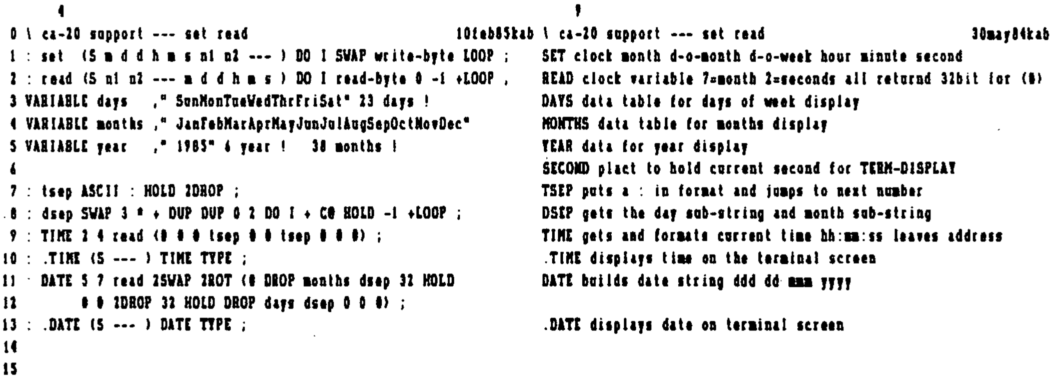
presented here. Read the book, - "STARTING FORTH" by Leo Brodie. Above all, enjoy!

FORTH-83 is in the public domain, and placed there kindly by Laxen and Parry. This program OSI-CLK.F83 I have placed in the bulletin board in Downey, CA., called "NORTHSTAR DOWNEY" (213) 861-2313. Anyone who wishes to leave a message for me, may also do it at NORTHSTAR.



```
      4                                                                 9
 0 \ ca-20 support --- set read                               10feb85kab \ ca-20 support --- set read                          30may84kab
 1 : set  (S m d d h m s n1 n2 --- ) DO I SWAP write-byte LOOP ;     SET clock month d-o-month d-o-week hour minute second
 2 : read (S n1 n2 --- m d d h m s ) DO I read-byte 0 -1 +LOOP ,     READ clock variable 7=month 2=seconds all returnd 32bit for (#)
 3 VARIABLE days   ," SunMonTueWedThrFriSat" 23 days !               DAYS data table for days of week display
 4 VARIABLE months ," JanFebMarAprMayJunJulAugSepOctNovDec"          MONTHS data table for months display
 5 VARIABLE year   ," 1985" 6 year !   38 months !                   YEAR data for year display
 6                                                                   SECOND place to hold current second for TERM-DISPLAY
 7 : tsep ASCII : HOLD 2DROP ;                                       TSEP puts a : in format and jumps to next number
 8 : dsep SWAP 3 * + DUP DUP 0 2 DO I + C@ HOLD -1 +LOOP ;           DSEP gets the day sub-string and month sub-string
 9 : TIME 2 4 read (# # # tsep # # tsep # # #) ;                     TIME gets and formats current time hh:mm:ss leaves address
10 : .TIME (S --- ) TIME TYPE ;                                      .TIME displays time on the terminal screen
11 : DATE 5 7 read 2SWAP 2ROT (# DROP months dsep 32 HOLD            DATE builds date string ddd dd mmm yyyy
12       # # 2DROP 32 HOLD DROP days dsep 0 0 #) ;
13 : .DATE (S --- ) DATE TYPE ;                                      .DATE displays date on terminal screen
14
15
```

★ ★ ★

WHAT IF YOUR SUPERBOARD REFUSES TO BREAK?

By: John Horemans
Courtesy of TOSIE
Toronto Ohio Scientific Idea
 Exchange
P. O. Box 29
Streetsville, Ont.
Canada L5M 2B7

Sorry, I meant, doesn't respond to the BREAK key. The most common symptom is a screen full of characters, yet no action when the break key is held down.

If you think about it, the screen full of characters tells you quite a bit. First of all, the video is working. The video counters are also responding. Most likely too, you have the phase Ø signal to the CPU, pin 37. That leaves a whole area that needs little or no checking.

At this point, it is well to go over any recent changes or soldering you have done. All too often a near invisible thread of solder is left behind. A splash across two traces can be equally frustrating. Close examination, with the help of a magnifier, should find most of these problems quickly.

Check any recently installed chips for bent pins, or proper location of pin 1. More than one of my chips have been consigned to the bin because of this. An 8T28, for instance, will survive for a while like this, but certainly won't allow the computer to operate. Some chips, particularly 24 pin sizes, have a knack for bending the pin in under the chip, making the problem difficult to see. If possible, sight along the plane of the board to detect these pins.

If you have recently installed a ROM or EPROM chip, check the enable lines, pin 18 and 2Ø, as well as pin 21. Contrary to some OSI documentation, the 2716 you are likely installing, needs +5 on pin 21. Chip enable, pin 2Ø, and output enable, pin 18, are both active low. You may need to invert some of these signals, by moving a jumper, or taking a signal before it is inverted. Always check OSI's documentation against another source, or with your own probe. Remember to start off the 2716's at 1 MHz. They may not work at 2 MHz on the 6ØØ board until a few changes are made to the enable lines.

Still nothing? You can start checking signals. First and foremost is the low reset at pin 4Ø of the CPU. Press reset, a low pulse should then appear at pin 4Ø. To run, this pin must return to a high. The op-amp doing this on the REV-D Superboard/C1P has been known to quit.

After this, things get more serious. Check for a clock signal on pin 37. If your probe indicates a pulsed signal, it will likely be fine. Check too for the phase 2 clock on pin 39. This is the clock output from the CPU.

Check now for action on the address lines. On a reset the processor will get $FF page, so there should be a lot of activity on the address bus. An inactive line could be shorted, or loaded by some defect in the computer.

You need a monitor ROM to boot. With OSI's SYN6ØØ you also need BASIC 4, as the print routines there are used. Monitors like the C1E have a built in print routine, and can at least get to the ML monitor, to let you look around. As a matter of fact, long ago, my BASIC 4 ROM did pack up. It did run for a few seconds when it was cold. A plastic bag of ice (dangerous!) confirmed the problem, as it allowed the computer to run for a few minutes.

At this point, you had better start to follow the schematics, and try to isolate the problem. I have spoken to someone who bought one of the $1Ø.ØØ Superboards. In desperation he used the PIA from a running computer connected to the address bus to look through the memory map. His problem turned out to be a shorted trace, so that one part of the ROM was repeated at other locations. Hopefully, you will spot your problem by eye.

Another possibility, at least as likely as a bad chip, are defective sockets. If you have removed/replaced a chip a number of times on the OSI 6ØØ board, cast a jaundiced eye toward the socket. They are of marginal quality, and are well known for their troubles.

Remember to go over any of your recent changes or fixes.

I know you do them perfectly the first time, but it never hurts to check it out. Then start through the trouble-shooting areas I have indicated. I must say that so far, I have always finally traced the problem, sometimes after a few nights of fruitless hunting. My hat goes off to someone I know who zapped his machine with 11Ø volts. His repair of the Superboard qualifies as a resurrection. He replaced 2 dozen TTL chips, and the CPU. Interestingly, his 2114 RAMS, being "delicate" MOS chips, survived. Just shows how tough these Superboards are! Of course, I must tip another hat to the inventive person mentioned above who used the lines from a parallel port to check through the address decoding. It just goes to show you what OSI hackers are made of. You certainly can't say that we didn't learn anything over the last few years.

★

SIMPLIFIED 5 1/4" HEAD UNLOAD

By: Ray Osborn
9a, Nairn Road
Rotorua, NZ

I implemented Dave Pompea's disk switch (from an old Aardvark) about a year ago on my dual-drive C4. It wasn't all that easy as I didn't have SAMS or an MPI schematic, but it worked. I have now simplified the logic.

Firstly, I don't believe in chopping up complex boards if it can be avoided so the only tracks cut are on the A13, and 1 on the 5Ø5. Secondly, my schematic only shows detail where it differs from that in the July 85 PEEK.

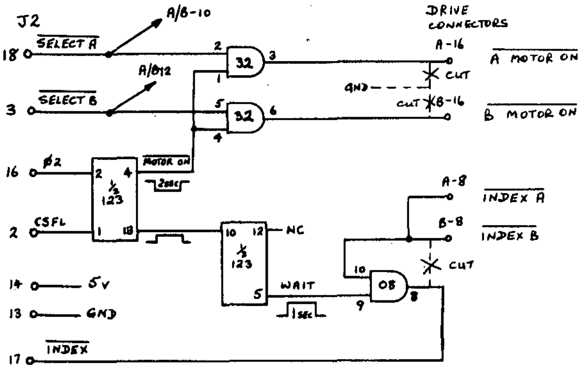
1. On the 5Ø5 find a convenient pad connected to U1A pin 22 (CSFL) and jumper to J2 pin 2 (unused). CSFL also seems to work on U1F pin 13.

2. Find another connected to U1A pin 25(Ø2) and jumper to J2 pin 16, also unused.

3. Make a track cut on the 5Ø5 from J2/2 to U4A pin 4 which appears to be unused by OSI.

4. Have a look at the A13 and cut tracks so as to:

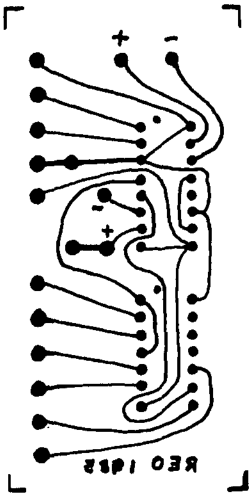
(a) Separate A and B drive connector pins 16 from each other and from ground (J2-13). Do this so that J2-3 remains connected to A and B drive pins 1Ø, and J2-18 remains connected to A and B connector pins 12. Without schematics I'm not sure what this does except perhaps to control logic other than motor on in the drives?



(b) Isolate A and B connector pins 8 from ground but not each other.

5. Make up PCB with additional components, about 1" x 2 1/2" is enough.

6. Insulate both sides of board with acetate sheet, run 1Ø jumpers to the A13 and tuck the board in between the 5Ø5 and A13.



I also used smaller capacitors and larger resistors as they take up less space. 6.3 volt 68mF tantalum with 68K for 2 secs and 33K for 1 sec.

That's all there is and it works like a charm. Hope it's useful.

★

★

V3.3 BUG REVISITED

We let you and author Paul Chidley down last month. Paul's article about the patches won't do you much good without the patches, so, belatedly, here they are!

```
 10 0000          ; SOURCE FOR 65D V3.3 PATCH 5" VERSION
 20 0000          ;
 30 0000           PAGEO  = $0000
 40 0000           MEMLO  = $00FE
 50 0000           MEMHI  = $00FF
 60 0000           SECTNM = $265E
 70 0000           TENMS  = $2678
 80 0000           SETTK  = $26BC
 90 0000           READDK = $2967
100 0000           CALL   = $2B11
110 0000           VIDSIZ = $DE00
120 0000           PIA    = $F700
130 0000          ;
140 2E79                  * = $2E79       ; SCRATCH BUFFER USED BY I/O
150 2E79          ;
160 2E79          ; V3PTH5 - OS65D V3.3 PATCH 5" VERSION
170 2E79          ;
180 2E79 EE 5E 26  V3PTH5 INC SECTNM      ; INCREMENT SECTOR #
190 2E7C A9 06            LDA #$06        ;
200 2E7E 20 BC 26         JSR SETTK       ; MOVE HEAD TO TRACK #6
210 2E81 20 B8 2E         JSR V3READ      ; GOTO $2EB8
220 2E84          ;
230 2E84 A9 34            LDA #$34        ; SET A = $34
240 2E86 8D 01 F7         STA PIA+1       ; STORE IN PIA AT $F701
250 2E89          ;
260 2E89          ; TPATCH - TIME DELAY PATCH (SEE TENMS & DELAY)
270 2E89          ;          USES A READ FROM $DE00 ON THE 540 BOARD
280 2E89          ;
290 2E89 A2 00     TPATCH LDX #$00        ; X=0
300 2E8B A0 00            LDY #$00        ; Y=0
310 2E8D E8        S1     INX             ; INCREMENT X
320 2E8E F0 20            BEQ SET.T       ; IF 0 THEN GOTO SET.T
330 2E90 C8        S2     INY             ; INCREMENT Y
340 2E91 F0 FA            BEQ S1          ; IF 0 THEN GOTO S1
350 2E93 AD 00 DE         LDA VIDSIZ      ; LOAD A FROM $DE00
360 2E96 30 FB            BMI S2          ; IF BIT#7 = 1 THEN GOTO S2
370 2E98 A0 01            LDY #$01        ; Y=1
380 2E9A AD 00 DE  S3     LDA VIDSIZ      ; LOAD A FROM $DE00
390 2E9D 10 FB            BPL S3          ; IF BIT#7 = 0 THEN GOTO S3
400 2E9F C8        S4     INY             ; INCREMENT Y
410 2EA0 F0 0E            BEQ SET.T       ; IF OTHEN GOTO SET.T
420 2EA2 A2 1F            LDX #$1F        ; X=$1F
430 2EA4 CA        S5     DEX             ; DEX
440 2EA5 D0 FD            BNE S5          ; IF <> 0 THEN GOTO S5
```