NOTE: All mods involving grounding should be 20 gauge solid, insulated wire. All mods involving signal paths should be no greater than 26 gauge and no less that 30 gauge insulated wire.

Sol

REV D to E

The following is a complete list of changes necessary to modify a Revision D Sol to a Revision E Sol. These changes are illustrated in Figures A & B for your convenience:

1. CLOCK WIDTH FIX:

On component side of Sol PCB, cut the trace between jumpers D and E (U90 and U91) of the clock generator. On the solder side of the PCB, connect a jumper from pin E to the feedthrough which leads to U91 pin 5. This brings PHASE 1 into 8080 spec at 140 ns rather than 70 ns without this mod.

2. PHANTOM GLITCH FIX:

On the solder side of the PCB, connect a jumper from U76 pin 4 to the feedthrough immediately below U76 pin 1. This assures the Sol will always power up with 4 PHANTOM cycles.

3. GROUND NOISE FIX:

On the solder side of the PCB, jumper pin 8 of U33, U50, U68 and U81 to the ground feedthrough leading to C45. Use 20 gauge insulated solid wire and keep the leads as short as possible. The fix shortens the return path to ground from the bus drivers. On occasion the present return path can be quite noisy.

4. PROTECT FIX:

On the solder side of the PCB, connect a jumper from the ground side of C11 to pin 70 of the 100-pin bus connector J11. This will ground the PROTECT line which is currently floating. Again, use 20 gauge insulated solid wire.

5. DMA/INTERRUPT UNSCRAMBLE:

On the component side of the PCB, cut the trace leading to pin 73 of J11 (S100 connector). Cut the trace leading to pin 1 of U45 on the solder side of the PCB. Also on the solder side, cut the second trace to the right of U64.

Jumper pin 73 of J11 to pin 1 of U45. Jumper from pin 28 of J11 to the feedthrough which formerly led to pin 73 of J11 before the trace was cut. Jumper the feedthrough directly below pin 1 of U45 to the feedthrough to the right of U64 pin 3. This fix will allow the Sol to be used with DMA devices. Helios II, and other DMA devices will not work without this fix.

6. MWRITE FIX:

On the solder side of the PCB, cut the trace leading to pin 7 of U93. On the solder side, cut the trace leading to the feedthrough immediately below pin 1 of U92. DO NOT CUT THE TRACE LEADING TO U92 PIN 1!! Connect a jumper from this feedthrough to pin 13 of U107.

On the solder side, jumper the feedthrough leading to U94 pin 9 to the trace which formerly led to U94 pin 7 before the trace cut. This fix decodes the MWRITE signal from SOUT and PWR even when a DMA device has disabled the Status Drivers. This is especially important to memory boards which require MWRITE and do not decode PWR and SOUT internally and DMA devices (such as Helios) which do not supply their own MWRITE to the bus.

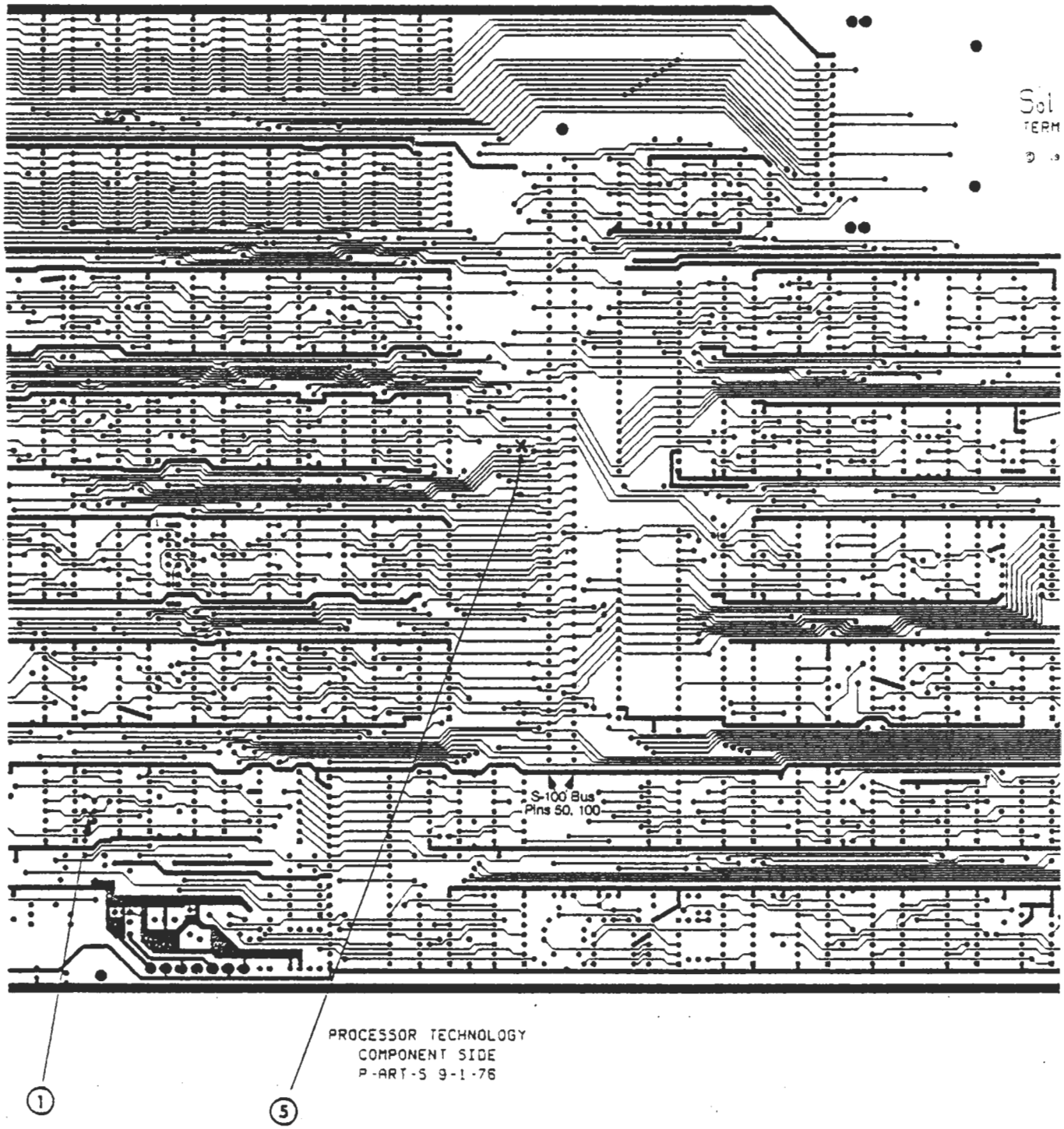
7. CURRENT LOOP FIX:

Cut the large trace which leads to R23 and R24 on the solder side of the PCB. Connect the now isolated end of R23 to the +12 Volt feedthrough as shown in Figure B.

8. CASSETTE RELAY FIX:

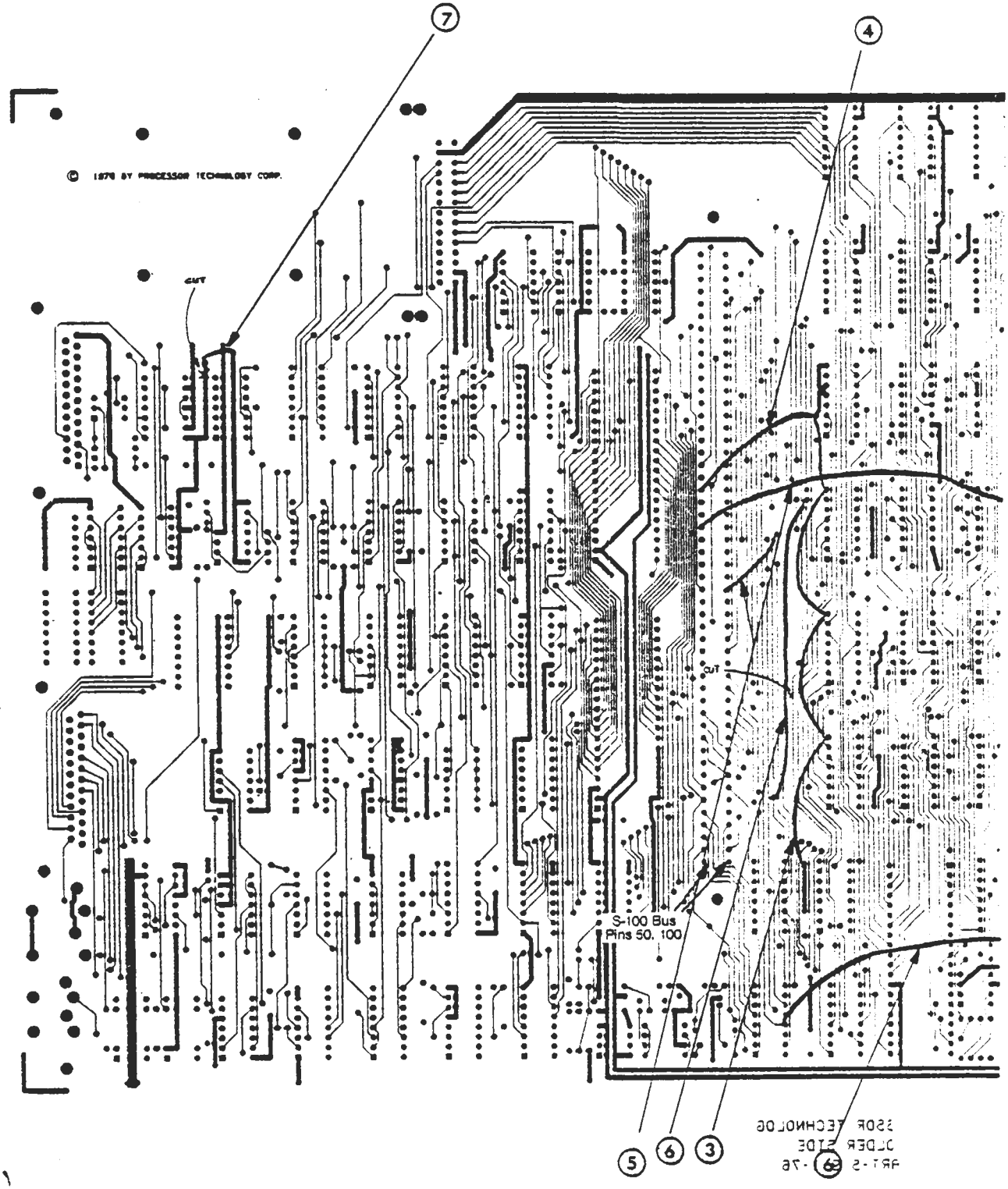
On component side of Sol PCB, cut the trace leading to the hot lead of J9. On the solder side connect a 6.8 ohm, 1/4 watt (5%) resistor between the hot pin of J9 and cathode (banded end) of D14. Also on the solder side, cut the trace leading to the hot pin of J8 to the feedthrough isolated by the trace cut. This change is adding a series resistance to the cassette drive jack to limit the amount of current drawn by the cassette motor from the reed relays. Certain models of cassettes might draw excessive current and burn out the relays without this fix.

NOTE: These changes were elaborated in Issues 2 and 3 of ACCESS. (April and June of 1977)



Sol-PC (Rev. D)—Component Side

Figure A



Sol-PC (Rev. D)—Solder Side
Figure B

REV E to F

The following is a list of changes to bring a Revision E Sol to a Revision F Sol:

1. Substitute a 270 ohm, 1/4 watt (5%) resistor for R21 (formerly a 470 ohm resistor).

This change is to increase the drive capability of the serial current loop and was elaborated in CHANGE NOTICE #10.

2. Change the value of VR3 from 50K pot to a 100K pot. This allows a wider range of variation when adjusting the VCO center frequency to account for part variation among vendors. Also, change the value of R154 from 100K to 47K.

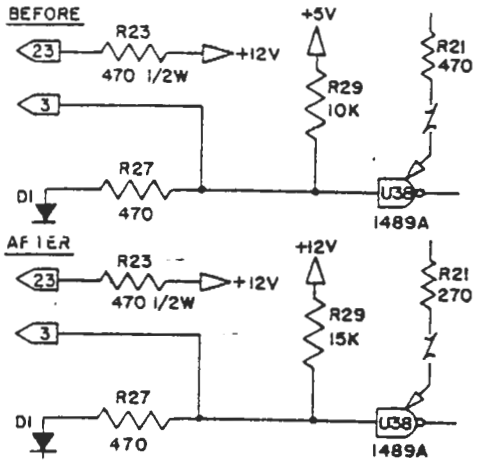
To reset this frequency, turn Sol on and measure pin 4 of U110. It should read 14.0 KHz or 71.4 usec. This change is advised in CHANGE NOTICE #11.

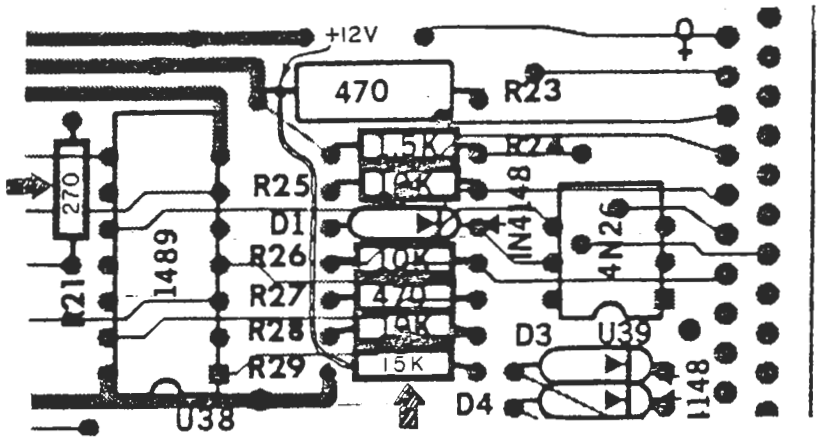
3. If your Sol has a revision B Regulator PCB, it may be necessary to perform a mod to the crowbar circuit to enhance its reliability. These changes may be found in CHANGE NOTICE #6-2, REV C.

REV F TO G

The following are the changes from a Revision F Sol to a Revision G Sol:

1. Substitute a 15K, 1/4 watt (5%) resistor for R29 (formerly a 10K resistor. Substitute a 270 ohm resistor in place of R21 (formerly a 470 ohm resistor). Install 1" length of tubing on R29 and bend lead near resistor body to form a 90-degree angle. With the PCB legend in the normal reading position, connect the angled end of R29 to the left hand lead of R23 (470 ohm). This connects the resistor to +12 Volts. See diagram below for details. This change was covered in CHANGE NOTICE #10 and is to increase the drive going into U38 of the serial current loop input.





REV G to REV H

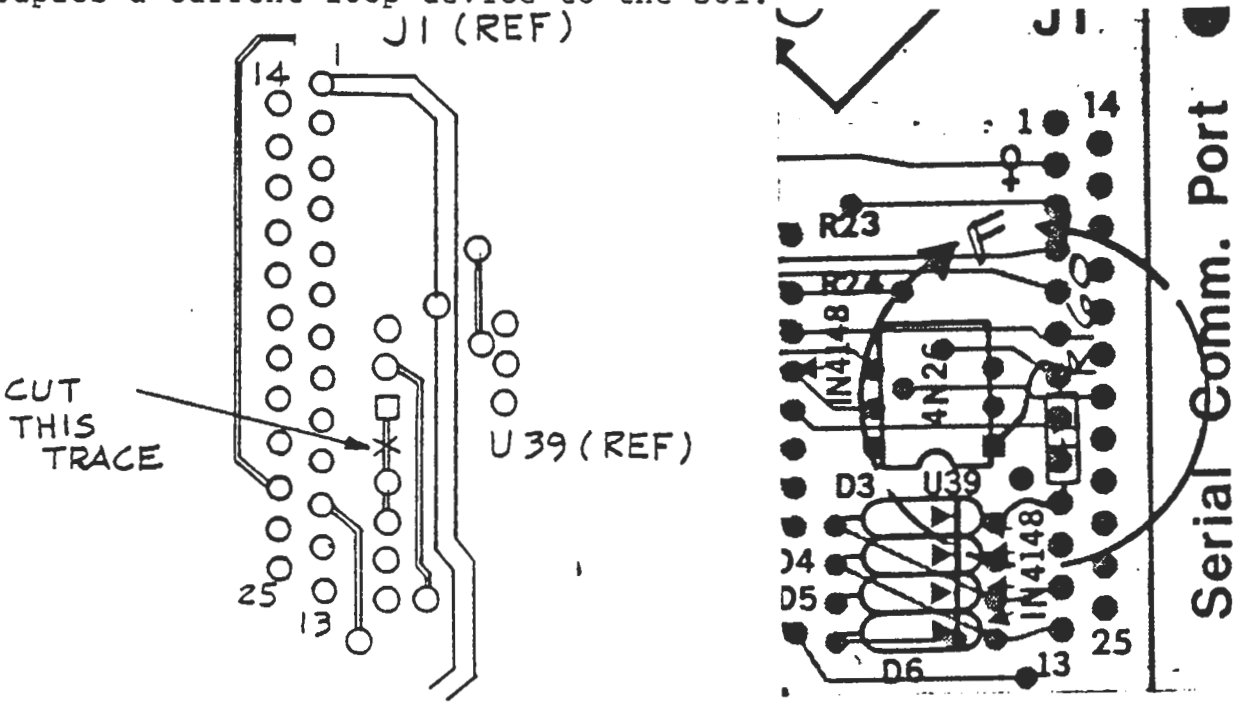
The following changes bring Revision G Sols to Revision H level:

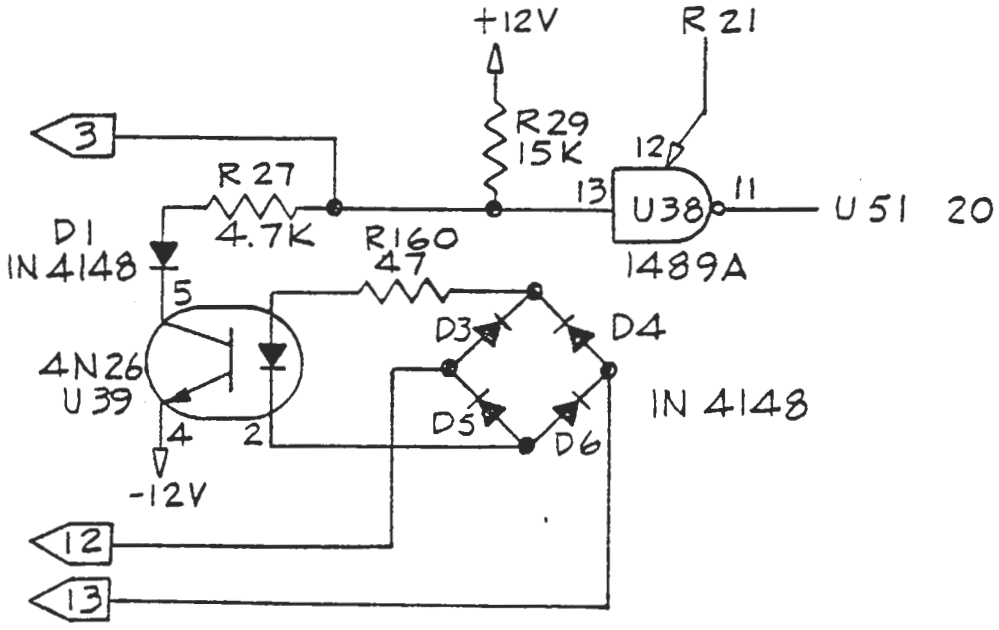
1. Change the value of R27 from 470 ohms to 4.7K ohms.

2. On the solder side of the PCB, cut the trace connecting U39 pin 1 to the cathode of D3. Check with an ohmeter to be sure the cut is complete.

3. Wrap one lead of R160 (47 ohm resistor) around pin 1 of U39 (4N26) and wrap the other end around the cathode end of D3 (band side of 1N4148). Solder both leads of R160 and check for possible shorts.

This change was to enhance the reliability of the opto-isolator which couples a current loop device to the Sol.





ProcessorTechnology

Processor Technology Corporation
7100 Johnson Industrial Drive
Pleasanton, CA 94566
(415) 829-2600
Cable Address - PROCTEC

Sol Update 731049

This Update describes a variety of changes to the Sol that are being incorporated into current factory production. Each of the eight headings below includes a title which is a brief functional description of the change or set of changes, followed by the revision level of the affected assembly before the change and after the change. The revision levels are identified by the capital letters of the alphabet, skipping the letters "I, O, Q, X." The revision level of PC boards is marked on the component side of the board.

Each section begins with a brief explanation of the change. Hardware changes that should be made to units of an earlier revision in the field are specified in this Update by the sub-headings "HARDWARE CHANGES." Changes to the Sol Systems Manual are specified by the sub-headings "Changes to Documentation."

The hardware changes in this Update take a Sol-PC (102000) at a REV H and bring it to a REV R, which is the current level of this assembly being shipped from the factory. After making the last set of changes, you will be instructed to mark the resulting revision level on the Sol-PC. (Not all the changes in this Update concern the Sol-PC.) Instructions for bringing a Sol-PC from G to H are contained in "Sol Update 731011." Supplementary instructions are contained in the Service and Maintenance Manual, page 4-22.

The Sol Systems Manual referred to in this Update is P/N 730000, Fourth Printing, February, 1978. This edition documents the Sol-PC at the H level.

Change 1: Jumpers on Sol-PC, P/N 102000 (H to J)

This is a documentation change only. Sol-PC's are manufactured with jumpers at locations A-B, D-E, F-G, I-J, N-P, and between U107-13 and a feedthrough pad near U105-1 as shown in detail G in a replacement page, Figure X-6, REV A. The new Fig. X-5, which is also supplied with this Update, shows the jumpers in place.

Changes to the Documentation

1) Replace the corresponding obsolete pages with the following replacement pages which are included in this Update:

   Fig. X-5, "Sol-PC Assembly," Sheet 2 of 2, REV A (102000-R)

   Fig. X-6, "Sol-PC Assembly," Sheet 1 of 2, REV A (202000-R)

   Fig. X-14, "Sol-PC Schematic, CPU and Bus," Sheet 1 of 5, REV A (1002002-L)

2) It is recommended that you transfer the pages of the earlier revision level to a separate binder labeled "History Copy." Because this is a documentation change only, the resulting REV level J need not be marked on the Sol-PC assembly. Intervening revision levels which have no impact on the Sol assemblies may not be mentioned in this and other Updates.

Change 2: Replacement of IC at U48 of Sol-PC (K to M)

This change covers the subject that was scheduled for Update 731012; no Update 731012 will be published.

This change modifies the Sol-PC to compensate for switching transition noise produced on the Sol PRDY line by the ParaSol Debugger. This change is necessary only on master and slave Sols used with the ParaSol Debugger; however, in order to have succeeding revision levels track without confusion, and to prepare all units for testing by the ParaSol Debugger, all Sol-PC's should be modified. All Sol-PC's of a revision level of M or higher have this modification.

Changes to the Documentation

Changes on the schematic and the assembly drawing are reflected in the replacement pages Fig. X-14, REV A and Fig. X-5, REV A, both of which are supplied with this Update.

In the parts list on page III-2, REV A, change the quantity of the 74LS00 from 3 to 2 and delete the designator U48. Enter the 74LS132 at the bottom of page III-2, REV A as follows:

PART: IC DESCRIPTION: 74LS132, Quad 2-input nandgate QTY: 1
Drawing No. X-5 Designator: U48

HARDWARE CHANGES

1) Replace the 74LS00 at U48 on the Sol-PC with a 74LS132 (P/N 701124).

This change raises the revision level of the Sol-PC from K to M.

Change 3: Shortened Screws in Expansion Subchassis

This change shortens two #10 wood side mounting screws so that the head of the rear screw does not obstruct the insertion of S-100 modules into the backplane.

Changes to Documentation

The two screws are item 41 on Fig. X-9, "Sol Top Assembly." Mark the change on your copy of Fig. X-9. In the parts list on page I-12, REV A, change the quantity of Machine Screw, 10-24 x 3/8 from 8 to 10.

On page I-12 REV A, delete "720079 Machine Screw, 10-24 x 1, minimum quantity 2."

HARDWARE CHANGES

1) Remove all S-100 modules from the backplane.

2) In the Sol Expansion Subchassis, as shown in Fig. X-9, "Sol Top Assembly, Sheet 2 of 3, Detail C," locate the 10-24 x 1" machine screw at the rear of the expansion chassis. This screw is directly at the rear of the lowest card guide. This is the screw that should be replaced. The other screw at the front of the expansion chassis need not be replaced.

3) Remove the screw. Taking precaution against having metal chips scatter over the Sol-PC and inside the Sol chassis, drill out the hole for the screw in the outer wall of the fabrication to 0.375 inch.

4) Vacuum or blow out any metal chips that may have been created.

5) Replace item 41 with a 10-24 x 3/8 machine screw.

As a result of this change, the Sol Top Assembly (101000-XX) becomes a REV level M.

Change 4: Extended Cassette BASIC

Extended Cassette BASIC is now shipped with the Sol 20 and Sol cassette based systems instead of BASIC 5. SOLOS/CUTER User's Manual is shipped with each Sol 20 (Order #400410). Refer to the Sol User's Manual, 1st edition, P/N 730021, (the new user's manual) for a current list of order configurations with order/part numbers.

Note that different capacity memory modules may be shipped with a Sol or Sol system to make up the total capacity required by the order.

Change 5: Resistor Value Changed on Sol-PC (M to N)

This changes lowers the value of R145 on the Sol-PC from 10K ohms to 47 ohms. As shown on the schematic, Fig. X-18, "Audio Tape I/O," R145 is part of a resistor network in the hysteresis comparator. Lowering the value of R145 to 47 ohms adjusts the cassette data interface circuitry for standardized testing whereby the audio out J6 is looped back to the audio in J7 during the test program.

Changes to Documentation

On the schematic, Fig. X-18, near J-7, Audio-In, change the value from R145 from 10K ohms to 47 ohms.

In the parts list, on page III-4, REV A, increase the quantity of the 47 ohm resistor from 3 to 4, and add the reference designator "145".

On page III-5, REV A of the parts list, reduce the quantity of the 10K ohm resistor from 31 to 30 and delete the reference designator "145".

HARDWARE CHANGES

1) Locate R145. On Fig. X-5, it is located in area A4.

2) Replace R145, 10K ohms, with a 47 ohm resistor of the same type. If the Sol-PC was removed for this change, leave it out for the next change.

This modification changes the revision level of the Sol-PC from M to N.

Change 6: Data Bus Timing on Sol-PC (N to P)

This changes resolves a potential data bus timing conflict during read cycles. The change consists in disconnecting the output of U45-10 from the input of U47-5 and jumpering J11, pin 97 ($\overline{SWO}$) to the input of U47-5. Refer to the schematic X-14 REV A, included in this Update.

HARDWARE CHANGES

Refer to Fig. X-6, REV A, "Sol-PC Assembly," Sheet 2, Detail H.

1) On the component side of the board, cut the trace that connects to U47-pin 5, close to where it passes by D7, as shown.

2) Using 30 AWG kynar insulated wire, install a jumper on the solder side of the board between U47-pin 5 and U107-pin 5.

This change revises the Sol-PC from an N REV level to a P. (The letter "O" is not used, to avoid confusion with the number "0".)

Change 7: Phasing Out Top Edge Connector on Sol-PC

Sol backplane assemblies (103000) will no longer have an extra backplane connector on the top of the backplane.

Changes to the documentation

Make a note on Fig. X-8, Sol Top Assembly, detail E, item 5.

In the parts list, on page VI-2, REV C, delete the 100-pin PC connector, designator 5.

No hardware change to existing units is needed.

Change 8: Polarizing Sol-PC/Keyboard Cable Assembly
(Sol-PC, 102000 P to R)

All units should have this modification to prevent potential damage

to the video display circuitry. This change polarizes the Sol-PC/Keyboard cable assembly by removing pin 20 on the Sol-PC header J3 and the Keyboard Assembly header J1 and inserting polarizing keys in the matching connector jacks at both ends of the cable. The same change is made to header J2 on the ParaSol Debugger PCB.

Changes to the Documentation

Note 7 has been added to Fig. X-5, REV A, "Sol-PC Assembly," Sheet 2, and Fig. X-6, REV A, "Sol-PC Assembly," Sheet 1.

In the parts list, on page VI-3, REV C, add to the flat 20-wire cable, designator 23, the description "Sol-PC/Keyboard Cable. After the last entry, add "Polarizing Key...For Sol-PC/Keyboard Cable (P/N 717062)--quantity: 2."

The part number of the polarized cable is 110380.

In the ParaSol Debugger Kit User's Manual, P/N 730018, add the following note to the assembly drawing of the ParaSol Debugger PCB, Fig 6-1: "Remove pin 20 from header J2; mates with polarized Sol-PC/Keyboard cable assembly 110380."

HARDWARE CHANGES

1) Remove pin 20 from header J3 on the Sol-PC.

2) Remove pin 20 from header J1 on the Sol Keyboard Assembly.

3) Remove pin 20 from header J2 on the ParaSol Debugger PCB Assembly.

4) Insert polarizing keys in connector pin-jack 20 at both ends of the Sol-PC/Keyboard Cable. These keys are available from the Customer Service Department of Processor Technology.

5) If the Sol-PC was at revision level H when the modifications in this Update were begun, mark the REV level "R" in the upper right-hand corner of the Sol-PC, oriented as in Fig. X-5; referring to Note 4. (Revision letter "Q" is skipped.)

6) Mark the REV level "C" on the right hand side of the ParaSol Debugger PCB Assembly, 900043. (The previous level documented by the ParaSol Debugger Kit User's Manual was B, despite the marking on Fig. 6-1.)

* Internal documentation corresponding to the eight changes above is listed below by heading number.

1) ECN 10286
2) ECN 10371
3) ECN 10403
4) ECN 10408
5) ECN 10416
6) ECN 10425
7) ECN 10440
8) ECN 10443

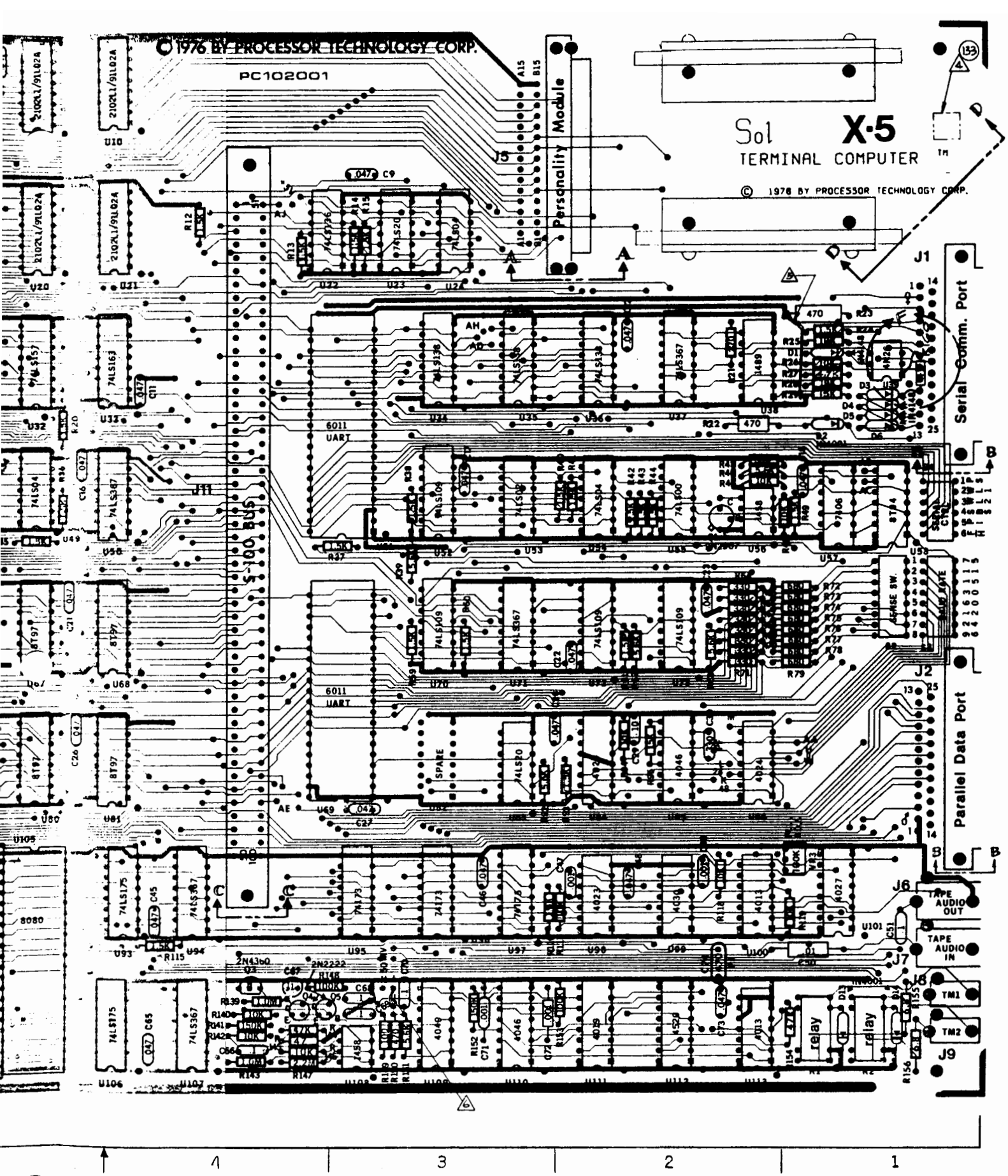
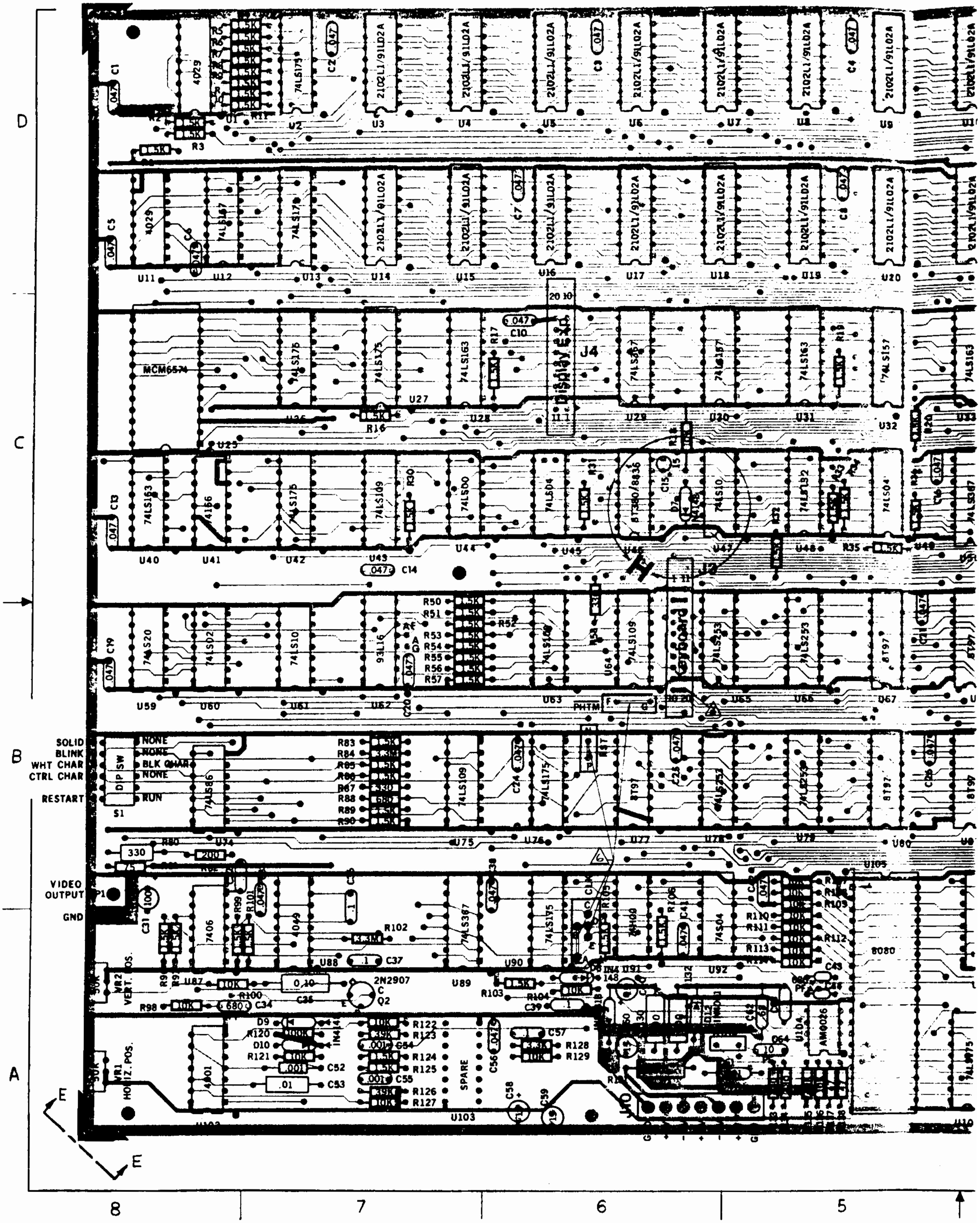
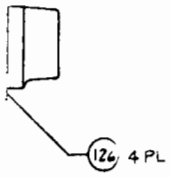


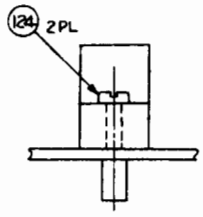
Fig. X-5. Sol-PC Assembly, Sheet 2 of 2 (102000-R)



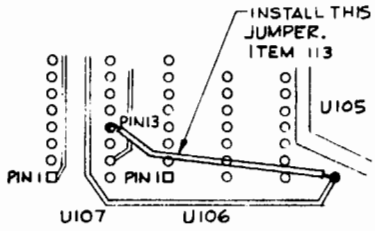
D
C
B
A
8
7
6
5
E
E
SOLID
BLINK
WHT CHAR
CTRL CHAR
RESTART
NONE
NONE
BLK CHAR
NONE
RUN
DIP SW
S1
VIDEO OUTPUT
GND
P1
VR1 HORIZ. POS.
VR2 VERT. POS.
MCM6574
4029
74LS175
74LS157
2102L1/91L02A
74LS163
74LS109
74LS00
74LS04
74LS10
74LS20
74LS02
93L16
74LS253
74LS386
74LS367
74LS132
8T97
8T380/8836
7406
4049
4001
74LS387
8080
AM6026
Display Exp.
J4
J2
Keyboard
2N2907
Q2
SPARE
U103



3
J·1 AND J·2 ⚠2



DETAIL C
MOUNTING OF BACKPLANE CONNECTOR (J·11) ⚠2



DETAIL G (SOLDER SIDE)
APPLIES ONLY TO FAB BOARDS – REV. E

NOTES:

⚠1 INSERT CARD GUIDE WITH OPEN END AWAY FROM J·5 CONNECTOR.

⚠2 MOUNT CONNECTORS TO BOARD BEFORE SOLDERING.

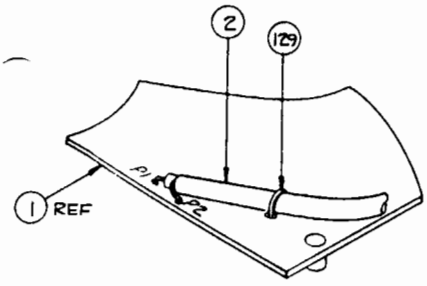
3. R91 IS NOT USED.

⚠4 MARK WITH REVISION LETTER IN APPROXIMATE POSITION SHOWN.

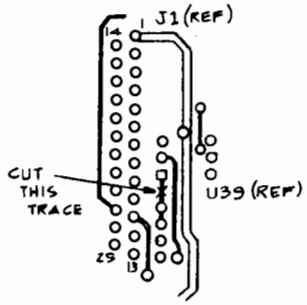
⚠5 DISCONNECT LEFT LEAD OF R29 AND SOLDER JUMPER WIRE FROM LEFT LEAD OF R29 TO LEFT LEAD OF R23 AS SHOWN. USE ITEM 113.

⚠6 INSTALL JUMPER, ITEM 112, 5 LOCATONS: A-B, D-E, F-G, I-J, N-P

⚠7 REMOVE PIN 20 FROM HEADER PRIOR TO INSTALLING HEADER ON BOARD



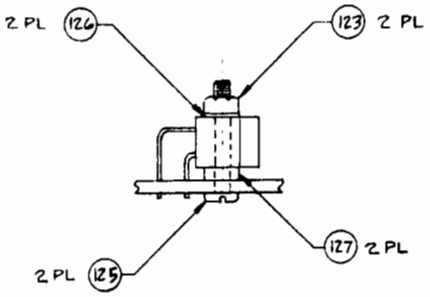
DETAIL E
MOUNTING OF VIDEO CABLE



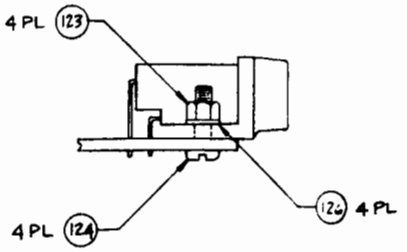
DETAIL F
VIEWED FROM THE SOLDER SIDE

REV A

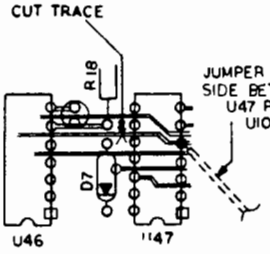
Fig. X-6. Sol-PC Assembly, Sheet 1 of 2 (102000-R)



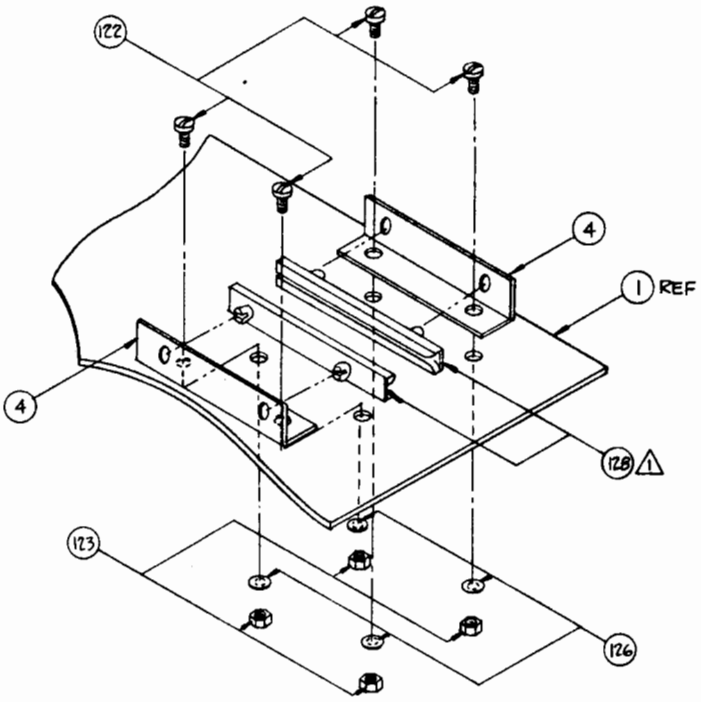
DETAIL A
MOUNTING OF J-5 ⚠2



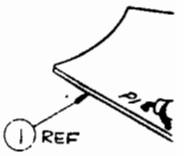
DETAIL B
MOUNTING OF J-1 AND J-2 ⚠2



DETAIL H
VIEWED FROM COMPONENT SIDE



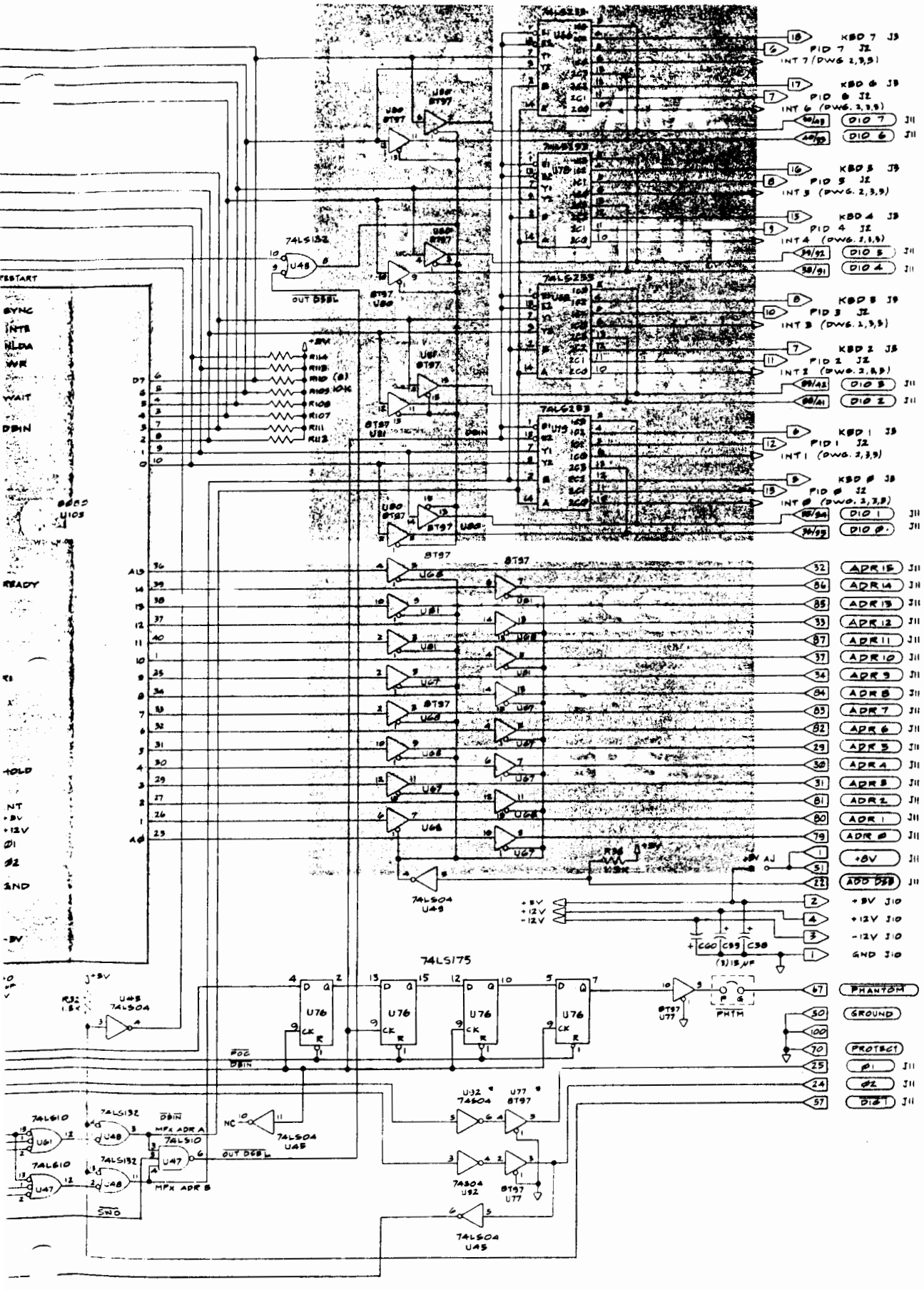
DETAIL D
PERSONALITY MODULE CARD GUIDE ASSY.



MOUNT

CUT
THIS
TRA

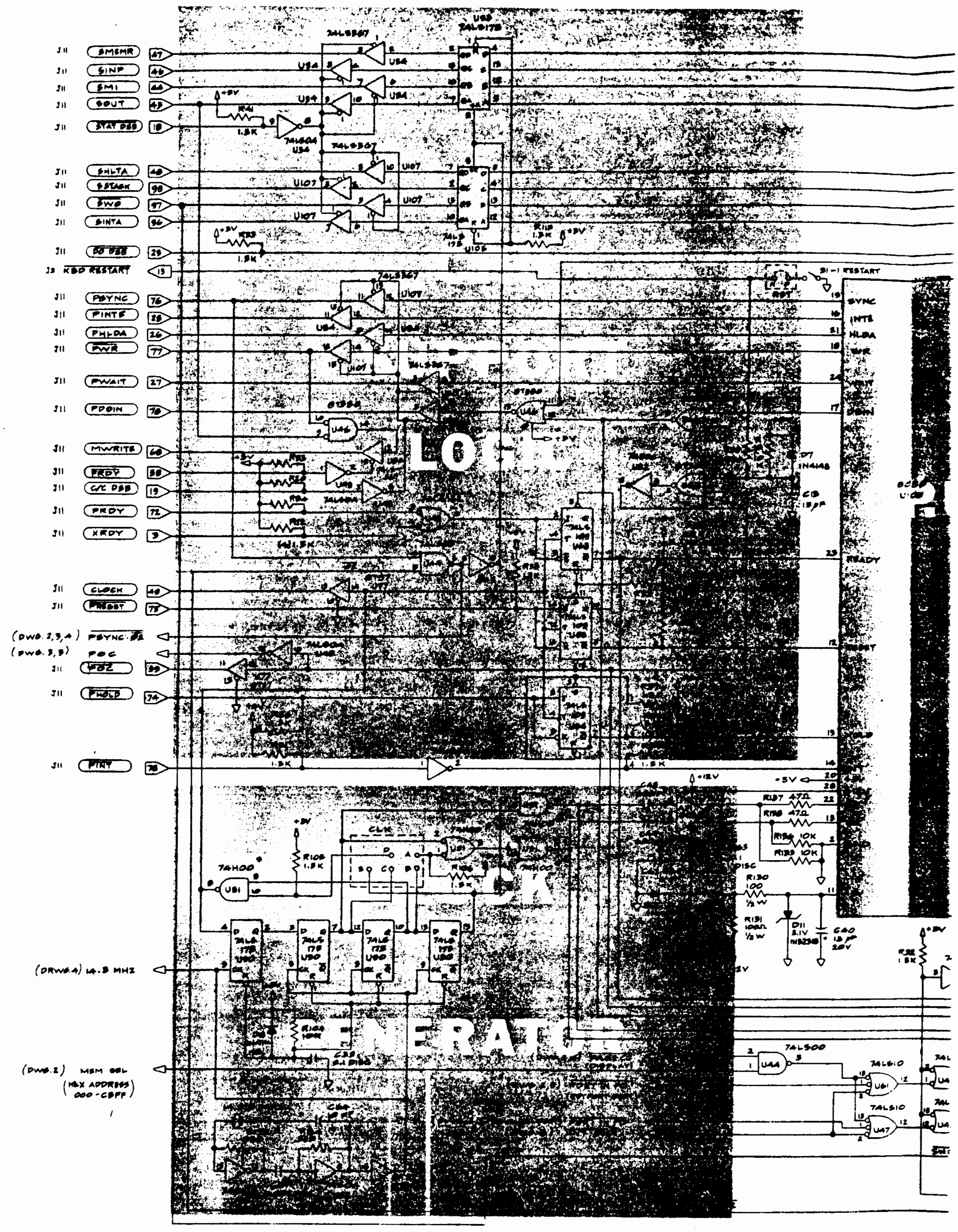
VIE



REV A

102002 L

Fig. X-14. Sol-PC Schematic, CPU and Bus, Sheet 1 of 5 (102002-L)



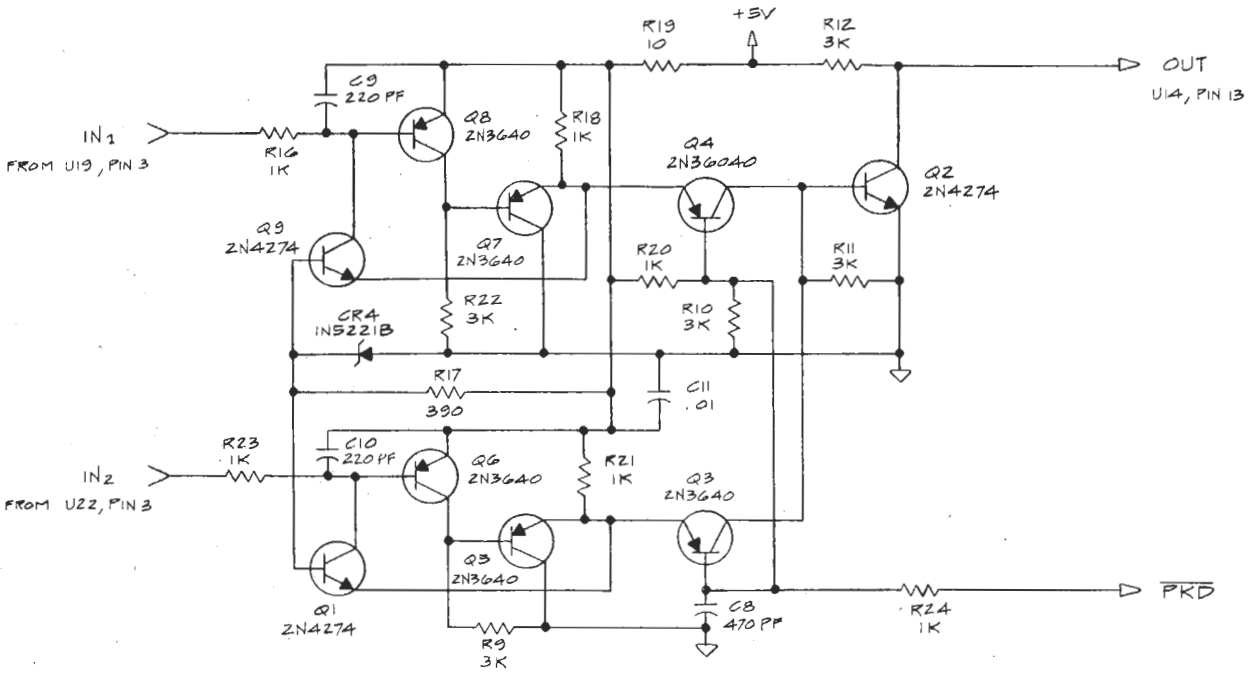
LOGIC
CK
GENERATOR
SMEMR
SINP
SM1
SOUT
STAT DSB
SHLTA
SSTACK
SWO
SINTA
DO DSB
KBD RESTART
PSYNC
PINTE
PHLDA
PWR
PWAIT
PDBIN
MWRITE
PRDY
C/C DSB
PRDY
XRDY
CLOCK
PRESET
(DWG. 2,3,4) PSYNC·Ø2
(DWG. 3,5) POC
PØ2
PHOLD
PINT
(DRWG.4) 14.3 MHZ
(DWG.2) MEM SEL
(HEX ADDRESS 000-CBFF)
S1-1 RESTART
SYNC
INTE
HLDA
WR
DBIN
READY
RST
+5V
+12V
74LS367
74LS04
74LS175
74H00
74LS00
74LS10
D7 1N4148
D11 5.1V 1N5231B

Processor Technology

Sol MANUAL ADDENDUM #1

Reference Section X, Drawing X-23.

A block function labelled "K.T.C." is shown between U-19 and U22, and U14. This block contains the Capacitive Switch Detector Circuit. The parts constituting this circuit are listed, and the assembly covered in Section V of this manual. The theory of operation is covered in Section VIII. At the time of publication of this manual, operation of this circuit was proprietary information, but has now been released. The schematic is shown below. Note on the schematic X-23 that this detail is shown here on this page.



PROCESSOR TECHNOLOGY CORPORATION 6200 HOLLIS STREET EMERYVILLE CA 94608 (415) 652-8080

Sol MANUAL ADDENDUM #3

Processor Technology

Cassette Recorders for use with Sol

Not all audio cassette recorders are suited for data storage use with the Sol. The following models have been tested and approved by Processor Technology:

1. Panasonic RQ-413AS
2. Realistic CTR-21

Some users have reported unsuccessful results with the Panasonic RQ-309 and the J. C. Penny Catalog #851-0018. If you should wish to select a different model, the following features, included on the models above, are necessary:

1. An AUX input. Although the Sol can be jumpered for low level Microphone level input, the procedure is no longer recommended.
2. A digital counter. The counter is necessary in locating programs on the cassette.
3. A tone control. The existence of a tone control is one indication of high quality electronics.

Even though a recorder has the three features, there is no guarantee that it will work properly for the purpose. Recorders vary greatly in the quality of their electronics. If possible, test the recorder with a long file before purchasing it, in both record (SAVE) and playback (GET or XEQ) mode. If the recorder is not working properly, either you will get an error message, or you will find differences between what was recorded and what was played back.

Observe the following pointers for best results:

1. Keep the recorder at least a foot away from the Sol, or other equipment which can generate magnetic fields. The recorder can pick up hum which may generate errors.
2. Keep the tape heads cleaned and demagnetized in accordance with the manufacturer's instructions.
3. Use high quality brand-name tape. Cheap tape can wear down the tape heads and give erratic results.
4. Bulk erase tapes before using.
5. Keep the cassettes in their protective plastic covers, in a cool place, when not in use. Cassettes are vulnerable to dirt, high temperatures, liquids, and physical abuse.
6. Set the tone control at midrange, and set the volume control about 2/3 full volume. The Sol has an automatic gain control circuit which compensates for a wide range of levels, but operation in the middle of this range will

give the most reliable results. Experiment to find the best setting of volume and tone controls.

7. On some cassette recorders, the microphone can be live while recording through the AUX input. Deactivate the mike in accordance with the manufacturer's instructions. In some cases this can be done by inserting a dummy plug into the microphone jack.

8. During recording, some recorders present the signal being recorded at the monitor or earphone output. In a system with two cassette recorders this could cause problems if an attempt was made to read from one recorder while the other was writing. Since both recorders share the same audio lines, the monitor output of the recorder which was recording could interfere with the signal being read from the other recorder.

9. If you record more than one file on a tape side, SAVE a special file, which could be named END, to let you know when you have played past the files of interest. After recording the last file on a side, rewind the tape, set the digital counter to zero, and issue a CATalog command (see SOLOS/CUTER User's Manual). As each file header is displayed, make a note of the reading on the digital counter, the exact name of the file, load address, and file length. Mark the cassette with this information to make file retrieval much easier.

If you experience a read error, use the following procedure to isolate the problem:

1. Check for proper settings, and make sure you have followed the pointers above.

2. Check cables for intermittant connections and shorts.

3. Note the exact reading of the digital counter at the time of the error.

4. Rewind the tape and try to read the same part of the tape again. If the tape reads without errors this time, the error was not recorded on the tape. If there is a read error at the same point, then the error is recorded on the tape.

5. Rewind the tape and record a file on the same part of the tape. Read the file. If the tape reads without errors, then the original read error was generated during the recording process. If there is still a read error at the same point, then the cassette itself is faulty.

Processor Technology

Sol MANUAL ERRATA NOTICE #3

1. Reference Section X, Drawings, Drawing X-17, Input/Output.

   In the Baud Rate Generator section of this schematic, the function of switch S3-7 is incorrectly shown as selecting 2400/4800 Baud, and S3-8 is incorrectly shown as selecting only 9600 Baud. Change the schematic to show that S3-7 selects 2400 Baud only, and that S3-8 selects 9600/4800 Baud. Draw a line connecting points L and M to indicate a jumper.

2. Reference Section VII, page VII-15, Table 7-2.

   In the Baud Rate column of this table, change "4800***" to read "9600***". Also, in the footnote with the triple asterisk, change the phrase "SDI operates at 9600 Baud..." to read "SDI operates at 4800 Baud..."

PROCESSOR TECHNOLOGY CORPORATION 6200 HOLLIS STREET EMERYVILLE CA 94608 (415) 652-8080

Processor Technology

ASSEMBLY PROCEDURE CHANGE NOTICE #6-2 Rev B

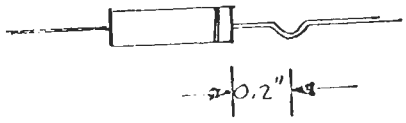
This Change Notice concerns the Sol-REG board and applies only to Revision Level B boards.

A problem was detected in early Sol-REG boards in which the "crow-bar" circuit would trigger without adequate cause and short circuit the 5-volt output. A circuit change has been made which will be reflected in Revision Level C and above boards to correct the problem. Revision Level B boards, however, require the following modification to correct the problem. Parts for this modification are supplied with your kit:

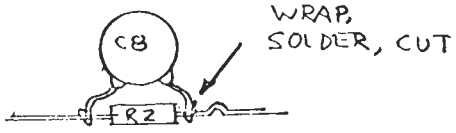
1) R2, 330 ohms, 1/4 watt, color code orange-orange-brown
2) R14, 100 ohms, 1/4 watt, color code brown-black-brown
3) D1, 1N5231B
4) C8, 0.047 uF disc ceramic

Assemble these parts as follows:

1. Form one lead of R2, R14, and the cathode (banded) lead of D1 for upright P.C. insertion as shown:

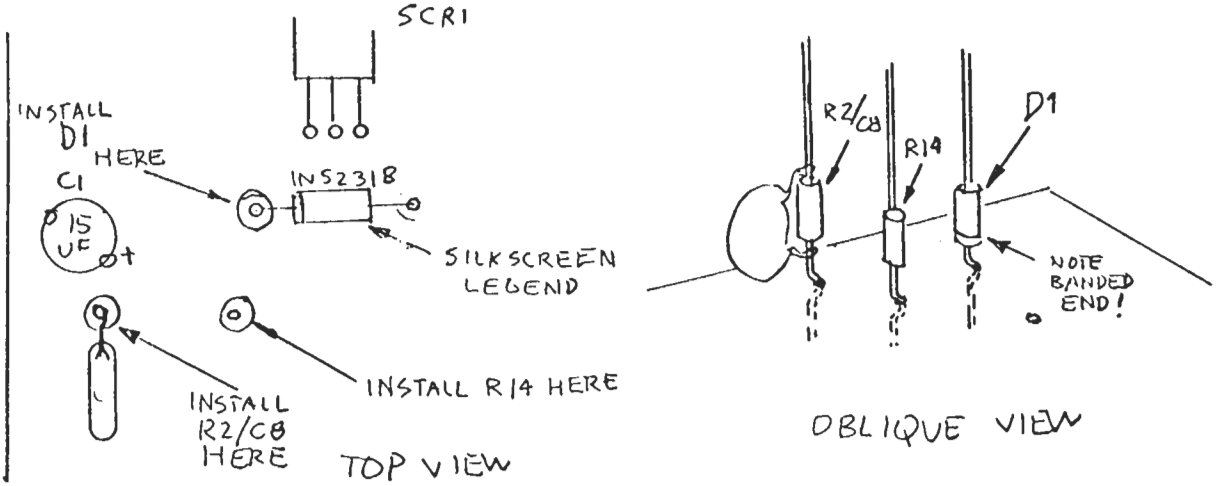


2. Solder C8 in parallel with R2 as shown:

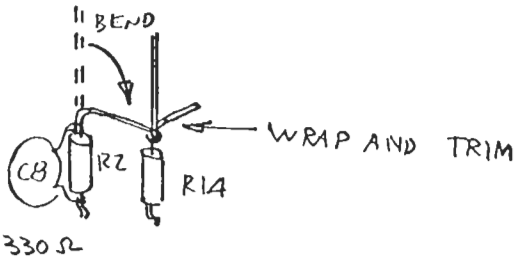


3. Install and solder R2-C8, R14, and D1 as shown below. Install the formed leads into the board with the unformed leads vertical. Position R2-C8 so that the body of C8 is parallel to the board edge and oriented away from C1.

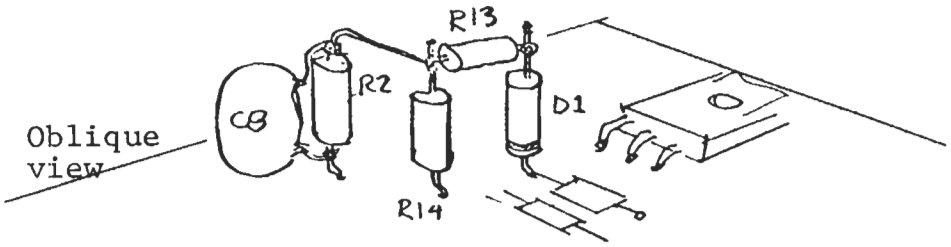
PROCESSOR TECHNOLOGY CORPORATION 6200 HOLLIS STREET EMERYVILLE CA 94608 (415) 652-8080



4. Bend the top lead of R2-C8 over towards R14, and bend it around the top lead of R14 one-eigth inch from the body of R14. Solder, and trim the excess lead of R2-C8 only.



5. Install R13 between the top leads of D1 and R14. Wrap R13's leads around R14 and D1 leads. Solder all connections at both points, and trim excess lead lengths. The resulting final configuration is shown below.



Schematic Diagram X-12 of the regulator includes these changes.

ProcessorTechnology

Processor Technology
Corporation

7100 Johnson Industrial Drive
Pleasanton CA 94566

(415) 829-2600
Cable Address PROCTEC

Sol MANUAL

ASSEMBLY PROCEDURE CHANGE NOTICE #6-2 REV C

Subjects: Crowbar Fix for Sol Rev B Regulator P.C. Board
Flat Washers in Final Cabinet Assembly

To enhance the reliability of the crowbar circuit which protects circuitry from overvoltage on the +5 volt supply, add the following additional parts, supplied with your kit, to Sol-Reg. Use the procedures given below after completing Step 13 of Section II, Sol Power Supply Assembly and Test. Make a note in your manual after Step 13 to remind you to do the additional steps below. Refer to the drawing on the next page as you add the parts to Sol-Reg.

Select the following additional parts from your kit:

1) R13, 330 ohms, 1/4 watt, color code orange-orange-brown
2) R14, 100 ohms, 1/4 watt, color code brown-black-brown
3) D5, 1N270
4) C8, .047 disc ceramic

( ) Step 1. Pass the two leads of C8 under the two leads of R2 (330 ohms) and bend around the leads of C8, close to the body of the resistor. Solder and trim the leads.

( ) Step 2. Wrap the leads of R14 around the right-hand leads of SCR1 and R2 (330 ohms), dressing the leads as shown in the drawing. Make sure the leads of R14 do not short to other leads of SCR1 or D1. Solder the lead to SCR1, but not the lead to R2 (330 ohms).

( ) Step 3. Wrap one lead of R13 (330 ohms) around the same right-hand lead of R2 (330 ohms), with the position of R13 parallel to D1 (1N5231B) as shown below. Solder the two leads which are wrapped around R2.

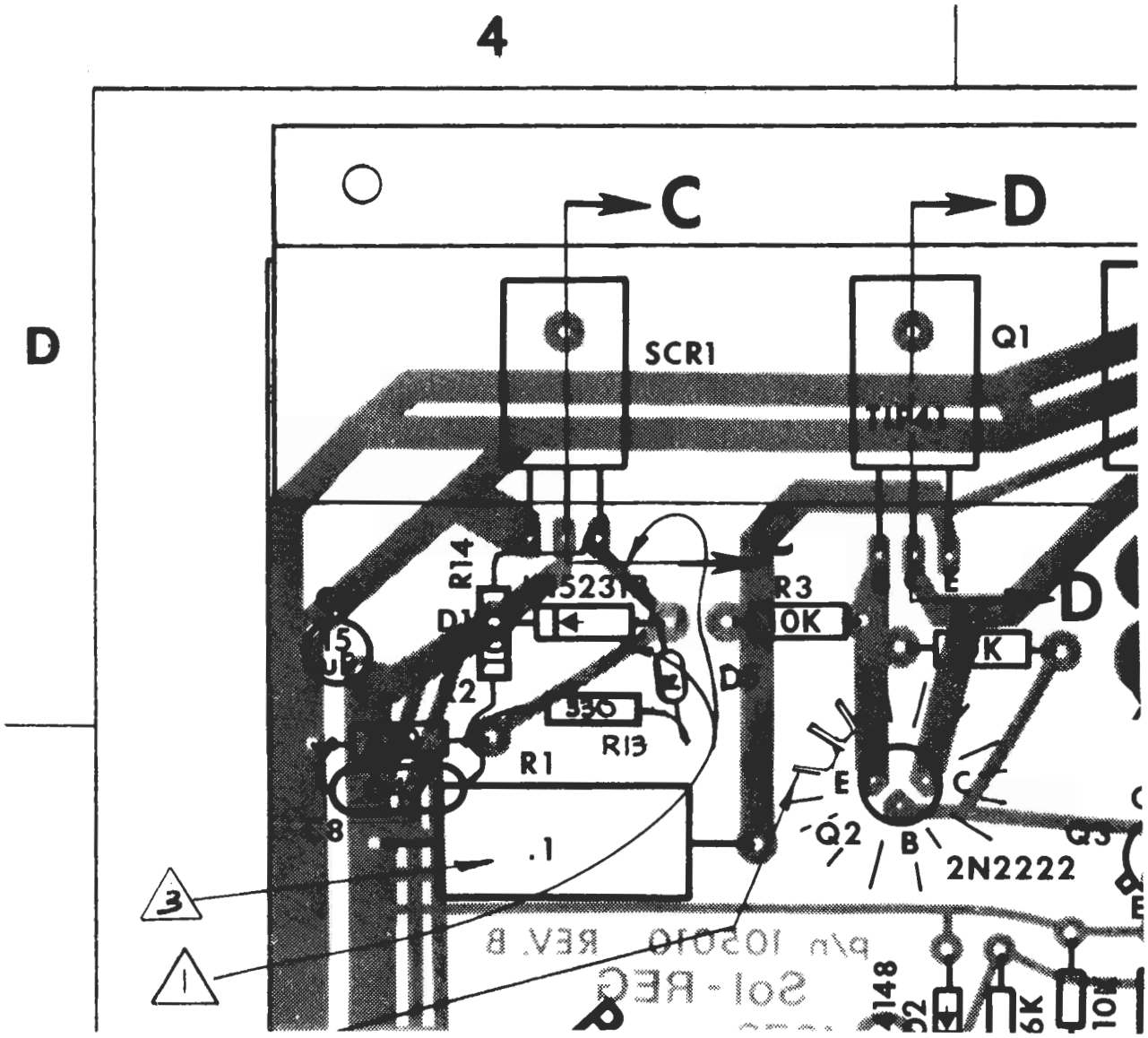
( ) Step 4. Wrap the leads of D5 (1N270) around the anode end (opposite banded end) of D1, and the loose end of R13 (330 ohms). Make sure the cathode (banded end) lead of D5 is connected to R13, not D1. Solder both ends of D5.

( ) Step 5. Trim all excess lead lengths, check lead dress, inspect for possible shorts or solder bridges.

The parts you have added are shown on the schematic Drawing X-16, with the exception of D5, which may be added by hand.

Add the underlined words to the second paragraph of Step 30, Section 6.6.3: "...pan head screws, #4 lockwashers, and #6 flat washers. Place lockwasher then flat washer on screw..." Add the following item to Hardware in Table 6-1: "8 (each) #6 flat washers." Add the following sentence at the end of Section 2.2.1: "Refer to the assembly drawing on page 2 of Assembly Procedure Change Notice #6-2 REV C, as you assemble the regulator."

Ref: ECN #10241



1 Cut these traces on the solder side of the PCB and add R13, R14 and C8 as shown.

3 Mount R1 approximately .15 from board surface.

Item	Page No.	Figure or Step No.	Changes
13	III-25	Step 28	Add note before Step 29, "Do Step 73 (p III-39) now."
14	III-27	Step 35	Add note to install: ( ) U93 74LS175, and ( ) U106 74LS175
15	III-30	Step 41	Delete installation of: ( ) U93 74LS175, and ( ) U106 74LS175
16	III-39	Step 71	Add note: "Mike input not recommended".
17	III-28	Step 38	In the third sub-step, add: "as in Figure 3-9".

In item 10 above, you made a note referring you to this Change Notice. Instead of the adjustment procedure given in Step 70 on page III-38, use the following procedure:

( ) Ground the "Audio In" jack J7 on Sol-PC.

( ) Apply power to Sol-PC.

( ) Using a high-impedance probe from an oscilloscope with a calibrated time base (a frequency counter is preferred, if available), monitor the VCO frequency appearing at pin 4 of U110 (type 4046).

( ) Adjust VR3 for a measured frequency of 14.0 kHz. This is a period of 71.4 usec.

Due to variations in the availability of ICs from various suppliers, a number of substitutions may be made of equivalent IC types. Please make the following changes in the manual on the pages given to reflect these possible substitutions:

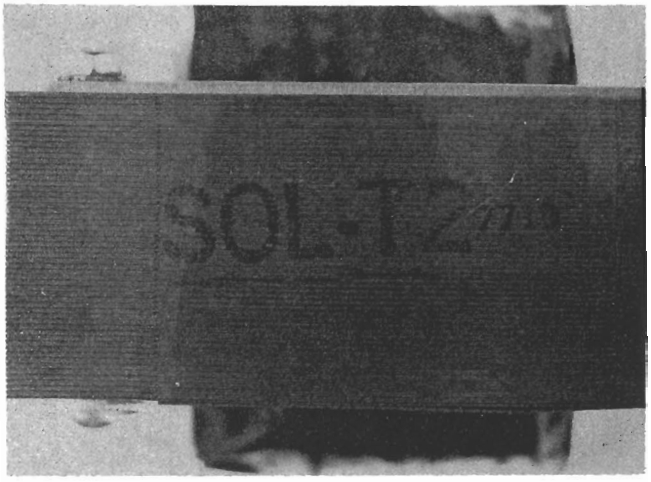
Item	Page Nos.	ICs	Main Type	Additional Substitutes
1	III-2, III-35	U95,6	74173	8T10
2	III-2, III-13	U104	AM0026	MH0026, 0026, "xx"0026
3	III-2, III-28 III-38	U51, U69	TMS6011NC	S1883, AV-5-1013, TR1602B
4	V-2, V-7	U1, U2	74LS175	25LS175
5	V-2, V-7	U18	8574	many possible equivalents

Processor Technology

CHANGE NOTICE #9

Refer to the Sol Systems Manual, Section X, drawing X-14. The main power transformer, Sol-T2, supplies power for the +8 V dc unregulated supply, which is used by S100 cards plugged into the backplane. Distributed regulators on each S100 card reduce this voltage to +5 volts regulated. Some Sol-T2 transformers supplied with the Sol 20 kits were designed for brown-out conditions; even though the A. C. power line voltage should drop below normal tolerances, these transformers can maintain the unregulated supply so that the +5 volt regulators do not drop out of regulation. Unfortunately, these transformers can provide over +11 volts at a normal line voltage of 120 volts. If the unregulated supply is lightly loaded by boards in the backplane, and the boards in the backplane place heavy demands on their +5 volt regulators, the result can be excessive dissipation in the regulators, activating their normal thermal shut-down circuitry. The power supply circuit can be modified with the addition of a bucking transformer which reduces the effective primary voltage at Sol-T2 by 10 or 20% thus reducing the unregulated supply by 10 or 20% to eliminate the problem. This modification is recommended for all Sols which have the brown-out transformers, if they are used at a line voltage of 110-120 v.a.c. A schematic of the new circuit is shown below.

The transformers can be identified by looking for markings on the laminations of the transformer, on either the top or sides. If the marking "Sol T2" in large letters is found as in the photos below, no modification is necessary. If the marking "4 3991" in small letters is found, make the modification. If any other or no marking is found, contact Processor Technology for further information. If a voltmeter is used to confirm an



overvoltage condition, unplug all boards in the backplane, disconnect the five-pin molex connector which supplies power to the backplane board, and make the measurements at the connector between the blue wire, +8 V dc, and the end white wire, ground.

The following parts are included in the modification kit:

(1) Triad F-57X 25.2 Volt CT Transformer, modified
(1) Molex commoning block, (07-01-70)
(1) 6-32 by .5" machine screw
(1) 6-32 hex nut
(1) #6 lockwasher
(5) Tywraps

With the covers removed from the Sol, and facing the front (keyboard) side, perform the following modifications:

1) Remove the A.C. line cord, and video cable from the rear panel.
2) Remove the four screws and lockwashers which hold the keyboard circuit board in place.
3) Remove the keyboard from the Sol and detach the interconnecting cable from J3 on the main circuit board.
4) Remove the two 8-32 by .5" screws on the right-hand chassis lip between the Sol-T2 transformer and the keyboard bracket. These screws go into inserts in the wood side panel. See the photo below for assembly details.
5) Place the new bucking transformer into position over the two holes from which the two 8-32 screws were removed, with the two black primary leads down, and the three secondary leads up. Reinstall the two screws and tighten.
6) Locate the #6 sheet metal screw in front of the MDA 980-1 bridge rectifier, from the photograph. Remove the screw from the bottom of the Sol chassis, and insert the 6-32 by .5" from the modification parts in its place.
7) Place the new commoning block over the screw and secure with the #6 lockwasher and 6-32 nut.
8) Remove the fan closure plate (designated part 1 in drawing X-1) from the chassis, as follows: Remove one screw from the bottom center of the plate, which holds the plate to the rear of the chassis. Remove two additional screws which go through the expansion back chassis into the lip of the fan closure plate. Lift the plate out vertically by rocking back and forth - it may be a tight fit.
9) Examine the cabling within the power supply. Two new wires must pass through this cabling from the area of the new transformer to the two existing commoning blocks beside the fan. Cut tywraps as necessary to accomodate the new wires.
10) Remove the black lead which is inserted into the number 2 position of the commoning block which is nearest the fan.

This lead is from the Sol-T2 transformer. Separate this lead from the twisted pair, back to its origin at the transformer.

11) Note that each of the leads to the new transformer has a different color: white, red, black, and yellow-red. Twist together the red and white wires into a twisted pair and route the pair beside Sol-T2, through the existing power supply cabling, to the two existing commoning blocks.

12) Insert the molex crimp pin from the red lead into position 2 of the commoning block nearest the fan, from which the other transformer lead was just disconnected. Push the crimp pin firmly into place until it clicks. Try to pull it out gently to confirm that it is properly seated.

13) Insert the crimp pin from the white lead into the position 2 of the commoning block farthest from the fan.

14) Insert the crimp pin of the black lead of the new transformer into position 5 of the new commoning block near the bridge rectifier.

15) Insert the crimp pin of the red-yellow lead of the new transformer into position 1 of the new commoning block.

16) Determine the line voltage at which the Sol will typically operate by a measurement or averaged series of measurements using an A.C. voltmeter.

17) If the typical line voltage is under 112 v.a.c., insert the crimp pin of the black lead of Sol-T2 which was previously detached into position 2 of the new commoning block, for a 10% voltage reduction at the primary of Sol-T2.

18) If the typical line voltage is over 112 v.a.c., insert the same crimp pin into position 4 on the new commoning block for a 20% voltage reduction.

19) Dress all leads, and install the tywraps provided to the cabling within the power supply.

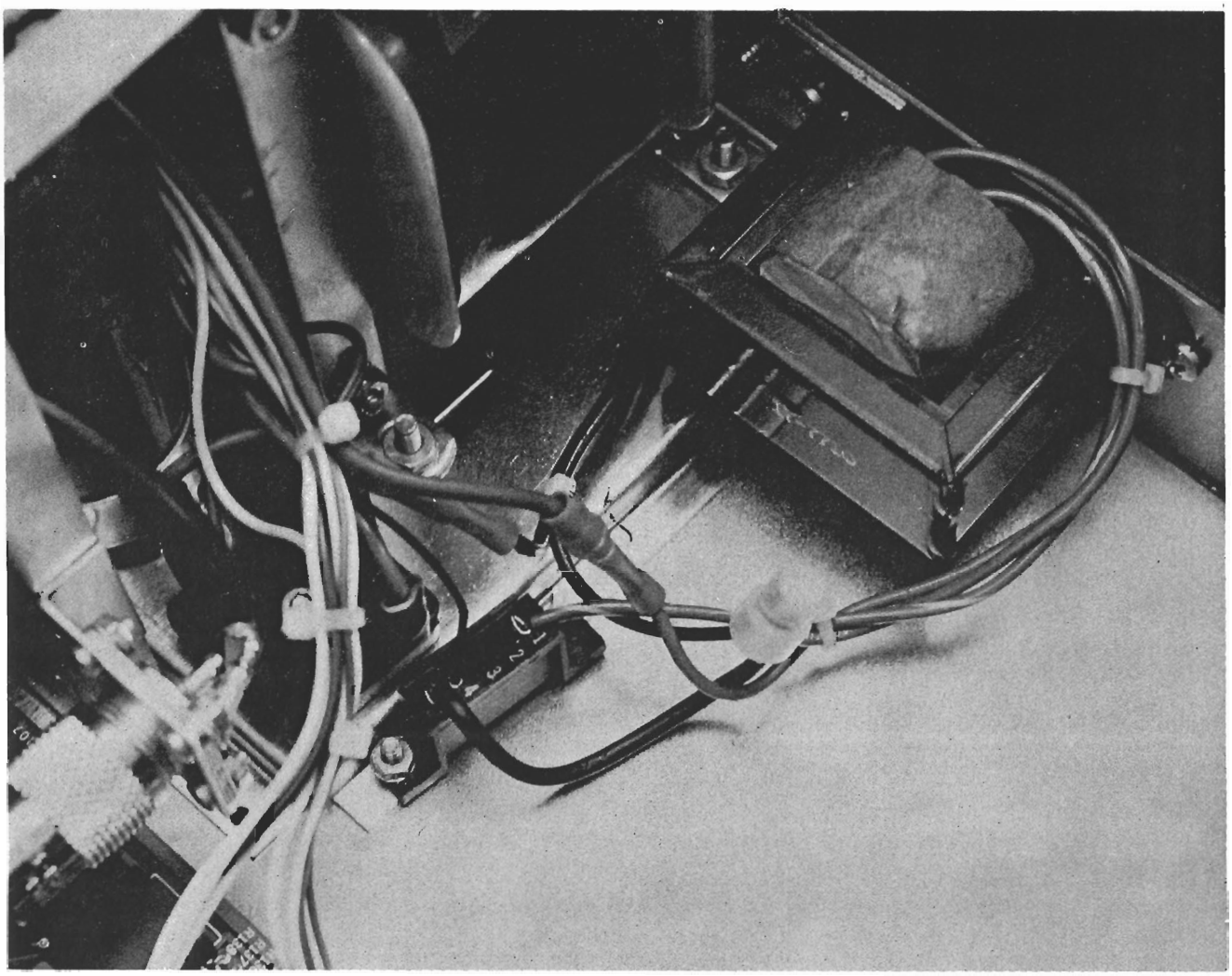
20) Push the fan closure plate back into position and secure with the three screws which were removed.

21) Plug the cable from the keyboard into J3 on the main circuit board. J3 is to the right of a similar jack.

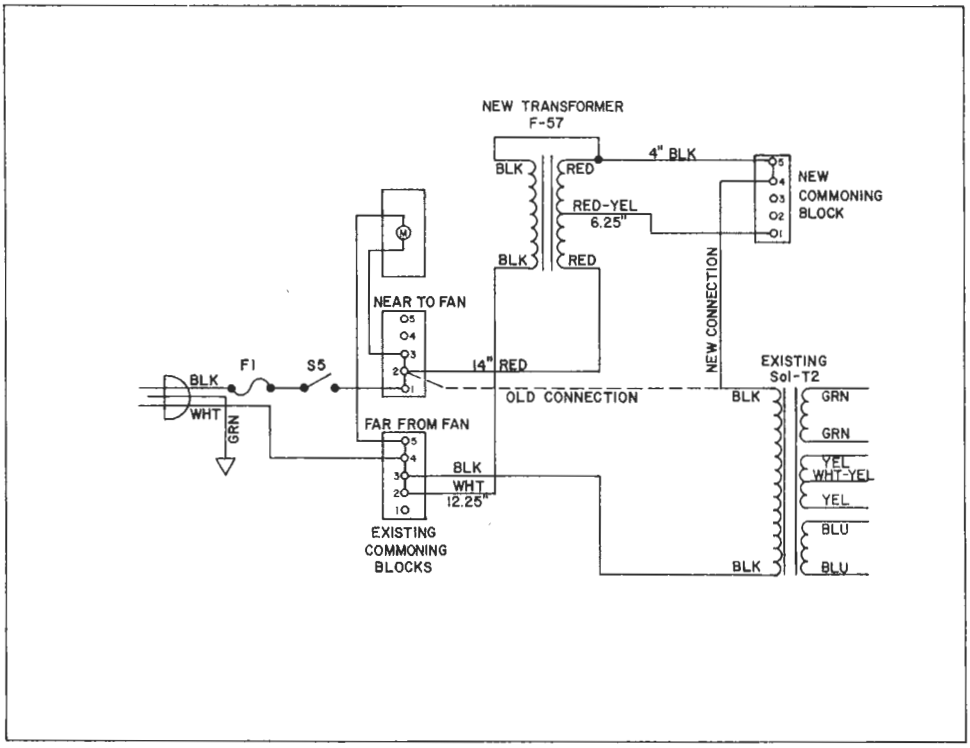
22) Place the keyboard back into position and secure with its four screws and lockwashers.

23) Apply power and measure the unregulated supply using the procedure described above, to confirm a 10 or 20% reduction.

Keep the pages of this notice in the Updates section of the Sol Systems Manual. Detach the schematic from page 4 and tape it into the blank area on drawing X-14.



CUT ALONG LINE



Sol MANUAL - CHANGE NOTICE

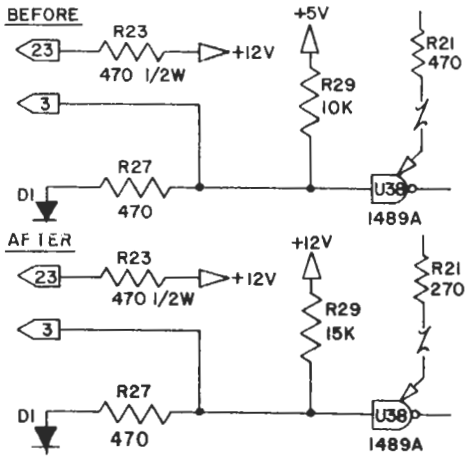
Processor Technology

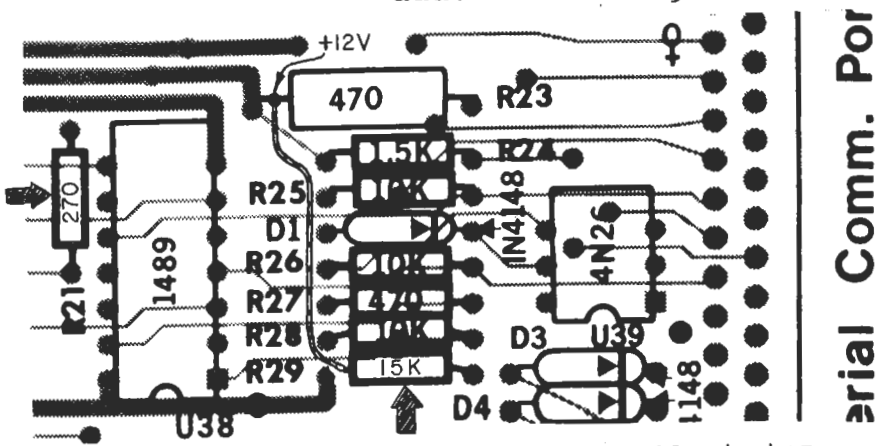
CHANGE NOTICE #10

Refer to Section X, Drawing X-17, Serial Data Interface/U.A.R.T. block. The section of U38 which has its input connected to pin 3 of J1 (Serial Loop Current Source) may not have enough drive at its input under worst case conditions, with the present values of R29, the input pull-up resistor, and R21, the resistor connected to pin 12 of U38. Substitute a 15K, 1/4 watt, 5% resistor for R29 (shown as 10K), and a 270 ohm, 1/4 watt, 5% resistor for R21 (shown as 470 ohm). These parts are included in the kit. Note in the schematics below that R29 is to be returned to +12 instead of +5. When R29 is installed, put the .9" length of tubing over the right hand lead. Clip this lead .1" longer than the tubing. With the legend on the P.C. board in normal reading position, hook this lead around the left-hand lead of R 24, a 1.5K resistor, and solder. Inspect the solder joint and lead dress for shorts. An assembly drawing detail of this modification is shown below.

Make the following changes in the manual before assembling Sol-PC.

Step	Page No.	Figure No., if any	Changes
1	X-17	Schematic, Input/Output	Change R21 value to 270 ohm
2	X-17	"	Change R29 value to 15K
3	X-17	"	Change R29 return to +12 V
4	X-3	Sol-PC Rev E Assembly	Change R21 value to 270 ohm
5	X-3	"	Change R29 value to 15K
6	X-3	"	Change R29 return to +12 V
7	III-3	Sol-PC Parts List	Add 1 270 ohm 1/4 watt, 5%
8	III-3	"	Change Qty. 470 ohm 1/4 watt to 2
9	III-3	"	Change Qty. 10K ohm to 31
10	III-3	"	Change Qty. 15K ohm to 2
11	III-33	Step 50	Change R21: 270 ohms, red-violet-brown
12	III-33	"	Change R29: 15K ohms, brown-green-orange
13	III-33	"	Under Step 50 instructions add: "See Change Notice #10"





PROCESSOR TECHNOLOGY CORPORATION 6200 HOLLIS STREET EMERYVILLE CA 94608 (415) 652-8080
CABLE ADDRESS "PROCTEC"

Ref: ECN NO.10059

ProcessorTechnology

Processor Technology
Corporation

7100 Johnson Industrial Drive
Pleasanton, CA 94566

(415) 829-2600
Cable Address - PROCTEC

Sol MANUAL - CHANGE NOTICE

CHANGE NOTICE #11

Refer to Section X, Drawing X-19, Sol Audio Tape I/O Schematic. U110, a 4046 phase-locked-loop IC, has its VCO center frequency adjusted by VR3, shown as a 50K potentiometer. The upper frequency limit of adjustment is determined by R154, shown as a 100K resistor. The lower frequency limit is determined by the total resistance of VR3 and R154. Due to extreme variations in the specifications of this part among various vendors, the range of adjustment provided by VR3 and R154 is occasionally insufficient. To correct this problem, the value of VR3 has been changed to 100K, and the value of R154 has been changed to 47K. Parts of these new values have been included in your kit or assembled Sol. To insure that the parts are correctly installed if you have a kit, and in any case to insure that the manual reflects these changes, make the following notes in the manual:

Item	Page No.	Figure or Step No.	Changes
1	X-19	Schematic	Change VR3 value to 100K
2	"	"	Change R154 value to 47K
3	X-3	Sol PC Assembly	Change VR3 value to 100K
4	"	"	Change R154 value to 47K
5	III-3	Table 3-1.	Change Qty 47K resistor to 2
6	"	"	Change Qty 50K pot. to 2
7	"	"	Add 1 100K pot.
8	"	"	Change Qty 100K resistor to 3
9	III-36	Step 60	Change R154 value to 47K, yellow-violet-orange; VR3-100K
10	III-38	Step 70	Add note: "See Change Notice #11."
11	III-7	Section 3.4	Under item 11, Oscilloscope, delete "(optional)", add "with calibrated time base".

Make the following additional changes in the manual, unrelated to the substitution of new values for VR3 and R154:

11	III-24	Step 28	After sub-step relating to Figure 3-8, add note: "Adjust VR1 and VR2 for centering of the display."
12	"	"	In last sub-step on this page, change U49 to U59.

Ref: ECN No. 10129

ProcessorTechnology

Processor Technology
Corporation

7100 Johnson Industrial Drive
Pleasanton, CA 94566

(415) 829-2600
Cable Address - PROCTEC

Sol MANUAL

CHANGE NOTICE #13

SUBJECT: Side Panel Assemblies. Superecedes Change Notice #12.

The two wooden side panels of the Sol are now supplied as completely assembled and finished subassemblies consisting of the walnut and masonite side pieces, the tinnerman plastic inserts, and 5/8" wood screws presently listed in the Parts List, Table 6-1. Section 6.6.2 contains the procedure which was used to assemble the panels. The procedure may be useful if it becomes necessary to refinish the walnut pieces; otherwise it is no longer necessary. Write the following note in the manual on page VI-8 next to the heading for Section 6.6.6: Skip to 6.6.3.

To achieve a better grip in the tinnerman plastic inserts in the side panels, the 10 screws which mate with the inserts have been changed from type 8-32 to type 10-24.

Document these changes in your manual now by making notes in the manual as indicated below. This will save you confusion later during assembly.

1) Change Table 6-1, page VI-2, as follows:

   a) Change "1 Left Side Piece, Walnut" to "1 Left Side Assy."
   b) Change "1 Right Side Piece, Walnut" to "1 Right Side Assy."
   c) Delete "1 Left Side Piece, Masonite".
   d) Delete "1 Right Side Piece, Masonite".
   e) Change quantity 8-32 x 1/2 Screw, Machine from 11 to 3.
   f) Change "2 8-32 x 1 Screw, Machine" to type "10-24".
   g) Delete "12 5/8 Screw, Wood".
   h) Delete "10 Tinnerman Plastic Inserts, Tapped".
   i) Add "8 10-24 x 3/8 Screw, Machine".

2) Change all references in Steps 18 and 30, pages VI-12 and VI--30 from type 8-32 to type 10-24.

3) Change items 9 and 10 in Drawing X-10 from 8-32 to 10-24.

Ref: ECN Nos. 10166, 10197

ProcessorTechnology

Processor Technology
Corporation

7100 Johnson Industrial Drive
Pleasanton, CA 94566

(415) 829-2600
Cable Address - PROCTEC

Sol MANUAL

CHANGE NOTICE #14

Step 27, on page II-14 calls for the connection of the white wire from the Sol-20 DC power cable to pad X4, a ground connection. This pad is too small; the wire should be run instead to the pad immediately to the right of pad T3, also a ground connection. This pad is shown on the legend as pad X-10. (NOTE: Some legends incorrectly show this pad as a second pad X-5. Make sure that the red-white lead goes to the pad labelled X-5 which is between C5 and FWB2.)

To avoid confusion when you reach this point in the assembly procedures, cross out "X4 (above R8)" in Step 27, and write in its place "X10(to right of T3)". If pad X10 is incorrectly labelled X5, the note should read "X5 (to right of T3)". Also make a note reading "See Change Notice #14."

Ref: ECN #10199

ProcessorTechnology

Processor Technology
Corporation

7100 Johnson Industrial Drive
Pleasanton, CA 94566

(415) 829-2600
Cable Address - PROCTEC

Sol MANUAL - CHANGE NOTICE

CHANGE NOTICE #15

SUBJECT: 2708 Personality Module, value change for R1 and R2.

If you have a Personality Module using 9216 ROMs, ignore this notice. The 100 ohm values for R1 and R2 shown in your manual may cause overheating under certain conditions. Use 130 ohm, 1/2 watt, 5% resistors instead, which may have been included in your kit.

To make sure your documentation reflects the improvement, note the change in pencil at the following places:

1) Section IV, Table 4-1.
2) Section IV, Step 2.
3) Section X, Drawing X-6, Assembly.
4) Section X, Drawing X-22, Schematic.

Ref: ECN #10231

ProcessorTechnology

Processor Technology
Corporation

7100 Johnson Industrial Drive
Pleasanton CA 94566

(415) 829-2600
Cable Address PROCTEC

Sol MANUAL

CHANGE NOTICE #16

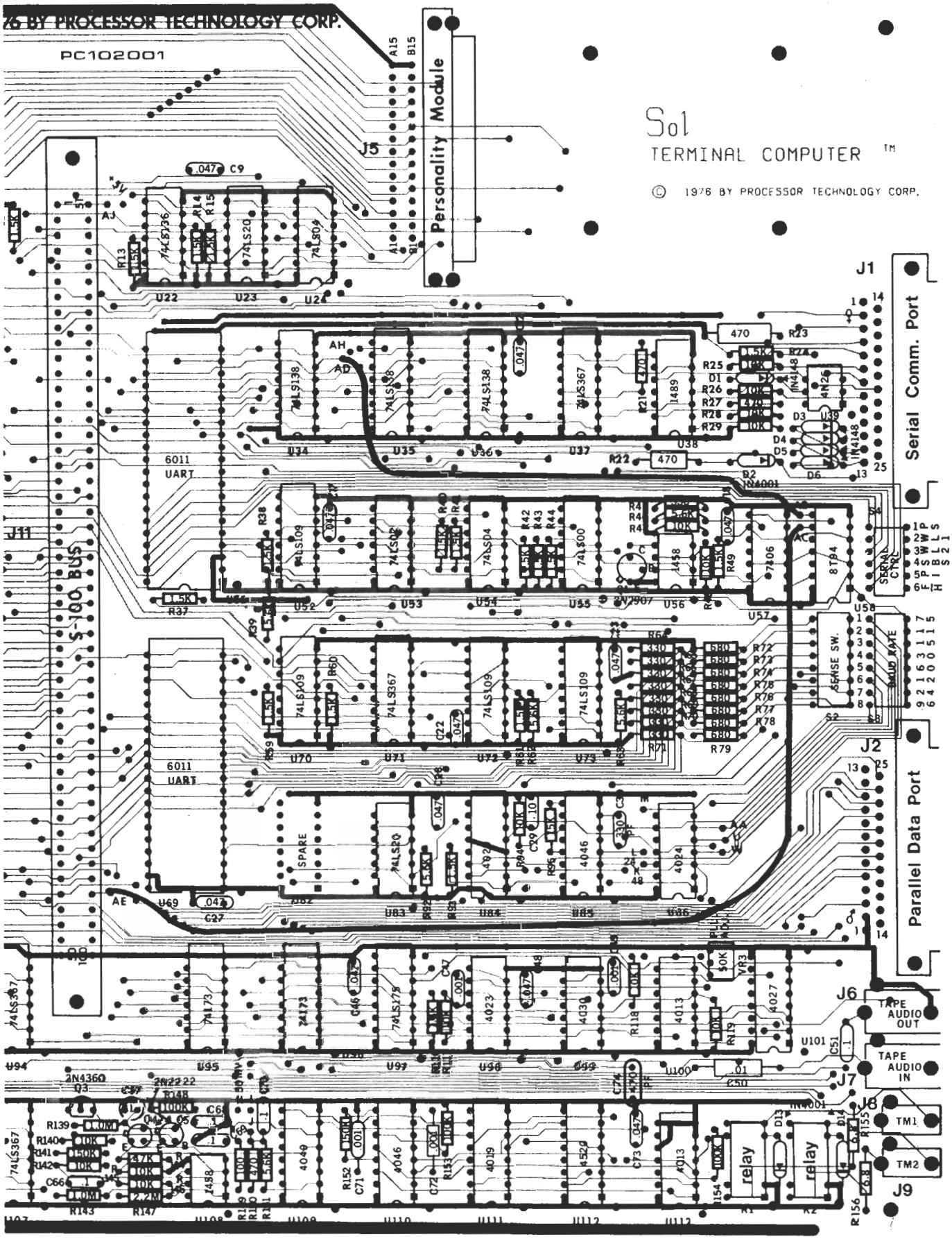
SUBJECT: Vectored Interrupt Capability for Sol

The Sol Computer and other S100 modules built by Processor Technology do not require use of the vectored interrupt capability which the 8080A microprocessor used in the Sol provides. The Sol provides, however, means for implementing vectored interrupt when the interrupt signal is made available by a circuit board inserted in the S100 bus, on S100 bus pin 96, SINTA. If a circuit board in the backplane generates interrupts, two jumpers, shown below, should be added to Sol P.C., to enable the SINTA signal to reach the memory decoder circuit. These jumpers may be added after completing the assembly of Sol-P.C., or even after the Sol is completely assembled and tested. The jumpers should be made from #24 solid, insulated wire (not provided). The electrical effects of these jumpers may be seen on Drawing X-16. These jumpers may be left in place even if no S100 board generates interrupts. S100 bus pin 96 may float with no interference.

( ) Step 1. Strip .1" of insulation from one end of each of two eight-inch lengths of wire, and insert into pads AC and AB (near U58) from the component side of the board. Solder and check for solder bridges.

( ) Step 2. Dress the wires as shown in the drawing below, trim to length, strip .1" from the loose ends. Insert the wire from pad AB into pad AD, and the wire from pad AC to AE. Solder and inspect for solder bridges.

( ) Step 3. Fix the long runs of wire to the board using silicone compound or tape.


76 BY PROCESSOR TECHNOLOGY CORP.
PC102001
Sol
TERMINAL COMPUTER ™
© 1976 BY PROCESSOR TECHNOLOGY CORP.
Personality Module
J5
J1
Serial Comm. Port
J2
Parallel Data Port
J11
S-100 BUS
6011 UART
6011 UART
SPARE
SENSE SW.
BAUD RATE
J6
TAPE AUDIO OUT
J7
TAPE AUDIO IN
J8
TM1
TM2
J9
relay
relay

Sol UPDATE 731011

SUBJECT: Optical Isolator (U39) Circuit Change

To enhance reliability of the optical isolator (U39) which couples data from a current loop device to the Sol, make the following circuit changes. Use the procedure given below after you complete Step 49 in Section III, Sol-PC Assembly and Test. Make a note in your manual after Step 49 to remind you of this change in the assembly procedure. Refer to the drawings on the next page while performing the following steps.

( ) Step 1. In Table 3-1, Page III-3, increase quantity of 47 ohm resistor from 2 to 3, reduce quantity of 470 ohm, 1/4 watt resistor from 2 to 1, and add one 4.7K ohm, 1/4 watt, 5% resistor.

( ) Step 2. In Step 50 on Page III-33, change R27 value from 470 ohms to 4.7K ohms (color code yellow-violet-red).

( ) Step 3. Complete Steps 50 through 58 in Section III.

( ) Step 4. Install R160 (47 ohm resistor, color code yellow-violet-black) as follows:

- ( ) Wrap one R160 lead around pin 1 of U39 (4N26) and the other around the cathode lead (banded end) of D3 (1N4148), dressing the leads as shown in Figure A.
- ( ) Solder both R160 leads in place and trim excess lead lengths.
- ( ) Inspect for possible shorts or solder bridges, especially between pins 1 and 2 of U39.
- ( ) On the back (solder) side of the board, the trace that connects pin 1 of U39 to the cathode lead of D3 must be cut. Using an Xacto knife or a razor blade, make two cuts approximately 1/8" apart, cutting across the trace down to the epoxy base. Insert blade tip beneath the cut section and gently work it away from the board. Be sure the "break" is free of solder.

Ref: ECN #10253

( ) Step 5. On Drawing X-17 in Section X, change value of R27 from 470 ohms to 4.7K ohms and add R160 (47 ohm resistor) between pin 1 of U39 and the cathodes of D3 and D4. (Refer to Figure B.)

( ) Step 6. Go on to Step 59 in Section III.

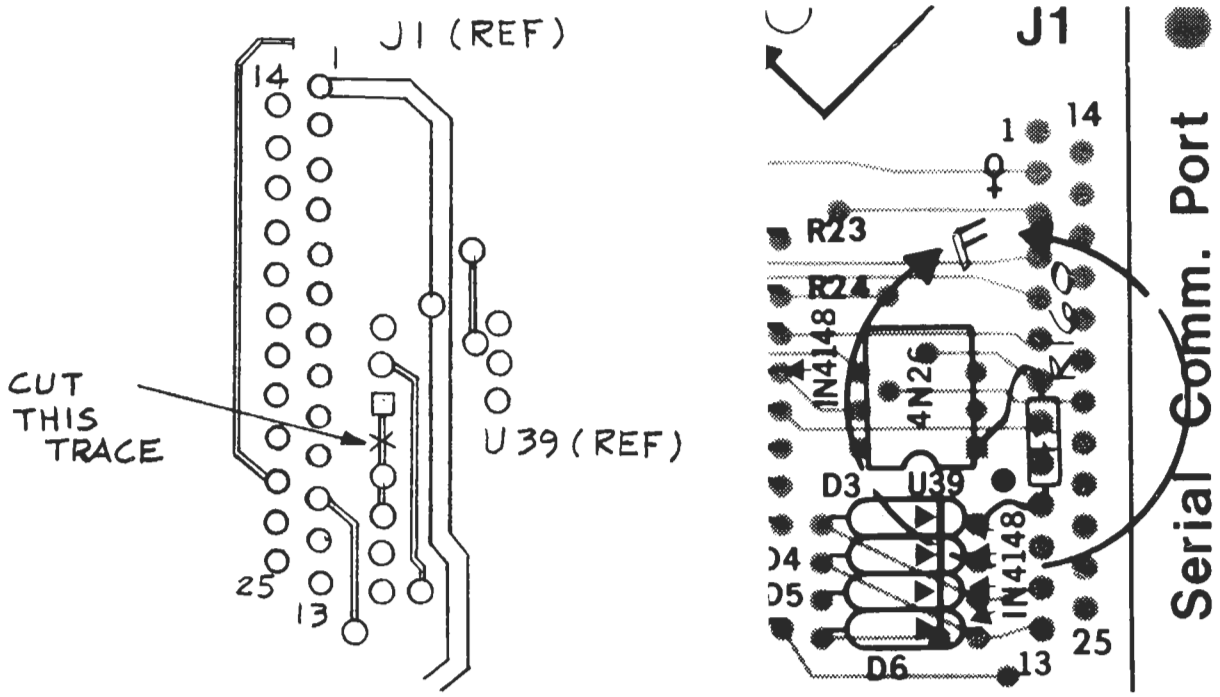


Figure A

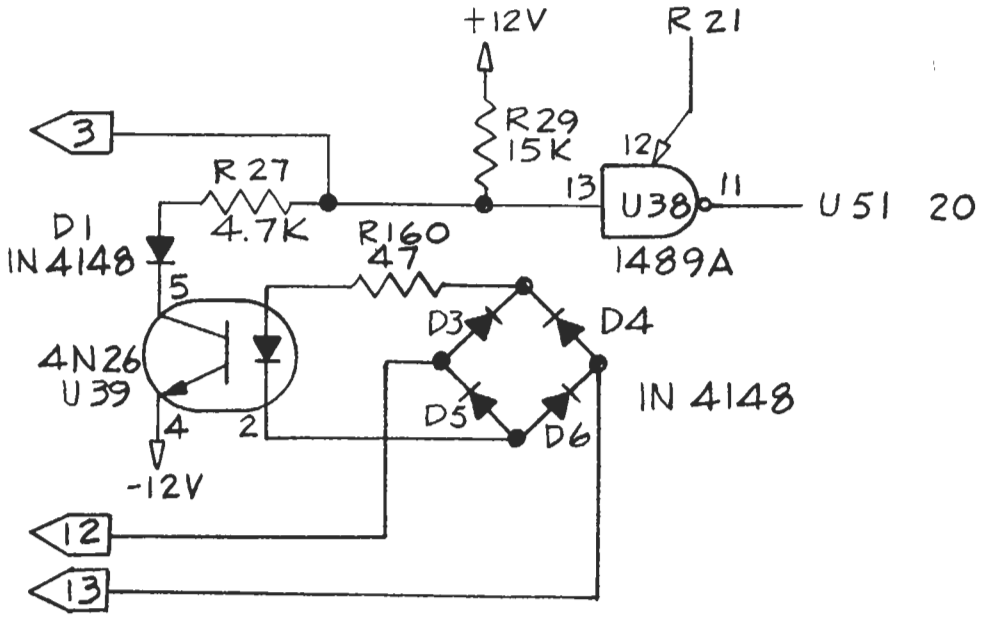


Figure B

ProcessorTechnology

Processor Technology Corporation

7100 Johnson Industrial Drive
Pleasanton, CA 94566

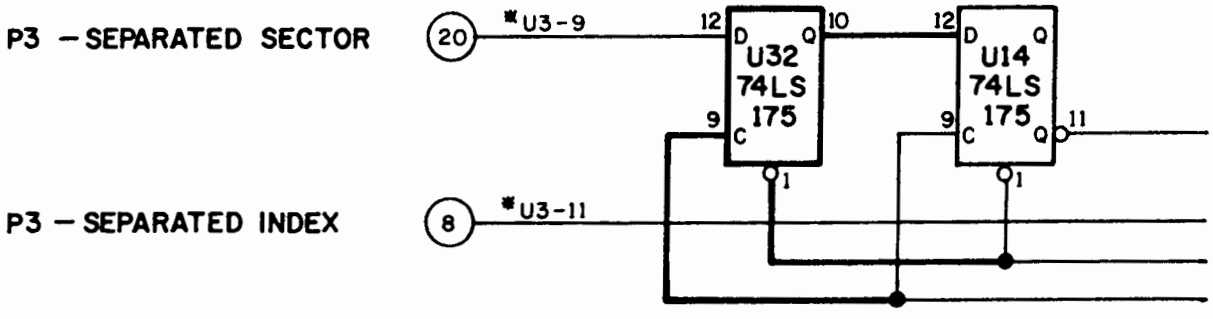
(415) 829-2600
Cable Address - PROCTEC

Helios Update 731048

This update describes modifications to Helios Formatter circuit boards being made at the factory. The changes are first described, with corrections to information in the Helios II Disk Memory System Manual. Then a step-by-step modification procedure is given, which factory-authorized service people may use to make the changes in the field, converting a Formatter board of assembly revision level E to level F.

Refer to the schematic of the Formatter board. The signal -SEPARATED SECTOR from the disk drive electronics appears at pin 12 of U14, a D-type flip flop, where it is synchronized by the clock signal. If the trailing edge of -SEPARATED SECTOR occurs too close to the rising edge of the clock input on pin 9, the Q output on pin 10 can have a short spurious pulse starting at the clock transition. The extra pulse is interpreted as an extra sector, and causes errors that PTDOS reports with the message "DISK STRUCTURE BAD". Since the error is caused by a rare coincidence of signals, it may not occur during extended periods of testing.

The detail of the schematic below shows an added D-type flip flop that eliminates the spurious pulse. The added flip flop, U32, is connected just as U14 was, and the spurious pulse can still appear at its Q output on pin 10. However, since the pulse starts on the rising edge of the clock pulse, it is too late to propagate through U14. The spurious pulse is of short duration, and is gone before the next clock. The output of U14 is the same as before the addition of U32, except that the occasional spurious pulse is removed.



Factory-authorized service people should modify all Formatter boards in the field, using the procedure given below. If the changes are made on Revision E assemblies, they become Revision F. The changes are independant of all previous modifications, however, and may also be made on assemblies prior to level E. Two parts are required: a 74LS175 (Quad D-type flip flop) and a 16 pin DIP socket. These parts are available from the Customer Service Department of Processor

Technology. Request part numbers 701146 (74LS175) and 713006 (socket). Jumpers should be made from 30 AWG kynar insulated wire, or 22 AWG solid insulated wire, as shown below.

1) Refer to the detail of the assembly drawing below. Install the 16 pin socket on the component side of the board at U32, making sure that pin 1 is seated in the square pad.

2) Holding the socket firmly in place, solder pins 7 and 15. This will secure the socket while jumpers are added.

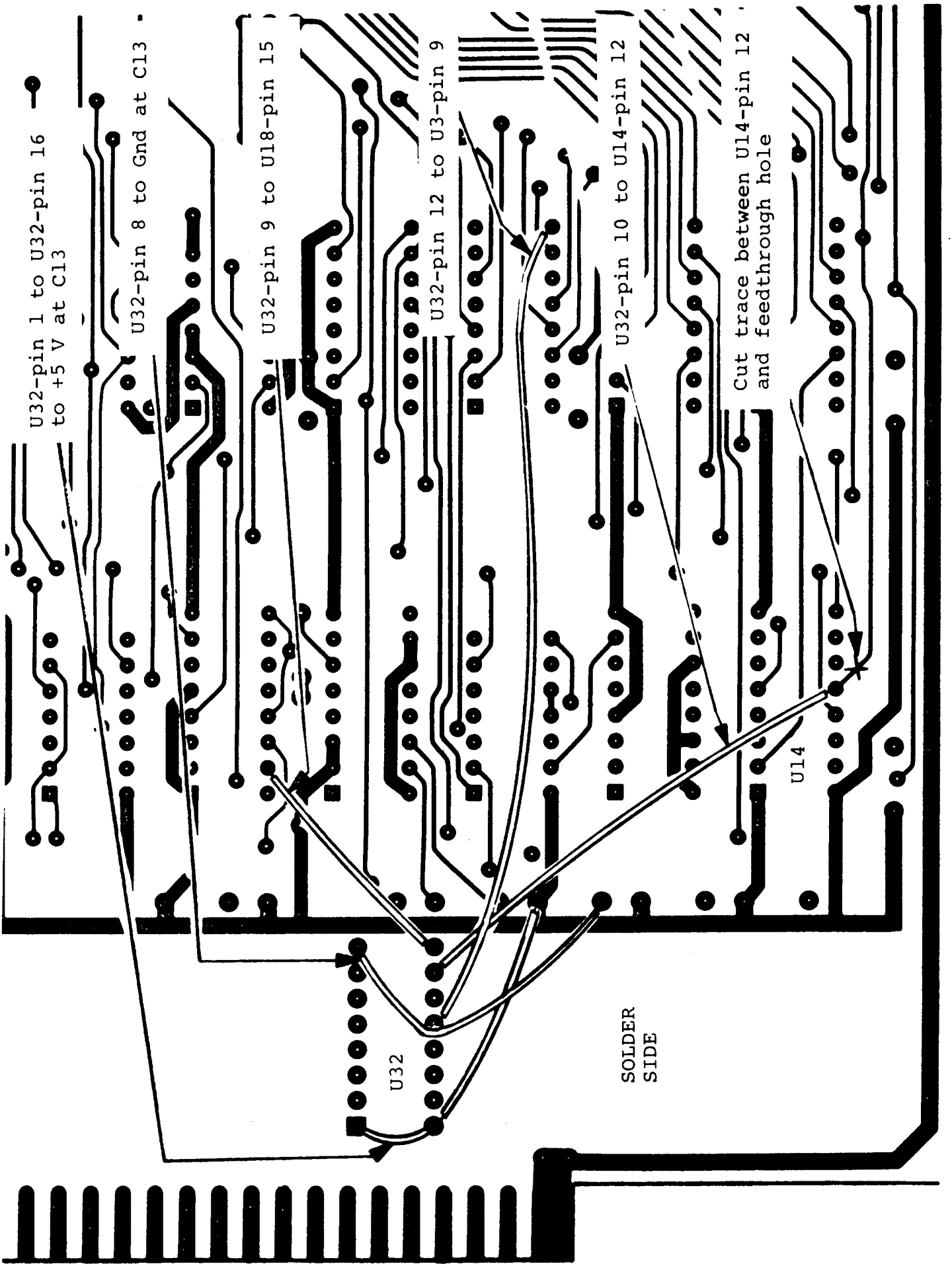
3) On the solder side, cut the trace between U14-pin 12 and a feedthrough hole, close to U14.

4) Solder the following 6 jumpers in place:

U32-pin 1 to U32-pin 16 (22 AWG wire)
U32-pin 16 to +5 V at C13 (22 AWG wire)
U32-pin 8 to ground at C13 (22 AWG wire)
U32-pin 9 to U18-pin 15 (30 AWG wire)
U32-pin 12 to U3-pin 9 (30 AWG wire)
U32-pin 10 to U14-pin 12 (30 AWG wire)

5) Install U32, a 74LS175, making sure that pin 1 is inserted in pin 1 of the socket.

6) If the modification was done on a level E assembly, cover the "E" with a label marked "F".



Ref. ECN 10431

Processor Technology
Corporation

7100 Johnson Industrial Drive
Pleasanton, CA 94566

(415) 829-2600
Cable Address - PROCTEC

Helios Update 731067

This update describes modifications to the Helios Controller circuit board being made at the factory. If you receive this update with a new or factory-serviced board, it describes the changes that were made. All Helios Controllers in the field should be modified according to the instructions in this update.

The changes increase the reliability of the transfer command register logic. To understand the function of the changes, read Section 7.10.1, subsection G1, of the Helios hardware manual, which describes the action of the transfer command register, U22, and read the circuit changes shown below. U22-pin 10, which selects the A or B inputs to the multiplexer is shown connected to a latch output before modification. The B inputs receive input from the data bus, and are selected for relatively long periods of time. Noise pulses appearing at U21-pin 12 can clock data appearing at the B inputs into the latches of U22 through NOR gate U21. This spurious data can cause a header to be written on the disk at the wrong time.

The modifications borrow an inverter section of U6 for a new function. This section was used in providing a signal to the jumper pin FX. Since a jumper from pin C to pin EX is never used, pin 5 of U23 can be driven with an equivalent signal appearing at U7 pin 13, two inversions before, freeing a section of U6. In the new circuit, shown below, U6 drives the Select input of U22-pin 10 to select the B (data) inputs only when port F1 is addressed, driving U21-pin 13 low. At all other times the A inputs are selected. If a noise pulse on U21-pin 12 happens to latch new data to U22's outputs, the data latched would be all ones from the A inputs, which does not cause a spurious header to be written.

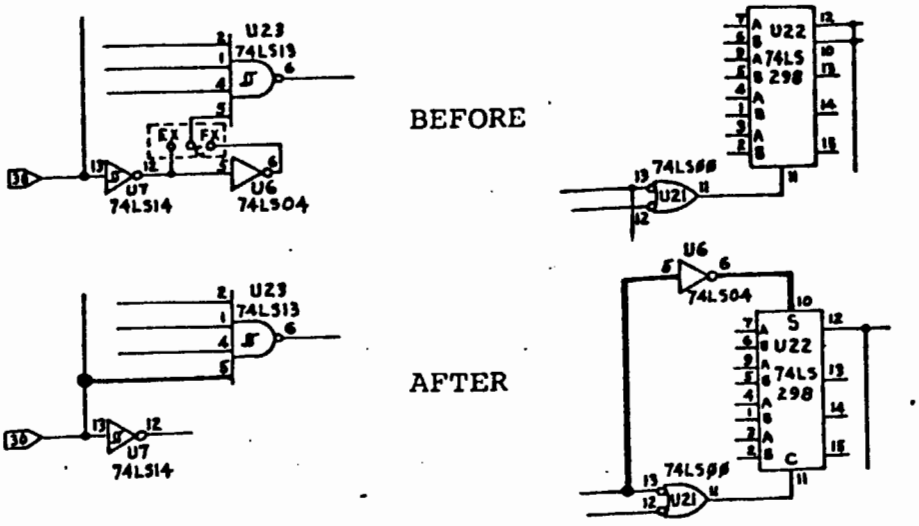
Three trace cuts and three jumpers shown in the figure below should be made as follows:

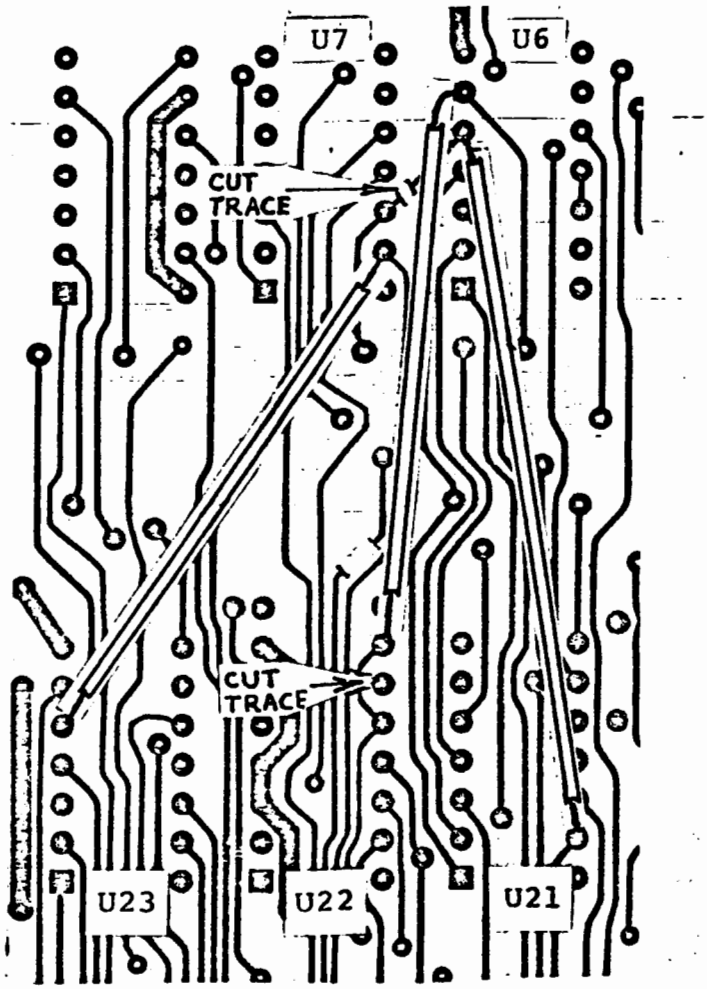
1) On the component side of the board, remove the wire jumper which connects the points marked "FX" and "C".

2) On the trace side of the board, cut the trace which connects U6-pin 5 to U7-pin 12, and the trace which connects U22-pin 10 to U22-pin 12.

3) Using 30 A.W.G. solid insulated wire, solder jumpers between U6-pin 6 and U22-pin 10, U6-pin 5 and U21-pin 13, and U7-pin 13 and U23-pin 5.

Ref. ECN 10346


BEFORE
AFTER
U23
74LS13
U7
74LS14
U6
74LS04
U22
74LS
298
74LS00
U21


U7
U6
CUT
TRACE
CUT
TRACE
U23
U22
U21

Processor Technology

Processor Technology Corporation

7100 Johnson Industrial Drive
Pleasanton CA 94566

(415) 829-2600
Cable Address PROCTEC

Helios II Update 731071

MODIFICATIONS TO CONTROLLER PCB ASSY 301000 (H TO J)

1.1 PRELIMINARY INSTRUCTIONS AND REASON FOR CHANGE

This update describes modifications to the Helios II Controller PCB which bring that PCB from assembly revision level H to J. The update also describes associated revisions to the Helios Technical Manual, 730009. The previous update (731067) which you should have received, brought the Controller PCB from level G to level H. It is important that you mark the current revision level of your Controller PCB when you have completed these modifications because the markings enable one to implement succeeding changes without confusion and are necessary for trouble-shooting and other purposes.

Identify the assembly revision level of your Controller PCB. (If you need help, refer to the Helios manual, section 3.2.1.B, "Identifying Revision Levels of Assemblies.") If it has been recently shipped from the factory, it may already be a "J", in which case you can skip to 1.4 of this update, "Updating Your Helios Manual." If it is an "H" or lower, you should perform the modifications because they result in improved low logic levels on the DI/O bus during DMA reads from memory, which means better reliability in your system.

1.2 DESCRIPTION OF THE CHANGE

Drivers (8839s) which contain an additional disable pin for the outputs are substituted for the 8833s at U47 and U48. Pins 7 of U47 and U48 are disconnected from ground and connected to the Q-output of the write flipflop (U39-6), which is high during DMA memory reads. Thus the output drivers of the controller are disabled during the DMA reads. In summary, the changes to the hardware required to implement this change consist in substituting two ICs, making three trace cuts, and installing three jumpers.

1.3 INSTRUCTIONS FOR MODIFICATION

1) Be sure you are familiar with the information in the Helios manual, Sections 3.2.3, Integrated Circuits and 3.3, Modifying PCBs.

2) Remove the 8833s from U47 and U48.

3) Make the following trace cuts:

   (Refer to Detail C, in this update.)

a) On the component side, cut the trace between U48-7 and U49-10.

b) On the solder side, cut the trace between U48-7 and U48-8.

c) On the solder side, cut the trace between U47-7 and U48-8.

4) Install two 8839s at U47 and U48 (P/N 701181).

5) Using #30 green Kynar insulated jumper wire, make the following jumper connections on the solder side:

a) U49-8 to U49-10.

b) U48-7 to feedthrough pad between U49-10 and U48-7.

c) U48-7 to U47-7.

6) Place a label marked "REV J" as shown in note 3 of Fig. 8-6, Controller PCB Assembly (Helios manual).

1.4 UPDATING YOUR Helios MANUAL

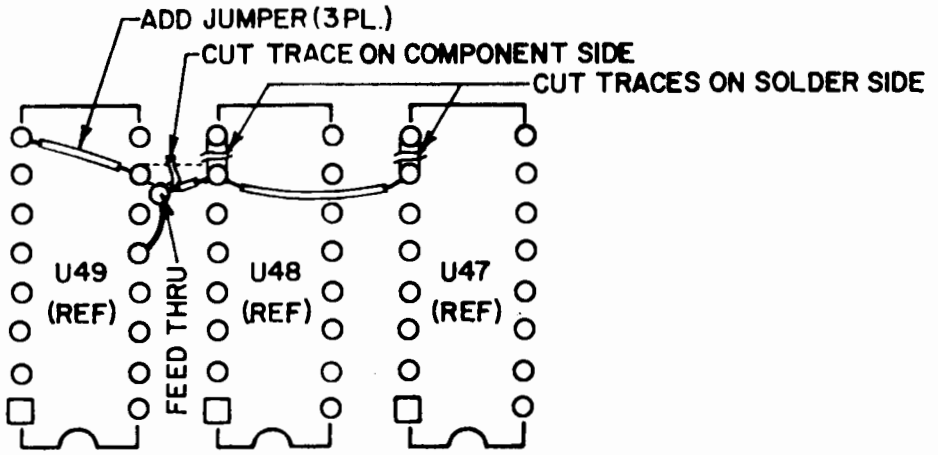
1) Insert this update in Section 10, Updates, following update 731067.

2) In the Appendix, Section 9, replace pages 9-13/9-14 REV A with the attached REV B sheet.

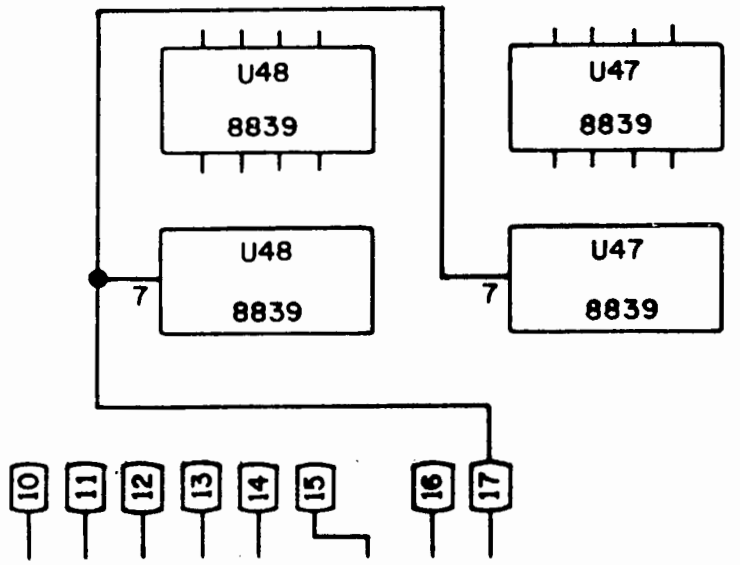
3) Mark the assembly drawing, Fig. 8-6 and the schematic, Fig. 8-11, with the words, "See Updates 731067 and 731071."

4) Change the assembly drawing, Fig. 8-6, Note 6, to read:

"30 AWG" not 24 AWG.



Detail C of Assembly Drawing (Fig. 8-6) Showing Modifications.
(Viewed from Solder Side.)



Detail of Controller Schematic (Fig. 8-11) after Modifications.

PARTS LIST - CONTROLLER PCB ASSEMBLY (301000J)
(Refer to Fig. 8-6)

ITEM #	PART #	QTY	REFERENCE CODE	DESCRIPTION
1	301001	1		Fab, PCB, Controller, REV C
6	701090	2	U18,21	74LS00, Quad 2-Input NAND-gate
7	701092	1	U19	}74LS02{; 9LS02, Quad 2-Input NOR-gate
8	701094	1	U6	74LS04, Hex Inverter
9	701098	3	U0,12,36	74LS08, Quad 2-Input AND
10	701100	1	U35	74LS10, Triple 3-Input NAND
11	701102	1	U17	74LS11, Triple 3-Input AND
12	701104	2	U8,23	74LS13, Dual 4-Input NAND
13	701106	1	U7	74LS14, Hex Inverter, Schmitt
14	701118	1	U40	74LS86, Quad 2-Input EX-OR
15	701120	5	U2,10,20, U37,39	74LS109, Dual J-K FF
16	701122	1	U15	74LS123, Dual Retrig. One-Shot
17	701128	1	U42	74LS138, 3-to-8 Line Decoder
18	701130	2	U14,16	74LS139, Dual 2-to-4 Line Decoder
19	701138	2	U13,34	74LS157, Quad 2-to-1 Line Multiplexer
20	701140	1	U11	74LS158, Quad 2-Input Inverter Multiplexer
21	701142	1	U5	74LS163, Synch 4-Bit Binary Counter
22	701144	1	U31	74LS174, Hex D FF
23	701146	2	U1,38	}74LS175{; 25LS175, Quad D FF
24	701148	7	U24-30	74LS191, Synch UP/DN Counter
25	701152	1	U33	74LS279, Quad S-R Latches
26	701156	1	U22	74LS298, Quad 2-Multiplexer, Store
27	701181	2	U47,48	8839, Quad T-S Transceiver, 16P
28	701186	11	U3,9,32,41, U43-46,49-51	}8T97{; DM8097, 74367, High Speed Hex Buf/Inv
29	701196	2	U52,53	9403, 4 x 16 FIFO Buffer
30	701162	2	U54,55	}7805{; LM340T-5, Volt Regulator, +5V, TO-220
33	702002	1	Q1	2N2222, Transistor, NPN
36	703005	1	CR1	1N4148, Diode, Silicon, Switching
39	705022	3	R1,4,5	Res, 220 ohms, 1/4W, 5%
40	705025	3	R2,3,6	Res, 330 ohms, CF, 1/4W, 5%
41	705061	2	R8,9	Res, 10K ohms, CF, 1/4W, 5%
42	705087	1	R7	Res, 1.5 ohms, 1/4W, 5%
43	705096	1	U4	761-5-R-220/330, Resistor 220/330 ohms, DIP, Network
46	707023	18	C3-7,10-22	Cap, .047uf, Disk Ceramic, +80 -20%
47	707032	2	C9,23	Cap, 1.0uf, Tantalum, 35V, 10%
48	707034	1	C1	Cap, 2.2uf, Tantalum, 10%
49	707036	2	C2,8	Cap, 15uf, Tantalum, 20V, 10%

PARTS LIST - CONTROLLER PCB ASSEMBLY (301000J) (Continued)

ITEM #	PART #	QTY	REFERENCE CODE	DESCRIPTION
52	713004	13		Socket, DIP, 14P, Solder
53	713006	39		Socket, DIP, 16P, Solder
54	713013	4		Socket, Strip, 12P, Solder
56	717050	1	P4	Connector Pin
57	717049	1	P4	Receptacle, Test Point, Printed Circuit, Right Angle, Orange
58	721022	1		Heatsink
59	720020	2		PHMS, 6-32 x 1/2",
60	720011	2		Hexnut, 6-32
61	720041	2		ITLW, #6
62	717003	2	P2,3	Header, Male PC, Mount, 50-Pin
64	711004	1		Label, Assy. Rev, 1/4"
65	716027-5	A/R		Wire, SS, Insul, 22 AWG, Green
67	716026-5	A/R		Wire, SS, Kynar Insul, 30 AWG (approx. 14 inches)

Processor Technology

Processor Technology
Corporation

7100 Johnson Industrial Drive
Pleasanton, CA 94566

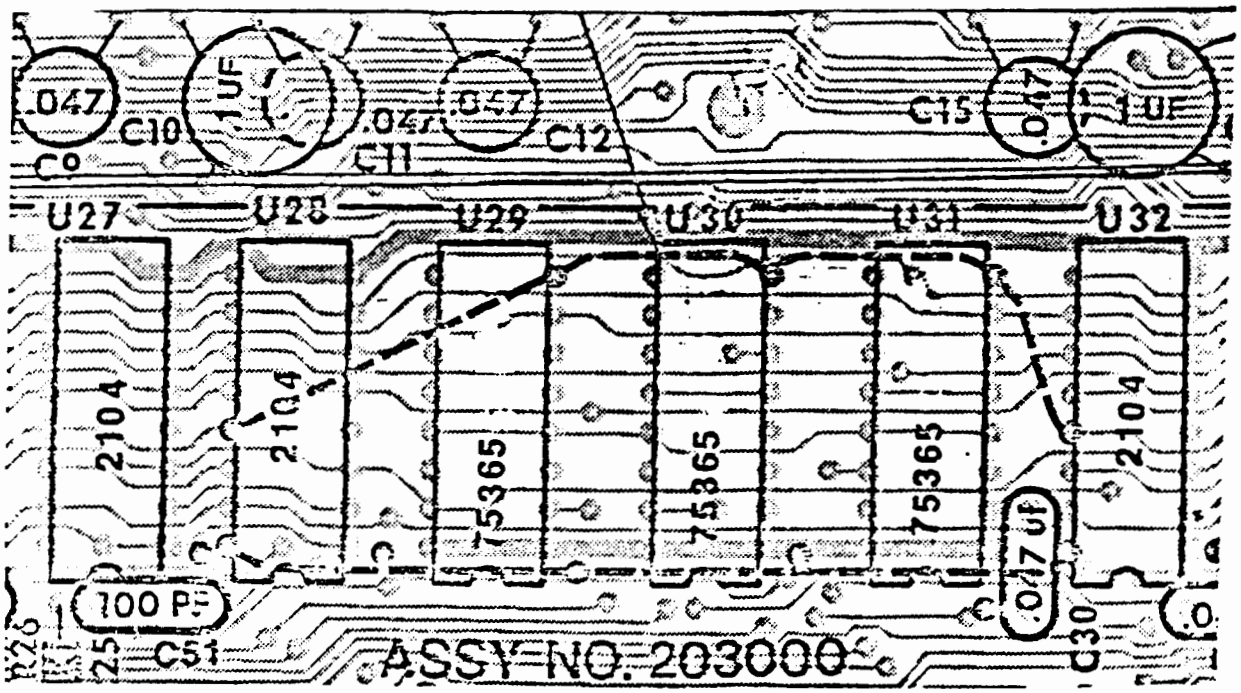
(415) 829-2600
Cable Address · PROCTEC

16KRA MANUAL

Change Notice #1

The 16KRA P.C. board you have received with this manual is designated as a REV J assembly. The portion of the assembly drawing (see page VI-1) printed below shows modifications which were made to increase reliability, which are not shown in the manual. Five jumper wires have been added to the trace side of the board, all at ground potential, to improve ground return paths. U30 is shown as a 75365--no alternate part 3207 should be used at this location. These changes involve no change to the schematic diagram, page VI-2, or to the Theory of Operation, Section V.

It is not recommended that these modifications be performed on P.C. boards of previous revision levels unless such boards exhibit intermittant failures.



Ref. ECN #10243

Processor Technology

Processor Technology
Corporation

7100 Johnson Industrial Drive
Pleasanton, CA 94566

(415) 829-2600
Cable Address - PROCTEC

nKRA Update 731047

This update describes modifications to nKRA memory boards (16KRA-1, 32KRA-1, 48KRA-1, and 64KRA-1) being made at the factory. The modifications are identical for the four types of board. Boards with the modifications are identified as Revision B assemblies.

Since this modification is performed on all boards in the nKRA family, this update is a revision to the following publications:

PRODUCT	PUBLICATION(S)
16KRA-1	32KRA-1 User's Manual (1st ed.), with 16KRA-1 Description
32KRA-1	32KRA-1 User's Manual (1st ed.)
48KRA-1	48KRA-1 User's Manual (1st ed.)
64KRA-1	48KRA-1 User's Manual (1st ed.), with 64KRA-1 Description

The first part of this update describes the nature of the changes and the corresponding corrections in the User's Manual. The second part describes the procedure that factory-authorized service people may use to make the changes to Revision A assemblies in the field.

Revision A boards depend on certain timing conditions in the Sol that are not present in a number of other 8080-based S-100 computers. The modifications eliminate dependence on unique timing conditions, making the boards compatible with many more S-100 computers based on the 8080. The modifications also improve timing margins, increasing reliability. They involve three trace cuts, three jumpers, and the replacement of U56, the "State Machine" PROM, with a PROM that is differently programmed.

The nKRA depends on the stability of S-100 status signals $\overline{\text{SWO}}$, SINP, and SOUT at the falling edge of the PHASE 2 clock, although in many S-100 computers these signals are not usable until the rising edge of the PHASE 1 clock. Changes in the operation request logic contained in U56 make the circuit tolerant of the later occurrence of these three status signals.

Three wiring changes described below are shown in Figure 1, a portion of the schematic included in the 32KRA-1 and 48KRA-1 User's Manuals:

1) The signal $\overline{\text{MSEL}}$ was required at U58-pin 3. The new logic in U56 removes the need for this connection, and U58-pin 3 is grounded.

2) The timing of the the signal PEND at U46-pin 10 is changed. U46 was clocked 150 nsec after the falling edge of PHASE 2, with a signal from the delay line. The change clocks U46-pin 12 by the rising edge of PHASE 2, shown also in Figure 1. This change results in more tolerance to the variations of clock timing in different S-100 computers.

3) Revision A boards contain a wiring mistake that does not affect normal operation. The correction connects U57-pin 4 to the bus signal SOUT, rather than to SOUT • PWR at U58-pin 4, which occurs 30 nsec later. This change improves timing margins.

Attached to this update are two pages containing the complete nKRA schematic, with several minor documentation improvements, as well as the changes described above. Pages containing new versions of Figure 5-2 and Table 5-1 are also attached, showing the changes to the operation request logic and the states of the state machine caused by the new programming of U56 (the state machine PROM). These pages replace the corresponding pages in both the 32KRA-1 and 48KRA-1 User's Manuals.

Text in Section 5, Theory of Operation, of both manuals should be changed as follows:

Change the second and third paragraphs in Section 5.4.1 to read:

> Operation request and sequencing is controlled by a state machine. Table 5-1, States of the State Machine, shows the possible states expressed in terms of the ROM inputs and outputs. The state machine changes its outputs REFR and REN at the fall of PHASE 2. It changes its output WE at the fall of REN. It changes its output PEND at the rise of PHASE 2.
>
> Each of the 11 possible states is a variation of one of the four types of memory cycles: Null, Write, Refresh and READ. (Refer to 5.3.3, Types of Memory Cycles.)

Change all of Section 5.5.1, Normal Access, Selected to read as follows:

> Normal access, selected or unselected with $\overline{SWO}$ high, or unselected with $\overline{SWO}$ low, causes a Read state followed by a Refresh state or a Null state.
>
> Normal access, selected, and $\overline{SWO}$ low causes a Read state followed by Refresh or Null state, followed by a Selected Write state.

It is recommended that factory-authorized service people modify all nKRA boards in the field to the Revision B assembly level, using the procedure below. Before starting the modifications, obtain a new part for U56, available from the Customer Service Department at Processor Technology. Request part number 726111.

1) Refer to Figure 2 below. On the component side of the board, cut the trace between U58-pin 3 and the nearby feedthrough hole. Also cut the adjacent trace between U58-pin 4 and U57-pin 4.

2) Refer to Figure 3 below. On the solder side of the board, cut the trace between U46-pin 12 and the feedthrough hole.

3) Install the three jumpers shown in Figure 3 on the solder side:

between U55-pin 9 and U57-pin 4,
between U46-pin 12 and U49-pin 13, and
between U58-pin 3 and U58-pin 7.

4) Remove and discard U56, and replace it with a new part from the factory that is stamped with the part number 726111.

5) Cover the Revision letter "A" in the top right-hand corner of the board with a label bearing the letter "B".

6) The delay line at U50 has a tendancy to work loose during transit. Apply insulating varnish where the delay line meets its socket, at four points near the corners. Be careful that the varnish does not penetrate into the socket. Keep the delay line tight in its socket while the varnish dries.

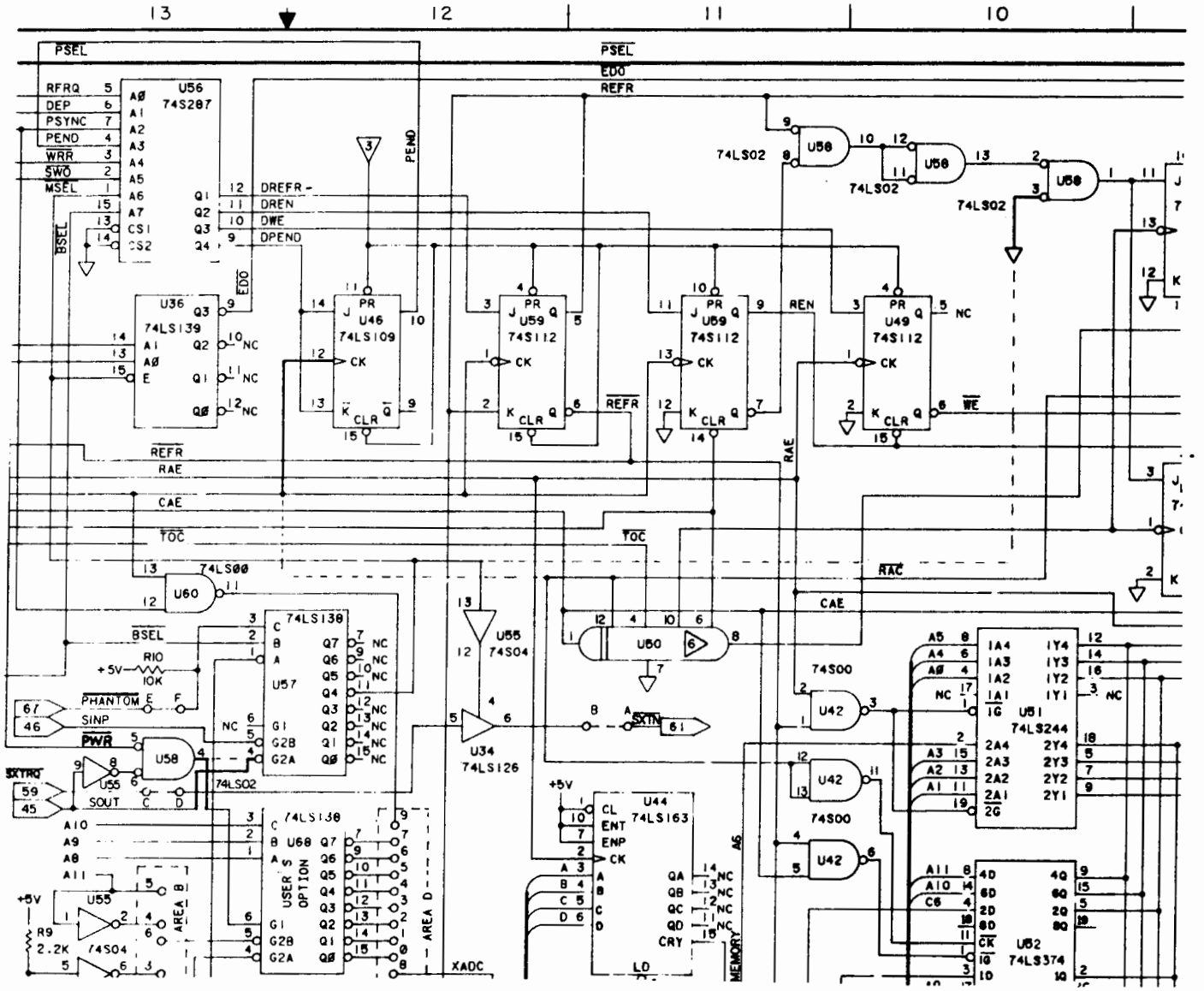


Figure 1. Portion of nKRA Schematic

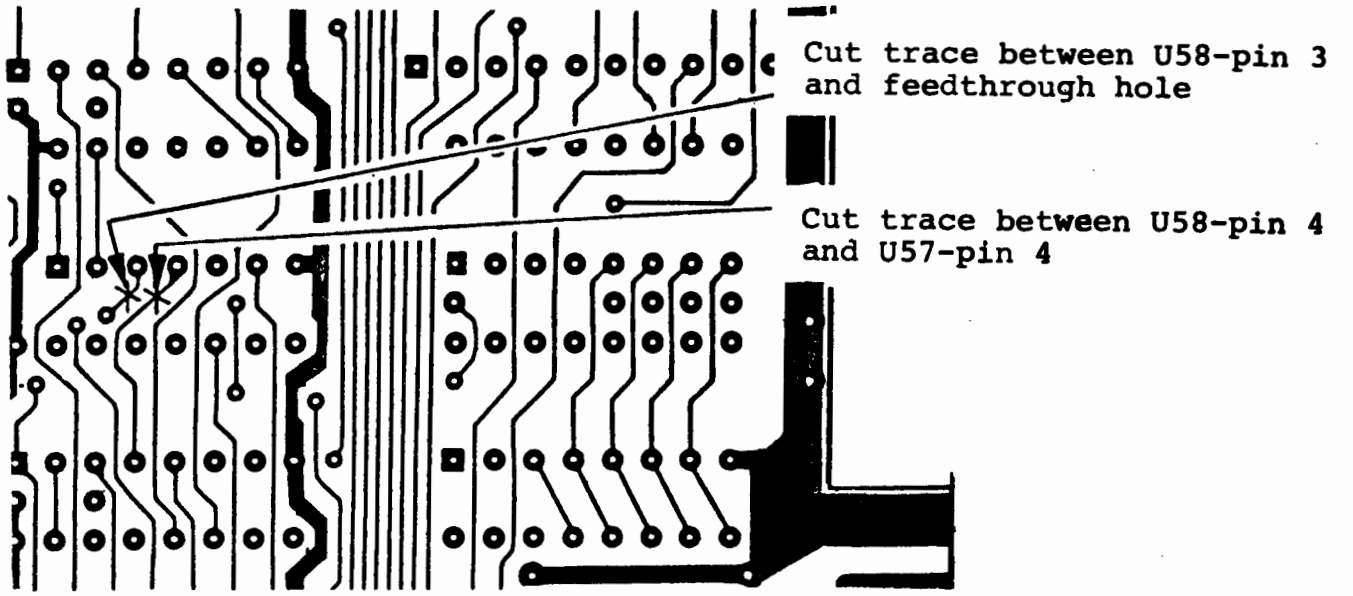


Figure 2. (Shows component side)

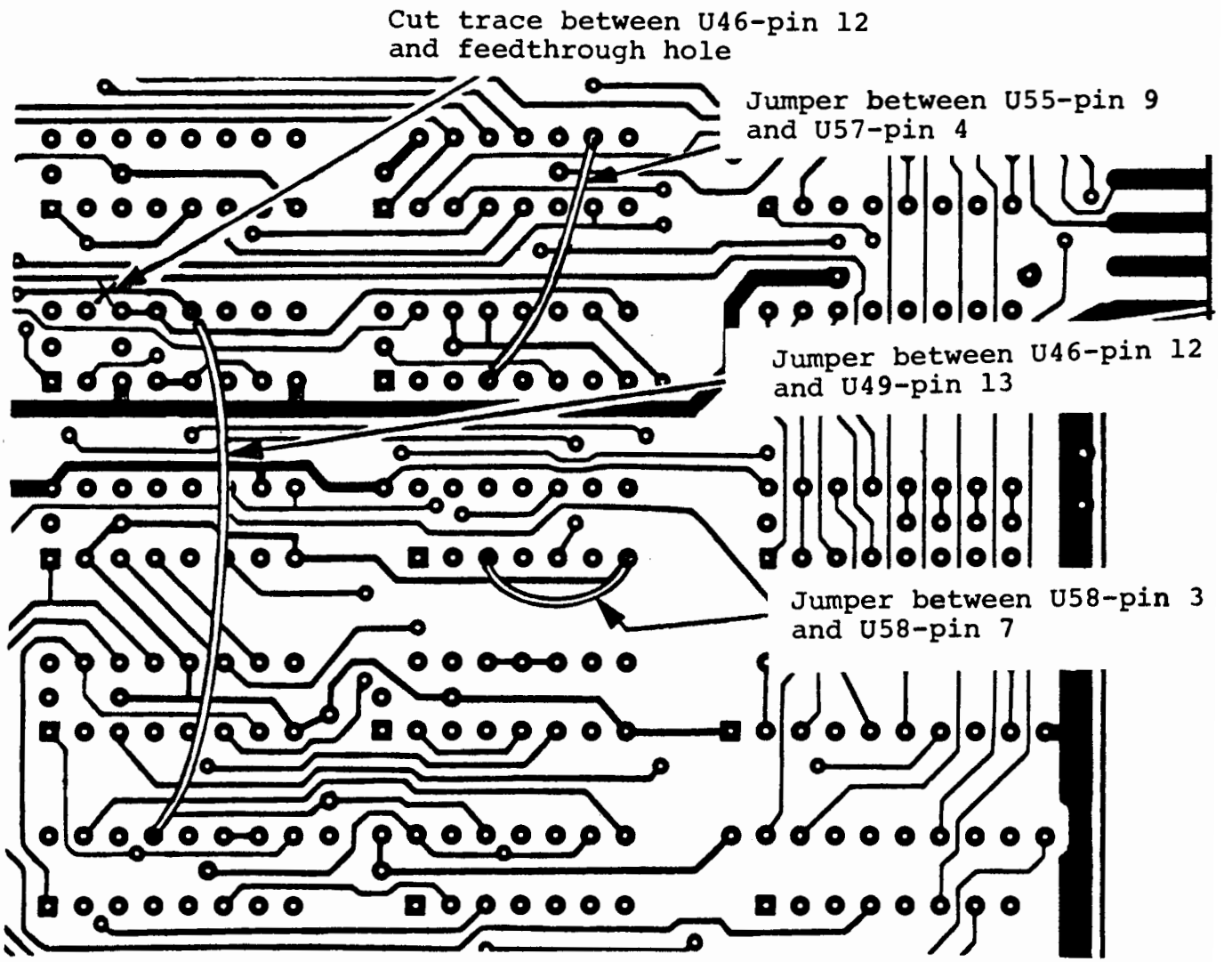


Figure 3. (Shows solder side)

ProcessorTechnology

Processor Technology
Corporation

7100 Johnson Industrial Drive
Pleasanton, CA 94566

(415) 829-2600
Cable Address - PROCTEC

16KRA & 32KRA Update 731Ø66

New modifications to the 16KRA and 32KRA circuit boards involving three trace cuts and three jumpers are being made by the factory. If you receive this update with a new or factory-serviced board, it decribes the modifications that were made. If you receive this update separately, it contains the instructions to make the modifications in the field.

Boards without the changes sometimes have very sharp noise pulses on the four RAS signals during Coincidence cycles. The changes, shown in Figure 1, ensure that the trailing edge of the four RAS signals go high before the four PAGE signals change, by gating U3Ø from a new signal, ME, instead of ME1 and ME2. The new ME signal occurs two gate delays earlier.

If you have a board that is exhibiting problems, it is recommended that you perform the modifications, whether or not you are sure there are noise spikes on the RAS lines. If you have a board that is working reliably, it is not necessary to make the changes. The modifications do not interact with any previous modifications or revision levels, and may be made on any board, without bringing it up to the current revision level. The instructions below assume that you have a 16KRA board that has "REV D", "REV E", or "REV F" etched in copper on the lower right-hand corner of the solder side of the board, or a 32KRA board that is marked "REV B". There is only one small difference between modifying a 16KRA or a 32KRA board, which is explained below.

1) On the component side of a 16KRA board, cut the trace that connects U3Ø-pin 13 and U29-pin 4, and the trace that connects U31-pin 13 and U3Ø-pin 13, as shown in Figure 2. On a 32KRA board cut the trace between U3Ø-pin 13 and the nearby feedthrough pad, and the trace that connects U31-pin 13 and U3Ø-pin 13, as shown in Figure 3.

2) On a "REV F" 16KRA or "REV B" 32KRA, cut the trace on the solder side that goes to U3Ø-pin 4, adjacent to pin 4's pad, as shown in Figure 4. On a "REV D" or "REV E" 16KRA there are two traces to pin 4, instead of the triangle of copper shown. Cut both traces, and solder a jumper on the solder side between U5Ø-pin 13 and U1Ø-pin 13.

3) Using 3Ø AWG solid insulated wire, install the three jumpers on the solder side shown in Figure 4. They connect:

U3Ø-pin 4 to U3Ø-pin 13,
U3Ø-pin 13 to U44-pin 5, and
U29-pin 4 to U31-pin 13.

Ref. ECN 1Ø337

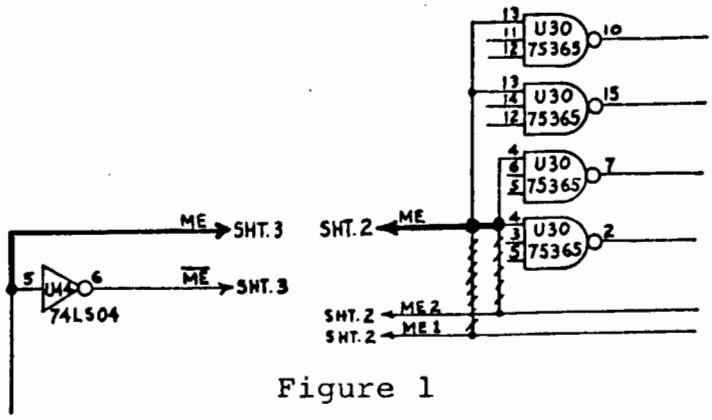


Figure 1

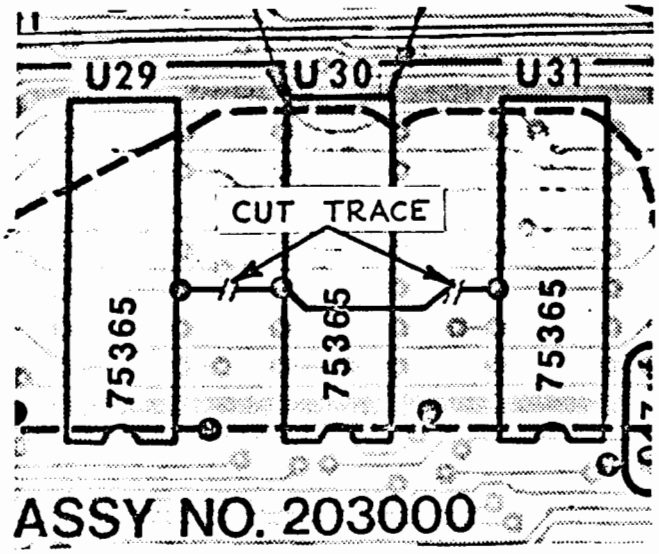


Figure 2

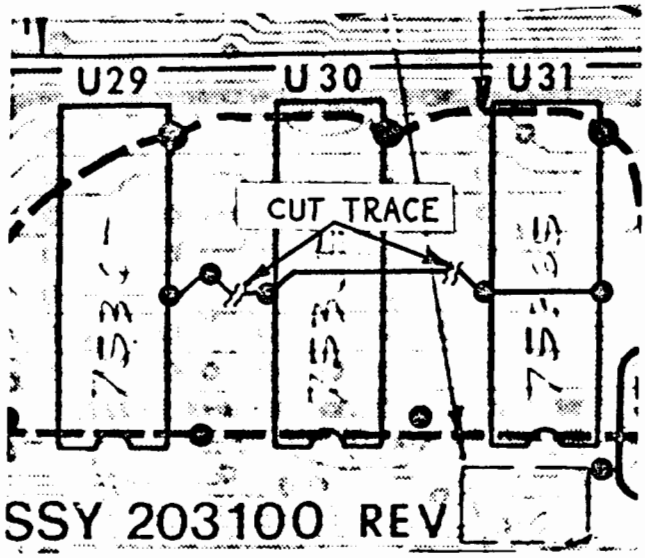


Figure 3

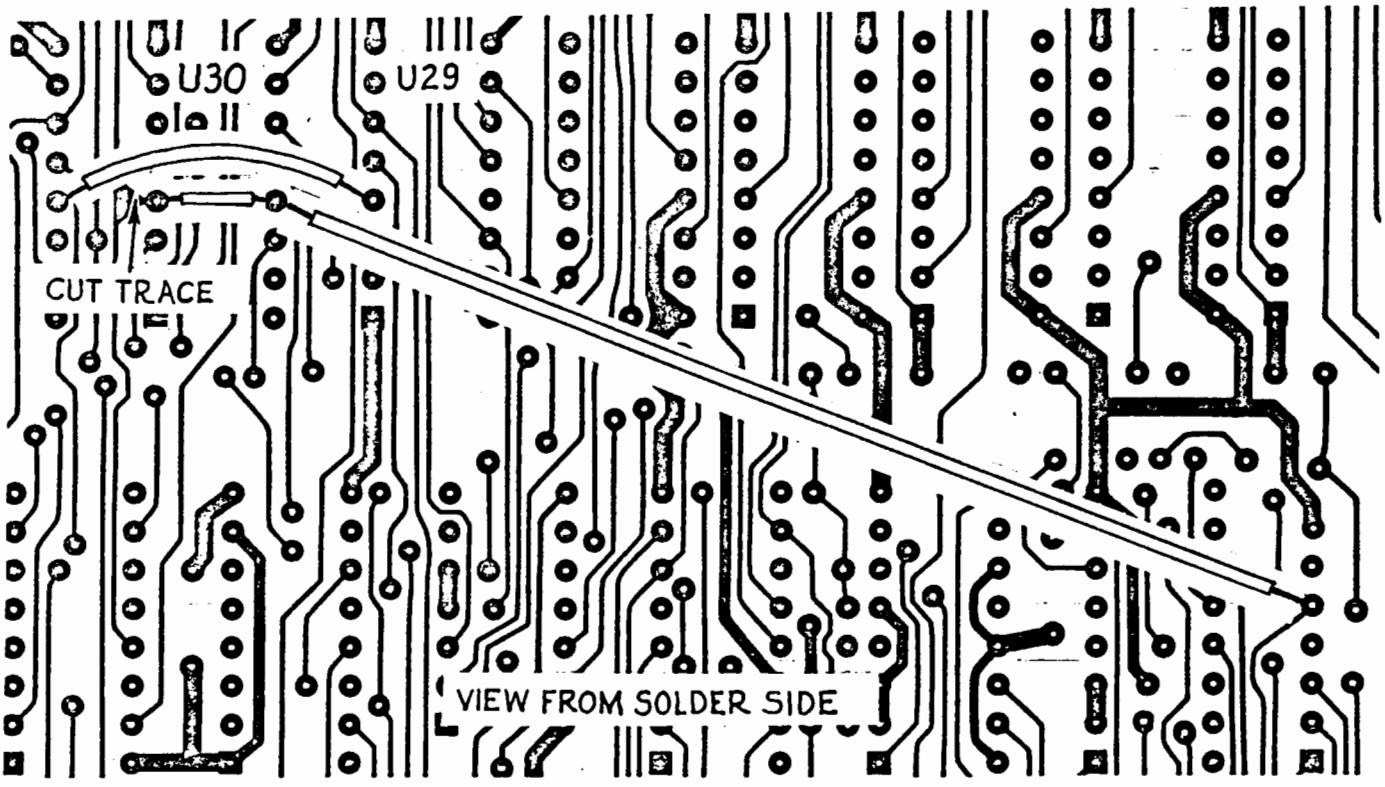


Figure 4

ProcessorTechnology

Processor Technology Corporation
7100 Johnson Industrial Drive
Pleasanton CA 94566
(415) 829-2600
Cable Address PROCTEC

16KRA MANUAL

Change Notice #2

The 16KRA P.C. board you have received with this manual is designated as a Revision K assembly. The portions of the assembly drawing (see page VI-1) printed on the next page show modifications which were made to increase reliability, which are not shown in the manual. These modifications serve two purposes. First, five jumper wires have been added (Detail A) to the trace side of the board, all at ground potential, to improve ground return paths. Second, changed and additional parts are shown (Details B and C) which insure that the timing of the Spontaneous Refresh Timer is within existing specifications, eliminating possible harmless but unnecessary WAIT states. None of these changes affect the circuit description given in the Theory of Operation, Section V, the timing patterns, or the Diagnostic Test, Section VII.

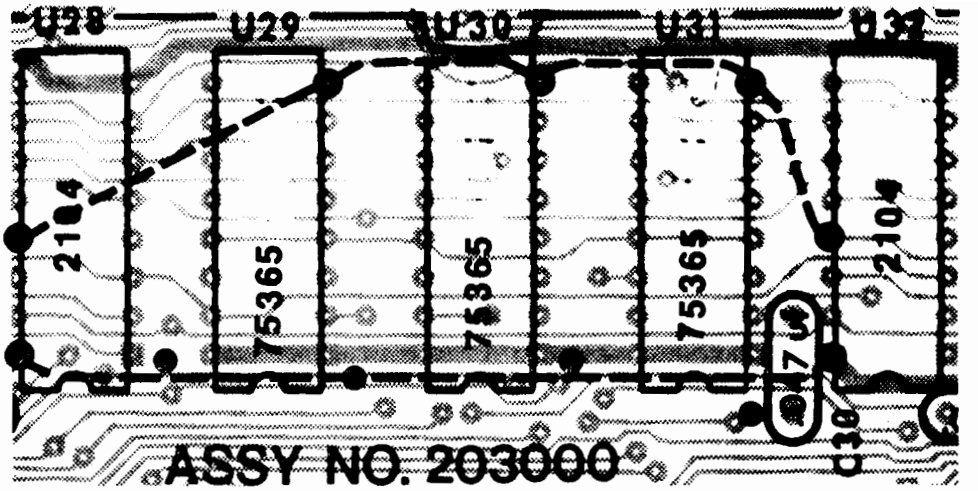
Since most P.C. boards of previous revision levels work reliably without these changes, it is not recommended that the modifications be performed in the field. On the other hand, boards exhibiting intermittent failures might benefit from the five jumpers shown in Detail A, and a 75365 part at U30, if a 3207 was used. If unnecessary WAIT states could cause a problem in a particular application, the additional modifications should eliminate them. These additional modifications involve not only parts changes but three trace cuts and one jumper. Contact the factory for details.

A table showing changed and additional parts used in these modifications is given below. All resistors are 1/4 watt, 5%.

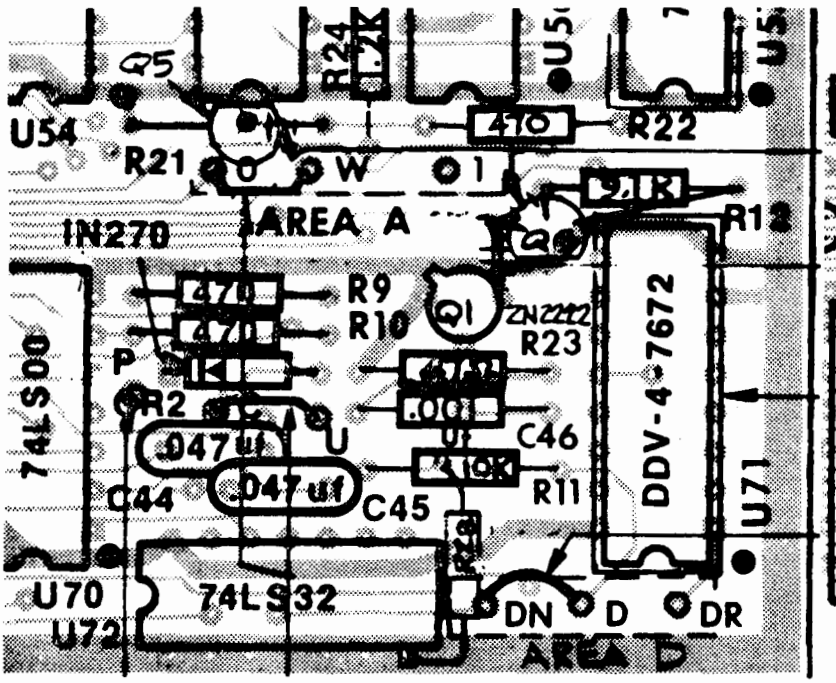
QTY.	VALUE	REF. DES.	NOTES
1	1.5 K	R21	Eliminated
1	76365	U30	Do not use 3207
1	470 ohm	R22	was 3.3 K
1	47 ohm	R23	was 300 ohm
1	10 K	R11	was 22K
1	9.1 K	R12	was 22K
2	2N4274	Q5, Q6	added parts
1	10 K	R28	added part
1	1 K	R29	added part
-	wire	--	used for jumpers

Detail D shows all but one of the changes to the Schematic Diagram (page VI-2) resulting from the modifications. Existing circuitry is shown with light lines; changed values, new parts, and rerouted connections are shown with bold lines. Detail E merely shows that R21 has been eliminated. Please make this note in the area of the main schematic corresponding to Detail D: "See Change Notice #2". Also cross out R21 on the main schematic.

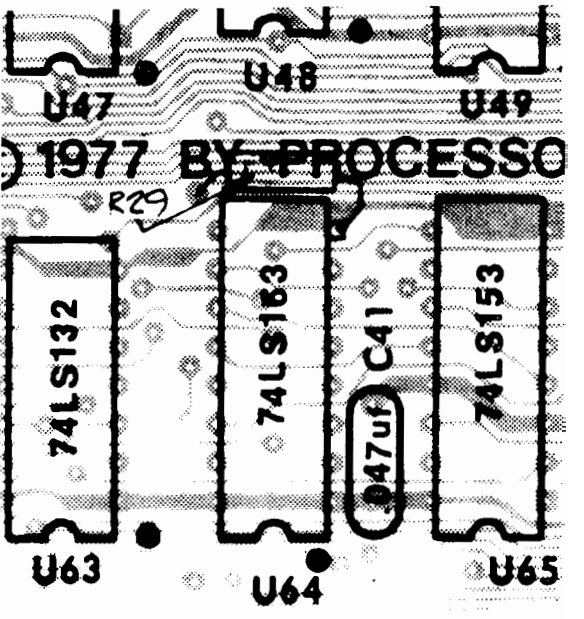
Ref. ECNs 10243, 10247, 10249



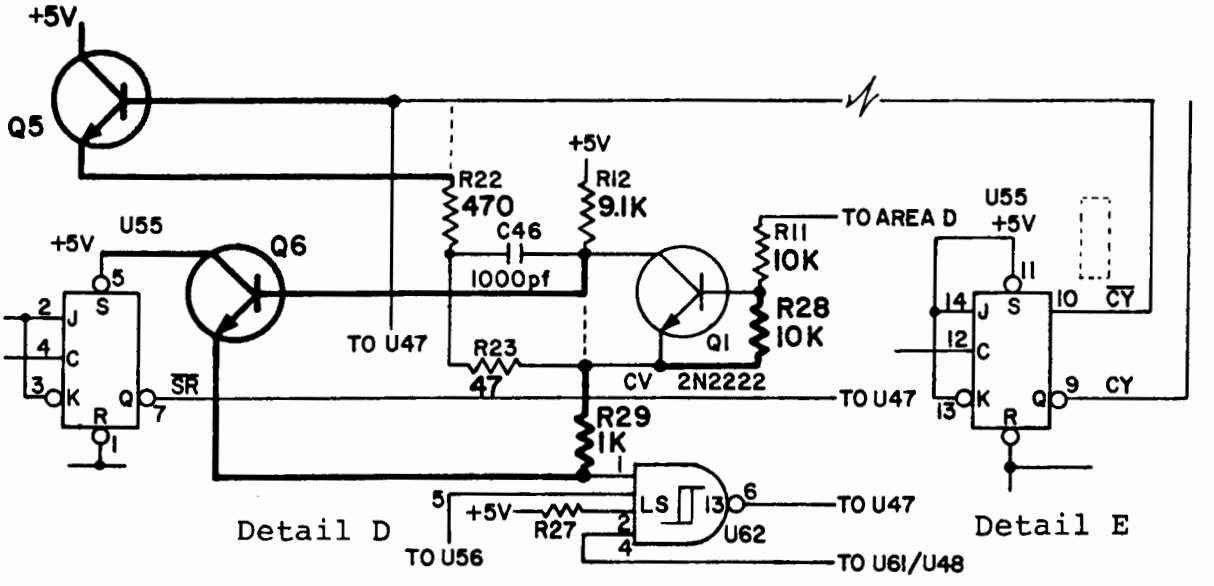
Detail A



Detail B

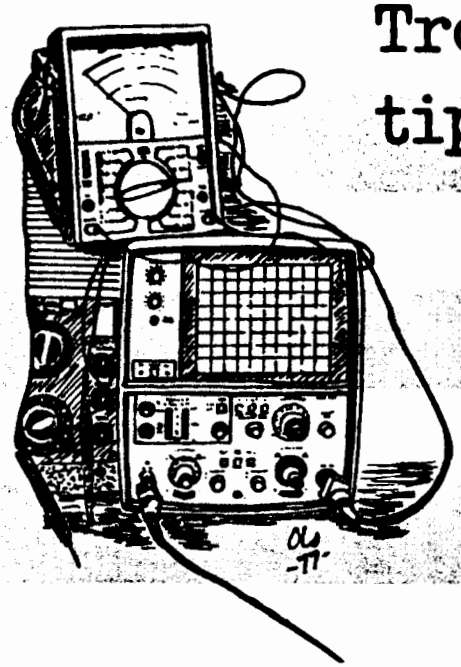


Detail C



Detail D

Detail E



Troubleshooting tips for 16KRA

Excedrin headache number 16

When a 16KRA has a serious headache and you want a systematic treatment to give it fast, fast relief (before you get a headache), try this approach:

Check voltages

1. We recommend using a dual trace oscilloscope capable of triggering at .2 microseconds. Instructions are given below for a single trace scope also.

2. Set the scope at an amplitude of 5 volts per division and check the 7 pin J2 connector. (J2 is located on the bottom lefthand corner of the board.) Starting from P2-1, you should read approximately +16v, -16v (unregulated) and +8v on P2 pins 1, 2, and 3 respectively.

3. If no voltage is present, there is probably an open connection on J2. You can test this by checking the incoming voltages on P2-5, 6 and 7, which should read +8v, -16v and +16v respectively. If you get no voltage here, see page 7.3 of the 16KRA manual for further instructions.

4. Next, check the on-board voltages. Select any RAM chip and measure the voltages at pins 1, 8, 9 and 16. You should read -5v, +12v, +5v and ground respectively. An extra hot heatsink indicates a short.

5. If, when powering up, the monitor is grossly distorted with large dark areas across the screen, +12v is shorted to ground. Frequently such a short is caused by

(continued on page 6)

Troubleshooting tips...

(continued from page 5)
the socket pins of U29, 30 and 31 slicing into the insulation of the Rev J jumper and shorting either +12v or +5v to ground. Simply bend this insulation away from the pin. Please DO NOT use a soldering iron because this voids the warranty. If you still cannot locate the short, send the board back to the factory for repair.

Check delay lines

1. The delay line in socket U71 is vital to the board's operation. A failure (whether hard or intermitent) is often due to the socket itself. Often a simple dump command will clue you into the problem. Type in the command "DU Ø EF <cr>." If a dump does not occur, the delay line is a likely source of trouble.

2. To troubleshoot U71, temporarily remove the D to DN jumper in area D. The board is now in an undefined state; this condition is sufficient for testing the delay line. Another approach is to put the board into a wait state by grounding U68-11. In the wait state, pin 72 of the S-100 bus should be at ground potential.

3. The delay line determines the pulse widths of MC and RC. If you are using a single trace scope, check the signal at U61-8. Whenever it goes high or low, the delay line should follow it. A floating signal (+2 volts) at U57-5, 2, 4 or 1 indicates a bad delay line. The pulse out of the delay line should be a crisp 350 nsec long. Any variation indicates a defective delay line.

4. With a dual trace scope, set the sweep to 100 nsec per division. U61-8 should have a stable pulse of 300 to 350 nsec. Due to the output of NAND gate U57-6, there will be a 100 nsec delay relationship between U61-8 and U57-5.

Trigger on U61-8 to observe the timing relationships. The result should be:

a. a 50 nsec delay between U57-5 and U57-2.
b. a 100 nsec delay between U57-2 and U57-4.
c. a 100 nsec delay between U57-4 and U57-1 with a pulse width of 300 to 350 nsec.

More tips later

Another set of problems can occur on the address lines. I'll discuss these in a future article. 'Til then, take two aspirin and get plenty of rest.

--Chuck Rosas

Processor Technology

Processor Technology Corporation

7100 Johnson Industrial Drive
Pleasanton, CA 94566

(415) 829-2600
Cable Address - PROCTEC

VDM-1 Update 731063

Subject: Errata in User's Manual

Certain improvements were made in the circuitry of the VDM-1 board you have received. Most of the changes are reflected in your User's Manual, but a few points were overlooked. To refer you to the correct information in this Update, make notes in your manual as indicated below.

1) Refer to Section II, page II-13, Figure 2-4.
The frequency of the crystal was changed from 13.5 MHz to 14.31818 MHz. The timing waveforms on Page II-13 are obsolete, and the new correct timing waveforms are given in Figure 2-4 on page 2 of this Update. Before starting assembly, make a note on Page II-13 to refer you to page 2 of this Update.

2) Refer to Section II, page II-14.
The value of R2 was changed from 330 ohms to 100 ohms, and the NOTE at the bottom of Page II-14 no longer applies. Delete the NOTE.

3) Refer to Section III, page III-1, section 3.1.1.
The preset of IC22 was changed from 0 to 11. Information in the second, third, and fifth paragraphs is incorrect. Strike out the whole second and third paragraphs, and the first phrase in the first sentence of the fifth paragraph, and make a note to refer you to the corrected paragraphs below:

"CHARACTER CLOCK, which defines the period of one character, is divided in IC22 and IC21, both of which are presettable four-bit binary counters. IC22 and IC21 start at counts 11 and 3 respectively when pin 6 of NAND gate IC15 is low (the start of each horizontal scan line.) Together, these counters count 102 CHARACTER CLOCKS to define horizontal timing at 64 usec (102 x 628 nsec = 64 usec.)"

"At the start of each horizontal scan line, IC22 counts from 11 to 15 and enables IC21 for one count. IC22 then counts 0 to 15 and enables IC21 for a second count. The sequence continues through five more groups of 16 character postions, and at this point IC21 is at its seventh count (binary 10.) Thus, pins 4 and 5 of IC15 are high, and pin 6 of IC15 goes low. This low loads 11 into IC22 and 3 into IC21 after the next CHARACTER CLOCK, and the cycle repeats. In this cycle, the first count of IC21 (binary 4) defines the left-hand margin of this display, and the fifth count of IC21 (binary 8) defines the right-hand margin of the display."

The following is the corrected first phrase in the first sentence of the fifth paragraph:

"On count 7 (binary 10) of IC21, ..."

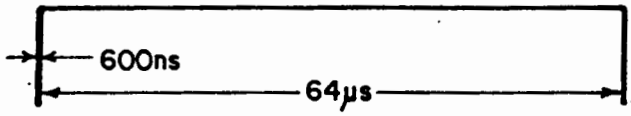
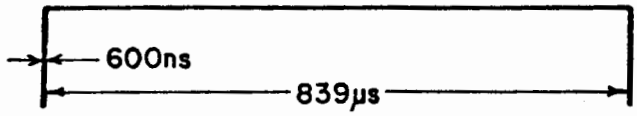
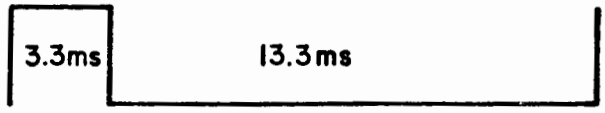
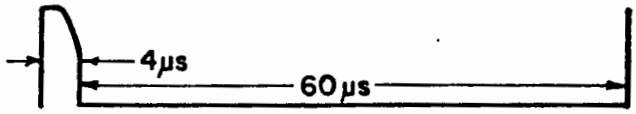
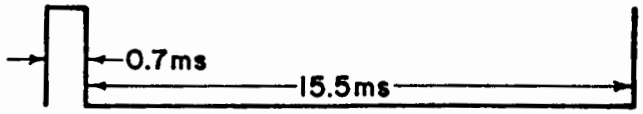
Check Point	Area	Waveform
( ) IC19, Pin 3	D-3,4	14.31818 MHz square wave. Has very rounded corners.
( ) IC20, Pin 12	B-5	
( ) IC15, Pin 6	C-6	
( ) IC 7, Pin 8	B-3	
( ) IC16, Pin 9	C-7	
( ) IC25, Pin 12	E-3	
( ) IC25, Pin 10	E-3	

Figure 2-4. VDM-1 Timing Waveforms.

ProcessorTechnology

Processor Technology
Corporation

7100 Johnson Industrial Drive
Pleasanton, CA 94566

(415) 829-2600
Cable Address PROCTEC

Sol-HyType Interface Update 731076

Subject: SolPrinter Interface

The interface used in SolPrinters 2 and 2E is a SolPrinter Interface Assembly, Part No. 910110, Revision A, made by rewiring the Sol-HyType II Interface (Assembly, Part No. 900044). The new circuit is shown on two drawings attached to this update which replace pages 6-2 and 6-5 of the Sol-HyType Interface User's Manual. Pages that replace those of Section 5, Theory of Operation, are also attached.

The SolPrinter Interface and the Sol-HyType Interface are not interchangeable: the Sol-HyType Interface cannot be used with a SolPrinter and its driver program, and the SolPrinter Interface cannot be used in a HyType II with the former driver program.

SolPrinters 2 and 2E contain the paper out, cover open, and ribbon out detect options. Each of these options involves an extra status line from the printer, whose signal must be made available to the printer driver program. Connections have been made to pass the three signals from the printer back to the Sol's parallel port.

The parallel port of Revision D Sol-PCs has a different arrangement of connections than Revision E and later Sol-PCs. The connections between a Sol-HyType Interface and the parallel port were such that the Interface could be used in D, E, or later Sol-PCs. The three added status lines in the SolPrinter Interface use additional connections, with the result that a SolPrinter Interface may not be used with a Revision D Sol-PC unless the parallel port is rewired like a Revision E Sol-PC or a Sol Parallel Prosthetic (Assembly, Part No. 106060A) is used.

ICs U1, U2, and U10 are no longer used. Their purpose was to raise the printer's ribbon, lowering it if no character had been printed for a period of time, and raising it again before printing the next character. The SolPrinter Interface gives control over ribbon lift to the driver program.

The printer driver program that is printed in the Interface user's manual and was supplied on cassette tape cannot be used with a SolPrinter Interface. The more powerful drivers named "Sol2" and Sol2E", supplied on all WordWizard System Disks and PTDOS 1.5 System Disks, must be used. The use of these drivers with programs other than WordWizard is described in PTDOS Update 731073. The Diagnostic and Exerciser programs also cannot be used.

Since the rewiring that creates a SolPrinter Interface is so extensive, and the Interface is subject to vibration, modification of Sol-HyType Interface boards in the field is not recommended. The Interface is available from Processor Technology as a replacement part, however.

Ref. ECNs 10445, 10468

Processor Technology

Processor Technology Corporation

7100 Johnson Industrial Drive
Pleasanton, CA 94566

(415) 829-2600
Cable Address - PROCTEC

PTDOS Update 731072

This update corrects errors and omissions that had not been detected when the new PTDOS User's Manual went to print. All changes apply to specified pages of the PTDOS User's Manual, Second Edition; they do not constitute an update to the first edition of the same volume.

p iv — The title of Section 6.4.3 should be "RESET,ERRL0,ERRL1, ERRL2."

p 1-3 — Add to the list of system control commands:

$PR Print a string on the CI output file.

p 2-2 — The example in the first paragraph should read:

```
ASSM PROG1,LIST,S=+L
```

Replace the first sentence of 5) with the following:

"A command line may consist of no more than 80 characters and must end with a carriage return."

In the fourth line of 5), strike the words "argument list."

p 2-3
p A3-1 — The syntax of the ASSM command should be

```
*ASSM  source{,list}{,object}{,error}{,symbol}{,S=options}
```

p 2-10
p A3-1 — The syntax of the CREATE command should be

```
*CREATE  filename{,type}{,blocksize}
```

p 2-33
p A3-3 — The syntax of the REATR command should be

```
*REATR  filename{,attributes}{,S=-W}
```

If S=-W is specified, REATR will not request verification of attribute change protection.

p 2-43
p A3-3 — The syntax of the SPACE command should be

```
*SPACE  file,how{,*}
```

p 2-48 — Replace the second line of the $PR description with:

"If a control character is to appear in the output string, it should be represented as a normal character preceded by the ^ character. For example, ^M will cause a CTRL-M (carriage return) to be output."

p 3-1 Replace the first paragraph of Section 3.2 with the following:

"A command has the following syntax:

program{_arguments}delimiter

or program,

where the delimiter may be a carriage return (0DH), a semicolon (3BH), or a NUL (00H).

"The program name is separated from its arguments by one or more blanks, shown above as an underline (_) for clarity.

The second paragraph of Section 3.2 should begin "Program names appearing in a command must be image files ... "

p 3-3 The occurrence of "SRESET" should be changed to "RESET."

p 5-2 Add to the list of file types:

50	IP	Language Processor
D4	T	Text File
FF	D	Device File

p 5-4 Change "2-block sectors" to "two-block tracks."

p 5-5 In the third paragraph change "conditioned" to "initialized." Also, the fourth line of that paragraph should begin "named NEXTID, which .... "

p 5-9 Change "code and data" to "first segment."

p 6-2 The number indicating the bottom of user memory on the memory map should be 0, not 100.

p 6-4 The fifth line in the entry point table should be:

RESET BCB0 Reset system

p 6-5 Operation code 20 should be described as

20 14 SREOP Reset and return (same as RESOP)

p 6-7 All occurrences of "SRESET" should be changed to "RESET."

p 7-3 The message printed if an error occurs during a call to the Explain Error Utility is:

ERROR: CAN'T EXPLAIN

The message printed for an illegal error code is:

ERROR:

without a subsequent CRLF.

p 7-16 The RESET system call has two equivalent operation codes:

RESOP	14	ØEH
SREOP	2Ø	14H

p 8-1 The UXOP control byte (not word) has the following format:

Bit 7:	Ø	RESET after message
	1	normal return to calling routine
Bit 3-6:	X	no effect
Bit 2:	Ø	CRLF before message
	1	no CRLF before message
Bit 1:	Ø	no "CALLED FROM" printed on second line
	1	"CALLED FROM" printed on second line
Bit Ø:	Ø	HL points to a string to be printed on the second line. The string must be terminated by a zero byte. Nothing is printed if HL = -1 (FFFFH).
	1	HL contains a 16-bit quantity to be printed on the second line.

p 8-2 Change both occurrences of "FFFFH" to "FFH"; operation and error codes are single-byte quantities.

The second sentence of the second paragraph should read:

"Error codes 1 through 29H may be returned from system calls."

P 8-3 Change the error message corresponding to error code 14H to MODULE DOES NOT EXIST.

p 8-3 The messages corresponding to error codes 29H and 2AH are
p 1Ø-3 "CATASTROPHE!" and "ILLEGAL COMMAND SYNTAX", respectively.

p 8-4 The first line of text on the page should begin "Module 3 ..." rather than "Segment 3..."

p 9-4 In the description of the calling sequence for the DTCTL driver routine, delete the descriptions of the contents of registers A and B, replacing them with:

A = operation code (the value that was in register B when the CONTROL/STATUS system call was made)

p 9-6 The next to last sentence in the explanation of Control/Status operation code 8 should begin "If the zero flag is cleared ..."

p 9-7 Step 4) should read:

Set the image file type to "D" with RETYPE.

p 9-8 In the discussion of GLUPS and GLBIO, replace "set" with "not equal to zero."

p 10-9 The error codes for ILLEGAL OPERATION should be 10 0AH.

p 10-10 Add the following error message description:

MODULE DOES NOT EXIST 20 14H

This message means that you have called UTIL with a module number that does not exist in the current utility file.

p 10-11 Change the description of NON-RESPONDING DRIVER to:

"This message indicates that a Control/Status request has been made to a device driver that does not support that Control/Status operation. For example, a request to load a random index block (Control/Status request 4) is not meaningful to the device driver for a printer."

p A3-1 The syntax of the COPY command should be

```
*COPY infile,outfile{,S=options}
*COPY O=outfile,infile{,infile...}{,S=options}
```

The syntax of the DISKCOPY command should be

```
*DISKCOPY {/{unit,S={I}{F}{V}{-W} (only one of I, F, and V).
*DISKCOPY {/}unit1,{/}unit2{,S={V}{-W}}
```

p A3-2 The syntax of the FILES command should be

```
*FILES  {/unit}{,filename specifications}{,T=type}
        {,L=order}{,S=options}
```

The following descriptions of commands replace those given in Section 2.2.

BLDUTIL Build or alter a utility file; list modules

```
*BLDUTIL utilfile{,I{module}=filename...}
         {,D=module{,module...}}{,S=L}
```

OPERATION: Modules are added to or deleted from the named utility file according to the arguments entered on the command line. BLDUTIL will create the utility file if it is nonexistent, and will build its directory if it is empty. More information about building and using utility files can be found in Section 6.4.5 of this manual. The format of utility files is described in Section 5.7.4.

ARGUMENTS:

utilfile is the name or number of the utility file to be built or altered.

I{module}=filename

will cause the named file to be inserted into the utility file. The named file must be in image format and have a starting address. If a module number follows "I," the file will be inserted as that module in the utility file. If that module number is already in use, BLDUTIL will ask whether you intend to replace the existing module; if your answer is "Y," the existing module will be replaced by the named file. If no module number follows "I," the file will be inserted at the first available position in the utility file directory. It is possible to insert more than one module by including multiple I=arguments in a BLDUTIL command line. "filename" is the name to be assigned to a new module.

D=module{,module...}

will cause the specified module(s) to be deleted from the utility file.

S=L will cause a listing of the numbers and names of all existing modules to be displayed on the CI output file.

EXAMPLES:

```
*BLDUTIL NEWTIL,I=POTTS,I6=ZAP,D=3,S=L,IØ=ZOT,D=3,4,5
```

COPY - Copy contents of file(s) to another file

```
*COPY infile,outfile{,S=options}
*COPY O=outfile,infile{,infile...}{,S=options}
```

INTERRUPTABLE

OPERATION: In the first form of the command, data are copied from the input file to the output file until an end-of-file is encountered on the input file. In the second form, zero or more input files are concatenated and written to the output file. An input file is never modified by COPY.

ARGUMENTS:

infile is the name or number of a file from which data are copied. In the second form of the command, the named files are copied in the order that they appear in the command line.

outfile is the name or number of the file to which data are copied. If the file does not exist, it is created with type "." and blocksize 1ØØH. Outfile need not have the same type and block size as infile.

The option string may be composed of one or both of the following:

A — means that the infile(s) are appended to the outfile. Otherwise, the outfile is overwritten by the infile.

-E — means that an endfile operation will not be performed on the output file after all data have been copied. Otherwise an endfile operation will be performed. (See the ENDF command, below.)

NOTES: The input file and the output file must not be the same, although a file may be copied to a file that has the same name but resides on a different unit.

This command may not be used if it would shorten a file open under more than one file number; such a file may only be lengthened if designated by the first number under which it was opened.

EXAMPLES:

```
*COPY MAN,MAN/1
*COPY TAIL,HEAD,S=A-E
*COPY O=MAN,MAN0,MAN1,MAN2,MAN3
```

DISKCOPY - Initialize, format, copy, or verify a diskette

```
*DISKCOPY {/}unit,S={I}{F}{V}{-W}  (only one of
                                    I, F, and V)
*DISKCOPY {/}unit1,{/}unit2{,S={V}{-W}}
```

INTERRUPTABLE

OPERATION: Each form of this command will perform one of the following functions:

- Initialize a new diskette for use by the Helios II Disk Memory System
- Format a diskette for use by PTDOS (or erase the diskette)
- Copy the contents of one diskette onto another
- Verify that the contents of one diskette are the same as the contents of another
- Verify that all data on a diskette can be read

ARGUMENTS AND NOTES:

INITIALIZE A NEW DISKETTE - DISKCOPY /unit,S=I

This command writes the proper control signals onto the diskette in the specified unit. Until a diskette has been initialized, the Helios

hardware cannot be used to read from it or write to it. **INITIALIZE A BRAND-NEW DISKETTE BEFORE PERFORMING ANY OTHER OPERATION ON IT.**

FORMAT A DISKETTE - DISKCOPY /unit,S=F

This command writes necessary files onto the diskette in the specified unit, so that PTDOS may use the diskette as a "data disk." If the diskette being formatted already has information on it, that information will be erased; thus, this command may be used to erase a diskette. A diskette need not be initialized prior to formatting.

COPY ONE DISKETTE TO ANOTHER - DISKCOPY /unit1,/unit2

This command makes a track for track copy of the diskette in unit1 to the diskette in unit2. If the destination diskette is brand-new, it must be initialized before this command is given. It is not necessary to format a new diskette before copying to it.

COMPARE TWO DISKETTES - DISKCOPY /unit1,/unit2,S=V

This command compares the data on the diskette in unit1 with the data on the diskette in unit2. If a difference is found, a message is printed and the operation is discontinued.

TEST READABILITY OF A DISKETTE - DISKCOPY /unit,S=V

This command reads every byte on the diskette. If a byte cannot be read, a message is printed and the operation is discontinued.

The -W option may be added to the option string of any of the above commands. If this option is not present, the command will require a carriage return between the time that it is first entered and the time that its function is actually performed; this feature is useful, because it enables the user to make a copy of a diskette not in the drive when the command is given. At any time before the carriage return, diskettes may be removed from the drive and replaced with other diskettes. In general, the -W option is used in situations in which user intervention is not desired: for example, in a command file to be executed with the SETIN command or the DO command macro preprocessor (see Section 4). This option also affects the handling of READ errors (see below).

ERRORS AND LIMITATIONS:

It is not possible to format or copy onto disk unit 0 or the default unit. A diskcopy from a unit to itself is also unacceptable.

WRITE ERRORS will result in a retry of the write operation. The retries will continue until a successful write occurs or the operation is aborted with the MODE key. A write error is an indication of a hardware problem; the diskette, the disk drive, or the controller electronics may be at fault.

READ ERRORS are handled according to the setting of the W option. If the -W is set, the error will be displayed on the console and the bad sector(s) will be ignored. Otherwise, the program will ask if a retry is desired and will continue to try until an "N" answer is given to that enquiry.

EXAMPLES:

```
*DISKCOPY /0,/1
*DISKCOPY /1,S=I
*DISKCOPY 0,1,S=V-W
```

FILES - Display list of files

```
*FILES {/unit}{,filename specifications}{,T=type}
       {,L=order}{,S=options}
```

INTERRUPTABLE

OPERATION: A list of files is printed on the CI output file. The arguments dictate which files will appear in the list; if a file matches more than one argument, it will still be listed only once.

Each entry in the list consists of the name of a file, the file type, the number of 256-byte sectors allotted to the file (one 4C0 block is equivalent to four 256-byte sectors), the block size, the file ID, the sector and track on which the first block of the file is recorded, the protection attributes of the file, and the location of the index block. (There is no index block if the file is not a random access file; also, the attribute field will be blank in many cases, because many files will not have any protection attributes.)

ARGUMENTS:

/unit — specifies the unit whose directory is the source of the list. If this argument is not present, the default unit is used. Only files that exist on the specified or default diskette will be listed.

filename specifications — In this command and in several others, a string may be used to represent all files whose names contain that string, or contain the string in a specific position in the name, for example, at the beginning. If one or more of these arguments are present in the command, only files whose names are identified by the string(s) will appear in the list. Otherwise, the names of all files that meet the other requirements will appear. Strings may be typed in upper or lower case, i.e., "NAME" and "name" designate the same file.

Filename specifications may take any of the following forms:

string	may be any legal PTDOS file name, not including a unit number. If a file with this name exists, it will be included in the list.
string>	causes all files whose names begin with the string to be included in the list.
<string	causes all files whose names end with the string to be included in the list.
<string>	causes all files whose names include the string to be included in the list.

T=type

If this argument is present, only files of the given type will appear in the listing. Specify image files by preceding the type with the letter "I"; specify types that are non-printing by preceding the hexadecimal value of the type with a #. T=#5 signifies a type whose hex value is 5; T=5 signifies a type whose value is the ASCII character 5.

L=order

This argument specifies the listing order for each group of files defined by a filename specification. "Order" may be one of the following codes:

- N Alphabetically by File Name (default)
- I Numerically by File ID
- T Alphabetically by File Type (secondary ordering by name)
- D By order within Directory
- A Numerically by Disk Address
- B Numerically by Block Size (secondary ordering by name)

S=options

The option string may be composed of one or more of the following:

- -H Suppress column headings on the listing.
- -I List selected files even if they are information-protected.
- Q List only the names of the selected files.

EXAMPLES:

```
*FILES /1,S=-I,T=#00
```

List all files of type 00, whether or not they are information-protected.

*FILES <AID>,<AER,KNOCK

List the file called KNOCK, and also any files that contain the string "AID" or end with the string "AER."

*FILES /1

List all files on unit 1, except information-protected files.

*FILES S=-I-H,SYSGLOBL

List the file SYSGLOBL if it is present on the default unit. The information will be listed even if the file is information-protected, and no column headings will be printed.

The DEBUG manual does not describe the screen save option, which is very useful for debugging interactive programs. If screen saving is enabled, the contents of the video display are saved when a breakpoint is encountered and restored when execution is resumed. The user must specify a 1K-byte area of memory in which to save the screen; this specification is made with the $Y addr command, in which addr is the hex address of the first byte of the save area. The $Q command enables screen saving, or disables it if it is already enabled.

ProcessorTechnology

Processor Technology Corporation
7100 Johnson Industrial Drive
Pleasanton, CA 94566
(415) 829-2600
Cable Address - PROCTEC

PTDOS & WordWizard Update 731074

Subjects: Revision Levels of PTDOS 1.5 System Disk
Revision Levels of WordWizard System Disk
Revision Levels of WordWizard Document Disk

The purpose of this update is to describe changes made on PTDOS 1.5 and WordWizard disks since their initial release in November, 1978. Information about "current" releases is current as of December 28, 1978.

REVISION LEVELS OF PTDOS 1.5 SYSTEM DISK

The first revision level of the PTDOS 1.5 System Disk was Revision C; the current level is Revision E.

Revision C

The Revision C disk contained the following software:

PTDOS resident files and commands	Release 1.5 (mod 0)
ASSM	Release 1.1 (mod 0)
EDIT	Release 1.1 (mod 0)
EDT3	Release 1.1 (mod 0)
DEBUG, DEBUG3	Release 1.1 (mod 0)
Extended Disk BASIC and related programs	Release 1.1 (mod 0)
FOCAL	Release 1.1 (mod 0)
CTAPE source and object	Release 1.0 (mod 0)
TREK80	Release 1.0 (mod 0)
XREF	Release 1.0 (mod 0)
Sol2	Release 1.0 (mod 0)
Sol2E	Release 1.0 (mod 0)
Sol3	Release 1.0 (mod 0)

Extended Disk BASIC is described in the Extended Disk BASIC User's Manual, Second Edition. Sol2, Sol2E, and Sol3 are device drivers for three SolPrinters and are described in PTDOS Update 731073. All other software on the disk is described in the PTDOS User's Manual, Second Edition.

Revision D

The following changes were made to update the disk to Revision D:

PTDOS resident files and commands were updated from Release 1.5 (mod 0) to Release 1.5 (mod 1). This change involved the correction of a bug in the GET command. (In order to transfer a device file, the GET command first changes the type of the file, then transfers the data, and finally changes the type back to D; the bug prevented the type

from being changed back to D if the attempt to transfer the file was unsuccessful.)

Sol3 was updated from Release 1.0 (mod 0) to Release 1.0 (mod 1). This change involved the implementation of the Vertical Increment operation (described in Update 731073) and the correction of a bug that caused the driver to drop the first character sent to it.

Revision E

The following changes were made to update the disk to Revision E:

Sol2 and Sol2E were updated from Release 1.0 (mod 0) to Release 1.1 (mod 0). Sol3 was updated from Release 1.0 (mod 1) to Release 1.1 (mod 0).

Several changes were made to update the three drivers:

1) In previous versions of the drivers, the tilde character (shifted ^) could not be printed. In the current release, the tilde character can be printed if the driver is not operating in Word Processor mode (see Update 731073).

2) In previous versions of the drivers, a formfeed character was printed as a question mark. In the current release, a formfeed causes the platten to be rotated until the carriage is at the top of a form.

3) In previous versions of the Sol2 and Sol2E drivers, there was a bug precluding a horizontal increment of 1024/120" or more. (The most common symptom was that the carriage would slam suddenly to one side and the printer would stop.) In the current release, that bug has been corrected.

4) In previous versions of the Sol2 and Sol2E drivers, a NUL character was absorbed. In the current release, the drivers respond to the NUL by performing any "collected" paper and carriage movement (see Update 731073).

5) In previous versions of the drivers, the apostrophe (shifted 7) was printed as an acute accent (´). In the current release, the printed character is actually an apostrophe (').

6) In previous versions of the Sol3 driver, a call to close the driver did not cause the internal buffer of the SolPrinter 3 to be flushed; as a result, the next call to open the driver would result in the printing of whatever characters had been left in the buffer. In the current release of Sol3, a call to close the driver causes the internal buffer of the SolPrinter 3 to be flushed.

REVISION LEVELS OF WordWizard SYSTEM DISK

The first revision level of the WordWizard System Disk was Revision A; the current level is Revision D.

Revision A

The Revision A System Disk contained the following software:

WordWizard System	Release 4.0 (mod 0)
PTDOS/WordWizard version	Release 1.5 (mod 0)
Sol2	Release 1.0 (mod 0)
Sol2E	Release 1.0 (mod 0)
Sol3	Release 1.0 (mod 0)
WIZ	Release 1.0 (mod 0)

All of these programs are described for WordWizard users in the WordWizard User's Manual, First Edition, and in WordWizard Update 731075. Sol2, Sol2E, and Sol3 are device drivers for SolPrinters and are also described in PTDOS Update 730173.

Revision B

The following change was made to update the disk to Revision B:

Sol3 was updated from Release 1.0 (mod 0) to Release 1.0 (mod 1). This change is identical to that made in Sol3 on Revision D of the PTDOS System Disk (see above).

Revision C

The following change was made to update the disk to Revision C:

PTDOS/WordWizard version was updated from Release 1.5 (mod 0) to Release 1.5 (mod 1). This change is identical to that made in PTDOS on Revision D of the PTDOS System Disk (see above).

Revision D

The following changes were made to update the disk to Revision D:

Sol2 and Sol2E were updated from Release 1.0 (mod 0) to Release 1.1 (mod 0). Sol3 was updated from Release 1.0 (mod 1) to Release 1.1 (mod 0). These changes are identical to those made in the drivers on Revision E of the PTDOS System Disk (see above).

REVISION LEVELS OF WordWizard DOCUMENT DISK

The first revision level of the WordWizard Document Disk was Revision A; the current level is Revision C.

Revision A

The Revision A Document Disk contained the following software:

WordWizard Document	Release 4.0 (mod 0)
Sol3:A source	Release 1.0 (mod 0)
WIZ:A source	Release 1.0 (mod 0)

These items are described in the WordWizard User's Manual, First Edition, and in WordWizard Update 731075.

Revision B

The following change was made to update the disk to Revision B:

Sol3:A was updated from Release 1.0 (mod 0) to Release 1.0 (mod 1) to reflect the corresponding change made in Sol3 on the WordWizard System Disk.

Revision C

The following change was made to update the disk to Revision C:

Sol3:A was updated from Release 1.0 (mod 1) to Release 1.1 (mod 0) to reflect the corresponding change made in Sol3 on the WordWizard System Disk.

ProcessorTechnology

Processor Technology
Corporation

7100 Johnson Industrial Drive
Pleasanton, CA 94566

(415) 829-2600
Cable Address - PROCTEC

WordWizard Update 731075

Subjects: SPACE-AVERAGING ON SolPrinters 2 AND 2E
ERRATA TO WordWizard User's Manual
SPECIFICATIONS FOR CUSTOM PRINTER DRIVERS
WIZ - THE ELECTRIC PENCIL SHARPENER

SPACE-AVERAGING ON SolPrinters 2 AND 2E

If you intend to use SolPrinter 2 or 2E, it is important that you understand space-averaging and how it affects the appearance of your copy.

When you type a document using the default justification mode (ON), WordWizard adds extra spaces between some words to make the right margin of your paragraph even. If you have already used WordWizard, you have surely noticed this effect on the screen: within a line the size of the gaps separating words will not necessarily be constant. On the other hand, when you print a justified document on the SolPrinter 2 or 2E, the spacing between any two words WILL be constant. The reason for this readjustment is "space-averaging."

Space-averaging works on both justified and unjustified text. The amount of space between any two words in a line is made constant, but the line continues to occupy exactly the same number of character positions that it did before. For example, if the line looks like this on the screen:

This is a line.

space-averaging will make it look like this when it is printed:

This is a line.

Notice that the length of the line has not changed, but the amount of blank space between words has been averaged. The calculation of the average between-word space is straightforward: the total number of blanks (i.e., spaces) in the line, except those occurring before the first non-blank character or after the last non-blank character, is divided by the number of gaps (some of which may consist of multiple spaces). In the example above, there are 6 blanks in the line, and 3 gaps, so a gap in the printed line will be 2 blanks wide. If there were 5 blanks in the line, as it appeared on the screen, each gap in the printed line would be equivalent to 1-2/3 blanks.

There are two characters that have an effect on space-averaging. The effect of the inverted L character is simple, but important: space-averaging does not occur on any line that ends with the inverted L. If a line of your document is supposed to contain gaps of unequal sizes, end that line by pressing RETURN. If a printed line no longer

has the format you intended (and if the same line has the correct format on the screen), you can be almost sure that you forgot to end the line with a RETURN.

Remember that the inverted L will also prevent a line from being justified while you type. If you want a line to be justified but not to have space-averaging, type the line without an inverted L and later - any time before printing - move the cursor to that line and press RETURN. Adding an inverted L to a line that you have already typed will not change the format of that line in your document.

The other character to affect space-averaging is the tilde (the shifted ^). The WordWizard User's Manual tells you to use a tilde to preserve a blank space that might otherwise be eliminated by a CLOSE PARAGRAPH command. For example, you might use tildes to preserve the two blanks after a period or a colon. When the printer performs space-averaging on a line that contains tildes, it counts each tilde not only as a blank, but also as a gap, so that two tildes in a row, or a tilde and one or more blanks, will result in a gap twice the size of the other gaps in the line. For example:

```
                is  a line.~  This
and             is   a line. ~This
and             is a   line.~~This
```

will all be printed as

```
                is  a line.   This
```

The effect of separating two words by one tilde is exactly the same as that of separating them by one or more blanks.

ERRATA TO THE WordWizard User's Manual

Please incorporate the following information on the indicated pages of your manual:

On page 1-4

In the second line of the fourth paragraph of Section 1.3.2, change both occurrences of the word "one" to "three."

On page 2-3

In the last line of the diagram, change the symbol <CR> to the word RETURN. On the Select Printer Menu, insert the item

```
2  SolPrinter 2E
```

and change the numbering of items later on the menu to 3, 4, and 5, respectively.

On page 2-5

Replace the last sentence of the second to last paragraph with:

"If you select function 1 or 5 by mistake, press the ESCAPE key to return to the System Functions Menu. If you select function 2, 3, or 4

by mistake, or if you decide to discontinue one of those operations, press the ESCAPE key to return to the starting prompt (>), and then give the BOOT command to restart the WordWizard."

On page 2-6

Under the heading "Make a new System Disk," change "Processor Technology" to "the WordWizard."

On pages 2-7 and 2-9

Change "Erase Drive No. 1" to

Initialize Unit 1
Initialization Complete

Change "Verify Disks" to

Compare Units 0 and 1

On pages 2-11, 2-12, and 4-2

Insert the following note in the descriptions of the HOME CURSOR, MODE SELECT, and LOAD keys, and also after the last sentence of Section 4.1:

"Between the time you press HOME CURSOR or MODE or LOAD and the time you press the next key (with which you want HOME CURSOR or MODE or LOAD to operate), the cursor will "blink." The only purpose for this blinking is to let you know that WordWizard is expecting you to press another key."

On page 2-16

In the second paragraph change 7000 to 7040.

On page 3-5

Add the following paragraph:

3.4 PROTECTING CONFIDENTIAL DOCUMENTS

In Section 2 of this manual you learned about the Remove Activity, which removes the name of a document from the Document Index and allows the associated disk space to be used for other text. You will notice that after you Remove a document, you will have fewer "pages left on disk" than you had previously. If you are concerned about the confidentiality of your documents, be aware that Removing a document does not cause the information to be erased from the disk. The text remains on the disk (albeit inaccessible through WordWizard) until other text is written over it or the disk is used as a "scratch disk" by one of the System Functions.

On page 7-3

Add the following entries to the column labeled "Default":

For .header lines, the number 6
For .bottom lines, the number 0
For .number, the number 0
For .spacing, the number 1

SPECIFICATIONS FOR CUSTOM PRINTER DRIVERS

WordWizard is capable of sending output to any printer that can be interfaced physically to the Sol, and logically to PTDOS and WordWizard. Driver programs for the three SolPrinters and the Diablo 1610/1620 are recorded on the WordWizard System Disk; if you intend to write a driver for some other printer, take note of the following specifications:

1) The custom driver must have the form and characteristics of a PTDOS device file, as described in Section 9 of the PTDOS User's Manual, Second Edition. DTBLK and DTITO must both have values of 1 in the driver table.

2) The driver must support the following Control/Status operation calls (numbers before the descriptions are operation codes):

   7 - Special Status Read (described in the PTDOS User's Manual)

   100 - Word Processor Mode

   Entry: E = non-zero to set Word Processor Mode
   E = 0 to clear Word Processor Mode
   Return: A = non-zero if Word Processor Mode was set before the call
   A = 0 if Word Processor Mode was clear before the call

   The driver must be initialized with Word Processor Mode clear.

   103 - Horizontal Increment

   Entry: E = new horizontal increment in 1/120 in.
   Return: A = old increment

   104 - Vertical Increment

   Entry: E = new vertical increment in 1/48 in.
   Return: A = old increment

   105 - Cancel Internal Buffer Contents

   Entry: none
   Return: none

3) When the driver operates in Word Processing Mode, as it always will with WordWizard, any call to the driver must be serviced within 1/10 second: that is, there must be a return to the

calling program after 1/10 second, whether or not a printing operation has been completed. If this specification did not exist, WordWizard could not be used for simultaneous printing and editing. When the driver is not in Word Processor Mode, i.e., when it is not being used with WordWizard, it must complete the operation for the current request before returning to the calling program.

4) When in Word Processor mode, the driver must handle certain characters in special ways:

   A carriage return (0DH) must generate an automatic linefeed (0AH), unless an FFH character has been encountered since the previous carriage return. The FFH character causes the linefeed to be suppressed. (The .under statement has the effect of inserting an FFH character, so that two lines will be "over-printed.")

   A linefeed (0AH) must be absorbed by the driver, i.e., there should be no action by the printer driver in response to a linefeed.

   An FFH character must cause a linefeed not to be generated after the next carriage return (see discussion of carriage return, above).

   The forced line ending character (01H) must be absorbed by the driver, rather than sent to the output device. (In the drivers for the SolPrinter 2 and 2E, the forced line ending also causes space-averaging to be suppressed.)

   The tilde character (the shifted ^) must be printed as a white space.

   All non-printable characters, other than those described above, should be printed as question marks (?).

5) The driver must reside completely between addresses 7800H and 8000H.

6) The object code for the driver must be stored in the file called CUST on the WordWizard Document Disk. Thereafter, when the user selects the fifth item from the Select Printer Menu, the custom printer will be selected: that is, the contents of CUST will be copied to a device file called PRINTER, to which WordWizard actually directs all printed output. (Do not store the code for your driver directly in PRINTER, because any selection from the printer menu will cause that file to be overwritten.)

The assembly language source code for the SolPrinter 3 driver is recorded, as an example, in the file called Sol3:A on the WordWizard Document Disk.

WIZ - THE ELECTRIC PENCIL SHARPENER

WIZ is a program that changes the structure of files developed with a program called the Electric Pencil I or II, so that such files may be edited with the WordWizard. If you are not now using either the Electric Pencil I or the Electric Pencil II, you need not read this item.

The object code for WIZ is recorded, with that filename, on the WordWizard System Disk, and will also be recorded on any System Disk made with the "Make a new System Disk" function. The assembly language source file, WIZ:A, is recorded on every Document Disk supplied with WordWizard, but will NOT be recorded on any Document Disk made with the "Make a new Document Disk" function. The source file is provided so that you can modify it, if your needs require.

The need for WIZ is due to differences between the file structures of an Electric Pencil file and a WordWizard document. Electric Pencil maintains text as a continuous stream of characters, with carriage returns only at the end of paragraphs (where the user types a linefeed). WordWizard maintains individual lines, each consisting of 1) the characters of the line as it appears on the screen, and 2) a final carriage return (0DH). Lines on which the user typed the RETURN character (producing the "inverted L" at the right boundary of the screen) have a CTRL-A character (01H) immediately preceding the carriage return.

To execute WIZ, proceed as follows:

1) Bootload PTDOS by inserting a PTDOS System Disk in unit 0 and typing

   BOOT RETURN

   (RETURN signifies pressing of the RETURN key).

2) Remove the PTDOS System Disk from unit 0, and insert a WordWizard System Disk. Insert the disk containing the Electric Pencil file into unit 1. (If your Electric Pencil file is on cassette, you will have to read it into memory and write it out to disk with the PTDOS WRITE command before you execute WIZ. The text buffer for the Electric Pencil begins at address E7AH.)

3) Type

   WIZ filename/l,nn:D RETURN

   where filename is the name of the Electric Pencil file, and nn is a decimal number specifying the maximum line length for the destination file. That number may be no smaller than 0, and no greater than 127; if the argument is absent, a value of 63 (the default line length for WordWizard) is used.

WIZ divides the continuous string of the Electric Pencil file into lines no longer than nn and copies each resulting line into the archive file (called ARCH) on the WordWizard System Disk. Each carriage return (resulting from a user-typed LINEFEED) or formfeed

(resulting from a user-typed LOAD) in the original file becomes the sequence CTRL-A/carriage return in the destination file, so that the paragraph and forced line endings of the original file are preserved in the destination file. (WIZ does not actually alter the original file.)

To transfer the contents of ARCH to a WordWizard Document Disk, proceed as follows:

1) With the WordWizard System Disk in unit 0 and a Document Disk in unit 1, execute WordWizard by typing

    BOOTLOAD RETURN

    after the PTDOS prompt (*), or

    BOOT RETURN

    after the starting prompt (>).

2) When the Activity Menu appears on the screen, select the Retrieve Activity, and specify the name you want the document to have.

When the Retrieve operation is complete, the name of the new document will appear in the Document Index, and the document may be examined and edited like any other WordWizard document. When you edit the document, you will notice that it is not justified, and that the length of a line will never be greater than the line length you prescribed.

ProcessorTechnology

Processor Technology
Corporation

2100 Industrial Drive
Pleasanton CA 94566

(415) 829-2600
Cable Address PROCTEC

WordWizard Update No. 3

April 3, 1979

MANUAL UPDATE

MANUAL: WordWizard User's Manual, First Edition,
Manual Part No. 727211

SUBJECTS: WordWizard 4.D (mod 0)
Offset Print Control Statement
Center Command
Center and As typed Print Control Statements
Screen Print Activity
Type-ahead Feature
Archiving Documents for PTDOS
Menu Changes
New Printer Drivers
New Version of WIZ - THE ELECTRIC PENCIL SHARPENER
Technical Notes

CURRENT PUBLICATIONS: 731075

WordWizard 4.D (mod 0)

This update supplements the WordWizard User's Manual, First Edition, in describing a new version of the WordWizard System Disk, marked REV F on the diskette label. The REV F diskette contains the same programs as the previous release, except that WordWizard 4.D (mod 0) and WIZ 1.1 (mod 0) are new releases, and new printer drivers are included. Document disks prepared using previous releases may be used with the new REV F System Disk. However, Document Disks prepared with the new release are incompatible with all previous releases. The assembly language source file, WIZ:A, corresponding to WIZ 1.1 (mod 0) is recorded on new REV D WordWizard Document Disks.

When WordWizard is first started using the BOOT command, two messages appear briefly on the screen before the menu appears, identifying the releases of PTDOS and WordWizard that are present on the WordWizard System Disk. These messages, in conjunction with the marking on the diskette label, identify versions of the software that the diskette contains. This particular release is easily identified by the fact that the BOOT command leads to the System Functions Menu rather than the Activity Menu, and that the Activity Menu contains a new activity: "S. Screen Print."

The material below describes the features of the new software. With the exception of the menu changes, the new features are enhancements to the WordWizard program, rather than modifications to existing

features, therefore, this update consists of additions rather than corrections to the WordWizard User's Manual.

Offset Print Control Statement

The form of the Offset Statement is

```
.offset n        |_
```

where n is the number of columns by which subsequent lines will be offset to the right when printed. The default value of n is 0 (no offset). n may range from 0 to 31.

When used at the beginning of a document, this statement allows you to create a left margin on each printed page of the document without having to offset the paper in the printer, and without having to use an indented left margin while editing the document.

Here is a procedure for using the statement to create a left margin:

1) Determine what left margin your entire document needs, in tenths of an inch if it will be printed at 10 characters per inch, or in twelfths of an inch if it will be printed at 12 characters per inch. Count from the edge of the paper to where the first column of printing should appear.

2) Place the statement

```
.offset n        |_
```

at the beginning of your document, using for n the number you determined above for the margin.

3) SolPrinters 2, 2E, and 3 have a scale on the cover over the print head. Before printing, position the paper in the printer so that the left edge of the paper lines up with the left edge of the scale, at "zero". The edge of the paper will also line up with the red line on top of the print head of a SolPrinter 2 or 2E, if the print head is at its leftmost position. If you have some other printer, line up the left edge of the paper so that it is at the leftmost printable position.

4) When you edit your document, set the left margin at column 1 (left margin released). When you print the document, the text as it appears on the screen will be shifted to the right by n columns, to provide for a left margin n columns wide. The right-hand end of the line is offset too, so choose a right margin accordingly. For example, if you are typing a letter with the default margins (63 characters per line), and use ".offset 10" for a one inch margin, you will have a right margin of 1.2 inches. (8-1/2 inch wide paper is 85 columns wide at ten pitch. Printed lines will go from columns 10 to 73, leaving 12 columns, 1.2 inches, for the right margin.)

Normally, when you want an indented passage in a document, you indent the left margin before typing the passage or before using the CLOSE PARAGRAPH command. Thus the indent is visible on the screen. You can also insert an Offset Statement in the middle of the document to

create an indent. In this case, the indent will not be visible on the screen but will be evident when the document is printed. Remember that the Offset Statement shifts both ends of the line equally, while the LEFT MARGIN SET affects only the left margin. It will be necessary to reset the right margin when using the statement this way to provide for an even right margin in the printed document. Use both kinds of indents in one document can be confusing.

To cancel an indent (revert to an indent of 0), insert the statement "offset " (no n).

Center Command

You can center titles and other single lines of text with a new Center command. To use the command, just type the text you want centered anywhere on the line. If you want more than one blank separating words, use tildes to preserve the extra blanks. Then, with the cursor still on the same line, press the MODE key, then the caret (^) key. (This command is not shown on the keyboard labels.) The text will be centered within the current margin settings. Multiple blanks will be removed as if the CLOSE PARAGRAPH command had been given too--thus the need for tildes. You can redefine the "center" by resetting the left or right margin. However, all of the text to be centered must be within the margin settings, or characters will be "left behind" when the Center command is given.

Center and As typed Print Control Statements

The forms of the Center and As typed Statements are

```
.center n
.as typed
```

where n is a column on the printed page about which subsequent lines of text should be centered, and the "As typed" statement cancels the effect of a previous Center Statement--subsequent lines are no longer centered. The command ".center " (no n) centers text within the default margins, and is thus equivalent to the command ".center 31". Multiple blanks are removed from centered lines as if the CLOSE PARAGRAPH command had been given.

The primary use of the Center Statement is in centering titles and captions. Assume that you want to center the title "Table 7-1. Print Control Statements." above a table in the text of a report. The text of the table begins in column 1 on the screen--the left margin is released--and the right margin is at column 70. Therefore, you want the title centered about column 35. You would insert the following lines in the document, just above the table:

```
.center 35
Table 7-1.  Print Control Statements.
.as typed
```

To retain the two blanks between the period and the P, use a tilde for each of the blanks. When the document is printed, the title is centered over the table, even though it was typed at the left margin.

The ".as typed" statement cancels the ".center 35" statement after one line is centered--otherwise lines in the table itself would be centered too. The text of the title could be typed anywhere on the line, not necessarily at the left margin, and it would still appear centered on the printed page.

If a line of text is too long to be centered, it is centered as much as possible without losing characters. For example if you attempt to center a line containing 40 characters about column 10, an impossible task, the line will be printed starting in column 1.

The Center Statement can be used to center each of the lines in a column or passage of text. Note the effect of the Center Statement upon the following short poem. (The top version is the appearance of the text on the screen; the bottom version is the printed output.)

```
.center 30
Big fleas have little fleas
Upon their backs to bite 'em
And little fleas have littler fleas
And so ad infinitum
.as typed
```

Big fleas have little fleas
Upon their backs to bite 'em
And little fleas have littler fleas
And so ad infinitum

(From Big Fleas Have Little Fleas, by Robert Hegner, Dover, 1968)

Screen Print Activity

The Activity Menu contains a new activity called "S. Screen Print". This activity allows you to preview the way a document will appear when printed. In effect, the document is printed on the screen rather than on the paper in the printer. This activity allows you to see the effects of the print control statments in a document without having to wait for a printout. The page breaks caused by automatic pagination and the Eject and Widow Statements are visible, as well as titles, footers, centered lines, top and bottom margins, and offset. After reviewing the screen printing and taking note of any needed changes, you can edit the document to make the necessary changes in the document. Then when you finally do print the document, it is more likely to be perfect the first time through.

To use the Screen Print Activity:

1) At the Activity Menu, press the S key.

2) WordWizard will respond "Type name: ". Type the name of the document you wish to preview, and press RETURN.

3) WordWizard will ask "Skip how many pages?" as if you were going to print on the printer. Press ESCAPE to begin printing at the beginning of the document, or type a number of pages to skip, followed by RETURN.

4) WordWizard will now show "RETURN to print dot commands, ESCAPE to omit: ". Press the RETURN or ESCAPE key accordingly--the question has the same meaning as if you were printing on the printer. (The request "RETURN for sheet, ESCAPE for continuous forms " is not given.)

5) After you respond to the second request, the first screenful of text in the document will be displayed. The bottom line on the screen will give the line count of the last line of text on the screen, relative to the top of the printed page. The first line of text that would be printed on paper is numbered 1, and each line that would be printed is also numbered, even if it is blank. The line count starts over again at 1 after a page break.

6) If you press the down arrow key once, one more line will be printed on the screen. Pressing the down arrow together with the REPEAT key will cause printing to continue until either of the two keys is released. Pressing the CLEAR key will cause another screenful to be printed. Pressing the right or left arrow key alone or with the REPEAT key scrolls the display left or right, one column at a time. The divisions between pages are shown as bands on the screen, two lines high. Press the ESCAPE key at any time during the Screen Print Activity to return to the Activity Menu. After the last line of the document is printed, the down arrow key has no effect, and you can press the CLEAR key or the ESCAPE key to return to the Activity Menu.

Note the following features of the display:

The first line displayed, and the line just under the band that represents page divisions, corresponds to the first line to be printed on the page. It does not necessarily correspond to the top edge of the paper. For example, if you intend to position the top edge of the paper six lines above the the print head before you begin printing, then you should imagine a six line top margin as existing above the bottom of the band, and six fewer lines in the bottom margin above the top of the band.

The last line above the band corresponds to the last line defined by the Define Form Length Statement. If the default value of form length, 66, is in effect (for use with 11 inch paper at six lines per inch) then the last line above the band is line 66.

If there is an Under Statement in the document, a special symbol will appear at the end of the following line. That symbol will not be printed--its purpose is to indicate to you that that line and the following line will be overprinted when the document is printed on paper. The underline sequence increases the line count by only one.

The effects of all other print control statements are visible on the screen, just as they would be on paper, with the exception of the Vertical Spacing and Pitch statements, whose effects cannot be represented on the screen.

Type-ahead Feature

You no longer need to pause when you hear a click from the Helios while typing or editing--you can keep typing if you don't go too fast. When the Helios is through with its activity, the characters you typed will appear in a group on the screen, or the edit commands you gave will be performed. If you are a fast typist, you may need to slow down somewhat during the Helios activity to avoid possible loss of characters.

To scroll more than one line at a time, use the MODE key with the up or down arrow. The scrolling can be stopped with any key. If you used the up and down arrows with the REPEAT key to scroll through a document, several scroll commands would be accumulated during a disk access and then the screen would scroll many lines after the disk access.

Archiving Documents for PTDOS

After you give the name of a document during the Archive activity, a new question appears:

"WordWizard(RETURN) or PTDOS(ESCAPE) format?"

If you respond by pressing the RETURN key, the document will be archived normally for further use in WordWizard. If you respond with the ESCAPE key, all the "inverted L's" (01H) will be removed from the document as it is archived. This feature allows you to create and edit text or programs destined for use by other programs such as assemblers and compilers that cannot process the inverted L character. If you do not intend to use documents with programs other than WordWizard, you need never press ESCAPE in answer to this question.

Programmer's note: This feature allows you to manipulate text created in WordWizard with BASIC, FORTRAN, FOCAL, and assembly language programs, or even to create programs themselves with the WordWizard editor.

Menu Changes

When you give the BOOT command at the starting prompt, the System Functions Menu will appear rather than the Activity Menu. Thus you can use disk duplicating functions or select a printer before going on to the Activity Menu. This change also makes the transition between the Activity Menu and the System Functions Menu much faster. When the manual instructs you to type BOOT RETURN to get the Activity Menu on the screen, you will need to perform the additional step of selecting function "1. Enter the Word Processing Function" from the System Functions Menu.

New Printer Drivers

The Select Printer Menu contains two new choices:

5. DECwriter.
6. CDC Model 9317 Matrix Printer.

"Custom Printer" is now the seventh selection. These new selections are provided for the convenience of the owners of these printers, manufactured by Digital Equipment Corporation (DECwriter) or Control Data Corporation (CDC Model 9317 Matrix Printer.)

With the new release of WordWizard, if you try to print using a SolPrinter that is not plugged in, you will see the message "Printer not ready!" Press RETURN, and the program will display the message "Machine malfunction..." To correct the error, first make sure you have selected a printer that is part of your system. Then, if the selected printer is simply not plugged in, you should plug it in, press the UPPERCASE and REPEAT keys at the same time, and start the program again. If the printer is not connected at all - that is, if it is not plugged into the Sol and Helios - you should call a service person to install it for you.

Other printers may respond differently if they are not connected when you try to use them. For example, the DECwriter and CDC matrix printer will not display any message. If you try to print and the printer does not respond at all, check to see whether it is plugged in and connected to your system.

New version of WIZ - THE ELECTRIC PENCIL SHARPENER

WIZ, an executable program on the WordWizard System Disk and an assembly language source file, WIZ:A, on the WordWizard Document Disk, is described in WordWizard Update 731075. Improved versions, WIZ 1.1 (mod 0) and WIZ:A 1.1 (mod 0) replace the version described there.

In giving the WIZ command, you need not type ":D" after the line length argument--a decimal number is assumed. Just give the command:

```
WIZ filename/L,nn     RETURN
```

The WIZ program has been entirely rewritten so that it handles certain special cases better than the previous version.

Technical Notes

The following information is of interest only to programmers and service people. Most WordWizard users need not read on.

Scrolling has been improved so that the screen fills downwards with newly typed lines before scrolling begins.

In previous releases of WordWizard, there was a bug that caused the occasional appearance of an extra "cursor" on the screen. This bug,

which could also cause other problems, has been corrected for the new release.

When the previous releases of WordWizard were used on certain Sols, occasionally a single keystroke would result in two identical characters on the screen and in the document. The keyboard input routine has been changed to eliminate this problem.

If the Stop Printing Activity was used while printing to the SolPrinter 3, frequently a line of text was left in a buffer in the printer. This leftover line would be printed the next time printing was initiated. The Sol3 device driver has been changed to flush the buffer when printing stops.

A document file structure requirement has been removed. The previous releases of WordWizard have required that lines of text in the document have either a space character (20H) or an "inverted L" character (01H) preceding each carriage return (0DH). This requirement presented no problem unless text created by another program such as WIZ 1.0 was edited and that text did not have the required spaces. Then, if the JUMP TO END command were given, the last character on each line would appear doubled in the 64th column and would be jumbled into the document if the document were edited. (WIZ 1.1 (mod 0) does create text with the required spaces that is compatible with all releases of WordWizard to date.)

Processor Technology

Processor Technology Corporation

7100 Johnson Industrial Drive
Pleasanton, CA 94566

(415) 829-2600
Cable Address: PROCTEC

Software #1 Update 731070

The procedure described below corrects an error in the way that Software #1, Release 1.0 (MOD 0), accepts input from a driver other than the Sol keyboard. It is recommended that all users of the program make this "patch"; otherwise, an attempt to use an input device other than the Sol keyboard will have undesirable consequences. (Also, the documentation for any future patches will assume that this patch has been made.) In the examples the symbol <cr> is used to denote a carriage return, and the symbol > is the SOLOS/CUTER prompt.

The patch is made by reading the program into memory and making alterations there, then saving the corrected program on a cassette tape other than the production cassette. The commands used are those described in the SOLOS/CUTER User's Manual. This notice is intended to enable the user to make the necessary changes in SOFT1, and to have a minimal understanding of the processes involved, without having to refer to that manual. If you are not very familiar with the operation of cassette recorders, you may want to consult the appendix entitled "Using Cassettes" in your Software #1 Resident Assembler User's Manual.

1) Set up a cassette recorder for reading, and load SOFT1 with the SOLOS/CUTER GET command:

>GET SOFT1<cr>

2) When the prompt (>) returns, type

>EN 39<cr>

to indicate that the next number entered (see step 3) should replace the present contents of memory location 39H. If step 3 involved a list of numbers, those numbers would be placed in consecutive memory locations beginning at 39H.

3) When a colon (:) appears on the screen, type

47/<cr>

to enter the number 47H at location 39H.

4) When the prompt (>) returns, type

>EN FAF<cr>

to indicate that the next number entered should replace the present contents of memory location FAFH. Subsequent entries (i.e., the 51 in step 5) will occupy consecutive locations following FAFH.

5) When a colon (:) appears on the screen, type:

F9 51/<cr>

Notice that the list of entries must again be terminated by a slash. (In step 3 there was only one entry in the list, but the slash was still required.) When the prompt (>) returns, the patch has been completed.

6) Set up a cassette recorder for writing. Insert and position a cassette tape other than the production cassette. Be sure to leave at least half a minute at the beginning of the tape unrecorded.

7) After hitting the MODE SELECT key to return to SOLOS/CUTER command mode, type:

>SET XE=0<cr>

to set the starting address that will be written in the tape file header.

8) Type

>SAVE SOFT1 0 FB1<cr>

or

>SAVE SOFT1/2 0 FB1<cr>

(The first version assumes that the file is being written to a cassette in unit 1; the second assumes that the file is being written to a cassette in unit 2.) When the file has been recorded completely, the prompt (>) will be displayed again.

9) Before copying the patched version of SOFT1 to the other side of your cassette tape (NOT the production cassette), load the program FROM THE TAPE THAT YOU HAVE JUST WRITTEN to make sure that it works. If the input pseudo-port setting that you use when you execute SOFT1 is the Sol keyboard and you have access to another input device, try using the CUST command to return to SOLOS/CUTER, the SET command to change the input pseudo-port, and the EXEC command to restart SOFT1. (Use the example on top of page 2-8 in the Software #1 Resident Assembler User's Manual, but type "SET I= " instead of "SET O= ", and use the port designation that matches your device.) If you do not have another input device, just try several of the commands to see that nothing is amiss.

If the program seems to be working properly, use the CUST command to return to SOLOS/CUTER. If not, load SOFT1 again (as in step 1) and repeat steps 2 through 8, writing the new patched version over the old patched version of the program and testing your results.

10) To copy the patched version of SOFT1 to the other side of the tape on which you have recorded it, load the program again (as in step 1) from the ORIGINAL side of that cassette, and repeat steps 6 through 8 above, making sure you write the file to the REVERSE side of the cassette.

11) Note on your duplicate cassette that you have updated your copy of the Software #1 program from MOD 0 to MOD 0/1. When you execute the program, it will continue to identify itself as MOD 0; it is therefore very important that you make a record of the correct MOD number, so that you will know whether any further update notices apply to you.

Remember that Processor Technology software is copyrighted. Your duplicate is for your use for maintenance and backup purposes only. No copies should be given away or sold.

ALS 8

Processor Technology

Change Notice #1B

In order to receive the full benefit of the built-in features of ALS-8 (i.e., dynamic control of video display speed, simulator, etc.) we suggest the following procedures for loading and executing the ALS-8 program development system. These procedures assume that a VDM-1 video display module and the CUTER operating system are in use or that the computer involved is a Sol Terminal Computer with a SOLOS personality module.

a) Load ALS-8 from cassette using the GET command (not XEQ).

b) From the SOLOS/CUTER command mode, type EX EØ24, carriage return (CR). This causes the ALS-8 to write a pair of standard I/O drivers to memory page DØØØ where they can be altered to suit the I/O configuration of the system involved.

c) In a Sol, reset the computer by pressing the UPPER CASE and REPEAT keys simultaneously. In other computers, first stop the computer from running, then start up CUTER again, at location CØØØ. The video display will respond with a prompt.

d) Enter the data shown below. The underline indicates characters which are responses from SOLOS/CUTER:

ENTRIES FOR CUTER	ENTRIES FOR SOLOS
>EN DØ9F (CR)	>EN DØ9F (CR)
:Ø3 (CR)	:FC (CR)
:DØA6: 2F E6 Ø1 C9 (CR)	:DØA5: FA 2F E6 Ø1 C9 (CR)
:DØDØ: 77 FE (CR)	:DØDØ: 77 FE (CR)
:DØ96: 77 FE / (CR)	:DØ96: 77 FE / (CR)

SOLOS/CUTER should again respond with a prompt: > . The above entries modify the standard input and output drivers, which are written for a serial console interface, to work with a parallel console interface, as for a keyboard.

e) Type EX EØ6Ø to transfer control to the ALS-8. DO NOT try to start the ALS-8 at EØ24 as this will reinitialize the drivers, overwriting the entries just made.

f) Type Control-Z (CR), (i.e., while depressing the CTRL key type Z). This initializes the ALS-8 video display driver, clearing parameters.

g) Next type Control-S. The ALS-8 program will respond: SPEED?

h) Type one key from 1 to 9 to set the rate at which characters are added to the video display. A carriage return is not required for this entry. Normally type 1 for the fastest speed.

i) One other key, Control-A, is used by the video driver to toggle the cursor on and off. If the cursor is not on, typing Control-A will turn it on when the next character is typed. Typing Control-A again will turn the cursor off after the next character is typed.

PROCESSOR TECHNOLOGY CORPORATION 6200 HOLLIS STREET EMERYVILLE CA 94608 (415) 652-8080
CABLE ADDRESS "PROCTEC"

Processor Technology

Processor Technology
Corporation

7100 Johnson Industrial Drive
Pleasanton CA 94566

(415) 829-2600
Cable Address PROCTEC

ALS8 MANUAL

Change Notice #2

The cassette version ALS8 you have received with this manual contains a Revision B ALS8 program. Several new features have been added which are not described in the ALS8 User's Manual. Attached to this notice are several pages containing a new version of Appendix C of the manual, describing the new features, and integrating parts of the previous revision of Appendix C which still apply. To point you in the right direction if you or others try to refer to the old pages, you may wish to write on them, "See Change Notice #2".

APPENDIX C ALS8 REVISION B - SOLOS/CUTER INTERFACE

The ALS8 program is distributed on various media, including CUTS format cassette tape. This cassette consists of one file which loads into memory beginning at location DE00 (Hex), and ending at FFFF, the top of memory. Revision B and later cassette versions includes routines to link ALS8 with SOLOS or CUTER monitor programs, and new input and output driver routines. In the following text, SOLOS will be used to refer to both SOLOS and CUTER.

1.0 GETTING ALS8 RUNNING

To load and execute ALS8 under SOLOS or CUTER:

1. Be certain that 12K of RAM exists from D000 to FFFF.

2. If CUTER is used, make sure to load it at an address which does not conflict with ALS8.

3. Set SOLOS/CUTER pseudo-ports, if their default values for keyboard input and video output are not wanted.

4. Using the normal cassette recorder procedures, and the command to SOLOS or CUTER "XEQ ALS8," load ALS8.

5. ALS8 will automatically begin execution of the initialization routines at DE00, and then display a sign-on message followed by the ALS8 prompt, "READY."

As an alternate procedure, ALS8 may be loaded into memory with the command "GET ALS8," and execution may be begun with the command "EX DE00."

2.0 INITIALIZATION ROUTINES

The initialization routines perform a checksum on the ALS8 code which was just read into memory. If the tape has been damaged, a bad memory location is encountered, or other hardware problems have caused incorrect code to appear in memory, the message "## CHECKSUM ERROR ##" will appear. The error may not be serious, but it is best to try reloading the tape, correcting harware problems, if present.

Complete initialization at DE00 performs the following actions:

1. A link is created between ALS8 and SOLOS.

2. The symbol table address will be set to D700 (STAB).

3. A pointer to a return instruction will be set for the CONTROL-U custom command used in the editor. If no custom command address has been set and CONTROL-U is typed within the editor, a return will be made to the editor.

4. The FORM (formatted output) switch will be set off.

5. The ESET address (end of usable file space) will be set to the

6. If the output device is a VDM-1 or the Sol video display, the screen will be cleared, the cursor turned on, and the display speed set to 1 (fastest display).

7. Current SOLOS pseudo-ports will be used for all ALS8 input and output.

CAUTION

The initialization routine at DE00 depends upon the address of SOLOS or CUTER being passed in the 8080 H and L registers, to establish necessary SOLOS/CUTER routine addresses used by ALS8. Once the ALS8 has been initialized, the startup routine and address at DE00 is overwritten. DO NOT ATTEMPT TO RE-ENTER THE ALS8 AT ADDRESS DE00. Use one of the routines listed below.

ADDRESS	ACTION
E024	Reinitialization of all global RAM and tables
E000	Reset standard SYSIO drivers
E060	No change, except final I/O as set in SOLOS

3.0 VIDEO DISPLAY CONTROLS

SOLOS pseudo-ports may be changed at any time by returning to SOLOS with the command EX C000, and using the SOLOS SET commands described in the SOLOS/CUTER manual. ALS8 may then be reentered using one of the addresses given above. If the SOLOS default output pseudo-port 0 is in effect, ALS8 will use its own VDM driver, allowing the following controls over the video display:

KEYS TYPED	ACTION
CONTROL-A	Turns cursor on or off--toggle action
CONTROL-Z	Clears screen
CONTROL-S then 1-9	Sets display speed. 1 key gives fastest display, 9 slowest.
1-9	Sets display speed dynamically, during output to screen.
SPACE BAR	Stops output. Pressed a second and subsequent times, gives one line of output.
ALPHABETIC KEY	Resumes continuous output after a SPACE BAR stop.
1-9	Resumes output as above, at new speed
ESCAPE	Stops output, returns to ALS8 with READY prompt

4.0 COMPATIBILITY WITH OTHER VERSIONS OF ALS8

Other versions of ALS8 exist on PROM and ROM. These versions do not include the new Revision B input and output routines. Programs written for these other versions may still be run using Revision B ALS8, since the new routines are accessed with jumps from the previous I/O addresses as follows:

ROUTINE	OLD ADDRESS -->	NEW ADDRESS
STATUS	D0A4	DFC5
INPUT	DO98	DFD2
OUTPUT	DOA9	DFE1

5.0 NEW ARITHMETIC ROUTINES

Revision B ALS8 includes code which allows access to all the arithmetic routines of the assembler through a custom command. To use this feature, create a custom command as follows:

```
CUSTE /MATH/DFAA <CR>
```

Use the new command by typing its name:

```
MATH <CR>
```

ALS8 will respond with a colon (:). Next, type an expression to be evaluated. The expression can consist of numbers in decimal (default), hex, or octal formats, arithmetic operators (+ - * /), symbols, or letters representing registers. The result will be given in the current number base set with the ALS8 MODE command. Here are some examples of interchanges using the MATH command:

1)	MATH: 1024 <CR> 400	interpreted as a decimal number since no number base suffix was given Since the answer is 1024 Dec given as hex number, the MODE must be hex.
2)	MATH:EORMS E060	This symbol's value is given from the system symbol table.
3)	MATH:-1 FFFF	The address before 0 is considered to be the last address in memory.
4)	MATH:303Q C3	The octal code for JMP is converted to the hex code.
5)	MATH:0FFFFH+1 0000	The value returned is given in 16 bit precision.
6)	MATH:1024*4/2+A 807	The letter A represents the mnemonic for the Accumulator. Assembled to object code, its value is 7.
7)	MATH:LAST-FIRST 304	This subtraction detemines the byte count between two labels

The symbol value returned will be its value from the last assembly (example 7), or from the system symbol table (example 2). If the process of assembly created a symbol table or a cross-reference symbol table, the value returned by MATH will be 0000, since symbols are deleted as they added to a table.

When the arithmetic operators are used, the order of evaluation is the order given. Division produces only an integer result.

Numbers entered for evaluation must be given in the same format that the assembler expects. The following rules apply:

1. Hex numbers starting with A-F must have a preceding 0 added (0CC00H).

2. Hex numbers must have an H suffix (400H).

3. Octal numbers must have a Q suffix (377Q).

4. Decimal numbers need no suffix.

5. The following reserved letters may not be used for symbols since they represent registers. The value returned for each letter is given beneath it:

A	B	C	D	E	H	L	M
7	0	1	2	3	4	5	6

Remember that any time the ALS8 RAM area is reset by executing E024, all system symbols and custom command names are deleted, including the MATH custom command name. It may be recreated using the procedure given above.

The code for the ALS8 SOLOS/CUTER I/O routines and the MATH routine is shown below as it exists within Revision B ALS8.

```
DFA9 00                         DB      0
                        *
     DFAA               LAST1   EQU     $
                        *
                        ************************************************
                        *
                        *
                        *   <  MATH - CUSTOM COMMAND CODE  >
                        *
                        *
DFAA                            ORG     0DFAAH      custom command address
                        *
DFAA CD 80 FF           MATH    CALL    0FF80H      pick up args from IBUF
DFAD 21 34 D2                   LXI     H,0D234H    insert a space to trick ALS8
DFB0 36 20                      MVI     M,' '
DFB2 22 01 D1                   SHLD    0D101H      tell ALS8 where code is
DFB5 23                         INX     H           point to code+1
DFB6 CD 5D F1                   CALL    0F15DH      do the math
DFB9 EB                         XCHG
DFBA CD 5C E5                   CALL    0E55CH      print answer
DFBD CD 16 E2                   CALL    0E216H      print carriage return/line feed
DFC0 C3 60 E0                   JMP     EORMS       back to ALS8 - 'READY'
                        *
DFC3 00 00                      DB      0,0
                        *
     DFC5               LAST2   EQU     $
```

```
                        ;*****************************************************
                        ;*
                        ;*
                        ;*    <  ALS8 / SOLOS / CUTER  ROUTINES  >
                        ;*
                        ;*
                        ;*
                        ;* ALS8/SOLOS SYMBOL/ROUTINE ADDRESS & EQUATES
                        ;*
E060                    EORMS   EQU     0E060H      ALS8 'READY'
001B                    ESC     EQU     1BH         ESCAPE key code
C01F                    SINP    EQU     0C01FH      SOLOS/CUTER input routine
C019                    SOUT    EQU     0C019H      SOLOS/CUTER output routine
C807                    OPORT   EQU     0C807H      SOLOS/CUTER pseudo port value
FE77                    VDM     EQU     0FE77H      ALS8-VDM output routine
                        ;
D190                    CHAR    EQU     0D190H      character recieved hold address
                        ;
                        ;
                        ;******************
                        ;  ALS8 -  SOLOS/CUTER STATUS ROUTINE
                        ;
                        ;
DFC5                            ORG     0DFC5H      ALS8 status routine entry point
                        ;
                        ;
DFC5 3A 90 D1           STAT    LDA     CHAR        check if character waiting
DFC8 B7                         ORA     A           zero = no
DFC9 C0                         RNZ     .           yes - return zero flag set (NZ)
DFCA CD 1F C0           XXINP   CALL    SINP        call SOLOS/CUTER input routine
DFCD C8                         RZ      .           no, return with zero reset (Z).
DFCE 32 90 D1                   STA     CHAR        got input, save character,
DFD1 C9                         RET     .           and return zero flag set. (NZ)
                        ;
                        ;
                        ;******************
                        ;
                        ;  ALS8 - SOLOS/CUTER INPUT ROUTINE
                        ;
                        ;
DFD2 CD C5 DF           INP8    CALL    STAT        get status
DFD5 CA D2 DF                   JZ      INP8        no input, wait.
DFD8 E6 7F                      ANI     7FH         strip parity bit
DFDA 47                         MOV     B,A         save character
DFDB AF                         XRA     A           zero accumulator
DFDC 32 90 D1                   STA     CHAR        clear character waiting
DFDF 78                         MOV     A,B         get character back to reg A
DFE0 C9                         RET     .           return with character in reg A.
                        ;
                        ;
```

```
                        ********************
                        *
                        *   ALS8 - SOLOS/CUTER OUTPUT ROUTINE
                        *
                        *
DFE1 3A 07 C8           OUTP8   LDA     OPORT   check if VDM is the output device
DFE4 B7                         ORA     A       pseudo port address zero = yes
DFE5 CA 77 FE                   JZ      VDM     yes - go to ALS8 VDM routine
DFE8 CD C5 DF                   CALL    STAT    check if input first
DFEB CA F8 DF                   JZ      XXOUT   no input
DFEE C5                         PUSH    B       save character to be output
DFEF CD D2 DF                   CALL    INP8    get input character
DFF2 C1                         POP     B       restore character to be output
DFF3 FE 1B                      CPI     ESC     check if input was 'ESCAPE' key
DFF5 CA 60 E0                   JZ      EORMS   if so, stop output & return to ALS8.
DFF8 CD 19 C0           XXOUT   CALL    SOUT    go to SOLOS/CUTER output routine
DFFB 78                         MOV     A,B     return with output character,
DFFC C9                         RET     .       in reg A and B.
                        *
                        *
DFFD 00 00 00                   DB      0,0,0
                        *
                        *  The three zeros above leave address
                        *  DFFD, DFFE, & DFFF clear for user flags.
                        *
     E000               LAST3   EQU     $
                        *
                        *
                        ************************************************
```

ProcessorTechnology

Processor Technology
Corporation

7100 Johnson Industrial Drive
Pleasanton, CA 94566

(415) 829-2600
Cable Address - PROCTEC

Extended Disk BASIC Update 731Ø44

The PTDOS System Disk that this update accompanies contains Extended Disk BASIC. The Extended Disk BASIC User's Manual (first or second edition) assumes that BASIC is supplied on a separate diskette. In section 2.1 of the manual you are instructed to make a working version of BASIC using the file MAKBASIC. Since MAKBASIC will be on unit Ø instead of unit 1 when you use it, the command you should use in step 1 should be DO MAKBASIC <CR> (without the "/1"). Cross out the "/1" in your manual and you will perform the initialization correctly when you come to it.

Processor Technology

Processor Technology Corporation

7100 Johnson Industrial Drive
Pleasanton, CA 94566

(415) 829-2600
Cable Address: PROCTEC

Extended Cassette BASIC Update 731064

Subject: Errata and Addenda to User's Manual, First Printing
Fixing a bug in FOR/NEXT loop operation

This update contains a series of items of new or corrected text. Each item begins with the page number where the new text goes, and contains some surrounding text to help in locating its position. To have access to this new information when you need it, either mark the corrrections on the text pages where they apply, or make notes like "See Update 731064".

---

1. BOTTOM OF PAGE 2-2, TOP OF 2-3

Again type Y or N to remove or not remove an additional part of BASIC which performs trigonometric functions and certain other extended functions. The following functions cannot be used if Y is typed: SIN, COS, TAN, EXP, SQR, ATN, LOG, LOG10. After Y or N is typed, the READY message will appear.

As long as BASIC is in memory, the command

EX{ECUTE} 0 <CR>

will re-enter it.

After BASIC displays the READY message, you can enter programs and issue commands.

To leave BASIC and return to the SOLOS or CUTER monitor program, simply type BYE <CR>.

BASIC and its current program, if any, are not lost and you can reenter by typing the EX{ECUTE} 0 command.

When BASIC is executed for the first time, the BASIC code which was just loaded into memory is tested using checksums. If the code is not correct, the message CHECKSUM FAILED is displayed followed by two hexadecimal numbers: the correct checksum and the incorrect checksum. If you type a carriage return, you will enter BASIC, and BASIC may appear to operate properly. It is best, however, to try reading the tape again, after returning to SOLOS/CUTER by typing the UPPER CASE and REPEAT keys together. BASIC is recorded twice in succession on the cassette. If you get the same checksum error message after trying to read the first recording of BASIC, try reading the second recording. Repeated checksum errors can be caused by wrong settings of cassette recorder controls, dirty tape heads, poor adjustment of the cassette interface, bad memory locations, or other hardware problems.

2. MIDDLE OF PAGE 3-2

MODE SELECT	Aborts a running program, infinite loop, listing, and getting or saving operations. Deletes a line being typed. If used to stop a running program, all open files will be closed.

---

3. TOP OF PAGE 3-7

```
BASIC:  10 PRINT "ENTER Q, Y, AND Z"
User:   (Positions the cursor over Q and type X <CR>).
```

---

4. BOTTOM OF PAGE 3-8

If you edit any part of a program after interrupting execution, all variable definitions are lost. Thus you cannot stop a program's execution, change a statement in that program, and then CONTinue execution or print variable names. When a program run is terminated for any reason, all open files are closed, which also could interfere with subsequent CONTinuation.

---

5. TOP OF PAGE 4-5

TAB(exp)	Causes the cursor to move horizontally to the character position given by the value of exp (any numerical expression.) This function may only be used in a PRINT statement.
"&c"	Prints the control character c. Printing some control characters performs a function on the terminal. For example: Control M - Carriage return Control J - Line feed Control K - Home cursor and clear screen Control N - Home cursor Section VII of the SOL notebook has a complete list of control characters and the special symbols or control functions generated by printing control characters.
"&&"	Print a single ampersand (&).

---

6. TOP OF PAGE 5-8

In the above example, the variable names listed in parentheses after FNL in line 100 are called formal parameters. In user-defined functions, all formal parameters are locally defined within the function; if any statement in the function modifies the value of a variable which is also a formal parameter, the value of that variable outside the function will NOT be changed. This is true for numerical variables only, not strings, arrays, or matrices. For example:

```
 1 Q = 40                    | 30 RETURN Z
10 DEF FNA1(X,Y,Z)           | 40 FNEND
20 X=X+1,Q=X+Y,Z=Q/3         | 50 X=1, Y=2, Z=3
25 S = 4                     | 60 PRINT FNA1(X,Y,Z), X, Y, Z, Q, S
                             | RUN
                             |      1     1     2     3     3     4
                             | READY
```

Note that the values of X, Y, and Z, outside the function were not changed by line 20 which is inside the function. Note also that Q, which was not a formal parameter, WAS changed by line 20. Variable S, introduced within the function, retains its value outside the function.

---

7. BOTTOM OF PAGE 5-19

The filename ("name") used in this command must be the same as the name used when the file was created. The FILE statement may be used to create a file for subsequent PRINT statements, in which case the file name is assigned by the FILE statement and written on the file when the first PRINT statement is executed.

The file name consists of 1 to 5 characters and an optional unit number. The form is: name/unit, where unit is 1 or 2.
For example:

```
            DATA1/2   refers to a file named DATA1 in unit 2,
            STUFF     refers to a file named STUFF on unit 1,
```

the default unit.

---

8. PAGES 5-29 AND 5-30

Controlled INPUT Statement

General forms:

```
  INPUT, (#chars,t) var1, var2, ...     Enters values from the
            |   |       |     |         terminal and assigns
 numerical__|___|       |     |         them to var1, var2,
 expression            |_____|         etc.; however, only
                    variables          #chars characters can
                                       typed by the user and
                                       the user has t tenths
                                       of a second to
                                       respond.

  INPUT (#chars,t) " message", var1, var2, ...
                       |                 Same as above, but a
                       |__string         message is printed as
                      constant           a prompt before values
                      including          are accepted from the
                      its quotes         terminal, and before
                                         timing begins.
```

Examples:

```
  10 INPUT (3,10) X
 100 INPUT (20, 0) N$, A$
 200 INPUT (0 ,100) A, B, C
 300 INPUT (10,300) "WHAT IS THE DATE?" ,D$
```

The controlled INPUT statement lets you specify how many characters can be entered and how much time is allowed to respond. As soon as #chars characters have been typed, BASIC generates a carriage return and accepts no more characters. If the user

takes more than t tenths of a second to respond, BASIC assumes a carriage return was typed.

If the value of #chars is 0, as many as 131 characters can be entered. If the value of t is 0, the user has an infinite amount of time to respond. For example:

```
 5 DIM A$(3)
10 FOR X = 1 TO 9
20 FOR Y = 1 TO 9
30 PRINT X;" * ";Y;" = "
40 INPUT (3, 100) A$
42 IF A$ = "" THEN PRINT "YOU ARE SURE SLOW!": GO TO 30
45 A = VAL(A$)
50 IF A <> X*Y THEN PRINT "TRY AGAIN" : GO TO 30
60 NEXT Y
70 NEXT X
```

When executed, this program accepts a three-character answer from the user and waits 10 seconds for a response. If the user does not respond within 10 seconds the message YOU ARE SURE SLOW is printed. If the user types the wrong response, the message TRY AGAIN is printed.

5.8. ERROR CONTROL

BASIC detects many kinds of errors. Normally, if an error occurs, BASIC will print one of the error messages listed in Appendix 3. However, using the error-control statements described below, you can tell BASIC to execute another statement in the program instead. The ERR(0) function gives a string containing the last error message, which can be used in error control programming.

ERRSET and ERRCLR Statements

General forms:

ERRSET n (n = statement number)	Determines that statement n will be executed if any error is detected by BASIC.
ERRCLR	Cancels the last ERRSET statement.

Examples:

```
 10 ERRSET 75
100 ERRCLR
```

The ERRSET n statement lets you determine that statement n will be executed when any error occurs. If an error does occur and the ERRSET n statement does cause a transfer to statement n, before statement n is executed the ERRSET statement itself is

cancelled (as if an ERRCLR statement were executed.) Also, the transfer to statement n clears all current FOR/NEXT loops, GOSUBs, and user-defined function calls (as if a CLEAR statement were executed.)

The ERRCLR statement cancels the most recent ERRSET statement. If a statement executed after an ERRCLR statement produces an error, BASIC will print a standard error message (See Appendix 3,) rather than going to statement n. However, if the ERRSET statement is executed again, it will again set the error trap statement n, as if the ERRSET were encountered for the first time.

---

9. MIDDLE OF PAGE 7-4

3. The second dimension (columns) of mvar3 must be the same as the second dimension of mvar2.

4. The second dimension (columns) of mvar1 must equal the first dimension (rows) of mvar2.

7.5. MATRIX FUNCTIONS

Two matrix functions may be used to place the inverse or transpose of a matrix into another matrix.

Inverse and Transpose Functions

General Forms:	
MAT mvar1 = TRN (mvar2)	Places the transpose of mvar2 into mvar1. mvar1 and mvar2 must have opposite dimensions.
MAT mvar1 = INV (mvar2)	Places the inverse of mvar2 into mvar1. Both matrices must be square.

Examples:

```
10 MAT A = TRN(B)
20 MAT C = INV(D9)
```

mvar1 and mvar2 must not be the same matrix. No check is made to insure that mvar1 is not the same matrix as mvar2. If they are the same, unpredictable results will occur. As with all functions, the argument must be within parentheses.

---

10. TOP OF PAGE A1-7 ALSO ON SUMMARY CARD

POKE location, value
--
Places the specified value in the specified memory location. C

11. TOP OF PAGE A2-2

INT(n) Truncates n to its integer part.

---

12. MIDDLE OF PAGE A2-2

TAB(n) Moves the cursor or print head horizontally to character position n. Use only in a PRINT statement.

---

13. BOTTOM OF PAGE A2-2

string variable (exp1{,exp2})
Characters exp1 through exp2 of the specified string if exp2 is present. Characters exp1 through the end of the string if exp2 is ommitted.

---

14. BOTTOM OF PAGE A3-1

IN	Input error. The last VAL function attempted to determine the value of a string which did not contain a number.	Provide a string which contains a number. Study the program logic.

---

15. MIDDLE OF PAGE A3-2

NC	Not CONTinuable. The current program, if any, cannot be CONTinued.	Make sure a BASIC program is ready to run. You cannot CONTinue after editing a program, using the CLEAR command, etc.

---

16. MIDDLE OF PAGE A3-3

FP	Floating Point error. C cannot handle numbers greater than 10 to the 126th power, or less than 10 to the -126th power.	No solution.
NI	Not implemented. An attempt was made to use matrix or trig functions which were deleted.	See Section 2.1.

---

17. TOP OF PAGE A3-5

MS	Matrix Singular Error. The operation attempted cannot be performed on a singular matrix.	The operation cannot be performed on the data in the given matrix.
UD	Undimensionned matrix. A variable name was used which was not previously defined in a DIM statement.	DIMension the matrix in an earlier DIM statement.

Fixing a bug in FOR/NEXT loop operation

A bug can occur in FOR/NEXT loops, if a loop is constructed so that it will not allow execution of a nested inner loop. To fix this bug, you can read BASIC into memory, make a simple patch, and re-record the patched version, using this procedure:

1) Place the BASIC cassette in unit 1 and type GET BASIC <CR>.
2) Still in SOLOS/CUTER, type EN B50 <CR>.
3) Type the following, noting the spaces separating entries:

```
:C1 CA 40 0B/ <CR>.
EN 3F81 <CR>
:FE 9D/ <CR>
```

4) To save the patched BASIC now in memory, you can re-record on the same cassette, after taping over the two holes on the back of the cassette to allow re-recording, and recording 15 seconds of empty tape. Still in SOLOS/CUTER, type:

```
SET TYPE 42 <CR>
SET XEQ 0 <CR>
SAVE BASIC 0 3F84 <CR>
SAVE END FFFF 0001 <CR>
```

The first of your two recorded BASICs is now fixed. The following program should now RUN with no CS ERROR:

```
10 FOR I=1 TO 0
20 FOR K=1 TO 0
30 PRINT "THIS NEVER WILL GET PRINTED SINCE A FOR LOOP "
40 PRINT "CANNOT STEP BACKWARDS!"
50 NEXT K
60 NEXT I
```

When you have successfully patched the first recorded version of BASIC, you may wish to also SAVE BASIC a second time, writing over the second unpatched BASIC which follows on the tape. Before using the tape, be sure to remove the tape from the back of the cassette to insure "write protection."

This procedure replaces an incorrect fix published in the Processor Technology ACCESS newsletter, Volume Two, Number One, page six.

compatible with other PTC products. Specifically, if you have an 8080 or Z-80 microcomputer you're using with SOLOS, CUTER, SOL/CUTS cassette interface, or any functionally equivalent operating system, you might like to join. The membership is world-wide.

Currently, SOLUS has several good services going, beginning with their own bi-monthly (approximately!) newsletter to keep members informed on hardware, software, new products, bugs, local chapter meetings, and other items of interest. They maintain a Software Library which collects and distributes public domain programs for a nominal charge. Local chapters hold meetings to exchange software and ideas, and the club headquarters, in convenient proximity to PTC, keeps a close communication link going with us. Finally, qualified SOLUS volunteers are testing products for Sol compatibility and reporting their experiences in the newsletter.

To refresh you on the club's goal and organization: The stated goals are to facilitate communication among SOLUS members, to provide a mechanism of exchange for Sol-compatible software, to give PTC feedback from SOLUS members, and to encourage the development and testing of Sol-compatible hardware and software produced by independent sources. Their relationship with PTC and other manufacturers is co-operative but independent. SOLUS is primarily a personal computer users' group, but special interest groups can be formed within the club on subjects like commercial use, health care, education, scientific applications, etc. The society is supported by dues and volunteer efforts.

To join SOLUS: If you live in the U.S., Canada or Mexico, send $10.00 in check or money order to SOLUS, P.O. Box 23471, San Jose, CA 95153. If you live anywhere else, send $15.00. Dealers and manufacturers of Sol-compatible equipment or software should contact SOLUS at the above address for details on special memberships. □

What To Do With Your BASIC/5 Programs Now That You Have Extended Cassette BASIC

Unfortunately, programs written in Sol BASIC/5 won't run with the new Extended Cassette (formerly called 8K) BASIC interpreter. But you can save yourself several thousand tedious keystrokes by using the following assembly language program to save your BASIC/5 programs in text form on CUTS cassette. Then it will be easy to get and run a text file on the new interpreter whenever you wish.

The program uses the byte access tape routines in the SOLOS/CUTER monitor. CLIST is written to be used as custom output diver in conjunction with pseudo port 3.

When you use the program, load the machine code at the addresses indicated in the assembly listing below. Set the custom output port address to the beginning address of this program. Get and execute BASIC/5. Get the program you want to write to the tape. From BASIC/5, set the output port to 3 and type LIST.

After your program has been saved, this routine will set the current output port back to 0. □

```
                        *
000D                    CR      EQU     0DH     ASCII carriage return
000A                    LF      EQU     0AH     ASCII line feed
                        *
                        *  Solos equates
                        *
C01C                    AOUT    EQU     0C01CH
C007                    FOPEN   EQU     0C007H
C00A                    FCLOS   EQU     0C00AH
C010                    WRBYT   EQU     0C010H
C807                    OPORT   EQU     0C807H
                        *
                        *  The program starts here.
                        *  This is also the entry point.
                        *
                        *
CB00                            ORG     0CB00H
                        *
                        *
CB00                    CLIST   EQU     $
                        *
CB00 C5                         PUSH    B       Save registers
CB01 D5                         PUSH    D
CB02 E5                         PUSH    H
                        *
```

```
CB03 78                         MOV     A,B     get character
                        *
CB04 FE 0D                      CPI     CR      Carriage return?
CB06 C2 26 CB                   JNZ     CL2
                        *
CB09 21 73 CB                   LXI     H,LCHAR
CB0C 7E                         MOV     A,M     Get last character
CB0D FE 0D                      CPI     CR      If not two carriage returns in
CB0F C2 26 CB                   JNZ     CL2     ...in a row then nothing special
CB12 23                         INX     H       Else check if first set
CB13 7E                         MOV     A,M     ...of carriage returns
CB14 B7                         ORA     A       If not then close file and
CB15 C2 4B CB                   JNZ     CLOOP   ...reset program
CB18 3D                         DCR     A       Else say we already
CB19 77                         MOV     M,A     ...had first time through
CB1A 3A 62 CB                   LDA     UNIT    Get unit to write to
CB1D 21 63 CB                   LXI     H,HEADR  Point to tape header
CB20 CD 5E CB                   CALL    OPNOP   Open the file
CB23 C3 47 CB                   JMP     GBACK
CB26 78                 CL2     MOV     A,B
CB27 FE 0B                      CPI     LF+1
CB29 DA 2F CB                   JC      CL3
CB2C 32 73 CB                   STA     LCHAR
                        *
CB2F 3E 00              CL3     MVI     A,0     Reflect character to screen
CB31 CD 1C C0                   CALL    AOUT
                        *
CB34 3A 74 CB                   LDA     FIRST   Check if writing yet
CB37 B7                         ORA     A
CB38 CA 47 CB                   JZ      GBACK   Return if not
                        *
CB3B 78                         MOV     A,B
CB3C FE 0B                      CPI     LF+1
CB3E DA 47 CB                   JC      GBACK
                        *
CB41 3A 62 CB                   LDA     UNIT    Get unit to write to
CB44 CD 10 C0                   CALL    WRBYT   Write the byte
                        *
CB47 E1                 GBACK   POP     H       Restore registers
CB48 D1                         POP     D
CB49 C1                         POP     B
                        *
CB4A C9                         RET     .       ...to Basic/5
                        *
     CB4B               CLOOP   EQU     $
                        *
CB4B 3A 62 CB                   LDA     UNIT
CB4E CD 0A C0                   CALL    FCLOS   Close file
CB51 AF                         XRA     A
CB52 32 73 CB                   STA     LCHAR   Reset program for next
CB55 32 74 CB                   STA     FIRST   ...time through
CB58 32 07 C8                   STA     OPORT   Change output port to 0
CB5B C3 47 CB                   JMP     GBACK
                        *
                        *
     CB5E               OPNOP   EQU     $
                        *
CB5E CD 07 C0                   CALL    FOPEN
CB61 C9                         RET
                        *
CB62 01                 UNIT    DB      1       Change to 2 to use tape unit 2
                        *
CB63 43 4C 49 53        HEADR   ASC     'CLIST'
```

```
      54
CB68  00                        DB      0
CB69  54                        DB      'T'     'T' for TEXT
CB6A  00 01                     DW      256     Length of block
CB6C  00 00                     DW      0
CB6E  00 00                     DW      0
CB70  00 00                     DW      0       Three spares
CB72  00                        DB      0
                        *
CB73  00                LCHAR   DB      0
CB74  00                FIRST   DB      0
CB75  00                INFLG   DB      0
                        *
```

Interfacing To The Sol's Parallel Port

by Lee Felsenstein

The parallel input/output port on the Sol, J2, is intended to allow highspeed transfer of data between the CPU and outside devices. The circuitry was designed to be symmetrical, so that two devices having identical interfaces can communicate with each other. In contrast, the 3P+S circuit puts out a positive-going data strobe pulse but requires either a negative-going data strobe pulse or a steady level in the return direction. As a result, two 3P+S boards cannot communicate without additional circuitry, while two Sols can communicate through their parallel ports with nothing but wires between them.

Data is transferred on eight-bit-wide paths into and out of the Sol, through the pins of J2 labelled PID0-7 (data into the CPU) and POD0-7 (data out of the processor). Polarity of the data on these pins is positive—that is, a "high" logical level is equivalent to a binary "1." Data is latched at the output side of the interface and is not latched at the input side. This means that after an output instruction to the parallel port (port FD) is issued, the new data remains present at the output pins of the connector indefinitely, until the next output instruction is issued. Similarly, the external device which provides data to the Sol through J2 must keep its data present on the PID pins long enough for it to be "picked up" by the Sol CPU.

FIG 1.
HANDSHAKING SEQUENCE

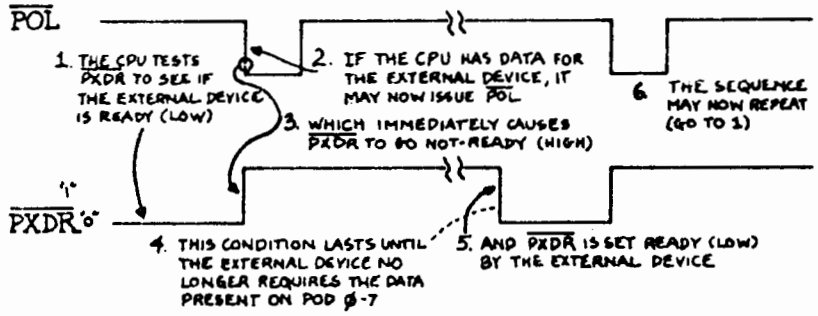
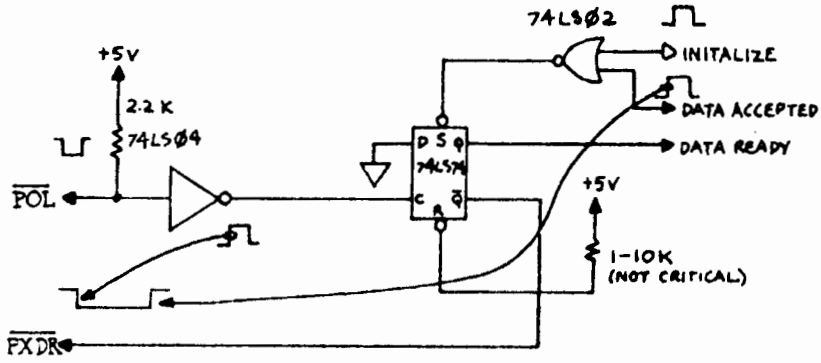


FIG.2.
CIRCUITRY FOR HANDSHAKING



Supervisory signals, known as "handshaking" signals, are provided in both directions to allow the devices on each side of the interface to know when data has been accepted and when each may present the next eight bits of data. Each side issues a negative-going "strobe pulse" when its data is ready and latched. In the direction going out from the Sol, this signal is called POL, for Parallel Output Load. The line over the signal name signifies that the signal is "active low." All of

Processor Technology Corporation
7100 Johnson Industrial Drive
Pleasanton, CA 94566
(415) 829-2600
Cable Address - PROCTEC

Optional Precision BASIC (Release 1.1, MOD 0) User's Notes

This notice describes the differences between the new Optional Precision BASIC (Release 1.1, MOD 0) and the standard Extended Disk BASIC (BASIC 1.0).

HOW IS THIS BASIC DIFFERENT?

The most obvious difference between Optional Precision BASIC and the standard Extended Disk BASIC is that with the new versions of BASIC, you have selected a precision: 6, 8, 10, 12, 14, or even 16 digits, and asked your dealer to put a BASIC with that precision on an Extended Disk BASIC Data Disk. If you already owned 8-digit precision BASIC, you may have asked your dealer to update a diskette WITHOUT changing the precision, simply in order to take advantage of the other improvements.

The expression "digits of precision" means the maximum number of significant digits used in the representation of a number. This number corresponds to the maximum number of digits that may be accepted as input to the program, the maximum number of digits to be printed or displayed as output from the program, and the maximum number of digits to be returned as the result of a computation. In BASIC 1.0 it is possible to deal with input, output and results having a maximum of 8 significant digits, thus "8 digits of precision." In Optional Precision BASIC, you have selected a precision that will apply to all calculations. The accuracy of all calculated results except those returned by Extended Functions (i.e., SIN, COS, TAN, EXP, SQR, ATN, LOG, LOG10, and exponentiation) will correspond to the precision selected; Extended Functions are calculated using the new precision but retain the accuracy of 8-digit BASIC.

When deciding what precision to buy, you probably considered several factors. The advantage of a large number of digits of precision is that the larger the number of digits of precision, the more accurate many calculations will be; for example, the result of division can be taken to more decimal places. The two other factors to consider are less favorable to the larger numbers of digits of precision: one is that calculations are performed faster when there are fewer digits of precision, and the other is that the same data will occupy more memory if higher precision is used than if lower precision is used.

Here is a summary of improvements that have been incorporated into Optional Precision BASIC.

1) In this new version, BASIC will never use memory above the lowest PTDOS system address, GLLOW. This feature makes it possible for the user to change GLLOW in PTDOS and then continue using BASIC without reinitializing; if the current value of GLLOW signifies an address lower than the top-of-memory address specified when BASIC was initialized, BASIC will use the value of GLLOW, instead of the value provided during initialization.

2) If a negative expression is given in any ON statement, control will fall through to the next statement, i.e., the ON statement will be ignored.

3) In BASIC 1.0, a Record Overflow (RO) error during an attempt to write to a random access file would cause the structure of that file to be damaged. With the new BASIC, file structure will remain intact.

4) If a numeric function with a string argument is specified as the second parameter in the OUT statement, e.g., OUT 0, ASC(T$), BASIC will no longer crash. It is still not permissible to specify a STRING in that position (although a NUMERIC function with a string argument is now acceptable); for example, the statement OUT 0, T$ will still cause an error message to be returned.

5) When RUN is followed by a line number, the data pointer for the READ statement will not be reset. (This rule applies only to the READ statement that reads data from a DATA statement.)

6) In the new BASIC, the SQR function will not be deleted if the other Extended Functions are deleted during initialization.

7) In BASIC 1.0, there was a 64K limit on the size of a random access file; in the new BASIC, there is no limit on the size of such a file.

8) In BASIC 1.0, the last record of a random access file did not have an end-of-record mark. In the new BASIC, all records have end-of-record marks. Any file written in BASIC 1.0 will still be readable in the new BASIC.

9) In BASIC 1.0, a number too large to be printed in the field allocated for it would be printed in exponential notation with an incorrect exponent. In the new BASIC, the exponent is correct.

10) Addition, subtraction, multiplication, and division operations now perform rounding more accurately than in BASIC 1.0, and the SQR function is faster and more accurate. The RND function is in closer conformity with Knuth's algorithm (described in Knuth, THE ART OF COMPUTER PROGRAMMING, v.2).

11) BASIC has some new error messages. The CC (Can't Convert) error replaces the IN error in BASIC 1.0, and has exactly the same significance that the IN error had. The error message IN will now be printed if the ERRSET statement is in effect and non-numeric input is given to a numeric INPUT statement (see #13, below). The UR (UnResolved line number reference) error arises in the situation described in #12, below. The FS error message is reported when a PTDOS error has occurred (see #14, below).

12) If you use the REN command and a statement in your renumbered program contains a reference to a non-existent line number, that statement will be changed so that it refers to line number 0, and the UR error message will be printed. (The renumbering process will never introduce this error to a program that did not already contain it; see Section 3.2 of your manual.)

13) If the INPUT statement expects numeric input and receives a non-numeric character, the following message is printed:

INPUT ERROR, RETYPE

The user is given a second chance to enter a number. If the ERRSET statement is in effect, the IN error message is made available through the ERR(0) function.

14) The new BASIC offers a function called SYST, with the following syntax:

SYST(expr)
where expr evaluates to 0.

EXAMPLE:
```
10 LET X=SYST(0)
20 PRINT "THE PTDOS ERROR WAS",SYST(0)
```

The SYST function returns the number of the last PTDOS error that occurred; this function will normally be used if an FS error has been reported, or if the ERRSET statement has kept an error message from appearing. SYST(0) can be used in conjunction with the ERR(0) function:

```
10 A$ = ERR(0)
20 IF A$(1,2) = "FS" THEN A$ = STR(SYST(0)9
30 IF A$ = "4" THEN PRINT "PROTECTED FILE"
```

Note that PTDOS error numbers, as listed in Section 6 of the first edition of the PTDOS manual, are given as hexadecimal values. SYST(0) returns the PTDOS error number as a decimal value.

15) The SET DS command has been expanded so that it has the following form:

SET DS = exp1{,exp2}

If exp2 is present, it determines whether you can change the display speed during output, and whether you can use the space-bar to stop output to the display temporarily. Here is a table showing the functions allowed for each value of exp2:

Value of exp2	Display speed control?	Space-bar stop?
0	YES	YES
1	YES	NO
2	NO	YES
3	NO	NO

SOME PROGRAMMING CONSIDERATIONS

1) Programs written to run in BASIC 1.0 will also run in Optional Precision BASIC; any given program will occupy no more or less memory than it did before, although the data stored during the execution of the program will take up more or less memory, depending on the precision of the new BASIC.

2) BASIC itself will now require more memory, regardless of precision. Before initialization, BASIC will now occupy memory locations from 100H to 4FFFH, inclusive. The initialized BASIC interpreter will also require more memory than it did in Extended Disk BASIC.

3) Programmers may want to change existing INPUT and PRINT statements to accept or print out numbers having of the new number of significant digits. (This consideration is important in the case of unformatted PRINT statements, because a longer number will occupy more character positions than before, and so affect the printing format. Formatted PRINT statements, on the other hand, need not necessarily be changed; just as in BASIC 1.0, if the field width allocated for a number is less than its precision, the number will be rounded to fit in the field. (See the discussion of formatted PRINT in the Extended Disk BASIC User's Manual.)

4) Commas separating elements in a PRINT statement will still cause each item of information to be printed at the beginning of a 14-character field, even in 16-digit BASIC. Thus, if two numbers are printed on a line in 16-digit BASIC, the first number will overflow to the second field, and the second number will be printed at the beginning of the third field.

5) The maximum field width to be used for free format output in floating point or exponential form now varies, depending on the precision of the BASIC. The formula for determining maximum field width is:

$$\text{maximum field width} = \frac{(4 * \text{precision}) + 15}{3}$$

ProcessorTechnology

Processor Technology Corporation

7100 Johnson Industrial Drive
Pleasanton, CA 94566

(415) 829-2600
Cable Address - PROCTEC

Extended Disk FORTRAN Update

Subject: Additional Programs on FORTRAN Diskette

The FORTRAN diskette which this update accompanies includes six files not described in the Extended Disk FORTRAN manual: ASSM, CLOCK.F, DIGIT, STAT.F, CMPLX.F, and SOLGO which is information-protected.

The file ASSM is an updated version of the disk assembler that must be used with the FORTRAN compiler. Since it is appreciably faster than the previous version, you will probably want to use it for any future assembly language programming as well. The ASSM command syntax has not changed, and the ASSM Subsystem Manual applies without update.

With a PTDOS system diskette in unit Ø (make sure you have a backup) and the FORTRAN data diskette in unit 1, the old version may be replaced with the PTDOS command line:

*REATR ASSM; GET /Ø, I=/1, ASSM, S=-I; REATR ASSM,KWINE <CR>

The asterisk * is the prompt provided by PTDOS. <CR> represents the typing of the CARRIAGE RETURN key.

The file CLOCK.F is the FORTRAN source code for a digital clock demonstration program that uses the PLOT subroutine to directly place a clock display on the video screen. The display is updated every five seconds. Before creating an executable image file, which will be called CLOCK, you must move the data file DIGIT to unit Ø where CLOCK expects to read it. With the FORTRAN diskette in unit 1 and a PTDOS system diskette in unit Ø, use the following PTDOS command:

*COPY DIGIT/1, DIGIT <CR>

Now CLOCK.F may be compiled and run with the PTDOS command line:

*FORTRAN CLOCK.F/1,,,CLOCK; CLOCK <CR>

The above command line creates an image file CLOCK on the default unit (normally unit Ø) from the FORTRAN source file CLOCK.F on unit 1. CLOCK may be rerun later by merely typing its name:

*CLOCK <CR>

When CLOCK begins executing, it will ask whether 12 hour or 24 hour format is desired and what the starting time is to be. A starting time 30 to 60 seconds in the future should be specified to the nearest ten seconds; the seconds unit's position must be 0. When an accurate time reference reaches the specified starting time, type any key. CLOCK will begin running at the specified time. Pressing the MODE key or the CONTROL and @ keys together will abort CLOCK and return control to PTDOS.

The value of IFACT, initialized near the beginning of the program, is used to control the clock's timing. It may have to adjusted slightly to regulate the speed of the clock. Increase IFACT if the clock runs fast; decrease it if the clock runs slow.

The file STAT.F is the FORTRAN source code for a demonstration program that performs a simple statistical analysis of data contained in a user-specified file. It may be compiled and run by typing the PTDOS command line:

```
*FORTRAN STAT.F/l,,,STAT; STAT <CR>
```

and rerun later by typing the command:

```
*STAT <CR>
```

When STAT begins execution, it will ask for the name of a data file to analyze and whether or not the user desires an ordered listing of the data on that file. It then produces a self-explanatory statistical summary on a file chosen by the user. The data file read by STAT is simply a collection of at least 10 and no more than 3000 numeric values separated by carriage returns. These values may be integer, floating point, or exponential values since they are read by a free-format READ statement. See Section 5.5.1. of the manual before creating a data file.

The file CMPLX is the FORTRAN source code for a set of subroutines that perform complex addition, subtraction, multiplication, division, absolute value, square root, and can also be used for conversion between rectangular and polar coordinates. They may be included in a user-written program by means of a FORTRAN COPY statement.

The last file, SOLGO, is an information-protected bonus program. Its purpose will be revealed in the future. Do not KILL it, but do not ask about it yet.

ProcessorTechnology

Processor Technology
Corporation

7100 Johnson Industrial Drive
Pleasanton, CA 94566

(415) 829-2600
Cable Address: PROCTEC

FORTRAN Update No. 2

February 27, 1979

FORTRAN UPDATE

SUBJECTS:

Distinction between "terminal" and "console" in Extended Disk FORTRAN Errata and Addenda to Extended Disk FORTRAN User's Manual, Revised Descriptions of Subroutines, Appendices 6 and 7.

MANUALS AFFECTED:

Extended Disk FORTRAN User's Manual, Manual Part No. 727101

CURRENT PUBLICATIONS:

731040

With the information contained in this update, the Extended Disk FORTRAN User's Manual describes Extended Disk FORTRAN, Release 1.1. (Without the update, the manual describes Release 1.0.)

The modifications described herein are of several sorts. Some are errata to the existing documentation; others, addenda reflecting improvements that have generated Release 1.1 of FORTRAN from Release 1.0. Still others are elaborations whose intent is to clarify certain aspects of FORTRAN not treated at length, or not given emphasis in the Extended Disk FORTRAN User's Manual.

Short revisions have been keyed to the pages in the manual where they should appear. The errata and addenda in the second part of the update are of this type. Longer revisions have not been keyed to pages; for the most part, they replace recognizable counterparts in the existing manual (e.g., the new description of a subroutine replaces the description of that subroutine in the manual). The treatment of "terminal" and "console," below, is a supplement to the manual, rather than a revision of existing material.

DISTINCTION BETWEEN "TERMINAL" AND "CONSOLE" IN EXTENDED DISK FORTRAN

The word "terminal," as used in the FORTRAN User's Manual, does not have exactly the same meaning as the word "console." The terminal comprises logical units 0 and 1, which may or may not correspond to the console input and output devices, respectively.

FORTRAN logical units 0 and 1 are permanently associated with the PTDOS Command Interpreter (CI) input and output files. (See the PTDOS User's Manual.) When the system is bootstrapped, the CI input file is file #0, the console input device, and the CI output file is file #1, the console output device. If nothing is done to change those settings, the FORTRAN ACCEPT and READ(0,...) statements will refer to the console input file, and the TYPE and WRITE(1,...) statements will refer to the console output file. In this case, the "terminal" will actually correspond to the "console."

In other cases, the CI input and output files are not associated with PTDOS files #0 and #1, i.e., the "terminal" does not actually correspond to the "console." The PTDOS SETIN and SETOUT commands explicitly change the assignment of CI input and output files; the CI input file assignment may be changed implicitly and temporarily by the DO command macro preprocessor.

If the CI input file assignment has been changed, the FORTRAN ACCEPT and READ(0,...) statements no longer refer to the console input device. Likewise, if the CI output file assignment has been changed, the FORTRAN TYPE and WRITE(1,...) statements no longer refer to the console output device. For example, if a FORTRAN program is executed as part of a command file (which automatically becomes the CI input file), an ACCEPT statement in that program will expect its input not from the console, but from the command file. Similarly, if CI output has been directed to a disk file, TYPE will send its output to that disk file, rather than to the console.

Unlike the TYPE and ACCEPT statements, READ and WRITE may use the console input and output files, respectively, even if logical units 0 and 1 are not currently associated with those files. The special file names $CONIN and $CONOUT, recognized by FORTRAN, always refer to the console, PTDOS files #0 and #1. If you wish to write a FORTRAN program that will take its input from the console, even if the CI input file has been changed, use the OPEN statement to associate a logical unit (other than 0 or 1) with $CONIN, and READ from that unit. If you want to guarantee that the output from a program will go to the console, even if CI output has been redirected, use the OPEN statement to associate a logical unit (other than 0 or 1) with $CONOUT, and WRITE to that unit. (The reason not to use 0 or 1 is that those unit numbers will already be associated with the current CI input and output files.)

ERRATA AND ADDENDA TO EXTENDED DISK FORTRAN USER'S MANUAL

The following changes should be made on the specified pages of the manual.

p 1-3

Add the following to the list of files required to use PTDOS FORTRAN:

```
RFORTGO  - FORTGO ORGed above D000H
FORTDEFR - definitions for use with RFORTGO
```

Add FORTDEFR to the list of files residing on the default unit. Add RFORTGO to the list of files residing on unit 0.

p 2-1

Replace the text of Section 2.1.2 with the following:

If a statement is too long to fit on a single line, it may be continued on one or more additional lines. Each continuation line must have a character other than blank or 0 in column 6 and blanks in columns 1-5.

If a character string is continued, its interpretation depends on whether or not the P=n parameter was specified on the FORTRAN command line. If this parameter is omitted, the statement lines are not padded with blanks between the final carriage return and column 72. For example, the statement

```
      TYPE 'THIS STATEMENT IS CONTINUED<CR>
     * BELOW'
```

outputs the line

```
THIS STATEMENT IS CONTINUED BELOW
```

to the terminal.

If P=72 appeared in the FORTRAN command line, the line output by the above example would have 24 blanks separating the last two words. (If P=64 appeared, there would be 16 blanks.)

p 2-3

Add the following at the bottom of the page:

It is also possible to specify a string constant in Hollerith format, i.e., by preceding the string with a decimal number and an H. The number specifies the length of the string that follows the H. No terminal delimiter is used. For example:

```
28HTHIS IS A HOLLERITH CONSTANT
```

p 2-4

In the fourth paragraph of Section 2.2.2, delete the phrase "but you can only assign up to four characters to an integer variable using an assignment statement."

Replace the last paragraph with the following:

A variable may not have the same name as one of the system functions listed in Appendix 3. In addition, COPY may not be used as a variable name.

p 3-2

Add the following to the first paragraph of Section 3.2:

The source program may be composed of no more than 62 routines, including the main program. System subroutines and functions referenced by the program are not included in this count.

p 3-3

Add the following to the second paragraph:

The $ preceding an OPTIONS declaration is optional.

p 3-4

Replace the description of the X option with the following:

Causes line numbers to be listed during execution tracing or when a runtime error occurs.

p 4-2

Add S=R and P=n to the list of options allowed in the FORTRAN command line.

Add the following to Section 4.1.1:

S=R

The S=R parameter specifies that the runtime package used by the quick-compile option will be loaded above D000H.

P=n

The P=n parameter specifies that input lines will be blank-padded out to column number n. (A carriage return will be converted to a blank also.) The only values allowed for n are 64 and 72.

p 4-4

Add SET UNIT to the list of runtime errors that may be caused by PTDOS operations.

Replace the second paragraph of Section 4.2.2 with:

PTDOS extends downward in memory from BFFFH to 9000H or below, depending on the size of the system-managed buffer area. The SOLOS ROM and scratchpad memory occupy the space from C000H to CBFFH, and the video display memory (Sol or VDM) extends from CC00H to CFFFH. (See the PTDOS User's Manual for further information on memory management.) A compiled FORTRAN program normally resides in user memory from 100H to the bottom of the PTDOS buffer area (e.g., 9000H). However, in a system with 64K of memory, the 12K extending from D001H to FFFFH is also available as user memory.

p 5-2

Replace the last paragraph of the description of assignment statements with the following:

A character string with as many as six characters may be assigned to a variable or array element.

p 5-10

Replace the list of codes and messages with the following:

The error codes and corresponding runtime error messages are:

1	INTEGER OUT OF RANGE
2	16 BIT CONVERSION ERROR
3	ARGUMENT COUNT ERROR
4	COMPUTED GOTO INDEX OUT OF RANGE
5	FLOATING POINT OVERFLOW
6	DIVIDE BY ZERO
7	SQRT OF NEGATIVE NUMBER
8	LOG OF NEGATIVE NUMBER
9	CALL STACK PUSH ERROR
10	CALL STACK POP ERROR
11	FILE OPERATION ERROR
12	ILLEGAL LOGICAL UNIT NUMBER
13	LOGICAL UNIT ALREADY OPEN
14	OPEN ERROR
15	LOGICAL UNIT NOT OPEN
16	SET UNIT ERROR
17	LINE LENGTH ERROR
18	INVALID FORMAT
19	I/O ERROR
20	INPUT ERROR
21	INVALID I/O LIST
22	ASSIGNED GOTO LABEL NOT IN LIST

The runtime error messages are explained in Appendix 6.

p 5-16

Replace the second sentence of the section entitled "Field Specifications: string" with the following:

Strings appearing in FORMAT statements may be delimited by single quotes or expressed in Hollerith format, i.e., with a preceding length count and H.

p 5-19

Add the following:

Field Specifications: Tw

The T tabs within the current record: i.e., the pointer specifying the next character to be input or output is positioned at character w of the input or output record. (The first character is numbered 1.)

For example:

```
      READ (0,10) I,J,K,I2
10 FORMAT (3I1,T1,I1)
```

In this example, the first, second, and third digits are read into variables I, J, and K, respectively. Then the pointer is moved back to character position 1 in the input line, so that the first digit is read again, this time into variable I2.

p 5-21

The second sentence in the first paragraph of Section 5.5.3 should end:

"...associate a logical unit number with the file name."

The following should be added to the end of that paragraph:

Logical unit numbers may range from 0 to 63. Logical units 0 and 1 refer to the PTDOS Command Interpreter input and output files, respectively.

The section entitled "File Unit Numbers" should be deleted.

p 5-26

Add the following to Section 5.5.5:

Unlike the formatted READ and WRITE statements, binary READ and WRITE statements cannot have empty input and output lists. (Such statements would have no effect.)

Add the following section:

5.5.6 DECODE and ENCODE Statements

The FORTRAN WRITE statement, discussed above, is a way of converting variables of different types and lengths, as well as literals, into a character string that appears either on an output device or in an output file. Conversely, the READ statement interprets a character string existing in an input file (or device), so that the components of that string may be manipulated separately within the FORTRAN program. PTDOS FORTRAN offers two statements whose operation is similar to that of WRITE and READ, except that the character string is situated in memory, rather than on an external device or file.

DECODE is like READ; it translates a character string into a series of values and assigns those values to items in an input list. It differs from READ in that the record being "decoded" is a string in memory, rather than a string typed on the console or read from another file.

ENCODE is like WRITE; it builds a character string from a series of values whose names are given in an output list. It differs from WRITE in that the record being "encoded" is put into a string in memory, rather than written on the display device or another file.

DECODE Statement

General form:

DECODE(string,length,format) input list

Under control of the specified format, interprets the character string in string, and assigns values to variables in the input list.

where string is a variable name or array name (not an array element) specifying the beginning address of the character string;
length is a number or variable specifying the length of the string in bytes, or the length of each record, if the format prescribes multiple records;
format is a format number or the name of a variable or array containing the format to be used, or an * to indicate free format; and
input list is a list of the variables into which the values derived from the character string will be placed.

Format and input list are subject to the same restrictions as in the READ statement.

DECODE is similar to READ. The character string beginning at location string and continuing for length bytes is read into the variables in the input list, according to the prescribed format. For example, the sequence of statements

```
C DECODE DATE INTO MONTH, DAY AND YEAR SO THAT JULIAN DATE MAY BE
C CALCULATED. EVERY TWO MEMBERS OF DATE CONTAIN A STRING OF THE FORM
C MM/DD/YY.
      DIMENSION DATE(2)
      DECODE(DATE,8,2000) MONTH, IDAY, IYR
 2000 FORMAT(I2,1X,I2,1X,I2)
```

will result in the assignment of the appropriate two-digit integers to the items in the input list. For example, if the string beginning at DATE(1) consists of the characters 12/08/75, MONTH will receive a value of 12, IDAY will receive a value of 8, and IYR will receive a value of 75. (The slashes will not be assigned, because the 1X's in the format cause them to be ignored.)

It is possible to DECODE a character string consisting of multiple records, if the format contains slashes (/). In that case, a slash indicates that the next item in the list should be read from the next record of the string, i.e., from the next group of length bytes in the string. For example, the string

```
'ABCDEFGHIJ'
```

might be decoded as a single record of length 10, or as multiple records, e.g., two records of length 5.

If a record in string is not as long as the format and input list would suggest - that is, if length is less than the number of characters required to satisfy the list - the rest of the input list will be filled as though there were additional blanks in the input record: string variables will be blank-filled, and numeric variables will be zero-filled. If a record is longer than the format and input list would suggest - that is, if length is greater than the number of characters required to satisfy the list, the list will be satisfied, and the rest of the record will be ignored.

ENCODE Statement

General form:

ENCODE(string,length,format) output list

Under control of the specified format, constructs a string consisting of the values named in the output list, and places that string in string.

where string is a variable name or array name (not an array element) specifying the beginning address of the character string;
length is a number or variable specifying the length of the string in bytes, or the length of each record, if the format prescribes multiple records;
format is a format number or the name of a variable or array containing the format to be used, or an * to indicate free format and
output list is a list of the variables whose values will be used to construct the character string.

Format and output list are subject to the same restrictions as in the WRITE statement.

The ENCODE statement is similar to WRITE. The values corresponding to the items in the output list are written, in order and according to the prescribed format, to memory locations beginning at string. For example, after the sequence of statements

```
C ENCODE A SUBTITLE IN ARRAY SBTITL
C
      DIMENSION SBTITL(3)
      DATA IAREA /5/, IREG /'WEST '/
      ENCODE(SBTITL,14,1000) IAREA, IREG
 1000 FORMAT('AREA ',I1,' - ',A5)
```

SBTITL will contain the character string AREA 5 - WEST . Notice that literals indicated in the format, as well as values named in the output list, are represented as characters in SBTITL.

The value of the length argument determines the length of the output record. If the string defined by the output list and format is longer than the specified number of bytes, only length bytes will be written to the output record, and the rest of the output list will not be encoded. If the string defined by the output list and format is shorter than the specified number of bytes, the remainder of the output record will be padded with blanks.

It is possible to ENCODE a character string consisting of multiple records, if the format contains slashes (/). In that case, the records are stored sequentially, and each record is length characters long. Records will not end with carriage returns; unless a carriage return is included explicitly in the format, i.e., as an ASCII value between backslashes in a literal string, no carriage returns will be put into string.

p 5-31

The first sentence should end:

" ... filled with blanks on the right."

In the subsequent example, change "nulls" to blanks."

p 5-32

Add the following to the description of the COMMON statement:

If an array name appears in a COMMON statement without dimension information, it must be dimensioned in a preceding DIMENSION, INTEGER, REAL, or LOGICAL statement.

p 5-40

Add the following:

5.9 EXECUTION TRACING

PTDOS FORTRAN provides an execution tracing facility for use in debugging programs. If tracing is enabled, FORTRAN will list on the console the name of each subprogram as it begins execution. In addition, for any routine that contains an OPTIONS declaration including the X option, the line number of each statement executed will be listed on the console. The line number displayed upon entry to a subprogram is always ????.

The form of the messages displayed on the console is:

```
Pgm is executing line number in routine name
```

where number is the line number (or ???? upon entry to a subprogram), and name is the name of the routine being executed.

Execution tracing is enabled and disabled by means of the following statements:

General form:

```
TRACE ON      Enables execution tracing.
TRACE OFF     Disables execution tracing.
```

p 6-1

In the third sentence of Section 6.1.1, delete the phrase "or chains to another program." After that sentence, add the following:

Open files will be closed automatically by the CHAIN subroutine unless its second argument is negative (see Section 6.4).

p 6-2

Delete the paragraph beginning "READ and WRITE statements ...", including the example.

p 7-1

The description of the ISIGN function should read:

The sign of exp; +1 if positive, -1 if negative, 0 if zero.

p 7-2

Delete the "f" preceding AMOD in the first example.

p A1-1 and A1-2

Add the following entries to the statement summary:

DECODE (string,length,format) input list	Reads values from the character string in string and assigns them to variables in the input list.
ENCODE (string,length,format) output list	Writes values from the output list to string.
TRACE ON	Enables execution tracing.
TRACE OFF	Disables execution tracing.

p A2-1

The following entries replace those in the list:

CALL BIT(var,disp,'op'{,result})	Sets, resets, flips, or tests a single bit.
CALL CBTOF (loc,disp,var{,flag})	Converts a binary number to its floating point equivalent.
CALL CHAIN ('program name'{,action})	Chains to another program.

Add the following entry to the list:

CALL OUT (port,value)	Outputs value to a port.

p A5-3

Make the following changes in the list of compilation error codes:

6F	Illegal trace statement
71	Comma missing in ENCODE or DECODE statement
74-7F	NOT USED
80	*FATAL* no program to compile

p A8-1

Reference 6 should be:

Programming Languages: History and Fundamentals
Jean E. Sammet
Prentice-Hall, Inc. 1969

REVISED DESCRIPTIONS OF SUBROUTINES, APPENDICES 6 AND 7

The following descriptions--as well as the revisions of Appendices 6 and 7--replace their counterparts in the manual. Where such a counterpart does not exist, the subroutine or function was added to FORTRAN after the release of the manual.

BIT Subroutine

General form:

```
CALL BIT (var, disp, 'action' {,result})
          /     \        \             \
variable  name   \ S,  R,  F,  or  T   \
                  expression             variable name
```

Sets, resets, flips, or tests a single bit of the variable var.

Examples:

```
CALL BIT (NEWVAR, 0, 'S')
CALL BIT (COUNT, 1, 'T', B1)
```

The BIT subroutine sets, resets, flips, or tests a single bit of the variable var. The third argument specifies which action should be taken.

'S' means set the bit to 1;
'R' means reset the bit to 0;
'F' means flip the bit, i.e., change 0 to 1 and vice versa
'T' means test the bit and return its value in the fourth argument.

The fourth argument must appear if the third argument is 'T', and may not appear otherwise.

The value of the expression disp determines which bit of var will be affected. To determine the proper value for disp, consider the internal format of a stored value:

```
              byte  byte  byte  byte  byte  byte
               0     1     2     3     4     5
    value=    nn    nn    nn    nn    0s    ee
                                      /     /
                                   sign exponent
```

Where each n is a digit, and each byte consists of 8 bits numbered 0 to 7, from right to left. The value to assign to disp is given by the equation:

$$disp = (byte * 8) + bit$$

Where "byte" is the number of the desired byte (0 to 5), and "bit" is the number of the desired bit (0 to 7) within that byte. For example, to set the last bit of a value (i.e., bit 0 of byte 5), use

$$disp = (5 * 8) + 0 = 40$$

To set the first bit of a value (i.e., bit 7 of byte 0), use

$$disp = (0 * 8) + 7 = 7$$

CBTOF Subroutine

General form:

```
CALL CBTOF (loc, disp, var [, flag])
           /     /       \       \
    variable  number of  variable  any
    name or   bytes              value
    memory
    address
```

Converts an 8- or 16-bit unsigned binary number whose location is determined by loc and disp into a floating point number, and stores that number in the variable var.

Examples:

```
CALL CBTOF (BVAR,0,DVAR)
CALL CBTOF (ARAY(1,1),6,VAL(2))
CALL CBTOF (X,1,Y,1)
```

The CBTOF subroutine converts an 8- or 16-bit unsigned binary number into its decimal floating point equivalent and stores the result in var.

If disp is positive or zero, loc is assumed to be the variable or array element that contains the binary form of the number. Disp specifies the displacement in bytes from the first byte of loc to the first byte of the binary number. For example, if the binary number occupies the second and third bytes of loc, the value of disp should

be 1; if the binary number occupies the first byte of loc, the value of disp should be 0.

If disp is negative, loc is assumed to contain or be an absolute memory address. If loc is a variable name or array element, that variable is assumed to contain the address of the first byte of the binary number. To specify an absolute address as the first argument, use the form $addr, where addr is a hexadecimal address.

If flag is omitted, the binary number is assumed to be a 16-bit quantity stored low-order byte first and ranging from 0 through 65535. If flag is present, the binary number is assumed to be an 8-bit quantity ranging from 0 through 255.

Program example:

```
   DIMENSION A(4)
   CALL FINFO('SUTIL',A)
   CALL CBTOF(A(1),0,ID)
   CALL CBTOF(A(1),6,NBLKS)
   CALL CBTOF(A(1),12,BLKSZ)
   WRITE(1,10)ID,NBLKS,BLKSZ
10 FORMAT('ID = ',I5,' # BLOCKS = ',I5,
  &' BLOCKSIZE  = ',I5)
   END
```

This program retrieves status information for disk file SUTIL in binary form. It converts the information to decimal form and displays it at the terminal.

CHAIN Subroutine

General form:

CALL CHAIN ('file'{,action})	Loads the specified file into memory, and executes it if it has a starting address.

Examples:

```
CALL CHAIN ('PART2')
CALL CHAIN ('OVERLAY1',-1)
```

The CHAIN subroutine loads an image format file into memory at its normal load address, and executes it if it has a starting address. The specified file may be any existing image format file; because CHAIN is used most frequently to chain FORTRAN programs, all of which are loaded at 100H, the loaded program usually overwrites all or part of the calling program.

If the specified file does not exist or if an error occurs during loading, a FILE OP error occurs.

If the loaded program contains a starting address, it will begin execution immediately at that address; if not, the CHAIN subroutine

will return control to the statement that follows the call to CHAIN. If it is necessary to pass an argument to an assembly language program that is loaded by CHAIN, that program should contain no starting address and should be executed by the CALL function. (See Section 7.4.)

If the loaded program does not overwrite the caller, an 8080 return instruction will return control to the FORTRAN statement immediately following the call to CHAIN. If it is necessary to return a value to the calling program, an assembly language program loaded by CHAIN should be executed by the CALL function.

The optional second argument of CHAIN may be used to specify whether the runtime package is to be reloaded and whether or not files left open by the calling program are to be closed by CHAIN. If the value of the action argument is negative, the runtime package will not be reloaded and open files will be left open. If the value of action is greater than zero, the runtime package will not be reloaded and open files will be closed. If action is equal to zero or is omitted, the runtime package will be reloaded and open files will be closed.

Note that if the runtime package is not reloaded, a loaded FORTRAN program must expect the runtime package to reside at the same location as did the program that called CHAIN, i.e., neither or both of them were compiled with the S=R option, and neither or both of them were compiled with the long compile option.

Program Example:

P1 is the name of the image file containing the object code for the following program:

```
TYPE 'THIS IS PART 1'
CALL CHAIN ('P2')
END
```

P2 is the name of the image file containing the object code for the following program:

```
TYPE 'THIS IS PART 2'
END
```

Execution:

```
User:                P1 <CR>
First program:       THIS IS PART 1
Second program:      THIS IS PART 2
                   STOP  END IN - MAIN
```

General form:

```
CALL CONTRL (unit,op,DEin,HLin,Aout,DEout,HLout)
             /    /    \  /        \    /    /
logical unit  operation \  values returned in
number        code       \ A,HL, and DE registers
                          \
                        values supplied
                        in DE and HL
                        registers
```

Allows program control over devices or returns information about devices.

Examples:

```
CALL CONTRL (0,2,0,'?',X,Y,Z)
CALL CONTRL (FILE,4,0,$6500,X,Y,Z)
```

The CONTRL subroutine provides a mechanism by means of which a FORTRAN program can make a Control/Status (CTLOP) system call. The CTLOP system call allows a user program to control a peripheral device or obtain information about its status.

A detailed description of the various operations that may be performed and the significance of the data supplied or returned in the A, DE, and HL registers may be found in Sections 7.2 and 9.2.2 of the FTDOS User's Manual, Second Edition.

Examples:

```
CALL CONTRL (0,2,0,'?',X,Y,Z)
```
Sets the console input prompt to ?.

```
CALL CONTRL (2,4,0,$6500,X,Y,Z)
```
Moves the index block of the random file associated with logical unit 2 from disk to memory location 6500H.

Program Example:

```
IMPLICIT INTEGER (A-Z)
CALL CONTRL (7,0,0,0,AR,DE,HL)
TYPE 'PROTECTION = ',AR
TYPE 'CHRS = ',DE
END
```

This program displays the protection attributes and device characteristics of logical unit 7.

General form:

```
CALL  MOVE  (n,locl,displ,loc2,disp2)
           /   /     /     /     \
expression   /expression  / expression
           /              /
   character string     character string
   or memory address    or memory address
```

Moves n bytes from locl to loc2. If either disp is positive or zero, the corresponding loc is the symbolic name of the starting location. If disp is negative, loc contains or is an absolute memory address.

Examples:

```
CALL MOVE (6,'ABCDEF',0,$CC00,-1)
CALL MOVE (2,A,-1,$CC00,-1)
CALL MOVE (1024,$CC00,-1,A,-1)
CALL MOVE (10,COUNT,2,ADD1,-1)
```

The MOVE subroutine moves n bytes from locl to loc2. The interpretations of locl and loc2 depend on the values of displ and disp2, respectively.

If either disp is positive or zero, the corresponding loc is assumed to be the symbolic name of the location containing the first byte of the string to be moved (if locl), or the first byte of the destination for the moved string (if loc2). Thus, locl or loc2 may be a variable name, an array name, or an array element. Locl may also be a literal string enclosed in single quotes; in that case, the literal string contains the first and subsequent bytes of the string to be moved. In any of these cases, disp specifies the displacement in bytes from the first byte of loc to the actual starting byte.

If either disp is negative, the corresponding loc is assumed to contain or be an absolute memory address. If loc is a variable name or array element, that variable is assumed to contain the address of the first byte to be moved, or the first destination location. To specify an absolute memory address, use the form $addr, where addr is a hexadecimal address.

Explanation of examples:

```
CALL MOVE (6,'ABCDEF',0,$CC00,-1)
```
Moves ABCDEF to address CC00H

```
CALL MOVE (2,A,-1,$CC00,-1)
```
Moves 2 bytes from the address stored in A to address CC00H

```
CALL MOVE (1024,$CC00,-1,A,-1)
```
Moves 1024 bytes starting with address CC00H to the address specified by A.

```
CALL MOVE (10,COUNT,2,ADD1,-1)
```
Moves 10 bytes, starting at the third byte of COUNT, to locations starting at the address stored in ADD1.

OUT Subroutine

General form:

CALL OUT (port, value) Outputs value to the specified port.

Example:

```
CALL OUT (254,0)
```

The OUT subroutine enables a FORTRAN program to output an 8-bit value to a specified hardware port. If value or port lies outside the range 0-255, its value modulo 256 is used.

Explanation of example:

```
CALL OUT (254,0)
```

This example outputs a zero to port FEH, the Sol video display scroll control register. As a result, all 16 lines of the video memory will appear on the screen, with line 0 at the top.

APPENDIX 6

EXECUTION ERRORS

A program may compile correctly and still generate "execution errors" at runtime. Unless an execution error is trapped by means of an ERRSET statement, it causes a message to be displayed and execution of the program to be terminated.

Execution error messages have the form:

Runtime error: message, called from location

Pgm was executing line number in routine name

where:

message is one of the messages shown below.

location — is the memory location of the error-producing call to the runtime package.

The locations assigned to individual statements are NOT indicated on any listing generated by FORTRAN. To obtain a listing with locations shown, first compile the program, specifying the B option (to get FORTRAN statements interspersed with assembly language code); then use the PTDOS assembler to assemble $FORTASM (or the "assembly" file named in the FORTRAN command line) and specify the listing option.

number — specifies the line in which the error occurred. That number will be intelligible only if the X option was declared in the named routine. (See the description of the OPTIONS statement in Section 3.2.) Otherwise, the line number will appear as ???? If several line numbers are listed, the error actually occurred in the first line specified.

name — is the name of the routine in which the error occurred.

Unless they are trapped, four of the execution errors - namely, FILE OP, I/O ERR, OPEN ERR, and SET UNIT - result in calls to the PTDOS Explain Error Utility (UXOP). UXOP supplies a detailed explanation of the error and returns control to PTDOS. After one of these errors has occurred, it is no longer possible to rerun the FORTRAN program in memory, because UXOP uses memory occupied by the runtime package. In order to run the program again, you must reload it from disk.

The error code for any runtime error may be supplied as an argument in the ERRSET statement, which has the form:

```
ERRSET n, v
```

and signifies "If error v occurs, transfer control to the statement labeled n. (See page 5-10 for a description of ERRSET.) It is advisable to assume that any variable will be undefined after an error that involves it.

The possible execution errors and their ERRSET codes are:

Code	Message	Meaning
3	ARG CNT	ARGUMENT COUNT ERROR Too many or too few arguments were passed to a subprogram.
22	ASN GOTO	ASSIGNED GOTO LABEL NOT IN LIST The variable in an assigned GOTO statement did not have the value of any label in the list following it (n1, n2, etc.).
10	CALL POP	CALL STACK POP ERROR A RETURN without a destination was executed. This error can be caused only by a user's assembly language program.
9	CALL PSH	CALL STACK PUSH ERROR More than 62 nested subprogram calls were made.
4	COM GOTO	COMPUTED GOTO INDEX OUT OF RANGE The variable specified in a computed GOTO statement has a value either less than one or greater than the number of statement labels specified.
2	CONVERT	16 BIT CONVERSION ERROR An overflow has occurred during the conversion of a number to internal 16-bit binary form. This error can occur during 1) the conversion of a unit number in an input/output statement, 2) the evaluation of a subscript, or 3) the conversion of a floating-point number to 16-bit binary form.
6	DIV ZERO	DIVIDE BY ZERO An attempt was made to divide by zero
11	FILE OP	FILE OPERATION ERROR An error occurred during an operation involving a PTDOS file. This error results in a call to UXOP; see the note above.

18	FORMAT	INVALID FORMAT A formatted READ or WRITE referred to an invalid FORMAT specification.
12	ILL UNIT	ILLEGAL LOGICAL UNIT NUMBER A logical unit number less than 2 or greater than 63 was passed to one of the input/output rougtines.
20	INPT ERR	INPUT ERROR The representation of a number in the input file contained an invalid character. Examples of invalid characters are a second decimal point in a number, an E in an F-type field, a decimal point in an I-type field, etc.
1	INT RANG	INTEGER OUT OF RANGE An integer operation resulted in a number too large to be stored.
19	I/O ERR	I/O ERROR An error occurred during a READ or WRITE operation, and there was no error label (for an error other than an end-of-file) or no end-of-file label (for an end-of-file error). This error results in a call to UXOP; see the note above.
21	I/O LIST	INVALID I/O LIST A formatted READ or WRITE statement contained an error in the input or output list. This error can occur only if the I/O list was constructed by a user's assembly language program.
17	LINE LEN	LINE LENGTH ERROR An attempt was made to read or write a record more than 250 characters long.
8	LOG NEG	LOG OF NEGATIVE NUMBER ALOG or ALOG10 was called with a negative argument.
14	OPEN ERR	OPEN ERROR An error occurred during the execution of an OPEN statement. This message encompasses all OPEN errors, except the case of a nonexistent file. (If the file named in the OPEN statement does not exist, it is created.) This error results in a call to UXOP; see the note above.
5	OVERFLOW	FLOATING POINT OVERFLOW A floating-point operation resulted in a number too large to be stored.

16	SET UNIT	SET UNIT ERROR An error occurred during reassignment of the default unit (drive) number. This error results in a call to UXOP; see the note above.
7	SQRT NEG	SQRT OF NEGATIVE NUMBER The SQRT function was called with a negative argument.
15	UNIT CLO	LOGICAL UNIT NOT OPEN There was an attempt to read, write, rewind, or perform some other operation on a logical unit that was not associated with a file.
13	UNIT OPN	LOGICAL UNIT ALREADY OPEN A file has been assigned a logical unit number already assigned to another file.

APPENDIX 7

COMPARISON OF PTDOS AND ANSI STANDARD FORTRAN

Extended Disk FORTRAN includes these extensions to ANSI standard FORTRAN (X3.9-1966)

* File management, including creating, killing, and changing attributes.
* Random access to data files
* Input from and output to device files, including character-at-a-time terminal input
* Direct control over the video display
* Free format input and output
* Optional end-of-file and error branches in READ and WRITE statements
* Arrays with up to seven dimensions
* Hexadecimal constants
* Character string data type and string move and compare routines
* ENCODE and DECODE statements for formatting data in memory
* IMPLICIT statement for implicit typing of variables and functions
* COPY statement to copy files of source statements into a FORTRAN source program
* Assembly language interface - embedded assembly language statements and calls to assembly language routines
* Access to absolute memory locations, including individual bits
* Program-controlled time delay
* Pseudo-random number generator function
* Program control of runtime error trapping
* Execution tracing
* Ability to chain a sequence of programs

PTDOS FORTRAN does not conform to ANSI X3.9-1966 in the following respects:

* Double precision variables, constants, functions, and format specifications are not provided.

* Complex variables, constants, and functions are not provided.

* There is no EQUIVALENCE statement.

* ANSI FORTRAN allows DATA statements such as:

```
DATA X,Y,Z/10,20,30/
```

  In PTDOS FORTRAN this statement would have to be changed to:

```
DATA X/10/,Y/20,Z/30/
```

* Statement functions are not allowed.

* The following format specifications are not available: carriage control characters, D, G, and P.

* If an array name appears without dimension information in a COMMON statement, it must be dimensioned in a preceding DIMENSION, INTEGER, REAL, or LOGICAL statement.

* A variable may not have the same name as one of the system functions listed in Appendix 3. COPY may not be used as a variable name.

* Only the first five characters of function or subroutine names or COMMON labels are retained. For example, PTDOS FORTRAN does not differentiate between MYSUB1 and MYSUB2.

* The following cannot be used for subroutine, function or COMMON names: A, B, C, D, E, H, L, M, SP, PSW, or any PTDOS reserved name contained in the files PTDEFS and NPTDEFS.

* The SIGN and ISIGN functions provided by PTDOS FORTRAN have a single argument. In ANSI FORTRAN, these functions have two arguments and transfer the sign of the second argument to the first.

* The hyperbolic tangent function, TANH, is not provided.

Processor Technology

Processor Technology Corporation
7100 Johnson Industrial Drive
Pleasanton, CA 94566
(415) 829-2600
Cable Address PROCTEC

8080 CHESS UPDATE 731043

Please enter the following changes into your manual. The changes are keyed to the Section of the manual in which they occur.

4.6 CASTLING

When two computers are being set up to play each other, disabling their castling is no longer necessary. The computers will castle when ready and will continue playing.

On page 4-4 of the manual, Section 4.6, delete the second line from the bottom: "3) When two computers are to play each other."

5.1 SAVING GAMES

On page 5-2, in the example at the top of the page, insert the letter "I" between the circumflex (^) and the number of the rank on each rank. For example, the eighth rank is: ^I8------B-.

6.1.1 Instructions for Setting the Depth Control

On page 6-1, in the example, insert the letter "D" between the circumflex (^) and 41. The example is: ^D41 RETURN

6.2 HOW THE PROGRAM FINDS ITS MOVE

In the second paragraph from the bottom of page 6-2, change "...preset default value 31 (three/one)..." to "preset default value 3/2 (three/two)..."

On page 6-3, delete the last sentence of the first paragraph.

Delete the second paragraph and Fig. 6-1. Make a note to refer you to the substitute text and diagram which follow.

On page 6-4, delete the first paragraph.

SUBSTITUTE TEXT (for Section 6, pages 6-3 and 6-4):

A negative score is assigned if the position represented by a node favors the player. A positive score is assigned if the position favors the computer. Positions are evaluated on the basis of material and positional advantage. In Fig 6-1, the depth is set to 3/2, or a total of 5 ply when a capture is found on the third ply or when a check occurs. The program evaluates the branches springing from the ply 1 nodes one at a time, starting from the terminal nodes, which are at the lowest ply of the branch. From the terminal nodes, the program works backwards up the tree to the starting position.

If the terminal node of a branch results from a player's move (as occurs on even-numbered plys) the program assumes the player shall have chosen his or her strongest move on the previous ply; therefore the program takes the score of the terminal node having the lowest and backs this score up to the next higher ply, assigning the score to the parent-node which represents the position from which the player would move. If the terminal node is a position resulting from the computer's move (as occurs on odd-numbered plys) naturally the program chooses the highest scoring terminal node and assigns the score to the terminal nodes's parent node which represents the position from which the computer would move.

This tree search method of alternately choosing minimum or maximum scores depending on whose move it is, is called the "minimax" method.

After backing up the score of the terminal nodes to the next higher ply, the program looks horizontally on the new ply and evaluates the other nodes belonging to the same branch, if any. Then it continues backing up the tree, using the minimax method, until the intial node on ply 1 has been evaluated. Then it examines the next branch in the same manner. When each of the ply 1 nodes has been scored, the program is now back to real time. Since it is the computer's move, the program chooses the ply 1 node with the maximum score. In the example presented by Fig. 6-1, the heavy arrows point out a path through the branch which the program has selected for its proposed development.

SECTION 7

On page 7-1, delete step 5).

In step 8), insert a "Z" between "^" and E2E4. The command is:
^ZE2E4

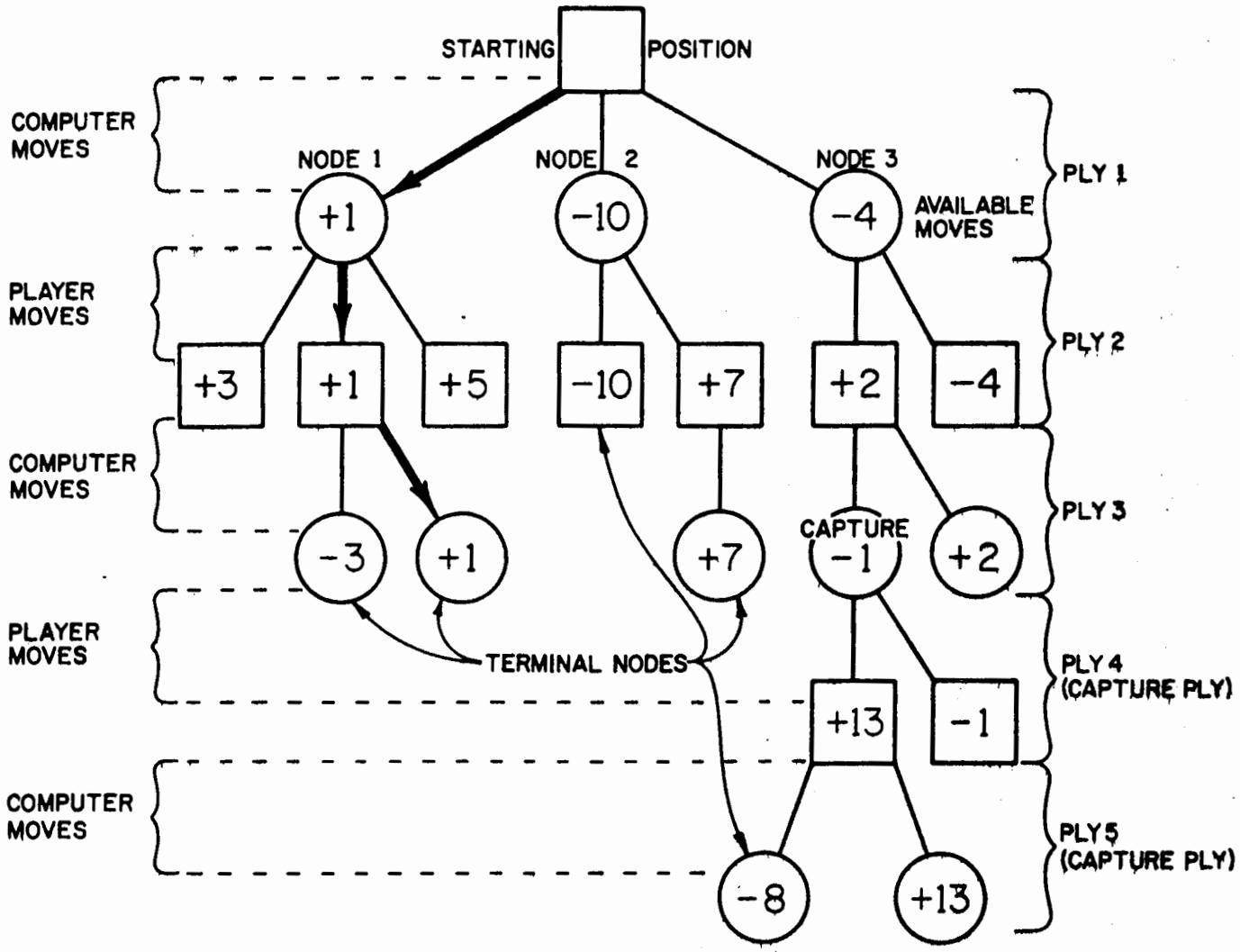


Fig. 6-1. Look-ahead Tree

ProcessorTechnology

Processor Technology Corporation
7100 Johnson Industrial Drive
Pleasanton, CA 94566
(415) 829-2600
Cable Address PROCTEC

8080 CHESS UPDATE 731045

Please enter the following changes into your manual. The changes are keyed to the Section of the manual in which they occur.

5.1 SAVING GAMES

On page 5-1, note the following:

Any number of ranks can be entered in any sequence.

You do not have to complete a rank of entries before pressing RETURN. The I-command changes only those squares to the left of the point where you pressed RETURN. Consequently, it is not necessary to re-enter ranks that are already displayed correctly according to the position you are setting up. This includes squares to the right of the last square that you changed. But, of course, all squares to the left of a changed square must be re-entered even if they are already displayed correctly.

Before you press RETURN to enter a rank, check your entry on the screen to make sure it is correct. If you made an error, delete the entry and start over. If you have entered a rank in error, simply re-enter the rank.

6.1.1 Instructions for Setting the Depth Control

The minimum legal depth setting is 2/1. (Note that the forward slant is not entered.)

Do not enter a zero in the second digit position at any setting. Settings below 2/1 may result in the computer making an illegal move or erroneously announcing a stalemate.

On page 6-1, change the line "n = The depth, a digit in arabic numbers 1 through 9" to read:

"n = The depth, a digit in arabic numbers 2 through 9."

At the bottom of page 6-1, substitute the following for the last two lines:

"Maximum finite setting is 9/9. Maximum indeterminate setting is 9U. Minimum legal setting is 2/1."

6.2.1 Setting the Depth of the Tree Search

Note that the time taken by the computer varies greatly with the positions arrived at. A setting a 4/3 can take anywhere from 2 minutes to a half hour or more per move.

6.3.2 How Many Nodes Did you Consider That Move?

Note the following additional comments:

Displaying the number of terminal nodes analyzed slightly increases the time it takes for the computer to arrive at its move.

There are only 4 hexadecimal digits available to display the number of nodes analyzed, giving a maximum display of FFFF. In higher depth settings, this number may be exceeded in which case the display resets to zero after FFFF and starts over again. To get a reasonably accurate estimate of the nodes analyzed at a giving setting, you can record the number of nodes analyzed for each move and the time taken. Then gradually increase the depth setting until a near maximum number is displayed. Finally, average the number of nodes the computer analyzes in a minute and use the average as a constant times the number of minutes per move.

Note that "hexadecimal" in the first line of 6.3.2 should be spelled with an "a".

The manual first edition, first printing, July 1978 describes 8080 Chess 1.0 (MOD 0). The revision level of the program is shown in the initial display.

ProcessorTechnology

Processor Technology
Corporation

7100 Johnson Industrial Drive
Pleasanton, CA 94566

(415) 829-2600
Cable Address PROCTEC

Cassette PILOT Update 731069

The procedures and suggestions in this notice enable you to correct or avoid a few of the problems that you might have encountered (or will otherwise encounter) in using PILOT, Release 2.2 (MOD 0). It is important that you make all suggested changes, even if they do not seem relevant. The instructions for any future modifications will assume that these earlier modifications have been made.

The first item below describes a "patch" to the PILOT interpreter. By following the instructions once, you can correct PILOT and then save the corrected version of the program on another cassette tape. The original tape will serve as your untouched archive copy of PILOT, and the corrected tape will be your working version. The second item below is a reminder about the range of numbers available in PILOT.

Explanations of what you are doing when you type a command accompany many of the steps involved in "patching" the program. If you do not understand the explanations, or if they simply do not interest you, just type the commands as they appear in the examples, and follow all other explicit instructions: for example, about how to set up the tape recorder. If you decide to read the explanations and are interested in learning more about the SOLOS/CUTER commands, you can find additional material in the SOLOS/CUTER User's Manual.

If you are not accustomed to using cassette recorders, you can consult the appendix entitled "Using Cassettes" in your PILOT manual.

---

1. PATCH THE PILOT INTERPRETER

The PILOT interpreter (the program that will interpret your programs so the computer can understand them) has two errors that you might already have noticed, and that you can correct very easily by following the instructions below. One error is related to the READ statement. If you have tried to read a tape written by the WAPP test program, or if you have used the WRite and Remark Write statements in a program and then tried to read the resulting file, you have been unsuccessful because of this error. The other error in PILOT makes it impossible for anyone who does not have a Sol Terminal Computer keyboard to leave the PILOT editor by typing either the MODE SELECT key or the CTRL and @ keys simultaneously. When you have made the patch, you will be able to use the MODE SELECT (on the Sol keyboard) or the CTRL and @ keys (on another keyboard) to return from the editor to PILOT restart.

You make a patch in a program by getting the program into the memory of the computer (by playing the tape on which it is recorded), and then changing the program while it is in memory. Then you record the program on another tape. THESE INSTRUCTIONS DO NOT ASSUME THAT YOU KNOW VERY MUCH ABOUT COMPUTERS OR PROGRAMMING. You can make the proposed alterations in PILOT without understanding PILOT at all.

In the examples below, the symbol <cr> stands for the RETURN key. Otherwise, unless an exception is noted, command lines should be entered just as they appear here; spaces between items are significant in all cases.

1) Set up a cassette recorder for reading. Put the PILOT program tape into the cassette, rewind it (using the SOLOS/CUTER CAtalog command to power the tape recorder), and position it past the leader. To enter the CAtalog command, type

CA<cr>

after the SOLOS/CUTER prompt (>). When the tape is positioned correctly, hit the MODE SELECT key to return to SOLOS/CUTER command mode.

2) Put the recorder in PLAY mode and type

GET PILOT<cr>

to load the PILOT interpreter into memory without executing it. When the program has been loaded successfully, the screen should look like this:

>GET PILOT PILOT 0000 1FF6

The numbers indicate that PILOT now occupies 1FF6 memory locations beginning at 0000. (1FF6 is a number in base 16.) To "patch," or correct, the program, you will be making changes at two locations within this range.

3) Type

EN 6F0<cr>

By entering this command, you tell the computer that you want to make a change in the contents of memory locations starting at 6F0 (another number in base 16). When a colon (:) appears on the screen, type

0 0 0 0 0/<cr>

right after the colon to put zeros in five consecutive memory locations beginning at 6F0. (Do not worry about why you are inserting zeros.)

4) Type the lines in the list below. The prompt (>) and the colon (:) are not part of the lines that you type; they appear on the screen to let you know that it is time for you to communicate with the computer. This step is like step 3, except that the entries to memory are different. (With the exception of the command EN, all of the items that contain letters are really numbers in base 16.)

```
>EN 1841<cr>
:0 0 0 0/<cr>
>EN 184A<cr>
:E6 7F C2 61 18/<cr>
>EN 1FF4<cr>
:71 B1/<cr>
```

When the prompt (>) returns after this last entry, you know you have "patched" the program, i.e., you have a correct version of PILOT in the memory of the computer. Now you have to SAVE this version on a cassette tape, so that you will not have to go through the patching procedure every time you want to use the program.

5) Set up a cassette recorder for writing. (If you are using the same recorder for both reading and writing, make sure that you have plugged the cables into the appropriate jacks.) Insert a cassette other than the PILOT program tape and rewind it, using the CAtalog command to power the correct tape unit. If you are using tape unit 2 for writing, you will have to specify a unit number in the CAtalog command: type

CA /2<cr>

to power unit 2. Let the tape run in RECORD mode for at least half a minute before you hit the MODE SELECT key to return to command mode.

6) Type

SET XE=100<cr>

This command provides information that must be written on the tape with the program.

7) Type

SAVE PILOT 0000 1FFF<cr>
or
SAVE PILOT/2 0000 1FFF<cr>

(The first version assumes that you are writing to unit 1; the second, that you are writing to unit 2.)

It will take a short time for the program to be SAVEd. When the SAVE operation has been completed, SOLOS/CUTER will again issue the prompt (>).

If you have followed these instructions exactly, you should have a correct version of PILOT recorded on a cassette tape other than the production cassette. We recommend that you make a backup copy of your good version of PILOT on the other side of the same tape. First test the program to make sure you have patched and recorded it properly:

8) Set up a cassette recorder for reading, and load the new version of PILOT as in step 1 and 2. The second of the two numbers that appear on the screen will now be 2000.

9) Execute PILOT by typing

EX 100<cr>

When a table of available commands appears on the screen, you are at the PILOT restart point.

10) Load the WAPP test program from the PILOT restart point by typing LOAD WAPP<cr>, and follow the instructions given in that program to write a file of multiple-choice questions and answers. (The WAPP program is recorded once on each side of the PILOT program tape, after PILOT and PLTST.)

11) Use the READ command from PILOT restart to execute the file written by WAPP. (When you are told to PREPARE TAPE 1, rewind it, position it correctly, and put the recorder in PLAY mode.) If you are able to execute the program, go on to step 12

now. If you get a TAPE READ ERROR, try to read the tape a few more times; if you are still unsuccessful, execute WAPP again, being very careful about your tape recorder settings. If you are able to read the multiple-choice file from the tape, but disturbing things begin to happen when you try to answer the multiple-choice questions, you probably made a typing error in step 3. Start over from step 1.

12) If you are using a Sol keyboard, go to step 13. Otherwise, type

EDIT<cr>

from PILOT restart to enter the editor, and then see whether you can exit by pressing the CTRL and @ keys simultaneously. If you find yourself back at the restart point, your patched version of PILOT is sound; go on to step 13. If you find yourself still in the editor, you made a typing error in step 4 and will have to start over from step 1.

13) Type BYE<cr> from PILOT restart. Load YOUR NEW VERSION of PILOT (as in steps 1 and 2, above), and set up a cassette recorder for writing. Rewind and position your cassette - NOT the production cassette - on its reverse side (again leaving at least the first half minute of the tape unrecorded), and save the corrected version of the PILOT interpreter by issuing the same command that you used in steps 6 and 7.

14) Note on your cassette that you have updated your copy of PILOT from MOD 0 to MOD 0/2. When you execute the program, it will continue to identify itself as MOD 0; it is therefore very important that you make a record of the correct MODification number, so that you will know whether you are ready to make any future patches.

---

2. DO NOT NEGATE -32768; 32768 IS OUT OF RANGE

If you keep in mind that the result of negating a negative number is always the corresponding positive number, and that the range of numbers acceptable in PILOT is -32768 to 32767, inclusive, it should be easy to remember not to enter a statement like

C: A=-(-32768)

or like C: B=-n%3 ,

where n is an expression that evaluates to -32768. If you do enter such a statement, PILOT will simply not negate the negative number or expression: -(-32768) will evaluate to -32768, no error message will appear, and no useful purpose will be served. In all other cases of computations whose elements or results are out of range, there is already an error message.)

Processor Technology
Corporation

7100 Johnson Industrial Drive
Pleasanton, CA 94566

(415) 829-2600
Cable Address - PROCTEC

Level I Business BASIC Description (Demo Version)

This notice describes the differences between Level I Business BASIC 1.0 and Optional Precision BASIC 1.1.

CHAIN statement

General form:

```
CHAIN string expression
      |
      |_string expression for the file name of
        a semi-compiled BASIC program
```

Although the XEQ statement can be used to load and run a BASIC program from within another BASIC program, the values of all variables are lost. The CHAIN statement also loads and runs another BASIC program (or the next segment of a long program) but variable values are not lost, provided they are declared as common using the COM statement in both program segments. (See below.)

Neither XEQ nor CHAIN affect parameters set by SET statements, or by the SYSTEM statements described below, nor do they change the status of files. For example if a FILE statement was used to open a data file, and a READ statement read part way into the file, then a new program segment loaded with a CHAIN statement can take up reading from the same point in the file. However, if a SET XI statement had moved the index block of a Random File into memory, it is moved back out by XEQ or CHAIN, and a new SET XI statement in the next program segment must be used to bring it back into memory. Note that only semi-compiled BASIC program files may be loaded using CHAIN.

Examples:

```
90 CHAIN "NEXTPROG"
```

or

```
50 A$ = "NEXTPROG"
60 CHAIN A$
```

COM Statement

General form:

```
COM var1 {, var2, var3 ...}
     |      |    |   |
     |______|____|___|_names of string or numeric
                       variables or arrays
```

The COM statement defines the list of variables and arrays as common to two or more program segments which are linked with CHAIN statements. The values of variables are preserved when a new program

or program segement is loaded with CHAIN, provided that the variables are listed in COM statements in both program segments. The values of variables which are not named in a COM statement in both program segments are lost forever, even if the original segment loaded again with CHAIN. More than one COM statement may be present in a program; each such statement makes more variables common. The order in which variables are listed is not significant. When string or array names are present in COM, their dimension must be present, as in a DIM statment.

For example, imagine a long program divided into five segments, each segment loading the next with CHAIN. If each program contained the three statements:

```
10 COM A, B, C, X, Y
20 COM Z1(10, 20), Z2(5, 5, 5)
30 COM L$(20), M$(100)
```

then all of the program segments could use the listed variables, as if the five segments were one program. The values of variables and arrays not listed would be lost, however.

The dimension of all arrays and strings must be present, and the dimensions must be the same in each COM statement which lists the array or string. The COM statement in the first program segment may also serve as a DIM statement. Strings and arrays may be assigned dimensions in a COM statement instead of a DIM statement. If COM statements in later segments list variables not previously made common, then they may be dimensioned there. When a simple variable or an array first appears in common, its value or values are set to 0. When a string first appears, it is set to the null string (""), as well as being dimensioned.

Consider two program segments A and B:

Segment A:

```
10 COM A, B, C, Z(1, 5, 5), L1$(100)
20 LET A=1, B=2, C=3, Z(1, 4, 3) = 4, L1$ = "X"
30 CHAIN "B"
```

Segment B

```
10 COM B, C, D, E$(20), L1$(100), Z(1, 5, 5)
20 PRINT B; C; D; E$; L1$; Z(1, 4, 3)
30 END
```

Values PRINTed by B:

```
2 3 0 X 4
```

After segment A CHAINs to segment B, only the values for B, C, Z, and L1$ are preserved. The value for A is lost. The variable D in segment B is PRINTed as 0 since it first appeared in B and was initialized there. Similarly a value for E$ is not PRINTed since it is initialized to the null string.

COM statements must be the first statements in any program segment

which contains them. Only REM statements may precede them. For example:

```
10 REM  Third Program Segment
20 COM A, B, C(10, 10)
30 REM  Commoned strings:
40 COM A$(5), B$(1000)
```

is legal.

```
10 COM A, B, Z$(10)
20 X = 0
30 COM Q(1, 5)
```

is not legal since statement 20 occurs before a COM statement and is not a REM statement.

CM Error

The CM (common) error will occur when an array or string is not given the same dimensions in COM statements which list it, or statements other than REM precede a COM statement.

Exact INPUT Statement

The Controlled INPUT Statement as described in section 5.7 of the manual, and item 13 of the User's Notes will now accept a negative value for the #chars argument. A negative value gives the same result as a positive value, in that as soon as the specified number of characters have been typed, BASIC generates a carriage return and moves on to the next program statment. Normally, if the user types his own carriage return before #chars, that carriage return will conclude the input. If a negative value is used for #chars, then carriage returns will be ignored, so that the user must type exactly #chars. The MODE SELECT key is ignored as well. All other keys, including cursor control keys, LINE FEED, CTRL-X, DELETE, etc., will be accepted and counted towards #chars but do not perform their usual editing functions. This form of the INPUT statement is useful when the user must type input such as ZIP codes in fixed length fields. For example,

```
10 INPUT (-5, 0)"ZIP CODE: ", Z
```

or:

```
10 PRINT
20 PRINT "ZIP CODE: ";
30 B$ = ""
40 FOR I = 1 TO 5
50   INPUT,(-1, 0)"", A$
60   IF A$ = "Q" THEN EXIT  9 : REM   START OVER--"QUIT"
70   B$ = B$ + A$
80 NEXT I
90 Z = VAL(B$)
100 PRINT
```

SYSTEM Statement

General form:

```
SYSTEM n {,arg1, arg2...}
       |    |_____|_____ number and type of allowable
expression               arguments depends on n.
for integer
0 to 11
```

Specific forms:

```
SYSTEM 0, {unit}, bs, nvar
            |      |    |_numeric variable
  disk drive|      |
  unit            |_expression for a block size
```

After execution of this statement, nvar will contain the number of blocks of the specified block size which are free for allocation on the diskette in the specified unit. If unit is not specified, the default unit will be used. Remember to use a decimal value for block size. For example:

```
100 SYSTEM 0, /1, 1216, B
```

```
SYSTEM 1, string {,nvar}
              |      |___numeric variable for starting address
              |      or address of first image block
string expression
for image file name
```

This statement loads the named image file (type I) into memory. The load address, present in all files of type I, is the first location loaded. If nvar is present the file's starting address is placed in it, or if the file does not have a starting address (e.g. data files), the address of the first image block loaded is placed in nvar. Loaded program files can be executed with the CALL statement. Loaded data files can be read with the PEEK function.

```
SYSTEM 2, string {,nvar1, nvar2 {,nvar3}}
             |       |       |        |___starting address
string expression    |       |_number of bytes to write
for file name        |_first address to be written
```

This statement stores the specified locations in memory into a file of the name given in the string expression. If a file of the given name does not exist, it will be created, of type I, using a blocksize of 4C0H. If the file already exists, it may be of any block size, but it must be of type I. The file is written as one image block. If nvar3 is present, its value will be placed in the file as a starting address. If there are no arguments other than the file name, the addresses corresponding to the video display screen (CC00H-CFFFH) will be saved, with no starting address.

SYSTEM 3

After this statement is executed, control characters typed by the user in command mode or in responce to an INPUT statement will not be echoed on output device, although they will be received by the program. Control characters can be PRINTed as usual, and the RETURN, LINE FEED, CTRL-X, and cursor control keys work as usual. However, the HOME CURSOR and CLEAR keys will no longer affect the screen.

SYSTEM 4

This statement cancels the effect of the SYSTEM 3 statement.

SYSTEM 5

Normally the MODE SELECT key or CTRL-@ may be used to abort any running BASIC program. When it is desirable to keep a program running when it may be subject to unintended or incorrect keystrokes, this statement may be used to inhibit the effect of the MODE SELECT key. Inhibition lasts only until a STOP or END statement is executed or the program is stopped by an error, or control otherwise passes to command mode. However, any time a program is run automatically by a PTDOS command of the form BASIC filename <CR>, MODE SELECT is disabled as if a SYSTEM 5 statement had been executed.

SYSTEM 6

This statement cancels the effect of the SYSTEM 5 statement.

SYSTEM 7

After this statement is executed, control characters typed by the user in command mode or in response to an INPUT statement will be ignored. They will not be echoed and they will not be received by the program. The RETURN, LINE FEED, CTRL-X, and cursor control keys are not affected by this command; their editing functions can only be disabled by an Exact INPUT statement.

SYSTEM 8

This statement cancels the effect of the SYSTEM 7 statement.

SYSTEM 9

This statement closes all data files opened in BASIC. Other PTDOS files are not affected.

SYSTEM 10

The SET XI statement moves the index block of a random access data file into memory for rapid access. The SYSTEM 10 statement frees the

memory area used by all such index blocks for reuse by new SET XI statements, without closing the associated file. If a random access file is CLOSEd, and a SET XI statment had put its index block in memory, then the memory used by the index block is freed, as if a SYSTEM 10 statement had been executed. In Extended Disk BASIC 1.0 and 1.1, memory used for such index blocks cannot be reused within a given program unless CLEAR, SCRATCH, or RUN is issued, or the program is edited.

SYSTEM 11

This statement resets a number of system features to default values, as if the following collection of statements were executed:

SET CM=1	The cursor will appear.
SET CP=0	The polarity of the video display will be normal.
SET DS=0,0	The video display speed will be set to its fastest.
	The space bar can be used to stop a program.
	The number keys can be used to control video speed.
SYSTEM 4	Control characters will be echoed.
SYSTEM 6	The ESCAPE key can be used to abort a running program.
SYSTEM 8	Control characters will be received by BASIC.

SYSTEM 12

This statement cancels the effect of the SYSTEM 13 statement, described below.

SYSTEM 13, string constant

This statement substitutes the text in the string constant for the message "DRIVE NOT READY". Whenever an attempt is made to read or write to a disk drive unit which does not have a disk inserted and ready for the operation, the string constant is printed, followed by a carriage return and line feed. If a disk is made ready after the message appears, typing any key but MODE SELECT will cause normal program operation to continue, except for the scrolling of the display caused by the appearance of the message. If MODE SELECT is typed when the message appears, the program is aborted and the message "FS ERROR IN LINE xxxx" will appear unless the ERRSET statment is in effect.

SYST Function

The SYST Function, as described in section 5.8 of the manual, can accept as arguments expressions which equate to 1 or 2 as well as zero. Note that SYST(0) gives the last PTDOS error number--only a new error will change the value.

SYST(1)

The MODE SELECT key or CTRL-@ can be used to abort a running program. This feature can be disabled by the SYSTEM 5 statement. The SYST(1) function returns the value of 1 if the program user typed the MODE

SELECT key while its abort function was disabled by a SYSTEM 5 statement. Once the value of 1 is read, it is cleared. A subsequent SYST(1) will return Ø, unless the user typed MODE SELECT again.

SYST(2)

This function has the USASCII value of the last control character entered from the keyboard (1 through 31), even if control character reception was disabled by the SYSTEM 7 statement. If no control character has been typed since the last SYST(2) function, the value of 128 is returned. Except in an Exact INPUT statement, the control characters RETURN, LINE FEED, CTRL-X, DELETE, and the cursor control keys perform immediate functions, and are not detected by a SYST(2) function reference.

ERRSET Statement

The ERRSET statement (See section 5.8) no longer clears all current FOR/NEXT loops, GOSUBs, and user-defined function calls. as in BASIC 1.Ø and 1.1. Normal program flow can resume after error control routines are executed.

User Defined Functions

Certain acceptable statements containing user-defined functions generate syntax errors in BASIC 1.Ø and 1.1. Level I Business BASIC does not have this bug.

March 21, 1979

For customers who are using the Sol "Subsystem B" or similar boards in an 8080 computer, the following software updates might be useful. The Subsystem B uses different keyboard input and status ports, therefore some software modification is necessary for Sol software. Perhaps not all software will need modification, but Subsystem B users will like the capability to patch their input/output routines with speed and ease. Once the following patches are made to the Bytefinder program, programmers can use the Bytefinder to make similar patches to their other programs.

PATCHES FOR BYTEFINDER:

Address offset	old data (SOLOS systems)	new data (CUTER systems)
5D2	DB FA	DB 00
360	DB FC	DB 03
5DA	DB FC	DB 03

The Bytefinder comes in four versions, addressed at 0000H, 4000H, 7000H, and D000H. The offset addresses given above are to be added to the starting address for the version you are using. For example, the first patch address would be 45D2H if you were using the 4000H version. After the patches are made, the patched version of Bytefinder can be saved using the CUTER save command. Refer to the SOLOS/CUTER manual for this information.

Processor Technology

BYTESAVER MODIFICATION FOR SOL

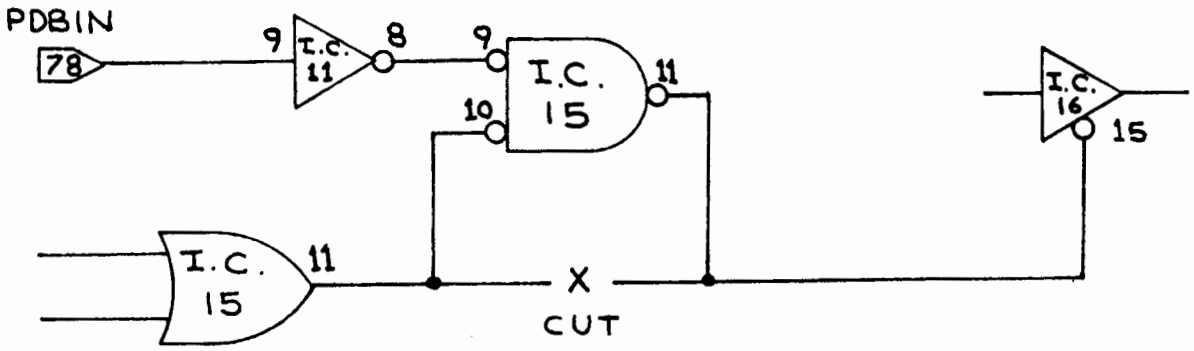
A simple modification of the Chromemco Bytesaver is necessary so that it can be used in the Sol. Check off each step as you complete it.

( ) First cut the trace connecting pin 11 of I.C. 15 (7432) to pin 15 of I.C. 16 (74367).

Next make these connections with small gage insulated wire:

( ) 1. Connect pins 11 and 10 of I.C. 15.
( ) 2. Connect pin 8 of I.C. 15 to pin 15 of I.C. 16.
( ) 3. Connect pin 8 of I.C. 11 to pin 9 of I.C. 15.
( ) 4. Connect S-100 Bus pin 78 (PDBIN) to I.C. 11 pin 9. Pin 78 is the 23rd from the left on the solder (Not Component!) side of the board.

The schematic now looks like this:



This modification gates data onto the bus only when PDBIN is high or active. This is necessary in the Sol where the Data IN and Data Out busses are connected together.

PROCESSOR TECHNOLOGY CORPORATION 6200 HOLLIS STREET EMERYVILLE CA 94608 (415) 652-8080
CABLE ADDRESS "PROCTEC"

Processor Technology

BYTESAVER PROGRAMMING ROUTINE

This short routine will program the contents of any 1K block of memory into a 2708 EPROM installed in socket 1 of the Bytesaver. The Bytesaver should be addressed at 6ØØØH.

A15 – L , A14 – H , A13 – H .

The routine is used as a custom command with the Solos/Cuter operating system. Enter the program at C9ØØH, or if you wish, reassemble it elsewhere. Then create a custom command by typing:

CU BURN C9ØØ (CR)

If the program has been re-assembled at an arbitrary address of NNNN, type:

CU BURN NNNN (CR)

(CR) means strike the return key: don't type out these characters.

Now to use the BURN custom command type:

BURN AAAA (CR)

Where AAAA is the starting address of the 1K block you wish to program into the 2708. The programming operation takes about 5 minutes, which is in accordance with the published programming instructions for the 2708. When the programming is complete the routine will return control to Solos/Cuter and a prompt will reappear on the screen.

PROCESSOR TECHNOLOGY CORPORATION 6200 HOLLIS STREET EMERYVILLE CA 94608 (415) 652-8080
CABLE ADDRESS "PROCTEC"

```
                        0000 **
                        0000 *** BYTESAVER ROUTINE ***
                        0000 **
     C33A               0000 SCONV  EQU   0C33AH  GETS PARAMETERS
                        0000 *
C900 CD 3A C3           0000 BURN   CALL  SCONV  SOURCE ADRS. TO HL
C903 7D                 0000        MOV   A,L GET LO ADRS BYTE
C904 B7                 0000        ORA   A
C905 C2 04 C0           0000        JNZ   0C004H IT MUST BE 0
                        0000 *
C908 22 28 C9           0000        SHLD  SAD    KEEP SOURCE ADRS.
C90B 01 00 00           0000        LXI   B,0    PASS COUNT=0
                        0000 *
C90E 11 00 64           0000 BLOOP  LXI   D,6400H  PROM ADRS
C911 2A 28 C9           0000        LHLD  SAD    SOURCE ADRS.
                        0000 *
C914 7E                 0000 PLOOP  MOV   A,M    GET SOURCE DATA
C915 12                 0000        STAX  D      ZAP THE PROM
C916 23                 0000        INX   H      BUMP SOURCE
C917 13                 0000        INX   D      & PROM ADRS.
C918 7A                 0000        MOV   A,D    CHECK HI ADRS.
C919 FE 68              0000        CPI   68H    PASS COMPLETE ?
C91B C2 14 C9           0000        JNZ   PLOOP  NOT YET
                        0000 *
C91E 03                 0000        INX   B      BUMP PASS COUNT
C91F 78                 0000        MOV   A,B
C920 FE 04              0000        CPI   4      1K PASSES ?
C922 C2 0E C9           0000        JNZ   BLOOP  NOT YET
                        0000 *
C925 C3 04 C0           0000        JMP   0C004H ALL DONE
                        0000 *
                        0000 **  RAM AREA **
                        0000 *
C928                    0000 SAD    DS    2      SOURCE ADRS.
                        0000 *
/
```

PROCESSOR TECHNOLOGY CORPORATION 6200 HOLLIS STREET EMERYVILLE CA 94608 (415) 652-8080
CABLE ADDRESS "PROCTEC"