OHIO SCIENTIFIC

SALES/TECHNICAL

NEWSLETTER # 21

October 12, 1979

ATTENTION TECHNICIANS

C2-OEM's Using Power One Triple Output Power Supply

This C2-OEM is being shipped with the following configuration on the 5 volt BUS:

Slots #1 and #2 will support 1.9 amps at 5 volts D.C.

Slots #3 thru #8 will support 6 amps at 5 volts D.C.

Serial System:

Slots #6 and #7 contain two memory boards each using 400 m.a. and slot #8 contains a 505 CPU board which uses 1.3 amps. This leaves a total of 3.9 amps for additional memory and/or accessory boards in slots #3 thru #5.

Video System:

Slot #5 contains a 540 video board which uses 1.5 amps. Slots #6 and #7 contain two memory boards each using 400 m.a. and slot #8 contains a 505 CPU board which uses 1.3 amps. This leaves a total of 2.4 amps for additional memory and/or accessory boards in slots #3 and #4.

It is recommended that additional boards be inserted beginning in the first available slot closest to the front of the unit taking into account the +5 volt loading on the power supplies.

Note: Slot #1 is at the rear of the unit.

C3-OEM's Using Power One Triple Output Power Supply

This C3-OEM is being shipped with the following configuration on the 5 volt BUS:

Slots #1 and #2 will support 1.9 amps at 5 volts D.C.

Slots #3 thru #8 will support 6 amps at 5 volts D.C.

Slots #5 contains a 510 CPU board which uses 2 amps, slots #6 and #7 contain two memory boards each using 400 m.a. and slot #8 contains a 470 floppy interface board which uses 700 m.a. This leaves a total of 2.5 amps for additional memory and/or accessory boards in slots #3 and #4.

It is recommended that additional boards be inserted beginning in the first available slot closest to the front of the unit taking into account the +5 volt loading on the power supplies.

Note: Slot #1 is at the rear of the unit.

C2-OEM/C3-OEM POWER SUPPLIES CONTINUED

Subject: Use of Power One supply in C2-OEM and C3-OEM.

Purpose of Change: To allow use of Power One Model HCAA-60W in C2-OEM and C3-OEM computers.

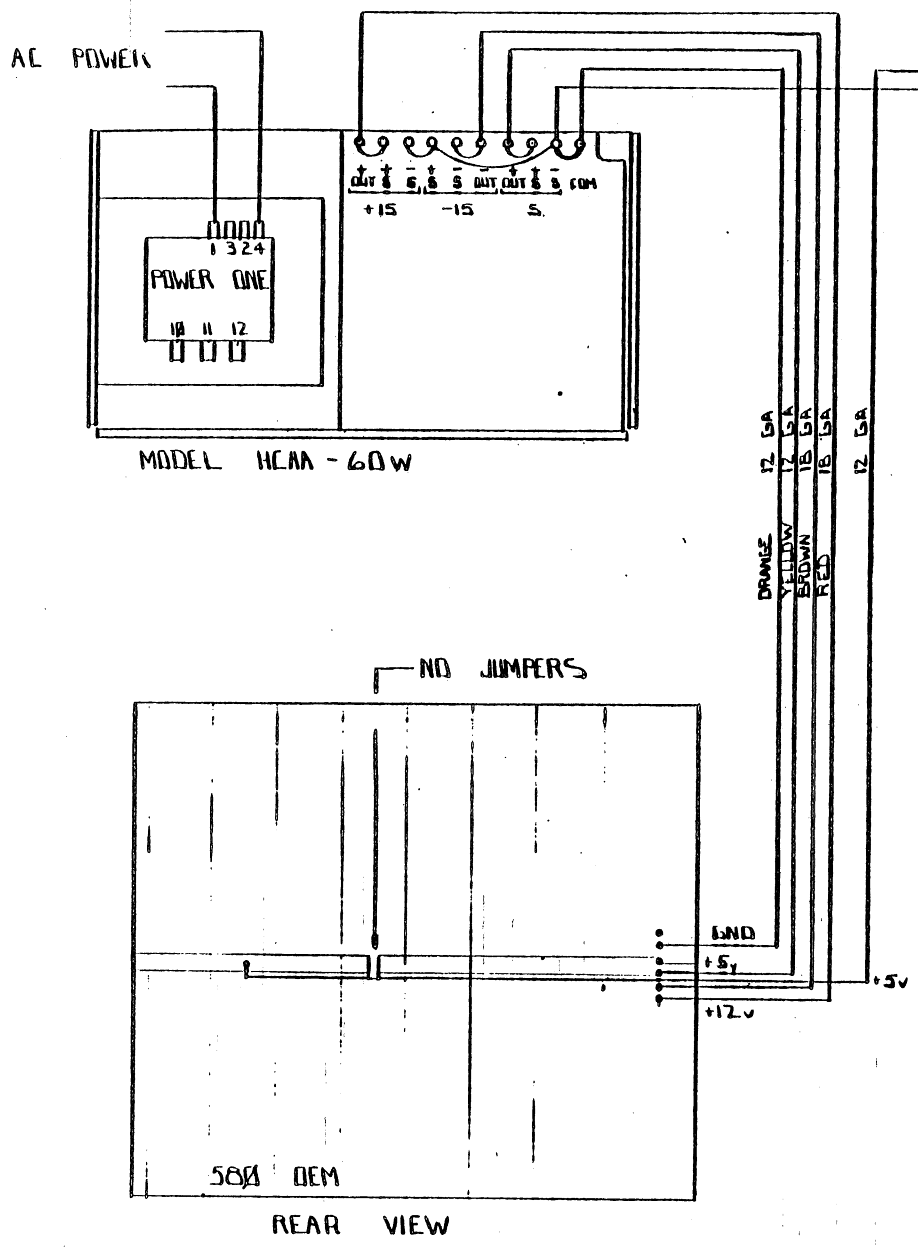
Change: All C2-OEM and C3-OEM computers will use a Power One Model HCAA-6W power supply in place of the Deltron MPS-2 power supply. The connections to the Power One transformer are different from the Deltron unit. The Power One transformer connections are listed on the power supply chassis.

The connections from the HCAA-60W power supply to the backplane are shown in the attached diagram.

Since the HCAA-60W can supply only 6 amps at +5 volts, it will be connected to the front 6 slots of the backplane (slots 3 thru 8). Cut the +5 volt foil which connects the +5 volts between the front 6 slots and rear 2 slots of the backplane. The +5 volts to the rear 2 slots of the backplane (slots 1 and 2) is connected to the 5 volt, 4.5 amp power supply as shown in the attached diagram. The following configurations will be used for C2-OEM and C3-OEM systems.

Slot #	C2-OEM (serial)	C2-OEM (video)	C3-OEM
1			
2			
3		540 B-1	510
4	Memory (optional)	Memory (optional)	Memory (opt.)
5	Memory	Memory	Memory
6	Memory	Memory	Memory
7			
8	505 B	505 B	470

Note: Slot #1 is at the rear of the unit.
The -15 volts must be adjusted for -9 volts.


AC POWER
1 3 2 4
POWER ONE
10 11 12
MODEL HCAA-60W
+ OUT
+ S
- S
- S
- OUT
+ OUT
+ S
- S
COM
+15
-15
5.
24v POWER SUPPLY
GND
+5v
5v 4.5A POWER SUPPLY
12 GA
12 GA
18 GA
18 GA
12 GA
ORANGE
YELLOW
BROWN
RED
NO JUMPERS
GND
+5v
+12v
+5v
580 OEM
REAR VIEW

WHAT HAPPENED TO THE 530 BOARD?

Ohio Scientific has been manufacturing the CM-4 16K dynamic memory board for over a year. From their introduction through approximately July, 1979, the CM-4 memory board performance was within acceptable bounds. Dynamic memory systems in the field were operating satisfactorily. In July of 1979, the company began encountering increasing problems with the CM-4 memory boards. Though some newly manufactured CM-4 memory boards were operational, the majority of 530 boards then produced, were not. The problem manifested itself in the form of soft errors.

In retrospect, the problems appeared to have been caused by "tighter" specifications for the 4027 dynamic memory chips. The specifications for these chips apparently "tightened" when the manufacturers die-shrunk the chips. Ohio Scientific engineering, after an extensive investigation, found several ways to improve the 530 board. These improvements include improved grounding, undershoot loading resistors on data and control lines as well as improved timing. The result is a far better product. Tests now indicate that with these modifications any 530 board ever produced by Ohio Scientific will perform as reliably as the CM-3 memory board. All the upgrades to the 530 board made by this extensive engineering effort are included here in one place for your reference. We strongly encourage you to make these modifications to any dynamic memory based computer in which memory problems are suspected. The end result will be a computer system with reliability rivaling that of our CM-3 static memory board based systems.

530 BOARD PRODUCT UPDATE

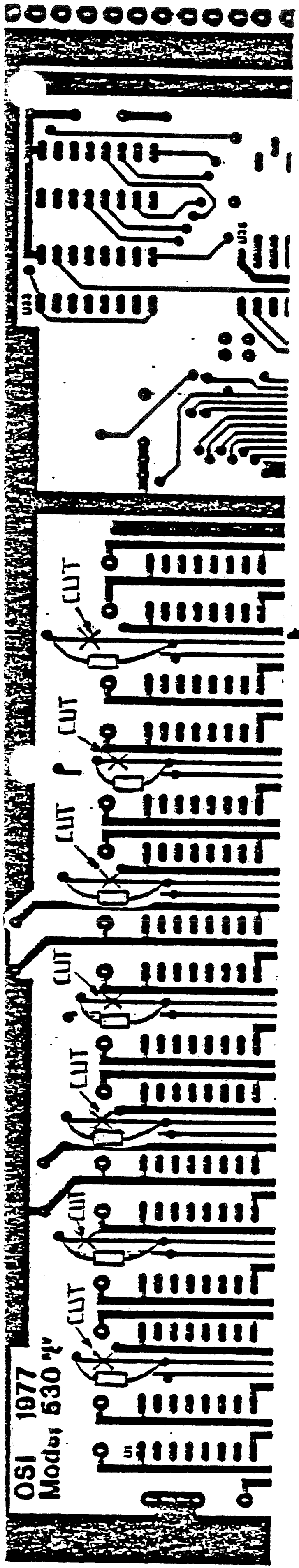
PURPOSE OF CHANGE: TO DECREASE NOISE ON THE 530 BOARD AND TO DECREASE NEGATIVE UNDERSHOOTS ON THE DIN LINES TO THE MEMORY CHIPS. ALSO A TIMING CHANGE IS BEING MADE TO INCREASE RELIABILITY OF MEMORY BOARDS.

CHANGE: MAKE EIGHT (8) TAPE CUTS AND ADD EIGHT (8) 150 OHM, ¼ WATT, 5% RESISTORS AS SHOWN IN DETAIL A AND DETAIL B OF THE ATTACHED SHEET.

MAKE TAPE CUT AND INSTALL JUMPER AS SHOWN IN DETAIL C. THIS CHANGES THE SIGNAL GOING TO PIN 2 OF U54 FROM DL1 TO IØ2. THIS CAUSES $\overline{RAS}$ TO BEGIN WITH IØ2 INSTEAD OF DL1.

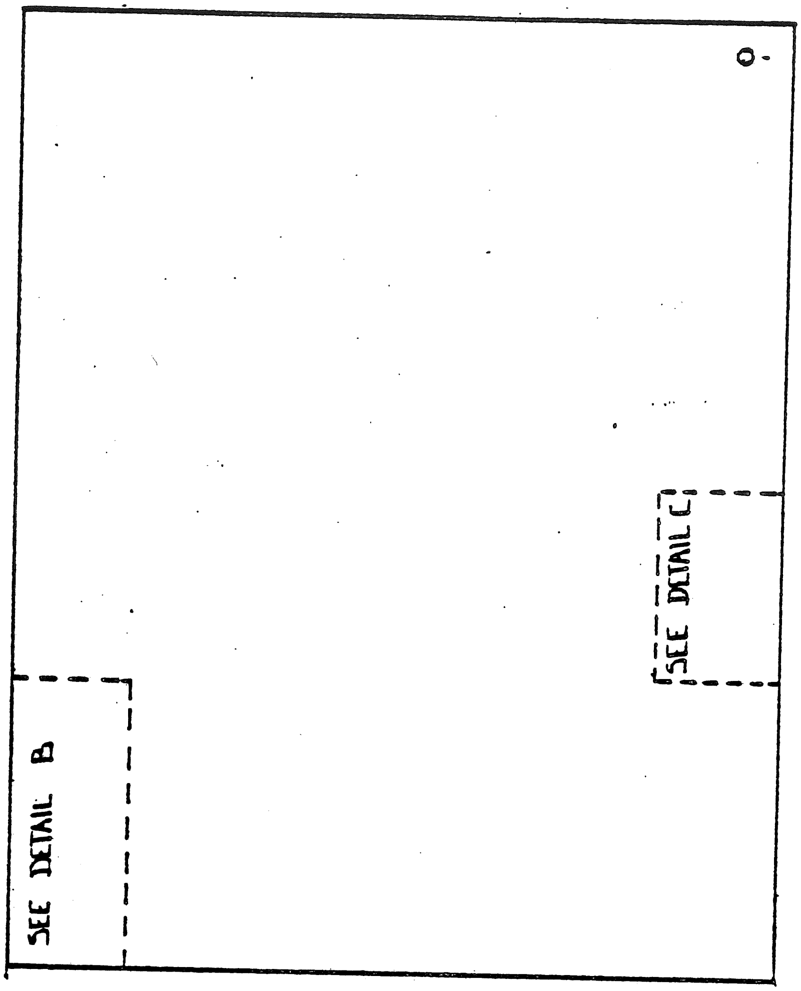
NOTE: THE FOUR (4) 100 OHM RESISTORS ON THE $\overline{RAS}$ LINES MUST BE INSTALLED ALWAYS AS SHOWN IN TECH LETTER NUMBER 19. IN ADDITION, WHEN THE $\overline{RAS}$ RESISTORS ARE INSTALLED A 74S138 SHOULD BE USED.

530 PRODUCT UPDATE CONT.

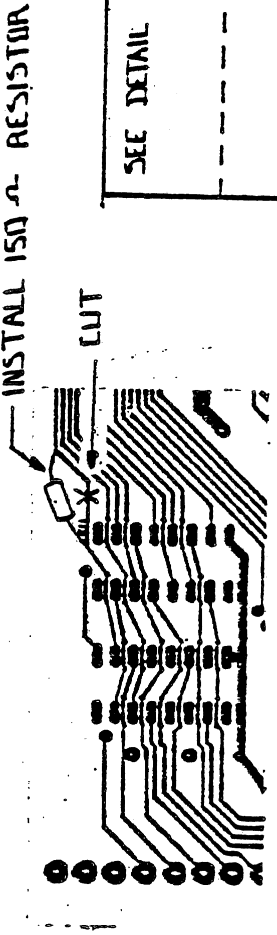


MAKE 7 TAPE CUTS WHERE SHOWN AND INSTALL 7 150 Ω RESISTORS

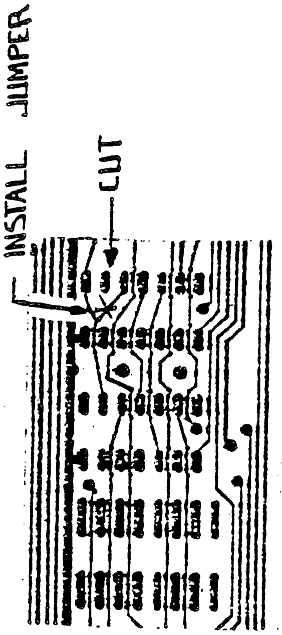
DETAIL A
COMPONENT SIDE OF 530 BOARD



REAR VIEW OF 530 REV C BOARD



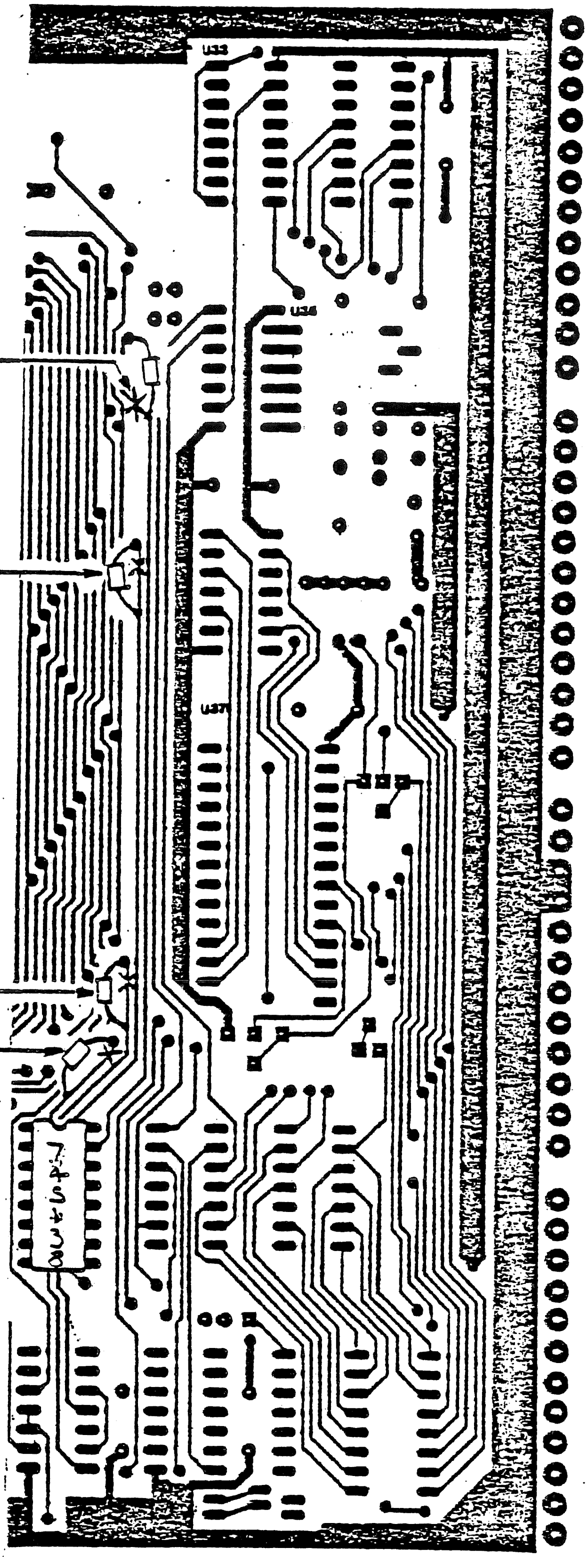
DETAIL B
REAR OF 530 BOARD



DETAIL C
REAR OF 530 BOARD

530 BOARD PRODUCT UPDATE CONT.

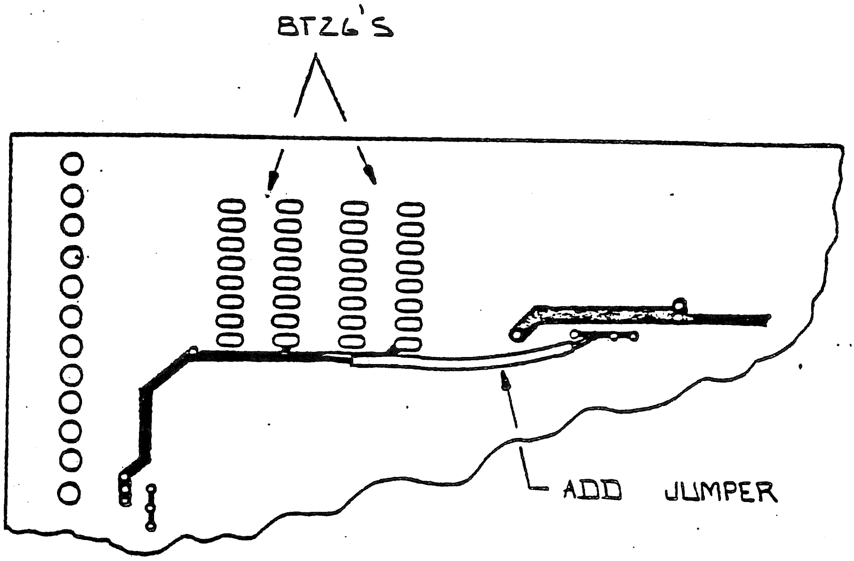
PURPOSE OF CHANGE: To decrease possible noise on the 530 Board and to decrease possible negative going undershoots on the $\overline{\text{RAS}}$ lines. These must not be greater than -1.0 volt.



MAKE THESE 4 CUTS ON COMPONENT SIDE OF 530 BOARD, AND INSTALL 4 100-Ω RESISTORS

CHANGE: All 530 Memory Boards should use 74LS138's instead of 74S138's. In some cases this may not be sufficient. This can be determined by scoping the $\overline{\text{RAS}}$ lines to make sure negative going undershoots are less than -1.0 volts. In these cases 100 ohm series resistors should be installed as shown in the drawing. When 100 ohm resistors are installed either 74S138's or 74LS138's may be used.

530 BOARD PRODUCT UPDATE CONT.



REAR VIEW OF
530 REV C MEMORY BOARD

The change shown above decreases ground noise between the 8T26's and the memory array. The jumper should be 18 gauge or heavier wire. The jumper should go from the tape that runs between pin 8 of the two 8T26's to the ground pad near the memory as shown above.

#21,9

OS-65U CD-74 VERSIONS ONLY

Correction to disc initialization routines to prevent possible improper initialization beyond cylinder 256, track 11 (55,265,279). This correction is for 6/79 and later releases of OS-65U for the CD-74. Please verify that the original contents are the same as shown in the "CHANGE" conversation before inserting this fix. If the original contents are not the same, disregard this fix!

```
RUN "CHANGE","PASS

DISK CHANGE UTILITY

MODE: HEX(H), DEC(D) ? H
UNIT ? E
ADDRESS OFFSET ? CØØ
ADDRESS ? 37D5
ØØØØ37D5   DØ ? ØA
ØØØØ37D6   D5 ? DØ
ØØØØ37D7   85 ? D4
ØØØØ37D8   A3 ? 6A
ØØØØ37D9   E6 ? 85
ØØØØ37DA   A4 ? A3
ØØØØ37DB   9Ø ? .
ADDRESS ? 3763
ØØØØ3763   DØ ? E6
ØØØØ3764 G 47 ? A4
ØØØØ3765   A5 ? DØ
ØØØØ3766   A3 ? 45
ØØØØ3767   Ø9 ? Ø6
ØØØØ3768   8Ø ? A3
ØØØØ3769   85 ? 38
ØØØØ376A   A3 ? 66
ØØØØ376B   DØ ? A3
ØØØØ376C ? 3F ? DØ
ØØØØ376D   ØØ ? 3E
ØØØØ376E   A9 ? X

OK
CLOSE

OK                 ORIGINAL CONTENTS
```

OS-65U POKES

The BASIC line below may be used to "bump" the cursor one position to the right each time it is executed. When the cursor moves to the 64th position it will automatically perform a carriage return (without a linefeed). This line is useful in those applications where a great deal of processing is being performed and a method of showing that the computer is active is required.

```
PRINT " ";:  If PEEK (22) >=63 Then PRINT CHR$(13);
```

OS-65U LEVEL III

Page 8-21 of the OS-65U LEVEL I/III manual contains an incomplete formula for checking semaphores. The correct program is shown below.

```
1Ø   For N=1 to 255
2Ø   If (PEEK(55333+N/8) AND 2 ∧ (N-INT(N/8)*8)) > Ø THEN 4Ø
3Ø   PRINT N
4Ø   NEXT N
```

OS-DMS CORRECTIONS

A problem has been found in a few of the OS-DMS programs. Under some rare conditions the program(s) could mis-"FIND" any entry. The corrections to the individual programs follow.

CORRECTION TO "GLPOST" IN THE GENERAL LEDGER

Enter the following lines into "GLPOST" and SAVE "GLPOST" with a password of "PASS".

```
5Ø91 X1=INDEX(1): REM SAV INDX
531Ø IFH$<>FC$(G) GOTO 54ØØ:REM NO MATCH
532Ø REM
533Ø IF G$<>FDLB$(G) GOTO 54ØØ
5332 IF FC$(6)="H" OR FC$(6)="A" THEN 541Ø
534Ø GOSUB 1Ø1ØØ:REM PRT RECORD
5345 GOTO 91Ø
54ØØ INDEX <1> =X1+LEN(H$):  GOTO 5Ø9Ø:REM BMP FND
541Ø PRINT "CANNOT POST $ TO THAT ACCOUNT  TRY AGAIN":  GOTO 55ØØ
55ØØ GOTO 7ØØ
```

CORRECTION TO "GLEDTM" IN THE GENERAL LEDGER

Enter the following lines into "GLEDTM" and SAVE "GLEDTM" using a password of "PASS".

```
5Ø91 X1=INDEX(1):REM SAV FND PTR
531Ø IF H$< >FC$(G) GOTO 54ØØ:REM NO MATCH
532Ø REM
533Ø IF G$< >FDLB$(G) GOTO 54ØØ
54ØØ INDEX<1>=X1+LEN(H$):REM BMP PST FND
541Ø GOTO 5Ø9Ø
```

A NOTE ON THE SERIAL MEMORY TEST

The serial memory test program recently released is very useful. However, one should not consider the MTP as a means of verifying a new piece of equipment. No memory test will catch all possible modes of memory failure. As a general rule of thumb, the MTP should be used only on equipment already in the field. New equipment should be rigourously tested and inspected before it is delivered to the customer. Do not assume that a piece of equipment is fine just because it passes the memory test program.

BASIC

There has been some misunderstanding concerning the SGN function in BASIC. The correct values returned by that function are shown below:

SGN (X) where X<Ø returns SGN (X) = -1

SGN (X) where X = Ø returns SGN (X) = Ø

SGN (X) where X>Ø returns SGN (X) = +1

OS-65U PNTR COMMAND

The following pages show the BASIC interface and machine code source for the PNTR command. The PNTR command returns one or two arguments. The command Syntax is:

PNTR Variable name1, Floating point variable 2 <CR>

OR

PNTR String Variable name1, FLT'N PNT Var2, FLT'N PNT Var 3 <CR>

The first form returns the address of variable 1's descriptor. The second form is for string variables only. This form returns the address of the string variable's actual contents in FLT'N PNT Var 2 and it's length in FLT'N PNT Variable 3. A null string (length = Ø) returns an address and length of zero.

Variables must have been defined before they are used in a PNTR command or an incorrect address can be returned. This is caused by BASIC moving all array descriptors upward in memory in order to add a simple variable descriptor. Note also that string variable's contents may be shifted if garbage collection occurs or string concatenation is performed.

PNTR COMMAND IMPLEMENTATION

1) Enter the source code as shown.

2) Assemble and save the PNTR command code to disc.

3) Using LOAD32 or LOAD48 "call" the machine code into OS-65U at $6ØØØ.

4) Return to OS-65U and enter a "NEW185".

5) Enter the subroutine as shown in the BASIC demo program.

6) SAVE the program to disc.

A GENERAL NOTE

When using the PNTR and other transistory commands a small amount of caution is required. First one must always disable these transistory commands before running another program. Second, when entering a transistory command into a BASIC program one must use the currently enabled reserved word. In other words, if one is entering the SWAP command and the EDITor is enabled, then one must enter the word EDIT in place of the word SWAP. This rule applies for the following comands:

RSEQ, EDIT, SWAP, PNTR, NULL

When the command being used is enabled by the BASIC program, then that command may be entered as a normal command. Also any commands such as SWAP that were entered as another command, e.g. EDIT, will be changed to the correct word upon enablement of the command. In other words, if "EDIT" was entered and then the SWAP command was enabled, the word EDIT would be changed to "SWAP".

SWAP COMMAND FIX

Line 6387Ø on page 13 of Tech Letter #16 should read as follows:

```
6387Ø POKE 8738,255 :  POKE 8739,Ø95
```

Instead of the incorrect line

```
6387Ø POKE 8738,ØØ : POKE 8739,96
```

OS-65U NEC PRINTER DRIVER

The following POKES will disable paging in the NEC printer driver for OS-65U.

1) Enter : LOAD"NECDRV"<CR>

2) For a serial system enter:

```
POKE 25147,24Ø : POKE 25148,27<CR>
(Normally (25147)=2Ø1 & (25148)= 1Ø)
```

For a video system enter:

```
POKE 25531,24Ø:  POKE 25532,27<CR>
(Normally (25531)=2Ø1 & (25532)=1Ø)
```

3) If the change is to be permanent enter:

```
SAVE "NECDRV", "PASS"<CR>
```

4) To make the change effective enter:

```
RUN<CR>
```

```
100 REM                        PNTR CMD DEMO
110 REM               (C) 1979 BY OHIO SCIENTIFIC, INC.
120 REM                      ALL RIGHTS RESERVED
130 REM                   WRITTEN BY R. WHITESEL 9/79
140 REM
150 REM
160 REM
170 PRINT: PRINT: PRINT: PRINT ":::::::::::::::::::::::::"
180 PRINT ": PNTR CMD DEMO PROGRAM :"
190 PRINT ":::::::::::::::::::::::::": PRINT: PRINT: PRINT
200 REM
210 REM ENABLE "PNTR" CMD
220 REM
230 GOSUB 63700
240 REM
250 REM MUST DECLARE VARIABLES BEFORE USE IN PNTR CMD
260 REM
270 DIM A$(10): X=0: DESC=0: ADR=0: LN=0: Y=0: QA$=""
280 REM
290 REM A LITTLE DEMO
300 REM
310 FOR X=1 TO 10
320 REM
330 REM CONVERT X TO A STRING
340 REM
350 QA$="A$("+MID$(STR$(X),2)+")"
360 REM
370 REM EQUATE A$(X)
380 REM
390 A$(X)="THIS STRING IS TEST STRING #"+STR$(X)
400 REM
410 REM GET ADDRESS OF A$'S DESCRIPTOR
420 REM
430 PNTR A$(X),DESC
440 REM
450 REM GET ADDRESS OF A$'S CONTENTS AND IT'S LENGTH
460 REM
470 PNTR A$(X),ADR,LN
480 REM
490 REM PRINT THE VALUES
500 REM
510 PRINT QA$;"'S DESCRIPTOR IS LOCATED AT ADDRESS";DESC;"."
520 PRINT QA$;"'S ACTUAL CONTENTS MAY BE FOUND AT ADDRESS";ADR;"."
530 PRINT "THERE ARE";LN;"CHARACTERS IN ";QA$;"."
```

```
540 PRINT QA$;"'S CONTENTS ARE SHOWN BELOW :": PRINT: PRINT
550 REM
560 FOR Y=ADR TO ADR+LN-1
570 PRINT CHR$(PEEK(Y));
580 NEXT Y
590 PRINT: PRINT "-----------------------------": PRINT: PRINT
600 POKE 2888,0: INPUT"HIT RETURN TO CONTINUE ";QA$
610 POKE 2888,27: FOR Y=1 TO 33: PRINT: NEXT Y
620 NEXT X
630 REM
640 REM GOSUB TO DISABLE "PNTR" CMD
650 REM
660 GOSUB 63900
670 REM
680 END
690 REM
700 REM
710 REM
63700 REM PNTR CMD ENABLE ENTRY POINT
63710 REM
63720 REM SAVE RESERVED WORD
63730 REM
63740 FOR X=9025 TO 9028: T=PEEK(X): POKE X+15727,T: NEXT
63750 REM
63760 REM SAVE DISPATCH ADDRESS
63770 REM
63780 T=PEEK(8738): POKE 24756,T: T=PEEK(8739): POKE 24757,T
63790 REM
63800 REM POKE RESERVED WORD "PNTR" INTO PLACE
63810 REM
63820 POKE 9025,ASC("P"): POKE 9026,ASC("N")
63830 POKE 9027,ASC("T"): POKE 9028,ASC("R")+128
63840 REM
63850 REM POKE "PNTR" DISPATCH ADDRESS INTO PLACE
63860 REM
63870 POKE 8738,255: POKE 8739,95
63880 RETURN
63890 REM
63900 REM DISABLE "PNTR" CMD ENTRY POINT
63910 REM
63920 REM RESTORE RESRVED WORD
63930 REM
63940 FOR X=9025 TO 9028: T=PEEK(X+15727): POKE X,T: NEXT
63950 REM
63960 REM RESTORE DISPATCH ADDRESS
```

```
63970 REM
63980 T=PEEK(24756): POKE 8738,T: T=PEEK(24757): POKE 8739,T
63990 RETURN
```

#21,17

.A

A

```
1000 0000          ;                PNTR CMD CODE
1010 0000          ;         (C) 1979 BY OHIO SCIENTIFIC, INC.
1020 0000          ;             ALL RIGHTS RESERVED
1030 0000          ;          WRITTEN BY R. WHITESEL 9/79
1040 0000          ;          RESIDES IN FRONT OF A BASIC
1050 0000          ;          PROGRAM AT $6000 AND UP.
1060 0000          ;
1070 0000          ;
1080 0000          ;PNTR MAY BE USED TO "GET" A POINTER TO THE DESCRIPTOR
1090 0000          ;OF A VARIABLE OR A POINTER TO THE ACTUAL STRING DATA
1100 0000          ;OF A STRING VARIABLE AND IN ADDTION, THE LENGTH OF THAT
1110 0000          ;STRING VARIABLE.
1120 0000          ;
1130 0000          ;
1140 0000          ;FLOATING POINT DESCRIPTOR :
1150 0000          ;
1160 0000          ;FIRST BYTE      SIGNED EXPONENT W SIGN COMPLEMENTED
1170 0000          ;SECOND BYTE     MSB OF BYTE IS IMPLIED ONE
1180 0000          ;                MSB REFLECTS SIGN OF FRACTIONAL VALUE
1190 0000          ;THIRD BYTE      LESS SIGNIFICANT BYTE
1200 0000          ;FOURTH BYTE     EVEN LESS SIGNIFICANT BYTE
1210 0000          ;FIFTH BYTE      LEAST SIGINIFICANT BYTE
1220 0000          ;
1230 0000          ;
1240 0000          ;INTEGER VARIABLE DESCRIPTOR :
1250 0000          ;
1260 0000          ;FIRST BYTE      MSB IS SIGN, MOST SIGINIFCANT BYTE
1270 0000          ;SECOND BYTE     LSB (NUMBER STORED IN 2'S COMPLEMENT)
1280 0000          ;
1290 0000          ;
1300 0000          ;STRING VARIABLE DESCRIPTOR :
1310 0000          ;
1320 0000          ;FIRST BYTE      LENGTH OF STRING ( 0 LEN SAYS ADDR IS MEANINGLESS)
1330 0000          ;SECOND BYTE     LOW ADDRESS OF STRING DATA
1340 0000          ;THIRD BYTE      HIGH ADDRESS OF STRING DATA
1350 0000          ;
1360 0000          ;
1370 0000          ;COMMAND SYNTAX :
1380 0000          ;
1390 0000          ; PNTR VAR1, VAR2 <CR>
```

```
1400 0000        ; WHERE VAR1 IS ANY VARIABLE, AND VAR2 IS A FLT'N PNT VAR
1410 0000        ;
1420 0000        ; OR
1430 0000        ;
1440 0000        ; PNTR VAR1, VAR2, VAR3 <CR>
1450 0000        ; WHERE VAR1 IS A STRING VARIABLE AND VAR2 & VAR3
1460 0000        ; ARE FLOATING POINT VARIABLES.
1470 0000        ;
1480 0000        ;
1490 0000        ;
1500 0000        ; PNTR VAR1, VAR2 <CR> RETURNS VAR2 SET TO THE ADDRESS
1510 0000        ; OF VAR1. NOTE THAT THE USE OF A UNDEFINED SIMPLE
1520 0000        ; VARIABLE WILL CAUSE THE ADDRESS OF ALL ARRAY VARIABLE
1530 0000        ; DESCRIPTORS TO CHANGE. NOTE ALSO THAT GARBAGE COLLECTION
1540 0000        ; OR STRING CONCATENATION CAN CAUSE THE ADDRESS OF
1550 0000        ; A STRING'S CONTENTS TO CHANGE.
1560 0000        ;
1570 0000        ; PNTR VAR1, VAR2, VAR3 <CR> RETURNS THE ADDRESS OF THE
1580 0000        ; STRING VARIABLE VAR1 IN VAR2 AND THE LENGTH OF VAR1'S
1590 0000        ; CONTENTS IN VAR3. BOTH VAR2 AND VAR3 MUST BE FLOATING POINT
1600 0000        ; VARIABLES. A NULL STRING WILL ALWAYS RETURN
1610 0000        ; ZERO AS THE ADDRESS OF VAR1'S CONTENTS.
1620 0000        ;
1630 0000        ;
1640 0000        ;
1650 0000        ;
1660 0000        ;                  EQUATES :
1670 0000        ;
1680 0000        ;GLOBAL :
1690 0000        ;
1700 0000        PTRGET=@07456     RETURNS PNTR TO VAR
1710 0000        CHRGET=@00300     RTS W CHR @ (TXTPTR,+1) + 1
1720 0000        CHRGOT=@00306     RTS W CHR @ (TXTPTR,+1)
1730 0000        COMCHK=@07023     CHKS CHR @ (TXTPTR,+1) = ","
1740 0000        FLOATC=@15504     FLOATS BIN IN MACHO,MACMHO
1750 0000        SNERR:=@7036      SYNTAX ERROR OUTPUT ROUTINE
1760 0000        FCERR:=@10320     FUNCTION CALL ERR OUT ROUTINE
1770 0000        ;
1780 0000        VARPNT=@00224     (PNTS TO VAR AFTER CALL TO PTRGET)
1790 0000        VARNAM=@00222     (VAR NAME AFTER CALL TO PTRGET)
1800 0000        FACEXP=@00256     FLT'N PNT EXP
1810 0000        FACHO:=@00257     FLT'N PNT MSB
1820 0000        FACMHO=@00260     FLT'N PNT MID MSB
1830 0000        FACLO:=@00262     FLT'N PNT LSB
1840 0000        ;
```

```
1850 0000             ;
1860 0000             ;
1870 6000                    *=$6000  RIDES IN & OUT W BASIC PROGRAM
1880 6000             ;
1890 6000             ;
1900 6000 F025        FNDPNT BEQ SN:ERR        ; EOL IS SN ERR
1910 6002             ;
1920 6002 202E0F             JSR PTRGET        ; GET PNTR TO VAR DESIRED
1930 6005 A594               LDA VARPNT        ; SAV PTR ON STK
1940 6007 48                 PHA
1950 6008 A595               LDA VARPNT+1
1960 600A 48                 PHA
1970 600B 20130E             JSR COMCHK        ; BETTER BE A COMMA OR SN ERR
1980 600E F017               BEQ SN:ERR        ; EOL IS A SN ERROR
1990 6010 209B60             JSR GETTYP        ; GET VAR TYPE
2000 6013 48                 PHA               ; SAV IT ON STK
2010 6014 202E0F             JSR PTRGET        ; GET PNTR TO VAR TO BE SET TO ADR
2020 6017 209B60             JSR GETTYP        ; MUST BE A FLT'N PNT NUMBER
2030 601A C905               CMP #$05
2040 601C D052               BNE FC:ERR        ; IT'S NOT SO- FC ERROR
2050 601E 20C600             JSR CHRGOT        ; END OF ARGUMENT LIST ?
2060 6021 F007               BEQ SETDES        ; YES- GO SET VAR TO ADR OF DESCRIPTOR
2070 6023 C92C               CMP #',           ; BETTER BE A COMMA !
2080 6025 F00A               BEQ SETADR        ; IT IS SO CHK IF VAR TO BE FOUND IS A $
2090 6027 4C1E0E SN:ERR      JMP SNERR:        ; SYNTAX ERROR EXIT
2100 602A             ;
2110 602A 68          SETDES PLA               ; CLR STACK
2120 602B             ;
2130 602B 207360             JSR MOVNUM        ; SET VAR2=ADR OF DESCRIPTOR OF VAR1
2140 602E A900               LDA #$00          ; RTS W NO ERRORS
2150 6030 60                 RTS
2160 6031 20C000 SETADR      JSR CHRGET        ; SCAN OFF ','
2170 6034             ;
2180 6034 F0F1               BEQ SN:ERR        ; EOL IS SN ERROR
2190 6036 68                 PLA               ; VAR1 MUST BE A STRING VAR
2200 6037 C903               CMP #$03
2210 6039 D035               BNE FC:ERR        ; NO SO- FC ERROR
2220 603B 68                 PLA               ; GET ADR OF ACTUAL STRING DATA
2230 603C 8593               STA VARNAM+1      ; USE VARNAM AS A TMP PNTR
2240 603E 68                 PLA               ; NOW THE LOW BYTE
2250 603F 8592               STA VARNAM
2260 6041 A000               LDY #$00
2270 6043 B192               LDA (VARNAM),Y    ; GET LENGTH OF STRING
2280 6045 48                 PHA               ; AND SAV IT
2290 6046 AA                 TAX               ; SAV IN X
```

```
2300 6047 A900            LDA #$00          ; MUST PASS 2 BYTES
2310 6049 48              PHA
2320 604A 8A              TXA               ; A LEN = 0 ?
2330 604B D003            BNE GETADR        ; NO- SO PASS ADDR
2340 604D 48              PHA               ; LEN = 0 SO ADR = 0
2350 604E F007            BEQ PSHADR        ; BRA TO PHA HIGH BYTE
2360 6050 C8      GETADR  INY               ; SET V2=ADR OF $ DATA, NOT $ DESCRIPTOR
2370 6051 B192            LDA (VARNAM),Y    ; LOW BYTE OF ADR
2380 6053 48              PHA               ; SAV ON STK FOR MOVNUM
2390 6054 C8              INY
2400 6055 B192            LDA (VARNAM),Y    ; AND NOW THE HIGH BYTE
2410 6057 48      PSHADR  PHA
2420 6058 207360          JSR MOVNUM        ; SET V2=ADR OF $ DATA OF V1
2430 605B 202E0F          JSR PTRGET        ; GET PNTR TO VAR3
2440 605E 209B60          JSR GETTYP        ; MUST BE A FLT'N PNT NUMBER
2450 6061 C905            CMP #$05
2460 6063 D00B            BNE FC:ERR        ; NO- SO FC ERROR
2470 6065 207360          JSR MOVNUM        ; SET VALUE INTO PLACE
2480 6068 20C600          JSR CHRGOT        ; MUST BE EOL OR SN ERROR
2490 606B D0BA            BNE SN:ERR
2500 606D A900            LDA #$00
2510 606F 60              RTS               ; RTS W NO ERRORS
2520 6070 4CD010  FC:ERR  JMP FCERR:        ; FUNCTION CALL ERROR EXIT
2530 6073         ;
2540 6073         ;
2550 6073         ;         SUBROUTINES :
2560 6073         ;
2570 6073         ;
2580 6073         ;MOVNUM MOVES THE TWO BYTE VALUE ON THE STACK
2590 6073         ;INTO THE FLT'N PNT VAR PNTD TO BY (VARPNT,+1).
2600 6073         ;
2610 6073 68      MOVNUM  PLA               ; MUST SAV RTS ADR ON STK
2620 6074         ;
2630 6074 AA              TAX               ; SAV IN X
2640 6075 68              PLA
2650 6076 A8              TAY               ; & Y
2660 6077 68              PLA               ; GET HI BYTE 1 ST
2670 6078 85AF            STA FACHO:
2680 607A 68              PLA
2690 607B 85B0            STA FACMHO        ; NOW THE LOW BYTE
2700 607D 98              TYA               ; PUSH RTS ADR BAC ON STK
2710 607E 48              PHA
2720 607F 8A              TXA
2730 6080 48              PHA
2740 6081 A290            LDX #16+@200      ; SET UP THE EXP
```

```
2750 6083 38                   SEC                 ; SO BCS TO NORMAL
2760 6084 20441B               JSR FLOATC          ; BIN TO FLT'N PNT
2770 6087 A5AF                 LDA FACHO:          ; RST SIGN BIT
2780 6089 297F                 AND #%01111111      ; B7 IS SIGN BIT
2790 608B 85AF                 STA FACHO:          ; STORE RESULT
2800 608D A000                 LDY #$00
2810 608F A205                 LDX #$05            ; 5 BYTES TO MOVE
2820 6091 B9AE00 MOVNM1        LDA FACEXP,Y        ; GET A BYTE OUT OF FAC
2830 6094        ;
2840 6094 9194                 STA (VARPNT),Y      ; STUFF INTO FLT'N PNT DESCRIPTOR
2850 6096 C8                   INY                 ; LOOK AT NXT
2860 6097 CA                   DEX                 ; LOOP COUNTER
2870 6098 D0F7                 BNE MOVNM1
2880 609A 60                   RTS
2890 609B        ;
2900 609B        ;
2910 609B        ;GETTYP RTS'S W A=VARTYP WHERE FLT'N PNT = 5,
2920 609B        ;STRING = 3, AND INTEGER = 2.
2930 609B        ;
2940 609B A905   GETTYP        LDA #$05            ; DEFAULT IS FLT'N PNT
2950 609D        ;
2960 609D 2492                 BIT VARNAM          ; + = FLT'N OR $
2970 609F 1008                 BPL TSTSTR          ; GO SEE IF $ OR FLT'N PNT
2980 60A1 2493                 BIT VARNAM+1        ; INT HAS TWO BMI'S
2990 60A3 10CB                 BPL FC:ERR          ; NO MUST BE USER DEFINED VAR
3000 60A5 A902                 LDA #$02            ; ITS AN INT
3010 60A7 D006                 BNE TYPRTS
3020 60A9 2493   TSTSTR        BIT VARNAM+1        ; STR OR FLT'N
3030 60AB        ;
3040 60AB 1002                 BPL TYPRTS          ; IT IS A FLT'N PNT
3050 60AD A903                 LDA #$03            ; IT IS A STRING VAR
3060 60AF 60     TYPRTS        RTS

.S0
```