE IT DISPLAY REF. LINE & MSB SAVE IT DISPLAY REF. LINE & MSB SAVE IT DISPLAY REF. LINE & MSB
2550 2560 2570 2580 2590 2600 2620 2640 2640 2640 2640 2640 264	DISL12	LDY 4500 STY COUNT LDA (PUTPTR),Y STA RESLO INY LDA (PUTPTR),Y STA RESHI INY LDA (PUTPTR),Y STA NR LDA PUTPTR LDA PUTPTR LDA PUTPTR LDA PUTPTR+1 ADC 4500 STA PUTPTR+1 ADC 4500 STA PUTPTR+1 LDA PUTPTR+1 LDA RUTPTR+1 LDA RUTPTR+1 JSR NUMOUT JSR CRLP LDY 4500 LDA (PUTPTR),Y STA RESHI JSR NUMOUT LDA (PUTPTR),Y STA RESHI JSR NUMOUT LDA (PUTPTR),Y STA RESHI JSR NUMOUT LDA (PUTPTR),Y STA RESHI JSR NUMOUT LDA (PUTPTR),Y STA RESHI JSR NUMOUT	CLEAR & LINES GET LINE & LSB SAVE IT GET MSB SAVE IT GET (OF REPERENCES SAVE IT GET CURRENT FOINTER ADD HEADER OFFSET SAVE IT BACK OUT GET MSB ADD ANY CARRY SAVE IT BACK OUT GET # OF REPS. 0 ? =>> DISLIG DISPLAY TEXT \$00 DISPLAY LINE # DO ANOTHER INIZ PETCH REP. LINE # LSB SAVE IT BUMP INDEX FETCH REP. LINE # MSB SAVE IT DISPLAY REP. LINE # MSB SAVE IT SAVE IT
2550 2560 2570 2580 2600 2620 2620 2630 2630 2630 2630 263	DISL12	LDY 4500 STY COUNT LDA (FUTPTR),Y STA RESLO INY LDA (FUTPTR),Y STA RESHI INY LDA (FUTPTR),Y STA RESHI LDA (FUTPTR) LDA (FUTPTR) LDA (FUTPTR) LDA (FUTPTR) LDA UTPTR) LDA UTPTR) LDA UTPTR) STA PUTPTR) LDA UTPTR) JSR CALF LDA (FUTPTR),Y STA RESLO INY STA RE	CLEAR & LINES GET LINE & LSB SAVE IT GET MSB SAVE IT GET ACREAT GET CURRENT FOINTER ADD HEADER OPFSET SAVE IT BACK OUT GET MSB ADD ANY CARRY SAVE IT BACK OUT GET MSB ADD ANY CARRY SAVE IT BACK OUT GET 4 0 P REFS. 0 ? =>> DISLIG DISPLAY LINE & DO <cr>CLP PAIR DO <cr>CLP PAIR DO <cr>CLP PAIR DO <cr>CLP PAIR DO SCR>CLP PAIR DO ANOTHER INIZ PETCH REF. LINE & MSB SAVE IT DISPLAY REF. LINE & MSB SAVE IT SAVE IT SAVE IT DISPLAY REF. LINE & MSB SAVE IT SAVE IT SAVE</cr></cr></cr></cr>
2550 2560 2580 2590 2610 2620 2640 2640 2640 2640 2640 2640 264	DISL12	LDY 4500 STY COUNT LDA (FUTPTR), Y STA RESLO INY LDA (FUTPTR), Y STA RESHI INY LDA (FUTPTR), Y STA RE LDA FUTPTR LDA FUTPTR LDA FUTPTR LDA FUTPTR+1 ADC 4500 STA FUTPTR+1 ADC 4500 STA RUTPTR), Y STA RESLO INY STA	CLEAR ♦ LINES GET LINE ♦ LSB SAVE IT GET MSB SAVE IT GET CURRENT FOINTER ADD HEADER OPFSET SAVE IT ADD HEADER OPFSET SAVE IT BACK OUT GET MSB ADD ANY CARRY SAVE IT BACK OUT GET MSB ADD ANY CARRY SAVE IT BACK OUT GET 4 0 P REFS. 0 ? =>> DISL16 DISPLAY IDECT 0 <0 <0 CR>-CLP> PAIR DO <0 <0 <cr>-CLP> PAIR DO <0 <0 <cr>-CLP> PAIR DO ANOTHER SAVE IT DISPLAY REF, LINE ♦ MSB SAVE IT DISPLAY REF, LINE ♦ MSB SAVE IT DISPLAY REF, LINE ♦ MSB SAVE T DISPLAY REF CONTER ADD REFERENCE LENTH</cr></cr></cr></cr></cr></cr></cr></cr></cr></cr>
2550 2560 2580 2590 2610 2620 2640 2640 2650 2640 2650 2650 2650 2650 2650 2650 2650 265	DISL12	LDY 4500 STY COUNT LDA (FUTPTR),Y STA RESLO INY LDA (FUTPTR),Y STA RESHI INY LDA (FUTPTR),Y STA RESHI LDA FUTPTR LDA FUTPTR LDA FUTPTR LDA FUTPTR+1 ADC 4500 STA FUTPTR+1 ADC 4500 STA FUTPTR+1 ADC 4500 STA FUTPTR+1 JSR NUHOUT JSR CHLF LDA (FUTPTR),Y STA RESLO INY STA RESLO STA FUTPTR LDA FUTPTR LDA FUTPTR+1 ADC 4500 STA FUTPTR+1 INC CUNT	CLEAR & LINES GET LINE & LSB SAVE IT GET MSB SAVE IT GET CURRENT FOINTER ADD HEADER OFFSET SAVE IT ADD HEADER OFFSET SAVE IT BACK OUT GET MSB ADD ANY CARRY SAVE IT BACK OUT GET 4 0 F REFS. 0 ? =>> DISL16 DISPLAY IDENT SAVE IT BURD INDEX FETCH REF. LINE & LSB SAVE IT DISPLAY REF. LINE & MSB SAVE IT SAVE IT SAV
2550 2560 2550 2580 2590 2610 2630 2640 2650 2640 2650 2640 2650 2640 2650 2640 2650 2740 2740 2770 2770 2770 2770 2770 277	DISL12	LDY 4500 STY COUNT LDA (FUTPTR), Y STA RESLO INY LDA (FUTPTR), Y STA RESHI INY LDA (FUTPTR), Y STA RE LDA FUTPTR LDA FUTPTR LDA FUTPTR LDA FUTPTR LDA FUTPTR+1 ADC 4500 STA FUTPTR+1 ADC 4500 STA FUTPTR+1 JSR NUHOUT JSR STROUT JSR STROUT JSR CLP LDA 4500 STA RESLO INY STA RESLO INY S	CLEAR & LINES GET LINE & LSB SAVE IT GET MSB SAVE IT GET CURRENT FOINTER ADD HEADER OFFSET SAVE IT ADD HEADER OFFSET SAVE IT BACK OUT GET MSB ADD ANY CARRY SAVE IT BACK OUT GET # 0 F REFS. 0 ? =>> DISL16 DISPLAY IT BACK OUT GET # 0 F REFS. 0 ? =>> DISL16 DISPLAY LINE # DO <cr><lp> PAIR DO ANOTHER SAVE IT BUMP INDEX FETCH REF. LINE # LSB SAVE IT DISPLAY REF. LINE # MSB SAVE IT DISPLAY REF. LINE # MSB SAVE IT DISPLAY REF. LINE # MSB SAVE IT DISPLAY REF. DISPLA</lp></cr></lp></cr></lp></cr></lp></cr></lp></cr></lp></cr></lp></cr></lp></cr></lp></cr></lp></cr>
22500 25500 25500 25500 26500 26600 26600 26600 26600 26600 26600 26600 26600 26600 26600 26600 26600 27100 27200 27200 27200 27200 27200 27200 27200 27200 27200 27200 27200 27200 27200 27200 27200 27200 27200 22000 27200 220000 22000 22000 220000 20000 2000000	DISL12	LDY 4500 STY COUNT LDA (FUTPTR), Y STA RESLO INY LDA (FUTPTR), Y STA RESHI INY LDA (FUTPTR), Y STA RESHI LDA FUTPTR LDA FUTPTR LDA FUTPTR LDA FUTPTR+1 ADC 4500 STA PUTPTR+1 ADC 4500 STA PUTPTR+1 ADC 4500 STA FUTPTR+1 ADC 4500 STA FUTPTR+1 ADC 4500 STA FUTPTR+1 JSR CRLF LDA (FUTPTR), Y STA RESLO INY STA RESLO INY STA RESLO INY STA RESLO INY STA RESLO INY STA RESLO INY CLC STA FUTPTR LDA FUTPTR	CLEAR # LINES GET LINE # LSB SAVE IT GET MSB SAVE IT GET CURRENT FOINTER ADD HEADER OFFSET SAVE IT BACK OUT GET 40 FRESS ADD HEADER OFFSET SAVE IT BACK OUT GET # 0 FRESS. 0 ? \Longrightarrow DISL16 DISPLAY LINE # DO CRO-CL> PAIR DO ANOTHER INIZ PETCH REP. LINE # LSB SAVE IT BUMP INDEX FETCH REP. LINE # LSB SAVE IT DISPLAY REF. LINE # LSB SAVE IT DISPLAY REF. LINE # MSB SAVE IT TOO SUBM OUTPUT COUNT FETCH # FRIMTED ON LINE DONE 10 ? NO =>> DISLI4
22500 22500 22500 22500 22500 26200 26200 26200 26200 26200 26200 26200 27100 27200 22800 22900	DISL12	LDY 4500 STY COUNT LDA (FUTPTR),Y STA RESLO INY LDA (FUTPTR),Y STA RESHI INY LDA (FUTPTR),Y STA RESHI LDA FUTPTR LDA FUTPTR LDA FUTPTR LDA FUTPTR+1 ADC 4500 STA PUTPTR+1 ADC 4500 STA PUTPTR+1 ADC 4500 STA FUTPTR+1 ADC 4500 STA FUTPTR+1 ADC 4500 STA FUTPTR+1 JSR CRLF LDA (FUTPTR),Y STA RESLO INY STA RESLO INY STA RESLO INY STA RESLO INY STA RESLO INY STA RESLO INY CLC STA FUTPTR+1 ADC 4500 STA FUTPTR+1 ADC 4500 STA FUTPTR+1 ADC 4500 STA FUTPTR+1 ADC 4500	CLEAR ♦ LINES GET LINE ♦ LSB SAVE IT GET MSB SAVE IT GET CURRENT FOINTER ADD HEADER OFFSET SAVE IT BACK OUT GET GET KACK OUT GET # 0 F REFS. ADD HEADER OFFSET SAVE IT BACK OUT GET # 0 F REFS. 0 ? => DISL16 DISPLAY LINE ♦ DO (CR>(L)P PAIR DO ANOTHER INIZ PETCH REF. LINE ♦ LSB SAVE IT BUMP INDEX FETCH REF. LINE ♦ MSB SAVE IT DISPLAY REF. LINE ♦ MSB SAVE IT TOO SUMP OUTPT COUNT FETCH ♦ FRINTED ON LINE DONE 10 ? NO ==> DISL14 YESI
2550 2560 2580 2580 2620 2630 2640 2630 2640 2650 2640 2650 2640 2650 2650 2640 2650 2640 2650 2650 2710 2720 2720 2720 2720 2720 2720 272	DISL12	LDY 4500 STY COUNT LDA (FUTPTR), Y STA RESLO INY LDA (FUTPTR), Y STA RESHI INY LDA (FUTPTR), Y STA RESHI LDA FUTPTR LDA FUTPTR LDA FUTPTR LDA FUTPTR LDA FUTPTR LDA FUTPTR LDA FUTPTR LDA RUTPTR LDA RUTPTR LDA STROLT JSR STROLT JSR STROLT JSR STROLT JSR CRLF LDA (FUTPTR), Y STA RESLO INY STA RESLO INY STA RESLO INY STA RESLO INY STA RESLO INY STA RESLI LDA FUTPTR LDA GUINT LDA FUTPTR LDA GUINT	CLEAR ♦ LINES GET LINE ♦ LSB SAVE IT GET MSB SAVE IT GET CURRENT FOINTER ADD HEADER OPFSET SAVE IT BACK OUT GET & OF REFERENCES SAVE IT BACK OUT GET # OF REFS. 0 ? =>> DISL16 DISPLAY IDENT SAVE IT BACK OUT GET # OF REFS. 0 ? =>> DISL16 DISPLAY LINE ♦ DO CRNCLF> PAIR DO ANOTHER INIZ PETCH REF. LINE ♦ LSB SAVE IT DISPLAY REF. LINE ♦ LSB SAVE IT DISPLAY REF. LINE ♦ MSB SAVE IT TOO SUMP OUTPT COUNT FETCH ♦ FRIMTED ON LINE DONE 10 ? CLEAR COUNTER
2550 2560 2580 2590 2610 2620 2620 2620 2620 2620 2620 262	DISL12 DISL13	LDY 4500 STY COUNT LDA (FUTPTR), Y STA RESLO INY LDA (FUTPTR), Y STA RESHI INY LDA (FUTPTR), Y STA RESHI LDA FUTPTR LDA FUTPTR LDA FUTPTR LDA FUTPTR LDA FUTPTR LDA FUTPTR LDA RUTPTR LDA RUTPTR LDA RUTPTR LDA RUTPTR LDA RUTPTR JSR STROUT JSR STROUT JSR CRLF JSR CRLF JSR CRLF JSR CRLF JSR CRLF JSR CRLF LDA (FUTPTR), Y STA RESHI LDA FUTPTR LDA FUTPTR LD	CLEAR ♦ LINES GET LINE ♦ LSB SAVE IT GET MSB SAVE IT GET CURRENT FOINTER ADD HEADER OPPSET SAVE IT BACK OUT GET & OF REPERENCES SAVE IT BACK OUT GET # OF REPS. ADD ANY CARRY SAVE IT BACK OUT GET # OF REPS. 0 ? => DISLIG DISPLAY IDENT SAVE IT DO (CROCLE) FAIR DO ANOTHER INIZ FETCH REP. LINE ♦ LSB SAVE IT DISPLAY REF. LINE ♦ LSB SAVE IT DISPLAY REF. LINE ♦ LSB SAVE IT DISPLAY REF. LINE ♦ MSB SAVE IT DISPLAY REF. LINE ♦ MSB SAVE IT DISPLAY REF. LINE ♦ MSB SAVE IT DISPLAY REF. LINE ♥ MSB SAVE IT DISPLAY REF. LINE ♥ MSB SAVE IT DISPLAY REF. LINE ♥ MSB SAVE IT TOO BURP OUTHOT COUNT FETCH ♥ FRIMTED ON LINE DONE 10 ? NO =>> DISLI4 YESI DO A (CROCLF) INIZ CLEAR COUNTER ADD SKUP A BIT GET A SPACE
2550 2560 2580 2590 2610 2620 2620 2620 2620 2620 2620 262	DISL12 DISL13	LDY 4500 STY COUNT LDA (FUTPTR), Y STA RESLO INY LDA (FUTPTR), Y STA RESHI INY LDA (FUTPTR), Y STA RESHI LDA FUTPTR LDA FUTPTR LDA FUTPTR LDA FUTPTR LDA FUTPTR LDA RUTPTR LDA RUTPTR LDA RUTPTR LDA RUTPTR LDA RUTPTR LDA RUTPTR LDA RUTPTR JSR SIROUT JSR CRLF JSR CRLF JSR CRLF JSR CRLF JSR CRLF JSR CRLF LDA (FUTPTR), Y STA RESHI LDA FUTPTR LDA GUINT LDA CLA STA CUINT LDA SOO STA FUTPTR LDA SOO STA CUINT LDA SOO STA SO	CLEAR ♦ LINES GET LINE ♦ LSB SAVE IT GET MSB SAVE IT GET OF REPERENCES SAVE IT GET CURRENT FOINTER ADD HEADER OPFSET SAVE IT BACK OUT GET & OF REPS. ADD ANY CARRY SAVE IT BACK OUT GET & OF REPS. 0 ? => DISLIG DISPLAY IDENT SAVE IT DO (CROCLE) PAIR DO ANOTHER INIZ FETCH REP. LINE ♦ LSB SAVE IT DISPLAY REF. LINE ♦ LSB SAVE IT DISPLAY REF. LINE ♦ LSB SAVE IT DISPLAY REF. LINE ♦ MSB SAVE IT TOO BURP OUTFOT COUNT FETCH ♦ FRIMTED ON LINE DONE 10 ? NO =>> DISLI4 YESI DO A (CROCLE)> INIZ CLEAR COUNTER AND SKIP A BIT GET A SPACE FRIMT IT FUICE. IN FACT
2550 2560 2580 2590 2610 2620 2620 2640 2620 2640 2650 2640 2770 2740 2770 2770 2770 2770 2770 27	DISLI2 DISLI3 DISLI4	LDY 4500 STY COUNT LDA (FUTPTR), Y STA RESLO INY LDA (FUTPTR), Y STA RESHI INY LDA (FUTPTR), Y STA RESHI LDA FUTPTR LDA FUTPTR LDA FUTPTR LDA FUTPTR LDA RUTPTR LDA RUTPTR LDA RUTPTR LDA RUTPTR LDA RUTPTR LDA RUTPTR LDA RUTPTR LDA RUTPTR JSR SIROUT JSR SIROUT JSR CRLF JSR CRLF JSR CRLF JSR CRLF JSR CRLF JSR CRLF LDA (FUTPTR), Y STA RESHI LDA FUTPTR LDA FUTPTR LD	CLEAR ♦ LINES GET LINE ♦ LSB SAVE IT GET MSB SAVE IT GET OF REPERENCES SAVE IT GET CURRENT FOINTER ADD HEADER OPFSET SAVE IT BACK OUT GET & OF REPS. ADD ANY CARRY SAVE IT BACK OUT GET & OF REPS. 0 ? => DISLIG DISPLAY LINE ♦ DO <rnc lf=""> PAIR DO ANOTHER INIZ FETCH REP. LINE ♦ LSB SAVE IT DISPLAY REF. LINE ♦ LSB SAVE IT DISPLAY REF. LINE ♦ LSB SAVE IT DISPLAY REF. LINE ♦ MSB SAVE IT DISPLAY REF. LINE ♦ MSB SAVE IT DISPLAY REF. LINE ♥ MSB SAVE IT TOO SUPP OUTFOT COUNT FETCH ♥ FRIMTED ON LINE DONE 10 ? NO ==> DISLI4 YESI DO A <cr>CLF> INIZ CLEAR COUMTER AND SKIP A BIT GET A SPACE FRIMT IT THICE, IN FACT SHOW WE DID A REFERENCE</cr></rnc>

Listing continued

2050		TCD	COLE	DO FINAL (CR)(LF)
3050		JSR	CRLF	DO TWO
3070	DISL16	LDA	REFBOT	GET BOTTOM ADDR.
3080		CMP	PUTPIR	SAME AS FUTPTR?
3090		ENE	DISLI/	NO! CONTINUE: ==> MAYBE, CHECK MSB
3110		OMP	PUTPTR+1	SAME ?
3120		BEQ	DISL18	YESI GO TO VARIABLES
3130	DISLI7	JMP	DISL12	NOT DONE: ==> DISL12
3140	DISLIS	JMP	VAR	GO DO VARIABLES NOW
31201	MOUTIP	LDA	REFROM	GET TABLE BOTTOM ADDR.
3170		STA	FETFTR	GIVE TO FETPIR
3180		LDA	REFBOT+1	
3190		STA	FETPIR+1	
3200	MOVEL	LDX	¥३00 (PETTETR),¥	PETCH & CHARACTER
3220		LDY	\$\$02	+ 2
3230		STA	(FETPTR),Y	MOVE IT UP TWO BYTES
3240		LDA	FEIPIR	GET PETPTR
3250		CMP	FUTPIR	AT INSERTION FOINT ?
3270		LDA	FETFTR+1	MAYBE, GET MSB
3280		СМР	PUTPTR+1	SAME ?
3290		BEQ	MOVE4	YES1 ==> MOVE4
3300	MOVE2	BNE	FEIPIR MOVES	NOT SOO2 ==> MOVE3
3320		DEC	FEIFIR+1	YESI DECREMENT MSB
3330	MOVE3	DEY		DECREMENT LSB
3340		STY	FEIPIR	SAVE IT BACK OUT
3350	MERTEA	JPLP	PLOVEL DECEMPTOR	CET BOTTON ADDR LSB
3370		ac		(BARN) IAL
3380		ADC	\$\$02	ADD 2
3390		STA	REFBOT	SAVE IT
3400		BCC	MLVE5	WATCH FUR PAGING
3420		LDA	REFBOT41	FETCH RESULT
3430		OP	MAXMEM	AT MEMORY TOP?
3440		BED	MOVES	=, OK ==> MOVES
3450		BCS	OMERR	PAST! ==> ERROR
3460	MOVES	RIS		AND QUIT
3480	OMERR	JSR	STRUT	TELL USER
3490		BY	TE CR, LF	
3500		.BY	TE 'OUT OF M	EMORY', CR, LP, \$00
3510		LDA	DEPAUL+1	GET CONSQLE DV#
3530		JMP	WARMINS	RETURN TO MENU
3540	,	0.4		
3550	VAR	LDA	STIK	LOAD 1ST TRACK OF TEXT
3560		CMCP DMCP	TRAKX	COMPARE TO CURRENT TRACK
3580		STA	TRAKY	GIVE IT TO 65D
2500				
3220		JSR	SWAP	* DOS CONTEXT *
3600		JSR JSR	SWAP SEEKX	* DOS CONTEXT * MOVE HEAD TO TRACK
3590 3600 3610		JSR JSR JSR	SWAP SEEKX READ	* DOS CONTEXT * MOVE HEAD TO TRACK READ TRACK + LANTINGE CONTEXT *
3600 3610 3620 3630	VARI	JSR JSR JSR JSR LDA	SWAP SEEKX READ SWAP SOF	* DOS CONTENT * MOVE HEAD TO TRACK READ TRACK * LANGUAGE CONTENT * GET START OF TENT
3600 3610 3620 3630 3640	VARL	JSR JSR JSR JSR LDA STA	SWAP SEEKX READ SWAP SOP INDEX	* DOS CONTEXT * MOVE HEAD TO TRACK READ TRACK * LANGUAGE CONTEXT * GET START OF TEXT GIVE TO INDEX
3610 3610 3620 3630 3640 3650	VARL	JSR JSR JSR LDA STA LDA	SWAP SEEKX READ SWAP SOF INDEX SOF+1	* DOS CONTEXT * MOVE HEAD TO TRACK READ TRACK * LANGUAGE CONTEXT * GET START OF TEXT GIVE TO INDEX
3610 3610 3620 3630 3640 3650 3660	VARI	JSR JSR JSR JSR LDA STA LDA STA	SWAP SEEKX READ SWAP SOF INDEX SOF+1 TXTPTR+1	* DOS CONTEXT * NOVE HEAD TO TRACK READ TRACK * LANGUAGE CONTEXT * GET START OF TEXT GIVE TO INDEX
3610 3610 3620 3630 3640 3650 3660 3660 3670	VAR1	JSR JSR JSR LDA STA LDA STA LDA STA	SWAP SEEXX READ SWAP SOF INDEX SOF+1 TXTPTR+1 \$SRCSTR PEPBOT	* DOS CONTENT * MOVE HEAD TO TRACK READ TRACK * LANGUAGE CONTENT * GIVE TO INDEX RESET BOTTOM OF
3590 3610 3620 3630 3640 3650 3660 3660 3660 3680 3690	VARI	JSR JSR JSR JSR LDA STA LDA STA LDA STA LDA	SWAP SEEXX READ SWAP SOF INDEX SOF+1 TXTPTR+1 #SRCSTR REFBOT #SRCSTR/256	* DOS CONTENT * MOVE HEAD TO TRACK READ TRACK * LANGUAGE CONTENT * GET START OF TEXT GIVE TO INDEX RESET BOTTOM OF RESET BOTTOM OF REFERENCE TABLE
3590 3610 3610 3620 3630 3650 3650 3660 3670 3680 3690 3700	VARI	JSR JSR JSR LDA STA LDA STA LDA STA LDA STA LDA STA	SMAP SEEKX READ SWAP SOF INDEX SOF+1 TXTFTR+1 \$SRCSTR REFBOT \$SRCSTR/256 REFBOT+1	* DOS CONTEXT * MOVE HEAD TO TRACK READ TRACK * LANGUGE CONTEXT * GET START OF TEXT GIVE TO INDEX RESET BOTTOM OF REFERENCE TABLE
3590 3600 3610 3620 3630 3640 3650 3660 3670 3680 3690 3700 3700	VARI	JSR JSR JSR JSR JSR JSR JSR JSR JSR JSR	SWAP SEEXX READ SWAP SOF INDEX SOF+1 TXTPTR+1 \$SRCSTR REFBOT \$SRCSTR/256 REFBOT+1 \$S00	* DOS CONTEXT * MOVE HEAD TO TRACK READ TRACK * LANGUAGE CONTEXT * GET START OF TEXT GIVE TO INDEX RESET BOTTOM OF REPERENCE TABLE
3590 3600 3610 3620 3630 3640 3650 3660 3670 3680 3690 3700 3710 3720	VARI	JSR JSR JSR LDA STA LDA STA LDA STA LDA STA LDA STA	SWAP SEEKX READ SWAP SOF INDEX SOF+1 TXTPTR+1 \$SRCSTR \$SRCSTR #SRCSTR #SRCSTR #SRCSTR \$SRC TXTPTR TXTPTR TXTPTR	* DOS CONTEXT * MOVE HEAD TO TRACK READ TRACK * LANGUAGE CONTEXT * GIVE TO INDEX RESET BOITION OF REFERENCE TABLE RESET TXTPTR TO BUFFER DITZ DITTAL TOPE
3590 3600 3610 3620 3630 3640 3650 3660 3670 3680 3690 3700 3710 3720 3730 3730	VARI	JSR JSR JSR LDA STA LDA STA LDA STA LDA STA STA JSR	SWAP SEERCK READ SWAP SOF INDEX SOF+1 TXTPTR+1 \$SRCSTR REFBOT \$SRCSTR/256 REFBOT+1 \$SRCSTR/256 REFBOT+1 TYPE STRCUT	* DOS CONTEXT * MOVE HEAD TO TRACK READ TRACK * LANGUAGE CONTEXT * GIVE TO INDEX RESET BOTTOM OF REFERENCE TABLE RESET TXTPTR TO BUFFER INIZ INITIAL TYPE
3590 3600 3610 3620 3630 3650 3650 3660 3660 3670 3680 3690 3710 3710 3720 3730 3730 3740 3750	VARI	JSR JSR JSR JSR LDA STA LDA STA LDA STA LDA STA STA JSR JSR .BY	SWAP SEERON READ SWAP SOF INDEX SOF+1 TXTPTR+1 #SRCSTR REFBOT #SRCSTR/256 REFBOT+1 #SRCSTR/256 REFBOT+1 #SRCSTR/256 REFBOT+1 #SRCSTR/256 STRCUT TYPE STRCUT	* DOS CONTENT * MOVE HEAD TO TRACK READ TRACK * LANGUAGE CONTENT * GET START OF TEXT GIVE TO INDEX RESET BOTTOM OF REPERENCE TABLE RESET TATPTR TO BUPPER INIZ INITIAL TYPE RIABLES', CR, LF, LF, \$00
3590 3600 3620 3630 3640 3650 3660 3660 3670 3680 3690 3700 3710 3720 3730 3740 3750 3750 3760	VARI	JSR JSR JSR JSR JSR JSR JSR LDA STA LDA STA LDA STA JSR JSR .BY	SWAP SEEDCO READ SWAP SOF INDEX SOF+1 INTFIR+1 4SNCSTR 4SRCSTR 4SRCSTR/256 REFPOT+1 4SRCSTR/256 REFPOT+1 4SRCSTR/256 REFPOT+1 FECR,LF, VA	* DOS CONTEXT * MOUE HEAD TO TRACK READ TRACK * LANGUAGE CONTEXT * GET START OF TEXT GIVE TO INDEX RESET BOTTOM OF REFERENCE TABLE RESET TXTPTR TO BUFFER INIZ INITIAL TYPE RIABLES', CR, LF, LF, \$00
3590 3610 3610 3620 3630 3650 3650 3660 3660 3690 3700 3700 3710 3720 3710 3720 3750 3750 3760	VAR1	JSR JSR JSR LDA STA LDA STA LDA STA LDA STA LDA STA JSR JSR JSR JSR JSR	SWAP SEERCK READ SWAP SOF INDEX SOF+1 INTFTR+1 #SRCSTR/256 REFBOT+1 #SRCSTR/256 REFBOT+1 #SRCSTR/256 REFBOT+1 #SRC0 TXTPER STRCUT FE CR, LP, 'VA GETLIN #SDO	* DOS CONTEXT * MOVE HEAD TO TRACK READ TRACK * LANSUNGE CONTEXT * GIVE TO INDEX RESET BOITION OF REFERENCE TABLE RESET TXTPTR TO BUFFER INIZ INITIAL TYPE RIABLES', CR, LF, LF, \$00 GET A LINE OF TEXT INIZ
3590 3600 3610 3620 3630 3650 3660 3650 3660 3690 3700 3710 3720 3730 3750 3750 3770 3770 3770 3770 377	VAR1	JSR JSR JSR LDA STA LDA STA LDA STA LDA STA LDA STA STA JSR LDA STA JSR LDA LDA STA LDA LDA LDA LDA LDA LDA STA LDA LDA STA LDA LDA STA LDA STA LDA STA LDA STA LDA STA LDA STA LDA STA LDA STA LDA STA STA LDA STA STA LDA STA STA STA LDA STA STA STA STA STA STA STA STA STA ST	SWAP SEERCK READ SWAP SOF INDEX SOF+1 TXTPTR+1 \$SRCSTR REFBOT \$SRCSTR/256 REFBOT+1 \$SRCSTR/256 TXTPTR TYPE STRCUT TYPE STRCUT TE CR,LP, 'VA GETLIN \$\$00 NLAB	* DOS CONTEXT * MOVE HEAD TO TRACK READ TRACK * LANGUAGE CONTEXT * GIVE TO INDEX RESET BOTTOM OF REFERENCE TABLE RESET TXTPTR TO BUFFER INIZ INITIAL TYPE RIABLES', CR, LF, LF, \$00 GET A LINE OF TEXT INIZ CHECK FOR E.O.F.
3590 3610 3610 3620 3630 3640 3660 3660 3670 3680 3700 3770 3730 3740 3750 3770 3750 3770 3770 3780 3790 3800	VAR1	JSR JSR JSR JSR JSR JSR JSR JSR JSR JSR	SWAP SEERCK READ SWAP SOF INDEX SOF+1 TXTPTR+1 \$SRCSTR REFBOT \$SRCSTR/256 REFBOT+1 \$SRCSTR/256 REFBOT+1 \$SOO TXTPIR TYPE STRCUT TE CR,LF, 'VA GETLIN \$SOO NLAH XVARL	* DOS CONTEXT * MOVE HEAD TO TRACK READ TRACK * LANGUAGE CONTEXT * GET START OF TEXT GIVE TO INDEX RESET BOTTOM OF REPERBINCE TABLE RESET TXTPIR TO BUFFER INIZ INITIAL TYPE RIABLES', CR, LF, LF, \$00 GET A LINE OF TEXT INIZ CHECK FOR E, O, F.
3610 3610 3620 3630 3650 3650 3660 3660 3700 3710 3720 3770 3770 3770 3770 3770 3770 377	VARI J	JSR JSR JSR JSR JSR JSR JSR JSR JSR JSR	SWAP SEERCK READ SWAP SOF INDEX SOF INDEX SOF+1 TXTPTR+1 \$SCSTR REFBOT \$SCSTR/256 REFBOT+1 \$SOO TXTPTR TYPE STROUT STROUT TYPE STROUT STR	* DOS CONTENT * MOVE HEAD TO TRACK READ TRACK * LANGUAGE CONTENT * GET START OF TEXT GIVE TO INDEX RESET BOTTOM OF REPERENCE TABLE RESET TXTPTR TO BUPFER INIZ INITIAL TYPE RIABLES', CR, LF, LF, \$00 GET A LINE OF TEXT INIZ CHECK FOR E.O.F.
3590 3600 3610 3620 3650 3650 3650 3650 3660 3760 3710 3770 3770 3770 3770 3770 3770 377	VARI , XVAR	JSR JSR JSR JSR JSR STA STA STA STA STA STA STA STA STA STA	SWAP SEERCK READ SWAP SOF INDEX SOF+1 INDEX SOF+1 INTFTR+1 #SRCSTR/256 REFBOT+1 #SRCSTR/256 REFBOT+1 #SRCSTR/256 REFBOT+1 #SRCUT TYPE STRCUT REFBOT+1 #SOO NUTY STRCUT REFBOT+1 #SOO NUTY SUBS STRCUT REFBOT+1 STRCUT REFBOT+1 #SOO NUTY REFBOT+1 STRCUT REFBOT+1 STRCUT REFBOT+1 STRCUT REFBOT+1 STRCUT REFBOT+1 STRCUT REFBOT+1 STRCUT REFBOT+1 STRCUT REFBOT+1 STRCUT REFBOT+1 STRCUT REFBOT+1 STRCUT REFBOT+1 STRCUT REFBOT+1 STRCUT STRC	* DOS CONTEXT * MOVE HEAD TO TRACK READ TRACK * LANSUNGE CONTEXT * GIVE TO INDEX RESET BOITOM OF REFERENCE TABLE RESET TXTPTR TO BUFFER INIZ INITIAL TYPE RIABLES', CR, LF, LF, \$00 GET A LINE OF TEXT INIZ CHECK FOR E.O.F.
3590 3610 3620 3650 3650 3650 3650 3650 3660 3760 3710 3770 3770 3770 3770 3770 3770 377	VARI , XVAR	JSR JSR JSR JSR JSR JSR JSR STA STA STA STA STA STA STA STA STA STA	SWAP SEEKCK READ SWAP SOF INDEX SOF+1 TXTPTR+1 \$SRCSTR REFBOT \$SRCSTR/256 REFBOT+1 \$SRCSTR/256 REFBOT+1 \$SRCSTR/256 STRUT TYPE STRUT TE CR,LP, 'VA GETLIN \$\$00 NLAE XVAR1 NLAL XVAR1 DISVAR TOBUFP, Y	* DOS CONTEXT * MOVE HEAD TO TRACK READ TRACK * LANGUAGE CONTEXT * GIVE TO INDEX RESET BOITION OF REFERENCE TABLE RESET TXTPIR TO BUFFER INIZ INITIAL TYPE RIABLES', CR, LP, LP, \$00 GET A LINE OF TEXT INIZ CHECK FOR E.O.P.
36300 3610 3620 3630 3660 3660 3660 3660 3660 3700 3710 3720 3730 3740 3750 3740 3750 3740 3750 3750 3760 3750 3760 3750 3760 3750 3760 3750 3760 3750 3760 3750 3760 3750 3760 3750 3760 3750 3760 3750 3760 3750 3750 3750 3750 3750 3750 3750 375	VAR1 , XVAR XVAR1	JSR JSR JSR LDA STA STA STA STA STA STA STA STA STA ST	SWAP SEEKCK READ SWAP SOF INDEX SOF+1 TXTPTR+1 \$SRCSTR REFBOT \$SRCSTR/256 REFBOT+1 \$SRCSTR/256 REFBOT+1 \$SOO TXTPIR TYPE STRCUT TE CR,LP, 'VA GETLIN \$SOO NLAH XVAR1 DISVAR DISVAR DISVAR XVAR1	* DOS CONTEXT * MOVE HEAD TO TRACK READ TRACK * LANGUAGE CONTEXT * GET START OF TEXT GIVE TO INDEX RESET BOTTOM OF REPERBACE TABLE RESET TATPTR TO BUFFER INIZ INITIAL TYPE RIABLES', CR, LF, LF, \$00 GET A LINE OF TEXT INIZ CHECK FOR E. O. F. DONE! FRINT TABLE LOOK AT CHARACTER E.O.L.?? => XVAR
3690 3610 3620 3630 3640 3650 3660 3660 3660 3700 3710 3720 3730 3740 3750 3770 3770 3770 3770 3770 3770 377	VARI XVARI	JSR	SWAP SEIRCX READ SWAP SOF INDEX SOF+1 TXTPTR+1 \$SRCSTR REFBOT \$SRCSTR/256 REFBOT+1 \$SOC TXTPTR TYPE STRCUT TE CR,LF, 'VA GETLIN \$SOO NLAH NLAL XVAR1 DISVAR TXBUFF, Y XVAR REBUTK WIGE	* DOS CONTECT * MOVE HEAD TO TRACK READ TRACK * LANGUAGE CONTEXT * GET START OF TEXT GIVE TO INDEX RESET BOTTOM OF REFERENCE TABLE RESET TATPIR TO BUFFER INIZ INITIAL TYPE RIABLES', CR, LF, LF, \$00 GET A LINE OF TEXT INIZ CHECK FOR E. O.F. DONE! PRINT TABLE LOOK AT CHARACTER EJOAL? =>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
3690 3610 3620 3630 3640 3650 3660 3660 3660 3700 3710 3770 3770 3770 3770 3770 377	VAR1 XVAR XVAR1	JSR	SWAP SEENCX READ SOF INDEX SOF INDEX SOF+1 INTFTR+1 #SNCSTR/256 REFBOT+1 #SNCSTR/256 REFBOT+1 #SNCSTR/256 REFBOT+1 #SNCSTR/256 REFBOT+1 #SNCSTR/256 MIAL STARUT REFBOT+1 #SNC STRCUT REFBOT+1 #SNC STRCUT REFBOT+1 #SNC STRCUT REFBOT+1 #SNC STRCUT REFBOT+1 #SNC STRCUT REFBOT+1 #SNC STRCUT REFBOT+1 STRCUT STRCUT REFBOT+1 STRCUT REFBOT+1 STRCUT REFBOT+1 STRCUT REFBOT+1 STRCUT REFBOT+1 STRCUT STRCUT REFBOT+1 STRCUT REFBOT+1 STRCUT STRCUT REFBOT+1 STRCUT STRCUT REFBOT+1 STRCUT REFBOT+1 STRCUT REFBOT+1 STRCUT REFBOT+1 STRCUT REFBOT+1 STRCUT REFBOT+1 STRCUT REFBOT+1 STRCUT STRCUT REFBOT+1 STRCUT STRCUT STRCUT REFBOT+1 STRCUT ST	* DOS CONTEXT * MOVE HEAD TO TRACK READ TRACK * LANSUNGE CONTEXT * GIVE TO INDEX RESET BOITOM OF REFERENCE TABLE RESET TXTPTR TO BUFFER INIZ INITIAL TYPE RIABLES', CR, LF, LF, \$00 GET A LINE OF TEXT INIZ CHECK FOR E. O. F. DONEL PRINT TABLE LOOK AT CHARACTER E.O.L.?? ==> XVAR REMARK? YESI SKIF TO NEXT LINE STRING LIFEDAP?
36300 36610 36610 36630 36630 36630 36630 36630 36630 37600 37700 37800 37700 37800 38800 38000 38000 38000 380000	VAR1 XVAR XVAR	JSR	SWAP SEEKCK READ SWAP SOF INDEX SOF+1 TXTPTR+1 #SRCSTR #SRCSTR/256 REFBOT+1 #SRCSTR/256 REFBOT+1 #SRCSTR/256 TXTPTR TYPE STRCUT TTPE CR,LP, 'VA GETLLIN #\$00 NLAE XVAR1 DISVAR TXCBUFY,Y XVAR #REMIX XVAR3	* DOS CONTEXT * MOVE HEAD TO TRACK READ TRACK * LANGUAGE CONTEXT * GIVE TO INDEX RESET BOTTOM OF REPERENCE TABLE RESET TXTPTR TO BUFFER INIZ INITIAL TYPE RIABLES', CR, LF, LF, \$00 GET A LINE OF TEXT INIZ CHECK FOR E.O.F. DONE! FRINT TABLE LOCK AT CHARACTER E.O.L.?? ==> XVAR REMARK? YES! => XVAR3
36300 36610 36620 36630 36630 36630 36630 36630 36630 37600 37700 37800 37700 37800 38800 38800 389000 39000 39000 39000 39000 39000 39000 39000 39000 39000 390000 390000 3900000000	VAR1 XVAR XVAR	JSSSSAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA	SWAP SWAP SEEKCK READ SWAP SOF INDEX SOF+1 TXTPTR+1 #SRCSTR #SRCSTR/256 REFBOT #SRCSTR/256 REFBOT #SRCSTR/256 REFBOT #SRCSTR/256 STRCUT TYPE STRCUT TYPE STRCUT TE CR,LP, 'VA GETLIN #\$00 NLAH XVAR1 DISVAR TXBUFP, Y XVAR #REMIX XVAR1 XVAR3 CASECK	* DOS CONTEXT * MOVE HEAD TO TRACK READ TRACK * LANGUAGE CONTEXT * GIVE TO INDEX RESET BOTTOM OF REFERENCE TABLE RESET TXTPTR TO BUFFER INIZ INITIAL TYPE RIABLES', CR, LF, LF, \$00 GET A LINE OF TEXT INIZ CHECK FOR E.O.F. DONE! PRINT TABLE LOCK AT CHARACTER E.O.L.?? ==> XVAR REMARK? YES! SKIP TO NEXT LINE STRING LITERAL? YES! ==> XVAR CORRET ANY LOWER CASE
3590 3600 3610 3620 3630 3650 3660 3660 3760 3770 3770 3770 3770 377	VAR1 , XVAR XVAR1	JSRSRAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA	SWAP SWAP SEENCK READ SWAP SOF INDEX SOF INDEX SOF INTFIN #SNCSTR/256 REFPOT+1 #SNCSTR/256 REFPOT+1 #SNCSTR/256 REFPOT+1 #SNC TXTPIE STRCUT TYPE STRCUT TYPE STRCUT TYPE STRCUT REFPOT+1 #SNC TYPE STRCUT STRCUT ST	* DOS CONTECT * MOVE HEAD TO TRACK READ TRACK * LANGUAGE CONTEXT * GET START OF TEXT GIVE TO INDEX RESET BOTTOM OF REFERENCE TABLE RESET TATPTR TO BUFFER INIZ INITIAL TYPE RIABLES', CR, LF, LF, \$00 GET A LINE OF TEXT INIZ CHECK FOR E. O.F. DONE! PRINT TABLE LOOK AT CHARACTER E. O.L.?" => XVAR REMARK? YES! SKIP TO NEXT LINE STRING LITERAL? YES! ==> XVAR3 CORRECT ANY LOWER CASE CHECK FOR LETTER
3590 3600 3610 3620 3630 3650 3660 3660 3700 3710 3770 3770 3770 3770 3770 377	VAR1 XVAR		SWAP SWAP SEENCK READ SOF INDEX SOF INDEX SOF INDEX SOF INTFIR TEST SOF INTFIR TYPE STROUT TECR, LP, 'VA GETLIN \$\$00 NLAH XVARI NLAL XVARI NLAL XVARI DISVAR TEDUFP, Y XVAR SCASDCK \$'2 I'Z+1	* DOS CONTEXT * MOVE HEAD TO TRACK READ TRACK * LANSUNGE CONTEXT * GIVE TO INDEX RESET BOITION OF REFERENCE TABLE RESET TATIFUR TO BUFFER INIZ INITIAL TYPE RIABLES', CR, LF, LF, S00 GET A LINE OF TEXT INIZ CHECK FOR E. O.F. DONEL PRINT TABLE LOOK AT CHARACTER E.O.L.?? ==> XVAR CONECT ANY LOWER CASE CHECK FOR LETTER NO ==>
36300 36100 36200 36300 36500 36600 36600 36600 37000 37100 37300 37700 37800 38800 38800 38800 38800 38800 38800 38800 38800 38900 39900 39900	VAR1 XVAR XVAR1	JSR ALASA AL	SWAP SWAP SEEKC READ SWAP SOF INDEX SOF+1 TXTPTR+1 \$SRCSTR REFBOT \$SRCSTR/256 REFBOT \$SRCSTR/256 REFBOT \$SRCSTR/256 NEAD TXTPTR TYPE STRCUT TE CR,LP, 'VA GETLIN \$\$00 NLAL XVAR1 DISVAR XVAR1 DISVAR **** TXOUT TXTBUFP,Y XVAR CASECX *** *** XVAR3 CASECX ***	* DOS CONTEXT * MOVE HEAD TO TRACK READ TRACK * LANGUAGE CONTEXT * GIVE TO INDEX RESET BOITION OF REPERENCE TABLE RESET TXTPTR TO BUFFER INIZ INITIAL TYPE RIABLES', CR, LP, LP, \$00 GET A LINE OF TEXT INIZ CHECK FOR E.O.P. DONE! PRINT TABLE LOOK AT CHARACTER E.O.L.?? ==> XVAR CORRECT ANY LOWER CASE CHECK FOR LETTER NO ==>
35500 36100 36200 36300 36500 36600 36600 36600 37000 37100 37300 37100 37300 37300 37400 37300 37400 37500 37700 375000 375000 375000 3750000000000	VAR1 XVAR XVAR1		SWAP SWAP SEEKCK READ SWAP SOF INDEX SOF INDEX SOF+1 TXTPTR+1 \$SRCSTR REFBOT \$SRCSTR/256 REFBOT+1 \$SRCSTR/256 REFBOT+1 \$SRCSTR/256 TXTPTR TYPE STRCUT STRCUT TYPE STRCUT TYPE STRCUT TYPE STRCUT TE CR,LP, 'VA GETLIN \$00 NLAE XVAR1 NLAE XVAR1 DISVAR TXBUFP, Y XVAR \$RENTK XVAR2 \$'2+1 GETVAR \$'2+1 GETVAR \$'2+1 GETVAR \$'2+1 GETVAR \$'2+1 GETVAR \$'2+1 CASECK \$'2	* DOS CONTEXT * MOVE HEAD TO TRACK READ TRACK * LANGUAGE CONTEXT * GIVE TO INDEX RESET EXTTOR TO TEXT GIVE TO INDEX RESET TXTPTR TO BUFFER INIZ INITIAL TYPE RIABLES', CR, LF, LF, \$00 GET A LINE OF TEXT INIZ CHECK FOR E, O, F. DONE! PRINT TABLE LOCK AT CHARACTER E, O, L, ?? ==> XVAR REMARK? YES! cm> XVAR CHECK FOR LETTER NO ==> YES! =>> FUNCTION ?
35500 36100 36200 36300 36500 36600 36600 36600 37000 37300 37300 37300 37300 37400 370000 370000 3700000000	VAR1 , XVAR XVAR1		SWAP SWAP SEEKCK READ SWAP SOF INDEX SOF INDEX SOF+1 TXITFIR+1 \$SRCSTR REFBOT \$SRCSTR/256 REFBOT+1 \$SRCSTR/256 REFBOT+1 \$SRCSTR/256 TYTFIR TYPE STRCUT TYPE CR,LP, 'VA GETLIN \$\$00 TXTFIR TYPE STRCUT TYPE STRCUT TYPE STRCUT TYPE STRCUT TYPE STRCUT TYPE STRCUT TYPE STRCUT TYPE STRCUT TYPE STRCUT TYPE STRCUT TYPE STRCUT TYPE STRCUT TYPE STRCUT TYPE STRCUT STRCU	* DOS CONTEXT * MOVE HEAD TO TRACK READ TRACK * LANGUAGE CONTEXT * GET START OF TEXT GIVE TO INDEX RESET BOTTOM OF REFERENCE TABLE RESET TATPTR TO BUFFER INIZ INITIAL TYPE RIABLES', CR, LF, LF, \$00 GET A LINE OF TEXT INIZ CHECK FOR E. O. F. DONE! FRINT TABLE LOCK AT CHARACTER E. O. L. ?? => XVAR REMARK? YES! SKIP TO NEXT LINE STRING LITERAL? YES! SKIP TO NEXT LINE
35500 36100 36200 36500 36500 36500 36500 36600 37100 37700 37700 37700 37700 37700 37700 37700 37700 37700 37700 37700 37700 37700 37700 37700 37700 37700 37700 37800 37800 37800 38800 38800 38800 38800 38800 38800 38800 38800 38800 38800 38900 39000 390000 39000 39000 390000 39000 39000 39000 39000 39000 390000 300	VAR1 XVAR	JERSKAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA	SWAP SWAP SEENCK READ SWAP SOF INDEX SOF INDEX SOF INDEX SOF INTFIR TECTOT INTFIR TYPE STRCUT STRCUT ST	* DOS CONTEXT * MOVE HEAD TO TRACK READ TRACK * LANSUAGE CONTEXT * GIVE TO INDEX RESET BOITOM OF REPERENCE TABLE RESET TATPTR TO BUFFER INIZ INITIAL TYPE RIABLES', CR, LF, LF, \$00 GET A LINE OF TEXT INIZ CHECK FOR E. O. F. DONEL FRINT TABLE LOOK AT CHARACTER E. O. L. ?? ==> XVAR REMARK? YESI SKIF TO NEXT LINE STRING LITERAL? YESI SKIF TO NEXT LINE STRING LITERAL? YESI SKIF TO NEXT LINE STRING TO NEXT LINE STRING TO REAL LINE STRING TO TEXT TO TEX
33590 3610 3620 3630 3650 3650 3650 3650 3650 3760 3770 3770 3770 3770 3770 3770 377	VAR1 XVAR XVAR1	JERERARA ARA ARA ARA ARA ARA ARA ARA ARA	SWAP SWAP SEENCX READ SWAP SOF INDEX SOF INDEX SOF+1 INDEX SOF+1 INDEX SOF+1 INDEX SOF+1 INTTFR ISCOT INTFR ISCOT INTFR STROUT TYPE STROUT TYPE STROUT TYPE STROUT TYPE STROUT TYPE STROUT TYPE STROUT TYPE STROUT TYPE INDEX STROUT INDEX STROUT TYPE INDEX STROUT INDEX STROUT INDEX STROUT INDEX STROUT STROUT STROUT INDEX STROUT INTFF INDEX STROUT STROUT ST	* DOS CONTEXT * MOVE HEAD TO TRACK READ TRACK * LANGUAGE CONTEXT * GIVE TO INDEX RESET BOITION OF REFERENCE TABLE RESET TXTPTR TO BUFFER INIZ INITIAL TYPE RIABLES', CR, LF, LF, \$00 GET A LINE OF TEXT INIZ CHECK FOR E.O.F. DONE! PRINT TABLE LOOK AT CHARACTER E.O.L.?? ==> XVAR GEMARK? YES! SKIP TO NEXT LINE STRING LITERAL? YES! SKIP TO NEXT LINE STRING LITERAL? YES! ==> YUNGTION ? NO ===> YES! SHOW FUNCTION SAVE IT BUMP TOBUFF INDEX
36100 36100 36200 36400 36500 36500 36500 36500 36500 36500 37700 37900 38000 38000 3900 3900 3900 3900 3900	VAR1 XVAR XVAR1	JERERAEAEAEAEAEAEAEAEAEAEAEAEAEAEAEAEAEA	SWAP SWAP SEEKC READ SWAP SOF INDEX SOF INDEX SOF+1 INDEX SOF+1 INDEX SOF+1 INDEX SOF INTEX INTE	* DOS CONTEXT * MOVE HEAD TO TRACK READ TRACK * LANGUAGE CONTEXT * GIVE TO INDEX RESET BOTTOM OF REPERENCE TABLE RESET TXTPTR TO BUPPER INIZ INITIAL TYPE RIABLES', CR, LP, LP, \$00 GET A LINE OF TEXT INIZ CHECK FOR E. O. F. DONE! PRINT TABLE LOCK AT CHARACTER E. O. L. ?? ==> XVAR REMARK? YES! SKIP TO NEXT LINE STRING LITERAL? YES! ==> FUNCTION ? NO ==> YES! I=> FUNCTION ? NO ==>
36100 36100 36200 36400 36500 36600 36600 36600 36600 37100 37700 37800 37900 39900 39000 39000 300000	VAR1 , XVAR XVAR1 XVAR2 XVAR3	JERERARARARARARARARARARARARARARARARARARA	SWAP SWAP SEEKC READ SWAP SOF INDEX SOF INDEX SOF+1 TXTPTR+1 \$SCSTR/256 REFBOT \$SRCSTR/256 REFBOT+1 \$SRCSTR/256 REFBOT+1 \$SOO TXTPTR TYPE STRCUT TYPE STRCUT TE CR,LP, 'VA GETLIN \$SOO NLAH XVAR1 NLAL XVAR1 DISVAR TXBUFP, Y XVAR \$REFNTK XVAR2 \$'2+1 GETVAR \$FPMIK XVAR2 \$'2+1 GETVAR \$FPMIK XVAR2 \$SIO TYPE XVAR1	* DOS CONTEXT * MOVE HEAD TO TRACK READ TRACK * LANGUAGE CONTEXT * GIVE TO INDEX RESET BOTTOM OF REFERENCE TABLE RESET TATPTR TO BUFFER INIZ INITIAL TYPE RIABLES', CR, LF, LF, \$00 GET A LINE OF TEXT INIZ CHECK FOR E.O.F. DONE! PRINT TABLE LOCK AT CHARACTER E.O.L.?? ==> XVAR REMARK? YES! SKIP TO NEXT LINE STRING LITERAL? YES! =>> FUNCTION ? NO ==> YES! => YES! => FUNCTION ? NO ==>
3590 3610 3620 3650 3650 3650 3650 3650 3650 3760 3770 3770 3770 3770 3770 3770 377	VAR1 XVAR XVAR1 XVAR2 XVAR3	je se	SWAP SWAP SEENCK READ SWAP SOF INDEX SOF INDEX SOF INDEX SOF INTFIR TITFIR INTFIR TYPE STRCUT TYPE STRCUT TE CR, LF, 'VA GETLIN 4500 NLAL XVARI DISVAR TXBUFP, Y XVAR2 4'1 STRCUT TXBUFP, Y XVAR2 4'2 I GETLAN 4'2 STRCUT TXBUFP, Y XVAR2 4'2 I GETVAR 4'2 SID CASBCK 4'2 SID SUBPP, Y XVAR1 TXBUFP, Y XVAR1 SID SUBPP, Y XVAR1 TXBUFP, Y XVAR1 TXBUFP, Y XVAR1	* DOS CONTEXT * MOVE HEAD TO TRACK READ TRACK * LANGUAGE CONTEXT * GET START OF TEXT GIVE TO INDEX RESET BOTTOM OF REFERENCE TABLE RESET TATPTR TO BUFFER INIZ INITIAL TYPE RIABLES', CR, LF, LF, \$00 GET A LINE OF TEXT INIZ CHECK FOR E. O. F. DONE! PRINT TABLE LOOK AT CHARACTER E. O. L.?? => XVAR REMARK? YES! SKIP TO MEXT LINE STRING LITERAL? YES! SKIP TO MEXT LINE STRING LITERAL?
36100 36100 36200 36300 36500 36500 36500 36500 36500 37100 37700 37700 37700 37700 37700 37700 37700 37700 37700 37700 37700 37700 37700 37700 37700 37700 37800 37700 37800 37800 38800 38800 38800 38800 38800 38800 38800 38800 38900 39000 39000 39000 39000 39000 39000 39000 39000 39000 39000 39000 39000 39000 39000 39000 39000 39000 39000 30000 300000 300000000	VARL XVARL XVARL XVAR2 XVAR3	JERERARARARARARARARARARARARARARARARARARA	SWAP SWAP SEENCK READ SWAP SOF INDEX SOF INDEX SOF+1 INTEX SOF+1 INTEX SOF+1 INTEX SOF+1 INTEX SOF SOF INTEX SOF SOF INTEX SOF SOF INTEX SOF SOF INTEX SOF SOF INTEX SOF SOF SOF SOF SOF SOF SOF SOF SOF SOF	* DOS CONTEXT * MOVE HEAD TO TRACK READ TRACK * LANGUAGE CONTEXT * GIVE TO INDEX RESET BOTTOM OF REFERENCE TABLE RESET TXTPTR TO BUFFER INIZ INITIAL TYPE RIABLES', CR, LF, LF, \$00 GET A LINE OF TEXT INIZ CHECK FOR E. O. F. DONE! PRINT TABLE LOCK AT CHARACTER E. O. L. ?? ==> XVAR STRING LITERAL? YES! SKIP TO NEXT LINE STRING LITERAL? YES! ==> YES! ==> YES! ==> YES! ==> YES! SHOW FUNCTION SAVE IT BUMP TOBUFF INDEX AND LOOP! BUMP TOBUFF INDEX FUND TRALING QUOTE?
33590 3610 3620 3630 3650 3650 3650 3650 3650 3760 3770 3770 3770 3770 3770 3770 377	VAR1 XVAR XVAR1 XVAR2 XVAR3		SWAP SWAP SEENCX READ SWAP SOF INDEX SOF INDEX SOF+1 UTTTR+1 #SRCSTR REFBOT #SRCSTR/256 REFBOT #SRCSTR/256 REFBOT #SRCSTR/256 STRUT TYPE STRUT TYPE STRUT TE CR,LP, 'VA GETLIN #\$00 NLAL XVAR1 DISVAR XVAR1 DISVAR XVAR1 DISVAR XVAR1 DISVAR ** XVAR2 #'A XVAR2 #'A XVAR2 #'A XVAR2 #'A XVAR2 #'A XVAR2 #'A XVAR2 #'A XVAR2 #'A XVAR3 CASECX #'A XVAR2 #'A XVAR3 CASECX #'A XVAR3 CASECX #'A XVAR3 CASECX #'A XVAR3 CASECX #'A XVAR3 CASECX #'A XVAR3 CASECX #'A XVAR3 CASECX #'A XVAR3 CASECX #'A XVAR3 CASECX #'A XVAR3 CASECX #'A XVAR3 CASECX #'A XVAR3 CASECX #'A XVAR3 CASECX #'A XVAR3 XVAR3 XVAR3 XVAR4 #'A XVAR3 XVAR3 XVAR4 #'A XVAR3 XVAR3 XVAR4 XVAR3 XVAR3 XVAR4 XVAR3 XVAR3 XVAR4 XVAR4 XVAR4 XVAR4 XVAR3 XVAR3 XVAR4	* DOS CONTEXT * MOVE HEAD TO TRACK READ TRACK * LANGUAGE CONTEXT * GIVE TO INDEX RESET BOTTOM OF REPERENCE TABLE RESET TXTPTR TO BUFFER INIZ INITIAL TYPE RIABLES', CR, LF, LF, \$00 GET A LINE OF TEXT INIZ CHECK FOR E.O.F. DONE! FRINT TABLE LOCK AT CHARACTER E.O.L.?? ==> XVAR CORRECT ANY LOWER CASE CHECK FOR LETTER NO ==> YES! ==> FUNCTION ? NO ==> YES! TOBUFF INDEX AND LOOP! BUNP TOBUFF INDEX AND LOOP! BUNP TOBUFF INDEX FIND TRALLING QUOTE? NO INOP INF NOR
33590 3610 3620 3630 3650 3650 3650 3650 3650 3660 3700 3710 3770 3770 3770 3770 3770 377	VAR1 XVAR XVAR1 XVAR2 XVAR3	JERERAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA	SWAP SWAP SEEKCK READ SWAP SOF INDEX SOF INDEX SOF+1 TXTPTR+1 \$SRCSTR REFBOT \$SRCSTR/256 REFBOT+1 \$SRCSTR/256 TYTPE \$SRCSTR/256 TYTPE STRCUT STRCUT STRCUT TYPE STRCUT STRCUT TYPE STRCUT TYPE STRCUT TYPE STRCUT TOBUFF, Y XVAR \$1 A XVAR1 TOBUFF, Y XVAR2 \$1 A XVAR1 TOBUFF, Y XVAR \$1 CASBCK \$1 A XVAR1 TOBUFF, Y XVAR \$1 CASBCK \$1 A XVAR1 TOBUFF, Y XVAR \$1 CASBCK \$1 CASCK \$1 CASBCK \$1 CASCK \$1 CASCK \$1 CASCK \$1 CASCK \$1 CASCK \$1	* DOS CONTEXT * MOVE HEAD TO TRACK READ TRACK * LANGUAGE CONTEXT * GIVE TO INDEX RESET BOTTOM OF REFERENCE TABLE REFERENCE TABLE REFERENCE TABLE REFERENCE TABLE REFERENCE TABLE REFERENCE TABLE REFERENCE TABLE INIZ INITIAL TYPE RIABLES', CR, LF, LF, \$00 GET A LINE OF TEXT INIZ CHECK FOR E.O.F. DONE! PRINT TABLE LOCK AT CHARACTER E.O.L.?? ==> XVAR REMARK? YES! SICH FON LETTER NO ==> YES! ==> FUNCTION ? NO ==> YES! => YES! SICH FUNCTION SAVE IT BUNG TOBUFF INDEX AND LOOF! BUNG TOBUFF INDEX AND LOOF! BUNG TOBUFF INDEX AND LOOF! SING TABLES CONTERCE FILL NEXT CHARACTER E.O.L.? ==> XVAR FILL NEXT CHARACTER E.O.L.? ==> XVAR
3610 3610 3620 3630 3650 3650 3650 3650 3650 3760 3770 3770 3770 3770 3770 3770 377	VAR1 XVAR XVAR1 XVAR2 XVAR2		SWAP SWAP SEENCK READ SWAP SOF INDEX SOF INDEX SOF INDEX SOF INTFIR TITFIR INTFIR TYPE STRCUT SUBJEF, Y XVAR \$''' STRCUT TYPE STRCUT TYPE STRCUT SUBJEF, Y XVAR \$''' SUBJEF, Y XVAR \$''' SUBJEF, Y XVAR STRCUT SUBJEF, Y XVAR STRCUT SUBJEF, Y XVAR STRCUT SUBJEF, Y XVAR STRCUT SUBJEF, Y XVAR SUBJEF, Y X XVAR SUBJEF, Y X XVAR SUBJEF, Y X XVAR SUBJEF, Y X XVAR SUBJEF, Y	* DOS CONTEXT * MOVE HEAD TO TRACK READ TRACK * LANSUAGE CONTEXT * GIVE TO INDEX RESET BOITOM OF REFERENCE TABLE RESET TATPTR TO BUFFER INIZ INITIAL TYPE RIABLES', CR, LF, LF, \$00 GET A LINE OF TEXT INIZ CHECK FOR E. O. F. DONE: FRINT TABLE LOOK AT CHARACTER E. O. L. ?? =>> XVAR GET A LINE OF TEXT INIZ CHECK FOR E. O. F. DONE: FRINT TABLE LOOK AT CHARACTER E. O. L. ?? =>> XVAR GET ANY LOWER CASE CHECK FOR LETTER NO ==>> YESI ==> FUNCTION ? NO ==> YESI SHOW FUNCTION SAVE IT BUMP TOBUFF INDEX AND LOOP! BUMP TOBUFF INDEX PEND TRALLING QUOTE? NO! INO FIND ANY CONSTRUCT PEND TRALLING QUOTE? NO! INO FIND ANY CONSTRUCTION SAVE IT BUMP TOBUFF INDEX AND LOOP! ==> XVAR
3590 3610 3620 3650 3650 3650 3650 3650 3650 3760 3770 3770 3770 3770 3770 3770 377	VARI XVAR XVARI XVARI XVAR3		SWAP SWAP SEENCK READ SWAP SOF INDEX SOF INDEX SOF INDEX SOF INTFE SSCF11 INTFTR+1 #SNCSTR/256 REFBOT #SNCSTR/256 REFBOT #SNCSTR/256 REFBOT #SNCSTR/256 REFBOT #SNCSTR/256 REFBOT #SNCSTR/256 REFBOT #SNCSTR/256 REFBOT #SNCSTR/256 REFBOT #SNCSTR/256 REFBOT #SNCSTR/256 REFBOT #SNCSTR/256 REFBOT #SNCSTR/256 REFBOT NLAL XVAR1 DISVAR #'E XVAR2 #'E SNCSTR/256 REFBOT #'E XVAR2 #'E XVAR2 #'E XVAR1 TDEUFF, Y XVAR2 #SNCSTR/256 REFBOT SNCSTR/256 REFBOT	* DOS CONTEXT * MOVE HEAD TO TRACK READ TRACK * LANGUAGE CONTEXT * GIVE TO INDEX RESET BOTTOM OF REFERENCE TABLE RESET TXTPTR TO BUFFER INIZ INITIAL TYPE RIABLES', CR, LF, LF, \$00 GET A LINE OF TEXT INIZ CHECK FOR E. O.F. DONE! PRINT TABLE LOCK AT CHARACTER E.O.L.?? =>> XVAR STRING LITERAL? YES! SKIP TO NEXT LINE STRING LITERAL? YES! SKIP TO NEXT LINE STRING LITERAL? YES! =>> YES! =>> YES! =>> YES! =>> YES! =>> YES! =>> FUNCTION ? NO ==> YES! HOM FUNCTION SAVE IT BUMP TOBUFF INDEX AND LOOP! END TABLES PICH NEXT CHARACTER E.O.L.? =>> XVAR YES! HOM F LOTE? NO ==> YES! SKIP TO NEXT LINE STRING 2000000000000000000000000000000000000
33590 3610 3620 3660 3660 3660 3660 3660 3660 3710 3710 3770 3770 3770 3770 3770 377	VAR1 XVAR XVAR1 XVAR2 XVAR3		SWAP SWAP SEINCX READ SWAP SOF INDEX SOF SOF INDEX SOF SOF INDEX SOF INDEX SOF SOF INDEX SOF INDEX SOF SOF	* DOS CONTEXT * MOVE HEAD TO TRACK READ TRACK * LANGUAGE CONTEXT * GIVE TO INDEX RESET BOTTOM OF REPERENCE TABLE RESET TATPTR TO BUFFER INIZ INITIAL TYPE RIABLES', CR, LP, LP, \$00 GET A LINE OF TEXT INIZ CHECK FOR E.O.P. DONE! FRINT TABLE LOCK AT CHARACTER E.O.L.?? ==> XVAR OURECT FOR E.O.P. DONE! FRINT TABLE LOCK AT CHARACTER E.O.L.?? ==> XVAR CHECK FOR LETTER NO ==> YES! ==> FUNCTION ? NO ==> YES! HOW FUNCTION SAVE IT BUMP TOBUFF INDEX AND LOOP! EUNP TOBUFF INDEX AND LOOP
3590 3610 3620 3630 3650 3650 3650 3650 3660 3700 3710 3770 3770 3770 3770 3770 377	VARI XVAR XVAR XVAR1 XVAR2 XVAR3 GETVAR		SWAP SWAP SEEKC READ SWAP SOF INDEX SOF SOF INDEX SOF SOF INDEX SOF SOF SOF SOF SOF SOF SOF SOF SOF SOF	* DOS CONTEXT * MOVE HEAD TO TRACK READ TRACK * LANGUAGE CONTEXT * GIVE TO INDEX RESET EXTITY OF TEXT GIVE TO INDEX RESET EXTITY TO BUFFER INIZ INITIAL TYPE RIABLES', CR, LF, LF, \$00 GET A LINE OF TEXT INIZ CHECK FOR E. O. F. DONE! FRINT TABLE LOCK AT CHARACTER E. O. L. ?? ==> XVAR CHECK FOR E. O. F. DONE! FRINT TABLE LOCK AT CHARACTER E. O. L. ?? ==> XVAR CORRECT ANY LOWER CASE CHECK FOR LETTER NO ==> YES! =>> FUNCTION SAVE IT BUMP TOBUFF INDEX AND LOOF! BUMP TOBUFF INDEX AND LOOF! BUMP TOBUFF INDEX AND LOOF! BUMP TOBUFF INDEX AND LOOF! SAVE AS 1ST CHARACTER GET A SPACE CLEAR 2ND CHARACTER E. O. L. ? ==> XVAR FIND TRALLING QUTE? NO LOOP! SAVE AS 1ST CHARACTER GET A SPACE CLEAR 2ND CHARACTER BUMP INDEX

				•
4120		LDA	TXBUFF, Y	GET NEXT CHARACTER
4140		QIP	4's	INTERGER ?
4150		BEQ	SETINT #'S	YES! ==>
4170		BEQ	SETSTR	YES! ==>
41.80		BEQ	SETARR	ARRAY ? YESi ===>
4200		CMP	#\$80 CFT5222	TOKEN ?
4220		CMP	#'0	CHECK FOR NUMERIC
4230		BCC	GETVA2	NO ===>
4250		BOC	GETVA0	YES! ==>
4260		JSR	CASECK	CORRECT FOR LOWER
4280		BCC	GETVA2	
4290		BCS	GETVA2	END: ==>
4310	GETVA0	LDX	VARNAM+1 #SP	CHECK 2ND CHAR. (LEAR?
4330		BNE	GETVAL	NOI LOOPI
4340		JMP	GETVAL	AND LOOP!
4360	SETINT	LDA	\$\$01 TE SELEZ	SHOW INTERGER
4380	SETSTR	LDA	# \$02	SHOW STRING
4390		ORA STA	TYPE TYPE	COMBINE WITH CURRE SET TYPE
4410		BNE	GETVAL	AND LOOPI
4420; 4430	SETARR	LDA	TYPE	GET CURRENT TYPE
4440		ORA	\$\$80 Trans	ADD ARRAY TYPING
4460	GETVA2	STY	TMPPTR	SAVE TOBUFF INDEX
4470 4480		LDA STA	#SRCSTR PUTTPTR	INIZ LOOK-UP POINT RESET TO TOP OF TA
4490		LDA	\$SRCETR/256	
4500		STA	PUTPTR+1	
4520	GETVA3	LDA	FUTPIR	CHECK FOR END 1ST
4540		BNE	GETVA	NO ==>
4550		LDA	PUTPTR+1	
4570		BNE	GETVA	NO ==>
4580 4590	GETVM	JMP LDY	NEWVAR \$SOO	YES! NEW! ===> INI2
4600		STY	TFLAG	CLEAR MATCH FLAG
4610		CMP	(PUTPIR),I	SAME AS CURRENT?
4630		BNE	GEIVA5	ND: ==> YES: CHECK 2ND
4650		LDA	(PUTPTR),Y	FETCH 2ND
4660		BNE	GETVA5	NO ==>
4680		INY	(0070700).V	YESI BUMP INDEX
4700		CMP	TYPE	
4710		BNE	GETVA5 TFLAG	NO! ==> YES! SET MATCH FLA
4730	GETVA5	LDY	#\$03	INIZ
4750		STA	NR	SAVE IT
4760		ASL	A	* 2 SAVE 10
4780		LDA	¥\$00	CLEAR
4790 4800		ROL STA	A TI+1	ROTATE ANY CARRY I SAVE AS MSB
4810		LDA	PUTPTR	FETCH FUTPTR
4820		ac	CLOPIR	SAVE FOR LATER
4840		ADC	#\$04 PERTENDE	ADD HEADER OFFSET
4860		LDA	PUTPTR+1	PETCH MSB
4870		ADC	4\$00	ADD ANY CARRY
4890		STA	PUTPER+1	SAVE MSB
4900		ac	FUIFIK	REFEICH NEW LOD
4920 4930		ADC STA	TI FUTPIR	ADD NR * 2 SAVE IT
4940		LDA	FUTPIR+1	FETCH MSB
4950		STA	PUTPTR+1	SAVE IT
4970		LDA	TFLAG CETVA3	FETCH MATCH FLAG
4990		LDA	NR	CHECK NR AGAIN
5000 5010		BEQ STA	GETVA8 T1	01 INSERT! SAVE IT
5020	•	LDA	OLDPTR	GET OLDPTR
5030		ADC	#\$04	ADD HEADER OFFSET
5050		STA	FETPTR OLDETEL	SAVE RESULT
5070		ADC	\$\$00	
5080 5090	GETVAG	STA	FETPTR+1 #SOO	INIZ
5100		LDA	(FETPIR),Y	FETCH REFERNCE
5120		BNE	GEIVA7	NO! ==>
5130		INY LDA	(PRTPTP).V	YESI BUMP INDEX CHECK MSB
5150		CHIP CHIP	LNHI	LINE CONTRACT
5160		BNE JMP	GEIVA/ GEIVA9	MATCHI SKIP ENTRY
5170				

NEXT CHARACTER L.? ==> GETVA2 INGER ? ERGER ? 1 mm> 1 mm> 1 mm> NY ? 1 mm> EN ? 2N ? 2N ? 2N ? CK AGAIN 4 mm> (==> RECT FOR LOWER CASE 28 FOR END OF NAME I ==> SK 2ND CHAR. AR? LOOPI I SAVE 2ND CHAR. LOOPI I DVTERGER V STRING BINE WITH CURRENT TYPE LCOP1 CURRENT TYPE ARRAY TYPING E AND FALL THROUGH E TOBUFF INDEX Z LOOK-UP FOINTER ET TO TOP OF TABLE TK FOR END 1ST => ==> NEWl ===> AR MATCH FLAG TH 15T CHARACTER E AS CURRENT? ==> 1 CHECK 2ND TH 2ND I CHECK 2ND CH 2ND PARE IT ==> I BUMP INDEX O CHECK TYPE ==> SET MATCH FLAGI HIOPREFS. III IT 5 II AR ATE ANY CARRY IN E AS MSB CCH FUTPTR 7E POR LATER HEADER OFFSET E IT CH MSB E IT FOR LATER TOO ANY CARRY E MSB ETCH NEW LSB P NR * 2 E IT CH MSB D MSB OF NR * 2 E IT CH MATCH FLAG JAR TRY NETTI =>> CK NR AGAIN INSERT! E IT



.

ł

CHI SKIP ENTRY! Listing continued

.

5130 CC TAG JAP CETVINA NOT LODY! 6230 JSR CHLP DO SERVANCEME 5200 MC (\$122 ADD LINE HER, CFFSET 7750 ISR CHLP 5770 5700 5700 5710	51.80	GETVA7	LDA	PETPTR	GET POINTER	5730	1	STA 1	REFBOT+1		6280		STA	NR	SAVE IT
520052005200575057506300JER CELP5210GET RUTTER575057505760576057605760576057605760576057605760576057605760576057605770 <td>51 90</td> <td></td> <td>ac</td> <td></td> <td></td> <td>5740</td> <td></td> <td>JMP (</td> <td>GETVA9</td> <td>AND LOOP!</td> <td>6290</td> <td></td> <td>JSR</td> <td>CRLF</td> <td>DO SEPARATION</td>	51 90		ac			5740		JMP (GETVA9	AND LOOP!	6290		JSR	CRLF	DO SEPARATION
S210 DD. NUTCE FOR LUNCAL S760 KIND DUTTE \$00,'%\$' S110 LAN FUTTER S770 S220 LAN FETTERN S770 S780 DISVAR LAN ASSCETE INIZ S130 ACC \$600 ACC \$600 ACC \$600 ACC \$600 SDR FUTTER SDR FUTE SDR FUTTER SDR FUTE	5200		ADC	4502	AND LINE PER OFFERT	5750:					6300		JSR	CRLF	
2520DATA FERTURA 1.3770, TODATA CONTRACT6320CLCADD BEADER OF2520BAC 1600STA RUTPTRTO TO FO C PALLE6330STA RUTPTRMACE 1604MACE 11 MOINE2520DEC 11DECREMENT CHECK COUNTERSTA RUTPTRTO TO FO C PALLE6330STA RUTPTRMACE 1604MACE 11 MOINE2520DEC 11DECREMENT CHECK COUNTERSTA RUTPTRCOUNTAR SOUTH AND	5200		000		ADD DING NET. OFFDD:	5760 KT	NTO	BVT	E \$00. '#\$'		6310		LDA	PUTPIR	GET PUTPTR
2530 2530 <th< td=""><td>5210</td><td></td><td>DIA T DA</td><td>PETPIR POMODA1</td><td></td><td>5770.</td><td></td><td></td><td>2 4001 94</td><td></td><td>6320</td><td></td><td>ac</td><td></td><td></td></th<>	5210		DIA T DA	PETPIR POMODA1		5770.			2 4001 94		6320		ac		
22.00 Dot 1900 SAVE IT 570 COUNT OF A RUTPER 00 TO COUNTER 6340 STA RUTPER SAVE IT 2520 DEC TIL DECOMENT COUNTER 5800 LAN ENCENT ESCONTAGE ALAOS HANDLE MESS 6330 DECOMENT COUNTER 5800 LAN ENCENT ESCONTAGE ALAOS HANDLE MESS 6330 DECOMENT COUNTER 5800 LAN ENCENT ESCONTAGE ALAOS HANDLE MESS 6330 LAN (RUTPER), T GET MESS, LANS MAINTERNA GET UNREANT FOUNDAME 6330 DECOMENT COUNTER S630 LAN (RUTPER), T GET MESS, LANS GET	5220		LUN	PEIPIRTI ACOO		5700 010	CUND 1	T FNA	1CDCCTD	TNT 7	6330		ADC	¥\$04	ADD HEADER OFFSET
2440 374 PERTURNAL DOLDBERT COLOR MALE 6330 EXCL DISVAG MARCE PAGE 2520 DEC TITMA DEDUBBENT CEDIT MARCE 5330 DEC DISVAG MARCE 5330 DEC DISVAG MARCE 5330 DEC DISVAG MARCE 5330 DEC DISVAG DESVE 5330 DEC DISVAG MARCE 5330 DEC DISVAG MARCE 5330 DEC DISVAG MARCE 5330 DEC DISVAG MARCE 5330 DESVE S330 DESVE MARCE 5330 DESVE MARCE DESVE S330 DESVE MARCE DESVE	5230		ALC	#\$UU		2/00 010	OVAR I	un .	ACCONC.		6340		STA	PUTPTR	SAVE IT
2230 DBC TI DBC NUMBER TORK CUMPER 5800 DATA DATA ALL RES 5360 DEC TITERAT FORMATION CONTROL RES Contro Res <	5240		STA	FEIPIR+1		5/90		01A .	FUIFIR	NOO UNTER NED	6350		BCC	DISVA6	WATCH PAGING
2520BRE GEVINGLOG "TIL LONGE5810SHA UPTPRGET CURRANT FOINTER6370DEX (100^{-1} KeV)INIZ2520LIX 1600S600CHE ESSUECOMPARENTGET CURRANT FOINTER6330GTA RESLO2520LIX 1600S600CHE ESSUECOMPARENTGET CURRANT FOINTER6330GTA RESLO2520LIX 1600S7AS650LIX 1007FEN, YGET REFL LINE6400IXA (FUTFEN, Y3530DIX 1007FEN, YS650CHE REFL VALS670REFL VAL6410LIA (FUTFEN, Y3530JEN (VTTFEN), YS680JEN CONFRANCES640SFA RESLSFK IT3530STA (RETAR), YS680LIA REFL VALS640JEN (NTTFEN), Y3530GET (SIGUE), YPTCH CLD RS590STA OULAGMARE IT CURRANT6430JEN (NTTFEN), Y3530LIA (CLDTEN), YPTCH CLD RS590STA CURLAGMARE IT CURRANT6450JEN (NTTFEN), TRUET3530GET (ALL), YSWIE ITS590STA CURLAGMARE IT CURRANT6450JEN (NTTFEN), TRUET, S590STA CURLAG6400JEN (TRUET, TRUET, S500STA CURLAGS500STA CURLAG3540LIA (CLDTEN, Y, STA TER,	5250		DEC	11	DECREMENT CHECK COUNTER	5800		LUA	SHLSIN 200	ALSU HANDLE POD	6360		TNC	PUTTER+1	
2270 GEVMA JSR MOVDIP CREATE SPACE IN TABLE 5420 LIX ALLAN MUTTRY COMMARY FOLMATE 5100 LIX HATTANDARY COMMARY FOLMATE S100 LIX HATTANDARY COMMARY FOLMATE S100 LIX HATTANDARY COMMARY FOLMATE S100 LIX HATTANDARY COMMARY FOLMATE COMMARY FOLMATE COMMARY FOLMATE S100 LIX HATTANDARY COMMARY FOLMATE COMMARY FOLMATE COMMARY FOLMATE COMMARY FOLMATE COMMARY FOLMARY COMMARY FOLMARY COMMAR	5260		BNE	GEIVAD	LOOP TIL DONE	5810		SIA .	PUTPIR+1		6370	DISVAG	LIN	#\$00	INTZ
5280LDX #5005810CMP REPORTCOMPARE TO EAC510COMPARE TO EAC510LDR100 mm5300STA (MUTTER), Y5560LDR MEDITALMOTES, LDRE MEDITAL <td< td=""><td>5270</td><td>GEIVAB</td><td>JSR</td><td>MOVEUP</td><td>CREATE SPACE IN TABLE</td><td>5820 DIS</td><td>SVAL</td><td>LDA</td><td>PUTPIR</td><td>GET CURRENT POINTER</td><td>6290</td><td>DIDINO</td><td>TTA</td><td>(0000000) V</td><td>GET REF LINE & LSB</td></td<>	5270	GEIVAB	JSR	MOVEUP	CREATE SPACE IN TABLE	5820 DIS	SVAL	LDA	PUTPIR	GET CURRENT POINTER	6290	DIDINO	TTA	(0000000) V	GET REF LINE & LSB
5290LDA LNCOSk40ENER LINVAL MATEL, CHECK MSB61.00CTAY MATEL, CHECK MSB61.00CTAY MATEL, CHECK MSB61.00CTAY MATEL, CHECK MSB61.00CTAY MATEL, CHECK MSB61.00CTAY MATEL, CHECK MSB61.00CTAY 	5280		LDY	\$ \$00		5830	1	OMP .	REFBOT	COMPARE TO END	6300		CTTA	DEST ()	
5300 STM (FUTTER), Y 5500 LDM (NUTTER), Y 5500 LDM (NUTTER), Y GET RESULT GET RESULT GET RESULT SEC OLD RESULT	5290		LDA	LNLO		5840		BNE	DISVAZ		6400		710		NUMP THINKY
5310INV580CMP MERSON16410Link (FUTAN), 7SAVE IT5320LOA (MUTTR), Y5860JSK CGLPYISI (D CLPAN-UP6420JSK RESELTRAVE IT5330LOA (MUTTR), Y5860JSK CGLPYISI (D CLPAN-UP6430JSK RAVE ITRAVE IT5330LOA (MUTTR), Y5860JSK CGLPYISI (D CLPAN-UP6430JSK RAVE ITRAVE IT5340LOX (MUTTR), YFPTCH CLD NR5590LOX (D CLPAN-UP6450JSK CUTCHRAVE IT5350CLC (LOUTR), YSWE ITSS10DISVA JUL JX (NONUL R6450JSK CUTCHRAVE IT5350CLC (LOUTR), YSWE ITSS10JSK COUNTCLAR COUNT6480ACC 4502ADD LINE RAVE5350GETA A LOUT TREPTREST CLARACTERSS10JSK COUNTCLARACTER6480ACC 4502ADD LINE RAVE5400LDA (RUTTR), YFETCH JST CLARACTER6480ACC 4502ADD LINE RAVESS10INT (CLPAR N), YFETCH JST CLARACTER6480ACC 4502ADD LINE RAVE5401LDA (RUTTR), YFETCH JST CLARACTER6480ACC 4502ADD LINE RAVESS10INT (CLPAR N), YFETCH JST CLARACTER6480ACC 4502ADD LINE RAVE5401LDA (RUTTR), YFETCH JST CLARACTER6480ACC 4502ADD LINE RAVESS10INT (CLPAR N), YFETCH JST CLARACTER6480ACC 4502ADD CLARACTERFETCH IST CLARACTER6480ACC 4502ADD CLARACTERFETCH IST CLARACTER </td <td>5300</td> <td></td> <td>STA</td> <td>(PUTPIR),Y</td> <td></td> <td>5850</td> <td>1</td> <td>LDA</td> <td>PUTPIR+1</td> <td>MAYBE, CHECK MSB</td> <td>6400</td> <td></td> <td>TINI</td> <td></td> <td>CET PER LINE & MSR.</td>	5300		STA	(PUTPIR),Y		5850	1	LDA	PUTPIR+1	MAYBE, CHECK MSB	6400		TINI		CET PER LINE & MSR.
5320LDA LNHI5870HNE DLEWALNUmpt6420SIN KERL IT5330STA (RUTPR),Y5880LDA DEPALT-IGET A SPACEYES I DO CLEAN-UP6430JSK NARUTRAWE IT5340LDA (GLUPR),YPTCH GLD NR5900DIX GUELANACE IT CORRENT6450JSK QUICHRAWITRINT5350LDA (GLUPR),YPTCH GLD NR5900STA GUELANACE IT CORRENT6450JSK QUICHRINT5350CLCADD 1S220DIX NC LDANACE IT CORRENT6450JSK QUICHRINT5380STA (GLUPTN),YRETKIEVE TSUFF INDEX5940DIX (RUTTR),YFORMAT6470LDA (RUTTR)5380STA (GLUPTN),YRETKIEVE TSUFF INDEX5940DIX (RUTTR),YFEDCI ADA (SOT6480LDA (RUTTR)5380STA (GLUPTN),YRETKIEVE TSUFF INDEX5940DIX (RUTTR),YFEDCI ADA (SOT6510CC DISNAT5400LDA (RUTTR),YRETKIEVE TSUFF5950DIX (RUTTR),YFEDCI ADA (SOT6510CC DISNAT5401LDA K000INIZSS80STA VARNAMBUP INDEX6510CC DISNATNACE IT5401LDA K000INIZSS80STA VARNAMBUP INDEX6510CC DISNATNACE IT5401LDA K000INIZSS80STA VARNAMBUP INDEX5510CLANATORNASS00CLANATORNA5401LDA K000INIZCLANATORNASS00IDA (RUTTR),YFETCI TYPE5530DIXATCC D	5310		INY			5860		QIP .	KEPBOI+I		6410		1 LUN	(PUIFIN) / I	
5330STA (FUPTR), Y5880JEK CRLSTA (FUPTR), Y6430JEK NUMUTHILR I T5330LDA (CLUPTR), YFETC (ALD NR5900STA CUPLAGNAD QUIT6440LDA NEW TTSTA (CLUPTR), YFETC (ALD NR5900STA CUPLAGNAD QUIT6440LDA NEW TTSTA (CLUPTR), YSTA (ALD ISTA (ALD I)STA (ALD I)	5320		LDA	LNHI		5870	1	BNE .	DISVAZ		6420		STA	RESHI	SAVE IT
5340LDM \$6335890LDM DEPAUL-1GET CURRALE JUW6440LDA \$59GET A SPACE5350LDA (CLEPTR), YFFTCH CLD NR5900STA CULFAGMAKE TO CURRANT6460JSR CUTCHTMICE5360CLCADD 15920DISVA2 LDY \$600TNIZ6470LDA \$700LDA \$700CLCTMICE5380STA (CLEPTR), YSNEW IT5930STY COUNTCLEAR COUNT6480CLCADD 1CLC5390STA (CLEPTR), YRETRELEVE TOBUPT INDEX5940LDA \$700TMICT6500STA NUTPTRADD 1.SEC CURRANT F5300STA VARNAHSAVE IT6500STA NUTPTRFETCH INDEX6510STA NUTPTRADD 1.SEC CURRANT F5400LDA \$600INIZSFT A SANAHHSAVE IT6500STA NUTPTRADD 1.SEC CURRANT F5400INIYNOI SUMP INDEX5970LDA (FUTPTR), YFETCH ITPE6520INC FUTPTR-15400STA STA TUPECLEAR VARLALE TYPE5980INYBUMP INDEX6540LDA CUNNT5400FINA STA TUPECLEAR VARLALE TYPE5980INYBUMP INDEX6540LDA CUNNT5400FINA WA WALABLE TYPES980INYBUMP INDEX6540LDA CUNNTFETCH IT5400FINA WALABLE TYPES980INYBUMP INDEX6550CLP ANAACX CUNNT5400FINA WALABLETYPEGET VARLANHSAVE IT6550CLP ANAACX CUNNT <t< td=""><td>5330</td><td></td><td>STA</td><td>(PUTPIR),Y</td><td></td><td>5880</td><td></td><td>JSR</td><td>CRLF</td><td>YEST DU CLEAN-OP</td><td>6430</td><td></td><td>JSR</td><td>NUMOUT</td><td>PRINT IT</td></t<>	5330		STA	(PUTPIR),Y		5880		JSR	CRLF	YEST DU CLEAN-OP	6430		JSR	NUMOUT	PRINT IT
5350LDA (CLOPTR), YPETCH CLO NR5900STA CUFLAGMARE IT CURRENT6450JSR OUTCHFRINT IT5360CLC5910HEIAND QUTT6450JSR OUTCHFRINT IT5300STA CULPTR), YSAVE IT6470LDA KUTPIRGET CURRENT F5370STA (CLIPTR), YSAVE IT5930STY COUNTCLEAR COUNT6480AUC 4502ADD LINE EXTR5390STA (CLIPTR), YREFECTI LAST CHARACTER5950STA VARNAHSAVE IT6490AUC 4502ADD LINE EXTR5400LDA (TRUTPR), YREFECTI LAST CHARACTER5950STA VARNAHSAVE IT6510BCC DISVATWATCH FROINS5400INAHONE INDEX5960INNBUMP INDEX6510BCC DISVATWATCH FROINS5400INAROI BUMP INDEX5960STA VARNAHSAVE IT6530DISVAT INC COUNTSHOW HE FROINS5400INAROI BUMP INDEX5960STA VARNAHSAVE IT6530DISVAT INC COUNTSHOW HE FROINS5400INANDI BUMP INDEX6000LDA (FUTPER), YFETCI TYPE6550DISVAT INC COUNTSHOW HE FROINS5400JAP XVARND GET NEXT LINE6000LDA (FUTPER), YFETCI TYPE6550BCD CISVABO.A (CESV-LEP)5400JAP XVARND GET NEXT LINE6000LDA (FUTPER), YFETCI TYPE6550BCD CISVABO.A (CESV-LEP)5400JAP XVARND GET NEXT LINE6000LDA (FUTPER), YFETCI TYPE	5340		LDY	#\$03		5890		LDA	DEPAUL+1	GET CONSULE DV	6440		LDA	#SP	GET A SPACE
5360CLCS910PCSAND QUIT6460JER CUTCHTMICE5370ADD 15520DIVAL LV 4500INIZ6470LAR PUTTRGET CURRENT F5380STA (CLUPTR), YSAVE IT5330STY COUNTCLAR COUNT6480CLCADD LINE RUEN5390GETVA DIY THPETRREINERVE TOBURPT INDEX5940LAN (FUTTER), YFETCI LST CHARACTER6540CLC 4690ADC 4502ADD LINE RUEN5400LAN (FUTTER), YRETELIVE TOBURPT INDEX5950STA VARNAMSAVE IT6510BCT UNRENT F5410BEQ GETVAA E.O.L.?S960STA VARNAM+ISAVE IT6530STA VATTTR5420INYNOI BUMP INDEX5970LDA (FUTTER), Y FETCI 2ND CHARACTER6550INC (FUTTER), F5430LAN \$000INIZ5990STA VARNAM+ISAVE IT6550CLC PUTTER), F5440STA 'NTFECLEAR VARIABLE TYPE6010ADD \$100MAR KTO 'N'I' TYPE6550CLC DISVA75450JAP XVARNAD GET VARIABLE TYPE6010AND \$100MAR KTO 'N'I' TYPE6550STA CUNN'CLEAR VARIABLE TYPE5400DAV XARAMRAD GET VARIABLE TYPE6010JAR STROTT6580STA CUNN'CLEAR CUNN'5400DAV XARAMGET VARIABLE TYPE6010JAR STOUT100 A (CCR)-CP6510DA (CCR)-CP5400DAV XARAMGET VARIABLE TYPE6010JAR STOUT100 A (CCR)-CP6510DA (CCR)-CP5400DAV XAR	5350		LDA	(OLDPTR), Y	PETCH OLD NR	5900		STA	OUFLAG	MAKE IT CURRENT	6450		JSR	OUTCH	PRINT IT
5370ADC 4501ADC 15520 DISW2 LDY 4500INI26670LDA PUTTERGET CURRENT P5380STX (CULT)SAVE IT5390STY COUNTCLARA COUNT6480CLCADD LINE RATE5390GETVAS LDY TMPPTRRETENTLYE TOURNET INSC5940LDA (FUTTER), Y FETCH IST CHARACTER6480ADC 4502ADD LINE RATE5400LDA TROUPT, REFETCH LAST CHARACTER5950STA VARNAMSAVE IT6500STA VARNA6510BCC DISVA7WATCH PACING5400IAN TYPENOI BURD TNEX5970LDA (FUTTER), Y FETCH IST CHARACTER6530DISVA TA COUNTFETCH IT5400TAN TYPECLEAR VARIALE TYPE5980STA VARNAMSAVE IT6530DISVA TA COUNTFETCH IT5400TAN TYPECLEAR VARIALE TYPE5990IANBURP INDEX6530DISVA TA COUNTFETCH IT5400TAN TYPECLEAR VARIALE TYPE6000LDA (FUTTER), Y FETCH TYPE6550CPU P \$0ADOR: 1075400STA TYPECLEAR VARIALE TYPE6000DISV2 IDMARK TO 'PN' TYPE6550DISVA DA (CLEAR COUNT5400DIRVARND GET NEXT LINE6000JSR STECUTYESI STOUT TI6550JSR COUPCLEAR COUNT5500LDA VARAMGET 2ARACTER6060JSR STECUTYESI STAN TTE6510JSR COUPCLEAR COUNT5500LDA VARAMGET 2ARACTER6060JSR CULPFETCH IDEX6610JSR COUPTI DA5500LDA VARAM<	5360		αc			5910		RTS		AND QUIT	6460		JSR	OUTCH	TWICE
5380STM (CLEPTE,); YSAVE IT5930STY CUUNTCLEAR COUNT6480CLC5390GETVA DUT THEPTRREFERCH TOBER5940LDA (FUTTER); YFETCH LST CHARACTER6500STM AUTOTR5400LDA TOBUEF, YREFERCH LAST CHARACTER5950STM VARNAMSAVE IT6510DC LIS/ANNOC 14025400DEN GETVAAE.O.L.?S500STM VARNAMSAVE IT6510DC LIS/ANNOC 1407NOC 14075400DA 1500INTE.O.L.?S500STM VARNAMSAVE IT6520IDK VARNAMNOC 0107NOC 1107NOC 11075400DA 1500INTE.O.L.?S500STM VARNAMNOE INC.NOE NOT NOT SUM PERDICS500IDK VARNAMNOC 0107PETCH ITF5400DA VARNAAND CONTINUE6000LDA (PUTTER), YPETCH TYPE6550DOR 107DOR 1075400DA VARNAND CONTINUE6010AND \$\$10NO =>6570LDA \$\$00D.N.O.K. (CMT.5401DA VARNAND CER NERT LINE6020BED LIS/21NO =>6570LDA \$\$00JER CULPO.K. (CMT.5400DA VARNAGET VARNAGET NERT LINE6020JER STM TTF6580STA CULPO.K. (CMT.5401DA VARNAGET NERT LINE6020JER STM TTF6590JER CULPO.K. (CMT.5500LDA VARNAGET NERT LINE6020JER STM TTF6590JER CULPD.A CCP-CLP5510STA (LER	5370		ADC	\$\$01	ADD 1	5920 DIS	SVA2	LDY	\$ \$00	INIZ	6470		LDA	FUTPIR	GET CURRENT POINTER
5390 CERVAS LDY TMPPTR 5400RETRIEVE TSGUPF INDEX5540LDA (RUTTEN),YFEICH LST CHARACTER6490ACC 4502ADD LINE ENTRY5400LDA TGUPY, RREFTCH LAST CHARACTER550STA VARNAMSAVE IT6500STA UUTTR5410BEQ GETVAAE.O.L.?5960DNBUMP INDEX6510BCC DISVATWATCH PRCING5420DNNOI BUMP INDEX5970LDA (PUTTR), YFETCH 120 CLARACTER6520DIX (PUTTEN), IFETCH 120 CLARACTER6520DIX (PUTTEN), I5400STA YUTTRS980STA VARNAM1SAVE IT6530DIX (PUTTEN), IFETCH 120 CLARACTER6540DIX (PUTTEN), I5401STA YUTTRS980STA VARNAM1SAVE IT6530DISVAT INC CUINTSHOW ME PRINT5401STA YUTTRS980STA VARNAM1SAVE IT6530DISVAT INC CUINTSHOW ME PRINT5403JMP XNRLAND CONTINUE6000LDA (PUTTPR), YFETCH TYPE6550DISVA DORNADORNA DORNA5401JMP XNRLAND CET NEXT LINE6010NAU \$\$10MACK TO 'N' TYPE6550DISVA DORNADORNA DORNA D	5380		STA	(OLDPTR),Y	SAVE IT	5930		STY	COUNT	CLEAR COUNT	6480		ac		
5400LDA TENUFF,YREFETCH LAST CRARACTER5950STA VARMASAVE IT6500STA FUTPTR540BEQ GETVAE.O.L.?5960INBUP INDEX5510BCC DISVAIWATCH PAGING5420INYNOI BUMP INDEX5970LDA (FUTPTR),YFETCH 2ND CHARACTER6510BCC DISVAIWATCH PAGING5430LDA \$500INIZ5980STA VARMANHISAVE IT6530DISVAI INC CUINTSHOW WE PHINE5440STA TYPECLEAR VARIALE TYPE5990INNBUMP INDEX6540LDA CUINTFETCH 1TPE5460GETVAN STA TYPECLEAR VARIALE TYPE6000LDA (PUTPTR),YFETCH 1TPE6550OCP \$\$600CA.K., CONT.5470JMP XVARND GET NEXT LINE6010SPT TMPTRHERCH TTPE6550BCC DISVAEO.K., CONT.5480INIZ6010SPT TMPTRND =>>6570LDA \$\$00O.K., CONT.5490INIZ6010SPT TMPTRND =>>6570LDA \$\$00O.K., CONT.5401LDA VARNHGET VARNAM 1ST CHAR.6050JETE 'TN', \$00SEGU TTI6580SPA CCUNT5510STA (REFEOT), YSAVE AT END OF TABLE6060JSR THE'TN', SAVE TTI6530JSR CTUP5510STA (REFEOT), YSAVE AT END OF TABLE6080JSR CHARACTER6610JSR CTUP5510INYBUMP INDEX6100CMP SEPSPACE ?6650JSR CTUP5510INYBUMP INDEX6100	5390	GETVA9	LDY	TMPPTR	RETRIEVE TXBUFF INDEX	5940	:	LDA	(PUTPTR),Y	FETCH 1ST CHARACTER	6490		ADC	#\$02	ADD LINE ENTRY
5410BEQ GETVANE.O.L.?5960INYBUMP INDEX6510BCC DISVATWATCH PRGING5420INYNO BUMP INDEX5970IDA (FUTPTR), YPETCH 1NO CHARACTER6530DISVAT INC COUNTSHOW WE PERDIT5430IDA VARLAALE TYPECLEAR VARLABLE TYPE5990INYBUMP INDEX6530DISVAT INC COUNTSHOW WE PERDIT5450JHP XVARLAND COMPTINUE6000IDA (FUTPTR), YPETCH ITPE6550CHP \$50ADONE 1075460GETVAR STA TYPECLEAR VARLABLE TYPE6010AND \$\$10MASK TO 'PN' TYPE6550CHP \$50AO.R., CONT.5460JHP XVARLAND GET NEXT LINE6020BED DISV21No =>>6570LDA \$600CLEAR COUNT5480INTZ6040JSR STROUTYESI SHOW IT16580STA COUNTCLEAR COUNT5510STA (REPROT), Y SAVE AT END OF TABLE6050JEYT TMPPTR6560JSR CTLPDO A <cc>CC>CC5520INYBUMP INDEX6070DISV21LDA VARNAMGET VARNAM IST CHAR.6050JSR CTLPDO A <cc>CC>CC>LDA VARNAMGET VARNAM IST CHAR.6050JSR CTLPDONE I CLEAN U5520INYBUMP INDEX6070DISV21LDA VARNAMGET CHARCTER6610JSR CTLPDONE I CLEAN U5530INYBUMP INDEX6100CMP 4SPMGET LINE CARACTER6640JMP DISVAIGO TO NEXT VA5540STA (REPBOT), YSAVE IT6100CMP 4SPM</cc></cc>	5400		LDA	TXBUFF, Y	REPETCH LAST CHARACTER	5950		STA 1	varnam	SAVE IT	6500		STA	FUTPIR	
5420INTNOI BURP TINEX570LDA (PUTTR), YFETCH 12ND CHARACTER6520INC PUTTR1.15430IDA (SOUTA)SAVE IT6530IDAYSAVE IT6530IDAYSHOW HE PRINT5440STA YARNAMISTA VARNAMISAVE IT6540IDAYAILADA COUNTPETCH 1175450JHP WARLAND CONTINUE6000LDA (PUTTR), YPETCH TYPE6550CHP \$\$0ADORE 10?5470JNP XVARAND GET NIST LINE6010ND \$\$10MASK TO 'PN' TYPE6560BCC LISVABDORE 10?5470JNP XVARAND GET NIST LINE6020BED DISV21NO =>6570LDA \$\$00O.K., CONT.5480GON STA (REPBOT), YSAVE AT EXE OF TABLE6050JSR STROUTYESI SHOW ITI6550JSR COLPDO A.K., CONT.5500LDA VARNAMGET VARNAM IST CHAR.6050JSR STROUTYESI SHOW ITI6530JSR CRUPDO A.CR>CLP5510LDA VARNAMGET VARNAM IST CHAR.6050JSR CITCHREPETCH INDEX6610BNE DISVASLOOP 'TIL DON5520INYBUMP INDEX6070DISV21 LDA VARNAMIGET IST CHARACTER6620JSR CITCHCLEARACTER5530LDA VARNAMGET VARIABLE TYPE6110BED DISVASPETCH TINEX6610JNP DISVALGO TO NEXT VA5550INYBUMP INDEX6130JSR CITCHPHINT IT6630JSR CITCHDONE I CLEAN TI5560INYBUMP INDEX6130 <td< td=""><td>5410</td><td></td><td>BEQ</td><td>GETVAA</td><td>E.O.L.?</td><td>5960</td><td></td><td>INY</td><td></td><td>BUMP INDEX</td><td>6510</td><td></td><td>BCC</td><td>DISVA7</td><td>WATCH PAGING</td></td<>	5410		BEQ	GETVAA	E.O.L.?	5960		INY		BUMP INDEX	6510		BCC	DISVA7	WATCH PAGING
5430LDA 4\$00INT25980STA VARNAH1SAVE IT6530DISVAJ INC COUNTSHCW WE PEDDA'5440STA TYPECLEAR VARLALE TYPE5990INYBUMP INDEX6550LDA COUNTPETCH TYPE5450JMP XVAR1AND CONTINUE6000LDA (UTTTR),YPETCH TYPE6550CMP 4\$0ADONE 1075460GETVAA STA TYPECLEAR VARLABLE TYPE6010AND 4\$10MASK TO 'PN' TYPE6550GCD LDSVABO.K., CONT.5460JMP XVARAND GET NEET LINE6020BED DISVANO =>6570LDA 4\$00O.K., CONT.5460JMP XVARGET VARUM IST CHAR.6050BED TISV2INO =>6580STA COUNTCLEAR COUNT5500LDA VARNAMGET VARUM IST CHAR.6050BET THPPTRREFETCH INDEX6610BNE DISVAEDOAE 1075510STA (REPBOT), YSAVE T6060LDY TMPPTRREFETCH INDEX6610BNE DISVAEDOAE 1025510INYBUMP INDEX6090LDA VARNAHGET 157 CHARACTER6620JER CLFDOAE 102FNAE5510INYBUMP INDEX6100CMP 450AJES 110 CHARTER6660JER CLFDOAE 102FNAE5510INYBUMP INDEX6100CMP 450AJES 110 CHARTER6660JER CLFDOAE 102FNAE5510INYBUMP INDEX6100CMP 450AJES 110 CHARTER6660JER CLFDOAE 102FNAE5510INYBUMP INDEX6100CMP 450A<	5420		INY		NO: BUMP INDEX	5970		LDA	(PUTPTR),Y	FETCH 2ND CHARACTER	6520		INC	PUTPTR+1	
5440STA TYPECLEAR VARIABLE TYPE5990INYBURP INDEX6540LAA COUNTFETCH IT5450JMP XVARAND COMTINUE6000LDA (RUTPIN), YFETCH TYPE6550COP \$\$50.DOP \$\$10.NO \$\$10.NASK TO 'PN' TYPE6550.DOP \$\$10.DOP \$\$10.DASK TO 'PN' TYPE6550.DOP \$\$10.DOP \$\$10.DASK TO 'PN' TYPE6550.DOC DISVABO.K., CONT.5470JNP XVARAND GET NEXT LINE6020BEQ DISV21.NO =>>6570.LDA \$\$00.O.K., CONT.5480.GON DISVARGET VARDAM IST CHAR.6030.STR TUPTFIS SHOW TTI6580.STA CUINTCLEAR COUNT5490DIA VARNAMGET TAD OF TABLE6060.LDY TMPPTRREFETCH INDEX6610.BAR CHAF ACACHER0.K., CONT.5500LDA VARNAM IST CHAR.6070DISV21.LDA VARNAM GET 15T CHARACTER6610.BAR CHAF ACACHER6610.BAR CHAF ACACHER0.K., CONT.5500LDA VARNAM-1GET 2ND CHARACTER6090.LDA VARNAM-1GET 2ND CHARACTER6630.JSR CHAF DONE! CLEAN U5510STA (REFEOT), YSAVE IT6090.LDA VARNAM-1GET 2ND CHARACTER6650.JSR CHAF DONE! CLEAN U5510IDA VARNAM-1GET 2ND CHARACTER6100.CMP 4SPSAVE CP6650.JSR CHAF5510IDA VARNAM-1GET 2ND CHARACTER6100.CMP 4SPSAVE CP6650.JSR CHAF5510IDA STA (REFEOT), YSHOW INDEX6100.DISVAI <t< td=""><td>5430</td><td></td><td>LDA</td><td>\$\$00</td><td>INIZ</td><td>5980</td><td></td><td>STA '</td><td>VARNAM+1</td><td>SAVE IT</td><td>6530</td><td>DISVA7</td><td>INC</td><td>COUNT</td><td>SHOW WE PRINTED ONE</td></t<>	5430		LDA	\$ \$00	INIZ	5980		STA '	VARNAM+1	SAVE IT	6530	DISVA7	INC	COUNT	SHOW WE PRINTED ONE
5450JMP XVARIAND CONTINUE6000LDA (UTUTEN), YPETCE TYPE6550CMP \$\$0.ADONE 1075460JMP XVARAND GET NEXT LINE6010AND \$\$1.0MASK TO 'FN' TYPE6550BCC DISVABO.K., CONT.5480JMP XVARAND GET NEXT LINE6020BEQ DISV21NO =>>6570LDA \$\$00O.K., CONT.5480MEMVAR LDY \$\$00INIZ6040JSR STRCUTYES1 SHOW IT16580STA COUNTCLEAR COUNT5500LDA VARNAMGET VARNAM IST CHAR.6050BYT 'FN', \$006600DISVAB DEC NRSHOW ACC>/FDA <cc>/FDA <cc< td="">A <cc>/FDA <cc>/FDA <cc< td="">A <cc>/FDA <cc< td="">A <cc< t<="" td=""><td>5440</td><td></td><td>STA</td><td>TYPE</td><td>CLEAR VARIABLE TYPE</td><td>5990</td><td></td><td>INY</td><td></td><td>BUND INDEX</td><td>6540</td><td>•</td><td>LDA</td><td>COUNT</td><td>PETCH IT</td></cc<></cc<></cc<></cc<></cc<></cc<></cc<></cc<></cc<></cc<></cc<></cc<></cc<></cc<></cc<></cc<></cc<></cc<></cc<></cc<></cc<></cc<></cc<></cc<></cc<></cc<></cc<></cc<></cc<></cc<></cc<></cc></cc<></cc></cc></cc<></cc></cc></cc></cc></cc></cc></cc></cc></cc></cc></cc></cc></cc></cc></cc></cc></cc></cc></cc></cc></cc></cc></cc></cc></cc></cc></cc></cc></cc></cc></cc></cc></cc></cc></cc></cc></cc></cc></cc>	5440		STA	TYPE	CLEAR VARIABLE TYPE	5990		INY		BUND INDEX	6540	•	LDA	COUNT	PETCH IT
5460 (EETVAA STA TYPECLEAR VARIABLE TYPE6010NO #\$10MASK TO 'PN' TYPE6560ECC DISVABO.K., CONT.5470JMP XVARAND GET NERT LINE6020BED DISV21NO =>6570LDA $\$500$ CLEAR COUNTCLEAR COUNTC	5450		JMP	XVARL	AND CONTINUE	6000		LDA	(FUTPTR),Y	PEICH TYPE	6550		QФР	#\$0A	DONE 10?
5470JHP XVARNAD GET NERT LINE6020BED DISV21NO ==>6570LDA 4\$005480,600BTY TMPPER600STY TMPPER5580STA COUNTCLEAR COUNT5490NEWVAR LDY \$\$00INIZ6040JSR STROITYESI SHOW ITI5590STA CREPTD A $\langle CR > LD \rangle$ 5510STA (REPOT), YSVE AT END OF TABLE6050LDY TMPPTRREPETCH INDEX6610DISVA BDC NRSHOW A PASS5520INYBUMP INDEX6070DISV21 LDA VARNAMGET 15T CHARACTER6620JSR CRLFDONE! CLEAR U5530LDA VARNAM+1GET 2ND CHARACTER6080JSR CHCHREPETCH INDEX6610JSR CRLFDONE! CLEAN U5530STA (REPOT), YSAVE IT6090LDA VARNAM+1GET 2ND CHARACTER6640JMP DISVALGO TO NEXT VA5550INYBUMP INDEX6100OP #SFSTACE ?66600.END XREF15550INYBUMP INDEX6130DISVAL LDA (FUTTR), YFETCH VARIABLE TYPE6650,.END XREF15570STA (REPOT), YSAVE IT6120JSR CUTCHNO! FRINT IT!66600.END XREF15580INYBUMP INDEX6130DISVAL (HUTTR), YFETCH VARIABLE TYPE6660.END XREF15570STA (REPSOT), YSAVE IT6140HASAVE CON STACK.END XREF15580INYBUMP INDEX6130DISVAHUA SAVE CON STACK.END XREF15610INAREPSOT, Y <td< td=""><td>5460</td><td>GETVAA</td><td>STA</td><td>TYPE</td><td>CLEAR VARIABLE TYPE</td><td>6010</td><td></td><td>and</td><td>\$\$10</td><td>MASK TO 'FN' TYPE</td><td>6560</td><td></td><td>BCC</td><td>DISVA8</td><td>O.K., CONT.</td></td<>	5460	GETVAA	STA	TYPE	CLEAR VARIABLE TYPE	6010		and	\$ \$10	MASK TO 'FN' TYPE	6560		BCC	DISVA8	O.K., CONT.
5460 6030STY TMPPTR5580STA COUNTCLEAR COUNT5490 NEWVAR LDY \$\$00 INIZ6040 JGR STEOUTYESI SHOW IT16590 JGR CRLFDO A (CR>CLP)5500 LDA VARNAMGET VARNAM IST CHAR.6050 BYTE 'PN', \$006500 DISVAB DEC NRSHOA A CR>CLP5510 STA (REFBOT), YSAVE AT END OF THALE6050 LDA Y TMPPTRREFETCH INDEX6610 BNE DISVASLOOP 'TIL DON5520 INYBUMP INDEX6070 DISV21 LDA VARNAMGET 19T CHARACTER6620 JGR CRLFDONE! CLEAN U5530 LDA VARNAM+1GET 2ND CHARACTER6080 JGR CUTCHPRINT IT6630 JGR CRLFDONE! CLEAN U5540 STA (REFBOT), YSAVE IT6090 LDA VARNAM+1 GET 2ND CHARACTER6640 JMP DISVALGO TO NEXT VA5550 IDA TYPEGET VARLABLE TYPE6110 BED DISVA3 YESI IGNORE TTI =>>6660 .END XREF15570 STR (REFBOT), YSAVE IT6120 JGR CUTCHNOI FRINT ITI 6660 .END XREF15580 IDA \$\$001 INITZ6140 HASAVE CN STRACKSAVE KIST DA \$\$001 STRA CUTCH \$\$001 JENTY6150 AND \$\$0F NASK TO LON NYBELE\$\$500 .END XREF15610 INYBUMP INDEX6160 TAXFPICH VARLABLE TYPE\$\$100 RET LINE \$\$ LSB6170 LDA KIND, XFPICH VARLABLE TYPE5610 INYBUMP INDEX6160 BED DISVAF INA \$\$100 ARRAY<	5470		JMP	XVAR	AND GET NEXT LINE	6020		BEQ	DISV21	NO ==>	6570		LDA	\$\$00	
5490 NEWVAR LDY #\$00 INIZ 6040 JSR STECUT YESI SHOW ITI 6590 JSR CRLF DOA (CR>LF) DOA (CR FL)	5480;					6030		8TY	TMPPIR		6580		STA	COUNT	CLEAR COUNT
5500LDA VARNAMGET VARNAM LST CHAR. 6050 .BTE 'FN', $$00$ 6600 DISVASDEC NRSLOW A PASS5510STA (REFBOT), YSAVE AT END OF TABLE 6060 LDY TMPPTRREFETCH INDEX 6610 BNE DISVASLDOP 'TIL DON5520INYBUMP INDEX 6070 DISV21LDA VARNAMGET 1ST CHARACTER 6620 JSR CTLPDONEI CLEAN U5530LDA VARNAMH1GET 2ND CHARACTER 6080 JSR CUTCHPRINT IT 6630 JSR CTLPDONEI CLEAN U5540STA (REFBOT), YSAVE IT 6090 LDA VARNAMH1GET 2ND CHARACTER 6660 JRP DISVA1GO TO NEXT VA5550INNBUMP INDEX 6100 OMP \$SPSPACE ? 6650 ,END XREF1 $G0$ TO NEXT VA5570STA (REFBOT), YSAVE IT 6120 JSR OTCHNOI FRINT IT 6660 .END XREF15580INNBUMP INDEX 6130 DISVA3LDA (PUTPR), YPETCH VARIABLE TYPE 6660 .END XREF15580INNBUMP INDEX 6130 DISVA3LDA (PUTPR), YPETCH VARIABLE TYPE 6660 .END XREF15590LDA \$\$001TNIZGET LINE \$\$1.58G170LDA KIND, XFETCH TYPE CHARACTER 5620 LDA KIND, XFETCH TYPE CHARACTER5610INYBUMP INDEX 6160 TAXPUT IN X 8000 FINT 8000 TI5620LDA LNLOGET LINE \$\$1.58G170LDA KIND, XFETCH TYPE CHARACTER 56200 56	5490	NEWVAR	LDY	#\$00	INIZ	6040		JSR	STROUT	YESI SHOW ITI	6590		JSR	CRLP	do a «cr»«lf»
5510STA (REFBOT), Y BUMP INDEXSAVE AT END OF TABLE6050LINY TMPPTR MARKAMH1REFETCH INDEX6610BNE DISVA6LOOP 'TTL DON5520INYBUMP INDEX6070DISV21LIA VARNAMGET 1ST CHARACTER6620JER CRLFDONE! CLEAN U5530LDA VARNAM1GET 2ND CHARACTER6080JSR CUTCHPRINT IT6630JSR CRLPDONE! CLEAN U5540STA (REFBOT), Y S550KAVE IT6090LDA VARNAM1GET 2ND CHARACTER6640JWP DISVA1CO TO NEXT VA5550IDA TYPEGET VARIABLE TYPE6110BED DISVA3YESI LENGRE TT! =>>6660.END XREF15570STA (REFBOT), Y S580KAVE IT6120JSR CUTCHNOI FRINT IT!6660.END XREF15580IDNBUMP INDEX6130DISVA3LDA (FNTR), YFETCH VARIABLE TYPE.END XREF15580IDNBUMP INDEX6160TAXPUT IN X5600STA (REFBOT), Y S640STA (REFBOT), YSHOW 1 ENTRY6150ADD \$\$00 KARK TO LOW NYBBLE5610INNBUMP INDEX6190JSR CUTCHNOI CHIBRI SHOW IT!5620LDA LINLOGET LINE # LSB6170LDA KIND, XFETCH TYPE CHARACTER5620LDA LINLOGET LINE # LSB6120JSR CUTCHNOI CHIBRI SHOW IT!5630STA (REFBOT), YBUMP INDEX6130JSR CUTCHNOI CHIBRI SHOW IT!5640INYBUMP INDEX6190JSR STRCUTSHOW ARRA	5500		LDA	VARNAM	GET VARNAM 1ST CHAR.	6050		.BYT	'E 'FN',\$00		6600	DISVA8	DEC	NR	SHOW A PASS
5520INYEUMP INDEX6070DISV21LDA VARNAMGET 1ST CHARACTER6620JER CPLFDONE!CLEAN U5530LDA VARNAM+1GET 2ND CHARACTER6680JSR CPLFFINT IT6630JSR CPLFDONE!CLEAN U5540STA (REPBOT), YSAVE IT6090LDA VARNAM+1GET 2ND CHARACTER6640JMP DISVALGO TO NEXT VA5550INYBUMP INDEX6100CMP #SPSPACE ?6650,5570STA (REPBOT), YSAVE IT6120JSR CUTCHNO! FRINT IT!5580INYBUMP INDEX6130DISVA3LDA (PUTPR), YFETCH VARLABLE TYPE6660.END XREF15570STA (REPBOT), YSAVE IT6120JSR CUTCHNO! FRINT IT!.END XREF15580INYBUMP INDEX6130DISVA3LDA (PUTPR), YFETCH VARLABLE TYPE5590LDA \$\$01INIZ6140PHASAVE CN STACK5610INYBUMP INDEX6150AND \$\$0FMASK TO LOW NYBELE5610INYBUMP INDEX6160TAXFUT IN X5620LDA LNLOGET LINE \$ LSB6170LDA KIND, XFETCH TYPE CHARACTER5630STA (REPBOT), YSAVE IT6220JSR CUTCHNO! OTHER! SHOW IT!5630LDA \$4066220JSR STRCUTSHOW ARRAY5650LDA NIHIGET LINE \$ MSB6200DISVA HARETRIEVE CHGINAL5660STA (REPBOT), YSAVE IT6220STS STRCUTSHO	5510		STA	(REFBOT), Y	SAVE AT END OF TABLE	6060		LDY	IMPPIR	REFETCH INDEX	6610		BNE	DISVA6	LOOP 'TIL DONE
5530LDA VARNAM+1GET 2ND CHARACTER6080JSR CUTCHPRINT IT6630JSR CRLP5540STR (REFBOT), YSAVE IT6090LDA VARNAM+1GET 2ND CHARACTER6640JMP DISVALGO TO NEXT VA5550INYBUMP INDEX6110BED DISVA3YESI LGNORE ITI6650;.END XREF15570STR (REFBOT), YSAVE IT6120JSR OTCHNOI PRINT ITI6660.END XREF15580INYBUMP INDEX6130DISVA3YESI LGNORE ITI6660.END XREF15580INYBUMP INDEX6130DISVA3LDA (FUTFR), YFETCH VARIABLE TYPE55905500STA (REFBOT), YSIW1ENTRY6150AND \$60PMASK TO LCM NYBELE5501INYBUMP INDEX6160TAXFUT UN X5610INYBUMP INDEX6160TAXFUT UN X5620IDA LNLOGET LINE # LSB61170LDA KIND, XFETCH TYPE CHARACTER5630STA (REFBOT), YBUMP INDEX6180BED DISVMFLA5640INYBUMP INDEX6190JSR CUTCHNOI CHIERI SHON ITI5650LDA LNLOGET LINE # LSB6200DISVM PLARETRIEVE ORIGINAL5640INYBUMP INDEX6210BHL DISVA5NO SUBSCRUPT?5670LDA \$6066220JSR TMEPTSHOW ARRAY5670LDA \$6066220JSR TMEPTSHOW ARRAY5690ADC REFBOTADD TO REFERENCE END6240 </td <td>5520</td> <td></td> <td>INY</td> <td></td> <td>BUMP INDEX</td> <td>6070 DI</td> <td>SV21</td> <td>LDA</td> <td>varnam</td> <td>GET 15T CHARACTER</td> <td>6620</td> <td></td> <td>JSR</td> <td>CRLF</td> <td>DONE! CLEAN UP LINE!</td>	5520		INY		BUMP INDEX	6070 DI	SV21	LDA	varnam	GET 15T CHARACTER	6620		JSR	CRLF	DONE! CLEAN UP LINE!
5540STA (REPBOT), YSAVE IT6090LDA VARNAM-1GET ZND CHARACTER6640JMP DISVALGO TO NEXT VA5550INYBUMP INDEX6100OMP #SPSPACE ?6650;6650;6550;5570STA (REPBOT), YSAVE IT6110BED DISVA3YESI IGNORE ITI =>6660.END XREP15570STA (REPBOT), YSAVE IT6120JSR OUTCHNOI FRINT ITI6660.END XREP15580INNBUMP INDEX6130DISVA3LDA (NUTTR), YFETCH VARIABLE TYPE5590LDA \$\$01INIZ6140HHASAVE ON STACK5610STA (REPBOT), YSHOW 1 ENTRY6150AND \$\$0FMASK TO LOW NYBBLE5610INNBUMP INDEX6150AND \$\$0FMASK TO LOW NYBBLE5620LDA LIALOGET LINE \$ LSB6170LDA KIND, XFETCH TYPE CHARACTER5630STA (REPBOT), YBUMP INDEX6190JSR OUTCHNOI ONIERT SHOW ITI5630STA (REPBOT), YSAVE IT6220STY TMPPTRYESI SAVE Y BERE5660STA (REPBOT), YSAVE IT6210BHL DISVASNO SUBSCRIPT? ==>5670LDA \$\$066220STY TMPPTRYESI SAVE Y BERE5680CLC6220STY TMPPTRYESI SAVE Y BERE5680CLC6230JSR STROUTSHOW ARRAY5710LDA REPSOTSAVE AS NEW BOTTOM6220BYT TWPPTR5710LDA REPSOTSAVE AS NEW BOTTOM6220LYT TWPPTR<	5530		LDA	VARNAM+1	GET 2ND CHARACTER	6080		JSR	OUTCH	PRINT IT	6630		JSR	CRLP	
5550INYEURP INDEX 6100 OUP #SPSPACE ? 6650 ; 5560 LDA TYPEGET VARIABLE TYPE 6110 BEQ DISVA3YESI IGNORE TTI =>> 6660 .END XREF1 5570 STA (REFBOT), YSAVE IT 6120 JSR OUTCHNOI FRINT ITI 5580 INYBUMP INDEX 6130 DISVA3LDA (RUTPTR), YFETCH VARIABLE TYPE 5590 LDA \$\$01INIZ 6140 PHASAVE ON STACK 5600 STA (REFBOT), YSIW 1 ENTRY 6150 ADD \$\$07MASK TO LOW NTBBLE 5610 INYBUMP INDEX 6160 TAXFUT IN X 5620 LDA LNLOGET LINE \$ LSB 6170 LDA KIND, XFETCH TYPE CIARACTER 5630 STA (REFBOT), YSUMP INDEX 6180 BEQ DISVA4P.P. ? =>> DISVA4 5640 INYBUMP INDEX 6190 JSR QUTCHNOI OTHERI SHOW ITI 5650 LDA LNHIGET LINE \$ MSB 6200 DISVA4PLARETRIEVE CHGINAL 5650 LDA LNHIGET LINE \$ MSB 6200 DISVA4PLARETRIEVE CHGINAL 5660 STA (REFBOT), YSAVE IT 6210 BFL DISVA5NO SUBSCRIPT? =>> 5670 LDA \$\$06G220STY TMPFTRYESI SAVE Y HERE 5680 CLC6230JSR STROUTSHO ARRAY 5710 JAR REFBOTADD TO REFERENCE END 6240 BYT Y 5710 JAR REFBOTADD TO REFERENCE END 6240 BYT Y 5710 JAR REFBOTAND TO REFERENCE EN	5540		STA	(REPBOT),Y	SAVE IT	6090		LDA	VARNAM+1	GET 2ND CHARACTER	6640		JMP	DISVAL	GO TO NEXT VARIABLE
5560LDA TYPEGET VARLABLE TYPE6110BED DISVA3YESI LGNORE TTI \Longrightarrow 6660.END XREF15570STR (REFBOT), YSAVE IT6120JSR OUTCHNOI FRINT TTI5500INNBUMP INDEX6130DISVA3LDA (RUTPTR), YFETCH VARLABLE TYPE5580LDA \$\$01INIZ6140PHASAVE ON STRACK5600STA (REFBOT), YSHOW 1 ENTRY6150AND \$\$0PMARS TO LOW NYBELE5610INVBUMP INDEX6160TAXPAT IN X5620LDA LINLOGET LINE \$ LSB6170LDA KIND, XFETCH TYPE CHARACTER5620LDA LINLOGET LINE \$ LSB6170LDA KIND, XFETCH TYPE CHARACTER5620LDA LINLOGET LINE \$ MSB6200DISVA4PLA5640INYBUMP INDEX6190JSR QUTCHNOI OTHERI SHOW ITI5650LDA LNHIGET LINE \$ MSB6200DISVA4PLA5660STA (REFBOT), YSAVE TT6210BHL DISVA5NO SUBSCHIP? =>>5670LDA \$\$066220SIY TMPFTRYESI SAVE Y BERE5680CLC6220JSR STROUTSHOW ARRAY5690ADC REFBOTADD TO REFERENCE END6240BYTE '(X)', \$005710LDA REFDOT+AND HOULE MSB6250LDY TWPFTR5710LDA REFDOT+SHOW BOTICH6250LDY TWPFTR5710LDA REFDOT+SHOULE MSB6260DISVA5 INY5710LDA REFDOT+SHOULE MSB6250<	5550		INY		BUMP INDEX	6100		CM₽	#SP	SPACE ?	6650;				
5570 STA (REFBOT), Y SAVE IT 6120 JSR QUTCH NOI FRINT ITI 5580 INY BUMP INDEX 6130 DISVA3 LDA (FUTPTR), Y FETCH VARIABLE TYPE 5590 LDA \$\$01 INIZ 6140 PHA SAVE ON STACK 5500 STA (REFBOT), Y SKOW 1 ENTRY 6150 AND \$\$0P MASK TO LOW NYBBLE 5610 INY BUMP INDEX 6160 TAX FUT IN X 500 5610 INY BUMP INDEX 6160 TAX FUT IN X 500 5620 LDA LNLO GET LINE # LSB 6170 LDA KIND, X FETCH TYPE CHARACTER 5630 STA (REFBOT), Y G180 BEQ DISVM P, P, T ==> DISVM 5640 INY BUMP INDEX 6190 JSR CUTCH NOI OTHERI SHOW ITI 5650 LDA LINH GET LINE # MSB 6200 DISVM PLA RETRIEVE CHIGINAL SUP STACK SUP STACK SUP STACK 5660 STA (REFBOT), Y SAVE IT 6220 STY TMPPTR YESI SAVE Y HERE 5630 G220 STY TMPPTR YESI SAVE Y HERE 5650 G220 STY TMPPTR SHOW ARRAY<	5560		LDA	TYPE	GET VARIABLE TYPE	6110		BEQ	DISVA3	YESI IGNORE IT! ==>	6660		, ENI	XREF1	
5580 INY BUMP INDEX 6130 DISVA3 LDA (RUTPTR), Y FETCH VARLAGLE TYPE 5590 LDA \$\$01 INIZ 6140 PHA SAVE ON STACK 5500 STA (REFBOT), Y SKW 1 ENTRY 6150 AND \$\$0F MASK TO LOW STACK 5610 INY BUMP INDEX 6160 TAX FUT IN X 5620 LDA LNLO GET LINE \$ LSB 6170 LDA KIND, X FUT IN X 5620 LDA LNLO GET LINE \$ LSB 6170 LDA KIND, X FUTCH TYPE CHARACTER 5630 STA (REFBOT), Y SUMP INDEX 6180 BEQ DISVA P.P. ? =>> DISVA 5640 INY BUMP INDEX 6190 JSR OTTCH NOI OTHER! SHOW IT! 5650 LDA LNHI GET LINE \$ MSB 6200 DISVA PLA RETRIEVE ORIGINAL 5650 LDA LNHI GET LINE \$ MSB 6200 DISVA PLA RETRIEVE ORIGINAL 5660 CLC 6220 JSR STROUT SHOW ARRAY SHOW ARRAY 5670 LDA \$\$06 6240 BYT TWPTR YESI SAVE Y HERE 5680 5700 STA REFBOT	5570		STA	(REFBOT),Y	SAVE IT	6120		JSR	OUTCH	NOI PRINT ITI					
5590 LDA 4\$01 INIZ 6140 PHA SAVE ON STRACK 5600 STA (REFBOT), Y SHOW 1 ENTRY 6150 AND 4\$0F MASK TO LOW NYBBLE 5610 INY BUMP INDEX 6160 TAX PUT IN X 5620 LDA LNLO GET LINE # LSB 6170 LDA KIND, X FETCH TYPE CHARACTER 5630 STA (REFBOT), Y BUMP INDEX 6180 BED BLOSVA P.P. ? => DISVA 5640 INY BUMP INDEX 6190 JER OUTCH NOI OTHERI SHOW ITI S650 5650 LDA LNHI GET LINE # MSB 6200 DISVAH PLA RETRIEVE ORIGINAL 5650 STA (REFBOT), Y SAVE IT 6210 BHL DISVAS NO SUBSCRIPT? ==> 5670 LDA \$\$06 6220 STY TMPFTR YESI SAVE Y HERE 5630 G200 JER STROUT SHOW ARRAY 5680 CLC 6220 JER STROUT SHOW ARRAY 5640 BYEE '(X) ', \$00 5710 JA REFBOT ADD TO REFERENCE END 6240 BYTE '(X) ', \$00 SHOW ARRAY 5710 JA REFBOTH MANDLE MEB <t< td=""><td>5580</td><td></td><td>INY</td><td></td><td>BUMP INDEX</td><td>6130 DI</td><td>EVA3</td><td>LDA</td><td>(FUTPTR),Y</td><td>PETCH VARIABLE TYPE</td><td></td><td></td><td></td><td></td><td></td></t<>	5580		INY		BUMP INDEX	6130 DI	EVA3	LDA	(FUTPTR),Y	PETCH VARIABLE TYPE					
5600 STA (REFEOT), YSHOW 1 ENTRY 6150 AND $\$ 00F$ NASK TO LOW NYBELE 5610 INYBUMP INDEX 6160 TAXPUT IN X 5620 LDA LNLOGET LINE $\$$ LSB 6170 IDA KIND, XFEUCH TYPE CHARACTER 5630 STA (REFBOT), Y 6180 BEQ DISVAMP.P.? \Rightarrow DISVAM 5640 INYBUMP INDEX 6190 JSR OUTCHNOI OTHERI SHOW ITI 5650 LDA LNHIGET LINE $\$$ MSB 6200 DISVAMP.A.RETRIEVE CRIGINAL 5650 LDA LNHIGET LINE $\$$ MSB 6220 DISVASNO SUBSCRIPT? 5660 STA (REFBOT), YSAVE IT 6210 BHL DISVASNO SUBSCRIPT? 5670 LDA $\$ 066$ 6220 STY TMPPTRYESI SAVE Y HERE 5670 ADC REFBOTADD TO REFERENCE END 6240 .BYTE '(X)', \$00 5710 SAVE AS NEW HOTTOM 6250 LDY TMPPTRRETRIEVE Y 5710 LDA REFDOTLSAVE AS NEW BOTTOM 6250 LDY TMPPTR 5710 LDA REFDOTLSAVE AS NEW EDTEM 6260 DISVAS INYBUMP INDEX	5590		LDA	\$\$01	INIZ	6140		PHA		SAVE ON STACK					
5610 INY BUMP INDEX 6160 TAX FUT IN X 5620 LDA LNLO GET LINE # LSB 6170 LDA KIND, X FFTCH TYPE CHARACTER 5620 STA (REFBOT), Y 6180 BEQ DISVA P.P.7 m>> DISVA 5640 INY BUMP INDEX 6190 JSR CUTCH NOI OTHER! SHOW TTI 5650 LDA LNHI GET LINE # MSB 6200 DISVA P.P.7 m>> DISVA 5660 STA (REFBOT), Y SAVE IT 6210 BHL DISVAS NOI OTHER! SHOW TTI S660 6220 STA (REFBOT), SAVE IT 6210 BHL DISVAS NO SUBSCRIPT? m>> 5670 LDA \$\$06 6220 STY TMPFTR YESI SAVE Y HERE 5680 CLC 6230 JSR STROUT SHOW ARRAY 5690 ADC REFBOT ADD TO REFERENCE END 6240 BYTE '(X)',\$00 SHOW ARRAY 5710 LDA REFDOTH SNUE AS NEW BOTTOM 62250 LDY TMPFTR RETRIEVE Y 5710 LDA REFDOTH SNUE AS NEW BOTOM 62500 LDY TMPFTR SHOW INDEX	5600		STA	(REFBOT),Y	SHOW 1 ENTRY	6150		and	#\$0F	MASK TO LOW NYBBLE					
5520 LDA LNLO GET LINE # LSB 6170 LDA KIND,X FETCH TYPE CHARACTER 5630 STA (REFBOT),Y 6180 BEQ DISVM F,P,7 model DISVM 5640 INY BUMP INDEX 6190 JSR OUTCH NOI OTHERI SHOW ITI 5650 LDA LNHI GET LINE # MSB 6200 DISVM FLA RETRIEVE CRIGINAL 5660 STA (REFBOT), Y SAVE IT 6210 BHL DISVAS NO SUBSCRIPT? model 5670 LDA \$\$06 6220 STY TMPPTR YESI SAVE Y HERE 6230 JSR STRCUT SHOW ARRAY 5680 CLC 6230 JSR STRCUT SHOW ARRAY 5700 STA REFBOT SAVE AS NEW BOTTOM 6220 JSY TMPPTR Y SHOW ARRAY 5700 STA REFBOT SAVE AS NEW BOTTOM 6220 JSY TMPPTR Y SHOW ARRAY 5710 LDA REFBOT SAVE AS NEW BOTTOM 6220 JY TMPPTR Y SHOW ARRAY 5710 JAR REFBOTH SAVE AS NEW BOTTOM 6220 JY TMPPTR Y SHOW ARRAY STA 5710 LDA RE	5610		INY	•	BUMP INDEX	6160		TAX		FUT IN X					
5630STA (REFBOT), Y6180BED DISVMA $P, P, P \Rightarrow DISVMA$ 5640INYBUMP INDEX6190JSR OUTCHNOI OTHERI SHOW ITI5650LDA LMHIGET LINE # MSB6200 DISVMA FLARETRIEVE ORIGINAL5660STA (REFBOT), YSAVE IT6210BFL DISVASNO SUBSCRIPT?5670LDA \$\$066220STY TMPPTRYESI SAVE Y HERE5670CLC6230JSR STROUTSHOW ARRAY5690ADC REFBOTADD TO REFERENCE END6240BYE '(X)', \$005710LDA REFBOTSAVE AS NEW BOTTOM6250LDY TMPPTR5710LDA REFBOTHSAVE AS NEW BOTTOM6250LDY TMPPTR5710LDA REFBOTHSAVE AS NEW BOTTOM6250LDY TMPPTR5710LDA REFDOTHADD TO REFERENCE END6240BYE '(X)', \$005710LDA REFDOTHSAVE AS NEW BOTTOM6250LDY TMPPTR5710LDA REFDOTHSAVE AS NEW BOTTOM6250LDY TMPPTR	5620		LDA	LNLO	GET LINE # LSB	6170		LDA'	KIND, X	FETCH TYPE CHARACTER					
5640 INY BUMP INDEX 6190 JSR QITCH NOI OTHERI SHOW ITI 5650 LDA LNHI GET LINE # MSB 6200 DISVAH PLA RETRIEVE ORIGINAL 5660 STA (REFBOT), Y SAVE IT 6210 BHL DISVAS NO SUBSCRIPT? 5670 LDA #\$06 6220 STY TMPFTR YESI SAVE Y HERE 5680 CLC 6230 JSR STROIT SHOW ARRAY 5690 ADC REFBOT ADD TO REFERENCE END 6240 BYTE '(X)',\$00 5700 STA REFBOT SNE M BOTTON 6220 BYTE '(X)', \$00 5710 LDA REFBOT ADD TO REFERENCE END 6240 BYTE '(X)', \$00 5710 LDA REFBOT BANDLE MSB 6250 LDY TMPFTR 5710 LDA REFDOTH BANDLE MSB 6260 LDY MEPTR	5630		STA	(REFBOT) .Y		6180		BEQ	DISVM	F.P.? ==> DISVA4					
5650 LDA LNHI GET LINE # MSB 6200 DISVA4 FLA RETRIEVE ORIGINAL 5660 STA (REFBOT), Y SAVE IT 6210 BHL DISVA5 NO SUBSCRIPT? m>> 5670 LDA \$\$06 6220 STY TMPPTR YESI SAVE Y HERE 5680 6220 STY TMPPTR YESI SAVE Y HERE 5680 ADC REFBOT ADD TO REFERENCE END 6240 BYTE '(X)', \$00 5700 STA REFBOT SAVE AS NEW BOTTOM. 6250 LDY TMPPTR RETRIEVE Y 5710 LDA REFBOTH BANDLE MSB 6260 DISVA5 INY BUMP INDEX	5640		INY		BUMP INDEX	6190		JSR	OUTCH	NOI OTHERI SHOW ITI					-
5660 STA (REFBOT), Y SAVE IT 6210 BHL DISVAS NO SUBSCRIPT? max 5670 LDA \$\$06 6220 STY TMPPTR YESI SAVE Y HERE 5680 CLC 6230 JSR STROUT SHOW ARRAY 5690 ADC REFBOT ADD TO REFERENCE END 6240 BYE '(X)', SOO 5700 STA REFBOT SAVE AS NEW BOTTOM. 6250 LDY TMPPTR 5710 LDA REFORTH SAVE AS NEW BOTTOM. 6250 LDY TMPPTR 5710 LDA REFORTH BANDLE MSB 6260 DISVAS INY	5650		LDA	LNHI	GET LINE # MSB	6200 DI	SVM	PLA		RETRIEVE ORIGINAL					
5670 LDA \$\$06 6220 STY TMPPTR YESI SAVE Y HERE 5680 CLC 6230 JSR STRCUT SHOW ARRAY 5690 ADC REFEOT ADD TO REFERENCE END 6240 BYTE '(X)', SOO 5700 STA REFEOT SAVE AS NEW BOTTOM. 6250 LJY TMPPTR 5710 LDA REFEOT+ SAVE AS NEW BOTTOM. 6250 LJY TMPPTR 5710 LDA REFEOT+ SAVE AS NEW BOTTOM. 6250 LJY TMPPTR	5660		STA	(REFBOT) .Y	SAVE IT	6210		BPL	DISVAS	NO SUBSCRIPT? ==>					
5680 CLC 6230 JSR STRCUT SHOW ARRAY 5690 ADC REFBOT ADD TO REFERENCE END 6240 BYTE '(X) ', \$00 5700 STA REFBOT SAVE AS NEW BOTTOM. 6250 LDY TMPPTR RETRIEVE Y 5710 LDA REFBOT+1 BHADLE MSB 6260 DISVAS INY BUMP INDEX	5670		LDA	\$\$06		6220		STY	TMPPTR	YESI SAVE Y HERE					
5690 ADC REFBOT ADD TO REFERENCE END 6240 .BYTE '(X)',\$00 5700 STA REFBOT SAVE AS NEW BOTTOM. 6250 LUY TWEPTR RETRIEVE Y 5710 LDA REFBOTH BANDLE MSB 6260 DISVAS INY BUMP INDEX	5680		CLC			6230		JSR	STROUT	SHOW ARRAY				<i>,</i>	
5700 STA REFBOT SAVE AS NEW BOTTOM. 6250 LDY TMPPTR RETRIEVE Y 5710 LDA REFBOT+1 HANDLE MSB 6260 DISVAS INY BUMP INDEX	56 90		ADC	REFBOT	ADD TO REFERENCE END	6240		BYT	TE '(X)',\$00						
5710 LDA REFBOTH 1 HANDLE MSB 6260 DISVAS INY BUMP INDEX	5700		STA	REFBOT	SAVE AS NEW BOTTOM	6250		LDY	TMPPTR	RETRIEVE Y					
	5710		LDA	REFBOT+1	HANDLE MSB	6260 DI	SVA5	INY		BUMP INDEX					
5720 ADC #\$00 6270 LDA (PUTPTR), Y FETCH # OF REFS.	5720		ADC	#\$00		6270		LDA	(PUTPIR),Y	FETCH # OF REFS.					

I H S Computer Services Introduces ALPHA/OMEGA Series Software

ALPHA/OMEGA Business Management System

- ***** Integrated Accounting System for hard disks -- G/L, A/R, A/P, Inventory, P/R, POS.
- $m{*}$ File locking on all Inventory and A/R functions for multi-user systems.
- Many advanced features, such as Departments (up to 99), Automatic Billing, Budgeting, Comparison with Previous Year, Detailed Cust/Vend/Inven records.
- * Fully screen formatted -- Uses OSI's CRT File to adapt to any terminal.
- $\pmb{\ast}$ OS-DMS Type 10 File Structure. Extensive use of Key Files for rapid access.
- # Ideal for almost all types of businesses. Easily integrates with specialized applications.

ALPHA/OMEGA Agricultural Management System

- * Comprehensive System for all fertilizer (liquid and dry) dealers.
- Comprehensive Crop Management from Lab Analysis to Field History. Complete and attractive reporting system for customers.
- Communications package for customers who have a computer. Runs on their computer and allows downloading of data over phone line.
- * Completely integrated with Alpha/Omega Business Management System.
- ***** Includes EPA reporting for restricted chemicals and MORE.

IHS Computer Services Route 1 Box 201B Port Republic, VA 24471 (703) 249 - 4833 By: David A. Jones 9226 NW 17 St. Pompano, FL 33065

Part II of a 2 part series on the 64 character modification.

For cassette and HEXDOS users, the simplest 64 character display to start using is ROM BASIC's output routine located at \$BF2D and called from \$FF69. When this code is exe-cuted, it checks location \$FFE0 for the initial cursor position (\$65), location \$FFE1 for the width minus one of one line (\$17) and \$FFE2 to deter-mine the model (\$00). Program-ming a new PROM and changing these locations to \$40. \$3F, and \$01 respectively forces the output to be 64 characters per line. The drawback here is you must give up 24 characcompatibility with any exist-ing CIP software that depends upon the normal print routine.

One could also write a new display driver for 64 charac-ters and keep the old for 24 characters thus retaining un-modified CIP compatibility. The new driver could be stored on tape, (possible but not practical), stored in EPROM (possible and practical if you have access to a PROM program-mer), or stored on disk if you have a disk based system (the easiest way to go).

The following code is written for the second approach, EPROM, but can be adapted to either of the other by lo-cating the code in RAM. OS65D users will need to use dif-ferent addresses for the flags and scroll routine but can eliminate the boot changes. I used the top of memory for this function and modified track zero to skip over this section when checking for the amount of memory present.

To start up in the correct mode and allow for mode chang-es, the boot routine at \$FEF0 must be changed. Originally, there was a screen clear routine in this code as well as che in the monitor at \$FE00. Eliminating the first screen clear and making the second a subroutine enables us to clear the screen with a JSR and use the now unused memory loca-tions to zero the new flags and set the screen width.

The input vector, originally \$FFBA, is changed to \$F9A0 where a check is made to see if the user wants to switch to the enhanced input routine.

1000 ;1/064 (INPU: ;12-5-83 TEMP =\$FC XTMP =\$FF TMP1 =\$0201 TMP2 =\$0202 PFLU =\$0202 PFLU =\$0202 CCFL =\$0207 SCT0 =\$0207 SCT0 =\$0207 SCT0 =\$0207 USER =\$0227 USER =\$028 XNON =\$9800 CONV =\$9800 CONV =\$8956 ERT =\$0740 ; 1/064 (INPUT/OUTPUT 64 CHARACTERS) 1010 1020 1030 00FC= 00FF= 1040 0201= 1050 0202= 1060 0204= 1070 0207= 1080 020A= 1090 020B= 1100 0217= 1110 0222= 1120 9800= 1120 9800-1130 895E= 1140 D7A4= 1150 DF00= CRT =\$D7A4 KYBD =\$DF00 KBRD =\$FD00 CLEAR =\$FE0E LEGL =\$FE0E ROLA =\$FEDA INPT =\$FEED ULDOUT=\$FF6C D0CDUT=\$FF6C 1160 FD00= 1170 FE0E= 1180 FE93= 1190 FEDA= 1200 FEED= 1210 FF6C= 1220 FFEE= BASOUT=\$FFEE ACIA =\$FFBF 1230 FFBF= 1240 1250 F9A0 1260 *=\$F9A0 1270 F9A0 2C0302 INPUT 1280 F9A3 1003 1290 F9A5 4CBFFF , BIT \$0203 BPL NSET JMP ACIA 1300 1310 F9R8 2000FD NSET , JSR KBRD 1320 F9AB C902 1330 F9AD F001 CMP #\$02 BEQ NEWVEC 1348 F98F 68 RTS 1340 F3HF 60 KIS 1350 ; 1360 F380 RDCFF3 NEWVEC LDA NEWIN+1 1370 F383 RD1802 STA \$0218 1380 F385 RDD0F3 LDA NEWIN+2 1390 F385 RD1902 STA \$0219 LDA NEWIN+2 1390 F385 RD1902 STA \$0219 STA \$0218 LDA NEWIN+2 STA \$0219 LDA NEWOUT+1 1398 F989 801982 1408 F98C 8087FA 1418 F98F 801882 1428 F9C2 8088FA 1438 F9C2 8088FA 1438 F9C5 801882 1448 F9C8 8086F3 1458 F9C8 8086F3 1458 F9C8 4001F9 NEWIN 1488 F9CE 4CD1F9 NEWIN 1498 F901 200382 1508 F904 1883 1510 F906 4CBFFF 1520 STA \$021A LDA NEWOUT+2 STA \$0218 STA \$F300 LDA #\$0D RTS JMP NEWIN+3 BIT \$0203 BPL KEY JMP ACIA 1510 F9D6 4LBFFF 1520 1530 F9D9 2000FD KEY 1540 1550 F9DC C901 CTRL 1560 F9DE D00A 1570 F9ED 200EFE 1580 F9E3 A900 1599 F9E5 8D1702 JSR KBRD CTRLA CMP #\$01 BNE CTRLH JSR CLEAR LDA #0 1590 F9E5 8D1702 1600 F9E8 R90D 1610 STA CFLG LDA #\$0D 1610 1620 F9ER C908 1630 F9EC D003 1640 F9EE 4CR4FB 1650 CHP #\$08 CTRLH BNE CTRLP JMP HEXX 1660 F9F1 C910 1670 F9F3 D015 1680 F9F5 48 CTRLP CMP #\$10 BNE CTRLX PHA LDA PFLG CMP #'P 1680 F9F5 48 1690 F9F5 00402 1700 F9F9 C950 1710 F9F8 D004 1720 F9F9 D004 1720 F9FF D002 1740 FA01 A950 SET 1750 FA03 800402 STOR CMP #'P BNE SET LDA #\$20 BNE STOR LDA #'P STA PFLG 1760 FH03 800402 1760 FA06 80A407 1770 FA09 68 1780 FA08 69 1800 FA08 C918 1800 FA08 0003 1810 FA08 40098 1820 1830 FA11 C919 STA PLA CRT CMP #\$18 BNE CTRLZ JMP XNON CTRLX 1820 1830 FALL C91A 1840 FAL3 D003 1850 FAL5 4C2202 CTRLZ CMP #\$IA BNE CURCHK JMP USER 1860 1870 1870 1880 FAI8 8D0202 CURCHK STA TMP2 1890 FAI8 8A TXA 1900 FAIC 48 PHA 1910 FAID 98 TYA 1920 FAIE 48 PHA 1930 FAIF AD0202 LDA TMP2 1940 1950 FA22 C9EF . CMP #\$EF COPY 1960 FA24 D005 1970 FA26 AD0102 BNE RUBOUT 1980 FA29 D058 1990 BNE OUT2 RUBOUT CMP #\$7F 2000 FA28 C97F 2000 FH28 L97F 2010 FH2D D004 2020 FH2F H95F 2030 FH31 D050 2040 BNE FORWD BNE OUT2 , FORWD CMP #\$EE 2050 FR33 C9EE 2060 FA35 D00E 2070 FA37 203CFB BNE BACK JSR PRINT .

CHECK LOAD FLAG IF SET GET CHARACTER CONTROL B ? BACK TO CALLING ROUTINE SET NEW INPUT VECTOR SET NEW OUTPUT VECTOR SET SCREEN WIDTH TO 64 CHECK LOAD FLAG IF SET GET CHARACTER CLEAR SCREEN HOMES CURSOR HEX CONVERSION TOGGLE FLAG 29TH LINE EXTENDED MUNITOR

SAVE CHARACTER

CTRI 2

SHIFT O CTRL >

Ű,

ł

4

12 •

Listing continued



۰.

By not switching unless directed, maximum compatibility with existing software is maintained (control B being the exception and the switch command). If the switch is executed then the input flow is through \$F9D1.

When *TB* is chosen, the mode is changed to 64 and the screen editor is enabled. The new display is 28 lines of 64 characters with a 29th line used for status. A "P" shows up here when the printer is enabled, a "T" when in the terminal mode, etc. The cursor positioning portion of this code was derived from Kerry Lourash's "Cursor Control for the ClP" which appeared in the May 1981 issue of Micro. His version was for a 24 character per line display and included more features than the enclosed code. Get hold of a back issue, especially if you're interested in implementing a windowed display.

Note that the default mode is 24, and the switch is made by 7B. (POKING locations 536, 537, 538, and 539 accomplishes this task inside a program). Also. note that the code has provisions for a parallel printer interface routine for which the code after the

2080	FA3A	EEØBØ2		INC	CURS
2090	FA3D	D03F		BNE	OUTI
2100	FA3F	EE0C02		INC	CURS+1
2110	FA42	18		CLC	
2120	FA43	9039		BCC	ודטט
2130				;	
2140	FR45	C9EC	BACK	CMP	#\$EC
2150	FA47	D009		BNE	UP
2160	FA49	203CFB		JSR	PRINT
2170	FA4C	2048FB		JSR	BACKUP
2180	FH4F	18		CLC	
2190	FA50	902C		всс	OUTI
2200				1	
2210	FH52	C915	UP	CMP	#\$15
2220	FR54	D013		BNE	DUWN
2230	F856	203CFB		JSR	PRINT
2240	FH59	HDUBUZ		LDH	CURS
2250	FASC	38		SEC	
2260	FH5D	E940		SEC	#\$40
2270	FHOF	800802		SIH	OUT
2280	FH62	BUIN		805	0011
2290	FH64	LEUCUZ		DEC	CURS+1
2300	FHOT	0015		BNE	0011
2310	5 000	0004	0.0144	, CMD	****
2320	60CD	0904	DOWN	DNE	# 304 0072
2330	FACA	207055		TCD	PRINT
2750	EA70	203660		ine	CHRS
2360	FRZ3	18		ci c	CORD
2370	FA74	6940		ADC	#\$40
2.380	FAZE	SDABA2		STA	CURS
2390	F879	9003		BCC	OUTI
2400	FA7B	EE0C02		INC	CURS+1
2410				2	
2420	FA7E	2028FB	Ουτι	JSR	PUTCUR
2430	FASI	A900		LDA	#Ø
2440	FA83	4CB7FD	0UT2	JMP	\$FD87
2450				1	
2460				<i>i</i> .	
2470	F'H86	4C89FA	NEWOUT	JMP	NEWOUT+3
2480	FH89	800202		SIA	TMP2
2490	FASC	48		PHA	
2500	FABD	88		TXA	
2510	FASE	48		РНА	
2520	FR8F	98		TYH	
2530	FR90	48		FHH	
2540	FH91	HD1702		LDH	LFLG
2550	FH94	0013		ENE	FKIN OLEOD
2560	FH96	ZUUEFE		J5R	ULENK ##00
23/0	rH99	M980		LUH	# \$ØØ
2580	FH9B	800882		518	LUK3
2390	FHYE	NUU		CTO	##UU CUPC+1
2000	r HHU	300002		210	PUTCHP
2610	r HH3	202868		JSK	FUILUR

BRANCH ALWAYS

· · ·

CTRL <

CTRL U

CTRL D

NON-PRINTING CHARACTER TO BASIC

TEMP SAVE

CLEAR SCREEN

HUME CURSOR

ú,

Continued on page 15

WATCH THIS SPACE IN APRIL FOR NEW PRODUCT ANNOUNCEMENTS

inc.

p.o. box 7276 denver, co 80207 (303) 428-0222

D&N MICRO PRODUCTS, INC.

TERMS \$3.00 shipping, Foreign orders add 15%, Indiana residents add 5% sales tax.

COMPUTER

MICRO-80 COMPUTER Z-80A CPU with 4Mhz clock and CP/M 2.2 operating system. 64K low power static memory. Centronics parallel printer port. 3 serial ports, 4" cooling fan. Two 8" single or double sided floppy disk drives. IBM single density 3740 format for 243K or storage, double density format for 604K of storage. Double sided drives allow 1.2 meg on each drive. Satin finish extruded aluminum with vinyl woodgrain decorative finish. 8 slot backplane, 48 pin buss compatible with OSI boards.

MODEL 80-1200	\$2995
2 8" Single sided drives	
MODEL 80-2400	\$3495

2 8" Double sided drives

MICRO-65 COMPUTER

6502 CPU with 2Mhz clock and DOS-65 operating system. 48K of low power static memory. 2 serial ports and 1 Centronics parallel port: 2 8" single or double sided drives. Satin finish extruded aluminum with vinyl woodgrain finish. 8 slot backplane, 48 pin buss compatible with OSI. Will run OSI 65D and 65U software.

MODEL 65-1	\$2995
2 8" Single sided drives	

MODEL 65-2 \$3495 2 8" Double sided drives

BP-5808 Slot Backplane \$ 47 OSI 48 pin Buss compatible

MEM-CM9 MEMORY/ FLOPPY CONTROLLER

24K memory/floppy controller card
uses 2114 memory chips, 18K and
1 16K partition, Supports OSI type
disk interface
24MEM·CM9\$325
16MEM-CM9\$260
8MEM-CM9\$180
BAREMEM-CM9\$ 50
Controller on assembled unit
add\$90
PIO 1600 Para IO aard 6 50

Supports 8K of memory, 2 16 bit parallel ports, 5 serial ports, with manual and Molex connectors.

PRINTERS

. \$409
. \$895
.\$1150

- 8510AP Prowriter, parallel ... \$419
- 120 cps, correspondence quality 8510APD Prowriter, serial \$585
- F10-40PU Starwriter, parallel \$1319 Letter quality daisy wheel
- F10-40RU Starwriter, serial . . \$1319
- F10-55PU Printmaster\$1610 parallel, Letter quality daisy wheel
- F10-55RU Printmaster, serial \$1610 DISK DRIVES AND CABLES
- 8" Shugart SA801\$385 single sided
- 8" Shugart SA851 \$585 double sided
- FLC-66 ft cable from D&N\$69 or OSI disk controller to 8" drive
- 5¹/₄" MPI B51 disk drive with . . \$450 cable, power supply and
- cabinet. Specify computer type. FLC-514 cable for connection \$75 to 514 drive and D&N or OSI controller with data separator
- controller, with data separator and disk switch. Specify computer type

HARDWARE OSI COMPATIBLE

IO-CA10X Serial Printer Port . . \$125 Specify Device #3 or #8

IO-CA9 Parallel Printer Port \$150 CMOS-MEM

64K CMOS static memory board, uses 6116 chips, 3 16K, 1 8K and 2 4K blocks, Partitionable for multiuser, OSI type disk controller, 2 IO mapped serial ports for use with D&N-80 CPU. Ideal way to upgrade from cassette to disk.

64K CMOS-MEM	\$490
48K CMOS-MEM	\$390
24K CMOS-MEM	\$250
16K CMOS-MEM	\$200
Controller	add . \$ 90
210 manned serial ports	add \$125

- on assembled memory board **280-IO** 2 IO mapped serial \$160
- ports for use with D&N-80 CPU card
- FL470 Disk Controller \$155 Specify 51/4 or 8" drive



3702 N. Wells St.

Fort Wayne, Ind. 46808 (219) 484-6414

STANDARD CP/M FOR OSI

D&N-80 CPU CARD

The D&N-80 CPU allows the owner of an OSI static memory computer to convert to Industrial Standard IBM 3740 single density disk format and CP/M operating system. Double density disk operation is also supported for 608K of storage on an 8" diskette. When used with a 5¼" disk system 200K of storage is provided. Includes parallel printer and real time clock. Also available for polled keyboard and video systems. Compatible with C2, C3, C4 and 200 series OSI computers.

D&N-80- P · · · · · · · · · · · ·	\$ 349
CP/ M 2.2 ·····	\$150

64K CMOS-MEM with D&N-80 CPU card \$450

HARD DISK DRIVER \$140 Allows D&N-80 CPU board to control OSI 40 or 80 meg hard disk unit. Will not destroy OSI files. Will also allow for a true 56K CP/M system. Specify 40 or 80 meg drive.

BUSS TRANSFER \$135 Allows for D&N-80 and OSI CPU to be in the computer at the same time. Toggle switch provides for alternate CPU operation.

DISK TRANSFER \$100 Utility program to transfer OSI CP/M format disk to IBM 3740 single density format. Will also transfer IBM to OSI format.

SYSTEM HARDWARE REQUIREMENTS

D&N-80 CPU, D&N FL470 or OSI 470 controller, 48K memory at 0000-BFFF, 4K memory at D000-DFFF, two disk drive cables. FORMATTRANSFER \$15 You supply software on 8" diskette D&N will transfer OSI CP/M format to IBM 3740 CP/M format. Can also transfer IBM 3740 CP/M format to OSI CP/M format. Original diskette returned. branch is not shown. My parallel printer interface is unique to my surplus printer so the particular driver is not of general interest but the hook is. You could use the check to branch to any other routine for your unique system. In addition $\uparrow X$ causes a jump to the extended monitor relocation and residing in EPROM at \$9800, again something unique to my system but can be implemented on yours by installing an EPROM there. Refer to my article in Feb 1983 Micro.

Control A clears the screen (POKING 535,0 clears the screen from inside a program the next time an output statement is executed. Control H branches to a hexadecimal to decimal conversion routine. see my article in PEEK(65) June 1982 for comments on the code. Control Z will cause a branch to a user routine located at \$0222 (546). Control Q, XON, and control S. XOFF, are also honored, see PEEK(65) Dec. 1980 for more information here.

The source code is divided into 2 sections, thus the separate declarations of labels and variables. Section 1 pertains to the new input/output routines and section 2 to the new boot mechanism.

The jumps at newin and newout are used to allow assembly without defining absolute addresses for the beginning of each of these routines.

To use the screen editor the cursor is positioned by using the control keys. Control U moves the cursor up, control D moves it down and control < and > move it left and right respectively. Control / reads into the input buffer any character the cursor passes over. Either shift O or rubout will erase a character already entered and will move the cursor back one space.

Referring to those back issues again, you'll note that the ClP does not use any of the code from \$F800 to \$FBFF. I previously suggested that a cassette load routine be located at \$F800-F89F (March 81) and a cassette save routine be located at \$F8A0-F99F. This leaves \$F9A0-FBFF free for our display driver. Not all of this space is required for this implementation so you can add some of your own enhancements. 2620 FRA6 801702 2630 STA CFLG 2630 2640 FAA9 AD0402 PRIN 2650 FAAC C950 2660 FAAE D006 LDA PFLG CMP #'P BNE CRET LDR TMP2 2670 F860 800202 2680 FAB3 4CB6FA 2690 JMP CRET 2690 2700 FH86 HD0202 CRET 2710 FH89 C900 2720 FH8B D014 2730 FH8B N014 2730 FH8F 8D0102 2740 FH8F 8D0102 2750 FH82 2030FB LOA TMP2 CMP #\$00 BNE ERASE LDA #\$20 STA THEI JSR ERINT 2750 FACS 2000F6 2760 FACS AD0802 2770 FAC8 18 2780 FAC9 6940 2790 FAC8 8D0802 LOA CURS CLC AUC #\$40 STA CURS JMP NEWL 2800 FACE 4CFCFA 2810 CNP #\$5F 2820 FAD1 C95F ERASE 2830 FAD3 DØ13 2840 FAD5 C60E 2850 FAD7 A920 BNE CCHAR DEC \$0E LDA #\$20 2850 FHD7 H920 2860 FAD9 8D0102 2870 FADC 203CFB 2880 FADE 2048FB 2890 FAE2 2033FB 2900 FAE5 4C1DFB LDH #\$20 STA TMPI JSR PRINT JSR BACKUP JSR PUTI JMP EXIT 2910 2920 FRES C920 CNP #\$20 CCHAR 2920 FRE8 C920 2930 FRE8 3031 2940 FREC 8D0182 2950 FREF 203CF8 2960 FRF5 203CF8 2970 FRF5 AD0802 2980 FRF8 293F 2990 FRFA D01E 3000 BMI EXIT STA TMPI JSR PRINT INC CURS LETTER LDA CURS AND #\$3F BNE OUT 3000 3000 3010 FAFC AD0802 NEWL 3020 FAFF 29C0 3030 FB01 8D0802 3040 FB04 C980 3050 FB04 C980 LDA CURS AND #\$C0 STA CURS ENE BLOCK LDA CURS+1 CMP ##D7 3060 FB08 AD0C02 3070 FB08 C9D7 3080 FB0D D008 BNE OUT JSR SCRL 3090 FB0F 2054FB 3100 3110 FB12 AD0802 BLOCK 3120 FB15 D003 3130 FB17 EE0C02 LDA CURS BNE OUT INC CURS+1 3140 3150 FBIA 2028FB OUT JSR PUTCUR 3150 FBIH 20. 3160 3170 FBID 68 3180 FBIE 68 3190 FBIF 68 3200 FB20 A PLA EXIT TAY PLA TRX 3210 FB21 68 3220 FB22 2089FB 3230 FB25 4C6CFF 3240 PLA JSR XONF JMP OLDOUT 3240 3250 3260 FB28 A9AD 3270 FB2A 8D0A02 3280 FB2D 200A02 3290 FB30 8D0102 PUTCUR LDA #\$AD STA CURS-I JSR CURS-I STA TMPI 3300 FB33 A980 3310 FB35 8D0A02 3320 FB38 A98B STA 11F1 LDA #\$8D STA CURS-1 LDA #\$88 PUTI 3330 FB3A D008 BNE PRINTI 3340 3340 3350 FB3C A98D 3360 FB3E 8D0A02 3370 FB41 AD0102 3380 FB44 200A02 3390 FB47 60 ĹDA **#\$8**D PRINT STA CURS-I LDA TMPI JSR CURS-I PRINTI RTS 3400 3410 FB48 AD0802 BACKUP LDA CURS 3420 FB4B D003 3430 FB4D CE0C02 3440 FB50 CE0B02 BNE BKI DEC CURS+I DEC CURS BKI 3450 F853 60 RTS 3460 3470 3470 3480 FB54 A007 3490 FB56 B962FB 3500 FB56 B962FB 3510 FB56 B9 3520 FB56 B8 3520 FB55 D08F7 3530 FB57 A207 LDY #7 SCRL LDA CODE-1,Y STA SCFM-1,Y DEY BNE MOVE MOVE LDX #7 BNE LINE 3540 FB61 D007 3550 3560 FB63 B9C0D0 CODE 3570 FB66 9980D0 3580 FB69 60 LDA \$DOCO,Y STA \$0080.4 RTS 3590 3600 F86A 200702 LINE JSR SCFM 3610 FB6D C8 3620 FB6E D0FA INY BNE LINE INC SCFN+2 INC SCTO+2 DEX BNE LINE 3630 FB70 EE0902 3640 FB73 EE0C02 3650 FB76 CA 3660 FB77 D0F1 3670 FB79 A920 LDA #\$20 LDY #\$3F 3680 FB78 A03F

CLEAR FLAG CHECK FLAG IF NOT SET WOULD JSR TO PRINTER ROUTINE HERE ERASE CURSOR IF CR SHIFT O ERASE CHAR UNDER CURSOR ERASE CURSOR DISPLAY CURSOR DUN'T FRINT CONTROL CHARACTERS START NEW LINE LDH STA CURSOR CHAR STA CODE TO BE RELOCATED TO RAM AT \$0207 SKIP CODE TO BE RELOCATED START FROM HERE -AND MOVE IT TO HERE

Listing continued

							•
3690 FB7D 9940D7	BLANK	STA \$D740.Y	BLANK LAST LINE	4460 FE27 68		PLA	
3700 6880 88		DEV		4470 FE28 60		RTS	
3710 EB81 10EA		DDI DI DNV		4480 FE29 FA		NOP	
3700 5007 0040				4490		NC.	
3720 F883 H940		LDH #\$40		1720		,	
3730 F885 800802		STA CURS		4300 FEF0		*=\$FEF0	I/O VECTORS
3740 FB88 60		RTS		4510		;	
3750		1		4520 FEFA AAF9		WORD INPUT	
7760 6000 40	VANC	040		4570 EEE2 COFE		HORD THEOT	
3730 7803 40	NUMP	rnn		4500 1212 0011		MORD DUIPUT	
3//0 FB8H H9F6		LDH #\$F0	SELECT RUWS I & 3	4340 FEF4 98FF		.WORD CIRLÇ	
3780 FBSC 8D00DF		STA KYBD		4350 FEF6 88FF		.WORD LOAD	
3790 FB8F A9C0		LDA #\$CH	SELECT COLUMNS 7 & 6	4560 FEF8 96FF		.WORD SAVE	
3800 ERGI 2000DE		BIT VUED	STOR2 (CONTROL S)	4570 FFFA 3001		HOPD NMT	
7910 5094 0000		DUE CONT	IC NOT CONTINUE	4580 FEEC DOFF		LIDOD UM	
3010 FB34 D00C		DNE CONT	IF NULL CUNTINUE	4600 FEFE 0001		. MORD VII	
3820 FB96 H9FC	<i>мн</i> іт	LDH #\$FC	RUWS U & I	4050 FEFE C001		.WORD IRQ	
3830 FB98 8D00DF		STA KYBD		4600		3	
3840 FB98 A9C0		LDH #\$C0	COLUMNS 7 & 6	4610 FF00 D8 B	юот	CLD	
3850 EB9D 2000DE		BIT KURD	RECIME? (CONTROL ON	4620 FF01 A2FE		INX #SEE	SET STACK POINTED
7040 5000 0054				4630 FE03 98			SCI SINCK PUTHIER
3060 FBR0 20F4		BNE WHII	IF NULL KEEP LUUPING	A640 5504 0000			
3870 FBH2 68	CONT	PLH		4680 Fred Hoen		LUY #10	
3880 FBA3 60		RTS		4630 FF06 BYEFFE P	'SET	LDA \$FEEF.Y	PRESET I/O VECTORS
3890		;		4660 FF09 991702		STA \$0217,Y	
ЗЭЛИ ЕВА4 8924	HEXX	108 #1\$		4670 FF0C 88		DEY	
7910 ERGE 20FFFF		TEP PACOUT		4680 FEAD D0E7		BNE PSET	
3000 5000 0004		13K 013001		4690 EE0E 2006EC			
3320 1013 1204		LUX #4		4700		JOK INTI	INITIALIZE ACIA
3930 FBHB 20EDFE	INNN	JSR INPT				;	
3940 FBAE 20EEFF		JSR BASOUT		4710 FF12 H014		LDY #\$14	CLEAR ALL FLAGS
3950 FBB1 2093FE		JSR LEGL		4720 FF14 A900		LDA #0	
3960 EBR4 86FF		STY YTMP		4730 FF16 990302 2	FRA	STA \$0203.V	
3030 1004 0011				4740 FEI9 89	LING	500 0000000	
3770 FBB6 H200		LDX #0		4750 6610 1000		DET	
3980 FBB8 20DHFE		JSR RULH		4700 FFIN IOFN		BPL ZERU	
<i>3990 FBBB A6FF</i>		LDX XTMP		4760		3	
4000 FBBD CA		DEX		4770 FFIC ADE 0FF		LDA \$FFE0	
4010 FBBE DOEB		RNF INNN		4780 FFIF 800002		STA \$0200	PRESET ORIGINAL
4020 EBC0 86EC		IDX TEMP					CURSOR LOCATION
4070 EPC2 0550		I DO TEMPAI		4700 5500 000555			CONSOR LOCHITON
4040 5004 205500				4790 FF22 200EFE		JSK LLEHK	ULEHR SUREEN
4040 FDL4 203203		JOR CONV		4800 FF25 8D00F7		STA \$F700	SET NARROW SCREEN
4050 FBC7 H900		LDH #\$UD		4810 FF28 EA		NOP	
4060 FBC9 20EEFF		JSR BASOUT		4820 FF29 EA		NOP	
4070 FBCC A90A		LDA #\$0A		4830 FF2A EA		NOP	
4080 FBCE 20EEFF		JSR BASOUT		4840 EE28 EA		NITP	
4090 FBD1 60		RTS		1050 FF2C F0		NOD	
4100				4000 1120 20		NOP	
4110		·		4000 FF20 ER		NUP	
4100 0000-		,		4870 FF2E EH		NUP	
4120 0000=		W5 HR =\$0000		4880 FF2F EA		NOP	
4130 0130=		NMI =\$0130		4890 FF30 EA		NOP	
4140 0100=		IRQ =\$0/C0		4900 FF31 ER		NOP	
4150 BD11=		CSTART=\$BD11		4910 FF32 EA		NUP	
4160 FC00=		DSKIN =\$FC00		4920 FF33 A000		LDY #0	
4170 FCA6=		INIT =\$FCA6		4930 FF35 895FFF 8	RMSG	IDA TEXT.Y	
4180 FE00=		VM =SEFAA		4940 FE38 E006		DEN INDEN	
4190 FF69=		DUTPUT=\$FE69		1950 FE70 20FEFF		ICO DOCOUT	
4288 FE88=		1100 =45598		4060 CE20 00			
4210 5506-		CONC -#CEOC		4300 FF30 C0		INY	
4220 5520-				4970 FF3E DOFS		BNE BASG	
7220 FF 38=		CIRLL #\$FF98		4980 FF40 2080F9 I	NPCH	JSR INPUT	
4230		· · · · · ·		4990 FF43 C940		CMP #'M	
4240 FE00		*=\$FE00		5000 FF45 D003		BNE WARM	
4250		;		5010 FF47 4C00FE		JMP VM	
4260 FE00 A228	VM	LDX #\$28		5020 FE48 C957 W	IARM	CNP #1W	
4270 FE02 98		TXS		5030 FF4C D003		ENE COLD	
4200 5507 00		<u> </u>		5040 EE4E 400000		INP USTART	
4200 FE03 DO				5050 FEST COA2 C	010	CMD #/C	
4290 FE04 200EFE		JSK LLEHK		5000 FF31 6343 6	.060		
4300 FEU7 84FF		STY SFF		3060 FF33 1003		BNE DIDK	
4310 FE09 84FB		STY \$FB	CLEAR ACIA FLAG	SUTU FFSS 4CTIBD		JHP CSIMKI	
4320 FE08 4C43FE		JNP \$FE43		5080 FF58 C944 D	215K	CMP #^D	
4330 FE0E 48	CLEAR	PHA	CLEAR SCREEN	5090 FF5A DUH4 *		BNE BOOT	
4340 FEOF A9DO		LDA #\$00		5100 FF5C 4C00FC		JMP DSKIN	
4350 FEII 85FF		STA SEF		5110		;	
4360 FE13 ANNA		IDY #A		5120 FF5F 44 1	TEXT	BYTE 'D/C/W/M	21
4370 FE15 84FF		STY SEE		5120 FE60 2F			
4700 CE17 0000	<u></u>	100 4400		5120 FECI 47			
4300 FEIT 1920	00	CTO ##20		5120 FF01 40 5100 FFC0 0F			
4390 FEIS SIFE	ιď	SIN (APE)Y		3120 FF62 2F			
4400 FEIB E6FE		INC SFE		5120 FF63 57			
4410 FEID DOFA		BNE C2		5120 FF64 2F			
4420 FEIF E6FF		INC \$FF		5120 FF65 4D			
4430 FE21 ASFF		LDA SFF		5120 FF66 20			
4440 FE23 C9D8		CMP #\$08		5120 FF67 3F			
4450 FE25 DOFA		BNE CI		5130 FF68 00		RYTE \$0	
1100 1 220 2010		erra VI		5/40			

THE NEW CHALLENGER PERSONAL COMPUTER

By: Bruce Showalter 857 Cedar Abilene, TX 79601

The Superboard II/ClP generated no small amount of enthusiasm when it was introduced in 1979. Even today it has a loyal following. Its big brother, the C4P, is an even more praiseworthy machine.

But times change. The competition learned by the Cl's example and went hard after the low-end user market. The result was that the Challenger's market was choked off by the flood of ZX-81s. TRS-80s, VIC-20s and Ataris.

The subsequent generations of OSI management have elected to continue pursuing the small business market. They evidently don't want to be burned at the personal market level again. But they don't have to, they can learn from the competition, just as the competition learned from them. Let's proceed, therefore, to propose a future version of the Challenger personal computer, the C-65. Almost every owner of the Cls has modified the hardware. This points to obvious inadequacies which should have been corrected by the designers early on. Their attempt to recover the fumble is indicated by the revised edition of the S.11/ClP. So, let's build in the most desired features from the outset.

One of the most popular features of the Challengers was their changeability. The hardware hackers loved 'em. With that in mind, let's take a page from the Apple hardware design and build with multiple

plug-in boards. We can use either the KIM-44 boards or the OSI-48 boards. If we choose the latter, the size should be about half the original. The only exception would be the video circuit, about which more later. The purpose of using smaller boards is to divide the computer into modules. Each module can be owner-customized or replaced entirely. Implicit details include sockets for each and every IC and use of LS-TTL throughout the system.

The first module is the CPU board. It consists of the microprocessor and suitable line buffers. There is also a clock circuit which can be easily modified or by-passed altogether. This design permits the owner to substitute another CPU of his own choosing. The CPU module is configured for a 1-mhz clock, but there is also a 2-mhz signal available for those who desire it. The WAIT circuit divides the clock speed in half. The CPU Reset line is connected to an R-C circuit which produces a Power -On-Reset pulse.

The MONITOR + BASIC module comes next. '2716 EPROMS are used to facilitate other 'operating systems and languages. We could use 2732 or 2764 EPROMS, but the 2716 is already well established. By putting firmware on a board by itself, the owner can change language and OS by merely switching boards.

Next, we have the KEYBOARD ADDRESS module. This consists of merely the decoders and Next, buffers. A cable joins this board to the keyboard itself. This module will most likely remain unchanged, unless a new location in the memory map is desired. As with the previous Challengers, we use a polled keyboard. However, significant improvements are made. A hexadecimal numeric pad is added. The RUBOUT key is relabeled CLS to incorporate that function. Since REPEAT is programmed into the OS, we relabel that key for BACK-SPACE. And we replace the LINE FEED key with CAPS LOCK. ESCAPE becomes CANCEL (Shift P). More will be said about the keyboard when we discuss the OS.

The CASSETTE + RS-232 I/O is built on the next module. Not much is different from the Cl circuitry, except that the RS-232 interface is installed. Note that the ACIA clock is derived from an on board source, rather than by dividing down a master clock signal. There are provisions for changing the ACIA clock from 4800 hz to 9600 hz. Preferably, a high speed cassette I/O (such as a VIC Rabbit) could be installed in place of the existing circuit. This would probably require a entirely new module.

Choice of the VIDEO module may be left to the purchaser, since tastes vary on this subject. Some prefer an 80column display, while others are satisfied with 64 or 48. Some prefer color and hi-res grahics. Perhaps the best bet would be to offer a 540 video board (less keyboard/analog inputs) and have second source provide alternates vendors (such as the Orion SEB). The buyer could elect to omit the 540 module from the C-65 package at the time of purchase if he didn't want that version.

RAM modules would be fairly standard. I'd recommend CMOS 2K x 8 chips, with each module holding 16K.

The DISK module would be sold like before, either as an expansion option or in a full fledged disk system. As with the video modules, secondsource vendors could provide alternate configurations.

The foregoing implies a burden on OSI to offer hardware documentation and licensing to other vendors. This omission other vendors. This omission in the past, I believe, desti-ned OSI's failure in the per-Softsonal computer market. ware and alternate hardware from second-source vendors have contributed immensly to the success of Apple, Commodore, TRS, Atari, and IBM. Texas Instruments' recent failure in this market lends even more support to this argument.

Before we leave our discussion of hardware, we should take a look at the bus. Neither the KIM-44 nor the OSI-48 busses are completely satisfactory to me. A table gives the revised C-65 bus definitions I propose. No attempt is made to allow for 80-type or 68-type processor lines. We'll leave CP/M to OSI's line of small business machines.

Now it's time to discuss the Operating System. For the machine-code hackers, a versatile MONITOR is a must with an assembler/editor. The keyboard format, we touched on earlier. To continue, we interpret either Left SHIFT, Right SHIFT, or SHIFT LOCK exactly the same: all character keys input their upper case symbol. Otherwise, the case depends

DISK DRIVE RECONDITIONING

WINCHESTER DRIVES

FLAT RATE CLEAN ROOM SERVICE. (parts & labor included) Shugart SA1002 5meg \$390.00 Shugart SA1004 10meg \$450.00 FLOPPY DRIVE FLAT RATES Parts & Labor Included (Missing parts extra) \$170.00 8" Double Sided Siemens 8" Single Sided Siemens \$150.00 8" Double Sided Remex \$225.00 8" Single Sided Shugart \$190.00 8" Double Sided Shugart \$250.00 \$120.00 51/4 M.P.I. Single Sided \$150.00 51/4 M.P.I. Double Sided ONE WEEK TURN AROUND TYPICAL You'll be notified of --1. The date we received your drive. Any delays & estimated completion date. Date drive was shipped from our plant. 3. Repairs performed on your drive. Parts used (#and description). 5 90 day warranty -Write or call for detailed brochure We sell emergency parts Phone: (417) 485-2501 FESSENDEN COMPUTERS 116 N. 3RD STREET OZARK, MO 65721 Eca 37 Introducing WORD PROCESSOR OS-65U 1.42 Floppy/Hard Disk Level 1 or Level 3 and DENVER BOARDS **#INTERFACED TO OS-DMS FILES #AUTOMATIC WRAP AROUND COMPLETE EDITING CAPABILITIES** FULL CURSOR CONTROL INSERT & DELETE TEXT SEARCH/SEARCH & REPLACE ***USER FRIENDLY MANUAL ***AND MUCH MORE IHS COMPUTER SERVICES Route 1 Box 201B Port Republic, VA 24471 (703) 249-4833 Sà \$195.00

upon whether CAPS LOCK is depressed. This key shifts only the alpha characters into upper case. All other characters are lower case. None of the following keys are affected by case: RETURN, SPACE, BACKSPACE, CONTROL, CANCEL. The RETURN function scrolls the display, but doesn't send a LINE FEED command to the ACIA.

BASIC-IN-ROM could stand some improvements. Right away, we

fix the Garbage Collector. CLS is a must. Next, we incorporate a GET or INKEY state-ment. Another feature I T consider especially useful is the handling of NULLS. recommend that the number NULLS be stored in RAM. Τ of So when we boot up, NULL = 10. But with a POKE, NULL could be set to anything from zero to 255. One more useful statement is PRINT AT. There are others which I will leave for you, the readers, to recommend.

KIM-44 TO C-65 BUS CONVERSION

PIN# KIM-44 MODIFICATION C-65

1	GND		GND
2	SYNC	DISCONNECTED	WAT M
د ۸	TPO	REDEFINED	TRO
5	_15		-15
6	NMT		NMT
7	RST		RST
8	D7	•	D7
9	D6		D6
10	D5		D5
11	D4		D4
12	D3	•	D3
13	D2	•	D2
14	Dl		Dl
15	DØ	· ·	DØ
16	BDSEL	DISCONNECTED	
17	+15		+15
18	DMA	REDEFINED E	XCLK
19	+8	DISCONNECTED	
20	+8	DISCONNECTED	
21	+5 CND	•	CND
22 N	GND	· · ·	GND
R	AND		AØ
ĉ	Al		Al
Ď	A2		A2
Ē	·A3		A3 .
F	A4	· ·	A4
G/H	A5		A5
J	A6	•	A6 -
K	A7 ·	·.	A7 ·
\mathbf{L}	8A		A8 ·
М	A9		A9
N	ALU		ALU
P	ALL		A11 N12
R	A12		A12 A13
د ۳	ALS AIA		
1 11	A15		A15
v	Ø2		Ø2
w	R/W		R/W
x	Ø2	DISCONNECTED	
Ŷ	+5		+5
7	GND		GND

OSI-48 TO C-65 BUS CONVERSION

PIN# OSI-48 MODIFICATION C-65

1 2 3	WAIT NMI IRO		WAIT NMI IRO
4	DD	DISCONNECTED	
5	DØ		DØ
6	Dl		Dl
7	D2		D2
8	D3		D3
9	D4		D4
10	D5		D5
11	D6		D6

12	D7		D7
13	D8	DISCONNECTED	
14	D9	DISCONNECTED	
15	DlØ	DISCONNECTED	
16	<u>D1</u> 1	DISCONNECTED	
17	RST		RST
18		REDEFINED	EXCLK
19	A19	DISCONNECTED	
20	A18	DISCONNECTED	
21	A16	DISCONNECTED	
22	A17	DISCONNECTED	
23	+12	INCREASED	+15
24	-9	INCREASED	-15
25	+5		+5
26	+5		+5
27	GND		GND
28	GND		GND
29	A6		A6
30	A7		A7
31	A5		A5
32	8A		A8
33	A9		A9
34	Al		Al
35	A2		A2
36	A3		A3
37	A4		A4
38	AØ		AØ
39	Ø2_		02_
40	R/W		R/W
41	VMA	TIED HIGH	+5
42	VMA-	02 REDEFINED	02
43	AlØ		A10
44	All		A11
45	A12		A12
46	A13		A1 3
47	A14		Al4
48	A15		A15

READER PROFILE

ED:

I think it's time I wrote and supported this super journal.

My contribution is a description of a uniquely expanded ClP which I bought in January 79 for \$330. Gradually, and I do mean gradually, I played with hanging stuff onto the expansion port.

Now, as depicted in the fig-ure, I've totally designed, engineered, and built a 44 pin bus card rack system, and it works!

Technically:

- the adapter / driver board connects the 40 pin DIP port to a 40 pin IDC header. This card generates signal DD and decodes slots in the rack.

- a digital I/O card uses two 6522s to' do general purpose stuff.

- a complex sound generator card occupies its own slot where a 6821 drives into a GI AY-3-8910. An ASCII keyboard will also be connected to the I/O Port of the AY-3-8910.

- the magic of the system is

an 80 column card based on the 6545 CRTC. I've dedicated 16K of CPU mapped memory to video. Scrolling is accomplished from a 20 key keypad hung on the digital I/O card. (Not used by OS 65D, though.)

- the floppy disk controller is a copy of the 470 design.

The system:

- runs C4 / C8 software has 2 8" Siemens SS / SD
- drives capability of 48 K memory,
- accomplished by disabling BASIC.
- printer OKIDATA 82A
- front panel switches: * 300 or 1200 baud

 - * BASIC in ROM on-off
 - * CPU clock 1 or 2 Mhz * Functional C4 or C1 enable
- switch - US Robotics "Password" modem

The superboard has been totally modified to provide for more address decoding. I installed the video mod from Progressive Computing and I enabled BASIC device #4 (parallel printer port).

I'd like to say that this has been a lot of fun, sweat, and tears. I've had to rediscover the meaning of much engineering design work. The more remarkable thing is that I've never had a formal course in electronics.

Coming shortly, I'll be adding a card designed by a Rockwell engineer. The card allows any 6502 machine to run CP/M. I will report my progress on that.



(GOLD UPGRADE)

ONE WEEK TURN-A-ROUND TYPICAL

SOKOL ELECTRONICS, INC. 474 N. POTOMAC ST. HAGERSTOWN, MD 2174Ø

(301) 791-2562



I need to acknowledge my beautiful wife who has tolerated the money, time, and boring computer conversations. She can be considered a computer widow, but I really appreciate her.

LETTERS

ED:

Our firm runs a three user, ten megabyte Denver Board OSI (a converted C2) with OS65U V.1.43 operating system.

We have been trying to use WP 3.3 word processor, but unfor-

I know y'all would like to know how much I've spent. Well, not including the printer or modem, I think it's been under \$1600.

Gene W. Anderson Sunnymead, CA 92388

tunately the computer "freezes" immediately after the response to the question:

"Do you want Device 8 to be set up for a serial printer?"

We have entered two new lines numbers 211 and 212 as stated in the September issue of PEEK, but this does not make any difference! Could you, or one of your readers, please help?

John S. Spry Wellington, Australia

John:

See "Bug Fix" further on this issue.

Peek(65) Staff

* * * * *

ED:

This is the first program I have submitted to PEEK(65). I felt California should be better represented. I have a C2-4PMF, but the program should certainly run on a C4 and I have been told it will run on a C1, but I cannot confirm this.

The program, re-written for came from Projects OSI. in Machine Intelligence by D. run, Heiserman. When first you are asked for two inputs The *, 42, in ASCII code. makes a nice creature (yes, Т have withstood the impulse to title the program Creature Features). An entry of 32 for the trail will give you an in-visible trail, while 161 will give a graphic symbol which makes a nice trail. The border is then drawn and the area within the borders is filled with 64 killable obstacles, randomly selected and placed. The creature or Alpha then zips around the screen at random, encountering obstacles and the border itself. Upon an encounter, there is а 50% chance the Alpha will effect a kill and continue on in the same direction. In the case of the border, the creature

NEED OSI COMPATIBLE PRODUCTS?

We sell SCIENCE AZTEC'S full line of OSI compatible PC Boards & Systems.

- 8550 Communications Interface (replaces 550)
- 8590 HD Controller (replaces 590/525)
- 8592 HD Interface
- 8588 Active Terminator
- 919 9 Slot back plane with active/ passive terminators
- 68000 Boards to be announced shortly

- BD 14" 80 MB Hard Disk, with controller, interface & cable
- 8470 Floppy Disk Controller (replaces 470)
- 8472 8" or 5¼" Floppy Controller -IBM Format
- 8516 GT 3.3 Mhz CPU, 64k Static
 DMA
- 8528 Up to 4 partition GT memory, Static RAM

Prices available for Disk Subsystems as well as complete Computer Systems

BECTERM

12 Trans-Canada West Levis, Quebec G6V-4Z2 418-837-5894

can try to take a bite out of the border - it can put a dent in the border, graphic symbol 153, but can never completely After the program escape. runs for awhile, you will find most of the obstacles gone and the borders badly chewed. Tf you use a visible trail and the Alpha becomes cornered by its own trail, it becomes rather nasty and chews its way to relative freedom. And the Alpha has no qualms about tur-ning cannibal if it encounters a like creature. If you use a trail code number between 128 and 154, the trail becomes impenetrable and the Alpha soon becomes hopelessly entrapped by its own trail. In all In all will run cases, the program until it is stopped with a CTRL'C'.

Possible modifications include adding color to the program and varying the number of ob-stacles. The subroutine in line 1040 can be used to display scoring how many times is the border hit, how "good", how many moves are "good", how many kills, etc.. For persons more interested in the theory of machine intelligence and psychology, I would refer them to Mr. Heiserman's book Tab #1391.

If there is much interest in

this program, I have a number of other programs available. The Beta programs display а learning response and could apply to the programming of а robot.

I enjoy PEEK(65) very much and am looking forward to the articles on I/O as this area is giving me some problems, I think more so since I have an older C2. I would like to connect a modem and I also have an ADM-2 display terminal At I would like to use. one time there was a local OSI User's Group, but it has fal-len on hard times.

3Ø REM**KILLER ALPHA DEMO, OSI V3.3**40 PRINT1 (28) 5Ø PRINT"STRIKE 'R. SHIFT' TO START ... IF PEEK(57100) <>3 THEN 60 N=RND(8):GOTO 6Ø 70 INPUT "SELECT A CREATURE CODE";CT 8Ø PRINT: INPUT "SELECT A TRAIL CODE"; TT PRINTI (28) : PRINT"YOUR CREATURE 90 LOOKS LIKE THIS--";CHR\$(CT): PRINT PRINT "ITS TRAIL LOOKS LIKE THIS--"; CHR\$(TT): PRINT 95 INPUT "IS THAT WHAT YOU WANT 100 (Y/N) ";S\$ IF SSC>"Y" THEN 70 105 REM**ALPHA MAINLINE, OSI V3.3** 107 110 PRINT! (28) : GOSUB 1000

- 112 DS="KILLER ALPHA DEMO":D=55188: GOSUB 1040
- CP=54328+INT(5*RND(8))-2+64* 115 INT(5*RND(8))-2:POKE CP, CT
- 120 FOR N=Ø TO 63 TP=53400+INT(1500*RND(8)): 125
- IF PEEK(TP)<>32 THEN 125
- IF TP>55039 THEN 125 POKE TP, INT(33*RND(8))+33: 126
- 130 NEXT N
- 135 CI=INT(5*RND(8))-2
- CJ=INT(5*RND(8))-2:IF CI=0 136 AND CJ=0 THEN 135
- 140 GOSUB 2000 145 IF NOT(CX=32 AND CY=32)
- THEN 165 150
- POKE CP, TT 155 CP=NP:POKE CP, CT
- GOTO 140 160
- KC=INT(2*RND(8)):IF KC=0 165 THEN 135
- 170 IF PEEK(NP)>=128 AND PEEK(NP) <=154 THEN FOKE NP, 153: GOTO 140
- POKE NP, 32:GOTO 140
- 1000 REM**BORDER, OSI V3.3**
- 1005 FØ=53376
- 1006 F1=53439 1007 F2=54976
- 1008 F3=55039 40 1010 FOR N=FO TO F1
- 1011 POKE N,128
- 1012 NEXT N
- 1015 FOR N=F2 TO F3
- 1016 POKE N,135
- 1017 NEXT N
- 1020 FOR N=F0 TO F2 STEP 64
- 1021 POKE N, 149:NEXT 1025 FOR N=F1 TO F3 STEP 64:
- POKE N, 149:NEXT 1030 POKE FØ,128:POKE F1,128 Continued

'It Flies'

From Gander Software

The Ultimate Personal Planner



- 30 DAY FREE TRIAL IF NOT SATISFIED, FULL REFUND UPON RETURN
- "Daily Appointment Schedule" Work Sheets for all Aspects
- "Future Planning List" sorted
- Year & Month Printed Calendar
- "To Do List" by rank or date.
- Transfers to Daily Schedule

A SIMPLE BUT POWERFUL TOOL FOR SUCCESS

Put the two most effective success techniques to work for you - every day of every year. Just five to ten minutes a day allows your mind and dreams to take charge of your life.

Set Your Goals: To reach a goal, you have to know where you are going. Just enter your goals or future appointments and let your computer remind you.

Set Your Priorities: Success depends upon doing first things first. Assign priorities (1-99) to your "To Do" list, let the computer keep them ranked by date or priority, and then get to work. When the time comes, the computer will help you transfer items to your choice of time on the daily Appointment Scheduler.

Technicalities - Appointment Scheduler: 18 time slots per day (you define) for 60 days. To Do List: 60 items ranked by date or priority. Future Planning: 60 long range items, date sorted; days to event or days overdue. Transfer to Scheduler; just tell it the date and time. Printed Calendars: Year on a page and one month box planning; any month, any year. System uses both Julian and Gregorian calendars to handle dates from 1910-2399 and produce day of the week. Screen and menu driven; DMS Keybase com-patible files. Detailed 38 page manual. Simple installation; FD to Multi HD. Files for 5 users=5;400 appointments. Unlimited Warranty.

HARDWARE: 48K OSI, 8" floppy or hard disk, serial terminal system, OS-65U v. 1.3 or later.

FEATURES: package allows configuration to ANSI standard and almost all non-ANSI terminals, AND user specification of printer port.

PRICE: \$150.00 (User Manual, \$25.00, credited toward TTP purchase). Michigan residents add 4% sales tax.

DEALERS: Your inquiries are invited. This program should be on every 65U machine, including your own. At dealer prices, you could bundle this superior package as a sales incentive.



Bug Fix for the WP-3 and WP-6502 Word Processing Programs

when using DBI's Denver boards.

```
WP-3 CHANGES
CREATE FILE - BASIC 4 25088 N PASS
WP-3 Program (WP-3)
     IF PEEK(16317)=5 THEN GOSUB 4000
24
     IF PEEK(16317)=5 THEN GOSUB 4010 :GOTO 140
115
215 IF PEEK(16317)= 5 GOTO 360
1070 IF PEEK (16317)=5 THEN GOSUB 4020 :GOTO 1080
1075 POKE8778,0:POKE8779, 152:X=USR(X)
4000 FLAG 52,3,0
4001 FLAG 52,5,0
4002 FLAG 52,6,0
4003 FLAG 52,8,0
4004 RETURN
4009 REM *** SET DV#8 FOR PARALLEL PORT ***
4010 FLAG 57,0,4,8
4011 RETURN
4019 REM *** SET DV#8 TO SERIAL PORT ***
4020 FLAG 57,0,1,8
4021 RETURN
WP-3 Program (BASIC)
80 IF X=5 THEN A$="BASIC4" : REM MULTIPROCESSING BASIC
WP6503 CHANGES
5 IF PEEK(65535)=254 THEN POKE 26885,76:POKE 26886,213:POKE
26887,104
WP6502 CHANGES
5 IF PEEK(65535)=254 THEN POKE 26876,76:POKE 26877,204:POKE
26878,104
```

Continued from page 20 1031 POKE F2,135:POKE F3,135 1035 RETURN 1040 FOR Y=1 TO LEN(D\$): POKE D+Y, ASC (MID\$(D\$,Y,1)):NEXT:RETURN 2000 REM**SEARCH AHEAD, OSI V3.3** 2005 NP=CP:CX=32:CY=32 2010 SI=SGN(CI):SJ=SGN(CJ):AI=ABS (CI):AJ=ABS(CJ) 2015 IF AI=0 THEN 2030 2016 AI=AI-1 2019 IF SI>0 THEN NP=NP+1 2020 IF SI>0 THEN 2025 2021 NP=NP-1 2025 CX=PEEK(NP) 2030 IF AJ=0 THEN 2045 2031 AJ=AJ-1 2034 IF SJ>0 THEN NP=NP+64 2035 IF SJ>0 THEN 2040 2036 NP=NP-64 2040 CY=PEEK (NP) 2045 IF NOT (CX=32 AND CY=32) THEN RETURN 2050 IF AI=0 AND AJ=0 THEN RETURN 2055 GOTO 2015 Robert Jents El Sobrante, CA 94803 * * * * * ED:

First, let me thank you for the software listings. I had no idea there was so much available.

In the November issue, Frank Glandorf mentioned that the locations for the comma and

MEM PLUS

BARE \$ 75	32K \$300	52K \$415
16K \$200	40K \$350	56K \$440
24K \$250	48K \$390	64K \$490

MEM+ Options Include:

 OSI compatible floppy disk controller 	add \$85
• RTC — Real Time Clock — day, date and time with lithium battery backup	add \$85
Centronics parallel printer interface with software for OSSED and OSSEU	add \$65

- software for OS65D and OS65U add \$65 • High reliability sockets for memory chips add 15%
- RTC only (OSI CA-20 replacement)
 \$195
- All boards feature solder mask, silkscreen, **gold-plated** edge connectors and a one year warranty.



High Resolution Color Graphics

Our new Color Plus board provides 256 x 192 highresolution graphics with 15 colors. Two 8-bit resolution joystick interfaces are included. Software extensions to OS65-D BASIC provide a superset of APPLE II graphics instructions.

Color Plus connects to the standard 48-pin bus or the 16-pin bus.

Pricing:		
CP-8 for C8 or C3 computers:		\$195
CP-4 for C4 computers (5V only):		\$245
CP-bare Bare board with software:	·	\$ 75

VISA, MasterCard, personal checks and C.O.D.s all accepted. Add \$5 per board for shipping and handling.

To order, or for more information, contact: Fial Computer 5221 S.W. Corbett Portland, Oregon 97201 (503) 227-7083



colon string terminators has been swapped. Well, those weren't the only ones! In the V3.3 reference manual, on page 21, is a table for the values to be poked for random file operation. The locations are 12042 and 12076. The table shows 12042 as the location for the number of records per track. This is wrong! 12076 is the right one. In using these pokes, the order is important. Poke 12042 first, then 12076. The values listed in the table are correct.

I recently bought and tried to use a modem on my C4P MF without any success. After fighting the program supplied by OSI (by the way, it works), I saw an ad by Aurora Software for an intelligent terminal program. I called and, while talking to them, I mentioned my problem. They told me that some C4s had the modem plug wired differently than others and to check that pin 5 was wired to ground (it wasn't). Maybe this will help someone else. I haven't received the program from Aurora yet. If no one else writes in about it, I'll let you know how it works.

Now, questions. Does anyone out there use a D&N Micro Z80 cpu card in their C4 or C8 system? I am interested in upgrading my system to something closer to a standard. Does anyone know of an 80column board for video systems? Orion Software was the only one I had heard of, and they're out of the business.

Norman Thorsen Poulsbo, WA 98370

Norman:

To the best of our knowledge, D&N Proxy Z80 boards support video systems, but must be ordered with a video EPROM.

Readers, how about the 80column question?

Peek Staff

* * * * *

ED:

I am writing about the article that appeared in the Feb. 1984 issue by Guy Vanderwaeren. As the author of the article in MICRO that was referenced, I would regard it as only courteous to give the full reference ("Building a Parallel Printer Interface", MICRO 53, #10 (Oct 82), p. 23, by Rolf B. Johannesen.) With regard to Mr. Vanderwaeren's modifica-

tion, I have only a couple of comments. First, with regard to the EPROM, if your system has a disk, the EPROM is obviously unnecessary, since the print routine can be made to reference the printer port you have built, and you can then forget it. Even with only BASIC-in-ROM, my printer code requires only 21 bytes and I don't regard this as so oner-ous that I would add an EPROM in preference to POKing this in each time I run. Secondly, there is a misunderstanding of the way the PRINT routine works in the ClP. When the ClP is first turned on, the user is asked first for the amount of memory, then the terminal width. If no number is entered to the width question, the value used by BASIC defaults to 72, and this determines the number of characters sent to the printer port before the return-line-feed is sent. At turn-on, another number may be entered - I have entered a number as large as 150 and had the program run correctly. Since BASIC alcorrectly. Since BASIC al-ready counts the number of characters before sending a return-linefeed to the printer port, it is quite unnecessary for the programmer to do this again. All of this is quite independent of the screen display routine, which does in-deed send a return-linefeed (but only to the screen) after every 24 characters.

Rolf B. Johannesen Rockville, MD 20853

AD\$

Send for free catalog, Aurora Software, 37 South Mitchell, Arlington Heights. IL 60005. Phone (312) 259-3150.

* * * * *

WANTED: C4PMF System. Prefer C4P DMF with 48K. Also mailing and word processing software. Send description to: Norman Thiel, 2021 Grismer #37, Burbank, CA 91504 or call 818-847-4664(day) 818-954-0549 (evening).

* * * * *

FOR SALE: LIKE NEW OSI C8PDF; 2-8" drives, 48K. Includes UCSD PASCAL & FORTRAN, Word Proc's WP6502 V1.2 & V1.3, WP-2 & WP-3, 65U V1.2, 65D V3.2 & V3.3, DAC, Plot Basic, Home Cont, OS-DMS Nucleus. Sort, Educa, Inv-I, Inv-II, A/R, A/P, G/L, Purchasing, Query, Bill of Materials, Quote Estimation. 5 Game, 8 MDMS, includes Planner & Plotter, 3 Educational, and 8 Misc. Program Diskettes, over 100 disks total. Also, many manuals for above, Tech Notes, PEEK(65)'s, video 100 Monitor & disk caddy. \$2500 or best offer. Jim Jansen, 12 Cedar. Great Falls, MT 59405, (406) 727-2110.

* * * * *

EPROMS: *BASIC3: fully corrected garbage collector, eliminates string bug. *BASIC1 /BASIC4: one key 24/48 video on Series 2 ClP/SB, one key screen clear, true backspace, more. *MONITOR: corrected keyboard, screen editor, BASIC shorthand, terminal, machine code dump, more. \$15 ppd. each EPROM. SASE for information. SOFTWARE SOLUTIONS, BOX 3753, Seattle, WA 98124.

* * * * *

FOR SALE: 32K OSI Challenger C3 with dual 8" drives. Okidata CP110 Printer with Centronics Interface. ADM-3A Terminal and Standard OSI Software. \$1500 for everything. (313) 736-3923 leave message.

* * * * *

C2-OEM (two cases) with 48K RAM, dual 8" floppies, includes Centronics interface. RS-232C board, OS-65D, OS-65U, plus miscellaneous software. Almost brand new. No documentation. Must sell. \$1500 (includes shipping) or offer. Rick Brown. 316 California #712, Reno, Nevada 89509, 702-322-9936.

* * * * *

FOR SALE: C8P with 48K. Includes Anadex DP-8000 bidirectional printer and interface, many diskettes, joysticks. Well maintained --all records. Available immediately. \$960. 860-4915 Reston, VA evenings.

* * * * *

FOR SALE: ADM-3A Lear Seigler CRT/Modem with acoustic coupler. Hardly used. Over \$1300 invested. \$400. Reston, VA 860-4915 evenings.

* * * * *

SUPER SALE: ClPMF w/24K, full documentation, Sams Manual. Cassettes & Diskettes Software & 0S-65D. Computer in excellent condition, BAD IC in drive. \$500 complete. 12" mon/ tv if you pay shipping. Oscar Frontera, Box 3517 Mayaguez, P.R. 00709. (809) 834-1950.

* * * * *



P.O. Box 347 Owings Milis, Md. 21117

BULK RATE U.S. POSTAGE **PAID** Owings Mills, MD PERMIT NO. 18

.....

DELIVER TO:



ŝ

۰.

GOODIES for **DSI** Users! PEEK (65) The Unofficial OSI Users Journal

P.O. Box 347 • Owings Mills, Md. 21117 • (301) 363-3268

()	C1P Sams Photo-Facts Manual. Complete schematics, scope waveforms and board photos. All you need to be a C1P or SII Wizard, just	\$7.95 \$	· · · · · · · · · · · · · · · · · · ·
()	C4P Sams Photo-Facts Manual. Includes pinouts, photos, schematics for the 502, 505, 527, 540 and 542 boards. A barragin at	¢15.00 ¢	· ·
(١	C2/C3 Same Photo-Facts Manual. The facts you need to renair the larger OSI computers. Fat with	\$15.00 \$	
•	<i>'</i>	useful information, but just	\$30.00 \$ <u></u>	
())	OSI's Small Systems Journals . The complete set, July 1977 through April 1978, bound and reproduced by PEEK (65). Full set only	\$15.00 \$	
())	Terminal Extensions Package - lets you program like the mini-users do, with direct cursor positioning, mnemonics and a number formatting function much more powerful than a mere "print using." Requires 65U.	\$50.00 \$	
, ,	١	RECED - RASIC program resequencer plus much more Global changes, tables of had references	•••••••	
	,	GOSUEs & GOTOS, variables by line number, resequences parts of programs or entire-programs, handles line 50000 trap. Best debug tool I've seen. MACHINE LANGUAGE - VERY FASTI Requires 65U. Manual & samples only, \$5.00 Everything for	\$50.00 \$	
()	Sanders Machine Language Sort/Merge for 0S-65U. Complete disk sort and merge, documentation shows you how to call from any BASIC program on any disk and return it or any other BASIC program on any disk, floppy or hard. Most versatile disk sort yet. Will run under LEVEL I, II, or III. It should cost more but Sanders says, "sell it for just"	\$89.00 \$	
()	KYUTIL - The ultimate OS-DMS keyfile utility package. This implementation of Sander's SORT/MERGE creates, loads and sorts multiple-field, conditionally loaded keyfiles. KYUTIL will load and sort a keyfile of over 15000 ZIP codes in under three hours. Never sort another Master File.	\$100.00 \$	
()	BOOKS AND MANUALS (while quantities last) 65V Primer. Introduces machine language programming.	\$4.95 \$	
())	C4P Introductory Manual	\$5.95 \$,
())	Basic Reference Manual — (ROM, 65D and 65U)	\$5.95 \$	
())	C1P, C4P, C8P Users Manuals - (\$7.95 each, please specify)	\$7.95 \$	
()	How to program Microcomputers. The C-3 Series	\$7.95 \$	
()	Protessional Computers Set Up & Operations Manual — C2-OEM/C2-D/C3-OEM/C3-D/C3-A/C3-B/C3-C/C3-C'	\$ 8.95 \$	
,				\$
Acc	י הוי	Int No. Expiration Date MD Residents a	dd 5% Tax	\$
Siar	าล	ture C.O.D. orders a	dd \$1.65	\$
Nam	ne	Postage & Hand	lling	\$ <u>3.50</u>
Stre	et	TOTAL DUE		\$
City		Zip POSTAGE MAY	VARY FOR OVER	SEAS

. . .

ł