Using PALs for Microcomputer Address Decoding

Bart Addis
August 1, 1985

Introduction

PALs, Programmable Array Logic chips, are easily customizable arrays of AND, OR, and NOT gates. They can be configured to replace "random logic" as well as standard functional units such as multiplexers, adders, shift registers, and counters. The advantages of PALs include reduced board space, the ability to quickly redesign a circuit, and in most cases, speed.

A good, although simple, use of PALs is to replace the address decoding logic (as well as various "random logic" functions) of a typical Single Board Computer (SBC). The computer I used was from the MAE 412 class, and has a 6502 microprocessor. This 6502 must address 2K of RAM, 4K of ROM, two 6522 Versatile Interface Adaptors (VIAs), and a pair of TIL311 hexadecimal displays.

The purpose of this example was to show first how a PAL can be designed by hand and then how a computer program can be used to design the circuit more easily. The software I used was AMD's PALASM20; the programmer was an Advanced Microcomputer Systems (AMS) PROM 2000-4 PAL programmer.

Description

SBC Logic Functions

The address space of the 6502 was assigned to the devices according to the following chart:

Address	Device
$0000-$07FF	6116 RAM
$4000	TIL311 displays
$8000-$800F	VIA #1
$A000-A00F	VIA #2
$F000-$FFFF	TI 2532 EPROM

For the RAM, ROM, and displays, the address logic need only decode two bits; for the VIAs, three are needed. Those devices being written to (RAM, TIL311s) must only be selected while the ϕ_2 clock is high. The resulting logic looks like:

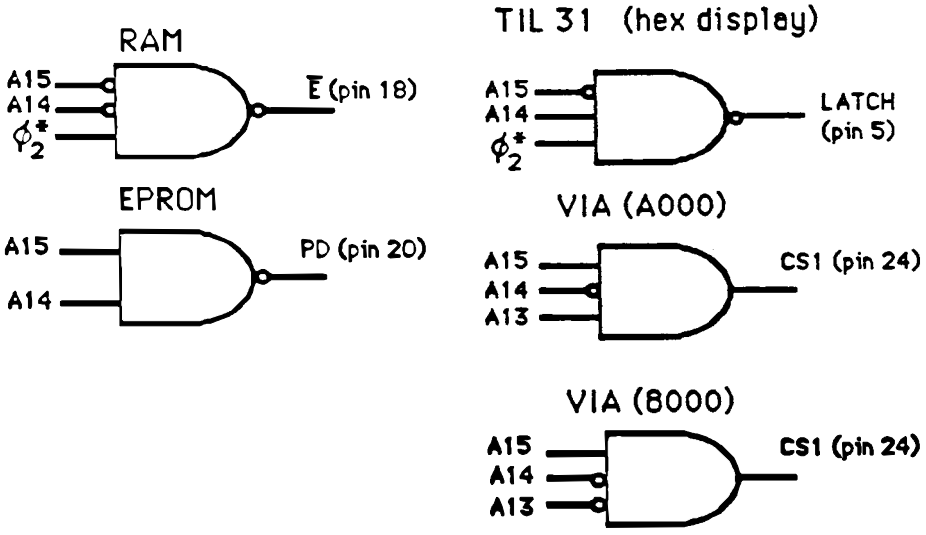


Figure 1: Address Decoding Logic.

In order to completely replace all of the 74XX/74LSXX chips on the SBC, two more functions are needed. One is to buffer the ϕ_2 clock, which must clock the VIAs as well as drive the PAL logic. 20mA of current sinking capability is required. A National PAL16L8 can sink 24mA, so it is acceptable. The buffered signal is called ϕ_2*.

The second additional function that the PAL must contain is to invert the power-up RESET signal, which is provided by a 555 timer circuit. The RESET signal is active high, but the 6502 RES' input is active low, so the signal must be inverted.

About PALs

A PAL, architecturally, is a programmable array of AND gates connected to a fixed (not programmable) array of OR gates. The 16L8 is a typical PAL, and its internal structure is shown on page 8. The diagram is drawn to be used as a programming sheet, so the AND gates, instead of each having 32 inputs drawn, are shown with only one input line that crosses all 32 of the *real* input lines. The *product* lines represent the outputs of the ANDs (the AND function, like an algebraic product, is written without a symbol: thus AB, the product of A and B, is also read as A AND B. Similarly, OR is often referred to as sum, and is written A + B). Internally, each one of the intersections in the input/product grid represents a fuse. A given AND gate is connected to only those inputs whose fuses aren't blown. The output of an AND that is connected to all the inputs (in its original, unprogrammed state) is always low, whereas the output of an AND that is totally disconnected (all fuses blown) is always high.

Notice that the 16L8 (and all PALs) have only two levels (AND-OR) of logic. For all combinatorial logic functions, two levels is sufficient: "any Boolean function can be expressed as a sum of minterms [products]."[1] The practical limitation of a PAL in this respect is that each OR can only accept 7 product terms (see page 8).

Design

The first thing that must be done with any design is to reduce it to the AND-OR "sum of products" form to fit it into a PAL. Since the functions in this case are all ANDs (or inverter/buffers), they are already simple enough.

The next step is to pick a PAL. I noticed that all of the functions but two (the VIA enables) are inverted. I picked a PAL16L8, which has 8 inputs, 8 outputs, and is active low (i.e. all the outputs are inverted--see page 8). Various other output configurations are available, including registered outputs. The VIA enables can be changed to their active-low equivalents, NORS.

The final design is:

[1]Morris Mano, Digital Logic and Computer Design. (Englewood Cliffs, N.J.: Prentice-Hall, Inc., 1979), p.49.

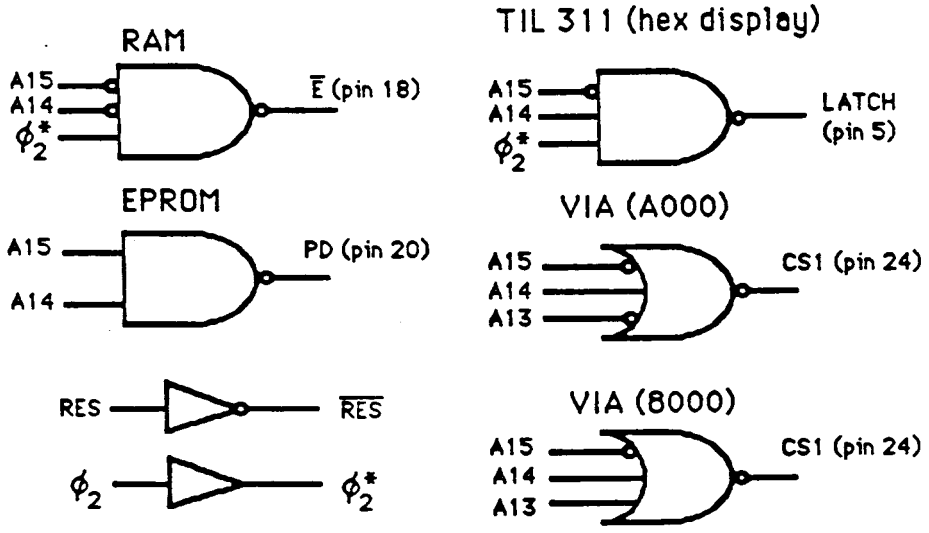


Figure 2: The Final Design.

Programming

Now that the PAL is laid out, it is time to put the design onto the chip. The easiest way to do it is by using an assembler (specifically, PALASM from MMI or AMD, PLAN from National, or CUPL from Assited Technology), whose input is a boolean equation and whose output is the JEDEC PLDTF (Programmable Logic Data Transfer Format) file accepted by the AMS software. The other option is to hand-code the PLDTF file itself, which is akin to hand-assembling a program: it is tedious, but one learns what is actually going on. I will use both methods to design this decoder.

Hand Assembly

The first step in designing the PAL by hand is to actually draw the logic functions onto the PAL diagram to determine what fuses need to be blown. First I assigned input and output pins as follows:

Input (pins 1-9)		Output (pins 12-19)	
ϕ_2	1	EPROM	19
A13	2	TIL	18
A14	3	RAM	17
A15	4	VIA #1	16
RES	5	VIA #2	15
		ϕ_2^*	14
		RES'	13

Next, I drew an "X" at each intersection that needed to be connected. The output buffer/inverter on the far right of each gate is actully a TRI-STATE gate. I wanted all the TRI-STATE gates to be active, so I tied all the enable lines (the top product term in each case) high by disconnecting all the inputs. The unused product terms, including those of the last, unused gate, were all left connected, which I indicated by putting an "X" in the AND gate outline. The finished diagram is on page 9.

Once the design has been drawn onto the PAL diagram, an input file for the programming software must be made. Although this file is just a bitmap of the fuses to be blown, it has a complicated format. Below is an outline of the format. It includes the important features. For an explanation of an item makred "optional" see the JEDEC specification on pages 10 and 11. Here is the format:

1. ASCII "STX" (02 hex) character
2. Design Specification Identifier (as many lines of comments as you care to write)
3. "*"
4. (optional) Device Code
5. "F"
6. Fuse Default State--"0" or "1"
7. "*"
8. Any number of *link* lines consisting of:
   a. "L"
   b. A 4 digit decimal address of the first link (addresses start in the upper left with 0000 and proceed left-to-right)
   c. bitmap of fuses starting at the given address.
      "0"=leave fuse connected
      "1"=burn fuse
   d. "*"
9. (optional) Checksum Field
10. (optional) Test Information
11. ASCII "STX" character
12. (optional) Sumcheck Field

A typical PLDTF file is on page 12.

Computer Assembly

Using the computer to create the PLDTF files can save a lot of time and prevent a lot of errors. The various assembler programs available all take input in the form of boolean equations, and some also accept netlists from schematic design programs. We have AMD's version of MMI's original software, PALASM20. The complete manual for PALASM20 is on the PALASM20 disk (PALASM20.HLP), and is in section 4 of the AMD PAL Handbook, so I will only give a simplified introduction to its use and syntax here.

The file starts with a 4 line header. The first line *must* start with the part number of the PAL to be used. The currently valid part numbers are:

AMPAL16L8	PAL16L8
AMPAL16H8	PAL16H8
AMPAL16R8	PAL16R8
AMPAL16R6	PAL16R6
AMPAL16LD8	PAL16LD8
AMPAL16HD8	PAL16HD8

Those part numbers correspond to all the parts in AMD's three conventional PAL families (standard, high speed, and half power), as well as to a large number of the parts made by other companies. Presumably AMD's newest parts (the 22V10 and 16P8) will be supported when they are shipped.

The rest of the header is treated as a comment, but for the sake of being standard, AMD recommends the following:

LINE 1: Ends with the words "PAL Design Specification"

LINE 2: User internal part number, name.
LINE 3: Device application name.
LINE 4: Company name and address.

For the SBC I used the lines:

```
AMPAL16L8                                          PAL Design Specification
SBC003                                                    Bart Addis 7/23/85
Single Board Computer Address Decoding Circuitry
Princeton University          MAE Microprocessor Lab      Princeton NJ, 08544
```

The next two lines list the pin names in sequential order from 1 to 20. the pin names can be up to eight characters long, and there must be exactly 20 of them. It is customary to name pin 10 "GND" and pin 20 "VCC" for obvious reasons. If a pin has no connection, it should be labeled "NC." Pin names can't inlcude the characters: = : * + ( ) ; " or periods. For the SBC I used the pin names:

```
phi2    a13    a14    a15     pureset  nc       nc     nc     nc      gnd
nc      nc     res    phi2star via8000 viaA000  ram    til    eprom   vcc
```

The last part of this simplified specification file is the list of equations that define the PAL's functions. The important symbols are:

```
/.....NOT
*.....AND
+.....OR
=.....EQUALS
```

The boolean functions are expressed as equations, with an output pin name on the left and the function on the right Parenthesis are not allowed. In the case of active-low PALs, the left hand pin names must start with / (because the output buffers invert the signals). The equations I used for the SBC are:

```
/phi2star  = /phi2
/res       = pureset
/ram       = /a14*/a15*phi2star
/til       = a14*/a15*phi2star
/eprom     = a14*a15
/via8000   = /a13 + a14 + /a15
/viaA000   = a13 + a14 + /a15
```

Many more specifications can be made, which I have not described. They are described in section 4 of the AMD PAL Handbook. They include:

- Testing the design by simulating the application of test vectors
- Loading PAL registers (16R8, 16R6, and 16R4 only)

An intermediate form of the output is available from PALASM in addtion to the PLDTF file. It is called a *map*, and shows what fuses will be burned, and where they are. It can easily be compared to the schematic created by a hand design to check for errors.

Using the PALASM software is simple. To use it, do the following (type the underlined characters or words, followed by a carriage return):

1. Put the disk containing the PALASM software in drive A:

2. Turn on or reboot the sytem if the prompt isnt "A>"

3. From the "A>" prompt, type PALASM20.

4. Type in the name of the file that the Design Specification is in.

5. If this is a test, have the output go to the CRT: C. Otherwise, send it to the disk: D. If you specify D, type in the filename. It is customary to use and extension of JED for JEDEC PLDTF files, and .MAP for map files.

6. Select an output format: JEDEC PLDTF: J or a map: M.

7. If you want both JEDEC and a map, and are saving them on the disk, select C to change the filename, and repeat step 6 again.

8. when you are finished, type Q to quit.

Listings of the PALASM input file, the JED output file and the .MAP file for the SBC are on pages 18-21.

Using the AMS Programmer

Although the AMS programmer is supposed to accept JEDEC input files, it actually has slightly different requirememnts. The most major is that it cannot accept a space at the beginning of or in the middle of a link line. Therefore the line:

L0000 1111 1111 1111 1111 1111 1111 0111 1111 *

must be converted into the line:

L0000 11111111111111111111111101111111*

Also, the AMS software can accept a file with only link fields (because that is the way it stores it internally. Furthermore, since all it reads from the file are the link fields, it dosn't require all the fancy ASCII codes (except for the asterisks and the F in the fuse default state line).

An important time-saving fact about PLDTF files is that fuses not specified in a link line revert to the default state. If that state is 0, all fuses not specified will remain connected.

The complete PLDTF file for the SBC001 chip is on page 13. It is the same whether done by computer or by hand. In the hand assembly example, I used the WordStar editor in "non-document" mode to create the file. For the computer example, I used WordStar to remove the extraneous spaces. In either case, any PC editor can be used. To help with the tedious job of creating a JEDEC file from a hand assembly, I created a file, called GENERIC.JED, that contains a sample header and a complete set of link lines. You can use it to create a JEDEC file more easily.

Finally, to program the PAL, one does the following (type the underlined text):

1. Turn the PC on if it isn't already on. If the PAL programmer cable isn't connected to the back of the PC, connect it before turning the PC on.

2. Put the AMS disk in the A drive.

3. From the DOS A> prompt type: palprog

4. Type in the base address: 0200

5. Specify the manufacturer: M

6. Specify the chip type: T

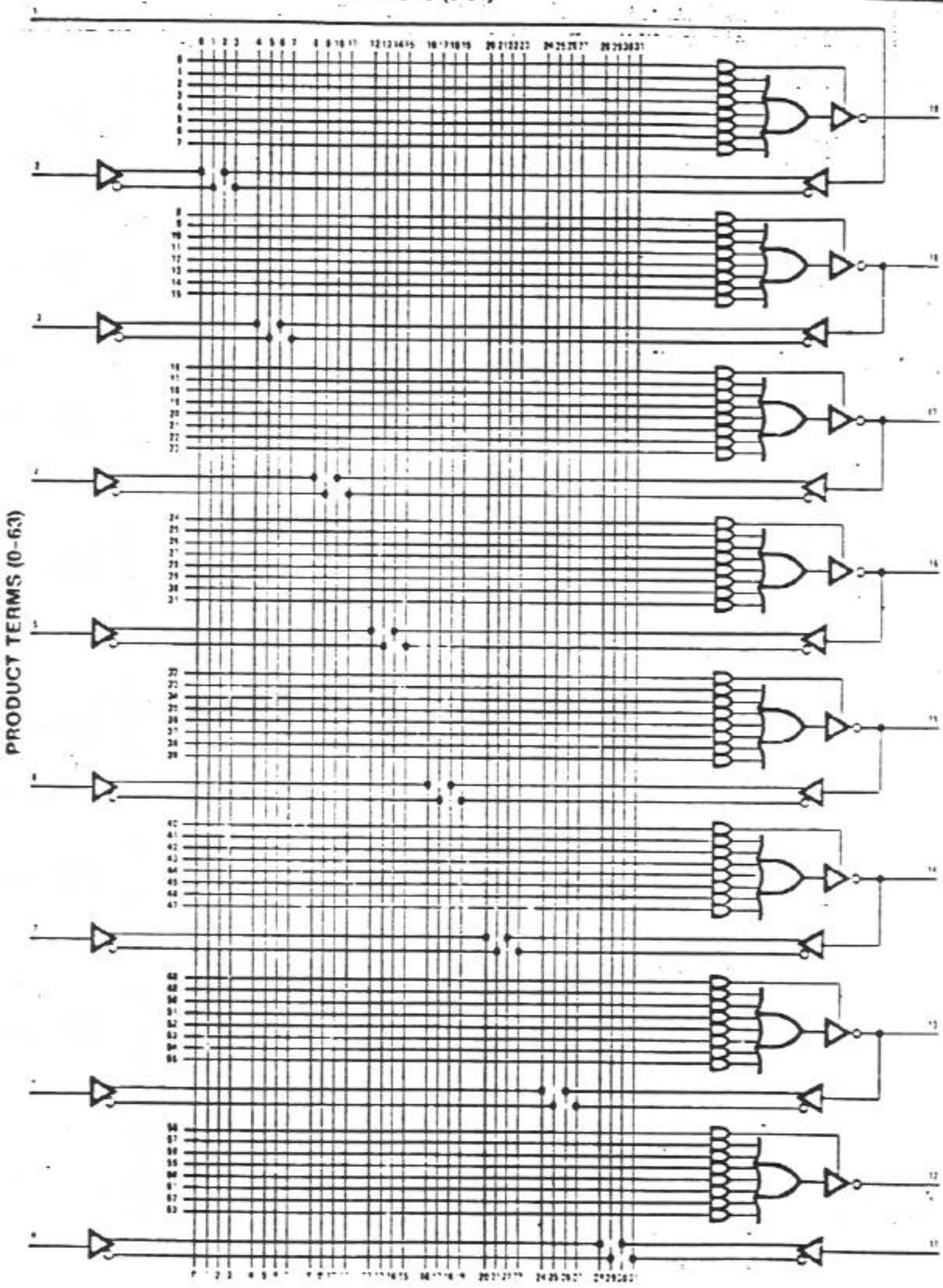
7. Load the PLDTF file: L

8. If you want, you can look at the bitmap: S, or edit it: B

9. Program the chip: P

10. Quit the program: E

A transcript of the programming session for the SBC PAL is on pages 14 through 17.

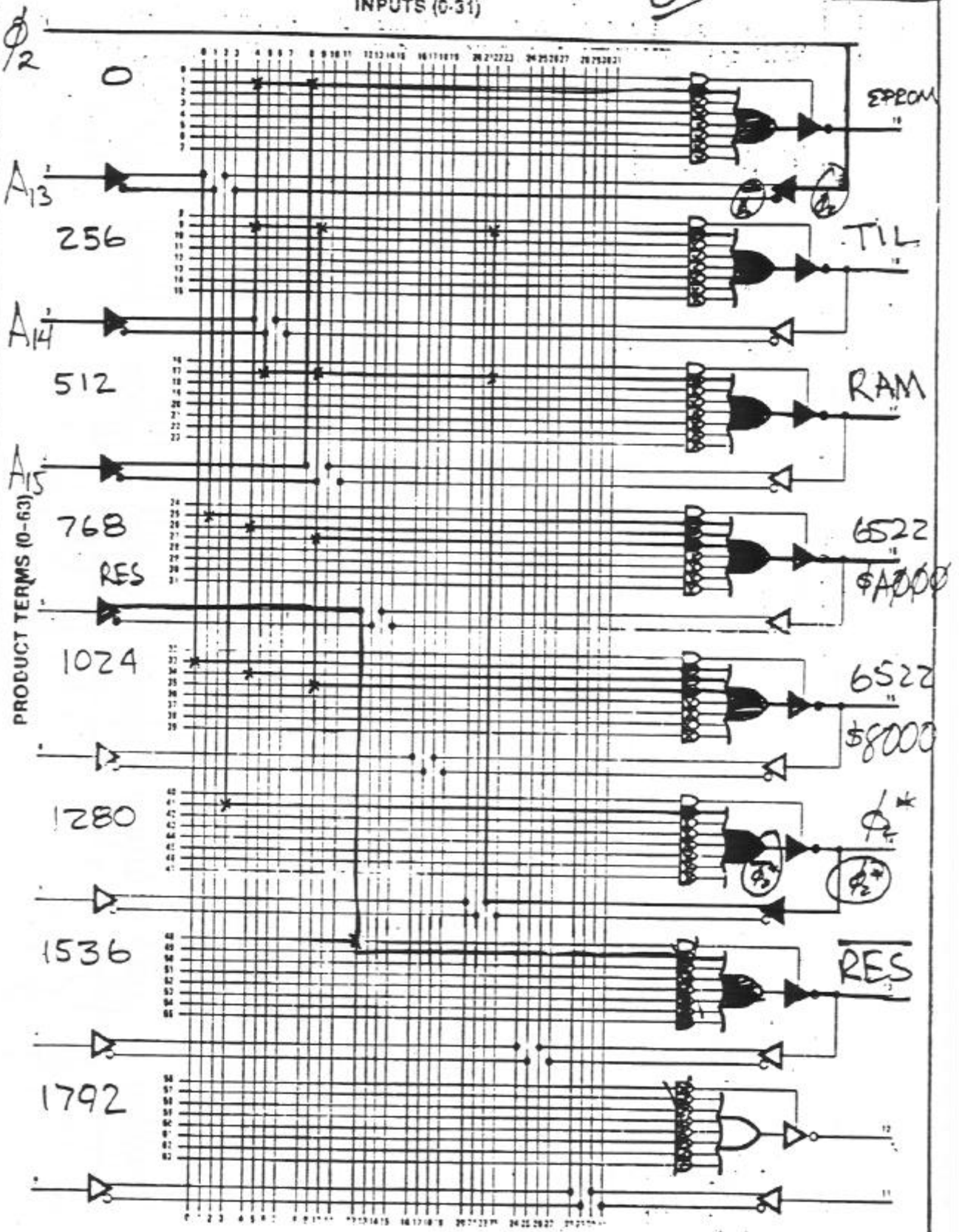


Logic Diagram, PAL16L8.

From National Bipolar Handbook, page.

PAL Des

SBC001 OK



0 = CONNEX BLANK LINE: ALL 0

Logic Diagram, PAL16L8. (NO FUSES BLOWN)

Adapted From National Bipolar Handbook, 255

Page ____

APPENDIX A

JEDEC Standard Programmable Logic Data Transfer Format "J"

The Joint Electron Devices Engineering Council (JEDEC) programmable logic data transfer format (PLDTF) is a universal transfer format for fuse and test information between hosts and intelligent device programmers. This format is an industry standard used by many commercial programmer manufacturers. It gives each fuse a unique decimal address (specified in a logic diagram schematic) and indicates the state of each fuse for the programmed part. The format consists of four main sections: the design specification identifier, fuse link information, structured functional test information, and the sumcheck. The output of this command is a fuse map ready to be downloaded to a PAL programmer.

Design Specification Identifier

The DESIGN SPECIFICATION identifier is a heading used by the designer to document the device to be programmed. The user is free to specify any documenting text desired. The AMD recommended format is identical to the first four lines of the PAL DESIGN SPECIFICATION. The heading is begun with an ASCII "STX" (02 HEX) and is terminated with an ASCII asterisk "*" (2A HEX). Notice that this requires the PAL DESIGN SPECIFICATION heading to be asterisk free.

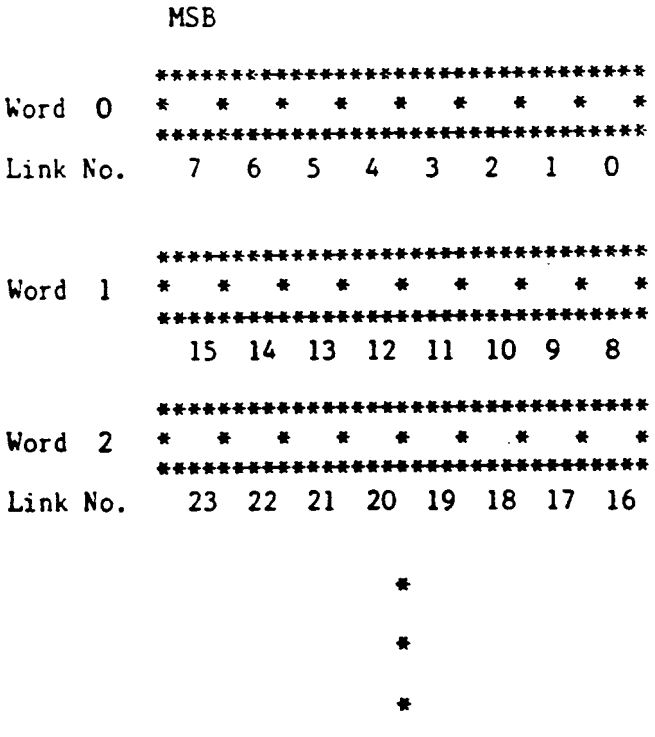
An optional device code can be specified, indicating to the device programmer the type of part to be programmed. This code is a variable length decimal number, preceded by an ASCII "D" (44 HEX) and terminated by an ASCII "*" (2A HEX).

Fuse Link Information

This section of the format defines the state of each fuse, and consists of three fields: fuse default state, link information, and checksum. The first field is optional and is used to indicate the fuse default state. If a fuse state isn't otherwise specified, the default state will be used. The default state is specified by an ASCII "F" (46 HEX) followed by an ASCII "0" (32 HEX) for a low resistance link or an ASCII "1" (33 HEX) for a high resistance link. The field is terminated with an ASCII "*" (2A HEX).

The second field, link information, identifies the state of each fuse individually. The field begins with an ASCII "L" (4C HEX) followed by an ASCII decimal fuse address of variable length, terminated by an ASCII "*" (2A HEX). This indicates the fuse address of the first fuse state. The individual fuses are specified by an ASCII "0" (32 HEX) or an ASCII "1" (33 HEX) indicating low resistance and high resistance respectively. The fuse address is incremented for each additional fuse state. Thus fuse states can be specified sequentially. Any number of fuse addresses may be specified by inserting additional "L" fields. If a link is specified 2 or more times, the last state replaces the preceding entries.

The third field is an optional checksum for the link information. This checksum is computed by performing a 16-bit addition of the 8-bit words constructed from the specified state of each fuse link in the device. The 8-bit words are constructed sequentially from the single bit fuse state definitions. The method of constructing these words is shown below.



The word encompassing the last link is constructed by setting zeros for all bit locations more significant than the last link. The 16-bit sum is expressed as 4 ASCII Hex characters. On ASCII "C" precedes the four Hex characters. The last character is followed by an ASCII "*".

Structured Functional Test Information

The structured functional test information is an optional field which can define test vectors to be used by intelligent programmable logic device programmers to test the logical functionality of a programmed device. These vectors specify the driven state for inputs and the checked value for outputs.

The field is specified with an ASCII "V" followed by a variable length decimal test vector address. The address is terminated by an ASCII "W". Following the test vector address is a series of 20 characters specifying the driven or tested state for each pin.

The format for each character in the test vector output is given below:

- H Test output for a logical HIGH
- L Test output for a logical LOW
- 1 Drive input to a logical HIGH
- 0 Drive input to a logical LOW
- X Irrelevant. If an output do not test. If an input drive to a logical LOW as a default.
- C Drive input from logical LOW to logical HIGH. (clock pulse).
- Z Test output for high impedance.

The test vector is terminated with an ASCII asterisk "*". Multiple test vectors are specified by incrementing the decimal vector address by 1 for each additional test vector.

Sumcheck

The sumcheck field provides redundancy to help in detecting errors in data transmission. This field is constructed by performing a binary addition of each character between the STX and ETX in the transmitted file. The resulting least significant 16 bits are the sumcheck. This number is represented as four Hex characters preceded by an ASCII asterisk "*" in the final printout.

Examples of the JEDEC transfer format are shown previously in descriptions of the JEDEC and SIMULATE commands.

From AMD PAL Handbook,
D 4-19

```
ENTER PAL20 OPTION: J

AMPAL16R4                                        PAL DESIGN SPECIFICATION
PATTERN NUMBER K2/044-C                             WARREN MILLER 4/1/82
DUAL BUS 4-BIT COUNTER WITH BIT SWAPPABLE OUTPUTS
ADVANCED MICRO DEVICES    901 THOMPSON PLACE  SUNNNYVALE CA 94086
*D9724
*F0*
L0000 1111 1111 1111 1111 1111 1111 0111 1111 *
L0032 1111 1111 1110 1111 1111 1011 1111 1111 *
L0064 1111 1111 1111 1111 1111 0110 1111 1111 *
L0256 1111 1111 1111 1111 1111 1111 0111 1111 *
L0288 1111 1111 1111 1110 1111 1011 1111 1111 *
L0320 1111 1111 1111 1111 1110 0111 1111 1111 *
L0512 1111 1111 1111 1111 1111 1111 1111 0111 *
L0544 1111 1111 1111 1111 1011 1011 1011 1111 *
L0576 1110 1111 1111 1111 1111 0111 1011 1111 *
L0608 0111 1111 1101 1101 1101 1101 0111 1111 *
L0640 1111 1111 1110 1110 1111 1111 0111 1111 *
L0672 1111 1111 1110 1111 1110 1111 0111 1111 *
L0704 1111 1111 1110 1111 1111 1110 0111 1111 *
L0736 1011 1111 1110 1111 1111 1111 0111 1111 *
L0768 1111 1111 1111 1111 1111 1111 1111 0111 *
L0800 1111 1111 1111 1011 1111 1011 1011 1111 *
L0832 1111 1110 1111 1111 1111 0111 1011 1111 *
L0864 0111 1111 1111 1101 1101 1101 0111 1111 *
L0896 1111 1111 1111 1110 1110 1111 0111 1111 *
L0928 1111 1111 1111 1110 1111 1110 0111 1111 *
L0960 1011 1111 1111 1110 1111 1111 0111 1111 *
L1024 1111 1111 1111 1111 1111 1111 1111 0111 *
L1056 1111 1111 1011 1111 1111 1011 1011 1111 *
L1088 1111 1111 1111 1111 1111 0111 1010 1111 *
L1120 0111 1111 1111 1111 1101 1101 0111 1111 *
L1152 1111 1111 1111 1111 1110 1110 0111 1111 *
L1184 1011 1111 1111 1111 1110 1111 0111 1111 *
L1280 1111 1111 1111 1111 1111 1111 1111 0111 *
L1312 1111 1011 1111 1111 1111 1011 1011 1111 *
L1344 1111 1111 1111 1111 1111 0111 1011 1110 *
L1376 0111 1111 1111 1111 1111 1101 0111 1111 *
L1408 1011 1111 1111 1111 1111 1110 0111 1111 *
L1536 1111 1111 1111 1111 1111 1111 0111 1111 *
L1568 1111 1111 1111 1111 1110 1011 1111 1111 *
L1600 1111 1111 1111 1110 1111 0111 1111 1111 *
L1792 1111 1111 1111 1111 1111 1111 0111 1111 *
L1824 1111 1111 1111 1111 1111 1010 1111 1111 *
L1856 1111 1111 1110 1111 1111 0111 1111 1111 *
C8C36*
89A3
```

From: AMD PAL Handbook,
page 4-11

13.

```
A>thpe   ype sbc001.pal
P^L16L8                                   PAL DESIGN SPECIFICATION
  .001                                    BART ADDIS 7-16-85
 .ngle Board Computer address decoding logic
.rinceton University    MAE Microcomputer Lab   Princeton NJ 08544
L0000 11111111111111111111111111111111*
L0032 11110111101111111111111111111111*

L0256 11111111111111111111111111111111*
L0288 11110111101111111111110111111111*

L0512 11111111111111111111111111111111*
L0544 11111011101111111111110111111111*

L0768 11111111111111111111111111111111*
L0800 10111111111111111111111111111111*
L0832 11110111111111111111111111111111*
L0864 11111111101111111111111111111111*

L1024 11111111111111111111111111111111*
L1056 01111111111111111111111111111111*
L1088 11110111111111111111111111111111*
L1120 11111111101111111111111111111111*

L1280 11111111111111111111111111111111*
L1312 11101111111111111111111111111111*

1536 11111111111111111111111111111111*
 568 11111111111101111111111111111111*
```

```
                    AMS PAL PROGRAMMER VER 2.1


ENTER BASE IO ADDR :  0200
The selected manufacturer is
The selected pal type is

SPECIFY MANUFACTURER (M)        SPECIFY TYPE (T)
READ PAL             (R)
PROGRAM PAL          (P)        BLOW FUSE
VERIFY PAL           (V)
LOAD FILE            (L)
SHOW PAL             (S)
WRITE FILE           (W)
EXIT                 (E)
PATCH PAL            (B)
ENTER COMMAND  :   m
SPECIFY MANUFACTURER --
MMI - 1        NAT - 2          AMD - 3          TI - 4
2
The selected manufacturer is NAT
The selected pal type is

SPECIFY MANUFACTURER (M)        SPECIFY TYPE (T)
READ PAL             (R)
PROGRAM PAL          (P)        BLOW FUSE
VERIFY PAL           (V)
LOAD FILE            (L)
SHOW PAL             (S)
WRITE FILE           (W)
EXIT                 (E)
PATCH PAL            (B)
ENTER COMMAND  :   t

ENTER NATIONAL SEMICONDUCTOR PAL TYPE
10H8 - 01       12H6 - 02       14H4 - 03       16H2 - 04
16C1 - 05       10L8 - 06       12L6 - 07       14L4 - 08
16L2 - 09       16L8 - 0A       16R8 - 0B       16R6 - 0C
16R4 - 0D       16X4 - 0E       16A4 - 0F
0a
The selected manufacturer is NAT
The selected pal type is 16L8

SPECIFY MANUFACTURER (M)        SPECIFY TYPE
READ PAL             (R)
PROGRAM PAL          (P)        BLOW FUSE
VERIFY PAL           (V)
LOAD FILE            (L)
SHOW PAL             (S)
WRITE FILE           (W)
EXIT                 (E)
PATCH PAL            (B)
ENTER COMMAND  :   l
```

```
ENTER FILE NAME   :sbc001.pal
**********************************
*  NO OF BYTES READ  :  0400     *
*                                *
**********************************
The selected manufacturer is NAT
The selected pal type is 16L8

SPECIFY MANUFACTURER (M)        SPECIFY TYPE (T)
READ PAL             (R)
PROGRAM PAL          (P)        BLOW FUSE     (F)
VERIFY PAL           (V)
LOAD FILE            (L)
SHOW PAL             (S)
WRITE FILE           (W)
EXIT                 (E)
PATCH PAL            (B)
ENTER COMMAND  :   s

Enter product line number 0 to 63 (or 0 to 79):
    0 1111 1111 1111 1111 1111 1111 1111 1111
    1 1111 0111 0111 1111 1111 1111 1111 1111
    2 0000 0000 0000 0000 0000 0000 0000 0000
    3 0000 0000 0000 0000 0000 0000 0000 0000
    4 0000 0000 0000 0000 0000 0000 0000 0000
    5 0000 0000 0000 0000 0000 0000 0000 0000
    6 0000 0000 0000 0000 0000 0000 0000 0000
    7 0000 0000 0000 0000 0000 0000 0000 0000
    8 1111 1111 1111 1111 1111 1111 1111 1111
    9 1111 0111 1011 1111 1111 1101 1111 1111
   10 0000 0000 0000 0000 0000 0000 0000 0000
   11 0000 0000 0000 0000 0000 0000 0000 0000
   12 0000 0000 0000 0000 0000 0000 0000 0000
   13 0000 0000 0000 0000 0000 0000 0000 0000
   14 0000 0000 0000 0000 0000 0000 0000 0000
   15 0000 0000 0000 0000 0000 0000 0000 0000

PRESS " PG UP "," PG DOWN ",OR ANY OTHER KEY TO EXIT

d

   16 1111 1111 1111 1111 1111 1111 1111 1111
   17 1111 1011 1011 1111 1111 1101 1111 1111
   18 0000 0000 0000 0000 0000 0000 0000 0000
   19 0000 0000 0000 0000 0000 0000 0000 0000
   20 0000 0000 0000 0000 0000 0000 0000 0000
   21 0000 0000 0000 0000 0000 0000 0000 0000
   22 0000 0000 0000 0000 0000 0000 0000 0000
   23 0000 0000 0000 0000 0000 0000 0000 0000
   24 1111 1111 1111 1111 1111 1111 1111 1111
   25 1011 1111 1111 1111 1111 1111 1111 1111
   26 1111 0111 1111 1111 1111 1111 1111 1111
   27 1111 1111 1011 1111 1111 1111 1111 1111
   28 0000 0000 0000 0000 0000 0000 0000 0000
   29 0000 0000 0000 0000 0000 0000 0000 0000
```

```
   30 0000 0000 0000 0000 0000 0000 0000 0000
   31 0000 0000 0000 0000 0000 0000 0000 0000

PRESS " PG UP "," PG DOWN ",OR ANY OTHER KEY TO EXIT

Q

   32 1111 1111 1111 1111 1111 1111 1111 1111
   33 0111 1111 1111 1111 1111 1111 1111 1111
   34 1111 0111 1111 1111 1111 1111 1111 1111
   35 1111 1111 1011 1111 1111 1111 1111 1111
   36 0000 0000 0000 0000 0000 0000 0000 0000
   37 0000 0000 0000 0000 0000 0000 0000 0000
   38 0000 0000 0000 0000 0000 0000 0000 0000
   39 0000 0000 0000 0000 0000 0000 0000 0000
   40 1111 1111 1111 1111 1111 1111 1111 1111
   41 1110 1111 1111 1111 1111 1111 1111 1111
   42 0000 0000 0000 0000 0000 0000 0000 0000
   43 0000 0000 0000 0000 0000 0000 0000 0000
   44 0000 0000 0000 0000 0000 0000 0000 0000
   45 0000 0000 0000 0000 0000 0000 0000 0000
   46 0000 0000 0000 0000 0000 0000 0000 0000
   47 0000 0000 0000 0000 0000 0000 0000 0000

PRESS " PG UP "," PG DOWN ",OR ANY OTHER KEY TO EXIT

Q

   48 1111 1111 1111 1111 1111 1111 1111 1111
   49 1111 1111 1111 0111 1111 1111 1111 1111
   50 0000 0000 0000 0000 0000 0000 0000 0000
   51 0000 0000 0000 0000 0000 0000 0000 0000
   52 0000 0000 0000 0000 0000 0000 0000 0000
   53 0000 0000 0000 0000 0000 0000 0000 0000
   54 0000 0000 0000 0000 0000 0000 0000 0000
   55 0000 0000 0000 0000 0000 0000 0000 0000
   56 0000 0000 0000 0000 0000 0000 0000 0000
   57 0000 0000 0000 0000 0000 0000 0000 0000
   58 0000 0000 0000 0000 0000 0000 0000 0000
   59 0000 0000 0000 0000 0000 0000 0000 0000
   60 0000 0000 0000 0000 0000 0000 0000 0000
   61 0000 0000 0000 0000 0000 0000 0000 0000
   62 0000 0000 0000 0000 0000 0000 0000 0000
   63 0000 0000 0000 0000 0000 0000 0000 0000

  ************ END OF FILE ************

PRESS " PG UP "," PG DOWN ",OR ANY OTHER KEY TO EXIT


The selected manufacturer is NAT
The selected pal type is 16L8

SPECIFY MANUFACTURER (M)        SPECIFY TYPE (T)
READ PAL                (R)
```

```
PROGRAM PAL             (P)       BLOW FUSE     (F)
VERIFY PAL              (V)
LOAD FILE               (L)
SHOW PAL                (S)
WRITE FILE              (W)
EXIT                    (E)
PATCH PAL               (B)
ENTER COMMAND  :   p
Insert pal into socket and press any key to continue.
Pal programming is completed....now verifying...

Pal verifies okay.

Remove pal from socket and press any key to continue.

The selected manufacturer is NAT
The selected pal type is 16L8

SPECIFY MANUFACTURER (M)       SPECIFY TYPE
READ PAL             (R)
PROGRAM PAL          (P)       BLOW FUSE
VERIFY PAL           (V)
LOAD FILE            (L)
SHOW PAL             (S)
WRITE FILE           (W)
EXIT                 (E)
PATCH PAL            (B)
ENTER COMMAND  :   e
A>
```

```
A>
A>type sbc003.pal
   PAL16L8                                          PAL Design Specification
   sbc003                                           Bart Addis  7/23/85
Single Board Computer Address Decoding Circuitry
Princeton University     MAE Microprocessor Lab                Princeton NJ 08544
phi2     a13      a14   a15        pureset nc       nc       nc       nc       gnd
nc       nc       res   phi2star   via8000 viaA000 ram      til      eprom    vcc

/phi2star = /phi2

/res = pureset

/ram = /a14*/a15*phi2star

/til = a14*/a15*phi2star

/eprom = a14*a15

/viaA000 = /a13 + a14 + /a15

/via8000 = a13 + a14 + /a15
```

```
A)type sbc003.jed
AMPAL16L8                                          PAL Design Specification

 C003                                              Bart Addis  7/23/85

Single Board Computer Address Decoding Circuitry
Princeton University     MAE Microprocessor Lab              Princeton NJ 08544

L0000 11111111111111111111111111111111*
L0032 11110111011111111111111111111111*
L0256 11111111111111111111111111111111*
L0288 11111011101111111111110111111111*
L0512 11111111111111111111111111111111*
L0544 11110111101111111111110111111111*
L0768 11111111111111111111111111111111*
L0800 10111111111111111111111111111111*
L0832 11110111111111111111111111111111*
L0864 11111111101111111111111111111111*
L1024 11111111111111111111111111111111*
L1056 01111111111111111111111111111111*
L1088 11110111111111111111111111111111*
L1120 11111111101111111111111111111111*
L1280 11111111111111111111111111111111*
L1312 11101111111111111111111111111111*
L1536 11111111111111111111111111111111*
L1568 11111111111101111111111111111111*
```

A)type sbc003.map

Single Board Computer Address Decoding Circuitry

```
                 11 1111 1111 2222 2222 2233
   0123 4567 8901 2345 6789 0123 4567 8901

 0 ---- ---- ---- ---- ---- ---- ---- ----
 1 ---- X--- X--- ---- ---- ---- ---- ---- a14*a15
 2 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 3 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 4 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 5 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 6 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 7 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

 8 ---- ---- ---- ---- ---- ---- ---- ----
 9 ---- -X-- -X-- ---- ---- --X- ---- ---- /a14*/a15*phi2star
10 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
11 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
12 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
13 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
14 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
15 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

16 ---- ---- ---- ---- ---- ---- ---- ----
17 ---- X--- -X-- ---- ---- --X- ---- ---- a14*/a15*phi2star
18 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
19 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
20 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
21 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
22 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
23 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

24 ---- ---- ---- ---- ---- ---- ---- ----
25 -X-- ---- ---- ---- ---- ---- ---- ---- /a13
26 ---- X--- ---- ---- ---- ---- ---- ---- a14
27 ---- ---- -X-- ---- ---- ---- ---- ---- /a15
28 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
29 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
30 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
31 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

32 ---- ---- ---- ---- ---- ---- ---- ----
33 X--- ---- ---- ---- ---- ---- ---- ---- a13*a13
34 ---- X--- ---- ---- ---- ---- ---- ---- a14*a14
35 ---- ---- -X-- ---- ---- ---- ---- ---- /a15*/a15
36 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
37 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
38 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
39 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

40 ---- ---- ---- ---- ---- ---- ---- ----
 1 ---X ---- ---- ---- ---- ---- ---- ---- /phi2
42 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
43 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
44 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
45 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
46 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
```

```
48 ---- ---- ---- ---- ---- ---- ---- ----
49 ---- ---- ---- X--- ---- ---- ---- ---- pureset
50 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
51 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
52 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
53 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
54 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
55 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

56 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
57 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
58 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
59 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
60 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
61 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
62 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
63 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

LEGEND:  X : FUSE NOT BLOWN (L,N,0)   - : FUSE BLOWN   (H,P,1)
NUMBER OF FUSES BLOWN =  685
```