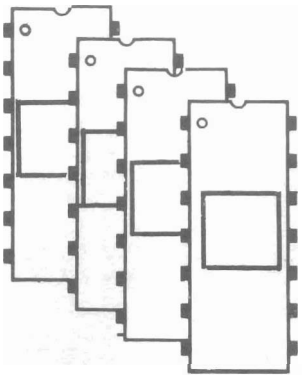
PROM WRITER



User's Manual

Copyright 1981.(C) All rights reserved.

ACKERMAN
DIGITAL SYSTEMS, INC.

Ackerman Digital Systems, Inc.
110 N. York Rd. - Suite 208
Elmhurst, Illinois 60126
(312) 530 - 8992

TABLE OF CONTENTS

IMPORTANT: AT NO TIME SHOULD AN EPROM BE PLACED INTO THE PROMBLASTER WITHOUT FIRST HAVING SPECIFIED THE TYPE TO THE PROMWRITER. FAILURE TO DO SO MAY RESULT IN DAMAGE TO THE EPROM AND/OR PROMBLASTER. ALSO NEVER RESET THE COMPUTER WITH AN EPROM IN THE PROMBLASTER.

ADS, INC. reserves the right to make changes to any products herein to improve reliability, function or design. Although the information in this document has been carefully reviewed and is believed to be accurate, ADS does not assume any liability arising out of the application or use of any product or circuit described herein; neither does it convey any license under its patent or copyrights nor the rights of others.

Copyright (c) 1981 by ads, inc.ALL RIGHTS RESERVED

I. Introduction

The ads PROMWRITER software is a package written in 8080 assembly language for execution under CP/M. It provides the capability of programming 19 different EPROMS, both single or three supply parts, 1K, 2K, 4K, 8K and 16K parts, both 24 and 28 pin packages when used with the ads PROMBLASTER. Commands are provided to read, program and verify EPROMS with optional offsets. Memory load, examine, display and sum functions as well as CP/M hex file load and save operations are provided.

II. Operation

The ads PROMWRITER is a transient program that operates under CP/M. It is invoked by typing its name, PROMxx, on the command line (xx=version). When started the PROMWRITER prints a signon message, then issues a prompt (:). All commands to the PROMWRITER consist of a command character followed by zero to three arguments. Type an H then a carriage return. An abbreviated list of commands should be displayed. In this list, X, Y, and Z represent hexadecimal command arguments, the word TYPESTR represents a seven character string identifying an EPROM type, and the word FNAME represents a CP/M file name.

If your PROMBLASTER is configured for a device address other than C0H-C3H, the first command you must use is the Base command. This configures the PROMWRITER for whatever PROMBLASTER device address you have selected. The next command you should enter is a Type command. This allows the PROMWRITER to setup the PROMBLASTER for the correct standby voltages for whatever type of EPROM you will be programming.

IMPORTANT: AT NO TIME SHOULD AN EPROM BE PLACED INTO THE PROMBLASTER WITHOUT FIRST HAVING SPECIFIED THE TYPE TO THE PROMWRITER. FAILURE TO DO SO MAY RESULT IN DAMAGE TO THE EPROM AND/OR PROMBLASTER. ALSO NEVER RESET THE COMPUTER WITH AN EPROM IN THE PROMBLASTER.

After specifying the EPROM type, any of the other commands may be used. To program an EPROM, place the unprogrammed part in the programming socket U8. MAKE SURE TO USE ONLY THE LOWER 24 PINS FOR 24 PIN EPROMS. Now using the Check command verify that the part is indeed unprogrammed. If the data you wish to program is already in memory, then give the Program command specifying the address range and EPROM offset. If the data is not in memory a Load command may be used to read it off of the disk. To copy an EPROM, place the programmed part into the socket U8 then use the Read command to copy the data to an area of memory. Now place an unprogrammed part into U8, Check it, then Program it. Any errors detected after programming will be printed on the console.

The failure of an EPROM to program may be caused by many problems. An EPROM that is not fully erased will not program ones. The Check command should always be used before programming any part to

verify that it is totally erased. Any pattern of addresses or bits that fail to program is a sign of a bad EPROM. Since the Verify command is automatically performed after a Program command, any attempts to program from an area of memory that is changing will cause apparent programming errors. Specifying the wrong EPROM type to the PROMWRITER can be catastrophic. Damage to the EPROM and/or PROMBLASTER may result. BE CAREFUL !

III. Commands

All commands to the ads PROMWRITER are given in upper or lower case. Any arguments required follow on the same line as the command. A leading space before the first argument is optional. Input is via the CP/M read line function so any of the line editing functions may be utilized before the carriage return. All numeric arguments required are in hexadecimal. The Read, Load and Fill commands will not overwrite the PROMWRITER or CP/M. All numeric arguments are checked against the size of the EPROM being programmed.

IMPORTANT: AT NO TIME SHOULD AN EPROM BE PLACED INTO THE PROMBLASTER WITHOUT FIRST HAVING SPECIFIED THE TYPE TO THE PROMWRITER. FAILURE TO DO SO MAY RESULT IN DAMAGE TO THE EPROM AND/OR PROMBLASTER. ALSO NEVER RESET THE COMPUTER WITH AN EPROM IN THE PROMBLASTER.

a. P - Program eprom

This command is used to program a range of memory into a previously specified type of EPROM. Three arguments are required; a starting memory address, an ending memory address, and an EPROM address offset. Both memory addresses may also be offset with the Offset command. The EPROM is programmed one location at a time starting with the EPROM offset location, and continuing through the memory address range given. For most single supply EPROMS a single 50 millisecond long programming pulse is applied to each location. Therefore the time to program is approximately t = (N x .05) seconds where N is the number of locations being programmed. For an 8K EPROM, t = (8192 x .05) = 409.6 seconds = 6.83 minutes. Some 8K EPROMS require only a 2 millisecond programming pulse with multiple passes. Three supply EPROMS require 100 passes through all addresses with a 1 millisecond programming pulse at each location. For a 2K three supply EPROM, t = (2048 x 100 x .001) = 204.8 seconds = 3.4 minutes. At the end of programming, a control-g (bell) is sent to the console and a Verify command is attempted. Any discrepancies are reported on the console. The command format is:

P x,y,z	Program from x to y offset z
where	x = starting memory address
	y = ending memory address
	z = prom offset

b. Unprogram eeprom

This command is used to unprogram (erase) a previously programmed EEPROM. The erase time varies for the different EEPROMs but it is usually under one second. A verify of the EEPROM's erasure is automatically performed after an unprogram is attempted. Any locations that fail to show erasure are reported. The format of the command is:

U	Unprogram EEPROM

c. Verify eprom

This command verifies the contents of an EPROM against memory. It is called automatically at the end of a Program command. The EPROM locations are compared one a time starting with the offset address and ranging from the starting memory address to the ending memory address. The memory addresses may also be offset via the Offset command. Any discrepancies between the EPROM and the memory contents are displayed on the console. The format of this command is:

V x,y,z	Verify from x to y offset z
where:	x = starting memory address y = ending memory address z = prom offset

d. Read eprom

This command reads the contents of an EPROM into memory. The contents of the EPROM locations are transferred into memory one at a time starting with the offset address and ranging from the starting memory address to the ending memory address. It is important to remember that the memory addresses may also be offset by a previous use of the Offset command. The Read command will not allow the PROMWRITER or CP/M to be overwritten. The command format is:

R x,y,z	Read from x to y offset z
where:	x = starting memory address y = ending memory address z = prom offset

e. Type of eprom

This command determines the type of EPROM being programmed. Any commands which are EPROM dependent; i.e. Program, Verify, Read, and Check will give an UNKNOWN EPROM TYPE error message before an eprom type has been specified. The type is specified as a seven character string. The command format is:

T typestr	Type of eprom

where: typestr = eprom type string

The following table lists the EPROM types that the PROMWRITER version 3.0 will program:

	typestr -	Description	- Size	-# of pins-	Pwr. Supplies
	INT2708	- Intel	2708 - 1K x 8	- 24 pins	- +5v,+12v,-5v
	TMS2708	- Texas Inst	2708 - 1K x 8	- 24 pins	- +5v,+12v,-5v
1K	MCM2708	- Motorola	2708 - 1K x 8	- 24 pins	- +5v,+12v,-5v
	TMS2508	- Texas Inst	2508 - 1K x 8	- 24 pins	- +5v
	INT2716	- Intel	2716 - 2K x 8	- 24 pins	- +5v
	TMS2516	- Texas Inst	2516 - 2K x 8	- 24 pins	- +5v
2K	MCM2716	- Motorola	2716 - 2K x 8	- 24 pins	- +5v
	HN42716	- Hitachi	2716 - 2K x 8	- 24 pins	- +5v
	TMS2716	- Texas Inst	2716 - 2K x 8	- 24 pins	- +5v,+12v,-5v
	INT2732	- Intel	2732 - 4K x 8	- 24 pins	- +5v
	IN2732A	- Intel	2732A - 4K x 8	- 24 pins	- +5v, Vpp=21v
	HN42732	- Hitachi	2732 - 4K x 8	- 24 pins	- +5v
4K	TMS2532	- Texas Inst	2532 - 4K x 8	- 24 pins	- +5v
	MCM2532	- Motorola	2532 - 4K x 8	- 24 pins	- +5v
	HN42532	- Hitachi	2532 - 4K x 8	- 24 pins	- +5v
	INT2764	- Intel	2764 - 8K x 8	- 28 pins	- +5v, Vpp=21v
	HN42764	- Hitachi	2764 - 8K x 8	- 28 pins	- +5v
8K	TMS2564	- Texas Inst	2564 - 8K x 8	- 28 pins	- +5v
	MC68764	- Motorola	68764 - 8K x 8	- 24 pins	- +5v
16K	TMS2528	- Texas Inst	2528 -16K x 8	- 28 pins	- +5v, Vpp=21v
EE	HN48016	- Hitachi	48016 - 2K x 8	- 24 pins	- +5v
PROMS	MCM2816	- Motorola	2816 - 2K x 8	- 24 pins	- +5v

Note that pin compatible parts may be programmed by specifying their equivalent type; i.e. Fujitsu MBM2716 may be programmed by using INT2716 type command.

IMPORTANT: AT NO TIME SHOULD AN EPROM BE PLACED INTO THE PROMBLASTER WITHOUT FIRST HAVING SPECIFIED THE TYPE TO THE PROMWRITER. FAILURE TO DO SO MAY RESULT IN DAMAGE TO THE EPROM AND/OR PROMBLASTER. ALSO NEVER RESET THE COMPUTER WITH AN EPROM IN THE PROMBLASTER.

f. Display memory

This command allows the user to examine a block of memory in hexadecimal. Two arguments are required; a starting memory address, and an ending memory address. Both memory addresses may include

offsets provided by the Offset command. Data is displayed 16 bytes per line in hexadecimal with the ASCII equivalentstrailing the bytes on the same line. The display may be suspended with the CP/M suspend output character (control-S) or stopped with any other character. The command format is:

	D x,y	Display memory from x to y
where:		x = starting memory address
		y = ending memory address

g. Base address

This command allows the user to change the I/O addresses that the PROMWRITER uses when it controls the PROMBLASTER. The default addresses used are C0H-C3H. The command format is:

	B x	Base addresses are now x to x+3
where:		x = starting I/O address in hex

The user may wish to patch the PROMWRITER to have it always default to a different I/O address. Location 103H contains the default PROMBLASTER base address that the PROMWRITER uses. By changing this location with DDT and then saving the patched version with the CP/M SAVE command the user can permanently change the default device addresses that the PROMWRITER uses.

h. Offset command

All memory addresses used by the PROMWRITER may use an optional offset specified by this command. This allows the user to setup a base address of a data area and then use relative addresses for all the commands. This offset should not be confused with the EPROM address offset. The default offset is 0000H. Any offset remains in effect until it is changed. Note that the Load hex file command may also specify an offset which has the same effect as using the Offset command. The command format is:

O	Offset examine
0000	0000 was old offset
O 1234	Offset change, 1234 is user's offset
1234	1234 is new offset

i. Load hex file

The PROMWRITER provides a means of loading INTEL hex format files from disk by name. This allows the output of an assembler to be programmed directly into an EPROM. The file is loaded at the address specified in the hex format with whatever offset is in effect at the time. The PROMWRITER will not allow itself or CP/M to be overwritten.

A new offset may be specified on the command line that will take effect for all subsequent commands. The command format is:

```
L filename,x     Load filename .HEX with offset x

where:           filename is CP/M .HEX file
                 x is optional hexadecimal offset
```

j. Write hex file

The PROMWRITER provides a means of writing INTEL hex format files from memory to disk by name. This allows the user to save EPROM contents on disk. The data is written to the file from the address specified with whatever offset is in effect at the time to the ending address specified in hex format. If the file specified exists the user is given the option of overwriting the old file. The command format is:

```
W filename,x,y   Write hex from x to y to filename.HEX

where:           filename is CP/M .HEX file
                 x is starting address (memory offset)
                 y is ending address (memory offset)
```

k. Query .HEX files

This command allows the user to examine the CP/M file directory on any disk for all .HEX files without having to exit the PROMWRITER. The names are printed 5 per line on the console. The command format is:

```
Q d              Query .HEX files on drive d

where:           d = CP/M drive specifier
```

l. Check for unprogrammed

Since EPROMS are only programmed by the addition of zeroes, not ones, it is necessary to verify that the EPROM is unprogrammed in a given range before an attempt is made to program it. This command checks a range of locations starting at a given offset in the EPROM for the unprogrammed state, and reports any locations that appear to already have been programmed. The command format is:

```
C x,y            Check eprom for unprogrammed

where:           x = number of locations to check
                 y = offset from start of eprom
```

m. Exit to CP/M

This command performs an orderly return to CP/M via the warm-boot entry point. The PROMBLASTER is not affected by this command however it is advisable to remove any EPROMS from the PROMBLASTER first. The command format is:

E	Exit to CP/M

n. Fill memory

This command allows the user to fill a range of memory with a constant. Three arguments are required; the starting memory address, the ending memory address, and the desired fill constant. Both memory addresses may be offset with the Offset command. All locations from the starting address through the ending address will be initialized to the specified constant. The PROMWRITER will not allow itself or CP/M to be overwritten. No attempt is made to verify that the constant stored correctly. Note that if both starting and ending addresses are the same only one location will be initialized. The command format is:

F x,y,z	Fill memory from x to y with z
where	x = starting memory address
	y = ending memory address
	z = hexadecimal value

o. Sum command

This command computes a checksum value for a given range of memory. The checksum is given in two forms; a zero sum which is a value that when added to the sum of the range specified will give a zero result, and a ones sum which when added to added to the sum of the range specified will give an all ones result. The command format is:

S x,y	Sum from x to y memory offset
where:	x = starting memory address
	y = ending memory address

p. Help command

This command provides a brief summary of all PROMWRITER commands on the console as well as a list of all valid EPROM types. The command format is:

H	Help with commands

This results in the following display of the PROMWRITER commands and EPROM types:

```
    P X,Y,Z     - PROGRAM FROM X TO Y PROM OFFSET Z
    U           - UNPROGRAM (ERASE) EEPROM
    V X,Y,Z     - VERIFY FROM X TO Y PROM OFFSET Z
    R X,Y,Z     - READ FROM X TO Y PROM OFFSET Z
    D X,Y       - DISPLAY MEMORY FROM X TO Y MEMORY OFFSET
    B Z         - CHANGE PROMBLASTER BASE PORT #
    O X         - SET AND SHOW MEMORY OFFSET X
    L FNAME,X   - READ A PROM CODE FILE (.HEX) WITH OFFSET X
    W FNAME,X,Y- WRITE FILE (.HEX) FROM X TO Y
    Q           - SHOW DIRECTORY OF PROM FILES (.HEX)
    C X,Z       - CHECK X BYTES IN PROM OFFSET Z FOR UNBURNED
    E           - GO TO CP/M
    F X,Y,Z     - FILL MEMORY FROM X TO Y WITH Z
    S X,Y       - SUM MEMORY FROM X TO Y MEMORY OFFSET
    H           - PRODUCE THIS LIST
    T TYPESTR   - SET EPROM TYPE TO TYPESTR
**************** EPROM TYPES ****************
TMS2508 INT2708 TMS2708 MCM2708 INT2716 MCM2716
HN42716 TMS2516 TMS2716 TMS2532 MCM2532 HN42532
HN42732 INT2732 IN2732A HN42764 INT2764 TMS2564
MC68764 HN48016 MCM2816 TMS2528
```

IV. Error Messages

DEVICE NOT EEPROM - sent in response to a unprogram command. The Unprogram command is invalid for non-EEPROM device types.

UNKNOWN EPROM TYPE - sent in response to a type command. The PROMWRITER doesn't recognize the type string as a valid EPROM type.

SYNTAX ERROR - sent in response to any command requiring arguments. The PROMWRITER requires more arguments than were supplied on the command line.

SIZE ERROR - sent in response to any command that implies an EPROM size. The PROMWRITER computes a value from the starting and ending addresses and the prom offset which must be less than or equal to the size of the part being programmed.

ADDRESS ERROR - sent in response to any command that will load memory. The PROMWRITER computes a value from the starting and ending addresses and the start of CP/M and the end of the PROMWRITER. Overwrites of the PROMWRITER or CP/M are not allowed.

FILE ERROR - sent in response to a hex file load

command. The PROMWRITER detected a checksum error when loading an INTEL hex format file.

FILE NOT FOUND - sent in response to a hex file load command. The PROMWRITER couldn't find a file named on the command line.

DIRECTORY FULL - sent in response to a hex file write command. The PROMWRITER was unable to create the file named on the command line because the directory was full.

DISK FULL - sent in response to a hex file write command. The PROMWRITER was unable to finish writing the hex file named on the command line because the disk is now full.

? - sent in response to an invalid command

OFST PR MM - sent in response to a Verify or Check command. The PROMWRITER found an error between the eprom(PR) and memory(MM) at the eprom offset(OFST). For a Check command the value of MM is FF hex.

x<-INVALID HEX - sent in response to any command requiring hex input. The PROMWRITER found the invalid hex character 'x' in the hex arguments of the command line.

V. Notes on earlier versions

Version 1.0 - First release version

Version 1.1 - Fix erroneous SIZE ERROR on CHECK command with a prior memory offset specified.

Version 1.2 - Fix LOAD command operation

Version 1.3 - Fix LOAD command handling of drive specifiers.

Version 2.0 - Add EEPROM capability. Add UNPROGRAM command. Add capability to interrupt long output messages on PROGRAM, VERIFY, CHECK and DISPLAY commands... Compressed by about 1/2 Kbytes over V1.3.

Version 3.0 - Add lower VPP control for REV 2 PROMBLASTER and newer EPROMS. Improve three supply programming algorithm. Changed MC68764 programming algorithm. Add WRITE hex file command. Improved DISPLAY command output format.

Version 3.1 - Fixed error in WRITE command

Version 3.1X- Allows modified PROMBLASTER to program Z7128's.

VI. PROMWRITER09 differences

The ads PROMWRITER09 is a package written in 6809 assembly language for execution under adsMON. It is contained in a 2K 2716 type EPROM and is position independent. It requires 64 bytes of RAM which it allocates off of the stack pointer when it is called. The PROMWRITER09 is invoked by jumping to the starting address where it is located with the adsMON 'E'xecute command. Because of the limited code space, and the adsMON interface, only a subset of the CP/M version commands are supported. These are listed below:

```
P x,y,z  - Program from x to y prom offset z
V x,y,z  - Verify from x to y prom offset z
R x,y,z  - Read from x to y prom offset z
T typestr- Set EPROM type to typestr
B z      - Change PROMBLASTER base port #
O x      - Set and show memory offset x
C x,z    - Check x bytes prom offset z for unburned
E        - Go to adsMON
S x,y    - Sum memory from x to y memory offset
```

The following are the valid EPROM typestr. Note that pin compatible parts may be programmed by specifying their equivalent type:

```
TMS2508 TMS2708 TMS2716 INT2716 MCM2532
INT2732 TMS2564 INT2764 MC68764
```

Unlike the CP/M version PROMWRITER09 does not check for overwrites. Caution should be exercised when specifying Offsets and Reading EPROMS into memory since it is possible to clobber the RAM variables and/or stack. Use of the on-card memory on the ads SBC09 for EPROM data storage is not recommended for this reason.

Memory display & change functions are provided by adsMON. Toggling between PROMWRITER09 and adsMON is accomplished with the 'E' command. Typing 'E'xit when in PROMWRITER09 causes adsMON to be entered. Return to PROMWRITER09 is possible by typing the adsMON 'E'xecute command with no address specified.

PROMBLASTER MODIFICATION SHEET

1) CALIFORNIA COMPUTER SYSTEMS CPU OWNERS

Recent information brought to our attention by our customers showed that our promblaster is not completly compatable with California Conputer Systems CPU. The modification listed below as supplied by our customer should correct this problem. ADS has not tested this modification on a CCS system and assumes no responsibility or liability for any problems that may occure to the promblaster or your system as a result of making this modification.

Our customer simply suggested connecting a jumper wire from Pin #13 of U21 to Pin #4 of U22 and omit use of the OP2 set of jumpers. Note figure #1 below.

The customer mod above simply takes the Ø clock signal on Pin #24 of the S-100 bus to clock U22, the board select latch. This will eliminate the need for clock option OP2 though it will put two LSTTL loads on Ø.

We at ADS would rather not load down any bus signal with any more than 1 LSTTL gate. With this in mind we recommend the following mod.

Near the S-100 finger cut the trace leading from Pin #25 (pSTVAL*) of the bus. Next install a jumper from Pin #12 of U21 to Pin #3 of U21. Now install the clock option jumper (A) on OP2. This mod will invert Ø of the clock twice bringing it back to its origional form without putting more than one LSTTL load on the bus. See figure 2 below.

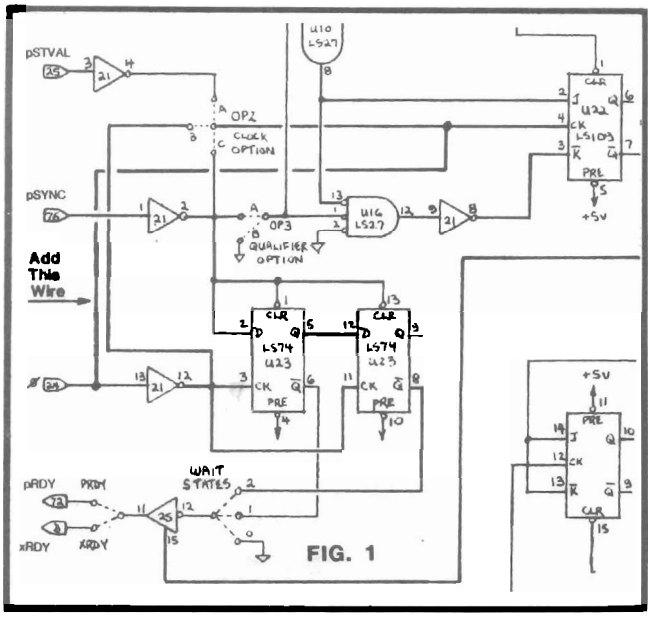


FIG. 1

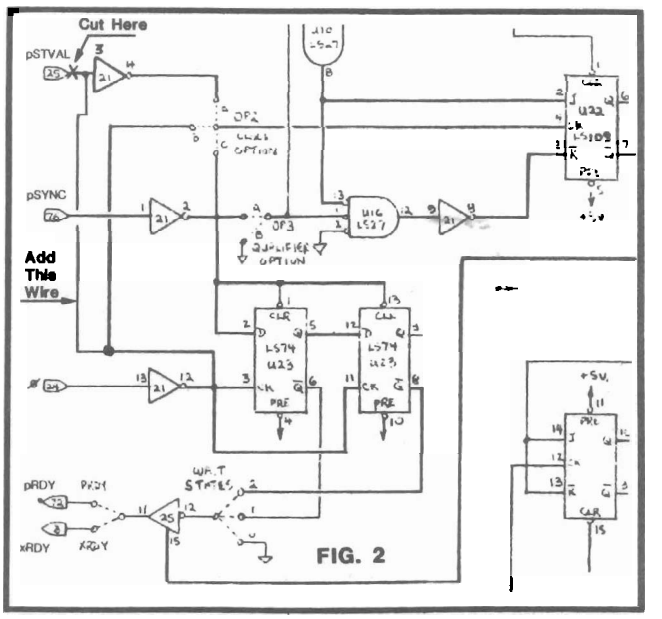


FIG. 2

**** ATTENTION ads PROMBLASTER OWNERS ****

The following modification to a revision 2 PROMBLASTER when used with version 3.1% PROMWRITER software for CP/M will allow the programming of Intel 27128 EPROMS. This capability is obtained at the expense of programming three-supply parts. Performing this modification on a PROMBLASTER voids any warranty implied or explicit. This same modification will allow the programming of Intel 27256's with a new version of the PROMWRITER when programming information and samples become available.

The modification is necessary to allow correct control of pin 26 on the programming socket U8. Until the advent of the 27128 pin 26 has either been +5v Vcc for all parts or +12v Vpe for three supply 2K parts. The 27128 requires pin 26 to be A13 and as such it must be controllable as +5v and 0v. Since no spare I/O lines are available on the PROMBLASTER some feature must be sacrificed to provide this control. The three-supply programming capability was chosen as the least-used, most easily sacrificed feature.

The least significant mode bit, I/O address + 2 bit 3 will be disconnected from the three-supply control circuitry on the board and connected to the pin 26 level shifter. Since other EPROMS use pin 26 as Vcc an active high current pull-up to +5v is required. The level shifter is altered to provide +5v instead of +12v. The passive supply of +5v to pin 26 is removed and a pull-down resistor is used to provide the 0v level when the level shifter is inactive. This allows I/O address + 2 bit 3 to provide Vcc or A13 to the programming socket U8 pin 26. The WRITE FUNCTIONS for I/O address +02 & +03 as shown on page 7 of the PROMBLASTER User's Manual will now behave as follows:

I/O addr +02 & +03 WRITE FUNCTIONS

I/O addr	B7	B6	B5	B4	B3	B2	B1	B0	P23	P22	P21	P1	
+02	data out dsbl	1	+5v P27 0v	+5v P2 0v	+5v P26 0v	+5v P21 0v	A9	A8					M O D E
+03	addr out enbl	+5v P23 0v	+5v P22 0v	+5v P20 0v	+25v P23 B6	+25v P22 B5	+25v P20 B4	1 0				+25v +5v	3
+02	data out dsbl	0	+5v P27 0v	+5v P2 0v	+5v P26 0v	+5v P21 0v	A9	A8					M O D E
+03	addr out enbl	+5v P23 0v	+5v P22 0v	+5v P20 0v	+21v P23 B6	+21v P22 B5	+21v P20 B4	1 0				+21v +5v	1

ads PROMBLASTER 27128 Modification

**** PROMBLASTER 27128 MODIFICATIONS ****

(1) REMOVE DIODE CR5.

(2) ON THE SOLDER SIDE OF THE PROMBLASTER, CUT THE TRACE FROM U6-9 AT THE PIN.

(3) ON THE SOLDER SIDE OF THE PROMBLASTER, CUT THE TRACE FROM U14-13 AT THE PIN.

(4) WITH AN INSULATED JUMPER WIRE CONNECT U6-9 TO U14-13.

(5) UNSOLDER THE EMITTER OF Q8 FROM THE PROMBLASTER.

(6) UNSOLDER THE LEAD OF R16 CLOSEST TO THE TOP OF THE PROMBLASTER.

(7) WITH AN INSULATED JUMPER WIRE CONNECT THE UNSOLDERED LEADS OF STEPS (5) AND (6) TO THE ANODE PAD OF DIODE CR5.

(8) ON THE SOLDER SIDE OF THE PROMBLASTER, INSTALL A 1K 1/4 WATT 5% RESISTOR FROM U8-26 TO U8-14.

NOTE: The modified PROMBLASTER should only be used with version 3.1X of the PROMWRITER software.

***** ATTENTION PROMWRITER V3.1 OWNERS *****

The following is a patch to the ads PROMWRITER version 3.1. When using the Query, Load or Write commands with .HEX files that are larger than 6K bytes (16K characters) the PROMWRITER fails to zero the reel number in the FCB. This can cause .HEX files created to not appear in the directory or to incorrectly Load after the first Query, Load or Write command. Also the Load command with no offset specified does not default to a zero offset for the load. The following patches correct the problems:

```
                ;*************************************
                ; ADS PROMWRITER V3.1 TO V3.2 PATCH  *
                ;*************************************
                ; 2-15-83 - THIS PATCH CORRECTS
                ; ERRORS IN THE (Q)UERY, (L)OAD AND
                ; (W)RITE .HEX FILES COMMANDS.
                 ---------------------------------------

 005C =         FCB     EQU     05CH    ;FILE CONTROL BLOCK
 005C =         FCBDN   EQU     FCB+0   ;*DRIVE
 0068 =         FCBRL   EQU     FCB+12  ;*CURRENT REEL
 0120 =         TXTPNT  EQU     120H    ;CMD LINE POINTER
                *---------------------------------------

 063A                   ORG     063AH
 063A CDC219            CALL    PATCH   ;CLR DRIVE & REEL
 063D 00                NOP             ;*FOR SEARCH
                *---------------------------------------

 06E1                   ORG     06E1H
 06E1 CDC219            CALL    PATCH   ;CLR DRIVE & REEL
 06E4 00                NOP             ;*FOR OPEN
                *---------------------------------------

 073F                   ORG     073FH
 073F C2CA19            JNZ     PATCH2  ;CHK4 NO OFFSET
 074B                   ORG     074BH
 074B =         SCINIT  EQU     $       ;RETURN FROM CHK
                *---------------------------------------

 0B87                   ORG     0B87H
 0B87 332E32            DB      '3.2'   ;PATCH REV NUMBER
                *---------------------------------------

 19C2                   ORG     19C2H
 19C2 AF        PATCH:  XRA     A       ;ZERO DRIVE & REEL
 19C3 325C00            STA     FCBDN   ;*
 19C6 326800            STA     FCBRL   ;*
 19C9 C9                RET
 19CA 2A2001    PATCH2: LHLD    TXTPNT  ;0-LAST CHAR = CR?
  9CD 2B                DCX     H       ;*
 19CE 7E                MOV     A,M     ;*
 19CF FE0D              CPI     0DH     ;*
 19D1 C24B07            JNZ     SCINIT  ;*NO-USE AS IS
 19D4 222001            SHLD    TXTPNT  ;*YES-PNT2 IT
 19D7 C34B07            JMP     SCINIT  ;**
 19DA                   END
```

To install the patch, edit and assemble the above code on your system. Then using DDT on the PROM31.COM file:

```
A>DDT PROM31.COM
NEXT  PC
1B00 0100
-IV32PTCH.HEX    <-This is the output from your patch assembly
-K
NEXT  PC
1B00 0100
-G0              <-Return to CP/M and save the patched version
A>SAVE 26 PROM32.COM
```

This upgrades a version 3.1 PROMWRITER to a version 3.2

eckerman
digital systems inc.

110 north york road, suite 208 elmhurst, illinois 60126

(312) 530-8992

Dear Promwriter Owner:

Recently we found a bug in the Promwriter Version 3.1 Software regarding use of the Q, L and W command on files larger than 6K Bytes.

The attached patch will correct this problem, or if you prefer ADS will update your current Promwriter. Just return you disk with a check for $5.00 to cover postage and handling and ADS will update your Promwriter for you.

We apologize for any inconvenience this may have caused.

Thank you.

Cordially,

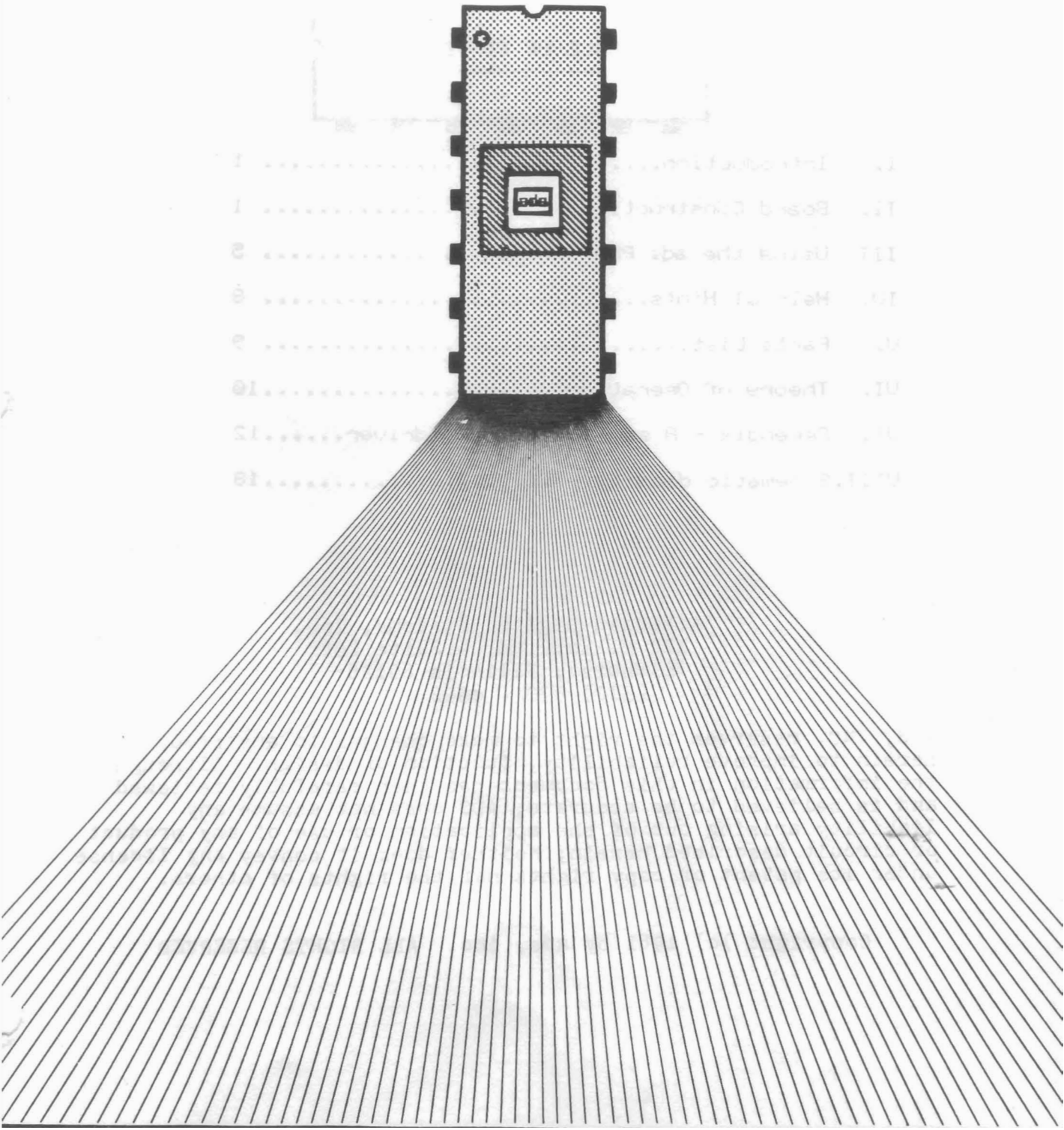
Dave Swoch

Dave Swoch
Vice Pres.
Operations

DS:mhp
encl.

PROMBLASTER

HARDWARE MANUAL



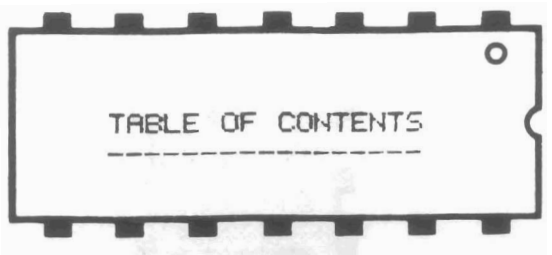


TABLE OF CONTENTS

Ackerman Digital Systems, Inc.
110 N. York Rd. - Suite 208
Elmhurst, Illinois 60126
(312) 530 - 8992

ADS, INC. reserves the right to make changes to any products herein to improve reliability, function or design. Although the information in this document has been carefully reviewed and is believed to be accurate, ADS does not assume any liability arising out of the application or use of any product or circuit described herein; neither does it convey any license under its patent or copy rights nor the rights of others.

Copyright)c) 1981 by ads, inc. ALL RIGHTS RESERVED

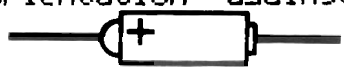
I. Introduction

The ads PROMBLASTER is an S-100 compatible EPROM programming board. It appears to the computer as four I/O ports. The address, data and the amplitude and timing of the various programming pulses for each different EPROM size and family are controlled by software. Either 1k, 2k, 4k, 8K or 16k single or three supply, 24 or 28 pin EPROMS may be programmed. The PROMBLASTER has an on-card switching regulator to provide the high voltage for programming EPROMS. A one millisecond timing reference is also provided for controlling programming pulse widths. The ads PROMWRITER software provides full feature control of the PROMBLASTER and is available under CP/M.

II. Board Construction

1. Begin construction of the ads PROMBLASTER by first examining it for obvious shorts. If an ohmmeter is available measure between address lines, data lines, and the +5 volt and ground for shorts.

2. Noting their orientation against the silk screen, install and solder the I/C sockets and BERG mini-Jump pins. No socket should be used for the dip switches S1 and S2.

3. Carefully observing the polarized capacitors orientation against the silk screen, install and solder the capacitors. 

4. Install the diodes next, matching their polarity with that of the silk screen. 

5. The dip switches S1 and S2 and resistors should be installed next.

6. Now install the regulators and their heat sinks. Heat sink compound should be used sparingly. The regulators fasten to the heat sinks and board with 6-32 x 3/8 screws and nuts.

7. Install the inductor L1.

8. Install the NPN and PNP transistors being careful to follow the emitter/collector/base pattern on the screen with the transistors you use. Note that the component designation appears next to the emitter pad.

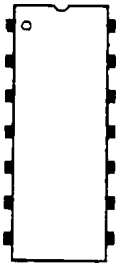
9. Apply power to the board and using Q13's heatsink for a ground verify that the output of the regulators match the values shown:

I.C.	Voltage	Test Point
Q13	+5vdc	5 TP
Q14	+5.7vdc	5.7 TP
Q15	+12.7vdc	12.7 TP
Q16	-5vdc	-5 TP

Remove power and carefully insert U26 the DC-DC converter then

re-apply power and verify that +26vdc is available across C11 at the HV TP.

10. Verify that +5vdc is present at the correct pins on the following I/C's:



I/C	Vcc	Gnd	Device
U1	20	10	74LS244
U2	20	10	74LS244
U3	20	10	74LS374
U4	20	10	74LS374
U6	20	10	74LS374
U7	20	10	74LS273
U8	28	14	SOCKET
U9	16	8	74LS155
U10	14	7	74LS27
U11	20	10	74LS682
U12	20	10	74LS682
U13	14	7	7406
U14	14	7	7406
U15	14	7	74LS02
U16	14	7	74LS27
U17	14	7	74LS04
U18	14	7	74LS02
U19	14	7	74LS08
U20	16	8	MC14020
U21	14	7	74LS04
U22	16	8	74LS109
U23	14	7	74LS74
U25	16	8	74LS368
U26	14	5	TL497

11. Verify the following voltages on the pins of the programming socket U8 (NOTE: A HIGH IMPEDANCE METER OR SCOPE MAY SHOW A HIGHER VOLTAGE (+5.7vdc)):

U8 Pin	voltage
1	+5vdc
20	+5vdc
21	+5vdc
22	+5vdc
23	+5vdc
26	+5vdc

Now using a jumper wire, temporarily apply a ground to the following points one at a time, and verify the U8 pin voltages:

Ground	- U8 Pin	- voltage
U13-6	- 1	- +26vdc
U13-4	- 20	- +26vdc
U13-8	- 21	- +12vdc
U13-2	- 22	- +26vdc
U14-2	- 22	- +12vdc
U13-12	- 23	- +26vdc
U14-6	- 23	- -5vdc
U14-12	- 26	- +12vdc

Again using a jumper wire, temporarily apply a ground to U14-10 and verify that the voltage from U26 measured across C11 at the HV TP lowers to +22 volts.

Again using a jumper wire, temporarily apply +5vdc to the following points one at a time, and verify the U8 pin voltages:

+5vdc	- U8 Pin	- voltage
U15-1	- 20	- 0vdc
U15-10	- 21	- 0vdc
U16-8	- 22	- 0vdc
U16-6	- 23	- 0vdc

12. Remove power and install the I/C's (do not bend over any pins and/or reverse the I/C's in their sockets).

13. The ads PROMBLASTER is designed to work in a 1 MHZ system. For use in faster systems the onboard wait state generator must be used. If you require zero, one, or two wait states for 1, 2, or 4 MHZ systems install the necessary jumper (WAIT STATES - 0,1,2) and select which of the two S-100 ready lines your system requires with jumper pRDY or xRDY.

14. The ads PROMBLASTER may be used with standard or extended device addresses. For use in extended device address systems, use jumper (DP1 - A). For use in standard device address systems, use jumper (DP1 - B).

15. The ads PROMBLASTER may be used with I.E.E.E. 696 or pre-standard CPU's. It is important to understand the differences in order to correctly configure the board. Most of the differences in timing occur in the operation of the pSTVAL* signal (pin 25). Many CPU's provide a clock signal, PHI 1, in place of pSTVAL* on the bus. This is acceptable as long as there is only one negative edge during the pSYNC interval that occurs after the address and status lines are valid. This is shown on the next page:

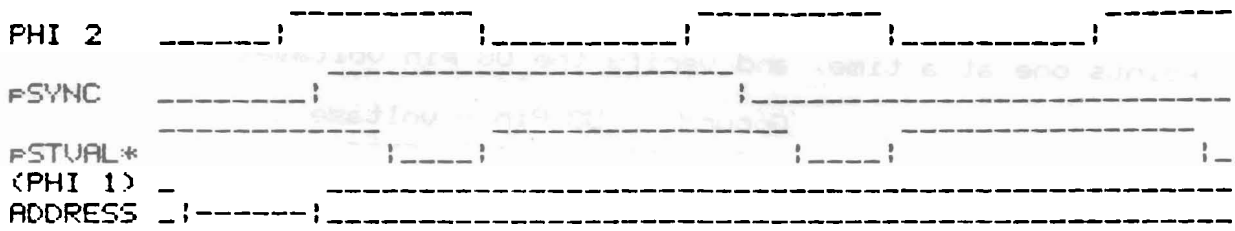


Figure 1. I.E.E.E 696 TIMING

Many PHI 1 signals don't meet this criterion. In the sample timing shown below there is a negative edge on PHI 1 during pSYNC when the address and status lines are not valid. This can cause erroneous device selects and improper operation. For those CPU's that don't provide a correct pSTVAL* or PHI 1 signal a jumper has been provided to allow the use of PHI 2. As shown below, clocking the device select on the negative edge of PHI 2 during pSYNC provides correct timing:

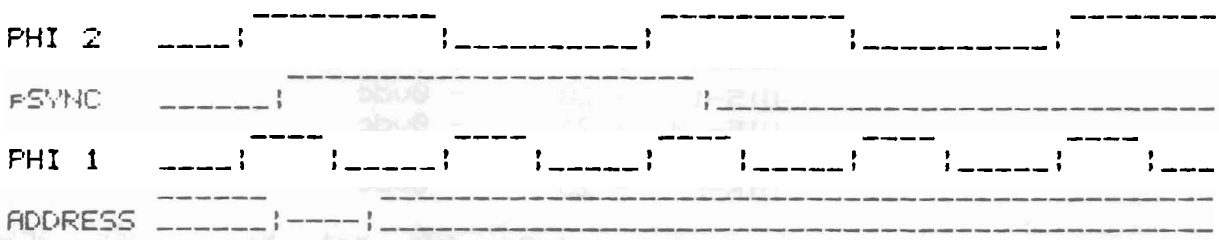


Figure 2. PHI 2 TIMING

There are also problems in the use of the PHI 2 signal. Some CPU's don't provide valid address and status prior to the negative edge of PHI 2 during pSYNC. For these cases a jumper is provided to allow the negative edge of pSYNC to clock the PROMBLASTER. In all of the above cases the pSYNC signal was used to qualify the device select clock signal. However when pSYNC is used as the clock this qualification must be defeated. An example of this timing is shown below:

Figure 3. pSYNC TIMING

The desired clock signal is selected with OP2. The OP2-A jumper should be used when your CPU provides a correct pSTVAL* or PHI 1 signal that operates as per figure 1 above. The OP2-B jumper should be used when your CPU does not provide a pSTVAL* or a compatible PHI 1 signal, but has a PHI 2 signal that occurs during pSYNC after the address and status lines are valid. The OP2-C jumper should be used with those systems where the address and status lines are not valid during PHI 1 or PHI 2 but go valid prior to the end of pSYNC.

The desired clock qualifier signal is selected with OP3. The OP3-A jumper should be used with OP2-A or OP2-B to allow clocking only during the pSYNC interval. The OP3-B jumper should be used with the OP2-C clock to always qualify the pSYNC clock signal.

The ads PROMBLASTER provides an optional ground jumper for pin 53 on the S-100 bus. On some pre-standard CPU's pin 53 is the Sense Switch Disable line (SSW-DSBL). The I.E.E.E. 696 standard eliminates SSW-DSBL and defines pin 53 as an extra ground line. The OPT GND jumper on the PROMBLASTER allows pin 53 to be a ground when connected or unaffected when disconnected.

16. The PROMBLASTER is now ready for use within your system. Select the group of four I/O addresses you want the board to respond to with switch S1. If you are using the extended device address option, you must also set switch S2 to the desired device page address. An open switch corresponds to a '1'. S1-6 is the most significant bit for the group of four I/O addresses, and S2-8 is the most significant bit for the extended device address. S1-7 and S1-8 are not used.

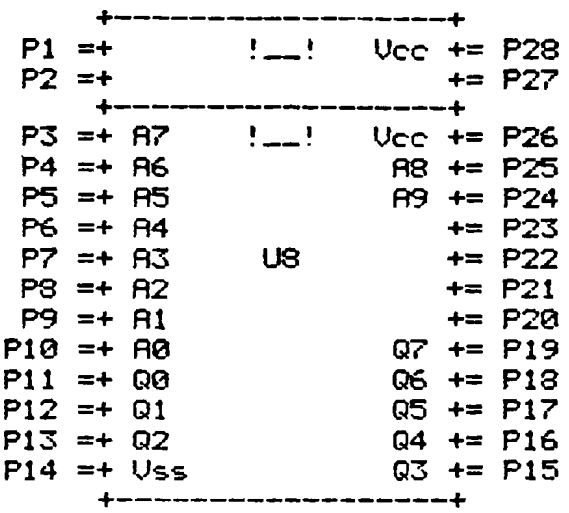
17. Install the board and verify that your computer and other I/O devices function normally.

III. Using the ads PROMBLASTER

The ads PROMBLASTER is controlled through four I/O ports. These are:

I/O Address	Read Function	Write Function
+ 00	Prom data in	Prom data out
+ 01	Reset hi volt	Prom A0-A7
+ 02	Timer status	Prom A8-A9, mode
+ 03	Reset timer	Prom hi volt control

In the following tables and descriptions the numbers P1-P28 refer to the pins on the device programming socket U8. Most EPROMs in a 24 or 28 pin package are functionally equivalent on many of their pins as detailed on the following page:



I/O address + 00 provides an eight bit data path to or from the programming socket U8. Data written to I/O address + 00 is latched and is presented to the EPROM data outputs when enabled. Reading from I/O address + 00 causes the data present on the EPROM data outputs to be input to the CPU.

I/O addr	B7	B6	B5	B4	B3	B2	B1	B0	
+00	P19	P18	P17	P16	P15	P13	P12	P11	READ/WRITE

I/O address + 01 provides a combined control and address function. Data written to I/O address + 01 is latched and is presented to the EPROM address lines A0 - A7 when enabled. Reading from I/O address + 01 causes the latch at I/O address + 03 to be cleared.

I/O addr	B7	B6	B5	B4	B3	B2	B1	B0	
+01	P3	P4	P5	P6	P7	P8	P9	P10	WRITE
+01	RESET I/O addr +03 latch contents								READ

I/O addresses + 02 and + 03 are combined timing, address and voltage control ports. Data written to I/O address + 02 enables/disables the data to the EPROM, controls the levels at three pins of U8 programming socket and supplies A8 - A9 as well. Data written to I/O address + 03 enables/disables the A0-A7 lines to the EPROM, and controls the levels at five pins of U8. Reading from I/O address + 02 causes the status of the one millisecond reference to be returned to the CPU on bit 7. Reading from I/O address + 03 causes the one millisecond timing reference to be restarted. To minimize the number of control bits needed, the voltages on the pins of U8 have been encoded such that two bits of I/O address + 02 set the mode of operation for many of the remaining bits. This is detailed in the tables on the following page:

I/O addr +02 & +03 WRITE FUNCTIONS

I/O addr	B7	B6	B5	B4	B3	B2	B1	B0	P26	P23	P22	P21	P1	
+02	data out dsbl	1	+5v P27 0v	+5v P2 0v	1	+5v P21 0v	A9	A8						MODE 3
+03	addr out enbl	+5v P23 0v	+5v P22 0v	+5v P20 0v	+25v P23 B6	+25v P22 B5	+25v P20 B4	1 0	+5v +5v				+25v +5v	
+02	data out dsbl	1	+5v P27 0v	+5v P2 0v	0	na	A9	A8						MODE 2
+03	addr out enbl	na	+5v P22 0v	+5v P20 0v	+25v P23 -5v	+25v P22 B5	+25v P20 B4	1 0	+12v +5v	-5v -5v		+12v +12v	+25v +5v	
+02	data out dsbl	0	+5v P27 0v	+5v P2 0v	1	+5v P21 0v	A9	A8						MODE 1
+03	addr out enbl	+5v P23 0v	+5v P22 0v	+5v P20 0v	+21v P23 B6	+21v P22 B5	+21v P20 B4	1 0	+5v +5v				+21v +5v	
+02	data out dsbl	0	+5v P27 0v	+5v P2 0v	0	na	A9	A8						MODE 0
+03	addr out enbl	na	+5v P22 0v	+5v P20 0v	+25v P23 -5v	+25v P22 B5	+25v P20 B4	1 0	+5v +5v	-5v -5v	+12v B5	+12v +12v	+25v +5v	

I/O addr +02 & +03 READ FUNCTIONS

I/O addr	B7	B6	B5	B4	B3	B2	B1	B0
+02	1 msec timer status							
+03	Reset 1 millisecond timer							

Before an EPROM can be put into U8 the socket must be setup with the correct voltage configuration. the data outputs must be disabled.

the address inputs should be disabled, and the chip select pin for the part should be made inactive. This is accomplished be loading the correct bit patterns into I/O addresses + 02 and +03. After placing the correct EPROM into U8, it may be read by latching the desired address into A0 - A7 via I/O address +01, setting up A8 - A?? via I/O addresses +02 and +03 and enabling the address to the EPROM. This is followed by making the desired chip select and/or output enable line active, again via I/O addresses +02 and +03, and then reading the data from the EPROM via I/O address +00.

Programming an EPROM involves more steps. After placing the EPROM into a properly configured U8 socket, the desired address is setup as for reading above. The EPROM must now be placed into programming mode. For single supply parts this may be as simple as raising the Vpp pin to +25v. Three supply parts require an additional pin be brought to +12v. The data to be programmed is latched into I/O address +00 and is enabled to the U8 socket. A programming pulse varying from one to fify milliseconds is then supplied to the EPROM. After this pulse, the data is disabled, a new address is supplied, and the process repeats. Single supply EPROMS may be programmed at any location randomly. Three supply parts require a continuous looping through all locations with null data supplied to the unaffected locations. A software example for Intel 2716 +5v EPROMS may be found in the Appendix.

IV. Helpful Hints

After writing PROMBLASTER software, check out its operation with a scope or multimeter first. Observe the relationship, amplitude and timing of the signals present on the programming socket U8 before attempting to read or program your part. It is very easy to toggle the incorrect bit causing the PROMBLASTER to apply the wrong voltage to the wrong pin. This may result in damage to the EPROM and/or PROMBLASTER. Mode 3 is used for all single supply EPROMS. Mode 2 is provided for three supply 2K parts. NOTE: THE EXTRA SUPPLY VOLTAGES WILL ONLY BE PRESENT AT U8 WHEN THE ADDR OUT ENBL BIT 7 OF THE CONTROL REGISTER IS SET IN MODES 2 & 0. THIS SHOULD BE DONE BEFORE A THREE SUPPLY PART IS INSERTED IN U8.Mode 1 is used for those single supply EPROMs that require the new lower programming voltage of +21 volts. NOTE: THE HV WILL LOWER TO +21V ONLY WHEN THE ADDR OUT ENBL BIT 7 OF THE CONTROL REGISTER IS SET IN MODE 1. A ONE SECOND DELAY SHOULD BE USED BEFORE ANY PROGRAMMING IS ATTEMPTED. Mode 0 is for three supply 1K EPROMS. Note that through proper control it should be possible to read 24 and 28 pin ROMS as well, provided that you know the mask-programmed chip select levels.

The ads PROMWRITER software is a package written in 8080 assembly language for execution under CP/M. It provides the capability of programming 19 different EPROMS, both single or three supply parts, 1K, 2K, 4K, 8K and 16K parts, both 24 and 28 pin packages when used with the ads PROMBLASTER. Some EEPROMS may be programmed as well. Commands are provided to read, program and verify EPROMS with optional offsets. Memory load, examine, display and sum functions as well as CP/M hex file load and store operations are also provided.

V. Parts List

Quantity	Indetifiers	Part	Description
	- INTEGRATED CIRCUITS	-	
2	- U15,U18	- 74LS02	- QUAD 2 IN NOR
2	- U17,U21	- 74LS04	- HEX INVERTER
2	- U13>U14	- 7406	- HEX INVERTER OC HV
1	- U19	- 74LS08	- QUAD 2 IN AND
2	- U10,U16	- 74LS27	- TRIPLE 3 IN NOR
1	- U22	- 74LS109	- DUAL J-K* FLIP FLOP
1	- U23	- 74LS74	- DUAL D FLIP FLOP
1	- U9	- 74LS155	- DUAL 2 TO 4 DECODER
2	- U1>U2	- 74LS244	- OCTAL BUS BUFFER
1	- U7	- 74LS273	- OCTAL D FLIP FLOP W/CLR
1	- U25	- 74LS368	- HEX BUS INVERTER W/3 S
3	- U3,U4,U6	- 74LS374	- OCTAL D FLIP FLOP W/3 S
2	- U11>U12	- 74LS682	- OCTAL COMPARATOR W/PULLS
1	- U20	-MC14020B	- 14 STAGE BINARY COUNTER
1	- U26	- TL497	- SWITCHING REGULATOR
2	- Q13-Q14	-LM340T-5	- 5V TO-220 REGULATOR
	-	-or MC7805-	" "
1	- Q15	-LM340T-12-	12V TO-220 REGULATOR
	-	-or MC7812-	" "
1	- Q16	-LM320T-5	- -5V TO-220 REGULATOR
	-	-or MC7905-	" "
	- TRANSISTORS & DIODES		
5	- Q9>Q13	- 2N2222	- NPN GP TRANSISTOR
9	- Q1>Q8,Q14	- 2N4403	- PNP GP TRANSISTOR
15	- CR1>CR15	- 1N4001	- 1A 50 PIV DIODE
	- RESISTORS	-	
1	- R69	- 2.7 OHM	- 1 WATT 5% RESISTOR
2	- R66>R67	- 150 OHM	- 1/4 WATT 5% RESISTOR
1	- R63	- 150 OHM	- 1 WATT 5% RESISTOR
1	- R20	- 750 OHM	- 1/4 WATT 5% RESISTOR
3	- R71,R77>R78	-1.2K OHM	- 1/4 WATT 5% RESISTOR
13	- R2,R4,R6	-2.2K OHM	- 1/4 WATT 5% RESISTOR
	-R8>R10,R12>R16-	-	" "
	- R64>R65	-	- " "
5	- R40,R50>R52	-3.0K OHM	- 1/4 WATT 5% RESISTOR
	- R62	-	- " "
4	- R1,R3,R5,R7	-3.3K OHM	- 1/4 WATT 5% RESISTOR
4	- R17>R19,R73	-5.1K OHM	- 1/4 WATT 5% RESISTOR
3	- R11,R68,R74	- 10K OHM	- 1/4 WATT 5% RESISTOR
1	- R70	-24.9KOHM	- 1/4 WATT 1% RESISTOR
2	- R75>R76	-120K OHM	- 1/4 WATT 5% RESISTOR

< continued on next page >

	- CAPACITORS & COILS		
1	C17	10 PFD	CERAMIC DISC CAPACITOR
1	C13	330 PFD	CERAMIC DISC CAPACITOR
1	C16	.002 UFD	CERAMIC DISC CAPACITOR
24	C3,C4,C7,C10	.01 UFD	CERAMIC DISC CAPACITOR
	C12,C14		" "
	C20>C37		" "
5	C1>C2,C8>C9	4.7 UFD	25V TANTALUM CAPACITOR
	C15		" "
5	C5>C6,C11	10 UFD	35V ALUMINUM CAPACITOR
	C18>C19		" "
1	L1	200 UH	0.2A INDUCTOR
	- MISCELLANEOUS		
2	S1,S2		8 POSITION DIP SWITCH
2			THM6106 TO-220 HEATSINK
2			THM6073 TO-220 HEATSINK
27			BERG MINI-JUMP PINS
7			BERG MINI-JUMPS
8	U1>U2,U3>U4		20 PIN I.C. SOCKET
	U6>U7,U11>U12		" "
4	U9,U20,U22		16 PIN I.C. SOCKET
	U25		" "
11	U10,U13>U16		14 PIN I.C. SOCKET
	U17>U19,U21		" "
	U23,U26		" "
1	U8		28 PIN L.I.F./Z.I.F. SOCKET

VI. Theory of Operation

The ads PROMBLASTER requires +8vdc and +/- 16vdc from the I.E.E.E. 696 bus for its power supplies. The +8vdc is regulated by Q13, C7, C8 and C18 to provide +5vdc for the TTL & CMOS logic. It is also regulated by Q14, CR11, C9, C10 and C19 to provide +5.7vdc for the EPROM configuration logic. The +16vdc is regulated by Q15, CR15, C1, C3 and C5 to provide +12.7vdc for the EPROM configuration logic and the DC to DC converter U26. The -16vdc is regulated by Q16, C2, C4 and C6 to provide -5vdc for the EPROM configuration logic.

The high voltage required to program EPROMs is not available on the I.E.E.E. 696 bus. It is generated by a DC to DC converter formed by C11, C12, C13, C14, C15, R69, R70, R71, L1 and U26. This forms a step-up switching regulator with the frequency of operation controlled by C13 and the voltage sampled across R71 compared with an internal 1.2v reference. The resulting output voltage is filtered by C11 and C12 and current limited by R69 then supplied to the EPROM configuration logic.

The reset circuitry uses portions of U19 to or the two I.E.E.E. 696 bus reset signals; POC* and SLAVE CLR* with a software reset

signal from U9. The resulting signal clears the PROMBLASTER control register U7 and the device select flop U22. This tri-states the address and data lines to the programming socket U8 and resets any high or minus voltages on the EPROM type-specific pins.

The I/O device cycle on the I.E.E.E. 696 bus is controlled by U9, U10, U11, U12, U16, U17, U21 and U22. The group of four device numbers set by switches S1 are compared with A2-A7 by U11. When a match is found, and the comparator is enabled by either sINP or sOUT via U18 it provides an enable signal to the section of U10 driving the device select flop U22. If an extended address option is selected via OP1 the page address set by switches S2 are compared with A8-A15 by U12. This provides an additional enable to U10. The signals pSTVAL*, PHI 2 or pSYNC may be selected to clock the device select flop U22 via clock option OP2. The data inputs to U22 may be qualified by pSYNC via OP3, U10 and U16.The qualified I/O cycle address match or mismatch is clocked into U22 to control a PROMBLASTER bus cycle.

A bus cycle wait-state generator is formed by U21, U23 and U25. A two bit shift register is formed by U23. This shift register is cleared by pSYNC and clocked (shifted) by PHI 2. Taps are supplied to the I.E.E.E. 696 as either pRDY or xRDY via U25 when the PROMBLASTER is selected.

The device address A0-A1 is decoded by dual decoder U9 when enabled by a device select from U22. One half of U9 is strobed by the I.E.E.E. 696 data input strobe, pDBIN. The other half is strobed by the data output strobe, pWR* via U17. Thus depending on A0-A1 of the I/O read cycle, data input strobes are provided to: enable the U8 device data bus for input via U2, reset the voltage control latch U7, input the status of the one millisecond timer via U25 or reset the one millisecond timer to start another timing cycle. Depending on A0-A1 of the I/O write cycle data output strobes are provided to: latch data to be provided to the device socket U8, latch address lines A0-A7 for the U8 socket, latch A8-A9 and the PROMBLASTER mode bits and some of the EPROM configuration logic inputs, or latch the remaining voltage control and EPROM configuration logic inputs. Write cycles also enable the data output buffer U1 to supply bus data to U3, U4, U6 and U7.

The ads PROMBLASTER contains EPROM configuration logic that is software controlled to provide the necessary high voltage and current signals on the type-specific pins of the programming socket U8. This logic operates in one of three modes determined by the state of bits 6 & 3 of the latch U6. Modes 0 and 2 are provided for three supply EPROMS.

Mode 0 or 2 is detected by U15 and U18 when U6's outputs are enabled by U7 bit 7 being set. This disables the TTL logic level drivers R17, R63, CR7, Q9 and U16 on pin 23 of U8 and enables -5v to that pin via R11, R12, R64, R65, Q6, Q12 and U14 as a VBB supply for 1K and 2K three supply EPROMs. It also disables the TTL logic level drivers R20, R73, CR2, Q13 and U15 on pin 21 of U8 and enables +12v to that pin via R9, R10, CR1, Q5 and U13 as a VDD supply for 1K and 2K three supply EPROMS. Mode 0 is decoded by U10 and U17. The resulting signal disables the TTL logic level drivers R18, R66, CR11, Q10 and

U16 and provides +12v to pin 22 of U8 via R13, R14, CR10, Q7 and U14 as a programming enable for 1K three supply EPROMs. Mode 2 is decoded by U10, U14 and U17 to provide a +12v level to pin 26 of U8 via R15, R16, CR6 and Q8 as a programming enable for 2K three supply EPROMs. These programming enable levels are activated by bit 0 of control latch U7 via U17 and U10.

Mode 1 is decoded by U19 and U17 when U6's outputs are enabled by U7 bit 7 being set and is used to lower the output voltage of the DC to DC converter U26 via R74, R75, R76, Q14 and U14 by switching another resistor R76 in parallel with the voltage sensing resistor R70. This causes the high voltage to ramp down to +22v in about 1 second for the newer high density EPROMs.

Mode 3 is not explicitly decoded but is the default mode of operation. High voltage pulses can be provided on pins 1, 23, 22 and 20 of the device programming socket U8 to supply VPP for different EPROMs. TTL logic levels can be provided on pins 2, 27, 23, 22, 21 and 20 to be used as additional address lines or chip selects for different EPROMs. The high voltage levels applied to a pin disable the TTL logic '0' level drivers.

A timer is provided on card to allow software to control the pulse widths of the applied voltage levels. The 2 MHZ utility CLOCK signal on the I.E.E.E. 696 bus is divided by 2 via U22 to 1 MHZ for reliable +5v CMOS operation and then counted by a 14 stage counter U20. The eleventh stage output goes high after 1024 counts (one millisecond) and stops the counting by clearing U22 via U18 and U19. This signal is available as an interrupt via U14 and U19 or as an input that may be polled via U18, U19 and U25. The timer (counter) is reset by a I/O read operation via U9, U13, U18, U21, R68 and C16. The time constant supplied by R68 and C16 insures an minimum RST pulse width for +5v CMOS operation.

VII. Appendix - a software example

The following is a 8080 code example for the ads PROMBLASTER. It allows programming, verifying, and reading of INTEL 2716 EPROMS. After assembly and loading, it is invoked via DDT. After execution at the various entry points in the function table, control is returned to DDT with a RST 07 instruction. The result of the function is returned in the Z flag. A non-zero Z flag indicates successful completion of the function jumped to. Note that the PROMBLASTER I/O routines maintain a RAM copy of the I/O port's status to allow setting and resetting of individual bits. This code segment is presented as an example of PROMBLASTER control software. Full feature PROMBLASTER control is available with the ads PROMWRITER software.

```
                ; ADS PROMBLASTER INTEL 2716 EXAMPLE
                ;
00C0 =          PRMBAS EQU      0C0H    ;PROMBLASTER BASE PORT #
                ;
0100                    ORG     0100H
                ;
```

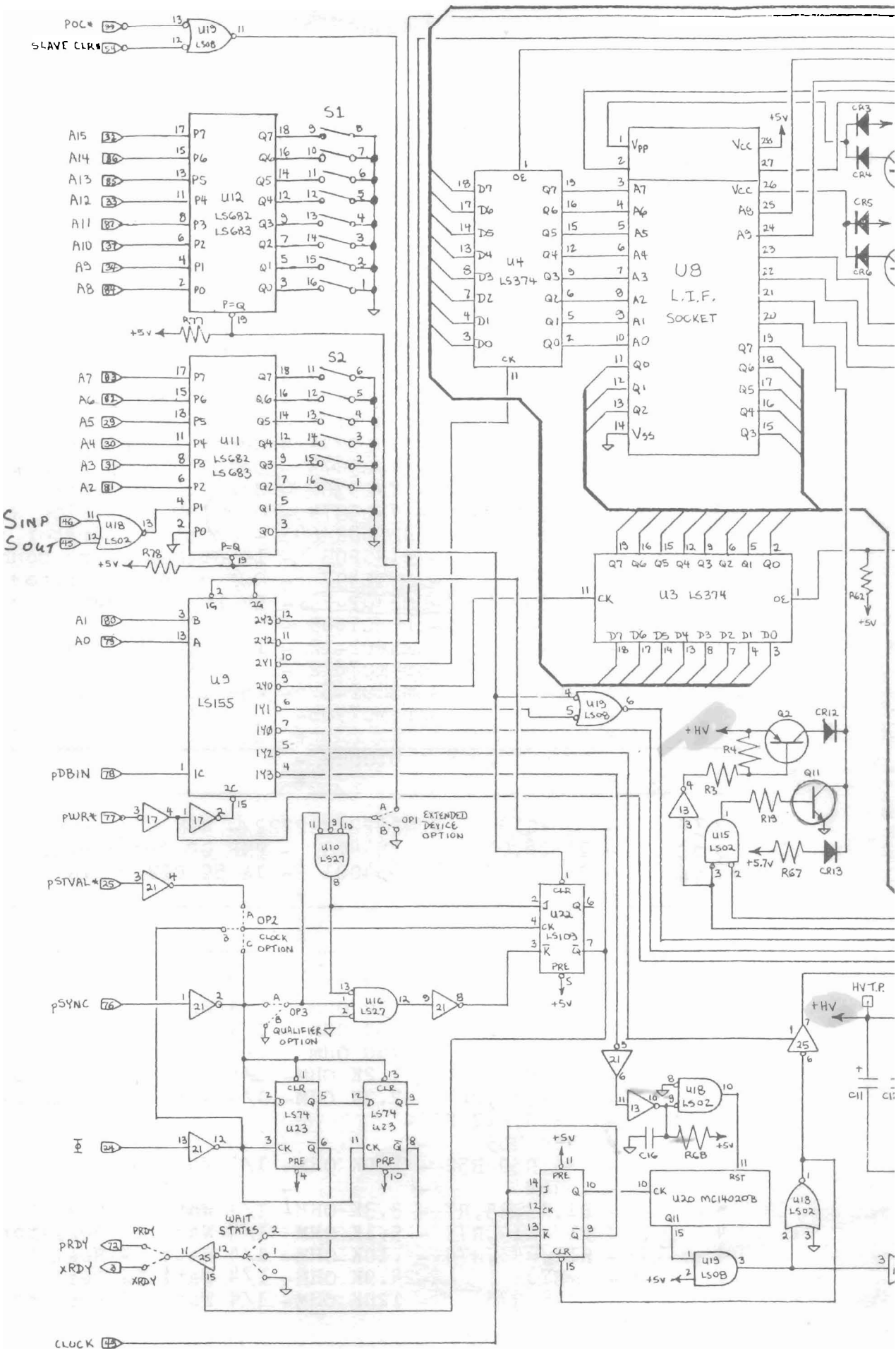
```
0100 CD6701            CALL    CONFIG  ;CONFIGURE PROMBLASTER FOR I2716
0103 FF                RST     7       ;CALL DDT
0104 CDDA01            CALL    CHECK   ;CHECK I2716 FOR UNBURNED STATE
0107 FF                RST     7       ;CALL DDT
0108 CD6A01            CALL    PROGRM  ;PROGRAM I2716 FROM -> TO
010B FF                RST     7       ;CALL DDT
010C CD8F01            CALL    VERIFY  ;VERIFY I2716 FROM -> TO
010F FF                RST     7       ;CALL DDT
0110 CDB501            CALL    READ    ;READ I2716 FROM -> TO
0113 FF                RST     7       ;CALL DDT
0114 C30000            JMP     0       ;CALL CP/M
                       ;
                       ; RAM DEFINITIONS
                       ;
0117 0000      FROM:   DW      0       ;FROM LOCATION
0119 0000      TO:     DW      0       ;TO LOCATION
011B 0000      POFF:   DW      0       ;PROM OFFSET
011D 00        PRDT:   DB      0       ;PROMBLASTER DATA PORT SAVE
011E 00        ADLO:   DB      0       ;PROMBLASTER ADDR LOW SAVE
011F 00        ADHI:   DB      0       ;PROMBLASTER ADDR HI SAVE
0120 00        VCTL:   DB      0       ;PROMBLASTER VOLTAGE CONTROL SAVE
                       ;
                       ; PROMBLASTER I/O ROUTINES
                       ;
0121 DBC0      PRDTIN: IN      PRMBAS
0123 C9                RET
                       ;
0124 D3C0      PRDTOT: OUT     PRMBAS
0126 C9                RET
                       ;
0127 321E01    ADLOOT: STA     ADLO
012A D3C1              OUT     PRMBAS+1
012C C9                RET
                       ;
012D 47        ORADHI: MOV     B,A
012E 3A1F01            LDA     ADHI
0131 B0                ORA     B
0132 321F01    ADHIOT: STA     ADHI
0135 D3C2              OUT     PRMBAS+2
0137 C9                RET
                       ;
0138 47        ANADHI: MOV     B,A
0139 3A1F01            LDA     ADHI
013C A0                ANA     B
013D C33201            JMP     ADHIOT
                       ;
0140 DBC1      RSVCTL: IN      PRMBAS+1
0142 C9                RET
                       ;
0143 47        ORVCTL: MOV     B,A
0144 3A2001            LDA     VCTL
0147 B0                ORA     B
0148 322001    VCTLOT: STA     VCTL
014B D3C3              OUT     PRMBAS+3
014D C9                RET
```

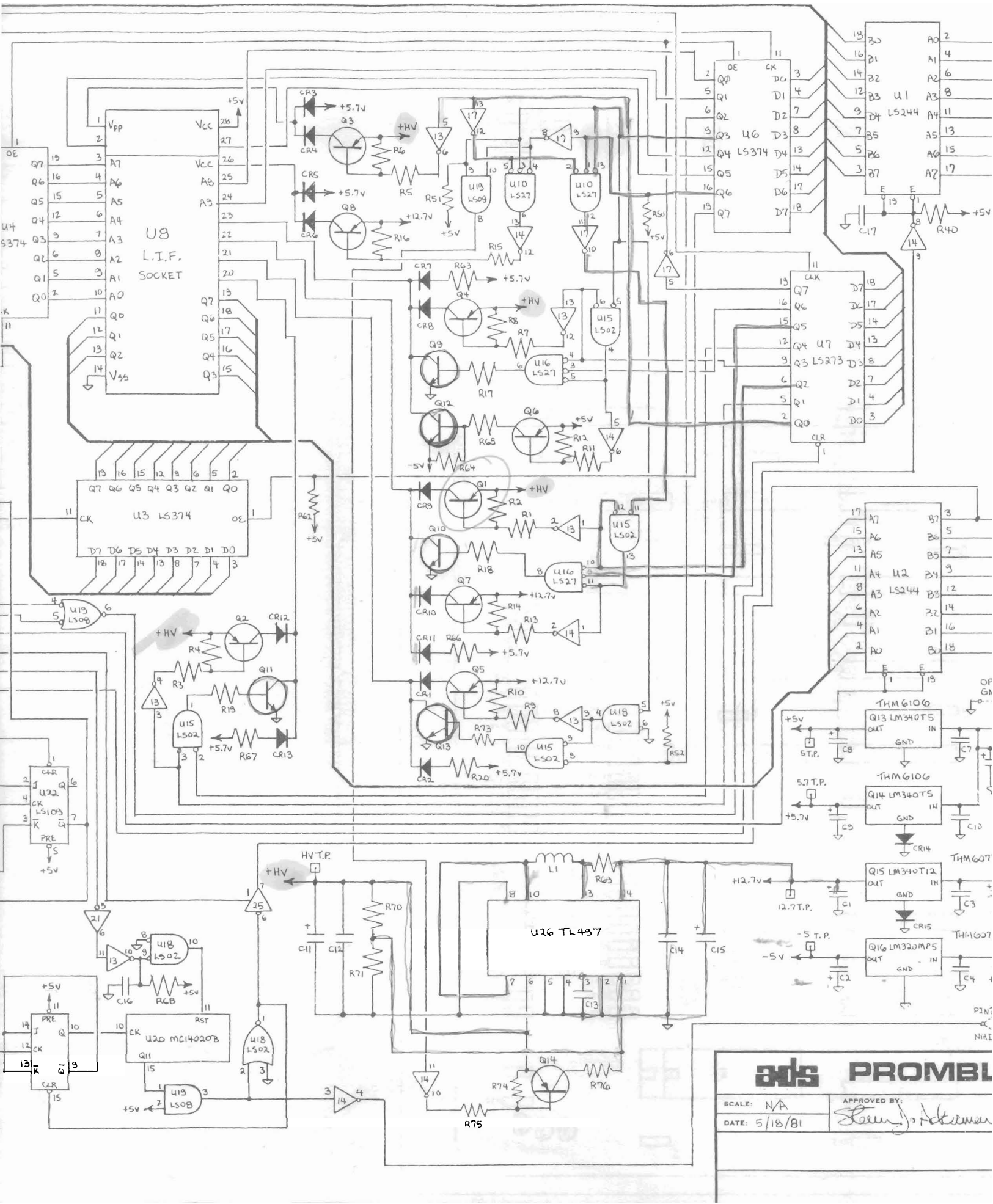
```
                        ;
014E 47         ANUCTL: MOV     B,A
014F 3A2001             LDA     UCTL
0152 A0                 ANA     B
0153 C34801             JMP     UCTLOT
                        ;
0156 CD6001     WAITIM: CALL    RSTTIM
0159 CD6301     WAITLP: CALL    CHKTIM
015C D25901             JNC     WAITLP
015F C9                 RET
                        ;
0160 DBC3       RSTTIM: IN      PRMBAS+3
0162 C9                 RET
                        ;
0163 DBC2       CHKTIM: IN      PRMBAS+2
0165 17                 RAL
0166 C9                 RET
                        ;
                        ; CONFIGURE THE PROGRAMMING SOCKET - U8
                        ;
0167 =          CONFIG  EQU     $
0167 C30002             JMP     I2716C  ;INTEL 2716
                        ;
                        ; PROGRAM I2716 FOR FROM -> TO LOCATIONS
                        ;
016A =          PROGRM  EQU     $
016A 210010             LXI     H,1000H ;FROM=1000H
016D 221701             SHLD    FROM    ;*
0170 21FF17             LXI     H,17FFH ;TO=FROM+2048
0173 221901             SHLD    TO      ;*
0176 =          PROGLP  EQU     $
0176 CD2802             CALL    I2716P  ;PROGRAM @ FROM
0179 2A1901             LHLD    TO      ;Q-FROM = TO?
017C EB                 XCHG            ;*
017D 2A1701             LHLD    FROM    ;*
0180 7C                 MOV     A,H     ;*
0181 BA                 CMP     D       ;*
0182 C28801             JNZ     NXTP    ;*
0185 7D                 MOV     A,L     ;*
0186 BB                 CMP     E       ;*
0187 C8                 RZ              ;RETURN IF FROM = TO
0188 =          NXTP    EQU     $
0188 23                 INX     H       ;FROM = FROM +1
0189 221701             SHLD    FROM    ;*
018C C37601             JMP     PROGLP  ;CONTINUE PROGRAMMING
                        ;
                        ; VERIFY I2716 FOR FROM -> TO LOCATIONS
                        ;
018F =          VERIFY  EQU     $
018F 210010             LXI     H,1000H         ;SETUP FROM
0192 221701             SHLD    FROM            ;*
0195 21FF17             LXI     H,17FFH         ;SETUP TO = FROM + 2048
0198 221901             SHLD    TO              ;*
019B =          VERILP  EQU     $
019B CD8002             CALL    I2716V          ;VERIFY I2716 @ FROM
```

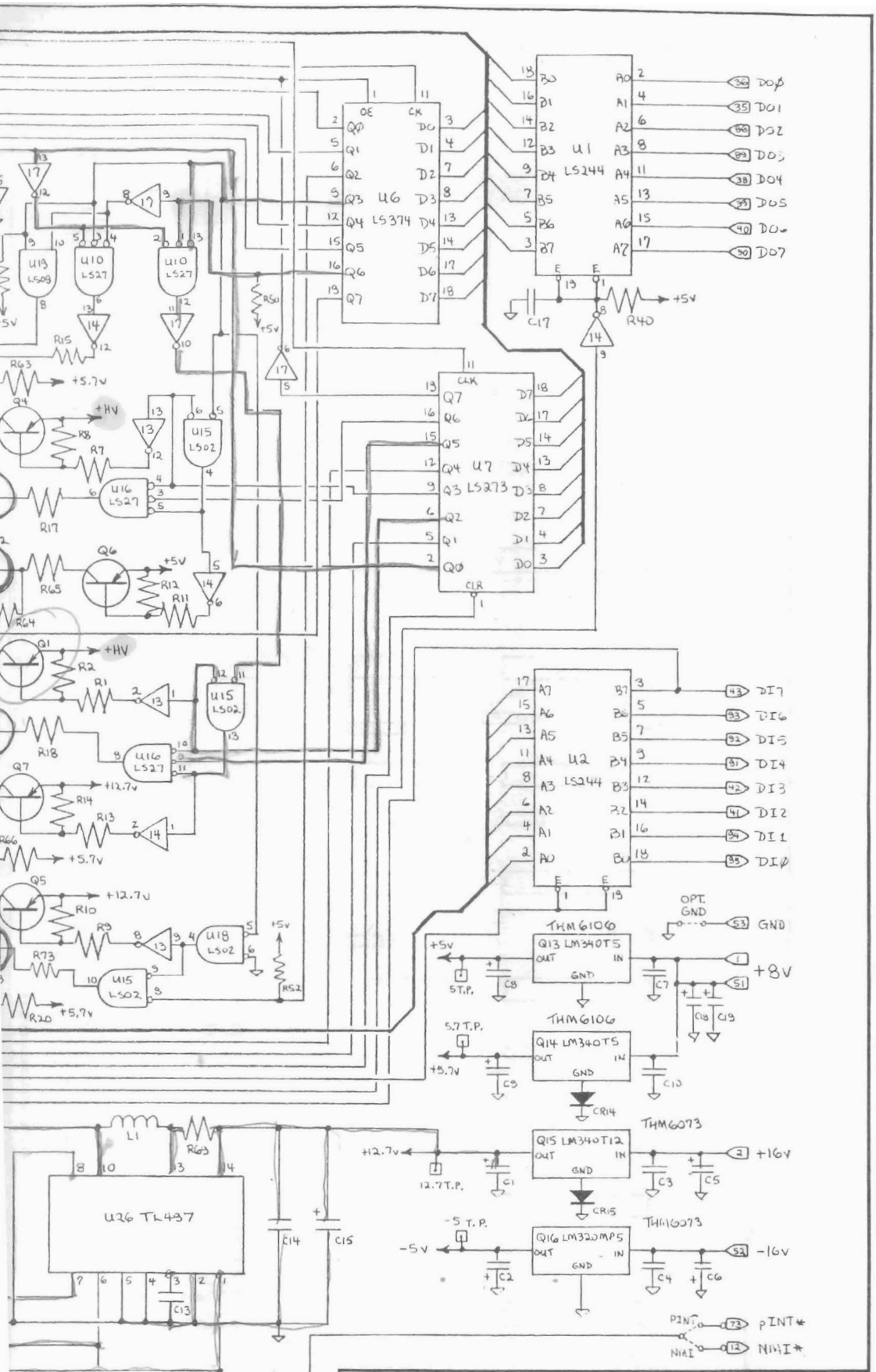
```
019E C0                 RNZ                     ;ERROR @ FROM
019F 2A1901             LHLD    TO              ;ADVANCE FROM, FROM=TO?
01A2 EB                 XCHG                    ;*
01A3 2A1701             LHLD    FROM            ;*
01A6 7C                 MOV     A,H             ;*
01A7 BA                 CMP     D               ;*
01A8 C2AE01             JNZ     NXTV            ;*
01AB 7D                 MOV     A,L             ;*
01AC BB                 CMP     E               ;*
01AD C8                 RZ                      ;RETURN IF DONE
01AE =          NXTV    EQU     $
01AE 23                 INX     H               ;FROM = FROM +1
01AF 221701             SHLD    FROM            ;*
01B2 C39B01             JMP     VERILP          ;CONTINUE VERIFYING
                ;
                ; READ I2716 FOR FROM -> TO LOCATIONS
                ;
01B5 =          READ    EQU     $
01B5 210010             LXI     H,1000H         ;SETUP FROM
01B8 221701             SHLD    FROM            ;*
01BB 21FF17             LXI     H,17FFH         ;SETUP TO = FROM +2048
01BE 221901             SHLD    TO              ;*
01C1 =          READLP  EQU     $
01C1 CD8902             CALL    I2716R          ;READ I2716 @ FROM
01C4 2A1901             LHLD    TO              ;ADVANCE FROM, FROM=TO?
01C7 EB                 XCHG                    ;*
01C8 2A1701             LHLD    FROM            ;*
01CB 7C                 MOV     A,H             ;*
01CC BA                 CMP     D               ;*
01CD C2D301             JNZ     NXTR            ;*
01D0 7D                 MOV     A,L             ;*
01D1 BB                 CMP     E               ;*
01D2 C8                 RZ                      ;RETURN IF DONE
01D3 =          NXTR    EQU     $
01D3 23                 INX     H               ;FROM=FROM + 1
01D4 221701             SHLD    FROM            ;*
01D7 C3C101             JMP     READLP          ;CONTINUE READING
                ;
                ; CHECK I2716 FOR FROM -> TO UNBURNED LOCATIONS
                ;
01DA =          CHECK   EQU     $
01DA 210010             LXI     H,1000H         ;SETUP FROM
01DD 221701             SHLD    FROM            ;*
01E0 21FF17             LXI     H,17FFH         ;SETUP TO
01E3 221901             SHLD    TO              ;*
01E6 =          CHEKLP  EQU     $
01E6 CD9102             CALL    I2716U          ;CHECK I2716 @ FROM FOR UNBURN
01E9 C0                 RNZ                     ;RETURN IF NOT
01EA 2A1901             LHLD    TO              ;ADVANCE FROM, FROM = TO?
01ED EB                 XCHG                    ;*
01EE 2A1701             LHLD    FROM            ;*
01F1 7C                 MOV     A,H             ;*
01F2 BA                 CMP     D               ;*
01F3 C2F901             JNZ     NXTC            ;*
01F6 7D                 MOV     A,L             ;*
```

```
01F7 BB             CMP     E               ;*
01F8 C8             RZ                      ;RETURN IF DONE
01F9 =          NXTC    EQU     $
01F9 23             INX     H               ;FROM =FROM +1
01FA 221701         SHLD    FROM            ;*
01FD C3E601         JMP     CHEKLP          ;CONTINUE CHECKING
                ;
                ; SINGLE SUPPLY 2K PARTS
                ;
                ;       INT2716,MCM2716,TMS2516
                ;---------------------------------------
                ;
                ; CONFIGURE PROMBLASTER U8 SOCKET FOR I2716
                ;
0200 =          I2716C  EQU     $
0200 3E60           MVI     A,01100000B     ;VPP=5V,G=5V
0202 CD4801         CALL    VCTLOT          ;*
0205 3EC8           MVI     A,11001000B     ;DISBL DATA, MODE 3
0207 CD3201         CALL    ADHIOT          ;*
020A AF             XRA     A               ;ADDR LO=0, DATA = 0
020B CD2701         CALL    ADLOOT          ;*
020E CD2401         CALL    PRDTOT          ;*
0211 C9             RET
                ;
                ; SETUP A0-A10 SUBROUTINE
                ;
0212 =          I2716S  EQU     $
0212 3A1701         LDA     FROM            ;ADDRESS SETUP
0215 CD2701         CALL    ADLOOT          ;*A0-A7
0218 3A1F01         LDA     ADHI            ;*A8-A10
021B E6F8           ANI     11111000B       ;**
021D 47             MOV     B,A             ;**
021E 3A1801         LDA     FROM+1          ;**
0221 E607           ANI     00000111B       ;**
0223 B0             ORA     B               ;**
0224 CD3201         CALL    ADHIOT          ;**
0227 C9             RET
                ;
                ; PROGRAM I2716 @ FROM SUBROUTINE
                ;
0228 =          I2716P  EQU     $
0228 CD1202         CALL    I2716S          ;ADDRESS SETUP
022B 3E80           MVI     A,10000000B     ;ADDR ENBL
022D CD4301         CALL    ORVCTL          ;*
0230 3E08           MVI     A,00001000B     ;VPP=25V
0232 CD4301         CALL    ORVCTL          ;*
0235 2A1701         LHLD    FROM            ;SETUP DATA
0238 7E             MOV     A,M             ;*
0239 CD2401         CALL    PRDTOT          ;*
023C 3E7F           MVI     A,01111111B     ;ENBL DATA
023E CD3801         CALL    ANADHI          ;*
0241 3E10           MVI     A,00010000B     ;PROGR=5V
0243 CD4301         CALL    ORVCTL          ;*
0246 0632           MVI     B,50            ;WAIT FOR 50 MSEC
0248 =          I2716L  EQU     $           ;*
```

```
0248 CD5601             CALL    WAITIM          ;*
024B 05                 DCR     B               ;*
024C C24802             JNZ     I2716L          ;*
024F 3EEF               MVI     A,11101111B     ;PROGR=0V
0251 CD4E01             CALL    ANDCTL          ;*ROGR=0V
0254 3EF7               MVI     A,11110111B     ;VPP=5V
0256 CD4E01             CALL    ANDCTL          ;*
0259 3E80               MVI     A,10000000B     ;DISBL DATA
025B CD2D01             CALL    ORADHI          ;*
025E 3E7F               MVI     A,01111111B     ;DISBL ADDR
0260 CD4E01             CALL    ANDCTL          ;*
0263 C9                 RET
                ;
                ; READ I2716 @ FROM SUBROUTINE
                ;
0264 =          I2716Q  EQU     $
0264 CD1202             CALL    I2716S          ;ADDR SETUP
0267 3E80               MVI     A,10000000B     ;ENBL ADDR
0269 CD4301             CALL    ORVCTL          ;*
026C 3EDF               MVI     A,11011111B     ;G=0V
026E CD4E01             CALL    ANDCTL          ;*
0271 CD2101             CALL    PRDTIN          ;READ PROM
0274 4F                 MOV     C,A             ;SAVE
0275 3E20               MVI     A,00100000B     ;G=5V
0277 CD4301             CALL    ORVCTL          ;*
027A 3E7F               MVI     A,01111111B     ;DISBL ADDR
027C CD4E01             CALL    ANDCTL          ;*
027F C9                 RET
                ;
                ; VERIFY I2716 @ FROM SUBROUTINE
                ;
0280 =          I2716V  EQU     $
0280 CD6402             CALL    I2716Q          ;READ PROM
0283 2A1701             LHLD    FROM            ;PNT2 DATA
0286 7E                 MOV     A,M             ;GET DATA
0287 B9                 CMP     C               ;TEST AGAINST PROM
0288 C9                 RET
                ;
                ; READ I2716 @ FROM SUBROUTINE
                ;
0289 =          I2716R  EQU     $
0289 CD6402             CALL    I2716Q          ;READ PROM
028C 2A1701             LHLD    FROM            ;PNT2 TARGET ADDR
028F 71                 MOV     M,C             ;SAVE DATA READ
0290 C9                 RET
                ;
                ; CHECK I2716 @ FROM FOR UNBURNED SUBROUTINE
                ;
0291 =          I2716U  EQU     $
0291 CD6402             CALL    I2716Q          ;READ PROM
0294 3EFF               MVI     A,11111111B
0296 B9                 CMP     C               ;TEST PROM FOR UNBURNED
0297 C9                 RET
                ;
0298                    END
```


POC*
SLAVE CLR*
U19 LS08
S1
A15
A14
A13
A12
A11
A10
A9
A8
U12 LS682 LS683
P=Q
R77
+5V
S2
A7
A6
A5
A4
A3
A2
U11 LS682 LS683
SINP
SOUT
U18 LS02
R78
A1
A0
U9 LS155
pDBIN
pWR*
U10 LS27
OP1
EXTENDED DEVICE OPTION
pSTVAL*
OP2
CLOCK OPTION
U22 LS109
pSYNC
OP3
QUALIFIER OPTION
U16 LS27
LS74 U23
Φ
WAIT STATES
PRDY
XRDY
CLOCK
U4 LS374
U8 L.I.F. SOCKET
Vpp
Vcc
Vss
U3 LS374
CR3
CR4
CR5
CR6
R62
U13 LS08
Q2
CR12
+HV
R4
R3
Q11
R19
U15 LS02
+5.7V
R67
CR13
HV T.P.
C11
C16
R68
U20 MC14020B
RST
Q11


U8
L.I.F.
SOCKET
U3 LS374
U4 LS374
U1 LS244
U2 LS244
U5 LS273
U6 LS374
U10 LS27
U11 LS02
U13 LS08
U15 LS02
U16 LS27
U18 LS02
U19 LS08
U20 MC14020B
U22 LS109
U26 TL497
THM6106
Q13 LM340T5
Q14 LM340T5
Q15 LM340T12
Q16 LM320MP5
+HV
+5V
+12.7V
-5V
HV T.P.
5T.P.
12.7T.P.
-5 T.P.
SCALE: N/A
DATE: 5/18/81
APPROVED BY:



PROMBLASTER

SCALE: N/A	APPROVED BY:	DRAWN BY SJA
DATE: 5/18/81		REVISED 4/18/82
		DRAWING NUMBER PB-2