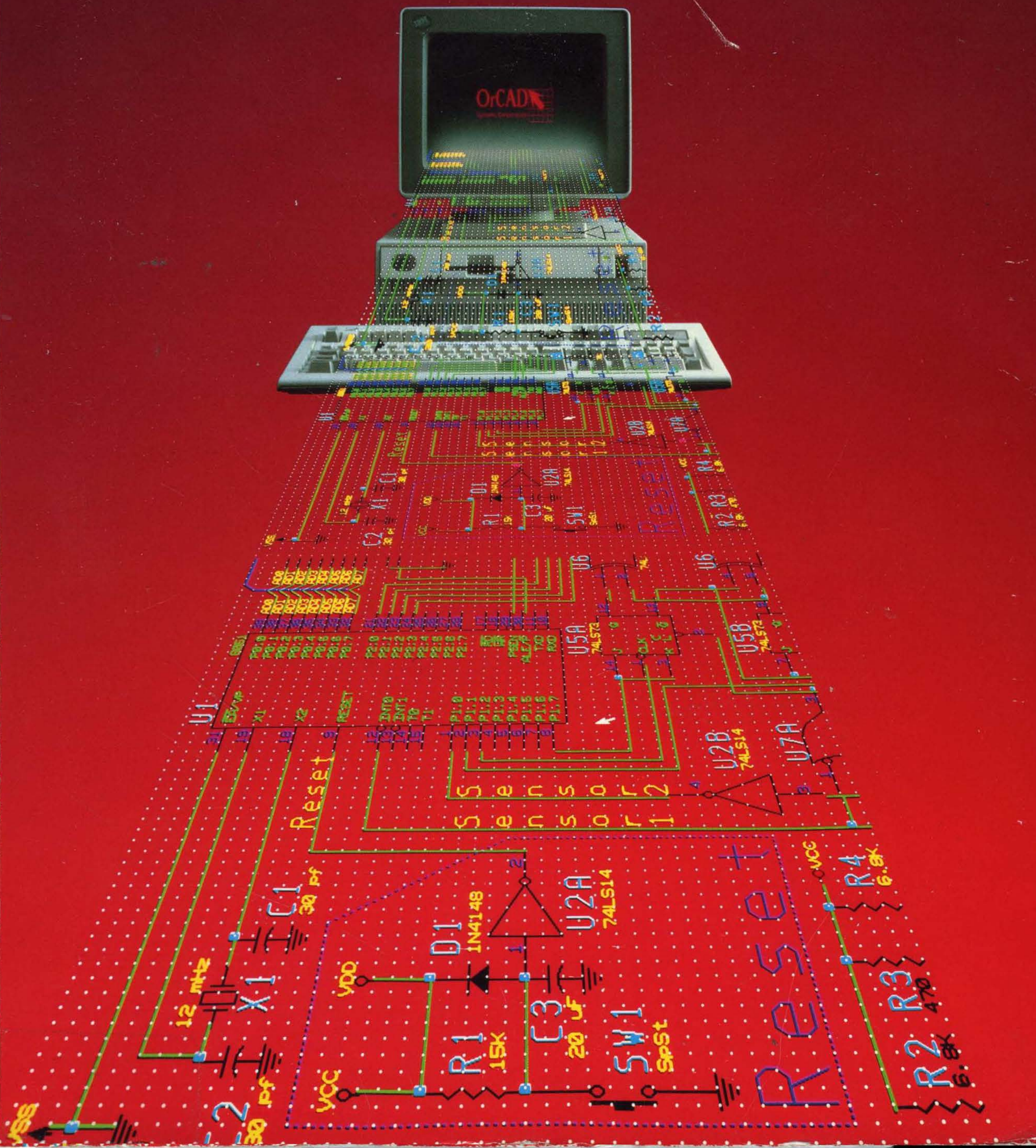
OrCAD/SDT III

V3.21



OrCAD™
Systems Corporation

VER 3.21

Schematic Design Tools

Document Number: OR8931A 8-22-89

OrCAD
3175 N.W. Aloclek Drive
Hillsboro, OR 97124-7135
(503) 690-9881

Sales & Administration	(503) 690-9881
Technical Support	(503) 690-9722
24-Hour Bulletin Board System	(503) 690-9791
FAX	(503) 690-9891

Technical Support
(503) 690-9722

Fax Number
(503) 690-9891

Sales & Administration
(503) 690-9881

Bulletin Board System
(503) 690-9791

- Eight part fields
- Over 3500 Unique library parts
- DeMorgan Equivalent parts
- Placement of wires, buses, connectors, labels, and junctions
- Real-time rubberbanding of wires and buses when objects are moved
- Part rotation and mirroring
- Moving, replicating, and deleting objects or blocks of objects
- Powerful step-and-repeat command
- Visible grid dots and angled bus entries
- Automatic panning of the worksheet
- Five zoom levels
- Over 100 user-assignable macros
- Unlimited levels of hierarchy
- On-Line part browsing and library directory
- PSpice analog simulation shell (PSpice not included)
- String searching
- Vertical text placement
- Suspension of session for DOS command execution
- Supports "A" through "E", and custom size worksheets

ANNOTATE:

This program scans a hierarchy or flat file and automatically updates all part reference designators (U?, R? etc., explained later). ANNOTATE also updates the pin numbers associated with the reference designators. ANNOTATE can handle very large, complex, and multiple worksheets.

BACKANNO:

The BACKANNO utility updates part reference designators in your design. The input to the program, a list of old and new reference designators, is used to update your schematic worksheets.

CLEANUP:

This utility checks the worksheet for wires, buses, junctions, labels, module ports, and other objects that are placed on top of each other. CLEANUP removes duplicate or overlapping wires, buses, junctions, and displays warning messages advising you of other duplicate objects.

COMPOSER:

If you choose to create library parts using a text editor, COMPOSER is the library utility that converts your custom library source files into the highly compressed library object files used by DRAFT. After COMPOSER compiles a source file into an object file, your custom library is ready for loading into DRAFT and the other utilities.

converts the graphical screen symbols to a data format readable by OrCAD/SDT III.

NETLIST:

This program generates a netlist of the worksheet signal and part connections in these formats: Algorex, Applicon (Leap and BRAVO!), Cadnetix, Calay, Computervision, EE Designer, EDIF, Flat EDIF, FutureNet (Pinlist and Netlist), Intergraph, Multiwire, PCAD, Spice, Racal-Redac, SALT, Scicards, Tango, Telesis, and Vectron formats. NETLIST also generates a general wire list.

PARTLIST:

This utility summarizes all the parts used in a schematic or group of schematic sheets. You can also merge user-specific information with the PARTLIST report.

PLOTALL:

PLOTALL plots a schematic or group of schematic sheets, including a hierarchy, flat file, or annotation file, in batch mode.

PRINTALL:

PRINTALL prints a schematic or group of schematic sheets, including a hierarchy, flat file, or annotation file, in batch mode.

TREELIST:

A program that scans a hierarchical organization of

Section 6:

Shows you how to use the utility programs included with OrCAD/SDT III. Each utility is discussed in separate sub-sections.

Section 7:

Introduces LIBEDIT, COMPOSER and DECOMP, and outlines the procedure for creating your own libraries. This section concludes with instructions for creating your own "custom" libraries.

Section 8:

A detailed tutorial discussing the use of the DRAFT command set and utilities. The tutorial creates a worksheet, where you will learn many ofthe concepts used in schematic capture.

Appendix:

Lists all of the library parts included with DRAFT. Outlines the configurations of the supported graphics boards, and lists the printer, plotter, and graphics board drivers supplied. Also included is the source code for the OrCAD printer and plotter drivers.

Index:

A complete Index to the manual.

1.3 TROUBLESHOOTING

If you have questions regarding how to run the program, be sure that you have properly configured

8. Configuration of your OrCAD/SDT III System (obtained from the main menu when you invoke "DRAFT /C")

The OrCAD Telephone Number is: **1-503-690-9722.** Technical Support hours of operation are Monday through Friday 8-4 Pacific Time.

NOTE

> The person who fills out and sends in the registration card is the only one entitled to customer support. The registration card must be received by OrCAD for continued telephone support. Your free telephone support will be extended to one year if you send in the warranty registration card within 90 days of product purchase.

At OrCAD, we are committed to giving our customers the finest products and support. Before your free support period expires, be sure you apply for the OrCAD Update Support Extension (OrCAD/USE). Contact OrCAD for information.

1.4 ORCAD BULLETIN BOARD LOGON PROCEDURES

OrCAD has available to registered users, a bulletin board service that operates 24 hours a day. If you have a 300, 1200, or 2400 baud modem, you can retrieve this information:

1.4.3 Logon Instructions

The first time that you logon to the bulletin board, you will not be allowed access to all of the system features. This is because you must "officially" register by entering your password and registration number. After you enter this information, you must leave a message, as a comment, for the bulletin board system operator (SYSOP) requesting that your security level be raised to that of "registered user". Allow 2 to 3 working days for the SYSOP to upgrade your security clearance.

Once your security level is changed to "registered user", the bulletin board is available for you to access all system features. Detailed logon and registration instructions are given below.

- DRIVER DISK 3

2. Disk File Reference Sheet - Located in the sealed package that contained the OrCAD supplied disks.

3. OrCAD Schematic Design Tools documentation.

You need to supply the following:

1. An IBM PC/XT/AT or compatible personal computer with two double-sided, double-density 360K floppy disk drives or a hard-disk system.

2. Or, and IBM PS/2 compatible computer with two 720K floppy disk drives or a hard disk (this is optional; you must have specifically requested this format when you ordered OrCAD/SDT III.

3. DOS Version 2.0 or greater.

4. A minimum of 512K bytes of PC memory (640K if you use LIBEDIT).

5. Seven formatted, double-sided disks for making backup copies of the OrCAD-supplied programs.

6. Five extra formatted, double-sided disks, if you are installing OrCAD/SDT III on a dual-floppy system.

2.2 PREPARING FOR INSTALLATION

Before you install OrCAD/SDT III on your system, you should be familiar with the DOS commands CHDIR (CD), COPY, DIR, DISKCOPY, FORMAT, and MKDIR (MD). If your system has a hard disk, you should also familiarize yourself with tree-structured directories. For a description of these commands and features, refer to your DOS user's manual.

For example, use five formatted, double-sided disks to make working copies of OrCAD/SDT III. These working copies are needed to load the drivers, libraries, and worksheet filenames. Since the files do not all fit on a single disk, separating them according to the procedure suggested below keeps disk swapping to a minimum.

Label five formatted floppy disks as follows:

DRAFT 1
DRAFT 2
DRIVER/LIBRARY
SHEET
LIBRARY UTILITY

Referring to the "Disk File Reference Sheet" to help you locate the desired OrCAD-supplied files, copy DRAFT.EXE and the ORCADSDT.OVL files from your OrCAD-supplied MASTER SOFTWARE DISKS to your working disk DRAFT 1. Executable files have the .EXE extension after the filename and overlay files use .OVL.

Also, copy the PDRAFT.OVL, PSPICE.BAT, and any desired macro files (with a .MAC extension), from the OrCAD-supplied MASTER SOFTWARE DISKS to the DRAFT 1 working disk.

Next, copy any desired utility programs from the OrCAD-supplied MASTER SOFTWARE DISKS to the DRAFT 1 working disk. Initially, we suggest that the ANNOTATE.EXE, CLEANUP.EXE, ERC.EXE, PARTLIST, and TREELIST utilities be copied.

```
Files Required                     Where Found
---------------------------------------------------
DRAFT.EXE       ---------------- MASTER SOFTWARE DISKS
ORCADSDT.OVL    ---------------- MASTER SOFTWARE DISKS
PDRAFT.OVL     ----------------- MASTER SOFTWARE DISKS
PSPICE.BAT     ----------------- MASTER SOFTWARE DISKS
Desired Macro File ----------- MASTER SOFTWARE DISKS
ANNOTATE.EXE    ---------------- MASTER SOFTWARE DISKS
CLEANUP.EXE     ---------------- MASTER SOFTWARE DISKS
ERC.EXE   -------------------- MASTER SOFTWARE DISKS
PARTLIST.EXE    ---------------- MASTER SOFTWARE DISKS
TREELIST.EXE    ---------------- MASTER SOFTWARE DISKS
```

Figure 2-1. DRAFT 1 Disk Files

```
Files Required                     Where Found
---------------------------------------------------
ORCADSDT.OVL ----------------- MASTER SOFTWARE DISKS
BACKANNO.EXE ----------------- MASTER SOFTWARE DISKS
CROSSREF.EXE ----------------- MASTER SOFTWARE DISKS
NETLIST.EXE  ----------------- MASTER SOFTWARE DISKS
PLOTALL.EXE  ----------------- MASTER SOFTWARE DISKS
PRINTALL.EXE ----------------- MASTER SOFTWARE DISKS
```

Figure 2-2. DRAFT 2 Disk Files

```
Files Required                     Where Found
---------------------------------------------------
PRINTER.DRV  ------------------------- DRIVER DISKS
Required Printer Driver -------------- DRIVER DISKS
Required Plotter Driver -------------- DRIVER DISKS
Required Graphics Card Driver -------- DRIVER DISKS
Library Files (.LIB extension)------- LIBRARY DISKS
```

Figure 2-3. Required DRIVER/LIBRARY Disk Files

Next, copy all of the files from each OrCAD-supplied DRIVER DISK to the DRIVER subdirectory, and all of the files from each OrCAD-supplied LIBRARY DISK the LIBRARY subdirectory.

Finally, the SHEET subdirectory is where you will put your schematic worksheet files. For now, the SHEET subdirectory remains empty.

Figure 2-5 shows the configuration tree we recommend for organizing your hard disk files. Follow STEPS 1 - 9 below to install OrCAD/SDT III on your hard disk system.

CD ORCAD <ENTER>

The DOS command CD enables you to change the current DOS directory to a new one you specify.

STEP 2:

Make three more subdirectories within ORCAD. Type:

MD LIBRARY <ENTER>

Then type:

MD DRIVER <ENTER>

Then type:

MD SHEET <ENTER>

STEP 3:

Insert the OrCAD-supplied MASTER SOFTWARE DISK 1 into drive A. Copy the files from MASTER SOFTWARE DISK 1 to the ORCAD directory. Type:

COPY A:*.* C: <ENTER>.

When the files are copied, remove the MASTER SOFTWARE DISK 1 from drive A.

STEP 4:

Insert the OrCAD-supplied MASTER SOFTWARE DISK 2 into drive A. Copy the files from MASTER SOFTWARE DISK 2 to the ORCAD directory. Type:

COPY A:*.* C: <ENTER>.

STEP 8:

Insert the OrCAD-supplied DRIVER DISK 2 into drive A. Copy the files from DRIVER DISK 2 to the DRIVER subdirectory. Type:

COPY A:*.* DRIVER*.* <ENTER>

The graphics drivers are now in the DRIVER subdirectory. Remove DRIVER DISK 2 from drive A.

STEP 9:

Insert the OrCAD-supplied DRIVER DISK 3 into drive A. Copy the files from DRIVER DISK 3 to the DRIVER subdirectory. Type:

COPY A:*.* DRIVER*.* <ENTER>

The graphics, printer, and plotter drivers are now in the DRIVER subdirectory. Remove DRIVER DISK 3 from drive A.

NOTE

If you are using PSpice, load the PSpice files and utilities in the OrCAD root directory (the directory containing DRAFT.EXE).

OrCAD/SDT III uses the MOUSE.COM driver for the Microsoft and Logitech mice, and the MSMOUSE.COM driver for the Mouse Systems Mouse. Many of these driver names use the /1 or /2 option switches for configuring them for use on the proper (COM1 or COM2) serial channel.

OrCAD/SDT III has no special mouse requirements. If you have mouse problems, refer to your mouse user's guide for information.

2.4 CONFIGURING ORCAD/SDT III

After you have installed the OrCAD/SDT III files on your system, you must configure the software to access the correct printer, plotter, graphic drivers, and libraries.

The configuration information that you supply will vary, depending on whether you are using a floppy or hard disk system. Examples are given below for each configuration case.

The procedures outlined in this section show you how to configure the following items:

Driver Prefix
Display Driver
Printer Driver
Plotter Driver
Library Prefix
Library Files
Worksheet Prefix
Macro File
Initial Macro
Macro Buffer Size
Hierarchy Buffer Size
Color Table / Pen Plotter Table
Template Table

```
            ::: CONFIGURATION OF OrCAD/SDT :::

DP - Driver Prefix
DD - Display Driver
PD - Printer Driver
PL - Plotter Driver
LP - Library Prefix
LF - Library Files






WP - Worksheet Prefix
MF - Macro Files
IM - Initial Macro              8192
MB - Macro Buffer Size          1024
HB - Hierarchy Buffer Size
CT - Color Table/Plotter Pen Table
TT - Template Table
U  - Update Configuration Information
Q  - Quit, Abondon to DOS
R  - Run Program

     Command?
```

Figure 2-6. The Configuration Menu

2.4.2 Driver Prefix

The driver prefix sets the subdirectory path or disk drive, enabling OrCAD/SDT III to load the printer, plotter, and graphics drivers.

To select driver prefix, type DP. A reverse video bar appears beside DP - Driver Prefix. Type the pathname of the directory that contains your device driver, then press <ENTER>.

2.4.5 Plotter Driver

The plotter driver configures OrCAD/SDT III with the plotter driver used with your plotter. You may also specify serial channel number, baud rate, parity, and word length.

To select Plotter Driver, type PL. A table appears that shows a list of the supported plotter drivers. Type the letter that corresponds to your plotter. If you are using a special driver, type S and enter the driver name.

After you have selected a plotter driver, type a colon (:) to configure OrCAD/SDT III for serial channel 1 or 2. Next, you may select the baud rate, parity, and word length your plotter needs for data transfer.

Then, type Q to quit and return to the configuration menu. If none of the drivers are to be used, type X.

NOTE

> Be sure that the serial channel, baud rate, parity, and word length match the characteristics of your plotter. Refer to your plotter manual if you need more information.

For additional plotter information, read the description of the PLOTALL utility in Section 6.

2.4.6 Library Prefix

Library Prefix sets the subdirectory path or disk drive that enables OrCAD/SDT III to load the part libraries.

Example:

```
::: Library Files :::
        DEVICE.LIB
        TTL.LIB
        MEMORY.LIB

        A - ADD File Name
        R - REMOVE File Name
        Q - Quit

Command?
```

The three part libraries listed above are loaded by OrCAD/SDT III and many of the utility programs. To remove a library, type R. The screen then looks as follows:

```
::: Library Files :::
        1 - DEVICE.LIB
        2 - TTL.LIB
        3 - MEMORY.LIB

        A - ADD File Name

Remove Which File Name? [1..3] Selection ->
```

Type the number that corresponds to the library you want to remove. Then, type Q to quit and return to the main configuration menu.

2.4.8 Worksheet Prefix

Worksheet Prefix sets the subdirectory path or disk drive specifier that enables OrCAD/SDT III to load the worksheet files.

NOTE

OrCAD supplies a couple of sample macro files called MACRO1.MAC and MACRO2.MAC, located on one of the MASTER SOFTWARE DISKS. Refer to the MACRO command in Section 4, for a description of this file.

2.4.10 Initial Macro

Initial Macro enables you to execute a specific macro automatically when DRAFT is invoked. For the Initial Macro to work, you must have previously loaded a macro file as outlined in Section 2.4.9 above.

To select Initial Macro, type IM. A reverse video bar appears beside IM - Initial Macro. Type the ASCII character name of the macro you want to run. Then, press <ENTER>.

For example, if you want to assign the F1 key to execute a macro, press the 'F' key, then the '1' key, followed by <ENTER>.

If you want to assign Ctrl-A to execute a macro, type ^ (Shift 6), then A. Then, press <ENTER>.

NOTE

To execute an initial macro, the <ENTER> key must be the first key you press when creating the macro. The macro must also be saved and loaded as a macro file as outlined in Section 2.4.9 above.

2.4.13 Color Table / Plotter Pen Table

Color Table / Plotter Pen Table enables you to modify the screen display and plotter colors for library parts, pin numbers and names, wires, buses, junctions, connectors, and other objects in the worksheet.

Type CT to display the color table / plotter pen table. Type M to display another screen of items that you can modify. Typing M once more returns you to the original screen. The table gives you the opportunity to modify the following parameters:

Color select:

Type C to change the color of an item. The prompt "Item ->" then appears. Type the number of the item you want to modify followed by an <ENTER>.

Then, the prompt "New Color ->" appears. From the color table right-hand column, enter the number of the desired color.

Pen select:

Type P to select a different pen. The prompt "Item ->" then appears. Type the number of the item that corresponds to the pen you want to modify.

The prompt "Enter Pen Number (0 = PAUSE, 1..16 = AUTO, 99 = IGNORE)" then appears. Type the new pen number followed by an <ENTER>.

If you enter 99, the plotter will not plot the object. If you enter 0 (zero), the plotter pauses, enabling you to change pens.

NOTE

If you notice that buses or object fills, have white spaces when plotting, it may be necessary to change this setting. Correct setting in some cases, may only be determined by experimentation.

Field Name select:

To change the name of Part Fields 1 through 8, type F. The prompt "Item ->" then appears. Type the number of the item whose name you want to modify followed by an <ENTER>.

The prompt "New Field Name ->" then appears. Type the new field name followed by an <ENTER>. This change will be reflected on the DRAFT Edit subcommand menu.

Type Q to quit and return to the main configuration menu.

Refer to "Four-Color Mode Configurations" in the Appendix for selecting colors on graphic boards in the four-color mode.

2.4.14 Template Table

Template Table enables you to modify the size of various fields in DRAFT schematics. This includes worksheet dimensions, text size, object sizes, borders, etc. The units are inches.

For example, with DRAFT, you can have five possible sheet sizes: A though E. With the Template Table, you can specify the horizontal and vertical dimensions of each sheet size.

CAUTION

> If you quit the program before updating the configuration information, all information is lost.

2.4.16 Returning to DOS with Quit

Quit enables you to return to DOS without updating the configuration information.

Type Q to quit the program and abandon to DOS. If any changes were made to the configuration menu, the program first asks if you want to update the configuration information before returning to DOS.

2.4.17 Run Program

Run program enables you to invoke DRAFT with the latest configured information that appears on the configuration menu.

Type R to invoke DRAFT or a utility program.

2.4.18 Dual-Floppy Configuration Example

Figure 2-7 below, shows OrCAD/SDT III configured for a dual-floppy system.

2.4.19 Sample Hard Disk Configuration

Figure 2-8 shows OrCAD/SDT III configured for a hard disk system.

```
            : : : CONFIGURATION OF OrCAD/SDT : : :

DP -  Driver Prefix           DRIVER\
DD -  Display Driver          EGA16E.DRV
PD -  Printer Driver          HPLASER4.DRV
PL -  Plotter Driver          HI.DRV
LP -  Library Prefix          LIBRARY\
LF -  Library Files
   -     CMOS.LIB
   -     DEVICE.LIB
   -     MOTO.LIB
   -     INTEL.LIB
   -     MEMORY.LIB
   -     ANALOG.LIB

WP -  Worksheet Prefix         SHEET\
MF -  Macro Files              MACRO1.MAC
IM -  Initial Macro            8192
MB -  Macro Buffer Size        1024
HB -  Hierarchy Buffer Size
CT -  Color Table/Plotter Pen Table
TT -  Template Table
U  -  Update Configuration Information
Q  -  Quit, Abondon to DOS
R  -  Run Program

      Command?
```

Figure 2-8. Sample Hard Disk Configuration

The driver, library, and worksheet prefixes have been configured as DRIVER\, LIBRARY\, and SHEET\ respectively. This corresponds to the like-named subdirectories you created within the ORCAD directory.

Type the following at the DOS prompt:

DRAFT <ENTER>

After DRAFT loads the drivers and libraries, a screen appears displaying the OrCAD logo. At the top of the screen, "Type any key to continue" appears. Type any key to display the Copyright Notice screen.

In a few seconds, "Type any key to continue" is repeated at the top of the screen. Typing any key displays: "Load File?" You now enter a filename, or begin a new worksheet as follows:

- To begin a new worksheet, press <ENTER> when the "Load File?" prompt is displayed

- To load a previously created worksheet, enter the filename and press <ENTER>

3.2 COMMAND AND MENU ORGANIZATION

DRAFT is an interactive, schematic capture program that uses pop-up command menus and prompts. The commands are further categorized into main command and subcommand menus. You can display both command menus on the screen by pressing <ENTER> or by using a mouse. Figure 3-1 illustrates the main command menu as it appears on the screen.

A subcommand menu is shown in Figure 3-2. Notice the word Place above the menu. This tells you that the Place command has been invoked, and you may select any of these subcommands: Wire, Bus, Junction, Entry (Bus), Label, Module Port, Power, Sheet, or Dashed line.

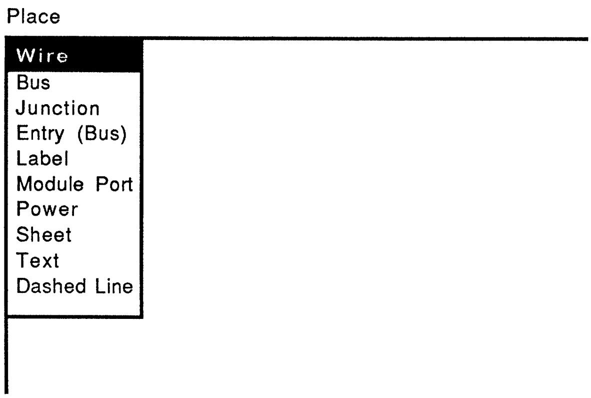


Figure 3-2. The Place subcommand Menu

Many of DRAFT's main commands have more than one level of subcommand. If you select the Wire subcommand (press W) from the Place subcommand menu for example, Figure 3-3 shows the subcommands that are displayed.

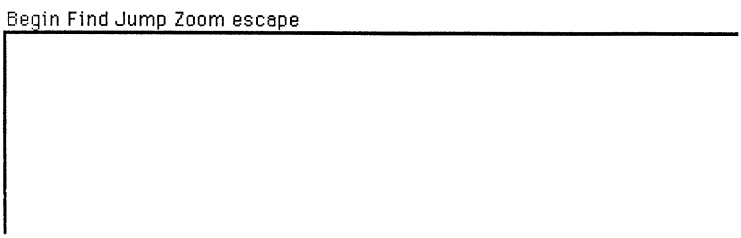


Figure 3-3. The Wire Subcommands

You execute one of these commands (shown on the prompt line) by pressing the first letter of the command

3.4.1 From DOS

From DOS you may load a worksheet when invoking DRAFT by typing:

DRAFT *filename* <ENTER>

Where *filename* is the filename of a worksheet to be loaded.

After DRAFT loads drivers and libraries, a screen appears displaying the OrCAD logo. At the top of the screen, the message "Type any key to continue" appears. Press any key to display the Copyright Notice screen.

In a few seconds, "type any key to continue" repeats at the top of the screen. When you type any key, the filename you specified on the DOS command line is loaded.

3.4.2 After Invoking DRAFT

You may load a worksheet after invoking DRAFT by typing:

DRAFT <ENTER>

After DRAFT loads drivers and libraries, a screen appears displaying the OrCAD logo. At the top of the screen, the message "Type in any key to continue" appears. Press any key to display the Copyright Notice screen.

In a few seconds, "type any key to continue" is repeated at the top of the screen. When you press any key, the message "Load File?" is displayed. You may now enter a filename, or begin a new worksheet as follows:

3.6 UPDATING FILES

The QUIT Update command updates the current version of the worksheet. If the current worksheet had been previously loaded from a file, that file is updated.

If the current worksheet is unnamed, DRAFT responds with "Write to File?" on the prompt line. Type in a filename and press <ENTER>.

3.7 EXITING DRAFT

To exit DRAFT and return to DOS without saving changes, select the "QUIT Abandon" command. If objects have been placed in the worksheet since the last update, DRAFT returns "Are you sure?" on the subcommand menu. Select [No] to abandon the subcommand. Select [Yes] to quit DRAFT without saving changes and return to the operating system.

3.8 PRINTING A FILE

To print a worksheet, use HARDCOPY. If you prefer to print from DOS, use the PRINTALL utility.

To print, select HARDCOPY. Be sure that the "HARDCOPY Destination" you select is "LPT", then choose the appropriate paper width using the HARDCOPY Width of Paper command. To print, select the HARDCOPY Make Hardcopy command.

Output resolution depends on which printer driver you have configured. For example, there are four HP LaserJet printer drivers available with OrCAD/SDT III (75, 100, 150, and 300 dpi). If you have the 150 dpi driver configured, then the output resolution is 150 dots per inch.

NOTE

> Version 5.00 of the Mouse Systems (IMSI) mouse driver has a bug that causes the cursor movement in the Y-axis to be opposite of what is expected. Prior versions and subsequent revisions do not have this problem. If you have Version 5.00 of the Mouse Systems driver, contact Mouse Systems for an update.

3.11 WORKSHEET FILE STRUCTURES

Design organization is an integral part of the schematic design process. Typically, a design moves through the refinement of general concepts to a final set of detailed schematic diagrams. With OrCAD/SDT III, schematic worksheets may be created using three different file structures. The file structures are:

- Flat File
- Hierarchy File
- One-Sheet

Typically, you will select the file structure for creating your schematic worksheet that best suits the complexity of your design. We recommended that you become familiar with all three file structures. The file structure you select determines how you use module ports (graphic objects used to represent signal connections between worksheets) and labels in the worksheet, and furthermore, how you invoke OrCAD/SDT III's utility programs.

Your introduction to worksheet file structures begins by examining a flat file structure. Discussions of the hierarchical and one-sheet file structures follow.

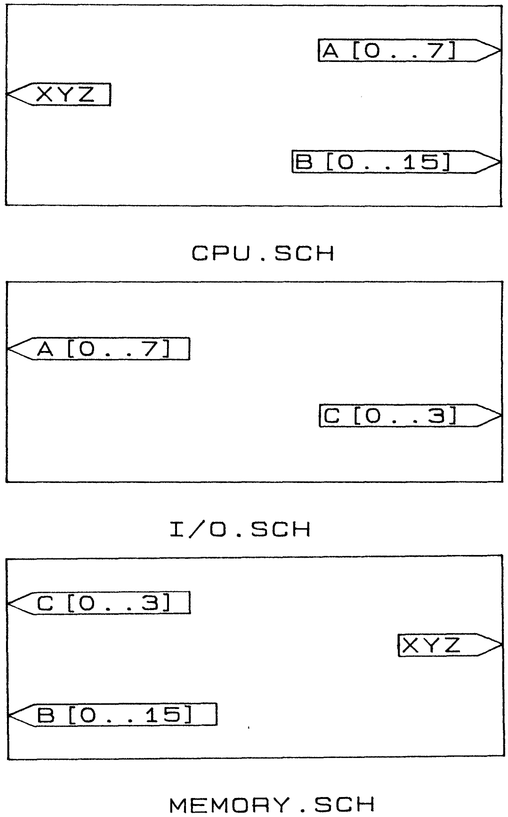


Figure 3-4. Flat File Structure

In this example, you see three separate worksheets with unique filenames: CPU.SCH, I/O.SCH, and MEMORY.SCH.

Signals (wires and buses) are connected one worksheet to another through identically named module ports. A[0..7] from the CPU.SCH worksheet is connected only to A[0..7] on the I/O.SCH worksheet. Likewise, the XYZ and B[0..15] module ports connect the CPU.SCH and MEMORY.SCH worksheets together. Finally, C[0..3] connects signal from the I/O.SCH and MEMORY.SCH worksheets together.

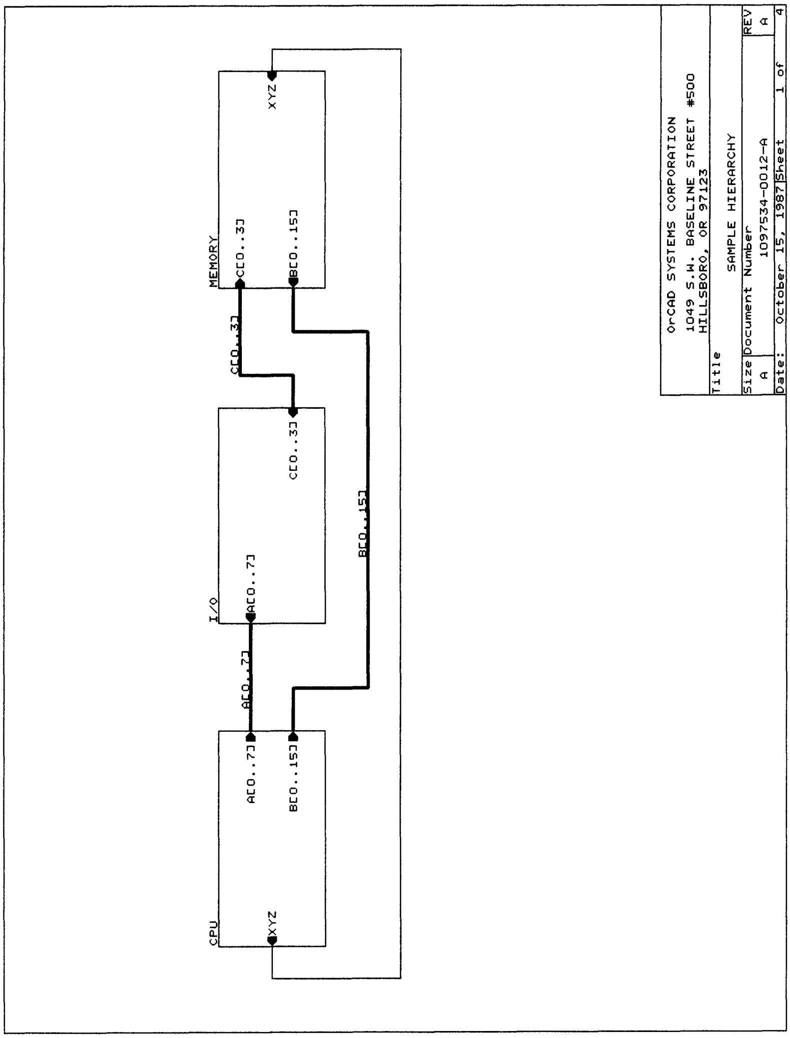


Figure 3-5. Hierarchy File Structure; COMPUTER.SCH

3.11.3 The One-Sheet File Structure

As the name implies, a one-sheet file structure is a design that is contained within one worksheet. Even though they are allowed, a one-sheet file structure design uses no module ports; since all signals remain within the worksheet. One sheet file structures may be handled as either flat or hierarchy file structures when you invoke the post-processing utility programs.

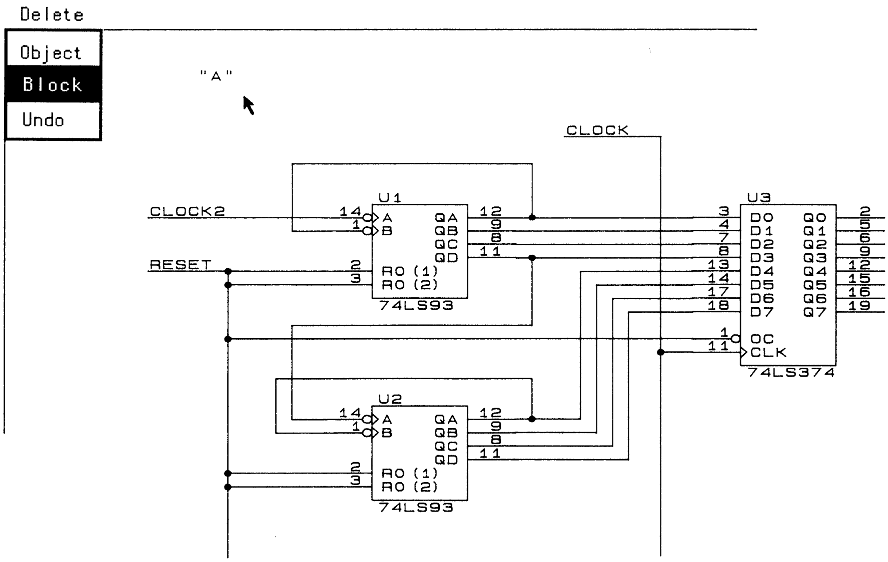


Figure 4-1. Placing the Cursor at a Begin Point

Select Begin. On the prompt line, DRAFT returns:

End Find Jump Zoom escape

As you move the cursor, a box is drawn that surrounds an area on the worksheet (refer to Figure 4-2 below).

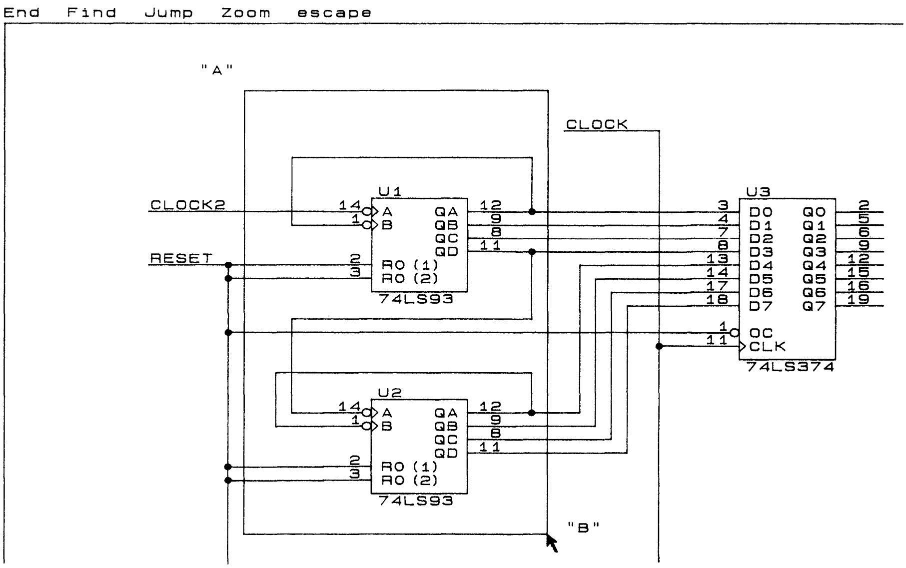


Figure 4-3. The Cursor Placed at the End Point

NOTE

> The shape that defines an area may be a point, horizontal line, vertical line, or rectangle.

Locating Objects

To locate an object or a specific area on a worksheet, use these subcommands:

Find:	To search for a string of characters. Refer to Section 4.6.

4.1 AGAIN

Again repeats the previously executed main level command. For example, if you previously selected PLACE, you may repeat PLACE with the AGAIN command.

remain at their original location. Only the outline symbols within the selected area move on the screen.

To place the moved objects in the new worksheet location, select Place. The screen is redrawn, placing the objects.

NOTE

> You may move and place a single object by positioning the cursor inside the object, then subsequently selecting Begin and End. It is not necessary to enclose the object in a box. Move and place the object as described above.

4.2.2 BLOCK Drag

BLOCK Drag moves objects while maintaining connectivity for rubberbanding wires and buses.

To drag a bus, you must enable the SET Drag Buses option (described in Section 4.15.3) to maintain bus connectivity.

BLOCK Drag executes the same as BLOCK Move.

4.2.3 BLOCK Fixup

BLOCK Fixup enables you to "fix up" wires and buses, making them orthogonal by adding new segments. When you select Fixup, DRAFT returns this subcommand menu:

Pick Find Jump Zoom escape

NOTES

If a node has more than one wire or bus connected, a menu enables you to select either Drag All or Pick One wire or bus.

Drag All enables you to drag all the wires or buses attached to a common node.

Pick One enables you to choose one wire or bus for fixup from those connected to a common node. When you select Pick One, DRAFT returns this subcommand menu:

Next
Previous
This

The wire or bus you select with Pick One is displayed in phantom (dashed lines). Next selects the next wire or bus for fixup.

Previous selects the previous wire or bus for fixup. Select This to fix up the wire or bus that you have selected (dashed lines).

Use Fixup for straightening non-orthogonal wires and buses by adding segments to them. Clean-up applications that do not need segments added should be done with BLOCK Drag.

Move the box to the desired worksheet location. Place the objects on the worksheet with the Place subcommand.

You may continue to place saved objects with BLOCK Get, or press <ESCAPE> to return to the main command menu.

4.2.6 BLOCK Import

With BLOCK Import you may retrieve objects stored in other files and place them in your current worksheet. To retrieve the contents of another file, select BLOCK Import. DRAFT returns:

File to Import?

Type the path and filename of the file you want to import and press <ENTER>. DRAFT returns:

Place Find Jump Zoom escape

Position the cursor on the worksheet where you want to place the contents of a file. Select Place to place the contents of the imported file on the worksheet. The imported objects are placed on the worksheet with the cursor in the upper left corner of the imported area.

4.2.7 BLOCK Export

With BLOCK Export you can save a defined worksheet area to a file. To export an object or worksheet area, define the area that you want to export (refer to Section 4.0, Defining an Area).

After you have defined the worksheet area, select BLOCK Export. Draft returns:

Export file name?

4.3 CONDITIONS

Conditions enables you to monitor your personal computer memory, and the memory available for these items:

- Worksheet
- Hierarchy buffer
- Macro buffer

Conditions has no subcommands. You may use the highlighted bar in the status menu as an aid in reading the table. To return to the main command level press either <ESCAPE> or <ENTER>.

What you see in the pop-up Conditions is described below.

4.3.1 Worksheet Memory Size

Shows worksheet memory size, in bytes, when Conditions was invoked. Tells you how much memory your worksheet uses, and about how much disk space will be needed to save it.

NOTE

> A worksheet uses memory space, even if it is blank, to hold frame and title block information.

4.4 DELETE

DELETE enables you to delete objects or blocks of objects. When you invoke DELETE, DRAFT returns a menu where you can select one of these subcommands:

- Object
- Block
- Undo

4.4.1 DELETE Object

As the name implies, you use DELETE Object to delete an object from the worksheet.

Select DELETE Object. On the prompt line, DRAFT returns:

Delete Find Jump Zoom escape

Place the cursor on the object you want to delete (you may use Find, Jump, and Zoom to help locate objects on the worksheet) and select the Delete subcommand.

If you want to delete one of two intersecting wires and you have placed the cursor at their intersection, the first wire drawn will be the first deleted. To delete the last wire drawn, move the cursor away from the intersection along the wire you want to delete and delete it.

If the cursor is pointing to more than one object, DRAFT returns:

Delete which Object?

4.5 EDIT

With the EDIT command, you can do the following:

1. Edit the title block, module ports, labels, power objects, sheet symbols, part reference designators, part names and fields.

2. Select different pinouts on devices with multiple parts per package.

3. Move a part reference designator and name to other locations.

4. Choose to make the part reference designator, the part value or both invisible.

When you invoke EDIT, DRAFT returns the menu:

Edit Find Jump Zoom escape

Move the cursor to what you want to edit and select Edit.

4.5.1 Editing Labels

To edit a label, place the cursor under the label name and select Edit. DRAFT returns the menu:

Name Orientation Larger Smaller

Select Name to edit the name of a label. When you select Name, the prompt "Name?" appears followed by the current label name. Use <RUBOUT> (the Backspace key) to erase it. Type the new name followed by an <ENTER>.

Select Orientation to change the orientation of the label to Horizontal or Vertical.

Select Name to edit the name of the power object. When you invoke Name, the name of the power object appears on the prompt line after "Power Name?" followed by the current power name. Use <RUBOUT> to erase the current name. Then type the new name followed by an <ENTER>.

Select Type to change the type of power object. The choices are Circle, Arrow, Bar, or Wave.

Select Orientation to change the orientation to top, bottom, left, or right.

For information about placing power objects on the worksheet, refer to 4.12.8, PLACE Power.

4.5.4 Editing Sheets

To edit hierarchical sheet symbols, place the cursor within the sheet symbol boundary and select Edit (use Find, Jump, or Zoom to locate the sheets you want to edit).

When you invoke Edit, cursor movement is restricted to the border of the sheet symbol. This helps you place the cursor at sheet net name locations. When Edit is invoked, DRAFT returns the menu:

> Add Delete Edit Name Filename Size Zoom escape

Use Add to add net connections between worksheets. To add a net name, place the cursor at the edge of the box where you want to place the net name Then, select Add.

DRAFT returns the prompt "Net Name?." Type the desired net name and press <ENTER>. A menu displays input, output, bidirectional, and unspecified net erase the old filename. Type a new file name and press <ENTER> to enter it on the worksheet.

Press <ESCAPE> to abandon any changes made to the filename.

Select Size to change the size of the worksheet displayed on the screen. DRAFT returns the menu:

End Jump Zoom escape

DRAFT places the cursor on the lower right corner of the worksheet. To change worksheet size, move the cursor until you reach the desired size, then select End.

For more information about worksheet sizes, refer to 4.12.8 Place Sheet, and Section 5, Hierarchy.

4.5.5 Editing Parts

Edit Parts enables you to edit and move part reference designators, values, and fields. It also enables you to select other packages on library parts with multiple parts per package and change the orientation of the symbol. Figure 4-4 illustrates a library part with its default reference designator and part value.

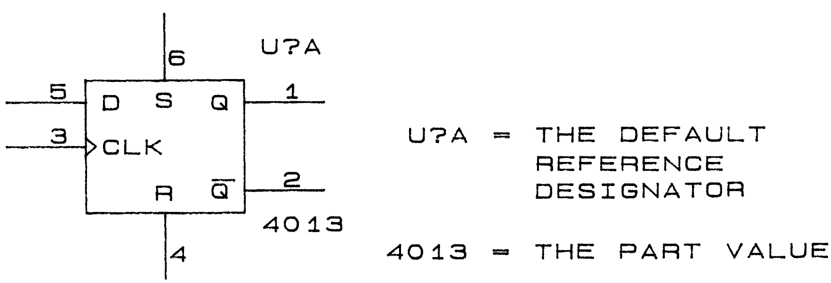


Figure 4-4. A CMOS 4013 Library Part with its Default Reference Designator and Part Value

Select Location to change the reference designator's location. DRAFT highlights the reference and part value designators and returns the menu:

Place Find Jump Zoom escape

You may move the reference designator anywhere in the worksheet using the cursor keys or mouse. Select Place to place the reference designator in the new worksheet location.

Refer to the utility Annotate (discussed in Section 6) for a method of automatically incrementing reference designators and changing the corresponding pin numbers of parts placed on the worksheet.

Select Visible to choose whether the reference designator appears on the screen or on hardcopy. When you select Visible, DRAFT lets you choose between [Yes] and [NO]. Yes makes the reference designator visible; No makes it invisible. The Visible command controls the visibility of reference designators at all Zoom scales.

Part Value

Select Part Value to edit or move the part values of components on the worksheet. Typical examples of part values are: 100K, 1N4004, .01 uf, 2N2222, and 80386.

When you select Part Value, DRAFT returns the menu:

Name Location Visibility

Select Name to edit the name of a part value. The prompt "Value?" appears. If the part already has a value assigned, it also appears on the prompt line. Press <RUBOUT> enough times to erase any old part

When you select a Part Fields, DRAFT returns the menu:

Name Location

Select Name to edit the name of a part field. The prompt "Part Field?" appears. If the part already has a name assigned, it also appears on the prompt line. Press <RUBOUT> enough times to erase any old part field. Then, type the new part field name on the prompt line and press <ENTER> to place it on the worksheet.

Select Location to change the part field's location. DRAFT highlights the designators and returns the menu:

Place Find Jump Zoom escape

You may move the part field anywhere in the worksheet using the cursor keys or mouse. Select Place to place the part field in the new worksheet location.

Orientation

To reposition a part, select Orientation. DRAFT returns the menu:

Rotate Convert Normal Up Over Down Mirror Zoom escape.

Rotate

The Rotate command rotates the part 90 degrees counterclockwise from its current position.

Convert

The Convert command is displayed when editing a part having another representation of the same part. For

Which Device

Which Device only appears when you edit a part that contains more than one part per package. An example would be a 74LS04 hex inverter. This part contains six inverters. You may select a different part in the package by selecting Which Device. Then, select the number that represents the device in the package you desire.

4.5.6 Editing the Title Block

To edit title block information, place the cursor inside the title block and select Edit. The title block is in the lower right worksheet corner. The title block information that you add or edit goes to a holding buffer, appearing in the title block on screen after you press <ESCAPE> to end the title block edit.

When you select Edit, DRAFT returns the menu:

Revision code
Title of sheet
Document number
Sheet number
Number of sheet
Organization name
1st Address Line
2nd Address Line
3rd Address Line
4th Address Line

Revision Code

Select Revision Code to add or edit the revision code. DRAFT returns the prompt "Revision Code?"

Type the desired revision number (three characters maximum). If you are editing an existing revision code, press <RUBOUT> enough times to erase the

Sheet Number

Select Sheet Number to add or edit a sheet number (any number up to 32767). DRAFT returns the prompt "Sheet Number?"

Type the sheet number. If you are editing an existing sheet number, press <RUBOUT> enough times to erase then current number. Then, type a new sheet number on the prompt line. Press <ENTER> to place the new sheet number in the holding buffer.

When you have finished editing the sheet number, press <ESCAPE> to update the title block.

Number of Sheet

Select Number of sheets to add to the number of worksheets (any number up to 32767). DRAFT returns the prompt "Number of Sheets?"

Type the desired number of sheets. If you are editing an existing number of sheets, press <RUBOUT> to erase the current number. Then, type the new number of sheets on the prompt line. Press <ENTER> to place the number of worksheets information in the holding buffer.

When you have finished editing the number of worksheets, press <ESCAPE> to update the title block.

Organization Name

Select Organization Name to add or edit an organization name (up to 44 characters). DRAFT returns the prompt "Organization Name?"

Type the desired name. If you are editing an existing organization name, press <RUBOUT> enough times

Revision History

You may want to add a revision history to the top of the title block. Use the PLACE Wire command to draw a revision history box and the PLACE Text command to add text within the box.

4.7 GET

Get retrieves objects from the part library database and places them in the worksheet as normal, rotated, or converted symbols. There are two ways objects can be retrieved from the library database.

1. Select Get. DRAFT returns "Get ?"

   Type the desired object name exactly as it appears in the part library directory. If the name typed does not match the library directory, the prompt line shows an error message. To verify the spelling of an object name, use the LIBRARY Directory command.

   Press <ENTER> to show the symbol outline on the screen. Refer to Section 4.7.2 for information on rotating and placing parts on the worksheet.

2. Select Get. DRAFT returns "Get ?"

   Press <ENTER>. DRAFT returns a subcommand menu displaying a list of part libraries. Select the library that you want to get a part from.

   When you select a library, a menu shows the selected library parts directory. Scroll the reverse video bar to the part name you select, then press <ENTER> to retrieve the part. The part outline symbol is placed on the screen.

   Refer to Section 4.7.2 for information on rotating and placing parts on the worksheet.

You can select TTL and other library part numbers created with a prefix and shorthand string (refer to Section 7, The Prefix Definition) from the library by

Jump Zoom escape Convert

Move the symbol to where you want to place it. You may use these subcommands to rotate or place the part in the worksheet.

Place

Select this subcommand to place the part in the worksheet.

Rotate

Selecting this subcommand rotates the part counterclockwise 90 degrees. Parts rotate in the sequence up, over, down, and normal (Figure 4-5 shows a rotated part).

Normal

Use this subcommand to rotate a part to its original position, as retrieved from the part library. This subcommand also returns parts that have been mirrored or converted to other shapes with Convert to their original position (Figure 4-5 shows a part placed in its normal position).

Up

Select this subcommand to rotate a part 90 degrees counterclockwise (equivalent to rotating it once from its normal position). Figure 4-5 shows a part placed in the up position.

Over

NOTE

When an outline symbol is placed over the same part already placed on the worksheet, the part seems to disappear. Moving the outline symbol will display the originally placed part.

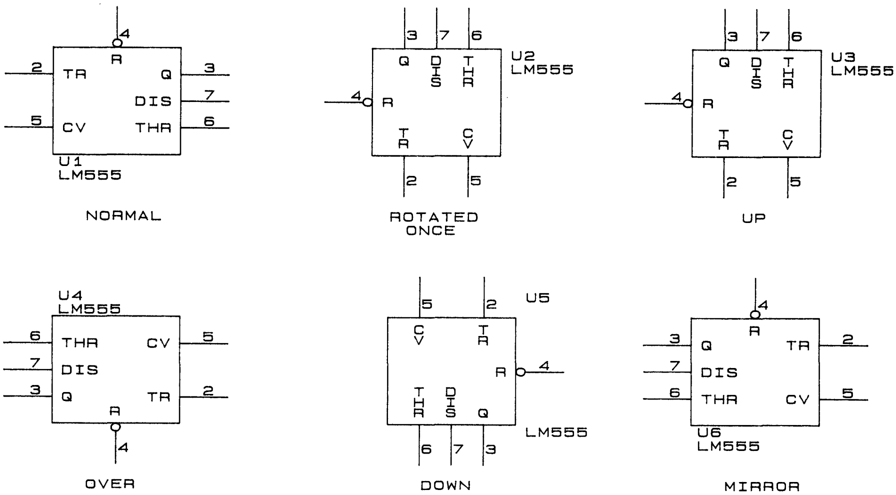


Figure 4-5. Parts Placed in the Normal, Rotated, Up, Over, Down, and Mirrored Positions

4.8.1 HARDCOPY Destination

You can send a worksheet either to a printer or to a file. This command selects the hardcopy destination. When you select Destination, DRAFT returns the menu:

LPT:
File

Select LPT: to send your worksheet to a printer.

Select File to send the worksheet to a binary file. When you select File, the prompt "Destination of Hardcopy?" appears followed by the default filename HARDCOPY.PRN.

To change the filename, press <RUBOUT> enough times to erase the default name. Then, type the new filename followed by an <ENTER>. You can specify a complete pathname including a drive specification. Remember that because the output file is a graphics file, it requires more disk space than the corresponding schematic file.

Files created this way may later be sent to the printer using the DOS COPY Command. For example, if you created a printer file called FRED.PRN, you can print it with the following DOS command:

```
COPY FRED.PRN prn: /b
```

Since the print file is a binary file, the DOS PRINT command will not work. For more information on COPY, refer to your DOS User's Manual.

To return to the main command level, press <ESCAPE>.

The Print Screen (PrtSc) key sends the worksheet contents that are displayed on the screen to a printer.

4.8.4 HARDCOPY Width of Paper

Width of Paper lets you choose between narrow and wide paper. To specify the width of the printer paper, select Width of Paper. DRAFT returns the menu:

Narrow
Wide

Select Narrow if you have narrow paper (8 inches wide). Select Wide if you have wide paper (13 inches wide),

To return to the main command level, press <ESCAPE>.

parameters, refer to the SET Grid Parameters command in Section 4.15.

To jump to a grid reference, follow these steps.

1 Select the Reference subcommand.

2. DRAFT returns "Jump to Reference". Select the desired Y-axis alpha grid reference (A, B, C, or D) from the menu.

3. Select the desired X-axis numeric grid reference (1 through 8) from the menu.

4. The cursor jumps to the grid reference location that you specified and you return to the main command level.

4.9.3 JUMP X Location

This subcommand moves the cursor a specific distance in the X- direction. Each incremental step represents 1/10th (0.1) inch on the worksheet if the SET Grid References Stay On Grid command is enabled; otherwise it is 1/100th (0.01) inch. The procedure for X-Location jumps is outlined below.

1. Select the X-Location subcommand.

2. DRAFT returns "Jump X". Enter the number of steps you want to jump A positive number moves the cursor to the right, and a negative number (-10, - 2.5, -30, etc.) to the left, the number of steps specified. If you enter 10 or +10, for example, the cursor jumps to the right 1 inch if you have enabled the SET Grid References Stay On Grid command.

4.10 LIBRARY

The Library command enables you to display library part list directories and view the parts in libraries that are configured with DRAFT. When the library command is invoked, DRAFT returns a selection menu:

Directory
Browse

4.10.1 LIBRARY Directory

The directory subcommand enables you to select a library and output its parts directory to screen, printer, or a file. When you select LIBRARY Directory, DRAFT returns a menu that displays a list of libraries that are currently configured in DRAFT. From the menu, select the library that you want a directory of.

The next screen menu enables you to choose the output device (Screen, Printer, or File).

Screen

This subcommand sends the library directory to the screen. Press any key to continue.

Printer

Select this subcommand to send the library directory to a printer.

File

Select this subcommand to output and save the library directory to a file. DRAFT returns "File?" on the

4.11 MACRO

The macro command enables you to capture, delete, initialize (erase), list, write to, and read macros from a file.

Schematic capture often involves repetitive tasks such as creating memory arrays, connecting wires and buses, or labeling items in the worksheet. Keystroke commands that are used to perform these tasks can be stored as macros, assigned to a key or key combination, then replayed by pressing the assigned key. This makes the entry of often-used commands less tedious.

DRAFT can record over 100 keyboard macros. These macros can be assigned to: function keys, selected keyboard keys, keys used with <CONTROL>, <SHIFT>, and <ALT>, or the middle button on a three-button mouse.

When you create macros or load them from a macro file they are stored in memory allocated as a macro buffer. The macro buffer defaults to 16,384 bytes of memory. If the buffer fills, the prompt line shows a warning message. To get more buffer memory you can either increase the buffer size (refer to Section 2.4.11) or delete unused macros.

To capture, delete, initialize, list, read, or write macros select the macro command from the main command level menu. DRAFT returns a selection menu of these subcommands:

Capture
Delete
Initialize
List
Read
Write

{^A}	{^B}	{^C}	{^D}	{^E}
{^F}	{^G}	{^I}	{^J}	{^K}
{^L}	{^N}	{^O}	{^P}	{^Q}
{^R}	{^S}	{^T}	{^U}	{^V}
{^W}	{^X}	{^Y}	{^Z}	
{^\}	{^]}	{^^}	{^_}	
{\0}	{\1}	{\2}	{\3}	{\4}
{\5}	{\6}	{\7}	{\8}	{\9}
{\-}	{\=}			
{\A}	{\B}	{\C}	{\D}	{\E}
{\F}	{\G}	{\H}	{\I}	{\J}
{\K}	{\L}	{\M}	{\N}	{\O}
{\P}	{\Q}	{\R}	{\S}	{\T}
{\U}	{\V}	{\W}	{\X}	{\Y}
{\Z}				
{F1}	{F2}	{F3}	{F4}	{F5}
{F6}	{F7}	{F8}	{F9}	{F10}
{^F1}	{^F2}	{^F3}	{^F4}	{^F5}
{^F6}	{^F7}	{^F8}	{^F9}	{^F10}
{SHIFT-F1}	{SHIFT-F2}	{SHIFT-F3}	{SHIFT-F4}	{SHIFT-F5}
{SHIFT-F6}	{SHIFT-F7}	{SHIFT-F8}	{SHIFT-F9}	{SHIFT-F10}
{\F1}	{\F2}	{\F3}	{\F4}	{\F5}
{\F6}	{\F7}	{\F8}	{\F9}	{\F10}
{BACK TAB}	{DEL}	{INS}	{^RIGHT}	{^LEFT}
{END}	{HOME}	{PGDN}	{PGUP}	{^PGDN}
{^PGUP}	{D}	{L}	{R}	{U}
{MMB}	{MACROBREAK}			

Figure 4-6. Valid Macro Key Names

4.11.2 MACRO Capture

To create a macro, select the capture subcommand. DRAFT returns "Capture macro?"

Press the key or keys you want to use for the macro label. The key(s) that you pressed appears on the prompt line. Refer to Figure 4-6 above, for a list of valid keys that can be assigned as macros.

Press <ENTER>. DRAFT returns "<macro>", informing you that you are in the macro capture mode.

g. Select the screen subcommand.

h. The directory is now displayed on the screen. Press <ENTER> twice to continue.

i. Press <M> to leave the macro capture mode.

To execute the macro press <F1>.

2. Using the PLACE command, this macro places junctions on the worksheet at the cursor location. We'll assign Ctrl A as the macro key.

   a. Select the macro command from the main command menu.

   b. Select the capture subcommand.

   c. Press the <Ctrl A> keys at the "Capture macro?" prompt, then press <ENTER>. You are now in the capture macro mode.

   d. Press <P> for the PLACE command.

   e. Press <J> for the junction subcommand.

   f. Press <P> to place the junction in the worksheet.

   g. Press <ESCAPE> to return to the main command menu.

   h. Press <M> to leave the macro capture mode.

   To execute the macro press <Ctrl A>.

Let's assign the macro we create to F3. We want to scroll through the Intel library directory, retrieve a part, move it to a worksheet location, place it, and return to the main command menu.

Enter the following:

a. Press <M> for macro.

b. Press <C> for capture macro.

c. Press <F3>, then press <ENTER>.

d. Press <G> GET, then press <ENTER>.

e. A window appears listing the libraries. Place the highlighted bar over the Intel library, then press <ENTER>.

f. Press <CTRL> <HOME> simultaneously, then press <ENTER>.

g. Press <CTRL> <HOME> again simultaneously, then press <ENTER>.

h. Press <ENTER> again.

i. Press <ESCAPE> (this returns you to the main command level).

j. Press <M> to close the macro.

When you press F3, the macro runs and stops with the Intel library window showing on the screen. You select the desired part, then press <ENTER>. The macro continues, then stops, displaying the "Place, Rotate..." menu. You may move the part around the worksheet and place it by pressing <ENTER>. The part must be placed using the <ENTER> key, not by typing <P>. This is because, a macro that has paused is waiting for

Select "No" to return to the main command menu, or "Yes" to erase all macros.

4.11.9 MACRO List

This subcommand shows a list of all the key names assigned to macros. To display the macro list select the list subcommand.

4.11.10 MACRO Write

This subcommand enables you to save all macros currently in DRAFT to a file. To save the macros to a file select write. DRAFT returns "Write all macros to?"

Type the path and filename and press <ENTER> to write the macros to the file.

Macro files may be automatically loaded each time DRAFT is invoked by placing the macro file name in the "Configuration of OrCAD/SDT III" menu (refer to Section 2.4.9).

4.11.11 MACRO Read

The Read subcommand enables you to load a macro file into DRAFT. To load a macro file select the Read subcommand. DRAFT returns "Read all macros from?"

Enter the path and file name that the macros are stored in, then press <ENTER> to load the macro file.

4.11.12 Macro File Format

A macro format is a simple ASCII file, and may be edited or created using a text editor. If you want to macro, and you want to nest the macro assigned to F2 within the new macro, press F2 at the appropriate time while you are creating the new macro.

To nest a macro in a file, insert the macro key, enclosed by curly brackets, inside the text of another macro. For example:

```
{F3}=g2115{F2}{}
```

4.11.14 OrCAD Supplied Macros

Included on the MASTER SOFTWARE DISK 1, is a sample macro file called MACRO.MAC. If you want to use these macros, be sure to install the MACRO.MAC file in the ORCAD root directory. Then, configure DRAFT to load this file as outlined in Section 2.4.9. For a complete description of each macro, refer to Appendix C.

Begin

Repetitively entering begin while the wire is being drawn enables you to redefine the wire's origin where it makes a 90 degree turn.

To continue drawing the wire from the 90 degrees turn, select begin where the turn starts (point A in Figure 4-8). You may also move to the end of the wire (point B in Figure 4-8) and select either Begin, End, or New, to fill it in. Dashes show a wire that has not been filled in by using the Begin, End, or New subcommands.

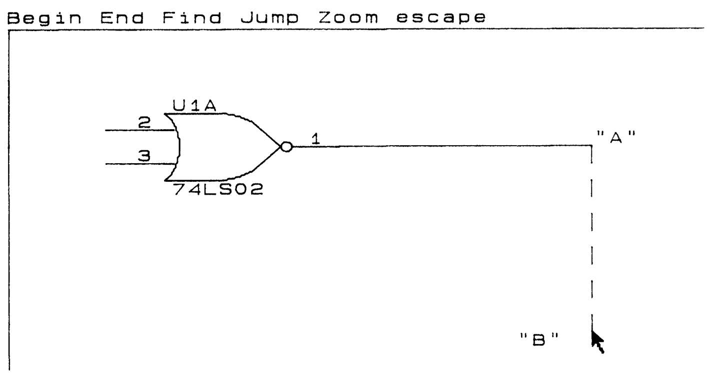


Figure 4-8. Drawing a Wire

Continue drawing the wire until you come to where you want it to end. To connect the wire to an end point, select either the End or New subcommands.

End

With the cursor at the end point select end. When you invoke this command, the program returns you to the main command menu.

Drawing a bus is identical to drawing a wire (refer to Section 4.12.1, PLACE Wire command).

NOTE

> If you are using the NETLIST utility program, you must label each bus with a label in the following format:
>
> BUSNAME[0..n]
>
> Where n is the decimal number of the last bus member, and BUSNAME is the name of the bus. Refer to Section 6, NETLIST, for more information.

4.12.3 PLACE Junction

On a worksheet, many wires and buses connect or cross each other. Junctions are placed on the worksheet to distinguish a connection from a cross-over. If you intend to have more than two wires or buses connect to a common node, always place a junction at that point. This tells the ERC and NETLIST utility programs that the node is a physical connection.

If you don't place a junction at an intersection of wires or buses, ERC and NETLIST interpret the intersection as a cross-over.

In many designs, you may want to connect a wire at 90 degrees to a bus. If you do, you must place a junction at the connect point. Junctions are not required if you use a bus entry (refer to Section 4.12.4).

To place a junction in the worksheet, select the junction subcommand. DRAFT returns:

To place a bus entry, select the place subcommand. Select the / or \ subcommands to change the bus entry angle. DRAFT shows the last bus entry angle when you invoke the Place Entry (Bus) command.

Select the Wire subcommand to place wire thickness entries. Use this subcommand when a wire is to exit or enter a bus from another object.

Select the Bus subcommand to place bus thickness entries. Use this subcommand when a bus makes a turn or is joined to another bus.

NOTE

> Junctions are not required to be placed in the worksheet to connect an angled bus entry to a bus.

4.12.5 PLACE Label

A label is an identifier placed on a worksheet that can connect signals (wires and buses) together without actually physically connecting them. You can place labels horizontally or vertically on a worksheet.

To place a label, select the Label command. DRAFT returns the prompt: "Label?" Type the name of the label followed by an <ENTER>. DRAFT then returns the menu:

> Place Orientation Value Larger Smaller Find Jump Zoom escape

Select Place to place the label on the worksheet. Before placing the label, you can make it horizontal or vertical

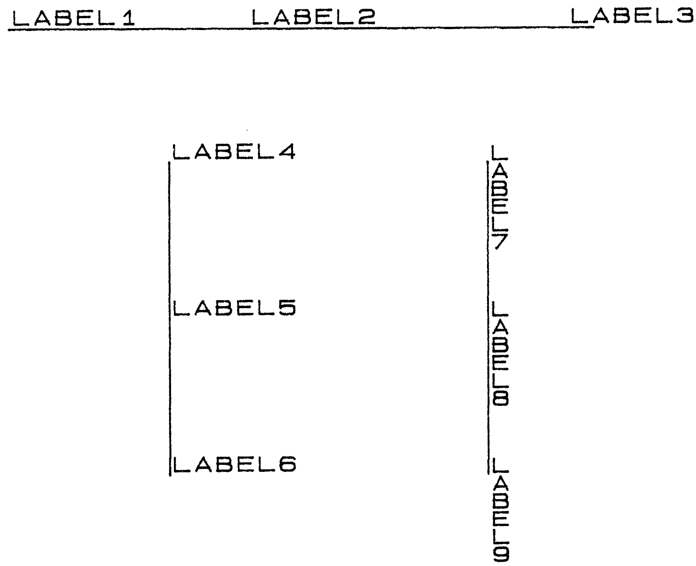


Figure 4-10. Correct Label Positions

Figure 4-11 shows incorrect label positions. None of the label hotpoints are next to the wire. LABEL21 for example, has its hotpoint (lower portion of the character "L") away from the wire.

Select Input if the module port is used as a signal input, Output if the module port is used as a signal output, Bidirectional if the module port is used as a bidirectional signal, or Unspecified if the module port is used to transfer power or "don't care" signals.

NOTE

> An unspecified module port must be used if power is transferred between worksheets. For additional information, refer to NETLIST in Section 6.

When you select one of the above commands, DRAFT returns the module port name to the screen. You may move it where you want it before placement. The following menu appears:

> Place Value Type Style Find Jump Zoom escape

Select Place to place the module port on the worksheet. Before placing the label, you can change its Value, Type, or Style.

A module port's value is whatever you typed after "Module Port Name?". Its type is input, output, bidirectional, or unspecified. Its style is how it looks on the screen. If you select Style, you can choose from the following:

> Right pointing
> Left pointing
> Both pointing
> Neither pointing

4.12.8 PLACE Power

Select the PLACE Power command to place power supply objects on the worksheet.

To place a power object on a worksheet, select Power. DRAFT returns the menu:

> Place Orientation Value Type Find Jump Zoom escape

The power object appears on the screen ready to be positioned and placed on the worksheet. Before placing the power object on the worksheet, you can change its orientation, value or type.

If you select Orientation, you can choose Top, Bottom, Left, or Right.

A power object's value is the text (for example, VCC) associated with it. If you select Value, DRAFT returns the prompt "Power Value? xxx" where xxx represents the current value. You can backspace over the current value or append to it. After typing the new value (for example, +5 , GND, + 5 VDC, -12 VDC, VSS, VEE, or any other text string), press <ENTER> to return to the menu.

A power object's type is its appearance on the screen. If you select Type, you can choose Circle, Arrow, Bar, or Wave.

Select Place to place the power pin where you want it on the worksheet. Then, press <ESCAPE> to return to the main command menu.

worksheet in a hierarchy, contains net names used to interconnect the present worksheet and the sheet represented by the symbol.

To place a sheet symbol select Sheet. Draft returns the menu:

Begin Find Jump Zoom escape

Select Begin, outline the area, then select End to finish it. DRAFT then returns the menu:

Add Delete Edit Name filename Size Zoom escape

Cursor movement is restricted to within the box walls of the hierarchical sheet. This enables you to place the cursor where you want the sheet names and nets.

Add

Add is used to add net connections between worksheets. To add a net sheet connection, place the cursor at the edge of the box where you want the name and select Add. DRAFT returns the prompt "Net Name?" Type the net name followed by an <ENTER>. DRAFT then returns the menu:

Input Output Bidirectional Unspecified

Select the appropriate type for your sheet connection.

Delete

To delete sheet connections, place the cursor at the connection's location and select Delete.

If you want to rename the file, select Filename and press <RUBOUT> enough times to erase the current name. Then, type the new name followed by an <ENTER>.

Size

Size increases or decreases the sheet size. When you select Size, DRAFT returns the menu:

End Jump Zoom escape

The cursor is automatically positioned on the lower right corner of the sheet. To change sheet size, move the cursor until you reach the desired size. Then, select End.

For more information, refer to Section 4.5.4, Editing Sheets, and Section 5.0, Hierarchy.

4.12.10 PLACE Text

Text allows you to place comments on your worksheet. Comments are useful for placing revision history, tolerance, and other information in the worksheet.

When you select Text, DRAFT returns the prompt "Text?". Type the text that you want as a comment and follow it with an <ENTER>. The following menu then appears:

Place Orientation Value Larger Smaller Find Jump Zoom escape

Before placing the comment on the worksheet, you can change its orientation, value, and size. If you choose Orientation, you can choose between horizontal and vertical. If you choose Value, you have the opportunity to edit the text before you place it. If you choose Larger,

4.13 QUIT

With the Quit command, you can enter and leave hierarchical worksheets, load, update, and write to files, clear the worksheet, suspend to DOS, abandon edits, and invoke PSpice and its options. When Quit is invoked, DRAFT returns the following menu:

Enter Sheet
Leave Sheet
Update File
Write to File
Initialize
Suspend to DOS
Abandon Edits
PSpice
Probe
Parts

4.13.1 QUIT Enter Sheet

The Enter Sheet command enables you to enter a hierarchical worksheet. If you are working on a worksheet, and want to enter another worksheet, select Enter Sheet.

If you have made any changes to the current worksheet, DRAFT returns "Abandon changes made?"

This message tells you that you will lose any changes made to the worksheet during the current work session. Select No to abandon the Enter Sheet command. If you select Yes, all changes made to the worksheet are lost, and DRAFT returns the menu:

4.13.4 QUIT Write File

The **Write File** enables you to save the current worksheet to any file youspecify. When invoked, DRAFT returns "Write to File?"

Type the desired path and file name followed by an <ENTER>. The worksheet is saved to the file specified, and DRAFT returns the QUIT menu.

Press <ESCAPE> to return to the main command level.

4.13.5 QUIT Initialize

Initialize enables you to either load a worksheet file or erase everything from it, thus clearing it. To perform these tasks select QUIT Initialize and follow one of the two procedures outlined below.

If there are objects on the worksheet, DRAFT returns "Are you sure? Yes or No". Select No to abandon the Initialize command and return to the main command level. Select Yes to clear the worksheet.

With a clear worksheet, DRAFT returns "Load File?" Type the filename that you wish to load followed by an <ENTER>. If the filename exists, the worksheet is loaded and displayed. If the filename does not exist, DRAFT returns <<<new worksheet>>>.

Press <ESCAPE> to return to the main command level.

4.13.6 QUIT Suspend to DOS

Suspend to DOS enables you to temporarily leave DRAFT and the worksheet, save the worksheet in memory, and return to DOS. Once you have suspended

This batch file expects the existence of two other files. One is "filename.CIR" and the other is "filename.NET", where filename is the name of your schematic file minus any file extension. You must create the .CIR file; the .NET file is created by DRAFT when you select PSpice.

filename.CIR

This file contains the specifications of the parts used in your schematic as well as PSpice control statements.

filename.NET

This file contains the connectivity information specific to your schematic that's needed by PSpice.

When you exit PSpice, you return to DRAFT. For information on the requirements of the .CIR file, refer to the PSpice manual.

4.13.9 QUIT Probe

Probe is a PSpice utility that aids the display of PSpice data. When you select Probe, DRAFT returns the prompt "Probe? PROBE.DAT." To implement this command, you must have purchased the PSpice analog simulator and have it installed in the ORCAD root directory.

PROBE.DAT is the name of the file made by PSpice. It contains the data describing the result of the PSpice simulation. If you have renamed this file and want to invoke Probe on the renamed file, backspace over PROBE.DAT and type the new filename followed by an <ENTER>. If you accept the name PROBE.DAT, just press <ENTER>.

4.14 REPEAT

When the **Repeat** command is invoked, DRAFT repeats the last entered object or label placed on the worksheet. Repeats are defined in Section 4.15.13, SET Repeat Parameters.

Example:

> You want to repeat and auto-increment a bus member label, a unit of one in the Y direction, at 1/10 inch steps. You want to place the labels in the range of A0 through A7. You set the following parameters in the SET repeat parameters command of Section 4.15.13 below.
>
> X Repeat Parameters = +0
>
> Y Repeat Parameters = +1
>
> Label Repeat Delta = +1
>
> Auto Increment Place = Not used in this example

Procedure:

1. Use the PLACE Label command and enter "A0" at the "Label?" prompt.
2. Select bus member as the label type.
3. Place the cursor where the label goes on the worksheet.
4. Place the label.
5. Press <ESCAPE> to return to the main command menu.
6. Press <R> to repeat.

4.15 SET

With the SET command, you can enable or disable the following DRAFT options:

- Auto panning
- Backup files
- Dragging buses when rubberbanding
- The error bell
- Having the left mouse button release execute <ENTER>
- Macro prompts
- Drawing non-orthogonal wires
- Showing pin numbers
- Disabling the standard title block
- Displaying cursor coordinates, grid dots, and grid references
- Enabling the cursor to move off-grid
- Setting repeat parameters
- The worksheet size, A through E

To change the status of an option, select SET. DRAFT returns the following menu:

NOTE

> Disabling Backup File can be dangerous. If your file should accidentally become damaged or erased, you will be unable to recover it.

4.15.3 SET Drag Buses

Drag Buses allows buses to be rubberbanded when you select a BLOCK Drag command. Because there are more points to locate when rubberbanding, system performance decreases when the BLOCK Drag command is executed.

When you select Drag Buses, DRAFT lets you choose between Yes and NO. Yes enables the rubberbanding of buses; No disables it.

4.15.4 SET Error Bell

Error Bell gives you the ability to enable or disable the error bell (your computer's speaker). When you enable this option, error messages and errors sound the speaker.

When you select Error Bell, DRAFT lets you choose between Yes and NO. Yes enables the error bell; No disables it.

4.15.5 SET Left Button

When Left Button is enabled, releasing the left mouse button executes the <ENTER> command for command line menus only. Pressing the left mouse button continues to execute the command that the video bar highlights.

When you select Auto Pan, DRAFT lets you choose between Yes and NO. Yes enables the display of pin numbers; No disables it.

4.15.9 SET Title Block

When Title Block is enabled, the standard title block is placed on the worksheet. With the option disabled, you may create a custom title block using the PLACE Wire/Bus and PLACE Text commands.

When you select Title Block, DRAFT lets you choose between Yes and NO. Yes enables the display of the standard title block; No disables it.

4.15.10 SET Worksheet Size

Worksheet Size enables you to select the worksheet size, A through E.

When you select Title Block, DRAFT lets you choose a letter from A throughE. The size corresponding to the letter is set in the Template Table during configuration.

4.15.11 SET X,Y Display

When X,Y Display is enabled, the upper right part of the prompt line shows the cursor coordinates. The worksheet origin (0,0) is the upper left corner. Coordinates do not appear on the screen until the cursor is moved.

When you select X,Y Display, DRAFT lets you choose between Yes and NO. Yes enables the display of the cursor coordinates; No disables it.

CAUTION

> Placing objects, wires, and buses with Stay on Grid disabledmay cause errors when using ERC and NETLIST. This is because wires and buses may look as if they are connected,when in fact, they are not. ERC and NETLIST may interpretthese connections as opens. OrCAD recommends that you do not place objects, wires, or buses in the worksheet with this parameter disabled.

Visible Grid Dots

When Visible Grid Dots is enabled, visible grid dots are displayed on the worksheet spaced 1/10 XY unit on Zoom scale 1, 2/10 XY unit on Zoom scale 2, 1/2 XY unit on Zoom scale 5, 1 XY unit on Zoom scale 10, and 2 XY units on Zoom scale 20.

When you select Visible Grid Dots, DRAFT lets you choose between Yes and NO. Yes enables the display of grid dots; No disables it.

4.15.13 SET Repeat Parameters

Repeat Parameters has four subcommands that are used to set the REPEAT command parameters. When you select Repeat Parameters, DRAFT returns the following menu:

> X Repeat Step
> Y Repeat Step
> Label Repeat Delta
> Auto Increment Place

X Repeat Step

Auto Increment Place

When Auto Increment Place is enabled, label names are automatically incremented or decremented when they are entered on the worksheet with PLACE Label Place. After a label has been placed, the numeric suffix of that label is changed by the amount specified by the Label Repeat Delta command.

When you select Auto Increment Place, DRAFT lets you choose between Yes and NO. Yes enables the automatic incrementing or decrementing of labels; No disables it.

4.15.14 SET Visible Lettering

You can choose to have some items on appear visible in Zoom scale 2. When you select Visible Lettering, the following menu appears:

Part FieldPin Number
Pin Name
Text
Module Port
Power Value
Sheet Name
Sheet Net
Title Block

To make an item visible, select it and choose Yes. To make it invisible, select it and chose No. Note that this option only affects how parts appears on the screen in Zoom scale 2.

For more information on the JUMP command, refer to Section 4.9, The Jump Command

The number in parentheses shows the current zoom scale (1-20).

4.17.2 ZOOM In

The ZOOM in subcommand enables you to zoom in on the worksheet for a more detailed view. The number in parentheses shows you what the zoom scale will be the next time you invoke ZOOM in.

4.17.3 ZOOM Out

The ZOOM out subcommand enables you to zoom out to display a larger worksheet area. The number in parentheses shows you what the zoom scale will be the next time you invoke ZOOM out.

4.17.4 ZOOM Select

The Select subcommand enables you to select any one of five zoom scales from a pop-up subcommand menu. When invoked, you can select zoom scale 1, 2, 5, 10, or 20 from the menu.

The introduction to hierarchical file structures begins by introducing you to the elements of hierarchical design. This follows with a discussion of the two types of hierarchical designs: simple and complex. Finally, you will learn how to create a simple and a complex hierarchy.

5.1 ELEMENTS OF A HIERARCHY

In a hierarchical structure, there is one "root" worksheet. This is the first level in the hierarchy. By using the PLACE Sheet command, individual blocks are placed in the root worksheet. These blocks are known as "sheet symbols" and they represent separate unique worksheets. To display each sheet symbol as a worksheet, just place the cursor inside the sheet symbol boundary and execute the QUIT Enter command. This places you one level down in the hierarchy.

When you enter a sheet symbol, the worksheet that represents that sheet symbol appears on the screen. If other sheet symbols are placed in the worksheet and you continue to enter them, another worksheet will appear on the screen. This hierarchical procedure can continue to a depth of over 200 levels with OrCAD/SDT III.

Connections between sheet symbols are made through "nets" that are placed along the left and right sides of the sheet symbol. When a sheet symbol is placed, nets are added to a worksheet using the "Add" subcommand. As in a flat file structure, module ports are used in a hierarchy to label signals that leave a worksheet. In a hierarchy, "nets" in the sheet symbols make "implied" connections to similarly named module ports that are placed in the actual worksheet.

For example, the nets, named XYZ, A[0..7], B[0..15], and C[0..3] in Figure 5-1, connect the CPU, I/O, and

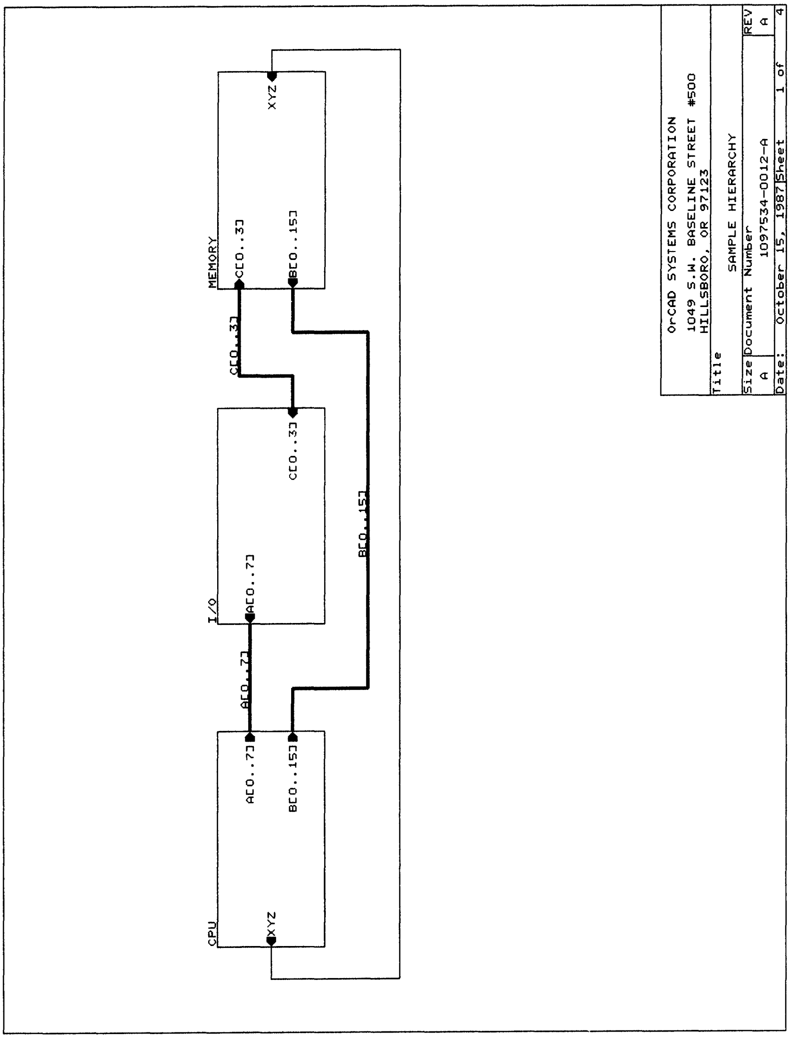


Figure 5-1. A Root Worksheet with Three Sheet Symbols

Listed below are many of the terms and definitions that are used in creating a hierarchical design. As in a flat file structure, it is important to follow these procedures.

represents the decimal number of the last member of the bus; only a zero (0) is valid in the first portion of the suffix ([2..n] for example, is not valid. Between the prefix and the suffix there must be no space.

Typical examples are:

ADDR[0..31]	(this bus has 32 members)
DATA[0..15]	(this bus has 16 members)
CONTROL[0..3]	(this bus has 4 members)

Labels must also be used to label the individual members, or signals, that come from a bus. As a rule, you may place any number of labels on a bus signal. When labels are placed on signals that come from a bus, they must be labeled in a form that corresponds to the bus they come from. This form is:

BUSNAMEx

Where BUSNAME is the same prefix name as the bus label, x is a decimal number in the range of [0 - n] taken from the suffix of the bus label, and n represents the decimal number of the last member of the bus. Between BUSNAME and x there must be no space.

Typical examples are:

ADDR0	(this would be used to label a signal that connects to the ADDR[0..31] bus above)

above, the sheet symbol named I/O has two nets: A[0..7] and C[0..3]. In this example, these nets connect to buses, which in turn connect to other sheet symbols. When a sheet symbol is placed, nets are added to a worksheet using the "Add" subcommand. It is important to realize, these nets are NOT module ports. Module ports are never placed in sheet symbols.

In a hierarchy, nets make "implied" connections to their respective module ports. For example, in Figure 5-1, the sheet symbol named I/O has two nets: A[0..7] and C[0..3]. If you enter the sheet symbol (using the QUIT Enter command) to obtain the I/O worksheet, there must also be two module ports named: A[0..7] and C[0..3].

To summarize, a net placed on a sheet symbol must have the same name as the module port that connects to it in the corresponding worksheet.

QUIT ENTER SHEET

The QUIT Enter Sheet command enables you to enter a hierarchical sheet symbol to obtain the corresponding worksheet. To enter a sheet symbol, select the QUIT Enter Sheet command from the main menu. Place the cursor inside the sheet symbol that you want to enter. Then, select the Enter subcommand.

QUIT LEAVE SHEET

The QUIT Leave Sheet command enables you to leave a worksheet and return one level up in the hierarchy to the sheet symbol. To leave a worksheet, select the QUIT Leave Sheet command from the main command level, then select the Leave subcommand.

Delete

To delete sheet connections, place the cursor at the connection's location and select Delete.

Edit

To edit a sheet connection, place the cursor at the connection's location and select Edit. DRAFT then presents a menu that allows you to choose between Name and Type.

To edit the net name, select Name. When the net name appears on the prompt line, press <RUBOUT> to erase it. Then, type the new net name and press <ENTER>.

To edit the net type, select Type. You can then chose Input, Output, Bidirectional, or Unspecified.

Name

The Name command allows you to edit the sheet name. The default sheet name is a question mark (?) located at the top of the sheet. Typical sheet names may be "Memory Array" or "Dynamic RAM Refresh circuitry".

To specify a sheet name, select Name. DRAFT returns "Sheet Name?" followed by the current sheet name. Press <RUBOUT> enough times to erase the current name. Then, type the desired sheet name. Press <ENTER> to place it at the top of the sheet.

Filename

Filename enables you to name the file representing the hierarchical worksheet. DRAFT automatically generates a filename based on the date and time of day circuitry may be placed into a worksheet, a sheet symbol is drawn to represent that worksheet, and that sheet symbol is placed in multiple instances within other worksheets. Like the simple hierarchy, a complex uses nets and module ports to connect signals from one worksheet to another.

5.4 EXAMPLE OF A SIMPLE HIERARCHY

In this Section, we will describe an example of a simple hierarchical design, discuss labeling, module ports, nets, sheet symbols, and other aspects of the design, and review examples on executing some of the utility programs.

The example schematic discussed is a three sheet simple hierarchy. It is defined as a simple hierarchy because none of the sheet symbols are replicated and used in multiple instances. The root sheet is illustrated in Figure 5-2 below, which has the title: CMOS CPU DESIGN.

When you create a hierarchy, the first step is to create the first sheet, or "root". This sheet is created like any other schematic. For this example, the root sheet is given the file name: CMOSCPU.SCH. As Figure 5-2 shows, the name of the worksheet is: CMOS CPU DESIGN, and appears in the title block.

Inside of the root sheet is the design's circuitry. In this case, the 80C51, 82C82, and discrete components. Also placed in the root sheet, are two sheet symbols: POWER SUPPLY and CMOS MEMORY.

These sheet symbols were placed in the worksheet using the PLACE Sheet command. The CMOS MEMORY sheet symbol contains the worksheet in which the systems memory is located. The POWER SUPPLY sheet symbol contains the worksheet in which the systems power supply is located.

When you create a sheet symbol, DRAFT automatically assigns it a file name based on the date and time of day. In this example, we have modified the default file names and assigned the CMOS MEMORY sheet symbol the file name: MEMORY.SCH. The POWER SUPPLY sheet symbol is assigned the file name: POWER.SCH.

The CMOS MEMORY sheet symbol contains four nets: A[0..7], WE, BACKUP, and AD[0..7]. These nets were placed into the sheet symbol using the PLACE sheet subcommand called "Add". These nets are NOT module ports. Connected to the A[0..7] and AD[0..7] nets, are buses that have labels placed on them with the same name as the net that they connect to. As mentioned earlier and in the NETLIST utility description in Section 6, this labeling procedure is mandatory for buses. Every bus must have a label placed on it, in the correct format. Connected to the WE net is a wire that goes to the PSEN signal on the 80C51. Last, a net named BACKUP is connected to a

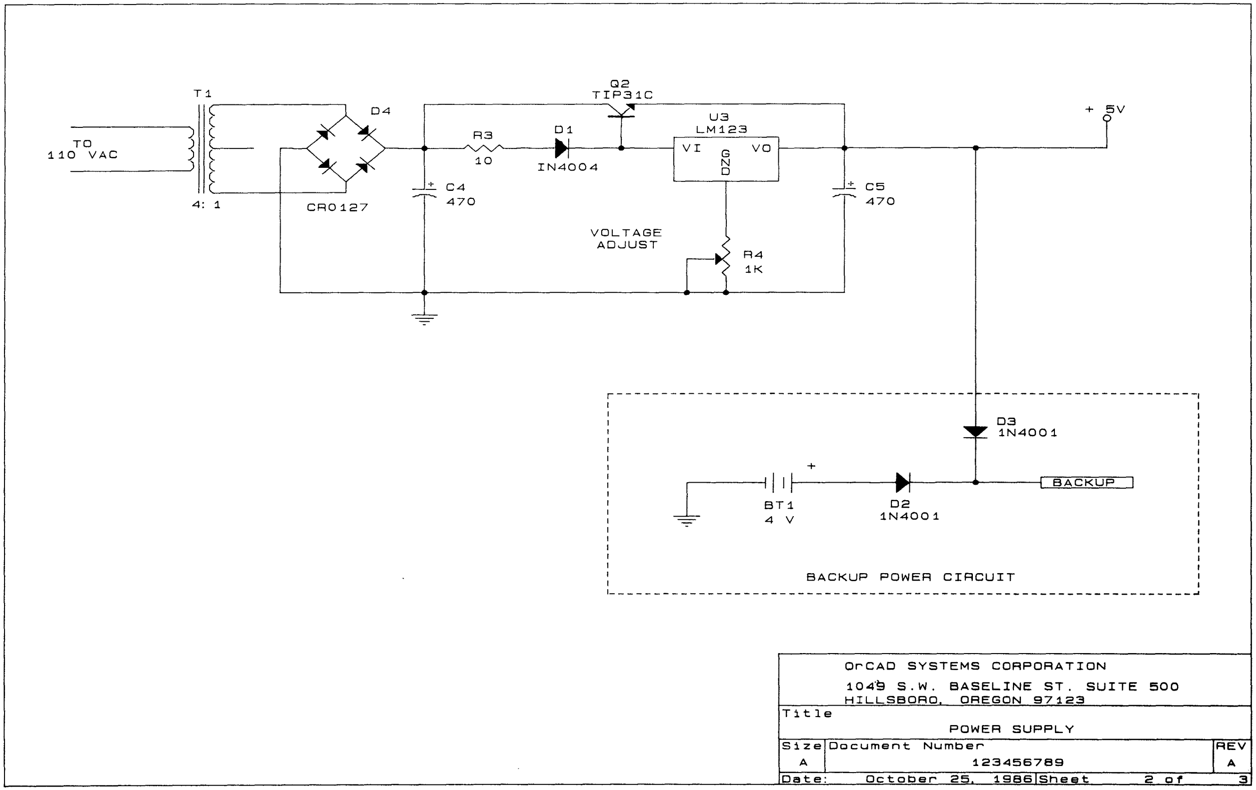


Figure 5-3. POWER SUPPLY Worksheet

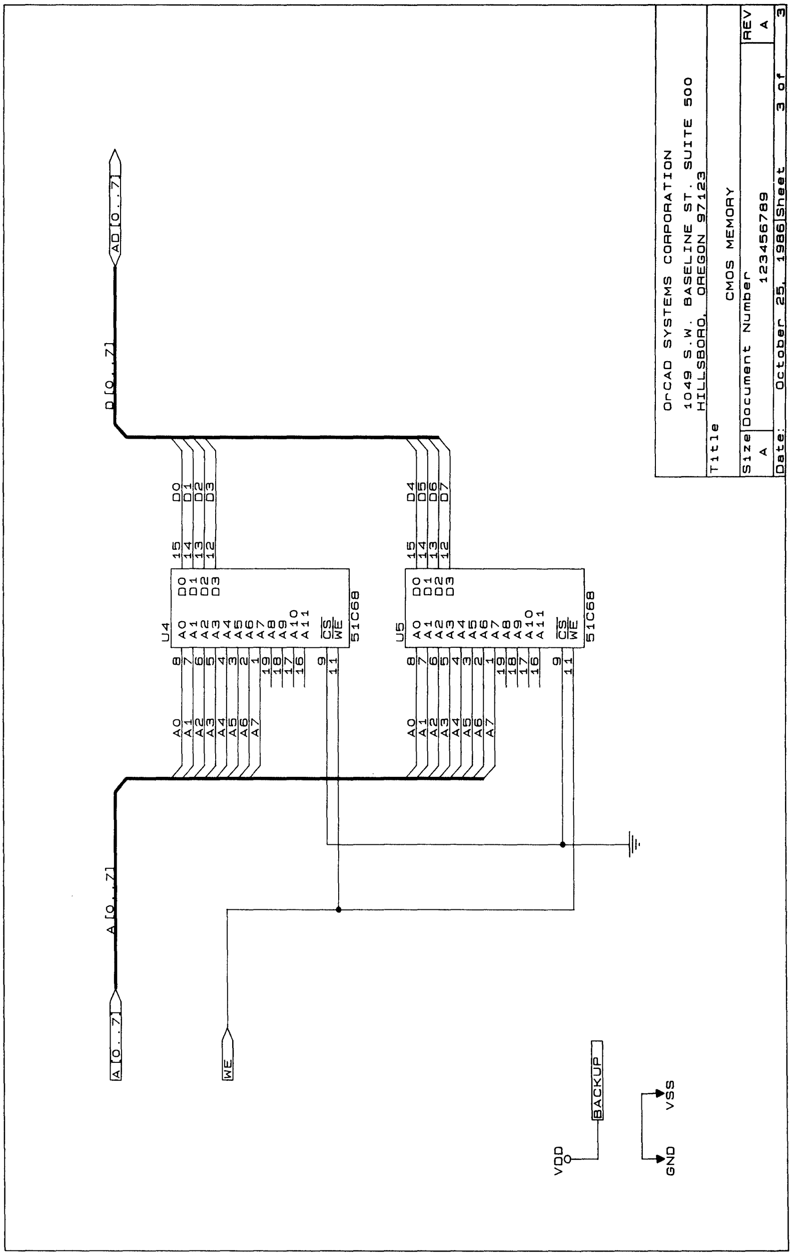


Figure 5-4. CMOS MEMORY Worksheet

PARTLIST utility programs. But first, we will review the conditions for designing a hierarchy and outline other helpful suggestions.

Helpful Design Suggestions:

1. Thoroughly read the discussion on the NETLIST utility located in Section 6.

2. Place Labels in the correct format on all buses (refer to previous discussions).

3. Place Labels in the correct format on all signals that connect to a bus (refer to previous discussions).

4. Place module ports in the correct format, on all signals that go off the worksheet (refer to previous discussions).

5. Do not put a blank space in any label or module port name.

6. When placing sheet symbols, use "nets", not module ports to connect to other sheet symbols.

7. Do not overlap wires or buses with other wires, buses, or object pins.

5.4.1 Invoking The ANNOTATE Utility On CMOSCPU.SCH

After the design has been completed, it should be run through ANNOTATE utility program (this assumes that you have not manually annotated the worksheet). To annotate the simple CMOSCPU.SCH hierarchy, enter the following from the DOS command line:

```
ANNOTATE CMOSCPU.SCH /M <ENTER>
```

ERC CMOSCPU.SCH ERROR.TXT /U <ENTER>

ERC is the name of the utility, CMOSCPU.SCH is the name of the root worksheet of the hierarchy, ERROR.TXT is the name of a text file to send the error information (this is useful for examining the results of the utility), the /U generates a unconnected pin report, <ENTER> refers to pressing the ENTER key

After the utility program has completed, all errors are sent to the ERROR.TXT file. To examine the file, use a text editor or word processing software. Examining the ERROR.TXT file results in the following:

```
"cmoscpu.sch"
UNCONNECTED REPORT
 X=  4.50  Y=  1.90 I/O              U1,P2.0
 X=  4.50  Y=  2.00 I/O              U1,P2.1
 X=  2.60  Y=  2.10 I/O              U1,INT0
 X=  4.50  Y=  2.10 I/O              U1,P2.2
 X=  2.60  Y=  2.20 I/O              U1,INT1
 X=  4.50  Y=  2.20 I/O              U1,P2.3
 X=  2.60  Y=  2.30 I/O              U1,T0
 X=  4.50  Y=  2.30 I/O              U1,P2.4
 X=  2.60  Y=  2.40 I/O              U1,T1
 X=  4.50  Y=  2.40 I/O              U1,P2.5
 X=  4.50  Y=  2.50 I/O              U1,P2.6
 X=  2.60  Y=  2.60 I/O              U1,P1.0
 X=  4.50  Y=  2.60 I/O              U1,P2.7
 X=  2.60  Y=  2.70 I/O              U1,P1.1
 X=  2.60  Y=  2.80 I/O              U1,P1.2
 X=  4.50  Y=  2.80 I/O              U1,RD
 X=  2.60  Y=  2.90 I/O              U1,P1.3
 X=  4.50  Y=  2.90 I/O              U1,WR
 X=  2.60  Y=  3.00 I/O              U1,P1.4
 X=  2.60  Y=  3.10 I/O              U1,P1.5
 X=  2.60  Y=  3.20 I/O              U1,P1.6
 X=  4.50  Y=  3.20 I/O              U1,TXD
 X=  2.60  Y=  3.30 I/O              U1,P1.7
 X=  4.50  Y=  3.30 I/O              U1,RXD

WARNING - POWER Supplies are CONNECTED GND <-> VSS
WARNING - POWER Supplies are CONNECTED VDD <-> + 5V

"POWER.SCH"
UNCONNECTED REPORT
 X=  1.10  Y=  1.10 Passive          T1,AA
```

5.4.4 Invoking The TREELIST Utility On CMOSCPU.SCH

To obtain a text file tree list of a hierarchy, use the TREELIST utility. This program is helpful in organizing a hierarchy that contains many worksheets. To execute the TREELIST utility on the simple CMOSCPU.SCH hierarchy, enter the following from the DOS command line:

TREELIST CMOSCPU.SCH TREE.TXT
<ENTER>

TREELIST is the name of the utility, CMOSCPU.SCH is the name of the root worksheet of the hierarchy, TREE.TXT is the name of a text file to send the tree information (this is useful for examining the results of the utility), <ENTER> refers to pressing the ENTER key

After the utility program has completed, the tree information is sent to the TREE.TXT file. To examine the file, use a text editor or word processing software. Examining the TREE.TXT file results in the following:

```
<<<Root File>>>
[CMOSCPU.SCH]
   POWER SUPPLY
   [POWER.SCH]
   CMOS MEMORY
   [MEMORY.SCH]
```

Discussion:

All worksheet file names are enclosed within brackets [file names], sheet symbol names are listed above the file names. In this example, we see the root file has a file name of: CMOSCPU.SCH. Below the root, are the remaining sheet symbols and their associated file names. File POWER.SCH belongs to the sheet symbol named POWER SUPPLY. File MEMORY.SCH belongs to the sheet symbol named CMOS MEMORY.

```
/N00013 U1(29) U4(11) U5(11);
/GND    SW1(COMMON) Q1(EMITTER) C3(2) U2(10) U2(9)
        C1(2) C2(2) U1(31),
        U1(20) BT1(-) C4(2) D4(DC OUT MINUS)
        R4(WIPER) R4(1) C5(2),
        U4(9) U5(9) U5(10) U4(10);
/VDD    R2(1) R1(1) Q2(EMITTER) U3(VO) D3(ANODE)
        C5(1);
        U2(20) U1(40) U5(20) U4(20);
/N00017 U2(19) U5(8) U4(8);
/N00018 U2(18) U5(7) U4(7);
/N00019 U2(17) U5(6) U4(6);
/N00020 U2(16) U5(5) U4(5);
/N00021 U2(15) U5(4) U4(4);
/N00022 U2(14) U5(3) U4(3);
/N00023 U2(13) U5(2) U4(2);
/N00024 U2(12) U5(1) U4(1);
/N00025 U2(1) U1(39) U4(15);
/N00026 U2(2) U1(38) U4(14);
/N00027 U2(3) U1(37) U4(13);
/N00028 U2(4) U1(36) U4(12);
/N00029 U2(5) U1(35) U5(15);
/N00030 U2(6) U1(34) U5(14);
/N00031 U2(7) U1(33) U5(13);
/N00032 U2(8) U1(32) U5(12);
/N00033 D3(CATHODE) D2(CATHODE);
```

```
NET.CMP  CALAY Component text file

10         R3      shape    -X-     -Y-     0
10 UF      C3      shape    -X-     -Y-     0
10K        R2      shape    -X-     -Y-     0
12 MHZ     X1      shape    -X-     -Y-     0
1K         R4      shape    -X-     -Y-     0
1N4001     D2      shape    -X-     -Y-     0
1N4001     D3      shape    -X-     -Y-     0
2.7K       R1      shape    -X-     -Y-     0
30 PF      C1      shape    -X-     -Y-     0
30 PF      C2      shape    -X-     -Y-     0
4 V        BT1     shape    -X-     -Y-     0
470        C4      shape    -X-     -Y-     0
470        C5      shape    -X-     -Y-     0
4:1        T1      shape    -X-     -Y-     0
51C68      U4      shape    -X-     -Y-     0
51C68      U5      shape    -X-     -Y-     0
80C51      U1      shape    -X-     -Y-     0
82C82      U2      shape    -X-     -Y-     0
CR0127     D4      shape    -X-     -Y-     0
```

8	1	U2	82C82
9	1	SW1	SPST
10	1	D1	IN4004
11	1	R3	10
12	2	C4,C5	470
13	1	Q2	TIP31C
14	1	R4	1K
15	1	BT1	4 V
16	2	D2,D3	1N4001
17	1	U3	LM123
18	1	T1	4:1
19	1	D4	CR0127
20	2	U4,U5	51C68

5.5 COMPLEX HIERARCHIES

As mentioned previously, a complex hierarchy is one in which some sheet symbols are used multiple times in the design. This type of hierarchical organization enables you to replicate blocks of common logic without having to create a separate worksheet. All you do is place the circuit to be replicated into a worksheet. Then, draw a sheet symbol to represent that worksheet, and place that sheet symbol in multiple instances within other worksheets.

Creating a complex hierarchy is just as easy as creating a simple hierarchy. The difference between the two types comes when handling and executing utility programs. A simple hierarchy stores the reference designators in each worksheet. However, a complex hierarchy uses one or more worksheets multiple times. Therefore, you cannot store the reference designators in the worksheets. In this case, an annotation file provides the method for storing the reference information in a binary format.

The annotation file is created using the ANNOTATE utility program. When you use other utilities, the annotation file is used as the input file instead of the root worksheet file name.

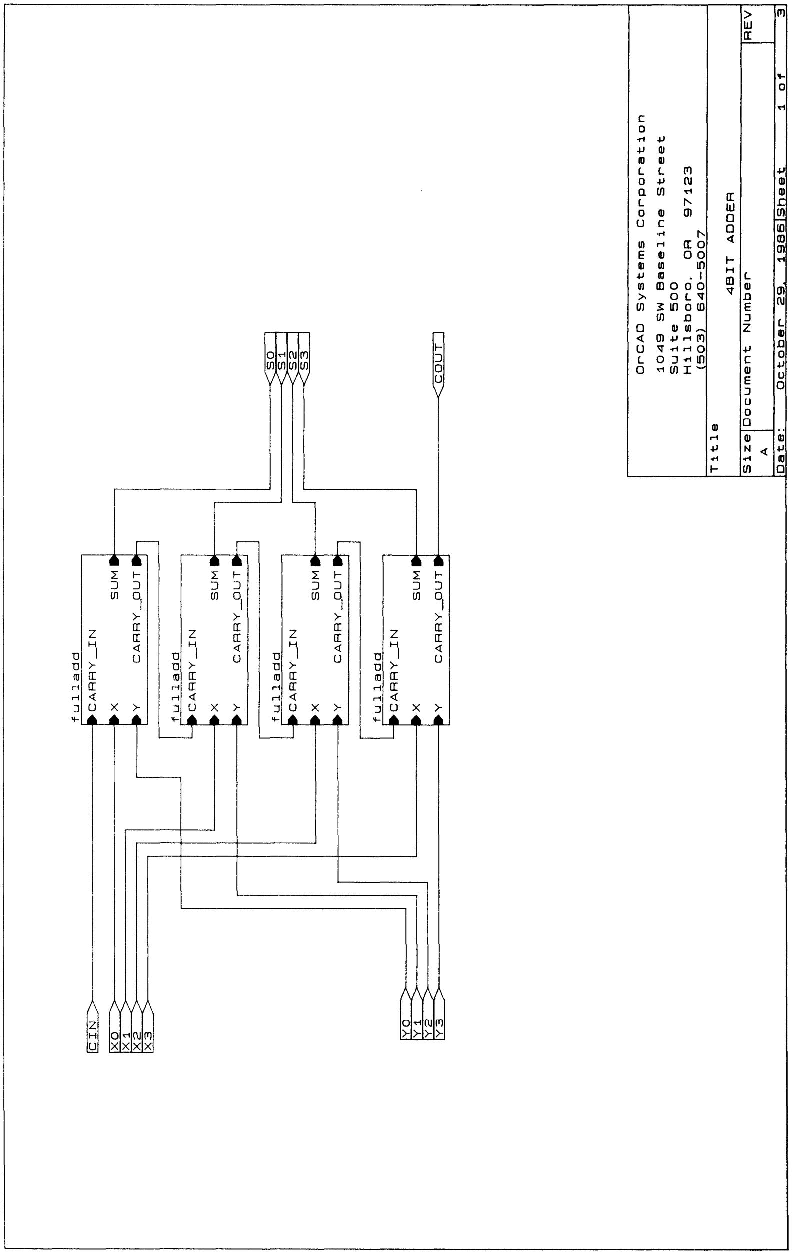


Figure 5-5. 4BIT ADDER Root Sheet

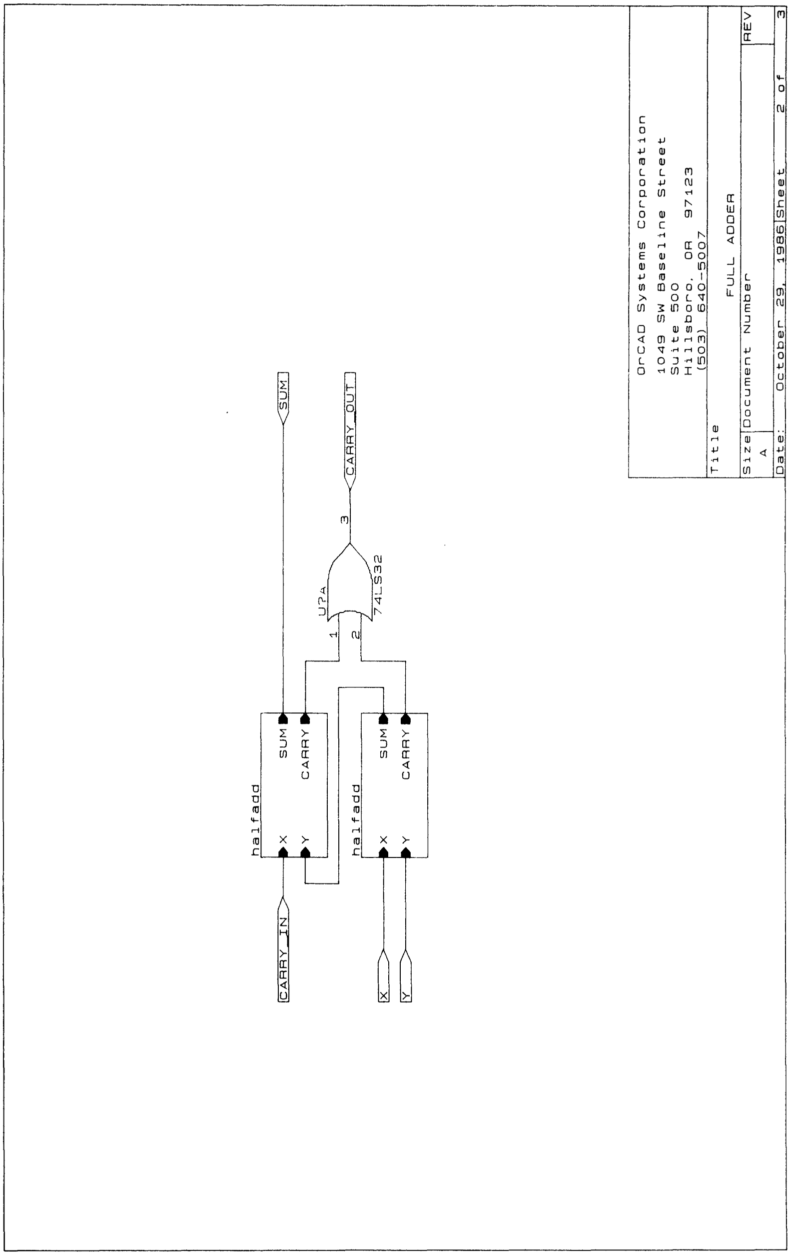


Figure 5-6. FULL ADDER Worksheet

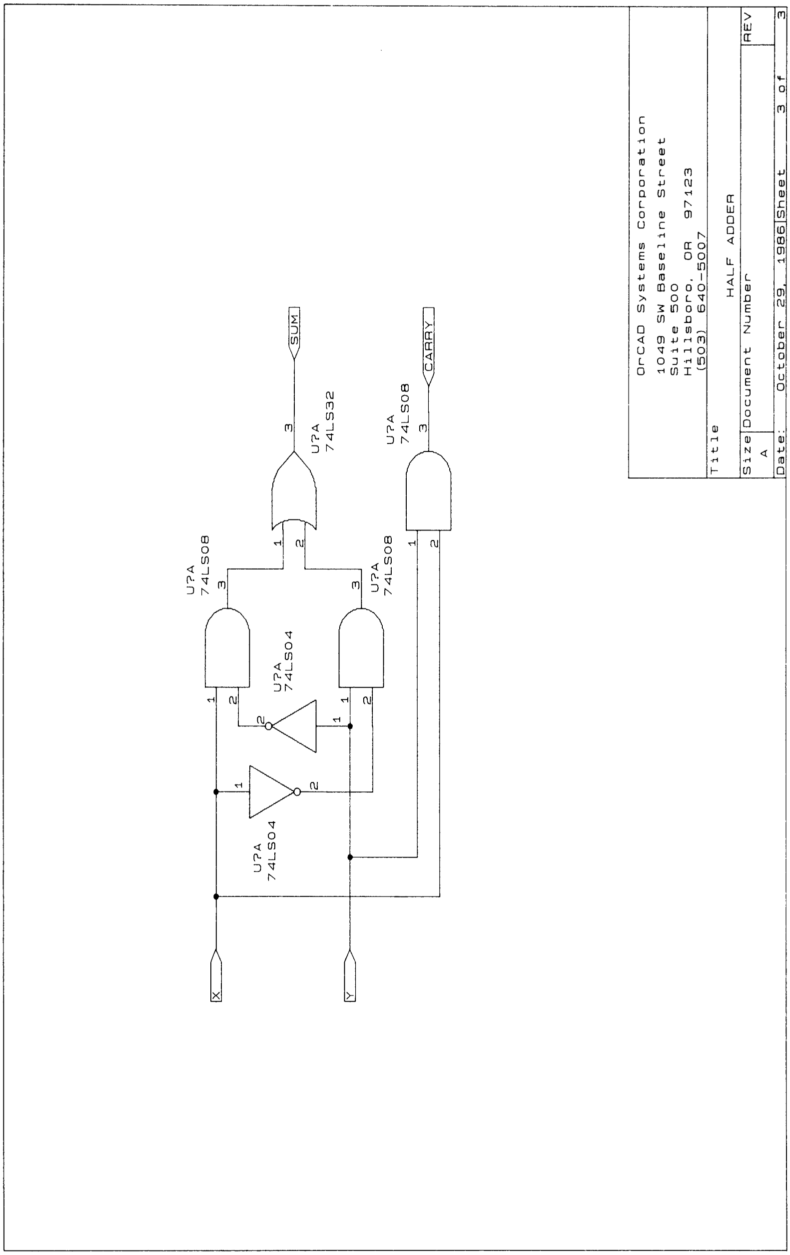


Figure 5-7. HALF ADDER Worksheet

utility program to generate the annotation file. To annotate the complex hierarchy, assume that the root worksheet file name is: 4BIT.SCH. Enter the following from the DOS command line:

ANNOTATE 4BIT.SCH 4BIT.ANN <ENTER>

ANNOTATE is the name of the utility, 4BIT.SCH is the name of the root worksheet of the hierarchy, 4BIT.ANN is the name of the annotation file, <ENTER> refers to pressing the ENTER key

After the utility program has completed, an annotation file is generated which will be used as the source in all other utilities. At this point, we should run the CLEANUP utility. However, if you feel confident in your design, you may go ahead and run ERC. This is what is illustrated next.

5.6.3 Invoking The ERC Utility

To run the ERC utility program, specify the annotation file: 4BIT.ANN as the source. Enter the following from the DOS command line:

ERC 4BIT.ANN ERROR.TXT /A /U <ENTER>

ERC is the name of the utility, 4BIT.ANN is the name of the Annotation file, ERROR.TXT is the name of a text file to send the error information (this is useful for examining the results of the utility), /A signifies that the source is an annotation file, /U generates a unconnected pin report, <ENTER> refers to pressing the ENTER key

In this example, the ERROR.TXT file will contain no errors.

```
[halfadd.sch]
halfadd
[halfadd.sch]
```

Discussion:

All worksheet file names are enclosed within brackets [file names], sheet symbol names are listed above the file names. In this example, we see the root file has a file name of: 4BIT.SCH. Below the root, are the remaining sheet symbols and their associated file names.

5.6.5 Invoking The NETLIST Utility

To obtain a net list of a hierarchy, use the NETLIST utility. For this example, a COMPUTERVISION net list will be extracted. To extract a COMPUTERVISION NETLIST, specify the annotation file: 4BIT.ANN as the source. Enter the following from the DOS command line:

NETLIST 4BIT.ANN NET.TXT
COMPUTERVISION /A /S <ENTER>

NETLIST is the name of the utility, 4BIT.ANN is the name of the Annotation file, NET.TXT is the name of a text file to send the net information (this is useful for examining the results of the utility), COMPUTERVISION is the name of the desired net list format, /A specifies that the source is a Annotation file, /S indicates a special net list format is desired, <ENTER> refers to pressing the ENTER key

After the utility program has completed, the net information is sent to the NET.TXT file. To examine the file, use a text editor or word processing software. Examining the NET.TXT file results in the following:

```
0052 S1                   U6-3
0053 N00053               U1-11  U8-10  U7-9  U8-4
0054 S2                   U6-11
0055 N00055               U6-8  U11-2  U12-3  U9-12
0056 Y0                   U3-5  U2-12  U4-4
0057 Y1                   U7-1  U5-4  U5-12
0058 S3                   U10-8
0059 Y2                   U7-11  U8-12  U9-4
0060 Y3                   U12-5  U11-4  U11-12
0061 COUT                 U10-6
0062 VCC                  U1-14  U2-14  U3-14 U4-14
                          U5-14  U6-14  U7-14 U8-14
                          U9-14  U10-14 U11-14 U12-14
0063 GND                  U1-7  U2-7  U3-7  U4-7
                          U5-7  U6-7  U7-7  U8-7
                          U9-7  U10-7  U11-7  U12-7
```

5.6.6 Invoking The PARTLIST Utility

To obtain a text file part list, use the PARTLIST utility. To execute the PARTLIST utility, enter the following from the DOS command line:

> PARTLIST 4BIT.ANN PART.TXT /A
> <ENTER>

PARTLIST is the name of the utility, 4BIT.ANN is the name of the annotation file, PART.TXT is the name of a text file to send the part information (this is useful for examining the results of the utility), /A signifies that the source is an annotation file, <ENTER> refers to pressing the ENTER key

After the utility program has completed, the part information is sent to the PART.TXT file. To examine the file, use a text editor or word processing software. Examining the PART.TXT file, results in the following:

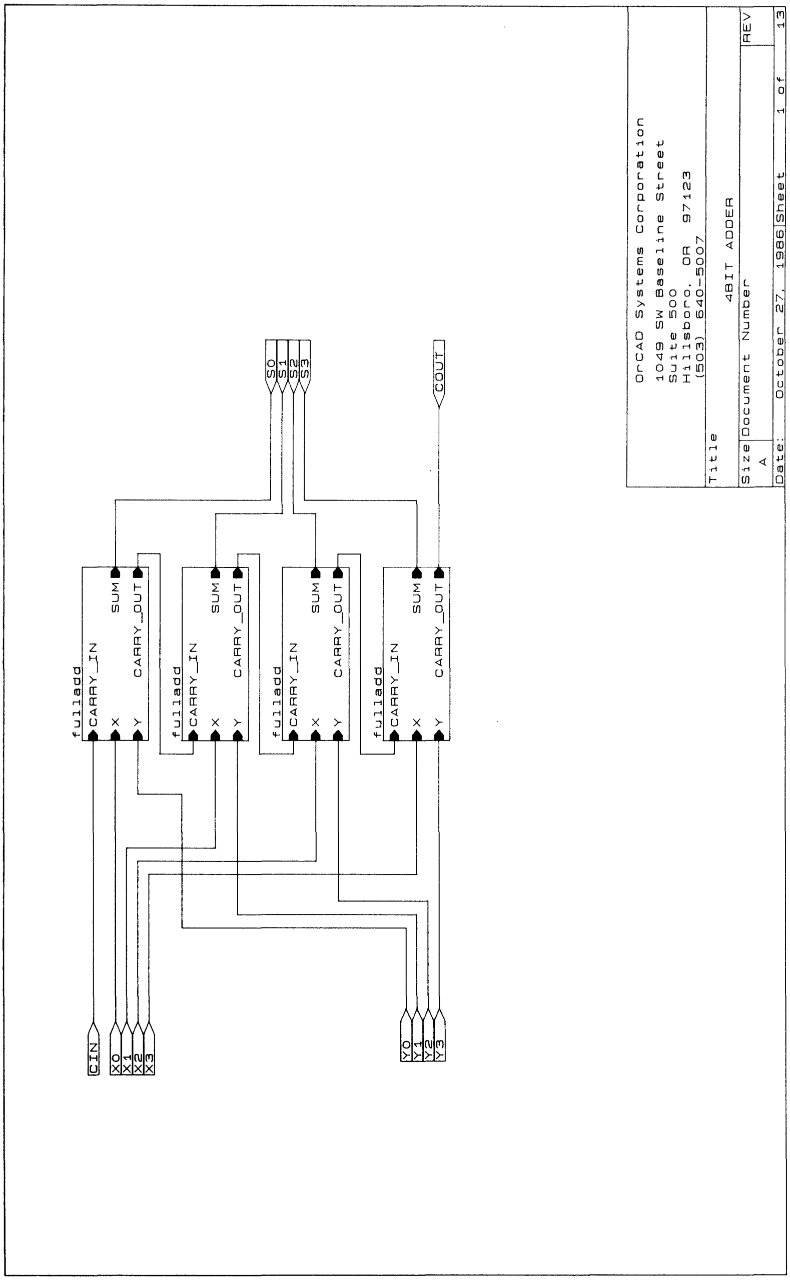


Figure 5-8. 4BIT Root Worksheet

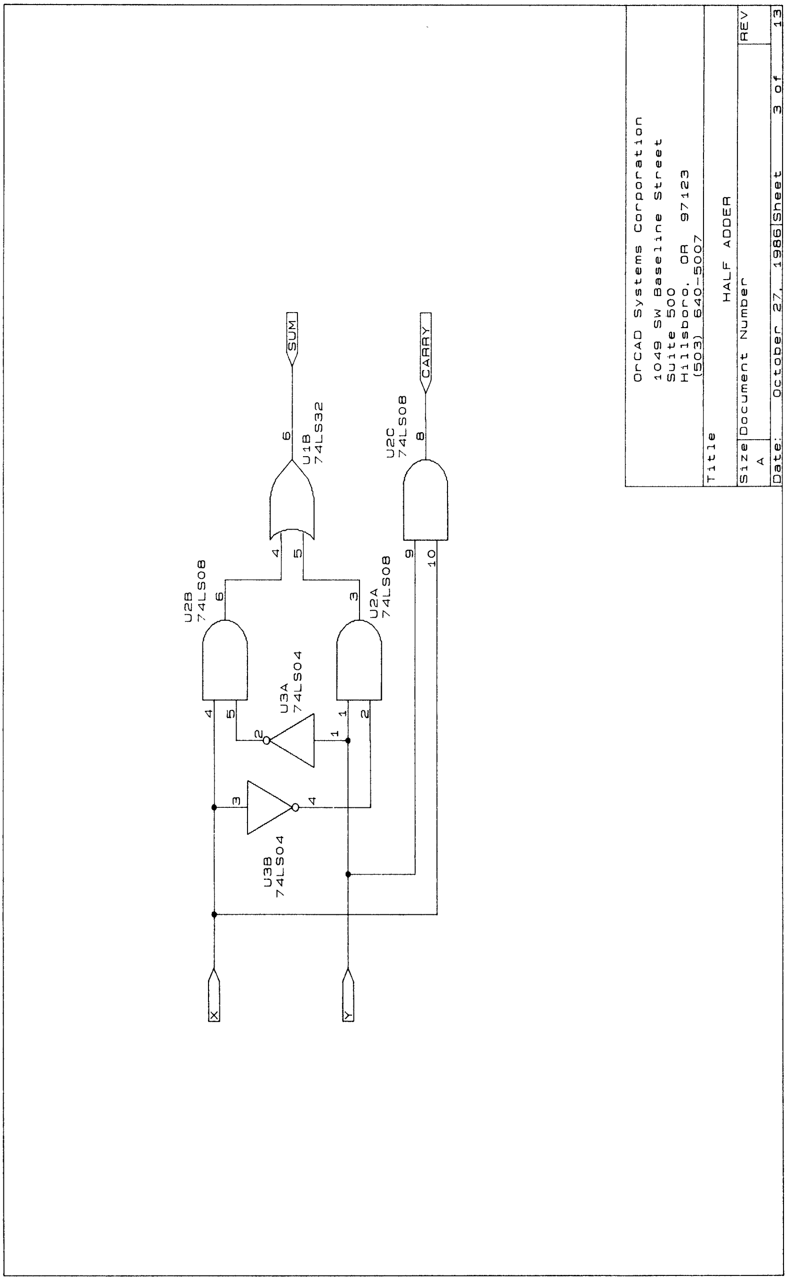


Figure 5-10. HALF ADDER Worksheet

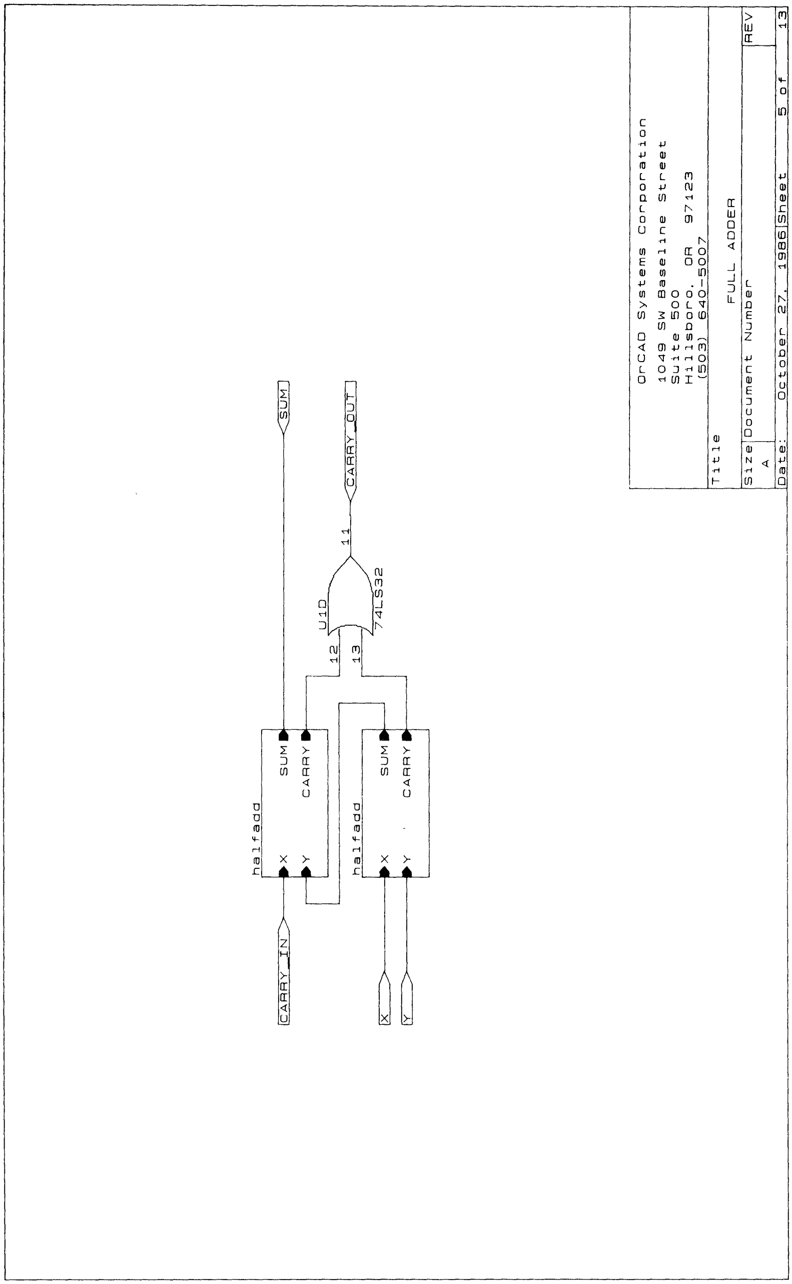


Figure 5-12. FULL ADDER Worksheet

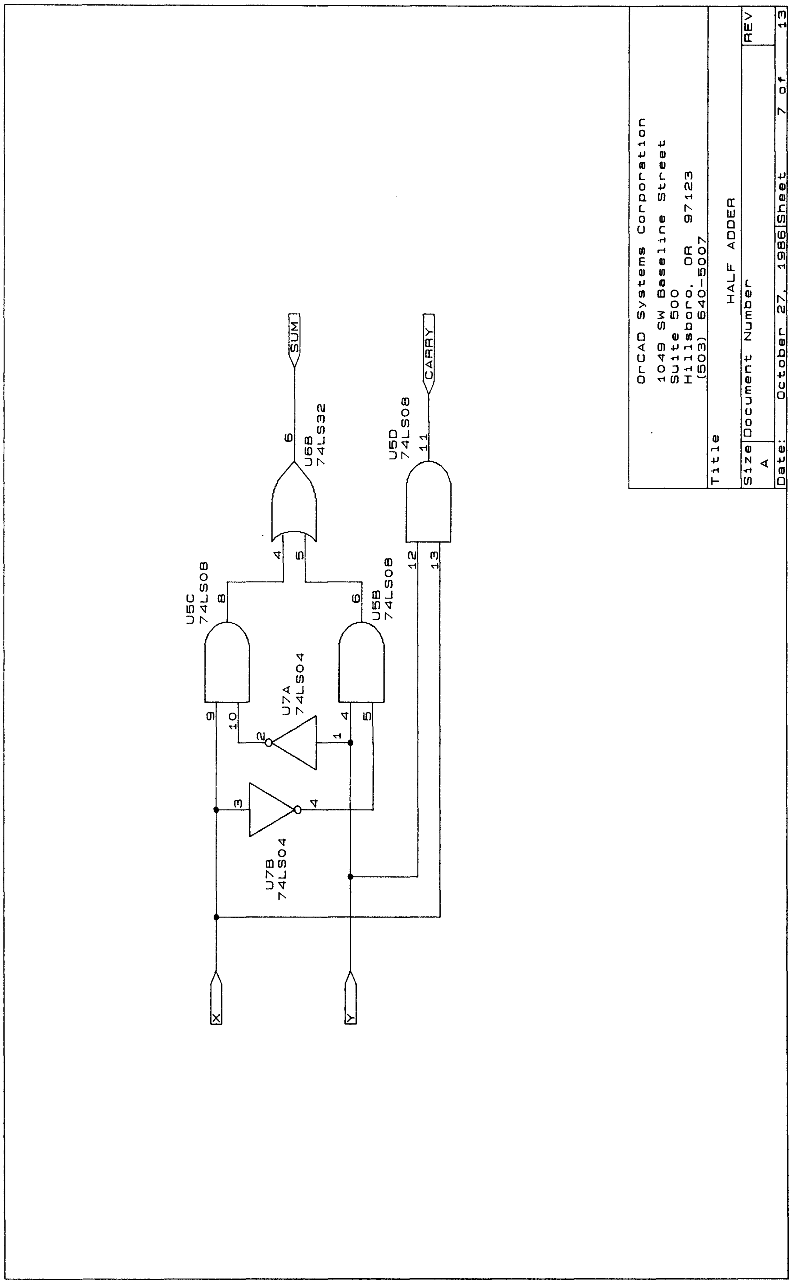


Figure 5-14. HALF ADDER Worksheet

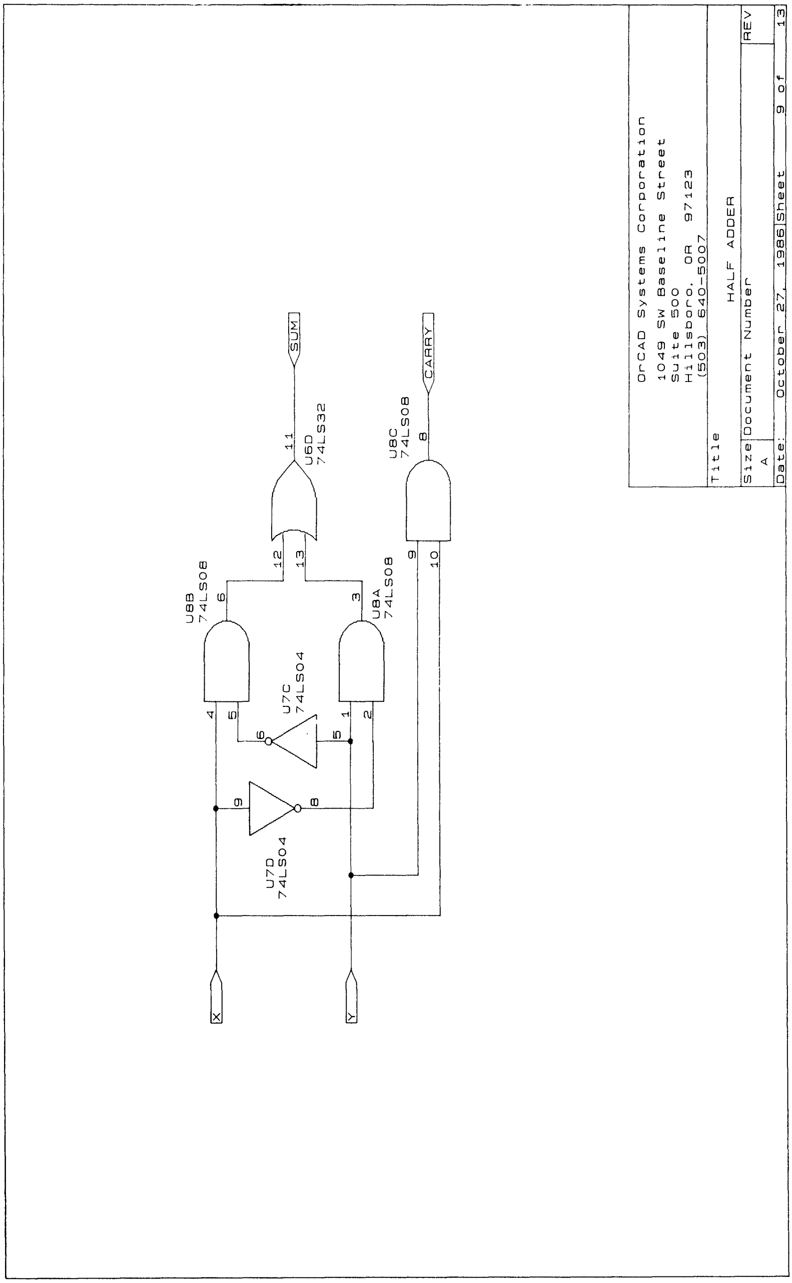


Figure 5-16. HALF ADDER Worksheet

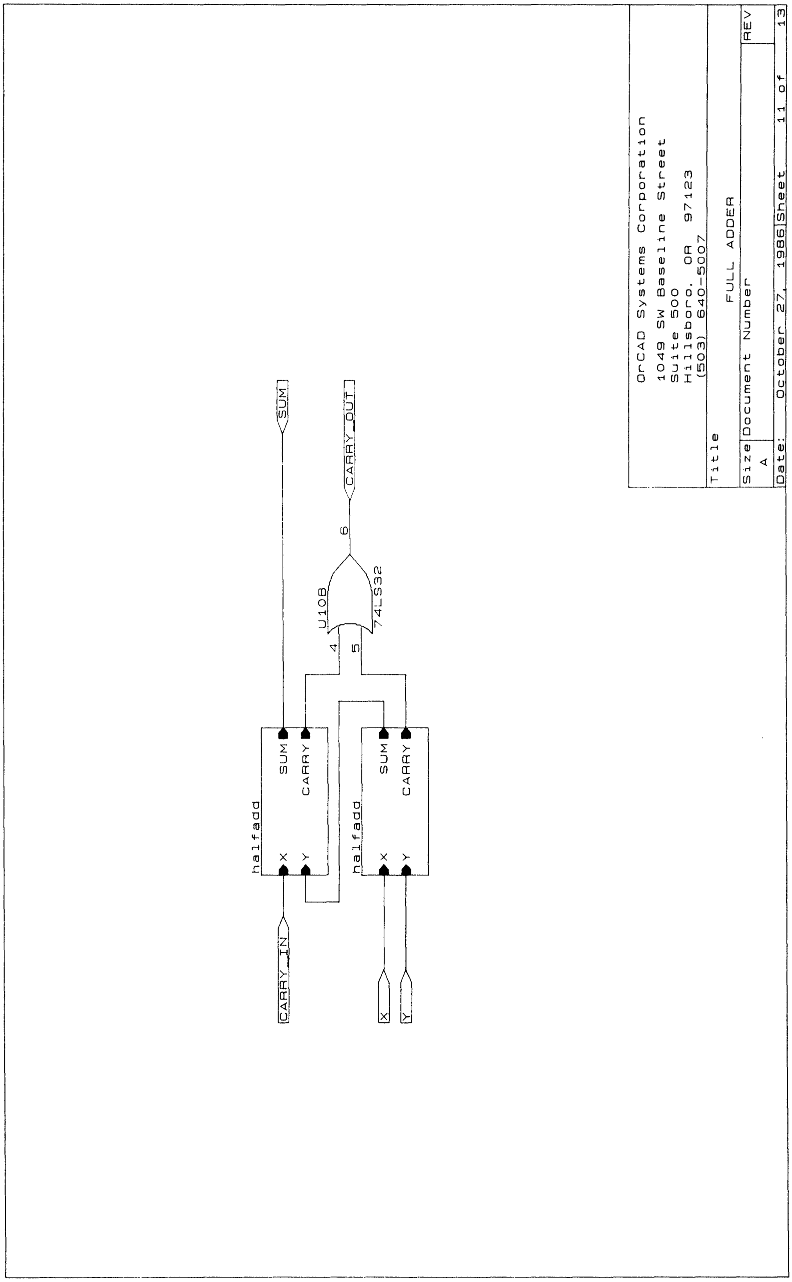


Figure 5-18. FULL ADDER Worksheet

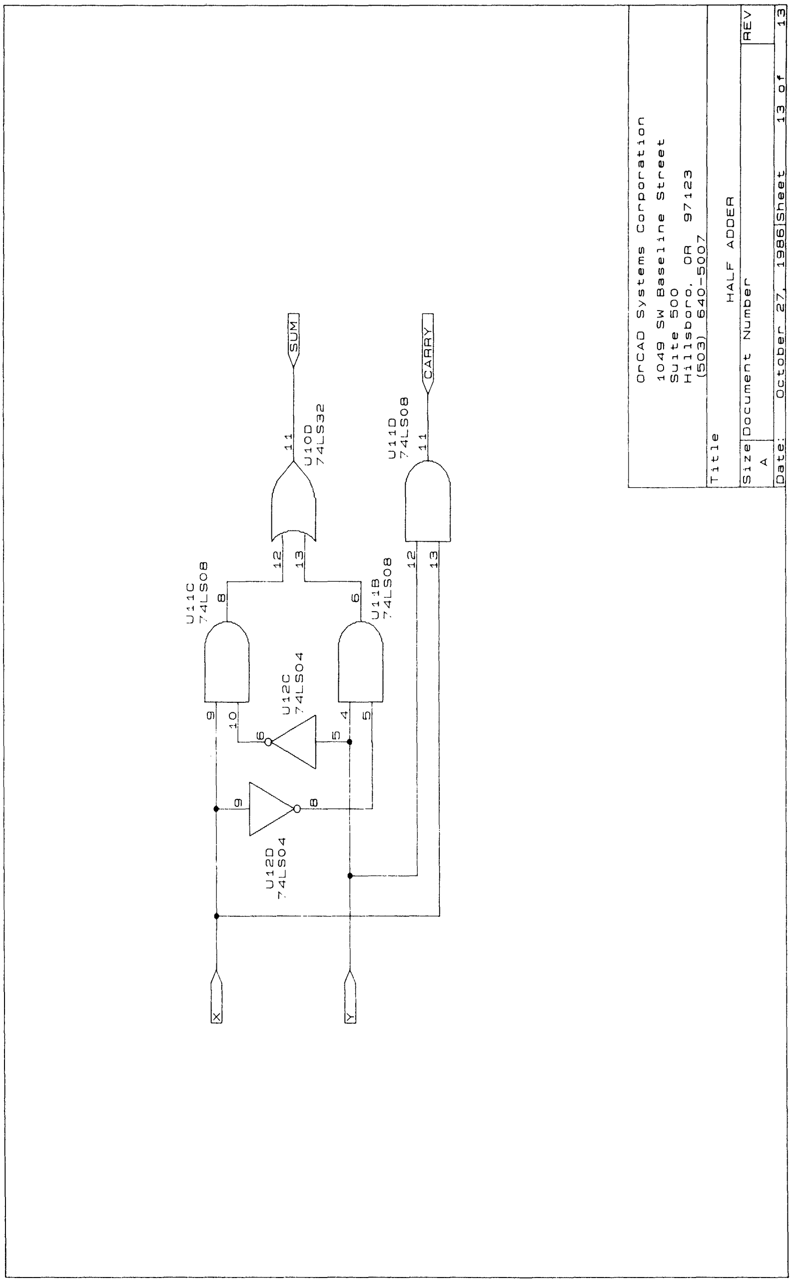


Figure 5-20. HALF ADDER Worksheet

uppercase or lowercase. Switches may be entered in any order on the invocation line.

Items in square brackets [ ] represent parameters which are optional. When you include this optional information, do not include the square brackets when you enter the parameter.

A source path may be one of three different types:

1. The path name to the root sheet of a hierarchical organization of sheets.

2. The path name of the annotation file created by the ANNOTATE program.

3. The path name of a text file which consists of a list of sheet path names. Each sheet path name must be either separated by a space or tab or be on different lines.

The WORKSHEET PREFIX specified in the configuration of OrCAD/SDT III will be used as the prefix of the sheet path names in items 1 and 3 above. This prefix may be overridden by placing a drive designation or a back slash at the beginning of the path. For example, FILE.SCH will be prefixed by the WORKSHEET PREFIX while A:FILE.SCH and \FILE.SCH will not be prefixed. When overriding, the name must be a valid and complete DOS path name.

Additionally, when overriding a root sheet path name of a hierarchical organization of sheets, only the root sheet path name will be overridden. All sheets scanned will use the WORKSHEET PREFIX and the file name specified for that sheet.

6.2.1 Invoking Utility Programs On Hierarchy File Structures

To invoke utility programs on hierarchical file structures, enter the name of the utility program followed by the worksheet file name from the DOS command line. For example: You have created a hierarchical worksheet with a root file name called COMPUTER.SCH. The hierarchy has three additional hierarchical sheets referenced in COMPUTER.SCH with the file names CPU.SCH, IO.SCH, and MEMORY.SCH. To invoke a utility, just enter the following from the DOS command line:

> UtilityName *computer.sch* [optional parameters]

Where UtilityName is the name of the utility program, *computer.sch* is the name of the root file in the hierarchy, and [optional parameters] are parameters for the utility program such as switch options or other file names.

6.2.2 Invoking Utility Programs On Flat File Structures

Invoking utility programs on flat file structures is handled slightly different than hierarchical files. A flat file structure is a collection of single worksheets. Since the worksheets do not contain references to other worksheet files as a hierarchy does, the flat file structure must contain a list of the worksheet filenames.

This list is a simple text file that you create, that contains the names of the individual worksheets. By invoking a utility program on the name of the text file, all information is merged together into the original worksheets.

For example: your single worksheet file name is called ONESHEET.SCH. To run a utility program, enter the following on the DOS command line:

> UtilityName ONESHEET.SCH /O [optional parameters]

Where UtilityName is the name of the utility program, /O treats the worksheet as a one sheet file structure, and [optional parameters] are any parameters that may be added. Such as switch options or destination file names.

6.2.4 Two Floppy Disk System Applications

If you configured your working floppy disks as recommended in Section 2, many of the utility programs will reside on the DRAFT disk. The DRAFT disk should be placed in drive A and the DRIVER/LIBRARY disk placed in drive B.

When invoking a utility program on a dual floppy system, enter the /E option switch after the file name. The /E switch will display the message "Type Any Key To Continue", enabling the system to pause for you to remove the DRIVER/LIBRARY disk in drive B and insert the SHEET disk.

For example, from the DOS command line enter:

> UtilityName *filename* /E [optional parameters]

where UtilityName is the name of the utility program, *filename* is the name of your worksheet file, /E enables the system to pause for you to switch disks, and [optional parameters] are any parameters that may be added. Such as switch options or destination file names.

is where the output of the program is to be placed. The ANNOTATE utility program does not process an annotation file. The *destination* must be included on the command line unless the /M switch is specified. The *destination* path name is any valid DOS file path name.

ANNOTATE updates reference designators in the order they were placed in the worksheet. Any object that has its reference designator manually edited, will be assigned a new reference designator when the worksheet is annotated. If you want to selectively change reference designators, and leave others unmodified, use the BACKANNO utility program.

The /C switch causes the configuration menu to be invoked. This allows the OrCAD/SDT III environment to be modified.

The /D switch causes ANNOTATE to descend into parts defined as sheetpath parts. That is, you can think of the utility as treating a sheetpath part as sheet during its operation.

The /E switch causes the utility program to display the message "Type Any Key To Continue", enabling the system to pause for you to remove the DRIVER/LIBRARY disk in drive B and insert the SHEET disk. This switch is used only on systems with two floppy disk drives.

The /F switch causes the ANNOTATE program to read the source as a text file, for flat file structure applications.

The /M switch causes the annotation information to be merged directly into the source instead of a separate annotation file. This switch may be specified for flat file where *root.sch* is the path and name of the root sheet in the hierarchy and *annotation.out* is the path and file name to place the annotation information

6.3.2 Flat File Structure Invocation Example

To annotate a flat file structure and merge the annotation information into the original sheets:

ANNOTATE *flatfile.txt* /F /M

where *flatfile.txt* is a text file containing a list of schematic file names to be scanned, /F is used to signify that flatfile.txt is a text file, and /M merges the information into flatfile.txt

6.3.3 One Sheet File Structure Invocation Example

To annotate a single sheet schematic and merge the annotation information into the sheet:

ANNOTATE *sheetname.sch* /O /M

where *sheetname.sch* is the name of the single sheet schematic, /O is used to signify that sheetname.sch is a single sheet schematic, and /M merges the information into sheetname.sch.

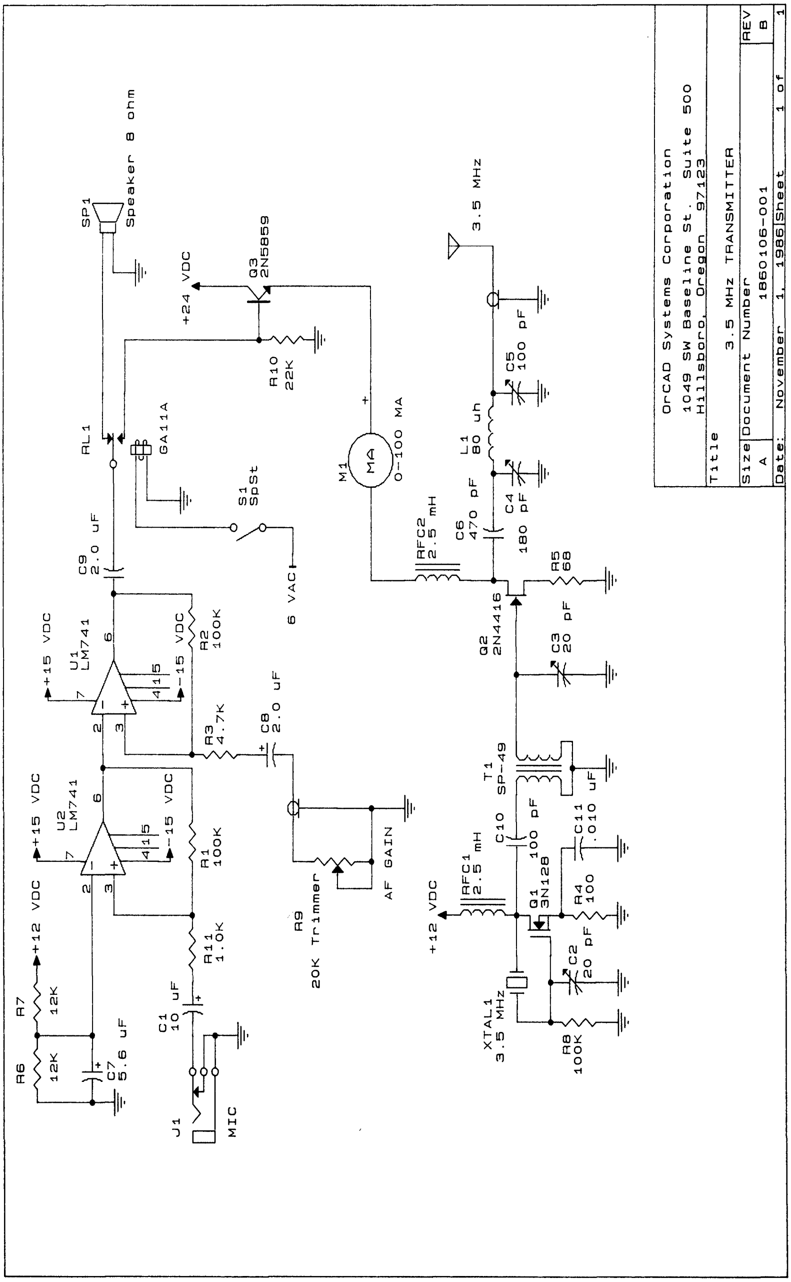


Figure 6-2. Annotated Version of Figure 6-1

6.4 BACKANNO

Purpose:

The BACKANNO program scans either a hierarchy, flat file, one sheet file structure, or a annotation file and updates part reference designators in your design. The input to the program, a WAS/IS text file containing old and new reference designators, is used to update the schematic worksheets.

Format:

BACKANNO source was/is
[/A][/C][/D][/E][/F][/O][/Q]

Remarks:

The source may be either the root sheet name of a hierarchical file structure, the name of a text file in a flat file structure, an annotation file, or the file name of a one sheet file structure.

If the source is the name of a text file in a flat file structure, the /F switch must be included. If the source is the file name of a one sheet file structure, the /O switch must be included. If the source is the annotation file created by the ANNOTATE program, then the /A switch must be included on the invocation line.

was/is is as text file containing the old and new reference designator pairs. The file may have any valid DOS path name. The format of the was/is file is discussed later in this Section.

where *root.sch* is the path and name of the root sheet in the hierarchy, and *was/is* is the name of the text file containing the old and new reference designators

6.4.2 Invocation Example Using A Flat File Structure

To back annotate a flat file structure containing multiple sheets:

BACKANNO *flatfile.txt was/is* /F

Where *flatfile.txt* is a text file containing a list of schematic file names to be scanned, *was/is* is the name of the text file containing the old and new reference designators, and /F is used to signify that flatfile.txt is a text file

6.4.3 Invocation Example Using A One Sheet File Structure

To back annotate a single sheet schematic:

BACKANNO *sheetname.sch was/is* /O

Where *sheetname.sch* is the name of the single sheet schematic, *was/is* is the name of the text file containing the old and new reference designators, and /O is used to signify that sheetname.sch is a single sheet schematic.

6.5 CLEANUP

Purpose:

The CLEANUP program scans either a hierarchy, flat file, one sheet file structure, or an annotation file checking for wires, buses, junctions, labels, module ports, and other objects that are placed on top of each other.

CLEANUP removes duplicate or overlapping wires, buses, and junctions, and displays warning messages advising you of other duplicate objects.

CLEANUP does not check for objects overlapping part leads, wires overlapping buses, or wire bus entries overlapping bus bus entries. This utility program should be used whenever you feel that there may be drawing errors in the worksheet. We recommend that all worksheets be checked with

CLEANUP to reduce errors and warnings that may occur when you use other utility programs.

Format:

CLEANUP source [destination]
[/A][/C][/E][/F][/G][/O][/Q][/R]

Remarks:

The source may be either the root sheet name of a hierarchical file structure, the name of a text file in a flat file structure, or the file name of a one sheet file structure.

If the source is the name of a text file in a flat file structure, the /F switch must be included. If the

The /G switch checks the entire worksheet for items that are placed off grid and reports them.

The /O switch causes the file name of the source to be read as a one sheet file structure.

The /Q switch causes the CLEANUP program to run "quietly". This means that only the invocation messages and error messages if any, are displayed. If this switch is not specified, the program will display intermediate tracking activity.

The /R switch is used to repeat the CLEANUP procedure if a file is too large for one pass. This switch causes the CLEANUP program to repeat the clean up process on a worksheet if needed. The program will repeat a limited number of times to avoid looping.

6.5.1 Invocation Examples Using Hierarchical Structured Files

1. To check sheets in a hierarchical schematic:

   CLEANUP *root.sch*

   Where *root.sch* is the path and name of the root sheet in the hierarchy

2. To check a sub-sheet in a hierarchical schematic:

   CLEANUP *subsheet.sch* /O

   Where *subsheet.sch* is the path and name of the sub-sheet in the hierarchy, /O signifies that the subsheet.sch file name is a single sheet

Where *annotation.out* is the output from the ANNOTATE program, /A causes the CLEANUP program to read annotation.out as an annotation file.

The destination is any valid DOS pathname and is where the output of the program is to be placed. If a destination is not specified, the output will be directed to the console.

The /A switch causes the CROSSREF program to read the source as an annotation file.

The /C switch causes the configuration menu to be invoked. The /F switch causes the CROSSREF program to read the source as a text file, for flat file structure application.

The /N switch causes the CROSSREF program to output just one listing of all parts sorted first by names, and then by references.

The /O switch causes the filename of the source to be read as a one sheet file structure.

The /P switch causes the CROSSREF program to output the XY coordinates for all parts.

The /Q switch signals the CROSSREF program to run "quietly." This means that only the invocation messages and error messages, if any, are displayed. If this switch is not specified, the program will display intermediate tracking activity.

The /R switch causes the CROSSREF program to output just one listing of all parts sorted first by references, then by names.

The /S switch causes the format of the output listing to be single-spaced.

The /V switch outputs a verbose format, which causes the header information to be included on every page.

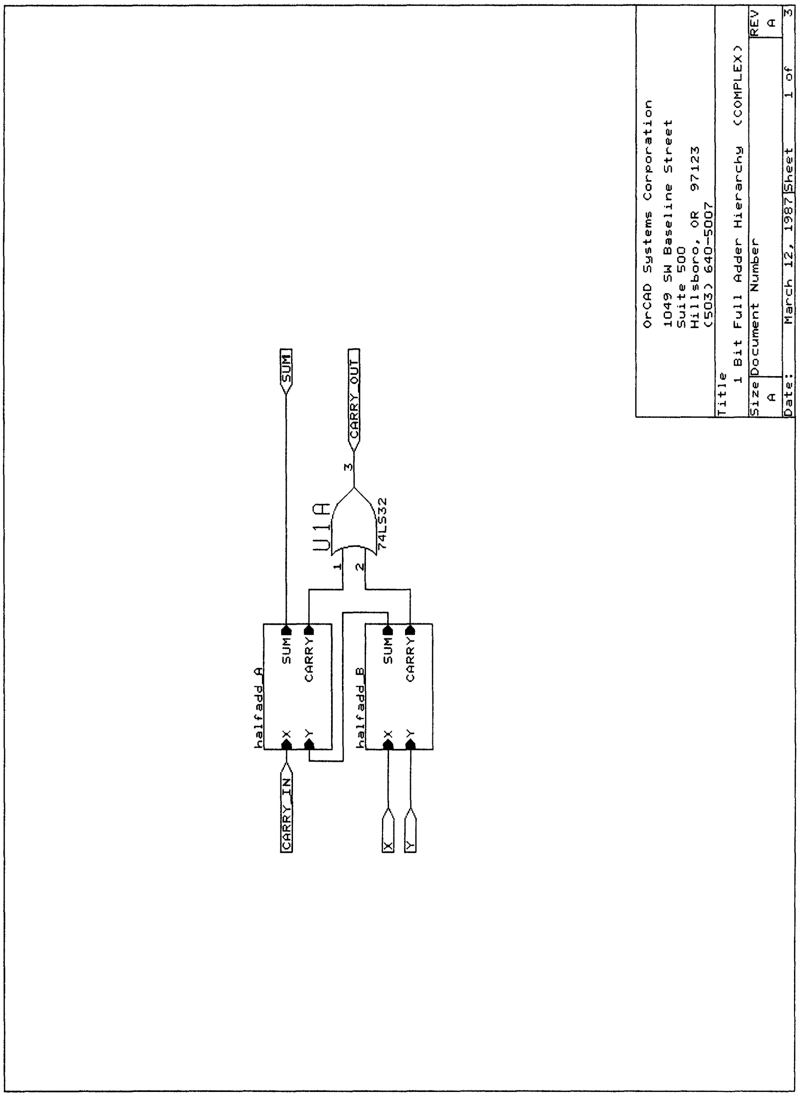


Figure 6-3. 1 Bit Full Adder Hierarchy

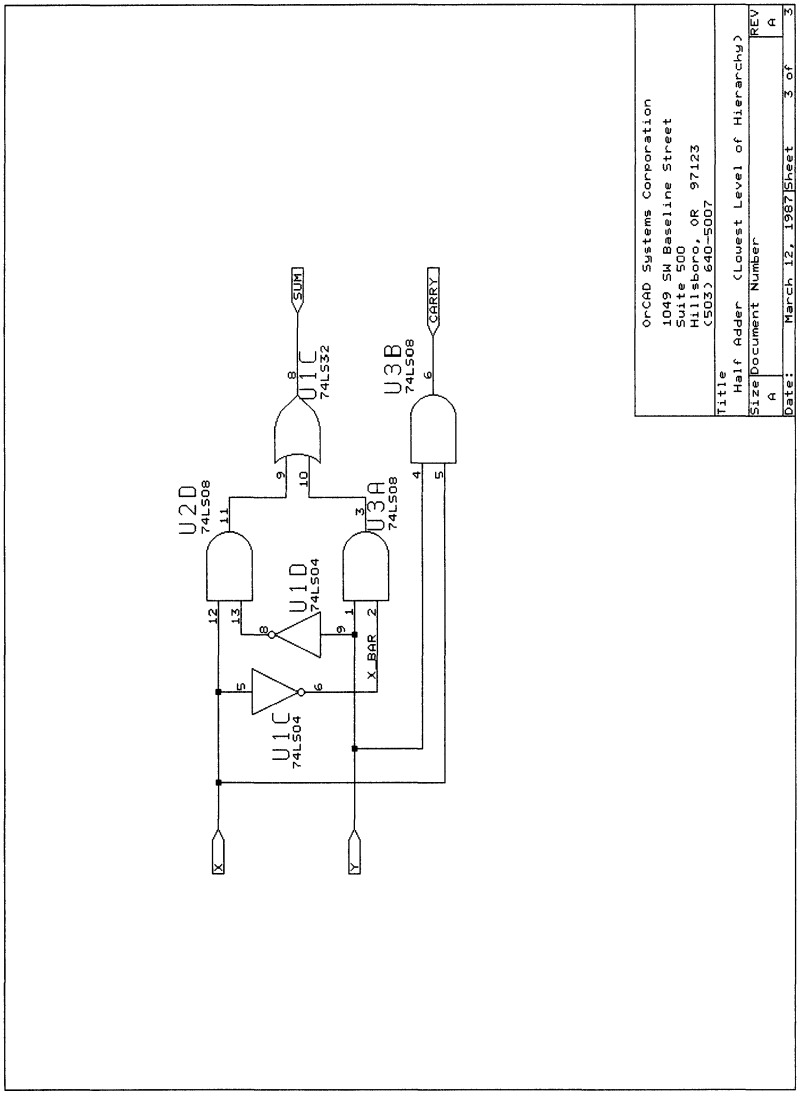


Figure 6-5. Half Adder B

9	74LS08	U4A	3	halfadd.sch
10	74LS08	U4B	3	halfadd.sch
11	74LS32	U1A	1	fulladd.sch
12	74LS32	U1B	2	halfadd.sch
13	74LS32	U1C	3	halfadd.sch

6.6.3 Example Using CROSSREF with /A, /R, /P, and /V

Here is another example. This example uses the four switches A, R, P, and V.

CROSSREF FULLADD.ANN FULL1.OUT /A /R /P /V

The file FULL1.OUT contains the following:

```
1 Bit Full Adder Hierarchy  (COMPLEX)   Revised: March 12,
1987
                                         Revision: A

OrCAD Systems Corporation
1049 SW Baseline Street
Suite 500
Hillsboro, OR  97123
(503) 640-5007

Part Cross Reference Listing September 10, 1987 8:52:58 Page
1
```

Item	Reference	Part	Sheet	Filename	X ,	Y
1	U1A	74LS32	1	fulladd.sch	4.70,	2.90
2	U1B	74LS32	2	halfadd.sch	5.50,	2.40
3	U1C	74LS32	3	halfadd.sch	5.50,	2.40
4	U2A	74LS04	2	halfadd.sch	3.30,	2.20
5	U2B	74LS04	2	halfadd.sch	3.80,	2.40
6	U2C	74LS04	3	halfadd.sch	3.30,	2.20
7	U2D	74LS04	3	halfadd.sch	3.80,	2.40
8	U3A	74LS08	2	halfadd.sch	4.30,	1.80
9	U3B	74LS08	2	halfadd.sch	4.30,	3.00
10	U3C	74LS08	2	halfadd.sch	5.50,	3.60
11	U3D	74LS08	3	halfadd.sch	4.30,	1.80
12	U4A	74LS08	3	halfadd.sch	4.30,	3.00
13	U4B	74LS08	3	halfadd.sch	5.50,	3.60

The /A switch causes the ERC program to read the source path as an annotation file.

The /C switch causes the configuration menu to be invoked. This allows the OrCAD/SDT III environment to be modified.

The /D switch causes ERC to descend into parts defined as sheetpath parts. That is, you can think of the utility as treating a sheetpath part as sheet during its operation.

The /E switch causes the utility program to display the message "Type Any Key To Continue", enabling the system to pause for you to remove the DRIVER/LIBRARY disk in drive B and insert the SHEET disk. This switch is used only on floppy-based systems.

The /F switch causes the ERC program to read the source as a text file, for flat file structure applications.

The /G switch checks the worksheet for Parts, Sheets, Labels, Module Ports, and Power objects that are placed off grid and reports them.

The /L switch causes the ERC program to produce a label report and place it in the destination file. A label report lists all labels and module ports that are connected in the worksheet.

The /O switch causes the filename of the source to be read as a one sheet file structure.

The /Q switch causes the ERC program to run "quietly". This means that only the invocation messages and error messages if any, are displayed. If this switch is not specified, the program will display intermediate tracking activity.

where *flatfile.txt* is a text file containing a list of schematic filenames to be scanned, and /F is used to signify that flatfile.txt is a text file.

2. To check one sheet in a flat file structure:

   ERC *sheetname.sch* /O

   where *sheetname.sch* is the name of the single sheet in the flat file structure, and /O is used to signify that sheetname.sch is a single sheet schematic.

3. To check a flat file structure and direct the output of the ERC program to a file:

   ERC *flatfile.txt* whatfile /F

   where *flatfile.txt* is a text file containing a list of schematic file names to be checked, whatfile is the path and filename to place the ERC information, and /F is used to signify that flatfile.txt is a text file.

6.7.3 Invocation Examples Using A One Sheet File Structure

1. To check a single sheet schematic:

   ERC *sheetname.sch* /O

   where *sheetname.sch* is the name of the single sheet schematic, and /O is used to signify that sheetname.sch is a single sheet schematic

2. To check a single schematic and direct the output of the ERC program to a file:

   ERC *sheetname.sch* whatfile /O

sheet schematic, *whatfile* is the path and filename to place the ERC information, /U is used to signify that sheetname.sch is a single sheet schematic, and /O is used to signify a one sheet schematic

6.7.6 Typical ERC Messages and Resolutions

Listed below, are the most common error messages produced by ERC and possible solutions to resolve the errors.

Message:

<<<WARNING>>> Unconnected MODULE PORT "......." at X= .... at Y= ....

Check For:

A bus that is not properly labeled. It must be named in the form: BUSNAME[0..n]. Any module port that is connected to a bus must also be named in the proper form: BUSNAME[0..n]. For further information, refer to the NETLIST utility in Section 6.

Message:

WARNING - POWER Supplies are CONNECTED .... <-> ....

Check For:

This may be a warning that is acceptable in your design. If you intentionally connected two power supplies together, this warning will appear. If you did not connect two power supplies together, this indicates that a potential problem may exist.

Message:

<<<ERROR>>> Sheet Net on a bus does not have a proper format...can not process...

Check For:

This error typically results from a bus connected to a hierarchical sheet net. The net name must match the name of the bus that it is connected to. The form should be: BUSNAME[0..n]. For further information, refer to the NETLIST utility in Section 6, and Section 5 on Hierarchy.

6.7.7 How ERC Determines What's an Error

Table 6-1 summarizes the rules used by ERC when determining invalid connections. For example, an output connected to an output is an error (E) while a power pin connected to an I/O pin is a warning (W).

In the table, one connection is listed horizontally, and the other is listed vertically. For example, if you have an output connected to an input, you can find out the ERC value by starting in the OUT column (the third column) and going down to the IN row (actually the first row). The value is a period which represents an acceptable connection. However, if you follow the OUT column down to the OUT row (the third row), you see an E, which indicates an error.

For definitions of the pin types, refer to Section 7, Libraries. The connections prefixed with an "m" are module ports. You can have four types of module ports: input (mI), output (mO), bidirectional (mB), and unspecified (mU). The connections prefixed with an "s" are sheet net names. As with module ports, you can have four types of sheet net names: input (sI), output (sO), bidirectional (sB), and unspecified (sU).

6.8 LIBARCH

Purpose:

The LIBARCH program takes all the library parts used in the schematic files and makes a single library source, an archived library, containing only parts which are usable for those schematic files.

The LIBARCH program scans a hierarchy, flat file, one sheet file structure, or an annotation file output from the ANNOTATE program.

Format:

LIBARCH source [destination] [/A][/C][/F][/O][/Q]

Remarks:

The source may be either the root sheet name of a hierarchical file structure, the name of a text file in a flat file structure, or the filename of a one sheet file structure.

If the source is the name of a text file in a flat file structure, the /F switch must be included. If the source is the filename of a one sheet file structure, the /O switch must be included. If the source is the annotation file created by the ANNOTATE utility, then the /A switch must be included.

The destination is any valid DOS path name and is where the output, the ASCII text file describing the parts using OrCAD's Symbol Description Language, is to be placed. If a destination is not specified, the output will be directed to the console. If the destination is the name of an existing file,

6.9 LIBLIST

Purpose:

The LIBLIST program takes a library object file and generates an ASCII text file listing all of the parts in that

Format:

LIBLIST <library object name> [destination] processing is currently taking place.

The format is the name of the special file format that you want the net list generated in. If you do not specify a format, then the format defaults to EDIF. If you want to generate a format other than EDIF, refer to the /S switch option below.

The /A switch causes the NETLIST program to read the source path as an annotation file.

The /C switch causes the configuration menu to be invoked. This allows the OrCAD/SDT III environment to be modified.

The /D switch causes NETLIST to descend into parts defined as sheetpath parts. That is, you can think of the utility as treating a sheetpath part as sheet during its operation.

The /E switch causes the utility program to display the message "Type Any Key To Continue", enabling the system to pause for you to remove the DRIVER/LIBRARY disk in drive B and insert the SHEET disk. This switch is used only on systems with two floppy disk drives.

The /F switch causes the NETLIST program to read the source as a text file, for flat file structure applications.

The /G switch checks the worksheet for Parts, Sheets, Labels, Module Ports, and Power objects that are placed off grid and reports them.

NETLIST creates a list of off-grid items and places that list in a file called source.GRD.

The /H switch removes all duplicate sheets in a complex hierarchy. This switch is only used when generating a

The /P switch outputs pin numbers instead of pin names in FutureNet and EDIF formats.

The /Q switch causes the NETLIST program to run "quietly". This means that only the invocation messages and error messages if any, are displayed. If this switch is not specified, the program will display intermediate tracking activity.

The /S switch is used to generate one of the special netlist formats. If the switch is not specified, the netlist is generated in EDIF format. Enter one of the following formats as listed below, along with the /S option, when invoking the NETLIST utility.

APPLICONBRAVO
APPLICONLEAP
ALGOREX
CALAY @
CADNETIX
COMPUTERVISION
EDIF
EEDESIGNER
FLATEDIF
FUTURENET *
INTERGRAPH
MULTIWIRE
PCAD
SALT
SPICE %
SCICARDS
RACALREDAC @
TANGO
TELESIS
VECTRON*
WIRELIST

@Creates two output files: components and nets
%Creates two output files: nets and net map
*PINLIST or NETLIST

NETLIST *subsheet.sch* /O

Where *subsheet.sch* is the path and name of the sub-sheet in the hierarchy, /O signifies that the subsheet.sch file name is a single sheet.

3. To obtain a netlist of a hierarchical schematic and direct the output of the NETLIST program to a file:

   NETLIST *root.sch whatfile*

   Where *root.sch* is the path and name of the root sheet in the hierarchy and *whatfile* is the path and file name to place the NETLIST information.

4. To obtain a netlist of a hierarchical schematic in CALAY format and direct the output of the NETLIST program to a file:

   NETLIST *root.sch whatfile* CALAY /S

   Where *root.sch* is the path and name of the root sheet in the hierarchy, *whatfile* is the path and file name to place the NETLIST information, CALAY is the desired format, /S signifies that a special format is desired (Calay in this example).

6.10.2 Invocation Examples Using Flat File Structures

1. To obtain a netlist of a flat file structure containing multiple sheets:

   NETLIST *flatfile.txt* /F

file, /S signifies that a special format is desired (Racal-Redac in this example).

6.10.3 Invocation Examples Using A One Sheet File Structure

1. To obtain a netlist of a single sheet schematic:

   NETLIST *sheetname.sch* /O

   Where *sheetname.sch* is the name of the single sheet schematic, /O is used to signify that sheetname.sch is a single sheet schematic.

2. To obtain a netlist of a single schematic and direct the output of the NETLIST program to a file:

   NETLIST *sheetname.sch whatfile* /O

   Where *sheetname.sch* is the name of the single sheet schematic, *whatfile* is the path and file name to place the NETLIST information, and /O is used to signify that sheetname.sch is a single sheet schematic.

3. To obtain a netlist of a single schematic in SCICARDS format and direct the output of the NETLIST program to a file:

   NETLIST *sheetname.sch whatfile* SCICARDS /O /S

   Where *sheetname.sch* is the name of the single sheet schematic, *whatfile* is the path and file name to place the NETLIST information, SCICARDS is the desired format, /O is used to signify that sheetname.sch is a single sheet the connection of signals across sheet boundaries. In this phase, there is a limit of about 1,000 signal names and a limit of about 1,000 to 6,000 total nets. For a flat file structure, this phase combines all sheets at the same time. If there is not enough memory for this phase to complete, then the design must be made hierarchical.

For a hierarchical design, an incremental resolution is made and the limits outlined above apply to each pass of the resolution process. This means that a much larger design can be handled with the hierarchical structure than the flat file structure.

6.10.6 Notes on Particular Formats

Many netlist formats require you to follow special pin numbering conventions. For example, OrCAD libraries have some of the pin names as text. These may need to be converted to whole numbers, depending on the desired netlist format. To convert pin names to whole numbers, see the procedures described in the SPICE netlist, below.

FUTURENET

The FutureNet system has two connectivity output formats, PINLIST and NETLIST. The PINLIST format is a part-oriented list that lists for each pin of a part, the net to which the pin is attached. The NETLIST is a net-oriented netlist that lists for each net, the part pins that are attached. The NETLIST format is extensively sorted by FutureNet. OrCAD does not do all of this sorting.

The FutureNet output is limited by the amount of available system memory. This is due to the requirement that the parts and nodes need to be cross referenced. The limit is approximately 8,000 parts and 12,000 total nets. This should be enough for most necessary for the PCAD PCB layout software. Because some versions of the PCAD utility software will not link the OrCAD/SDT III PCAD compatible netlist with the BINARY layering file, a second example shows how the OrCAD/SDT III PCAD compatible ASCII netlist file can be merged with an ASCII layering file.

This netlist format may require pin numbers instead of pin names in the library source file. In this case, you may have to modify the OrCAD supplied DEVICE.LIB library, as outlined below in the SPICE discussion.

Example One: Changing the netlist "ENVIRONMENT" statement

The sample netlist below is an example of the standard header line generated by OrCAD's NETLIST utility. In order for an OrCAD/SDT III PCAD netlist to be compatible to the PCAD PCB layout software the header line must be modified to include a PCAD BINARY layering file such as the generic 'LAYS.PCB'.

```
  {COMPONENT OrCAD.NET {ENVIRONMENT OrCAD.SDT}
{DETAIL {SUBCOMP
  {I 2114.PRT U6 {CN
      1 ADDRESS6
      2 ADDRESS5
      3 ADDRESS4
      4 ADDRESS3
      5 ADDRESS0
      6 ADDRESS1
      7 ADDRESS2
      8 XN00002
      9 VSS
     10 WR*/RD_1
     11 D3_1
     12 D2_1
     13 D1_1
     14 D0_1
     15 ADDRESS9
     16 ADDRESS8
     17 ADDRESS7
     18 VDD
  }}
  }}}
```

```
     {lyrstr "PADCOM" 7 "FLCOMP" 7 "PADSLD" 8
           "FLSOLD" 8 "PADINT" 9
           "FLINT" 9 "GNDCON" 10 "FLGCON" 10
           "GNDCLR" 12 "FLGCLR" 12
           "PWRCLR" 13 "FLPCLR" 13 "SLDMSK" 14
           "FLSMSK" 14 "DRILL" 15
           "FLDRLL" 15 "PIN" 4 "BRDOUT" 4 "FLTARG" 4
           "SLKSCR" 6
           "DEVICE" 5 "ATTR" 6 "REFDES" 6 "COMP" 1
           "SOLDER" 2 "INT1" 3}
     }
    {USER
     {PCAD
      {Vw 750 350 8}
      {Lv 24 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 1
1 0 0 2 2 0}
      {Gs 50 50}
     }
    }

    {DISPLAY
     [Ly "COMP"]
     [Ls "SOLID"][Wd 15]
     [Ts 125][Tj "CC"][Tr 1][Tm "N"]
            }

    {SYMBOL
     {PIN_DEF
     }

     {PIC
     }

     {ATR
      {IN
       {Org -32767 -32767}
       {Ty 255}
      }
     }
    }
```

resultant file can be converted back into a BINARY formatted file by using the PDIF-IN utility available from PCAD. The resulting file from the PDIF-IN utility is ready for use with the PCAD PCB layout software. It should be noted here that the file name given in the COMPONENT statement of the ASCII layering file should be changed to avoid the possibility of overwriting the existing BINARY layering file

```
  "LAYS.PCB".

  {COMPONENT LAYS.PCB

   {ENVIRONMENT
    {PDIFrev 1.30}
    {Program "PC-CARDS Version 0.02"}
    {DBtype "PC-Board"}
    {DBrev 1.00}
    {DBtime "Oct. 22, 1985       2.03 p.m.       "}
    {DBunit "MIL"}
    {DBgrid 1}
    {lyrstr "PADCOM" 7 "FLCOMP" 7 "PADSLD" 8
          "FLSOLD" 8 "PADINT" 9
          "FLINT" 9 "GNDCON" 10 "FLGCON" 10
          "GNDCLR" 12 "FLGCLR" 12
          "PWRCLR" 13 "FLPCLR" 13 "SLDMSK" 14
          "FLSMSK" 14 "DRILL" 15
          "FLDRLL" 15 "PIN" 4 "BRDOUT" 4 "FLTARG" 4
          "SLKSCR" 6
          "DEVICE" 5 "ATTR" 6 "REFDES" 6 "COMP" 1
          "SOLDER" 2 "INT1" 3}
    }

    {USER
     {PCAD
      {Vw 750 350 8}
      {Lv 24 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0
1 1 0 0 2 2 0}
      {Gs 50 50}
     }
    }

    {DISPLAY
     [Ly "COMP"]
     [Ls "SOLID"][Wd 15]
     [Ts 125][Tj "CC"][Tr 1][Tm "N"]
    }

    {SYMBOL
     {PIN_DEF
```

SPICE:

OrCAD can create a netlist that is larger than most PC based Spice programs will accept. Consult your Spice manual for the limits. If your PC meets the memory requirements of Spice, the largest Spice netlist is capable of being generated. This should work for all PC versions of Spice.

The part value is used to pass modeling information to the netlist. Use the special PSPICE.LIB or SPICE.LIB libraries supplied by OrCAD when generating a Spice netlist. If you desire to modify or create your own Spice library, you must support the proper model pin numbers. To implement this, use the DECOMP or LIBEDIT utility programs and convert the desired OrCAD supplied libraries to a source file (refer to Section 7). Make the appropriate changes to the source file and recompile the modified library back to an object file using COMPOSER or LIBEDIT.

All library part pin names should be changed to reflect the model node index. To find out the proper ordering node, refer to your Spice manual.

As an example of what to change, the OrCAD supplied NPN transistor has the pin names defined as base, emitter, collector in the DEVICE.LIB library. For Spice to understand the nodal information, the pin names must be changed from base, emitter, collector; to 2,3, and 1 (as defined in the Spice manual). Therefore, the library source file for the NPN will be as follows:

supplied DEVICE.LIB library, as outlined above in the SPICE discussion.

6.10.7 Applications of Labels

Labels are used to connect signals together from one worksheet area to another without using wires or buses. They are also mandatory for labeling buses when they are placed in the worksheet. As a rule, you may place any number of labels on a bus or wire.

For example, assume you have a signal labeled ABC in the worksheet and you would like to connect another object in the worksheet to the same signal. Instead of drawing a wire from ABC on one side of the worksheet to the other object, you can label each signal with a Label as shown in Figure 6-6.

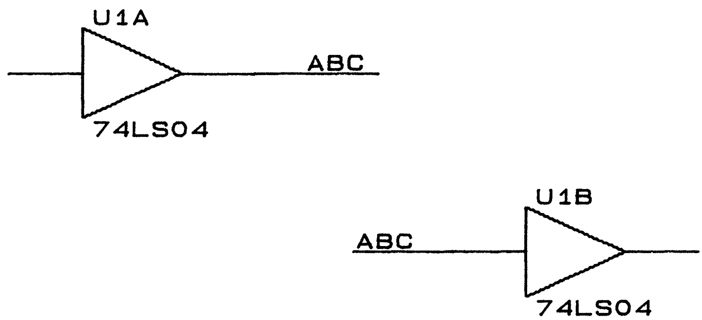


Figure 6-6. An Example of Using a Label

Labels are also used for buses. Every bus must be labeled for the NETLIST program to properly associate the bus with the individual members of the bus.

Bus labels must be in the form:

BUSNAME[0..n]

Where BUSNAME is called the "prefix" and it represents the name of the bus. [0..n] is called the

Then its members are labeled as follows:

BUSNAME*x*

x is a decimal number in the range of [0 - 9]. BUSNAME and x must have no space between them. In general, the prefix of the bus member's label must be the same as the prefix of the bus's label. Also, the appended decimal number must be within the bus's range.

An example of the use of bus and signal labels is illustrated in Figure 6-8. Label A[0..9] is placed on the bus. Labels A0 through A9 are for the bus signals. Notice the letter A in the labels. The prefix for the bus's label is the same as the prefix for each of the bus signal labels.

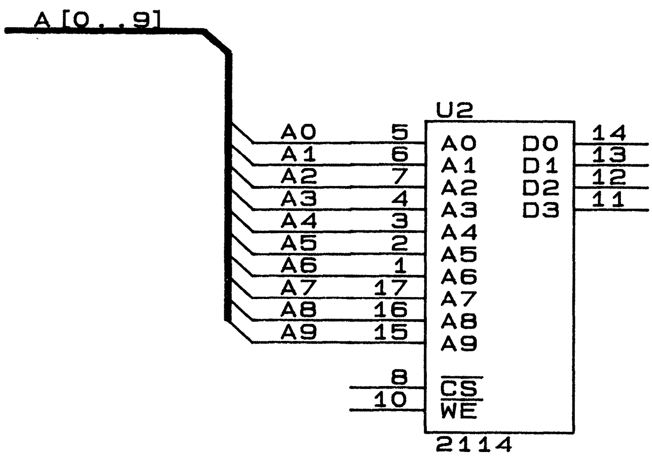


Figure 6-8 Using Labels for Buses and Bus Members

6.10.9 Applications of Module Ports

Any signal that is to go off the worksheet must do so via a module port. In a flat file structure, all module ports are considered to be global. In a hierarchical structure,

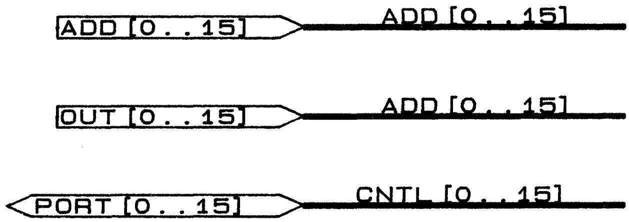


Figure 6-9 Connecting Buses to Module Ports

Signals (individual wires or non-bus signals) may be connected directly to module ports. Module ports that are connected to non-bus signals may be named in any format. They are not required to have a suffix. Figure 6-10 illustrates typical examples of signals that are connected to module ports.

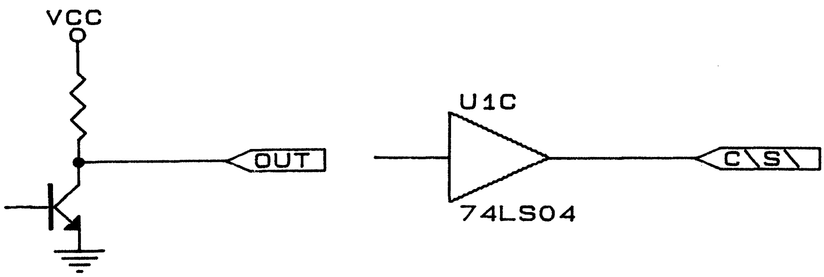


Figure 6-10 Connecting Signals to Module Ports

6.10.10 Splitting Buses

A special feature of DRAFT, enables you to split buses in your worksheet. Figure 6-11 illustrates how to do this. In this figure, the bus labeled A/D[0..15] is attached to a module port having the same name. Members of this bus are connected to U7 through the labels A/D0 through A/D15.

6.10.11 Multiple Labels on a Bus

A bus may have more than one label placed on it. In actual applications, bus labels may be placed anywhere on the bus and still be associated with their respective bus signal labels.

6.10.12 Combining Labels

Figure 6-12 illustrates an example of combining labels. Label MEM[0..11] is placed on the bus which contains 12 members. U1 is connected to the bus via labels MEM0 through MEM11.

Notice on the left side of the figure that the label MEM10 has another label C\S\ placed next to it, and that the label MEM11 has another label W\E\ placed next to it.

Also, C\S\ and W\E\ are labels that have been placed on pins 8 and 10 of U2 and U3. The example shows how to connect signals MEM10 and MEM11 to U2, by labeling them appropriately as C\S\ and W\E\.

In the case of U3, the C\S\ and W\E\ signals are connected to the 2114 device without being physically connected to the bus.

Labels MEM0 through MEM9 on U2 and U3 connect the 2114 address lines (A0 - A9) to the bus.

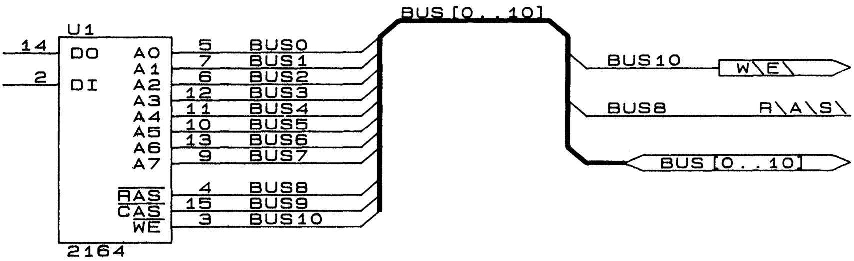


Figure 6-13. Connecting Labels to Module Ports

6.10.14 Handling and Isolating Power

Power connections are handled in a number of ways. Most parts in the libraries supplied by OrCAD have defined power and ground pins. These pins are hidden from the drawing, but nevertheless are part of the symbol definition.

To make connections from the outside world to the hidden power pins in the library part, a power object is used (PLACE Power command). For example, assume that you have a CMOS device placed in the worksheet. This device has been defined to have a VDD and VSS power pin in the library source file. If you want another signal from the outside world connected to the same VDD2 potential as in the CMOS device, just connect that signal to a power object named VDD.

Power objects are global. Global means that a signal (power in this case) connects to other signals that are global having the same name. This is true for any worksheet file structure. The NETLIST utility connects all similar power names and signals together. The power handling ability of OrCAD/SDT III, makes it extremely easy to isolate different power sources.

The NETLIST program will treat certain parts in the library as a power object if they are defined a special way. The four types of grounds in the DEVICE.LIB select the PLACE command, and change the "Value" to be whatever is required.

NOTE

> To find power pin numbers on library components, use the LIBRARY Browse command.

6.10.15 Connecting Different Power Object Names

In the OrCAD-supplied libraries, many of the devices have been defined to have VCC as the positive supply voltage pin. Others have VDD defined as the positive supply voltage pin. If you want them both to operate from the same power supply, you must connect them together.

Likewise, many of the libraries have GND and VSS defined as the return power pins. If you want to have them connected together and be common to each other, you must connect them together also.

To connect power supply pins together, or connect a power supply pin to any other supply voltage, you must place a power object for each different supply in the worksheet. One power object must be named the same as one of the supply voltages, VDD for example. The other power object must be named the same as the remaining supply voltage, VCC for example. Finally, the power objects that are placed in the worksheet must be connected together with a wire. The following Figure illustrates four examples.

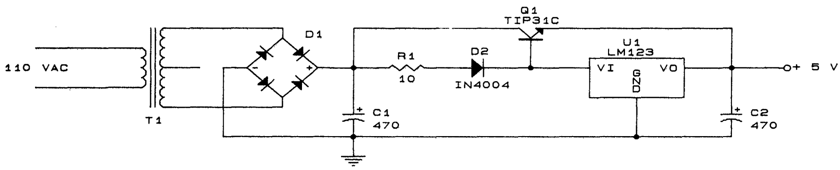


Figure 6-15. Power Supply with a Power Object

6.10.16 Connecting Power Objects to a Module Port

There are cases when you wish to isolate power in the worksheet. One way to do this would be to make all new parts with new power pin names. Not only would this be time consuming, but it would be very difficult to keep track of which parts are to be used on which schematic for any particular supply.

Isolating power without having to create new parts is done by connecting a module port to a power object. The NETLIST program will then supersede the use of the power object with a module port. Only the module port will be passed off of a sheet to isolate circuitry used on another worksheet.

If a power object is to transfer power from one worksheet to another, either in a flat file or hierarchical structure, it must be connected to an "unspecified" module port. Any other type of module port is not accepted by ERC. Three examples of connecting power objects to module ports are illustrated in Figure 6-16.

This design is was created as a three-sheet hierarchy. The root sheet, in Figure 6-17, contains the CPU and control circuitry of the design. Two hierarchical sheet symbols are also placed in the root worksheet. One sheet symbol represents the power supply; the other represents the memory that is to be battery backed up.

Notice that a VDD power object is placed in the root worksheet and is connected to a +5V power object. Since the 80C51 and the 82C82 power pins are labeled as VDD in their library source files, the +5V and VDD power objects will connect plus 5 volts from the power supply (Figure 6-18) to the VDD pins in the NETLIST output.

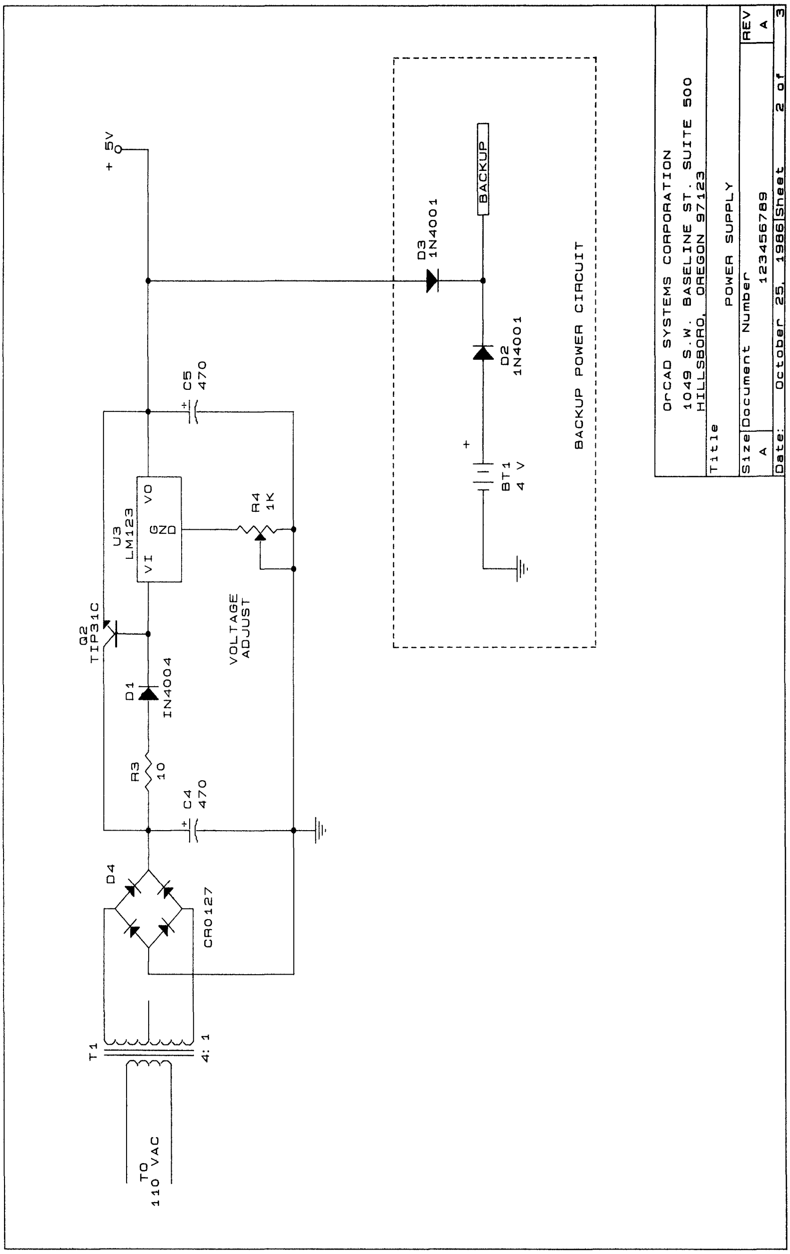


Figure 6-18. Power Supply Sheet

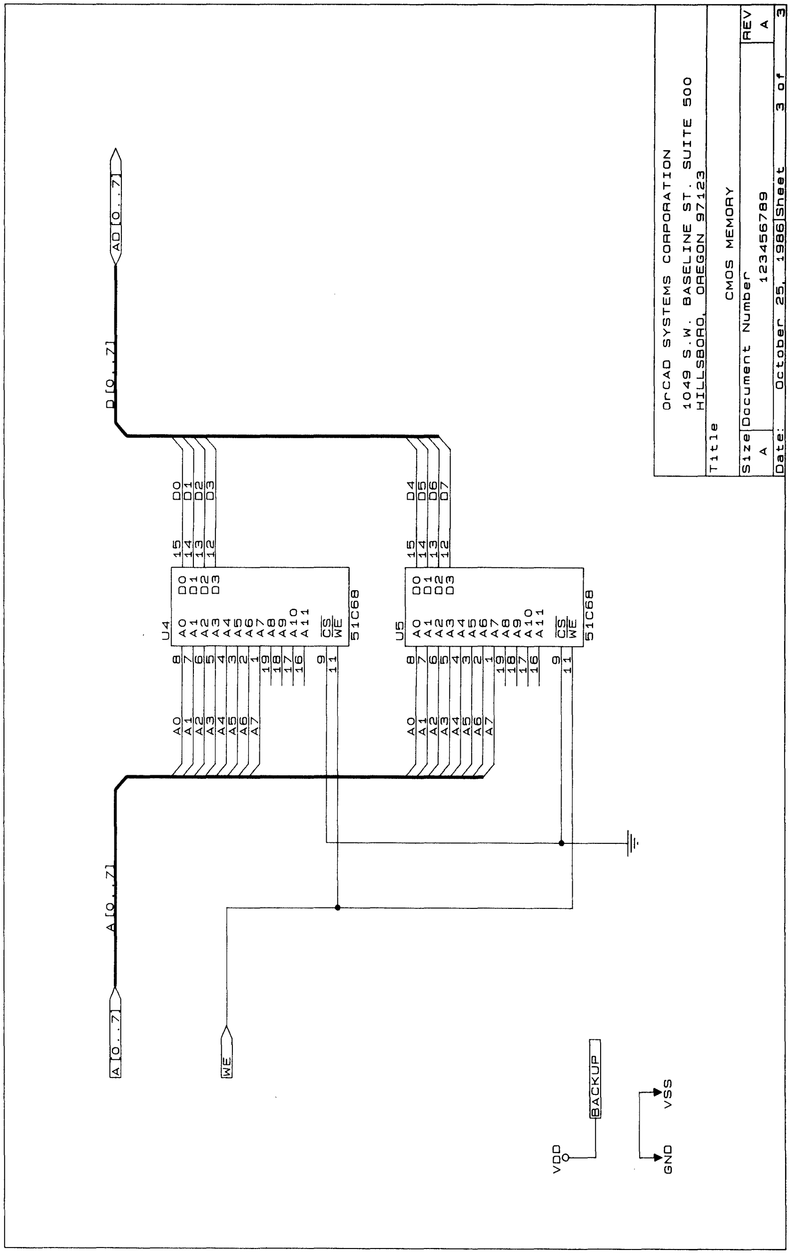


Figure 6-19. CMOS Memory Sheet

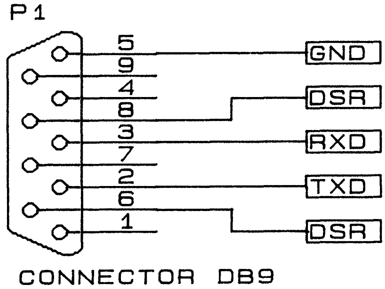


Figure 6-20. Handling Connectors

Making large connectors with alphanumeric pin names can easily be handled by making the connector a block symbol. To make a connector with alphanumeric pin names, make the part with zero "0" parts per package. Do not use the token GRIDARRAY or "1" part per package. Listed below is a library source example of an IBM 62 pin edge connector. Because of the large size of the example, only a portion is shown. Since there are no pin numbers defined in the source file, the NETLIST utility uses the pin name information. In this example, B1 through R31 would be used as the pin numbers in the netlist.

```
'CONNECTOR IBM'
REFERENCE 'J'
10   32   0
L1   PAS   'B1'
L2   PAS   'B2'
L3   PAS   'B3'
L4   PAS   'B4'
      .
      .
      .
L30  PAS   'B30'
L31  PAS   'B31'
R1   PAS   'A1'
R2   PAS   'A2'
R3   PAS   'A3'
R4   PAS   'A4'
      .
      .
      .
R30  PAS   'R30'
R31  PAS   'R31'
```

6.11 PARTLIST

Purpose:

Creates a summation of all parts used in a group of schematic sheets. The PARTLIST program will scan either a hierarchy, flat file, one sheet file structure, or the annotation file output from the ANNOTATE program.

Optionally, user specific part information may be added in a text, or "include file". If this user specific information is included, then the PARTLIST program will output the parts found in the order that they appear in the include file. Any parts not in the include file are placed at the end of the report.

Format:

> PARTLIST *source* [*destination*] [*include*]
> [/A][/C][/D][/E][/F][/I][/O][/Q][/S][/V]

Remarks:

> The source may be either the root sheet name of a hierarchical file structure, the name of a text file in a flat file structure, or the file name of a one sheet file structure.
>
> If the source is the name of a text file in a flat file structure, the /F switch must be included. If the source is the file name of a one sheet file structure, the /O switch must be included. If the source is the annotation file created by the ANNOTATE program, then the /A switch must be included on the invocation line.
>
> The destination is any valid DOS path name and is where the output of the program is to be placed.

The /I switch causes the PARTLIST program to include user supplied information in the output.

The /O switch causes the file name of the source to be read as a one sheet file structure.

The /Q switch causes the PARTLIST program to run "quietly". This means that only the invocation messages and error messages if any, are displayed. If this switch is not specified, the program will display intermediate tracking activity.

The /S switch causes the format of the output report to be single spaced rather than defaulted to double spaced.

The /V switch outputs a verbose format, which causes the header information to be included on every page.

6.11.1 Invocation Examples Using Hierarchical Structured Files

1. To create a parts list of a hierarchical schematic:

   PARTLIST *root.sch*

   Where *root.sch* is the path and name of the root sheet in the hierarchy.

2. To obtain a parts list of a sub-sheet in a hierarchical schematic:

   PARTLIST *subsheet.sch* /O

   Where *subsheet.sch* is the path and name of the sub-sheet in the hierarchy, /O signifies that the subsheet.sch file name is a single sheet.

3. To direct the output of the PARTLIST program to a file:

3. To create a parts list of a flat file structure and direct the output to a file:

   PARTLIST *flatfile.txt whatfile* /F

   Where *flatfile.txt* is a text file containing a list of schematic file names to be checked, *whatfile* is the path and file name to place the PARTLIST information, /F is used to signify that flatfile.txt is a text file.

4. To create a parts list of a flat file structure, direct the output to a file, and include additional information:

   PARTLIST *flatfile.txt whatfile include.txt* /F /I

   Where *flatfile.txt* is a text file containing a list of schematic file names to be checked, *whatfile* is the path and file name to place the PARTLIST information, *include.txt* is the name of the include file, /F is used to signify that flatfile.txt is a text file, and /I signifies that an include file is specified.

6.11.3 Invocation Examples Using A One Sheet File Structure

1. To create a parts list of a single sheet schematic:

   PARTLIST *sheetname.sch* /O

   Where *sheetname.sch* is the name of the single sheet schematic, /O is used to signify that sheetname.sch is a single sheet schematic.

2. To create a parts list of a single schematic and direct the output to a file:

   PARTLIST *sheetname.sch whatfile* /O

The header line begins with a pair of single quotes with no characters or space between them. The remainder of the line contains the header information that you want to include. For each part to include information, a separate line in the file is created which begins with the part name as it appears in your worksheet. The name must be enclosed within single quotes (such as '74LS00').

After you enter the part name, place the information on the same line that you want to be included ("Resistor 1/4 Watt 5%" for example). For both types of lines, header and part, the line will be left justified to the first non-space character of the information portion of the line. When the PARTLIST program has finished scanning the sheets, it then scans the include file to include the rest of the line after any part name that matches. The following is a sample of the include file format.

```
''         DESCRIPTION                   Part Order Code
'1K'       Resistor 1/4 Watt 5%              10000111003
'4.7K'     Resistor 1/4 Watt 5%              10000114703
'22K'      Resistor 1/4 Watt 5%              10000112204
'1uf'      Capacitor Ceramic Disk            10000211006
'.1uf'     Capacitor Ceramic Disk            10000211007
'.01uf'    Capacitor Ceramic Disk            10000211008
'.001uf'   Capacitor Ceramic Disk            10000211009
'7400'     TTL Quad Two Input NAND Gate 10001040000
'74LS00'   TTL Quad Two Input NAND Gate 10002040000
'74S00'    TTL Quad Two Input NAND Gate 10003040000
'74ALS00'  TTL Quad Two Input NAND Gate 10004040000
'74AS00'   TTL Quad Two Input NAND Gate 10005040000
'7402'     TTL Quad Two Input NAND Gate 10001040002
'74LS02'   TTL Quad Two Input NAND Gate 10002040002
'74S02'    TTL Quad Two Input NAND Gate 10003040002
'74ALS02'  TTL Quad Two Input NAND Gate 10004040002
'74AS02'   TTL Quad Two Input NAND Gate 10005040002
```

```
Motor Drive Circuitry                 Revised: June 22, 1986
786-256A-001                          Revision: 1
Bill Of Materials   September 15, 1986  14:48:48  Page 1

Item Quantity Reference Part DESCRIPTION Part Order Code
________________________________________________________

 1    1    U1     8051   Intel Controller     10002048505
 2    1    X1     12 mHz Crystal              10000820006
 3    2    C1,C2  30 pf  Capacitor Mica       10000483736
 4    1    C3     20 uF  Capacitor Tantalum   10000486353
 5    1    U3     2732   32K EPROM            10002734645
 6    1    U4     8282   Latch                10008475663
 7    2    R2,R4  6.8K   Resistor 1/4 Watt 5% 10038437622
 8    1    R3     470    Resistor 1/4 Watt 5% 10038437862
 9    1    U5     74LS73 J-K Flip Flop        10008756353
10    1    U6     74LS86 2 input EX-OR Gate   10008756349
11    1    FUSE1  2 AMP  Slow Blow Fuse       15000063731
12    1    T1     SC140  2 AMP Triac          10040000295
13    1    R8     47     Resistor 1/4 Watt 5% 10038437023
14    2    D2,D3  1N4004 Diode                10092735660
15    2    Q2,Q3  TIP110 Power NPN            12000838388
16    1    R9     5 ohm  Resistor 1/4 Watt 5% 10038430005
17    1    SW1    SpSt   B&K 100 MA Switch    10842100954
```

The /C switch causes the configuration menu to be invoked. This allows the OrCAD/SDT III environment to be modified.

The /D switch causes PLOTALL to descend into parts defined as sheetpath parts. That is, you can think of the utility as treating a sheetpath part as sheet during its operation.

The /E switch causes the utility program to display the message "Type Any Key To Continue", enabling the system to pause for you to remove the DRIVER/LIBRARY disk in drive B and insert the SHEET disk. This switch is used only on systems with two floppy disk drives.

The /F switch causes the PLOTALL program to read the source as a text file, for flat file structure applications.

The /G switch causes GRID REFERENCES to be included in the sheet when plotting.

The /O switch causes the file name of the source to be read as a one sheet file structure.

The /P switch plots the worksheet to a configured printer instead of plotter (Make sure you have adequate disk space. This utility creates temporary files).

The /Q switch causes the PLOTALL program to run "quietly". This means that only the invocation messages and error messages if any, are displayed. If this switch is not specified, the program will display intermediate tracking activity.

The /S switch scales the plot by a user specified scale factor entered at invocation time. The scale factor is a schematic file names to be plotted, /F is used to signify that flatfile.txt is a text file.

2. To plot one sheet in a flat file structure:

PLOTALL *sheetname.sch* /O

Where *sheetname.sch* is the name of the single sheet in the flat file structure, /O is used to signify that sheetname.sch is a single sheet schematic.

6.12.3 Invocation Examples Using A One Sheet File Structure

To plot a single sheet schematic:

PLOTALL *sheetname.sch* /O

Where *sheetname.sch* is the name of the single sheet schematic, /O is used to signify that sheetname.sch is a single sheet schematic.

6.12.4 Invocation Example - Directing The Output To A File

To direct the output of the PLOTALL program to a file:

PLOTALL *sheetname.sch whatfile* /O

Where *sheetname.sch* is the name of the single sheet schematic, *whatfile* is the path and file name to place the PLOTALL information, and /O is used to signify that sheetname.sch is a single sheet schematic.

To configure OrCAD/SDT III for the plotter driver, select the letter that corresponds to the plotter driver that you are using at the "Enter Letter to Select the Plotter to be used->" prompt. If your plotter name does not appear, press <S> for Special. When selected, the plotter driver name only appears on the "CONFIGURATION OF OrCAD/SDT III" menu.

To modify the serial channel, baud rate, parity, or word length configuration, press the colon <:> key at the "Enter Letter to Select the Plotter to be used->" prompt. Enter a "1" for serial channel 1, or a "2" for serial channel 2 at the "Channel 1 or 2 ->" prompt.

After you select the serial channel, you may select the baud rate, parity, and word length that is required.

To modify the baud rate, enter the letter that corresponds to the baud rate you require at the "Baud Rate ->" prompt.

To modify the parity, enter the number that corresponds to the parity you require at the "Parity ->" prompt.

To modify the word length, enter the number that corresponds to the word length you require at the "Word Length ->" prompt.

When you return to the "CONFIGURATION OF OrCAD/SDT III" menu, be sure to update the information by pressing the letter <U>.

A typical configuration for HP, HI, and IOline plotters is:

Baud Rate: 2400 or 9600
Parity: No Parity
Word Length: 8 bits

Use the driver HI.DRV for IOline plotters.

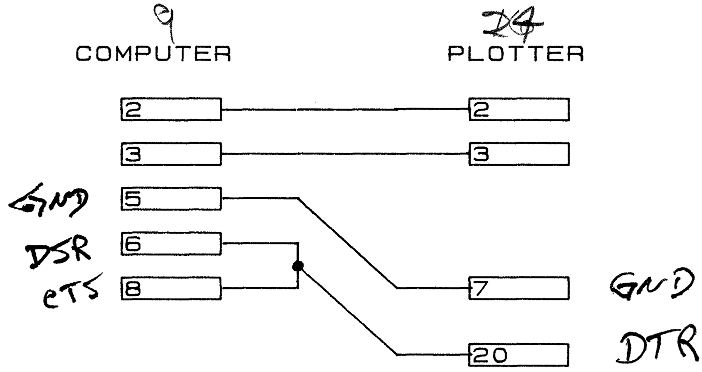


Figure 6-22. PC AT 9-Pin Cable Wiring Diagram

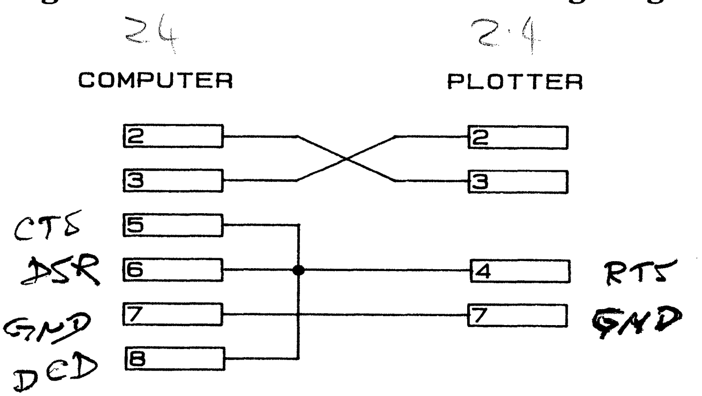


Figure 6-23. PC/XT 25-Pin Cable to IOline Plotter

6.12.8 Plotter Problems

Most plotter problems are typically a result of incorrectly wired plotter cables. If you have difficulty with your plotter, check the following items before proceeding or calling OrCAD:

1. Wire the cable as shown in Figures 6-21, 6-22, and 6-23 above. If your plotter works with another software package, and does not work with OrCAD, the first item to check is the wiring of your cable. Chances are, the other CAD

This assumes that you are using serial channel 1 (COM1) and have your plotter set for 2400 baud, no parity, 8 data bits, and 1 stop bit. For more information on the MODE command, refer to your DOS users guide for Asynchronous Communications.

After the serial channel has been configured, send the plot file to the plotter using the DOS COPY command as follows:

COPY *whatfile* COM1: <ENTER>

Where *whatfile* is the name of the plot file.

If the plotter works, this indicates the problem may be in the plotter cable (incorrectly wired), or the hardware handshaking is incorrectly set (check PLOTALL configuration).

If the plotter does not work, this indicates that there is a hardware problem. Check the the following: serial card, incorrect serial channel configuration, plotter hardware, or a cable problem.

6. If yours is an IOline plotter, be sure you have PROM version 114 or greater.

6.12.9 Output Scaling

OrCAD controls the output scaling using the /S switch, at invocation time.The /S switch scales the plot by a user specified scale factor entered at invocation time. The scale factor is a decimal number in the form: #.###, not all trailing digits are needed. No check is made if the plot will fit on the plotter, nor are multiple sheets plotted if they are too big.

example, you can plot a C size worksheet on B, C, or D and it will work fine. If you plot an E worksheet on A paper, you will not be able to read the writing. If you plot an A worksheet on E size paper, the bit mapped devices will be "grainy".

When you are directed by the program to change paper or pens, always wait until the plotter has finished the present plotting activity. Before sending a plot directly to the plotter, be sure that the plotter is on line, the pen(s) are properly set up, and the paper size is correct. When you have a pen that must be manually changed, the PLOTALL program will pause and inform you of the objects to be plotted with the new pen.

6.12.11 HP Plotters

The HP plotter family has a facility to set the corner points of the plot and automatically scale the plot to be within these points. These points are called P1 and P2.

NOTE

> PLOTALL draws in .001" resolution and ignores the preset P1 and P2 values

PLOTALL assumes the origin of the plot is the lower left corner of the page (when the finished plot is viewed). Rotation and paper size must be set before you run a plot.

If the plotter's origin (0,0) is not the lower left corner, it may be moved via the template table in the configuration menu. To move the origin, configure the plot X-offset and Y-offset (integer value -32.768" to +32.767") in the template table. Note that this origin

Press <C> <T> to obtain the "Color Table / Plotter Pen Table". At the "Command ->" prompt, press the <P> <M> keys. Then, press <9> <9> followed by the <ENTER> key. The plotter pen is now IGNORED for drawing the worksheet title block and border.

If you have the 150 dpi driver configured, then the output resolution is 150 dots per inch.

The /A switch causes the PRINTALL program to read the source path as an annotation file.

The /C switch causes the configuration menu to be invoked. This allows the OrCAD/SDT III environment to be modified.

The /D switch causes PRINTALL to descend into parts defined as sheetpath parts. That is, you can think of the utility as treating a sheetpath part as sheet during its operation.

The /E switch causes the utility program to display the message "Type Any Key To Continue", enabling the system to pause for you to remove the DRIVER/LIBRARY disk in drive B and insert the SHEET disk. This switch is used only on systems with two floppy disk drives.

The /F switch causes the PRINTALL program to read the source as a text file, for flat file structure applications.

The /G switch causes GRID REFERENCES to be included in the sheet printout.

The /O switch causes the file name of the source to be read as a one sheet file structure.

The /Q switch causes the PRINTALL program to run "quietly". This means that only the invocation messages and error messages if any, are displayed. If this switch is not specified, the program will display intermediate tracking activity.

The /W switch causes the printing to formatted for wide paper. With this switch, the printing will be setup for 13" wide paper based on the parameters of the sheet in the flat file structure, /O is used to signify that sheetname.sch is a single sheet schematic.

6.13.3 Invocation Examples Using A One Sheet File Structure

1. To print a single sheet schematic:

   PRINTALL *sheetname.sch* /O /S

   Where *sheetname.sch* is the name of the single sheet schematic, /O is used to signify that sheetname.sch is a single sheet schematic.

2. To direct the output of the PRINTALL program to a file:

   PRINTALL *sheetname.sch whatfile* /O

   Where *sheetname.sch* is the name of the single sheet schematic, *whatfile* is the path and file name to place the PRINTALL information, and /O is used to signify that sheetname.sch is a single sheet schematic.

NOTE

Since whatfile is a binary print file, it will consume an extensive amount of disk space.

The file whatfile may be sent to a printer using the DOS COPY Command. For example, enter the following at the DOS prompt:

COPY whatfile prn: /b

6.14 TREELIST

Purpose:

Scans a hierarchical file structure of schematics to display the sheet names, sheet path names, and optionally the date of last modification. This utility is useful for organizing and keeping track of the hierarchical worksheets.

Format:

TREELIST *source* [*destination*]
[/A][/C][/D][/E][/F][/Q]

Remarks:

The *source* is the name of the root sheet of a hierarchical organization of schematics. Only a sheet name will be accepted.

The *destination* is any valid DOS path name and is where the output of the program is to be placed. If a path is not specified, the output will be directed to the console.

Libraries are not required to be loaded for operation of this utility program. Therefore, the /E switch is not required.

The /A switch causes the program to read the source path as an annotation file. This will cause the annotation file to be updated.

The /C switch causes the configuration menu to be invoked. This allows the OrCAD/SDT III environment to be modified.

The /D switch causes the program to include the date of last modification in the output for each sheet scanned in and file name to place the TREELIST information.

6.14.2 Sample Output from the TREELIST Utility

```
Scanning                    "SHEET\286sys.sch"
Scanning                    "SHEET\286proc.sch"
Scanning                    "SHEET\286io.sch"
Scanning                    "SHEET\286hd.sch"
Scanning                    "SHEET\286fd.sch"
Scanning                    "SHEET\286lan.sch"
Scanning                    "SHEET\286gd.sch"
Scanning                    "SHEET\286mem.sch"
   <<<Root File>>>
   [286sys.sch]                  February 23, 1986
     Processor and Control
     [286proc.sch]                    August 28,
1986
     Peripheral Interface
     [286io.sch]                 January 20, 1986
        Hard Disk Interface
        [286hd.sch]              October  7, 1986
        Floppy Disk Interface
        [286fd.sch]              December  6, 1986
          LAN Interface
          [286lan.sch]             <empty
worksheet>
          Graphics Display Subsystem
          [286gd.sch]              May 31, 1986
       Memory Array
       [286mem.sch]                June 16, 1986
```

DISCUSSION

The first eight lines illustrate the TREELIST program scanning each sheet in the hierarchy retrieving pertinent information.

Text enclosed within brackets [] represents the file name of the hierarchical sheets, [286sys.sch] for example. The date represents the date of the last sheet modification.

Text placed below the worksheet file name and date, on the same indentation level, represent the names of sheet symbols that are placed in worksheet. "Processor

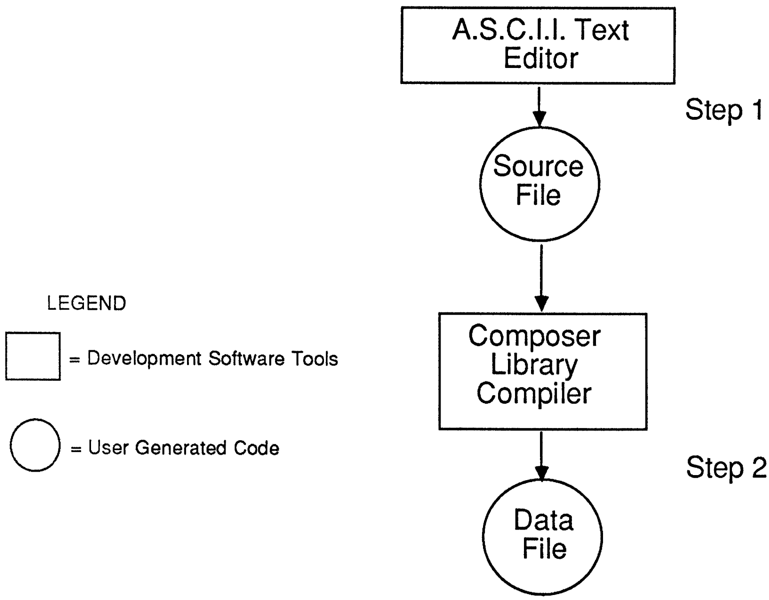


Figure 7-1. The Library Development Process Using COMPOSER

7.1 OVERVIEW OF ORCAD LIBRARIES

OrCAD supplies a number of part libraries for use with ORCAD/SDT. OrCAD-supplied libraries are shipped as library data files. This is for your convenience (data files are ready to use) as well as to save disk space. A library source file takes up much more disk space than its corresponding library data file. A source file can be four or five times as large as its data file.

Part of configuring ORCAD/SDT means choosing what libraries it will have access to. The libraries you choose at configuration time are loaded into RAM when you invoke ORCAD/SDT. This eliminates disk searching and provides for quick part retrieval.

Names

A part has one or more names. Parts have identical symbols are represented in a library as one part with multiple names. For example, 2114, 2146, 2149, etc. all identify the same symbol and represent a 1K X 4 static RAM.

Sheetpath designator

A sheetpath designator is a pathname that references a schematic file. By using a sheetpath designator, you can construct a part symbol and define it as previously constructed schematic. Sheetpath designators provide a higher level of abstraction for the circuit under construction and are useful for frequently used circuits.

Pins

Each pin has a type, shape, a number, and possibly a name.

Type. Typical pin types are input, output, bidirectional, etc. A pin can have only one type. Except for power, the pin type is not apparent from the representation of the part on the screen. Power pins are invisible.

Shape. The pin shape determines how the pin appears on the screen. A pin can have either a normal or a short lead. A normal lead can also have a clock symbol or an inversion bubble; a short lead cannot.

Number. The package number of a pin appears outside the part beside the lead.

Name. The pin name is associated with the pin function. For example, A0 represents an address line, and CLR represents the clear function.

Body

The body is either a block or a graphic. A block is a rectangle like a memory chip. A graphic has curved edge like an AND gate.

Because LIBEDIT, DRAFT and the other ORCAD/SDT utilities use the same overlay file (ORCADSDT.OVL), most users place LIBEDIT.EXE in the same directory as DRAFT.EXE, ORCADSDT.OVL, and the other utilities.

However, if you put LIBEDIT in a different directory, you must also place another copy of ORCADSDT.OVL in that directory; and you must then configure LIBEDIT separately. For example, configuring DRAFT changes the copy of ORCADSDT.OVL used by DRAFT, and configuring LIBEDIT changes the copy of ORCADSDT.OVL used by LIBEDIT.

To configure an OrCAD/SDT III utility, invoke it with the /c switch. For example, to configure LIBEDIT, type the following command:

LIBEDIT /c

The configuration menu then appears. This is the same configuration menu that appears when you configure DRAFT, even though some of the fields have no meaning for LIBEDIT. Refer to Section 2.? for detailed information about the configuration menu.

7.3.2 Invoking LIBEDIT

To invoke LIBEDIT, first make its directory the current directory by issuing the DOS **cd** command. Note that this section assumes that LIBEDIT.EXE is contained in the directory, C:\ORCAD. Then, type LIBEDIT. You can optionally follow LIBEDIT with the name of a library data file that you want to edit.

CD \orcad
LIBEDIT *filename*.LIB

If you leave out the name of a library data file, LIBEDIT prompts you for one. The operation is similar part from that library, you don't see the part's shape. The bitmap defining the shape is still there. It's just that LIBEDIT is not designed to deal with bitmaps.

If you call up a bitmap part and write it back to the library with the **LIBRARY Update Current** followed by **QUIT Update** or **QUIT Write**, you save the part as it appears on the screen -- that is, without a body. You have lost the bitmap. Before writing to the library, ensure that the part has a body as it appears on the screen.

If you use LIBEDIT to design a part with vectors and then write it out to the library, the part is stored both as a vector part and a bitmap part. The vector part is used for plotting, and the bitmap part is used for screen work. The reason for this is that bitmap parts are easier to manipulate on the screen and hence improve screen performance whereas vector parts result in higher quality plots. For example, vector-based parts don't show jagged curves.

7.3.5 LIBEDIT Commands

Many of the LIBEDIT commands are familiar to you from ORCAD/SDT. The main command menu consists of the following commands:

Again	Origin
Body	Pin
Conditions	Quit
Get Part	Reference
Jump	Set
Library	Tag
Macro	Zoom
Name	

7.3.7.1 BODY Line

Use the **Line** command to draw lines. This command is similar to the **PLACE Wire** command in ORCAD/SDT. When you select **Line**, the following menu appears

Begin
Jump
Origin
Tag
Zoom
escape

To draw a line, select **Begin**. LIBEDIT returns the following menu line:

Begin End New Jump Tag Zoom escape

To draw a line, select **Begin** and move the cursor. To complete the line, select **Begin**, **End** or **New**.

Begin	Ends the current line and begins a new line where the previous line ended.
End	Returns you to the main command menu.
New	Ends the current line. You can begin a new line at a different location with **Begin**.

The other items in the **Line** menu (**Jump**, **Origin**, **Tag**, and **Zoom**) appear in many LIBEDIT menus. They operate the same as their corresponding members in the main command menu. **escape** returns you to the **BODY** menu.

7.3.7.3 BODY Arc

Use the **Arc** command to place an arc within the part outline. An arc is a section of a circle. With the Arc command, you can draw an arc ranging from 0o to 90o.

When you select **Arc**, the **Arc** menu appears. It's identical to the **Circle** menu and contains the following items:

Center
Jump
Origin
Tag
Zoom
escape

To place an arc, move the cursor to where you want the center of the circle from which the arc is to be taken. Then, select **Center**. Another menu appears. It is identical to the **Edge** menu that appears under **Circle** and contains the following items:

Edge
Jump
Origin
Tag
Zoom
escape

Now when you move the cursor, you see the circle expand. You also see a radius extending from the center of the circle to the cursor. Move the cursor to where you want the one end of the arc to be. Select **Edge**. Then, move the cursor to where you want the other end of the arc to be.

Note that you can only move within a quadrant. (A circle is divided into four quadrants by horizontal and

Ensure that you place the text within the part outline. If any portion of the text lies outside the part outline, the results are unpredictable. Refer to Section 7.? "SET Show Body Outline" for more information abut part outlines.

Select **Place** to place the text. The *Text?* prompt then reappears, ready for you to enter another text item. If you press **<Enter>** without typing any text, the **BODY** menu returns.

The other items in the **Text** menu (**Jump**, **Origin**, **Tag**, and **Zoom**) appear in many LIBEDIT menus. They operate the same as their corresponding members in the main command menu. **escape** returns you to the **BODY** menu.

7.3.7.5 BODY IEEE Symbol

The **IEEE Symbol** places IEEE/ANSI special symbols within the part outline. For more information about these symbols, refer to ANSI/IEEE Std 91-1984.

When you select **IEEE Symbol**, a menu listing IEEE symbols appears. That menu is shown below. The indented items indicate submenus underneath particular IEEE symbols.

- Negation
- Active Low
    - Left
    - Right
- Arrow
    - Left
    - Right
- BiDirectional
- Dynamic
    - Left
    - Right
- Non-logic

The other items in the **IEEE Place** menu (**Jump**, **Origin**, and **Zoom**) appear in many LIBEDIT menus. They operate the same as their corresponding members in the main command menu.

7.3.7.6 BODY Fill

The **Fill** command shades the enclosed area around the current cursor position. For example, one use of the **Fill** command is to darken the symbols for diodes.

When you select **Fill**, the following menu appears:

Fill
Jump
Origin
Tag
Zoom
escape

To darken an enclosed area around the cursor, select **Fill** from this menu.

Note that if you edit a part further after using **Fill**, you lose your fills. This is actually a LIBEDIT feature. For example, if you remove the boundary on a filled area, the fill can take over the whole screen. There is no easy way to recover from such a fill takeover. Consequently, you must reserve the use of the Fill command for the end of your editing session.

The other items in the **Fill** menu (**Jump**, **Origin**, **Tag**, and **Zoom**) appear in many LIBEDIT menus. They operate the same as their corresponding members in the main command menu. **escape** in the **Fill** menu returns you to the **BODY** menu.

7.3.7.9 BODY Size of Body

The **Size of Body** command changes the size of the part being edited. When you select **Size of Body**, the following menu appears:

Place
Jump
Origin
Tag
Zoom

To increase or decrease the size of the part, move the cursor. When the size of the part is what you want it to be, select **Place**.

Whenever you decrease the size of a part, you always lose some detail in its representation. For example, restoring a part to its original size after you've shrunk it doesn't return the original version. For that reason, if you choose to decrease the size of the part, LIBEDIT asks for confirmation.

The other items in the **Size of Body** menu (**Jump**, **Origin**, **Tag**, and **Zoom**) appear in many LIBEDIT menus. They operate the same as their corresponding members in the main command menu. Note, however, that moving the cursor also changes the size of the part. Jumping to a tag is a quick way of resizing the part. **escape** in the **Size of Body** menu returns you to the **BODY** menu.

7.3.7.10 BODY Kind of Part

When you are building a part, you can chose whether is it a Block or a Graphic part. When you select **Kind of Body**, the following menu appears:

Block
Graphic

7.3.8 CONDITIONS

The **CONDITIONS** command provides information about LIBEDIT. It provides information about the following items: Free Library Objects, Free Symbol Memory, Free Body Memory, Free Vector Memory, Free Macro Buffer, and Free System Memory.

Free Library Objects	This number tells you how many names you can add to the current library. Note that the number represents names, not parts. Also, a part may have multiple names. However, the extra names that result from a prefix definition do not count.
Free Symbol Memory	This number represents the amount of bytes available for additional library symbols. LIBEDIT symbols include part names, pin names, pin positions, block symbols, etc.
Free Body Memory	This number represents the amount of bytes available for additional library bitmaps. Note that LIBEDIT does not use bitmaps. When you make a part with LIBEDIT, you use vectors instead. However, bitmaps are needed for screen work -- for example, DRAFT requires a bitmap to display a graphic part. Consequently, when you add a graphic part to a library a bitmap version is stored as well as a vector version.

7.3.10 GET PART

The **GET PART** command allows you to retrieve a part from a library for editing. When you select **GET PART**, the following prompt appears:

Get?

To retrieve a part from the selected library, type the name of the part and press **<Enter>**. The part then appears on the screen.

If you press **<Enter>** without specifying a part, LIBEDIT displays a list of part names. You can then select the desired part.

7.3.11 IMPORT PART

The **Import** command reads in a part from another file (an export file) created with the **Export** command. Export files can contain only one part definition. **Import** reads in that part definition as the current part. It overwrites any existing current part.

7.3.12 JUMP

Use the **Jump** command to quickly move the cursor to a different part of your drawing. When you select **Jump**, the following menu appears:

A tag
B tag
C tag
D tag
E tag
F tag
G tag
H tag
X location
Y location

Note that updating the library with the **Update Current** command modifies the copy of the library that's in memory, not the copy on disk. To modify the copy of the library on disk, use the **QUIT Update File** or the **QUIT Write to File** command.

Creating or editing a part consists of the following steps:

1. Invoke LIBEDIT and specify a library. LIBEDIT reads the library data file into memory.

2. Retrieve a part. Edit it. The copy of the library in memory remains unchanged.

3. Issue the **LIBRARY Update Current** command. The copy of the library in memory gets the new or modified part. You could instead decide to create an export file and not modify the library.

4. Issue the **LIBRARY Update File** or **LIBRARY Write to File** command. The copy of the library on disk gets the new or modified part. Issuing either of these commands without previously issuing **LIBRARY Update Current** results in no modification to the library, even if you have retrieved and modified or constructed a part.

7.3.13.2 LIBRARY List Directory

List Directory lists the names of all the parts in the library.

7.3.13.3 LIBRARY Browse

The **Browse** command in LIBEDIT operates the same as the **Browse** command in ORCAD/SDT. When you select **Browse**, the following menu appears:

All parts
Specific parts

7.3.13.4 LIBRARY Delete Part

Use the **Delete Part** command to delete parts from a library. When you select **Delete Part**, the following prompt appears:

Part?

To delete a part, type the name of the part and press **<Enter>**. If **SET Confirm Deletion** is set to **No**, the part is removed from the library. If **SET Confirm Deletion** is set to **Yes**, LIBEDIT asks you to confirm the deletion. Then, if you choose **Yes**, the part is removed from the library.

The *Part?* prompt returns. You can delete another part by typing its name and pressing **<Enter>**. If you press **<Enter>** without specifying a part, the **LIBRARY** menu returns.

7.3.13.5 LIBRARY Prefix

Use the **Prefix** command to edit the library's prefix definition. When you select **Prefix**, the following menu appears:

Add
Delete
Edit
Quit

ORCAD/SDT uses a library's prefix definition when you obtain a part with the **GET** command. Instead of entering the full name of a part, you can enter just the part's suffix. For example, if TTL.LIB is one of ORCAD/SDT's libraries, and you enter the suffix 04, a pop-up menu lists the following parts:

74LS04
74S04

best way to construct it is to read in the existing part, edit the part, rename it, and *then* update the library.

A part may have a list of names. For a part to be unique, each member in that list must be unique. For example, assume that you do the following:

- You retrieve part that has the names A, B, and C.
- You edit the part definition.
- You change one of its names (for example, C to D).
- You add the part (now with names A, B, and D) to the library.

The result is that the library now contains two parts -- one with the names A, B, and D (the new part) and one with the name C (the old part). When you add a new part, it replaces any existing part that has the same names. Because the new part doesn't have the name C, you don't overwrite that part. To delete C from the library, use the **LIBRARY Delete** command.

When you select **NAME**, the following menu appears:

Add
Delete
Edit
Prefix

7.3.15.1 NAME Add

To add a name to a part, select **Add**. The prompt *Name?* appears. Type the name you want to give the part followed by an **<Enter>**. The prompt *Sheet Path?* then appears. If the part is a sheet, type the pathname of the schematic file followed by an **<Enter>**. If the part is not a sheet, just press **<Enter>**.

7.3.17 PIN

Use the **PIN** command to add, delete, or edit pins. When you select **PIN**, the following menu appears:

Add
Delete
Name
Pin-Number
Type
Shape
Jump
Zoom
escape

7.3.17.1 PIN Add

To add a pin to a part, place the cursor where you want the new pin to be and select **Add**. LIBEDIT then queries you for the Name, Pin Number, Type, and Shape of the pin you are adding.

7.3.17.2 PIN Delete

To delete a pin in a part, place the cursor at the unwanted pin and select **Delete**. LIBEDIT then asks you to confirm the deletion. If you choose **Yes**, the pin is deleted. If you choose **No**, you return to the **PIN** menu.

Note that if **Confirm Deletion** under the **SET** command is set to **No**, LIBEDIT will not ask you to confirm the deletion of the pin.

7.3.17.3 PIN Name

Use the **Name** command to edit the pin string definition of an existing pin.

Power	A power pin is one that expects voltage or a ground. For example, on the 74LS00 NAND gate, pin 14 expects Vcc, and pin 7 should be connected to ground.
Open Collector	An open collector gate has the collector resistor omitted. This allows the collectors of several gates to be connected together with a single pull-up resistor. For example, pin 1 on the 74LS01 NAND gate is an open collector output.
Open Emitter	An open emitter gate has the emitter resistance omitted. The proper resistance must be added externally. An open emitter gate is used with ECL logic and is analogous to an open collector gate. For example, MC10100 is an open emitter gate.
Passive	A passive pin is typically one connected to a passive device. A passive device does not have a source of energy. For example, a resistor lead is a passive pin.

After you select a shape, the **PIN** menu returns. You can now choose another **PIN** command or press **<Esc>** to return to the main command level.

The other items in the **PIN** menu (**Jump** and **Zoom**) appear in many LIBEDIT menus. They operate the same as their corresponding members in the main-level menu. **escape** returns you to the main command level.

7.3.18 QUIT

Use the **QUIT** command to exit LIBEDIT and return to DOS. With the **QUIT** command, you can also write to a file, update the current library you are editing, initialize the current LIBEDIT session, and suspend to DOS.

Initializing the session means that the current part you are editing is removed from memory, and LIBEDIT requests the name of a another library to edit. The old library file is not updated. Any editing since the last library update is lost.

When you select **QUIT**, the following menu appears:

Update File
Write to File
Initialize
Suspend to DOS
Abandon Edits

7.3.18.1 QUIT Update File

The **Update File** command writes the latest copy of the library being edited to disk. This command is *not* the same as the **LIBRARY Update Current** command.

Type in the name of the library file you want to edit and press **<Enter>**.

7.3.18.4 QUIT Suspend to DOS

With the **Suspend to DOS** command, you can temporarily leave LIBEDIT and return to DOS. You can then enter DOS commands. LIBEDIT remains suspended in memory. To return to LIBEDIT, type

exit

7.3.18.5 QUIT Abandon Edits

Use the **Abandon Edits** command to leave LIBEDIT and return to DOS.

If you have a part on the screen that contains specifications not currently in the library file you're editing or if you have made changes to the current library, the **Abandon Edits** command asks you to confirm your decision to leave.

Choosing **No** returns you to the main command level. Choosing **Yes** exits LIBEDIT.

7.3.19 REFERENCE

With the **REFERENCE** command, you can specify or edit the part's reference designator.

The default reference designator is U?A. The ? is a placeholder for the occurrence of the device, and A is placeholder for the part in a multiple-part-per-package device. The OrCAD utility ANNOTATE replaces ? with a number indicating the instance of the part and increments A through A, B, C, etc., once for each part of the package used.

7.3.20.2 SET Backup Files

When you enable **Backup File**, a backup file is created when you write or update files using the **QUIT** command. The backup file contains the previous version of the library being edited.

7.3.20.3 SET Confirm Deletion

When you select confirm deletion, LIBEDIT asks you to confirm a selected deletion. This gives you a chance to change your mind after you have selected a delete command. **SET Confirm Deletion** affects all deletion commands except the **Body Delete** command.

7.3.20.4 SET Error Bell

When you select **Error Bell**, error messages sound your computer's speaker.

7.3.20.5 SET Macro Prompts

When you select **Macro Prompts**, the commands making up your macros are displayed on the screen when the macro is invoked.

7.3.20.6 SET Power Pins Visible

When you select **Power Pins Visible**, the part's power pins appear on the screen. For example, if you set **Power Pins Visible** and display the part 74LS00 in the TTL library, the pins 14 and 7 appear. For the 74LS00, 14 is Vcc and 7 is Gnd.

Note that power pins may overlap existing pin names or the part name. Typically, power pins are not displayed.

7.3.22 ZOOM

With the **ZOOM** command, you can change the size of the part on the screen. The three possible scales are **1** (the default), **Half**, and **Quarter**. **Half** scale shows more detail than scale **1**, and **Quarter** scale shows more detail still.

When you select **ZOOM**, the following menu appears on the screen:

Center (1)
In (1)
Out (Half)
Select

The number or word in parentheses after **Center** indicates the current scale.

7.3.22.1 ZOOM Center

Center re-centers the displayed portion of the sheet around the cursor. This command is useful for centering an object on the screen so that you can easily edit it.

For example, if an object is displayed partially off the screen, you may center it by placing the cursor near the object and selecting **ZOOM Center**.

7.3.22.2 ZOOM In

ZOOM In Selects the next more detailed scale (a larger view).

7.3.22.3 ZOOM Out

ZOOM Out selects the next less detailed scale (a smaller view)

You will see the message :::type any key to continue::: twice: once at the OrCAD startup screen, and again at the copyright screen. Type any key to continue.

LIBEDIT briefly displays the message "working", then displays the main screen: a vertical line on the left side, a horizontal line across the top, the cursor arrow, and the screen coordinates in the upper right corner (see Figure 7-2)

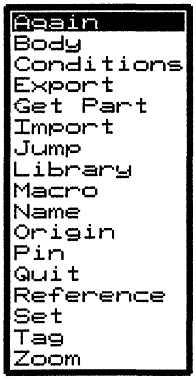


Figure 7-2. The Main LIBEDIT Menu

As you move the cursor with the direction keys or mouse, the screen coordinates change.

Because you made the TTL library the current library when you invoked LIBEDIT, bring in the 373 with the **Get Part** command. Press **G**, or press <ENTER>, move the highlight to the **Get Part** command and again press <ENTER>.

NOTE

> The following examples assume that your display is set to a zoom scale of 1.

Select **Dot** by typing **D**. LIBEDIT places the pin you have specified at the cursor location (you will notice that the pin was placed over other pins).

Now move the cursor one step to the right, to cursor coordinates (+3.0, +12.0). We'll add a G pin here using the same steps we used to add the OC pin.

Invoke the **Add** command. At the "Pin Name?" prompt, type "G" (capital letter, no quotes) and press <ENTER>.

At the "Pin Number?" prompt, type "11" (no quotes) and again press <ENTER>.

At the "Pin Type?" prompt, select **Input**.

At the "Pin Shape?" prompt, select **Line** (type **L**).

LIBEDIT places the pin at the cursor location.

Now we'll delete the "old" OC and G pins. Move the cursor to the original pin 11 (cursor coordinates + .0, +11.0) and delete it by typing **D**. Repeat this action for the original pin 1 (cursor coordinates + .0, +10.0).

Press <ESC> to return to the LIBEDIT main menu.

Press **L** to invoke the **Library** command. LIBEDIT displays these **Library** command options: **Update Current**, **List Directory**, **Browse**, **Delete Part**, and **Prefix**.

Press **U** to update the current part.

If you want to save your changes to the TTL.LIB file, press <ESC>, then **Quit**. On the prompt line, LIBEDIT displays "Quit ttl.lib?" and a menu of options: **Update File**, **Write to File**, **Initialize**, **Suspend to DOS**, and **Abandon Edits**.

LIBEDIT performs the action and places you at the top of the **Body** command menu.

Select **L** for **Line**. LIBEDIT displays the **Line** command options: **Begin**, **Jump**, **Origin**, **Tag**, **Zoom**, and **escape**.

With the mouse or arrow keys, position the cursor just below the U?A reference designator, and to the right above the bubble (coordinates +6.0, + .0). Maintaining the same X-coordinate, move the cursor down the Y-axis to just above the library part number (screen coordinates +6.0, +4.0). We can see that the arcs we must draw to represent the nose of the NAND gate must be 4 units in diameter (it must have a radius of 2 units). LIBEDIT draws arcs in 90 degree sectors, so to draw the nose of the NAND gate, we must draw two arcs.

Position the cursor on the left edge of the bubble (at coordinates +6.0, +2.0) and type **O** to select **Origin**. Notice that this resets the cursor coordinates to zero.

With the arrow keys or mouse, move the cursor 2 units to the left (to coordinates -2.0, + .0), and up 2 units (to coordinates -2.0, -2.0). Select **Begin**. The prompt line displays: **Begin**, **End**, **New**, **Jump**, **Origin**, **Tag**, **Zoom**, **escape**.

Now move the cursor four units to the left (to coordinates -6.0, -2.0) and again type **B**. Move the cursor down 4 units (to coordinates -6.0, +2.0). Type **B** again. Finally, move the cursor 4 units to the right (to coordinates -2.0, +2.0). Your display should look like Figure 7-3 below.

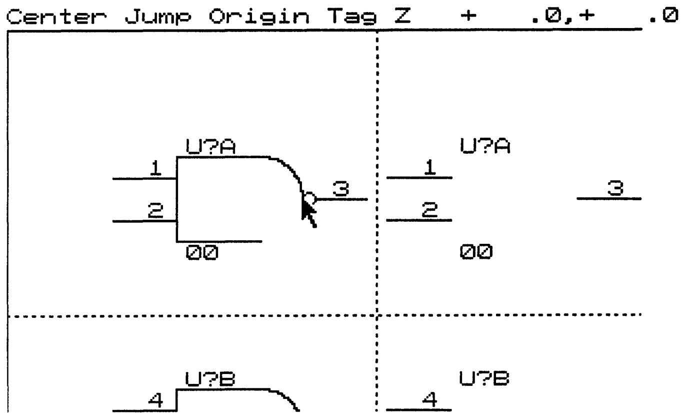


Figure 7-4. Drawing the First Arc

Now, we can draw the lower arc in much the same way. Move the cursor to the left 2 units (to coordinates -2.0, +0.0). Type **C** for center. Move the cursor down 2 units (to coordinates -2.0, +2.0) and type **E** for edge. Move the cursor 2 units to the right and up 2 units (to coordinates +0.0, +0.0). Again, type **E**. Your display should now look like Figure 7-5.

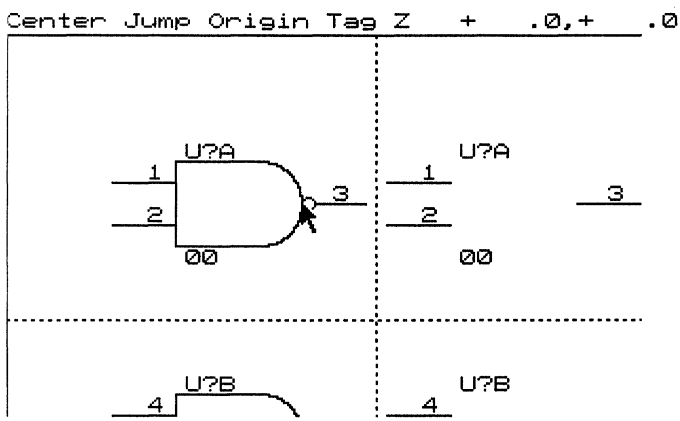


Figure 7-5. The Completed NAND Gate Body

2. Compose the source file using the COMPOSER utility. This is similar to a compilation; it produces another file, a data file readable by DRAFT. The convention is to give this data file a .LIB extension.

7.4.1 Invoking COMPOSER

COMPOSER *source* *library* **<Enter>**

where:

source is the name of the ASCII text file that describes your custom parts using OrCAD's Symbol Description Language. The .SRC extension is a convention, not a requirement.

library is the name of the resulting library file. If you give the name of an existing file, COMPOSER asks if you want to overwrite the existing file. You cannot append to an existing file.

<Enter> represents the ENTER key on your computer. DOS commands are executed when you press **<Enter>**.

Both *source* and *library* may be complete pathnames -- that is, if either is in a directory other than your current directory, you must specify the complete path. Consider the following example:

COMPOSER custom.src custom.lib **<Enter>**

The files COMPOSER.EXE and custom.src are in the same directory, and this directory is your working other than your current directory, you must specify the complete path. Consider the following example:

DECOMP custom.lib custom.src **<Enter>**

The files DECOMP.EXE and custom.src are in the same directory, and this directory is your working directory. The example just shown creates the file custom.src in your working directory. Here is another example that uses complete pathnames.

DECOMP \orcad\library\custom.lib
\orcad\library\custom.src **<Enter>**

The file DECOMP.EXE is in your working directory, which is not necessarily \orcad\library. The example just shown creates the file custom.src in the directory \orcad\library.

Note that DECOMP does not return your original source; it makes its own. For example, the comments in your original source are not reproduced. DECOMP adds its own comments. Also, DECOMP may rearrange the order of the part definitions. DECOMP lists parts in numeric order followed by parts in alphabetical order. For example, assume that you define two parts, one called resistor and one called 7400. You place resistor in your source file before 7400, run COMPOSER, then DECOMP to produce a new source file. Unlike your original source file, the new source file has 7400 listed before resistor.

7.4.4.1 Prefix Delimiters

The prefix definition is delimited by the keywords, PREFIX and END. The initial delimiter is the keyword PREFIX all alone on a line. Subsequent lines contain the prefix definition itself. The terminating delimiter is the keyword END all alone on a line.

7.4.4.2 A Prefix Definition is Required

All source files must begin with a prefix definition. If you decide your custom library doesn't need a prefix definition, you must still supply a null prefix. A null prefix consists only of the delimiting keywords.

7.4.4.3 Examples of Prefix Definitions

Here is how a null prefix definition looks.

```
PREFIX
END
```

Here is an example of a non-null prefix definition. The example comes from OrCAD System's TTL source library, TTL.LIB.

```
PREFIX
'74LS'    =    'LS'
'74S'     =    'S'
'74ALS'   =    'ALS'
'74AS'    =    'AS'
'74HCT'   =    'HCT'
'74HC'    =    'HC'
'74ACT'   =    'ACT'
'74AC'    =    'AC'
'74F'     =    'F'
'74'
END
```

7.4.4.4 Use of the Prefix Definition

DRAFT uses the prefix definition when you obtain a part with the **GET** command. Instead of entering the

The shorthand string enables you to bypass the pop-up prefix menu and still enter an abbreviated part name. For example, you can obtain the part 74HC04, by supplying the **GET** command with the abbreviated name HC04. This is possible because HC is a shorthand string for 74HC.

- Close the prefix definition with the keyword END followed by **<Enter>**.

7.4.5 The Part Definition

The part definition defines the part's name, its size (in unit lengths on the screen and in tenths of an inch on the printed worksheet), the number of parts per package and the pin functions (input, output, open collector, etc.).

7.4.5.1 Types of Part Definitions

There are two types of part definitions: block symbol definitions and bitmap definitions. You don't have to group your block definitions and bitmap definitions together. For example, your source file may contain a block definition, followed by a bitmap definition, followed by another block definition.

Block and bitmap definitions follow much the same syntax. A bitmap definition looks like a block definition followed by a bitmap. When COMPOSER sees a bitmap, it uses that bitmap to represent the part, rather than defaulting to a square or rectangle.

7.4.5.2 Components of a Part Definition

A part definition has the following fields:

- An optional bitmap. Use this if the symbol you want is not a square or rectangle.

- An optional conversion. This only has meaning if you've defined a bitmap. The most common use for converted bitmaps is to specify the DeMorgan equivalent of the defined part.

7.4.6 Block Symbol Definition

Illustrated below is an example of a block symbol definition. The example does not represent a real part, although it is similar to a JK flipflop. Figure 7-6 shows the symbol produced by this block definition.

```
'74EXAMP'
REFERENCE         'LATCH'
6    10   2
L1   3    11   SHORT IN        'J'
L5   1    13   DOT CLK IN      'CLK'
L9   2    12   SHORT IN        'K'
B3   15   14   DOT IN          'CL'
T3   4    10   DOT IN          'P'
R1   6    7    OUT             'Q'
R9   5    9    OUT             'Q\'
T0   16   16   PWR             'VCC'
B0   8    8    PWR             'GND'
```

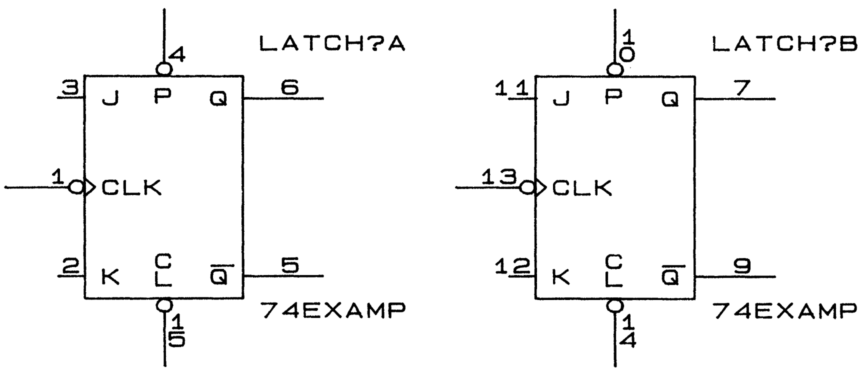


Figure 7-6. The Block Symbol for 74EXAMP

question mark will be replaced with a number. For example, if the reference designator for a resistor were R? and there are 16 resistors in your design, ANNOTATE would change the designators to R1, R2, ... R16.

This example has more than one part per package, so the reference designator appears with an A after the question mark. ANNOTATE then sequences the letters. ANNOTATE would convert the A of the second part into a B. For example, after running ANNOTATE, the first two occurrences of the 74EXAMP would appear as LATCH1A and LATCH1B, the next two as LATCH2A and LATCH2B, etc. If you omit the REFERENCE line, the default designator U?A appears.

What appears when you call up a part is determined as follows:

1. If the device has 0 parts per package and you do not specify a REFERENCE key word, none appears. Nor does the part name appear.

2. If the device has 0 parts per package and you specify a REFERENCE key word, it appears. It consists of the string you specified followed by a question mark. ANNOTATE replaces the ? with a sequential number. The part name also appears.

3. If the device has one or more parts per package, and you do not specify a REFERENCE key word, a default reference designator (U?A) appears. ANNOTATE replaces the ? with a sequential number that identifies the occurrence of the device and replaces the A with a letter that cycles through the parts of a device. The part name also appears.

7.4.6.5 Pin Definitions

The rest of the example consists of the pin definitions. Consider the second pin position as an example.

The first field L5 locates the pin on the left side of the part in the fifth position counting from the top down. The Y dimension was given as 10 and hence specified 11 possible positions, 0 through 10. The first possible position is L0 and the last is L10. The specified pin position (L5) is 0.5 inches from the top of the part, when seen on the printed worksheet.

The next two fields for pin position L5 identify the pin numbers, 1 for the first part of the package and 13 for the second part of the package. The pin at L5 specifies DOT to obtain the inversion bubble and CLK to get the clock symbol. In this case, DOT and CLK are modifiers of the pin function, IN. Finally, the 'CLK' gives the pin a name.

As further examples, consider R9 and B3. R9 puts a pin on the right side in the ninth position, and B3 puts a pin on the bottom in the third position counting right. The two power supply connections are at the top and bottom in the zero position.

Note that the vertices of a part have two possible representations, For example, L0 and T0 specify the same pin location (the upper lefthand corner). Consequently, if you place a pin at T0 and another at L0, those pins will be shorted.

Figure 7-8 shows a grid that represents the possible pin positions for the 74EXAMP library part.

7.4.6.7 Selectively Displaying Pins

If the device has more than one part per package, you can selectively display the pins. For example, assume you wanted to display the power pins VCC and GND, but only on the second part of the device, not on the first. You could do that by coding the last two lines of the block symbol as follows.

```
TO    0    16    PAS'VCC'
BO    0    8     PAS'GND'
```

When you place this symbol on the screen, the power pins do not display because the first column of pin locations contains a 0. When you place another symbol on the screen, it looks identical to the first. Both are called LATCH?A, and neither shows the power pins.

However, if you exit DRAFT and run the ANNOTATE utility with the /M option (which causes the annotation information to be merged into the sheet directly) and then look at the sheet again with DRAFT, you'll see the two parts labeled LATCH1A and LATCH1B. The power pins appear only on the second part of the device, LATCH1B.

If a device has more than one part per package, you may want power pins on some parts present and the power pins on other parts not to be present.

This technique also works for non-power pins. By specifying a pin number of 0, you can cause a pin not to appear for the part of a package.

But if your device has one part per package, specifying a pin number of 0 does not prevent the pin from appearing. The pin appears with a pin number of 0.

If your device has 0 parts per package, you cannot specify pin numbers, and consequently, none appear.

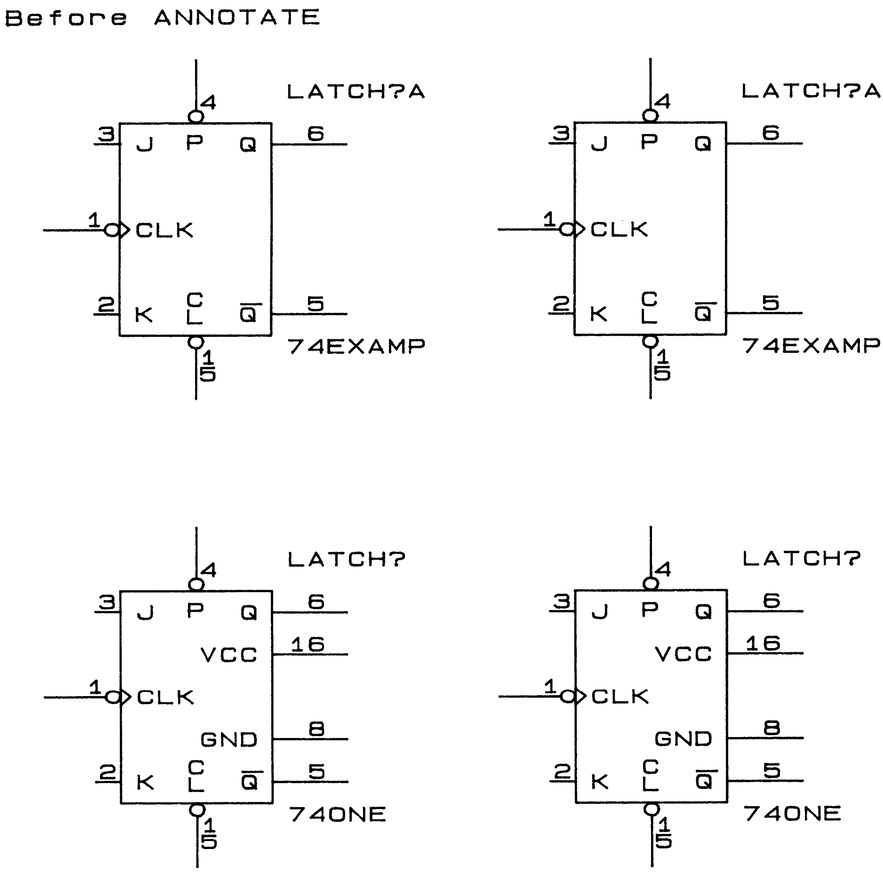


Figure 7-9A. Before Annotation

MOTO.LIB. The definition is quite long so only the first few lines are shown. Figure 7-10 shows the resulting screen figure.

```
'68020'
15    66    GRIDARRAY
L1    C2    CLK IN      'CLK IN'
L3    J12   IN          'I\P\L\0\' {the \ bars the pin name}
L4    J13   IN          'I\P\L\1\'
L5    H12   IN          'I\P\L\2\'
                .
                .
                .
```

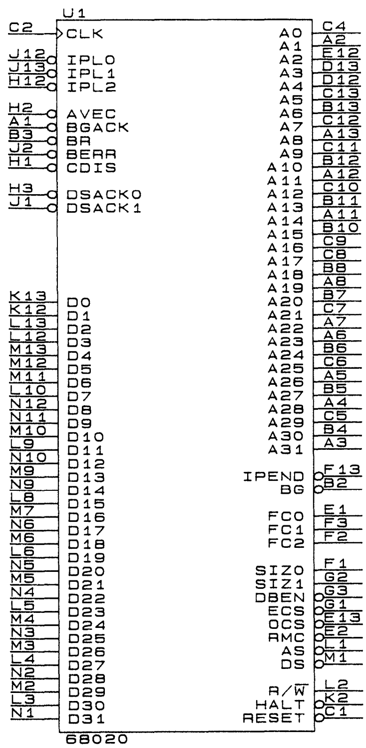


Figure 7-10. The Block Symbol for 68020

7.4.6.9 Pin String

The pin string is delimited by single quotes. If you want a single quote as part of the pin string, you must use two

There are four points you should keep in mind when creating bitmaps, as opposed to block symbols.

1. You have to pay more attention to pin placement. The pin definition line and the bitmap have different scales, and you need to take the conversion into account when you draw the symbol.

2. Although you can put a pin name in the pin definition, the pin name will not appear on the screen. The pin name will, however, be recognized by the NETLIST utility.

3. A bitmap symbol gives you the opportunity to define a converted symbol. Bitmap devices always have a normal form. You have the option of also defining a converted form. Block symbols cannot have a converted form. When you use the GET command and extract a part from a library, it appears in normal form. The resulting menu enables you to choose its converted form instead. You define what the converted form is when you create the library source file. Typically, users define the converted form as the DeMorgan equivalent of the normal form.

4. The maximum number of bits allowed in a bitmap is 16,384. However, bitmaps are allocated in blocks of eight. So consider a bitmap that consists of 5 rows of 33 bits. Rather than 5 * 33 = 165 bits, the space actually taken up by such a bitmap is 8 * 40 = 320 bits.

The bitmap begins after the last pin definition. A pound sign (#) indicates that the pixel bit is turned on, and a period (.) indicates that the pixel bit is turned off.

Each . or # in the bitmap represents a screen pixel spacing of 0.01 inch in the X direction. Each line of the bitmap represents 0.01 inch in the Y direction.

types are PAS for passive, and there are no pin names (however, they may be specified).

The X size is specified as 3; so the bitmap has lines that are 31 characters long. The Y size is specified as 2, so the bitmap has 21 lines. The part definition specifies two pins, one on the left in the first position (L1) and another on the right in the first position (R1). The pin positions are always spaced 0.1 inch apart. As far as the bitmap is concerned, pin positions are at lines 0, 10, 20, 30, etc.

The line numbers are enclosed in comment delimiters. It isn't necessary to number the bitmap lines this way, but doing so makes the bitmap more readable.

You can reduce the size of bitmaps used in your library by observing the following rules.

1. An empty row (one that has only dots) can be represented by a dot in the zeroth column. If that is the only character on the row, then the row is held as cleared.

2. Empty rows below the actual symbol need not appear in the bitmap.

3. Periods are not required after the last # in a row.

Here is an example of the same resistor definition that follows the reduction rules just described. Figure 7-11 shows the symbol that results from this part definition.

```
{Part Definition for a resistor}
'RESISTOR'
REFERENCE   'R'
3 2   0
L1    PAS   ''
R1    PAS   ''

                    {Top side}
```

The next example should clarify the use of conversion bitmaps. First, is the definition of the 7400. Then, comes the conversion bitmap -- it's the DeMorgan equivalent of the 7400. Figure 7-12 shows the normal and converted symbols that result from this part definition.

The 7400 has five pins, two of which are power pins that do not appear in the symbol. The screen size is 6 X-units and 4 Y-units. It has four parts per package.

The conversion bitmap uses the same XY size and parts per package as the normal bitmap. You must, however, redefine the pin types. Note the DOT keyword missing from the redefinition of the pin at R2.

Also note that the conversion bitmap has the same number of parts per package as the normal bitmap. The number of parts per package determines how many columns of pin numbers appear in the definition. The converted definition must have the same number of columns as the normal definition.

```
CONVERT
L1     1      4      9      12     IN     'I0'
L3     2      5      10     13     IN     'I1'
R2     3      6      8      11     OUT    'O'
T0     14     14     14     14     PWR    'VCC'
B0     7      7      7      7      PWR    'GND'

{00}#################################
{01}.#...............................####
{02}..#..................................###
{03}...#....................................##
{04}....#.....................................##
{05}.....#......................................##
{06}......#.......................................#
{07}..###.#........................................#
{08}.#...##.........................................#
{09}#.....##.........................................#
{10}#.....##..........................................#
{11}#.....#.#..........................................#
{12}.#...#...#..........................................#
{13}..###....#...........................................#
{14}.........#............................................#
{15}.........#.............................................#
{16}..........#.............................................#
{17}..........#..............................................#
{18}..........#...............................................#
{19}..........#................................................#
{20}..........#.................................................#
{21}..........#................................................#
{22}..........#...............................................#
{23}..........#..............................................#
{24}..........#.............................................#
{25}.........#.............................................#
{26}.........#............................................#
{27}..###....#...........................................#
{28}.#...#...#..........................................#
{29}#.....#.#..........................................#
{30}#.....##..........................................#
{31}#.....##.........................................#
{32}.#...##.........................................#
{33}..###.#........................................#
{34}......#.......................................#
{35}.....#......................................##
{36}....#.....................................##
{37}...#....................................##
{38}..#..................................###
{39}.#...............................####
{40}#################################
```

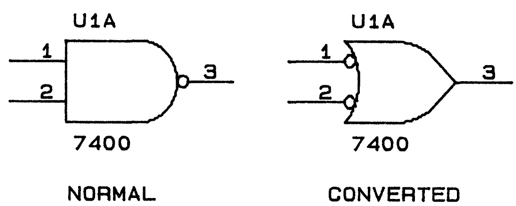


Figure 7-12. The 7400 Symbol and its Conversion

4. You cannot continue along a path that goes against the arrows. For example, in Figure 7-13 after making a part definition you cannot choose to return to the prefix definition.

5. Text enclosed in ovals represents an identifier. Text enclosed in squares represents a token.

Identifiers

Identifiers serve as placeholders for a more detailed level of syntax structure. They do not represent command syntax or tokens. Rather, they provide the ability to give a overview of the syntax. When you create the part, you must work down through all the nested identifiers. For example the syntax diagram for a library source file shown in Figure 7-13 has two identifiers (prefix definition and part definition) and no tokens.

Tokens

Tokens are the building blocks of a library source file. Just as a sentence is made up of words, a library source file is made up of tokens. A token belongs to one of the following categories:

GRIDARRAY	Specifies that the device is a pin-grid array. Used in place of the number of parts per package.
HIZ	Identifies the pin as a high impedance (3-state) output.
IN	Identifies the pin as an input.
I/O	Identifies the pin as input/output.
OC	Identifies the pin as open collector.
OE	Identifies the pin as open emitter.
OUT	Identifies the pin as an output.
PAS	Identifies the pin as passive.
PREFIX	Delimits the beginning of a prefix definition.
PWR	Identifies the pin as a power pin. The PWR keyword prevents a pin from being displayed.
REFERENCE	Takes an argument (an ASCII string representing a reference value). Overrides the default reference value.
SHORT	Specifies that the pin lead lengths be 0.1 inch instead of the normal 0.3 inch.

you must choose between *pin#* or *grid*; and you can choose a repeated number of each.

7.4.8.3 Prefix Definition

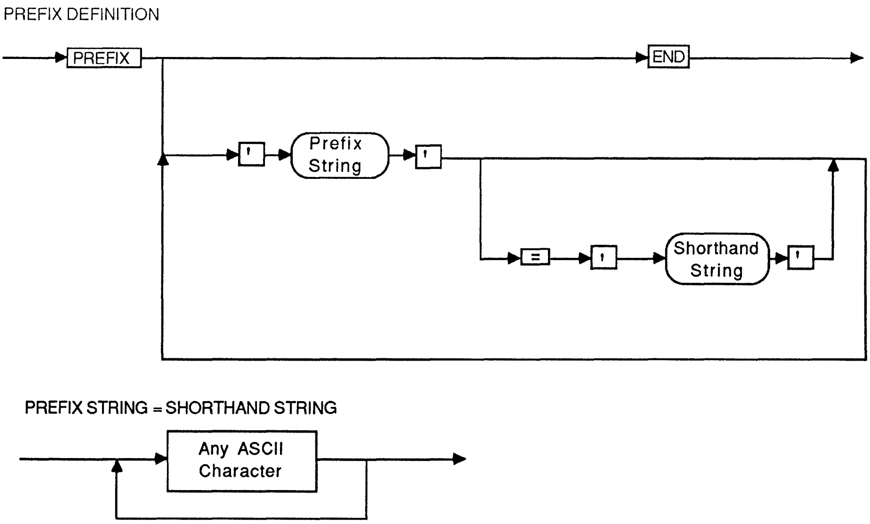


Figure 7-14. Syntax Diagram For a Prefix Definition

```
PREFIX

[ 'prefix string'  [= 'shorthand string' ]]
                    .
                    .
                    .
END
```

where:

prefix string	A character string of up to seven printable ASCII characters. You can have a maximum of 16 prefix strings.

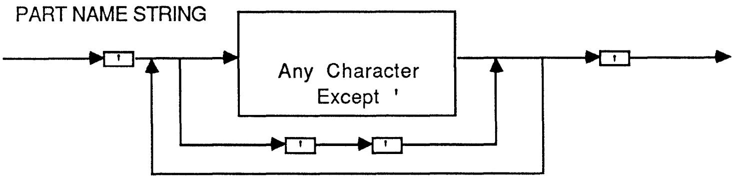


Figure 7-16. The Part Name String

'part name string' [REFERENCE *'ref string'*] *X size Y size* {*parts* / *pckg*, GRIDARRAY] *pin definition* • • • [*bitmap definition*] [*conversion bitmap*]

where:

part name string — A character string of up to seven printable ASCII characters that identifies the part. This is the string that can be used as an argument for the **GET** command.

ref string — A character string of printable ASCII characters. If present, the reference designator replaces the default reference designator.

EXAMPLES

Example 1:

```
'2114'   '2148'
6       14   1

pin definition
.
.
.
```

(Two part name strings. These may be on the same or separate lines).

Example 2:

```
'7474' '74ALS74' '74LS74' '74S74'
'74HC74' '74AC74'
6    6    2
pin definition
     .
     .
     .
```

pos

A letter followed by a number. The letter is one of the following: T, L, R, B where:

T indicates the top of the symbol.
L indicates the left side of the symbol.
R indicates the right side of the symbol.
B indicates the bottom of the symbol.

The number represents the distance along the indicated side. The distance is measured in unit lengths on the screen and in 0.1 inches on the printed worksheet. For example, if the block symbol were 6X by 10Y the grid used for placing pins is as follows. The figure below shows the location of L3.

grid

A letter followed by a number. *grid* represents the pin-grid array pin number. You can only choose a pin-grid array pin number if, in place of parts/pckg, you chose the keyword GRIDARRAY. The letter must be in the range A through S; the number must be in the range 1 through 15.

GRID ARRAY PIN NAME

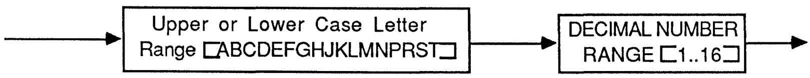


Figure 7-20. Gridarray Pin Name Syntax Diagram

SHORT

A keyword that places a short lead length at the specified pin. The normal lead length is 3 screen units or 0.3 inches on the printed worksheet. When the SHORT keyword is present, the lead length is 1 screen unit or 0.1 inch on the printed worksheet. The SHORT keyword cannot describe a pin that also has either the CLK or DOT keywords.

You can use the CLK keyword in conjunction with the DOT keyword to produce a DOT CLK symbol. The figure below shows the DOT CLK symbol.

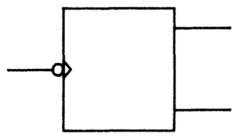


Figure 7-22B. The DOT CLK Symbol

IN

A keyword that identifies the pin as an input.

OUT

A keyword that identifies the pin as a standard totem-pole output.

I/O

A keyword that identifies the pin as a dual function input/output pin.

OC

A keyword that identifies the pin as an open collector or open drain.

PWR

A keyword that identifies a power pin, such as Vcc, Gnd, Vss, Vdd, and others. Power pins are not displayed on library parts when they appear on the screen or printed worksheet. However, the NETLIST utility connects all power supply pins that are defined in library source files.

pin name

A character string that represents a name for the specified pin. For block symbols, this name appears on the screen or the printed worksheet. Pin names do not appear on the screen or the printed worksheet when they are part of pin definitions for a a bitmap symbol. However, you may still choose to use pin names in bitmap symbols. The NETLIST utility still recognizes them, and you may find them useful as personal references.

You can enter pin names either in upper- or in lower-case, but they always appear in upper-case.

The backslash (\) and single quote (') are special characters. A backslash (\) after a character indicates that the character has a bar over it. If you want to bar multiple characters, you must place a backslash after each character. The single quote delimits the part name string. If you want a single quote as part of the pin string, you must escape it with another single quote.

7.4.8.6 Bitmap Definition

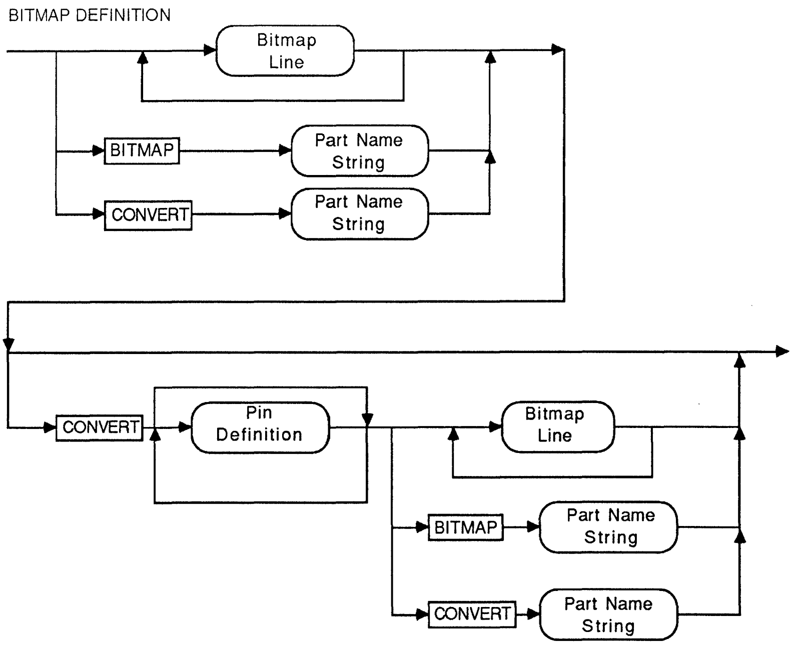


Figure 7-24. The Bitmap Syntax Diagram

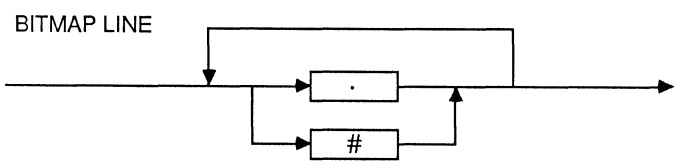


Figure 7-25. The Bitmap Line

[{.,#} ...,BITMAP *'part name'* ,CONVERT *'part name'*]
•
•
•

- Periods are not required after the last # in a line.

- If you use the BITMAP option, be sure that the part name you refer to is previously defined. Be aware that DECOMP may reorder part definitions. DECOMP order parts numerically and then alphabetically.

EXAMPLES

```
Example 1:
      'CAPACITOR'
      2     3     0
      T1    PAS   ''
      B1    PAS   ''
{0123456789012345567890123456789 0} {   0     1     2      3}

{00}..........#
{01}..........#
{02}..........#
{03}..........#
{04}..........#
{05}..........#
{06}..........#
{07}..........#
{08}..........#
{09}..........#
{10}..........#
{11}..........#
{12}..........#
{13}#####################
{14}.
{15}.
{16}.
{17}.
{18}.....###########
{19}...###....#....###
{20}.##.......#......##
{21}#.........#........#
{22}..........#
{23}..........#
{24}..........#
{25}..........#
{26}..........#
{27}..........#
{28}..........#
{29}..........#
{30}..........#
```

Example 2: BITMAP '7400'

DECOMP orders parts numerically and then alphabetically.

EXAMPLES

```
Example 1:      CONVERT
       L1       1       4       9       12      IN      'I0'
       L3       2       5       10      13      IN      'I1'
       R2       3       6       8       11      OUT     'O'
       T0       14      14      14      14      PWR     'VCC'
       B0       7       7       7       7       PWR     'GND'

{00}#################################
{01}.#...............................####
{02}..#..................................###
{03}...#....................................##
{04}....#.....................................##
{05}.....#......................................##
{06}......#.......................................#
{07}..###.#........................................#
{08}.#...##.........................................#
{09}#.....##.........................................#
{10}#.....##..........................................#
{11}#.....#.#..........................................#
{12}.#...#...#..........................................#
{13}..###....#...........................................#
{14}.........#............................................#
{15}.........#.............................................#
{16}..........#.............................................#
{17}..........#..............................................#
{18}..........#...............................................#
{19}..........#................................................#
{20}..........#.................................................#
{21}..........#................................................#
{22}..........#...............................................#
{23}..........#..............................................#
{24}..........#.............................................#
{25}.........#.............................................#
{26}.........#............................................#
{27}..###....#...........................................#
{28}.#...#...#..........................................#
{29}#.....#.#..........................................#
{30}#.....##..........................................#
{31}#.....##.........................................#
{32}.#...##.........................................#
{33}..###.#........................................#
{34}......#.......................................#
{35}.....#......................................##
{36}....#.....................................##
{37}...#....................................##
{38}..#..................................###
{39}.#...............................####
{40}#################################
```

Example 2: CONVERT '7400'

7.4.9 Examples of Source Libraries

Using the techniques just described, we can now examine some examples of library source files. Remember, comments are enclosed within braces {...}. We present three sample source libraries.

SAMPLE_1.SRC	This file contains some typical block symbol definitions. It has some memory devices and two microprocessors, the 8086MAX (the 8086 in maximum mode) and the 68020 (a pin-grid array).
SAMPLE_2.SRC	This file contains some standard TTL devices. Note the use of a prefix definition.
SAMPLE_3.SRC	This file contains some typical bitmap definitions. It has an AND gate with its DeMorgan equivalent, an antenna, a capacitor, a triode, and other non-square or non-rectangular symbols.

Although these examples separate the kinds of definitions (block from bitmap) into different libraries, that is not a requirement. You can intersperse block and bitmap definitions in the same file.

To use these sample files, construct a pure ASCII file (no formatting characters) and execute the COMPOSER utility. Then reconfigure DRAFT to recognize the new library. For example, to compose the sample files, enter the following commands on the DOS command line. The file COMPOSER.EXE and the sample files must be in the same directory.

{All of the above memory devices have the same block symbol. The part's size is 6X by 14Y, and it has one part per package.}

```
{256K x 1 dynamic RAM}
'21256'  '51C256'  '50256'  '50257'  '81256'  '81257'
'6256'  '6257'  '4256'  '4257'  '41256'  '37256'
6     14    1
L1    5     IN    'A0'
L2    7     IN    'A1'
L3    6     IN    'A2'
L4    12    IN    'A3'
L5    11    IN    'A4'
L6    10    IN    'A5'
L7    13    IN    'A6'
L8    9     IN    'A7'
L9    1     IN    'A8'
L11   4     IN    'R\A\S'
L12   15    IN    'C\A\S\'
L13   3     IN    'W\E\'
R1    14    HIZ   'D0'
R3    2     IN    'D1'
T0    8     PWR   'VCC'
B0    16    PWR   'VSS'
```

{The following are microprocessor devices}

```
'8086MAX'
13    32    1
R1    16          I/O   'AD0'
R2    15          I/O   'AD1'
R3    14          I/O   'AD2'
R4    13          I/O   'AD3'
R5    12          I/O   'AD4'
R6    11          I/O   'AD5'
R7    10          I/O   'AD6'
R8    9           I/O   'AD7'
R9    8           I/O   'AD8'
R10   7           I/O   'AD9'
R11   6           I/O   'AD10'
R12   5           I/O   'AD11'
R13   4           I/O   'AD12'
R14   3           I/O   'AD13'
R15   2           I/O   'AD14'
R16   39          I/O   'AD15'
R17   38          OUT   'A16/S3'
R18   37          OUT   'A17/S4'
R19   36          OUT   'A18/S5'
R20   35          OUT   'A19/S6'
R22   34          OUT   'B\H\E\/S7'
R24   26    DOT   OUT   'S0'
R25   27    DOT   OUT   'S1'
R26   28    DOT   OUT   'S2'
R28   32          OUT   'R\D\'
R29   29    DOT   OUT   'LOCK'
```

```
R3     11     DOT    OC     'C'
R2     12     DOT    OC     'B'
R1     13     DOT    OC     'A'
T0     16            PWR    'VCC'
B0     8             PWR    'GND'

'7474'        '74ALS74'     '74AS74'      '74LS74'      '74S74'
'74HC74'      '74AC74'
6      6      2
L2     2      12            IN     'D'
L4     3      11     CLK    IN     'CK'
B3     1      13     DOT    IN     'CL'
T3     4      10     DOT    IN     'P'
R4     6      8             OUT    'Q\'
R2     5      9             OUT    'Q'
T0     14     14            PWR    'VCC'
B0     7      7             PWR    'GND'
```

{This part has two parts per package. The far left column represents the pin position. The second column contains the pin numbers of the first D flip-flop in the package. The third column contains the pin number of the second D flip-flop in the package. Note the power pins on 14 and 7. They are required for both devices.}

{End of the source file, SAMPLE_2.SRC}

7.4.9.3 SAMPLE_3.SRC: A Bitmap Symbol Library Source File

{Beginning of the source file, SAMPLE_3.SRC}

```
PREFIX
END
```

```
CONVERT
L1     1      4      9      12     IN     'I0'
L3     2      5      10     13     IN     'I1'
R2     3      6      8      11     OUT    'O'
T0     14     14     14     14     PWR    'VCC'
B0     7      7      7      7      PWR    'GND'
{00}#################################
{01}.#...............................####
{02}..#..................................###
{03}...#....................................##
{04}....#.....................................##
{05}.....#......................................##
{06}......#.......................................#
{07}..###.#........................................#
{08}.#...##.........................................#
{09}#.....##.........................................#
{10}#.....##..........................................#
{11}#.....#.#..........................................#
{12}.#...#...#..........................................#
{13}..###....#...........................................#
{14}.........#............................................#
{15}.........#.............................................#
{16}..........#.............................................#
{17}..........#..............................................#
{18}..........#...............................................#
{19}..........#................................................#
{20}..........#.................................................#
{21}..........#................................................#
{22}..........#...............................................#
{23}..........#..............................................#
{24}..........#.............................................#
{25}.........#.............................................#
{26}.........#............................................#
{27}..###....#...........................................#
{28}.#...#...#..........................................#
{29}#.....#.#..........................................#
{30}#.....##..........................................#
{31}#.....##.........................................#
{32}.#...##.........................................#
{33}..###.#........................................#
{34}......#.......................................#
{35}.....#......................................##
{36}....#.....................................##
{37}...#....................................##
{38}..#..................................###
{39}.#...............................####
{40}#################################
```

8.1 Assumptions

Hard Disk. When describing the directory structure recommended for hard disk users, this section assumes that you have configured DOS to show the directory structure as part of your prompt. For example, the default prompt for drive c is C>. If the directory C:\ORCAD is your working directory, this section assumes that your prompt is C:\ORCAD>. To configure DOS this way, issue the DOS command,

```
C>PROMPT$P$G
```

If you prefer, include this line in the file autoexec.bat and place it in the root directory of your boot device so that the configuration takes place automatically every time you boot up.

Mouse. Although not required, a mouse makes life much easier. This example assumes that you have a mouse with its driver installed.

OrCAD/SDT III makes use of the left and right mouse buttons. If your mouse has a middle button, you may define it as a macro key. The left mouse button executes <ENTER> and the right mouse button executes <ESCAPE>.

If you are not using a mouse, you can execute <ENTER> by pressing the <RETURN> key and <ESCAPE> by pressing the <ESC> key. Typing <RETURN> after invocation brings up the main command menu. You can then execute a command by typing the first letter of the command name. You can also highlight the command to be executed by moving the highlight with the directional arrow keys and then choosing the highlighted command by pressing <Return>.

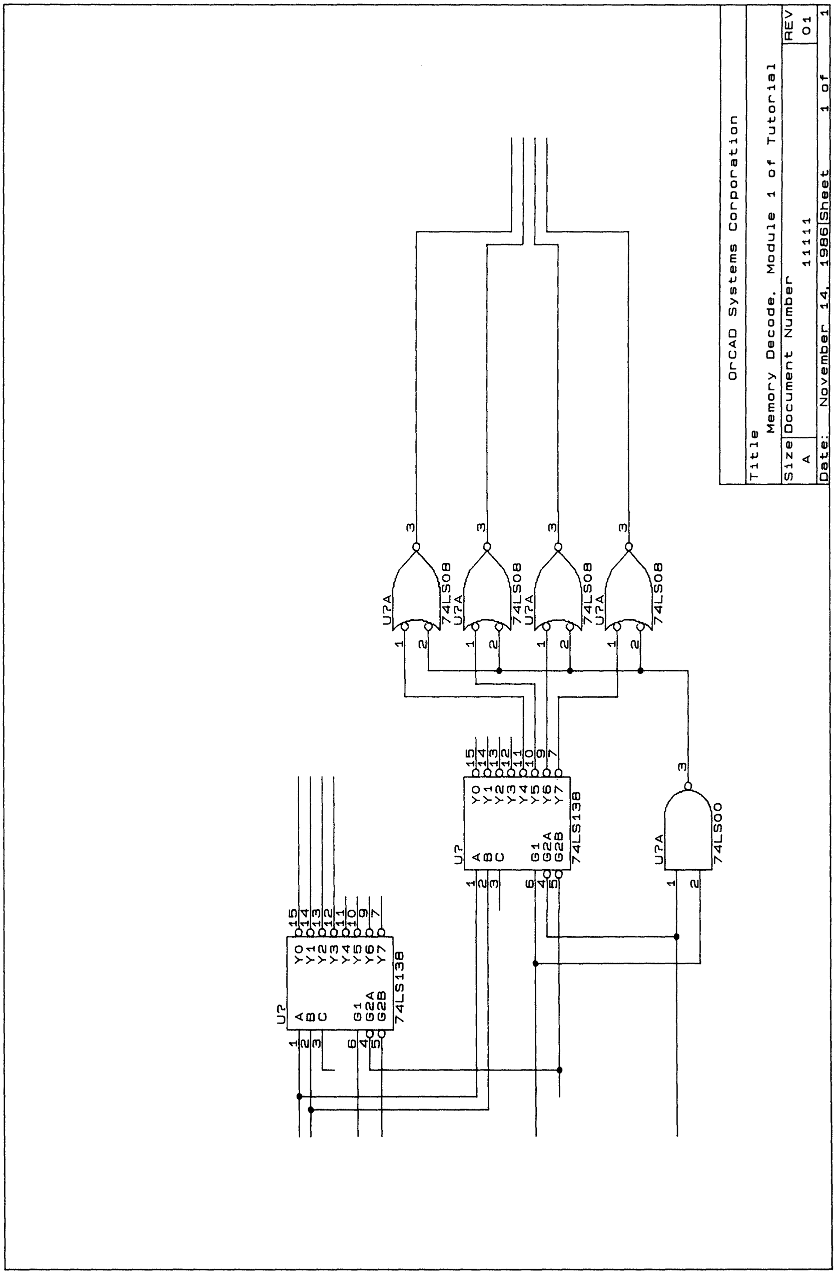


Figure 8-1. The Worksheet after Performing Module 1

2. Specify the name of your work file when DRAFT requests a load file. If you invoke DRAFT without a worksheet, you receive the prompt load file? after the company logo and the copyright message. Respond with the name of your worksheet.

   C:\ORCAD>DRAFT

   <OrCAD logo appears; type any key to continue.>
   <Copyright information appears; type any key to continue.>

   load file?TUTOR.SCH

3. Specify the name of the file when you update it or write it. If youinvoke DRAFT without a worksheet and respond with an <ENTER> when DRAFT requests a load file, DRAFT assumes you are working with an unnamed worksheet. If you choose to write that unnamed worksheet to a file, DRAFT requests the name of the file.

   C:\ORCAD>DRAFT

   <Respond with <ENTER> when load file? appears. Construct the file. Choose the QUIT command. Choose the Write command.>

   Write to file?TUTOR.SCH

Action: **Panning across the Worksheet.** First, remove the menu from the screen by pressing the right button on your mouse. This is equivalent to pressing the <ESC> key. Then, move the mouse to the lower right corner and keep moving it until the title block appears. The screen will pan to keep up with your mouse.

If you set Auto Pan to NO, the screen will not pan; but you can still get to different parts of your sheet by invoking the **JUMP** command.

Now move the mouse to the upper left hand corner, returning the screen to its original location on the work sheet.

Press the left button. The main command menu appears. The highlight is on **AGAIN**. Select the **AGAIN** command by pressing the left mouse button. This executes and repeats the last command that you invoked. Because that command was **SET**, the SET menu reappears.

More SET Options

Backup File determines whether or not a backup file is produced. With this option set, your disk will have two work files when you exit DRAFT, TUTOR.SCH (the file you created) and TUTOR.BAK (the backup file). The backup file is the resulting file from your last edit session. If you're editing a brand new file (as in this example), you of course, won't have a backup. Unless your disk space is severely limited, it is always good practice to let Backup File remain at YES.

same title block you looked at when you tried out autopanning. You may want to disable this default title block and construct one of your own.

Leave **Worksheet Size** set at A. This restricts your worksheet to 8.5 by 11 inches unless you changed the Template Table when you configured OrCAD/SDT III. The actual working area is, of course, somewhat smaller due to space allocated to the border. **X,Y Display** defaults as NO. Change it to YES.

Action: **Setting X,Y Display to YES.** Move the mouse so that the highlight is on X,Y Display. Press the left button. Place the highlight on YES and press the left button again. Move the mouse in any direction and observe the XY coordinates in the upper righthand corner of the screen.

This is a good time to get a feel for the size of your worksheet. The units are inches on the printed worksheet. The upper left corner is 0,0; the upper right corner is 9.50,0.00. Therefore, the maximum X distance is 9.50. The lower left corner is 0.00,7.00. Therefore, the maximum Y distance is 7.00. It comes as no surprise that the lower right corner is 9.50,7.00.

Grid Parameters

If you select **Grid Parameters** from the SET menu, three choices will be displayed: **Grid References, Stay on Grid**, and **Visible Grid Dots**. Selecting any of the three produces a YES/NO menu. Each toggles an environmental setting.

application occurs when you need to produce a series of similar labels -- for example, address signals labeled A0 through A7.

Action: **Closing the SET menu.** Press the right button.

So far you have invoked DRAFT and set the environmental parameters. You have chosen the XY display in the upper righthand corner and made the grid dots visible. Now, it's time to get some parts.

8.3.3. Using the Libraries

This module uses parts from TTL.LIB.

Action: **Extracting a Part from a Library.** Press the left button. Move the mouse to highlight the GET command. Press the left button. The Get? prompt appears.

Press the left button again. You see a list of the configured libraries. Select TTL.LIB; that is, put the highlight on TTL.LIB and press the left button. A part list appears. Scroll down through the part list by moving the mouse until the highlight is on 138. Then, press the left button. Another menu appears. Select the 74LS138.

You have extracted the part 74LS138 from the library. You can see it on your screen, but it's not yet part of your worksheet because you haven't "placed" it. The part moves with your mouse. When it moves, it appears as an outlined symbol. This is a part symbol with much of the detail removed. If you let the mouse stay in one position for a few seconds, the detailed part type 138, you see the menu that lists all the valid parts with 138 as a suffix.

Deleting Parts

You now have two parts on the worksheet. At this point, you could delete or move them. To delete a part, select the **DELETE** command from the main command menu. This displays the DELETE menu, which displays three options: **Object, Block**, and **Undo**. If you want to delete a part, select the Object subcommand and move the mouse until the cursor is placed on the part you want to delete.

Now, you can delete the object in two ways. You can press <d> on your keyboard. Or you press the left button twice, once to display the menu and again to choose Delete. If you had set the environmental characteristic Left Button to YES, pressing the left button once would accomplish the same result.

Because of the way the deletion is done, some dots may remain on the screen. They are not really on the worksheet. Pressing the <ESC> key or the right button causes the screen to redraw, and the extra dots disappear.

Don't be hesitant about deleting one of these parts just for practice. You can restore the part by selecting DELETE from the main command menu and Undo from the DELETE menu.

Moving Parts

The Move command is part of the BLOCK menu. You can move an object or define an arbitrary block on your worksheet for moving. For example, to move one of the 74LS138 decoders, perform the following steps.

Action: **Extracting A Part from TTL.LIB.** From the main command menu, select **GET**. Then, select the library TTL.LIB, then 08, then 74LS08.

Notice that the part appears as an AND gate. To make the schematic for this example, you want the part to appear as a NOR with negated inputs. These two representations are DeMorgan equivalents.

To get the DeMorgan equivalent, press the left button, highlight **Convert,** and press the left button again. Then, place the part as you would ordinarily. To get the normal symbol back, you would repeat the above procedure, but highlight Normal instead of Convert.

The commands along the prompt line should read as follows.

Place Rotate Convert Normal Up Over
Down Mirror Find Jump Zoom escape

If you are not at this point in the menu structure, review the last two Actions, especially the one that describes how to extract the 74LS138. What you want to do is extract a 74LS08, but convert it to its DeMorgan equivalent before placing it.

The other options on the Place menu have to do with the appearance of the screen symbol and screen location. For example, if instead of Convert, you had chosen Mirror you would get the part's mirror image. Find and Jump quickly move the cursor to a different location on the screen. Selecting escape returns you to the main command menu. This may also be accomplished by pressing the right mouse button.

The ZOOM command gives you an overview of the entire worksheet. It's the same ZOOM that appears as part of the main command menu. This is a good time to give it a try.

Action: **Using ZOOM.** Press the left button to display the main command menu. Select **ZOOM**. The ZOOM menu appears. Highlight **Select**, press the left button, highlight 2, and press the left button again. You can now see your whole worksheet on the screen.

To return to normal size, press the left button, highlight **AGAIN**, and press the left button once more. The ZOOM menu reappears. Selecting **AGAIN** always repeats the last item that you selected.

You can choose Select as before, except this time, choose a 1 for normal size. The number displayed in parentheses indicates the level you will zoom into, and 1 is where you began.

So far in this module, you have placed seven parts. One of which, was a DeMorgan equivalent.

command list along the prompt line reads as follows: **Begin End Find Zoom Escape.**

1. Press the left button twice to start drawing.

2. When you want to change direction, press the left button twice.

3. When you want to start a new wire, press the left button, highlight New, and press the left button again. A new wire is one that's not connected to an existing wire. Selecting **New** allows you to place the start of another wire at a different location. It's like picking up the pen when drawing.

4. When you've finished drawing the last wire, press the left button, highlight **End**, and press the left button again. This returns you to the main command level.

A Macro for Placing Wires. Placing wires is a task you will perform frequently when drawing schematics. Although Module 2 describes how to make a macro, it makes sense to include here a short description of a macro to place wires. With this macro, one key gets you from the main command level to the point where moving the cursor draws the wire.

To define the macro on function key <F2>, perform the following steps.

1. From the main command level, select **MACRO**. Then, select **Capture**. The message <macro> appears on the prompt line.

2. Press function key <F2>. Then, press <ENTER>.

3. Press the keys <P>, <W>, and <B>, for PLACE, Wire, and Begin.

Action: **Placing More Wires.** Refer to Figure 8-1 and place the remaining wires.

Place a wire from pin 10 of the lower 74LS138 to pin 1 of the second 74LS08. The wire is vertical along X=4.40.

Place a wire from pin 8 of the lower 74LS138 straight across to pin 1 of the third 74LS08.

Place a wire from pin 7 of the lower 74LS138 to pin 1 of the fourth 74LS08. The wire is vertical along X=4.30.

Place a wire from pin 2 of the first 74LS08 all the way down to (4.5, 5.8). Then, turn left and connect the wire to pin 3 on the 74LS00.

Place a wire from pin 1 of the 74LS00 to (1.00, 5.70).

Place a wire from 6 of the lower 74LS138 to (1.00, 4.50). Then place another wire from pin 5 to (1.30, 4.70). Then, connect pin 4 to pin 1 of the 74LS00. This wire is vertical along X=2.50. This is not a true connection until you place a junction. Placing a junction comes later.

Place a wire from pin 3 of the second 74LS08 to (8.50, 4.40). This wire is vertical along X=7.70.

Place a wire from pin 3 of the third 74LS08 to (8.50, 4.50). This wire is vertical along X=7.70.

Place a wire from pin 3 of the fourth 74LS08 to (8.50, 4.60). This wire is vertical along X=7.80.

8.3.5 Placing Junctions

Wires that cross do not represent a connection. To make a connection, you must place a junction. A junction is unnecessary if the two wires are placed end-to-end. For example, you need a junction where the wire from pin 4 (G2A) of the lower decoder (the lower 74LS138) connects to the wire from pin 1 of the NAND gate.

Action: **Placing a Junction.** From the main command menu, select PLACE. From the PLACE menu, highlight Junction and press the left button. Move the mouse to where you want the junction. Press the left button twice. You can now place another junction by moving the mouse to another location and pressing the left button twice. When you no longer have junctions to place, press the right button to escape.

Action: **Editing the Title Block.** From the main command menu, select the EDIT command. Make sure that the cursor is within the title block. Then, press the left mouse button twice. A menu indicating the fields in the title block appears. Select each field in turn and type the appropriate information.

For example, to enter the Revision code, highlight **Revision code** and press the left button. The prompt Revision code?, appears on the prompt line. Type in: 01, for the revision code followed by <ENTER>.

Notice that the title block does not yet show what you typed.

Now, highlight the next field, **Title of sheet**. Continue through the fields until you've entered all the information you want to place. You can leave fields blank. For example, you have the option of four address lines, but Figure 8-1 doesn't show any in use.

When you've completed entering the information in the title block, press the right button, or the <ESCAPE> key. Notice that the information now appears on the screen. You are back at the main command level.

8.3.7 Updating the Worksheet

At this point, you can choose either to QUIT and Update the file or QUIT and Write to a file. The difference is that update defaults to the currently

8.3.9 Exiting DRAFT

Time for a break? You bet!

Action: **Exiting DRAFT**. From the main command menu, select **QUIT**. Then, select **Abandon Edits**. The DOS prompt appears.

This is the end of Module 1. Module 2 completes the schematic that we began in Module 1. Module 3 gives examples of using the OrCAD utilities.

8.4 MODULE 2: CONSTRUCTING A WORKSHEET, PART 2

Summary: This module completes the drawing of the sample schematic. It includes creating and placing a custom part, defining a macro, adding input and output module ports, placing a bus, and placing power symbols.

Module 1 required one part library, TTL.LIB. Module 2 requires two additional libraries: DEVICE.LIB, and a custom library. Figure 8-2 shows the worksheet at the conclusion of Module 2.

8.4.1. Creating the Custom Library

Module 2 requires the use of a time-delay chip that is not in the OrCAD-supplied libraries. You must create a custom library that contains the part definition. With OrCAD/SDT III, there are a couple of ways to create library parts. One method, uses the graphical object editor called LIBEDIT, in which parts are created on the screen. The other method, uses a text file to describe the part and other utilities to convert the source to an object file. Since, Section 7 outlines the procedure for using LIBEDIT, we will use the second method to create the custom part required in this module.

To do that, we will: 1) create a library source file, 2) run COMPOSER on that source file to create a library file, and 3) reconfigure DRAFT to use the custom library. For additional library information, refer to Section 7.

To create the source file, use a text editor that makes a pure ASCII file with no embedded control characters. This is the same kind of text editor, you would use to create program source files. For example, either Wordstar in the non-document mode or EDLIN satisfies this requirement.

Action: **Creating a Custom Library.** Use a text editor to create the following text file. Call the file TUTOR.SRC and place it in the library directory. Enter the following text in the file.

command. However, if the Module 1 environment is one you plan to use frequently, it's convenient to define a macro. The action below defines the function key <F1> as a macro that makes the grid dots visible and starts up the XY display.

Action: **Defining a Macro**. Press the left button to display the main command menu. Select **Macro**; that is, highlight Macro and press the left button. Then, select **Capture**.

The message, "Capture macro?" appears on the prompt line. Press the <F1> function key. The characters F1 appear on the prompt line. Then, press <ENTER>. The message <macro> appears on the prompt line. This begins the macro definition.

Press <s>. The SET menu appears. Then, press <g>. The grid parameters menu appears. Press <v>, then <y>. You are now back at the main command level.

Press <s>. The SET menu reappears. Press <x>, then <y>. You are now at the main command level. Press <m> to end the macro definition.

Note that in Module 1 you defined a macro on F2 and ended the macro capture with CTRL-END, rather than <m>. If you end a macro capture in the middle of a command sequence, as in Module 1, you must use CTRL-END. At the main command level, either CTRL-END or <m> work. The macro defined above would accept either CTRL-END or <m>. The macro in Module 1 required CTRL-END.

depending on the design. For example, all buses require a Label to be placed on it.

The worksheet from Module 1 has a number of unlabeled inputs and outputs. The next Action describes how to place the input module ports.

Action: **Placing Module Ports.** From the main command level, press the left button. The main command menu appears. Select PLACE. Then, select Module Port. The prompt line reads Module Port Name?

Type the name of the first module port. In this case, it's A18. Follow it with an <ENTER>. Another menu appears, identifying the type of module port. Select **Input**.

The module port appears on the screen. Move it so that its right tip touches the input line to pin 1 on the first decoder. Press the left button twice to place the module port.

Once again, the prompt line reads, Module Port Name? Continue the above procedure and place all the input module ports. They are as follows.

Input Module Port Name	XY Location
A17	(0.60, 2.60)
DACK0BRD\	(0.10, 3.00)
CAS\	(0.50, 3.20)
RAS	(0.60, 4.50)
RAM ADDR SEL	(0.20, 4.70)
DACK 0	(0.40, 5.70)

This concludes the input module ports. Placing an output module port is similar. Just select **Output** instead of **Input** after you type the module port name.

Action: **Placing Bus Entries.** From the main command menu, select PLACE. Then, select Bus (entry). A slash appears at the cursor tip.

For this particular schematic, you want a backslash rather than a slash. (This decision is cosmetic only.) Press the left button, highlight the \, and press the left button again. Move the cursor so that the backslash connects the wire from pin 15 to the top decoder to the top of the bus. Press the left button twice. Now, press the <ESCAPE> key. Next, we will use the REPEAT command to place the other three Bus (entries). Press, <r> three times. Notice, that the remaining entries are automatically placed.

8.4.5 Label and Module Port Conventions

In a design, a label must be placed on every bus that is used in a worksheet. This informs the NETLIST utility program how many members are associated with a bus. Labels are placed in a worksheet with the PLACE Label command.

Bus labels must be in the form:

BUSNAME[0..n]

Where BUSNAME is called the "prefix" and represents the name of the bus. [0..n] is called the "suffix", where n represents the decimal number of the last member of the bus; only a zero (0) is valid in the first portion of the suffix ([2..n] for example, is not valid. Between the prefix and the suffix there must be no space.

Action: **Placing Labels on Bus Signals.** From the main command menu, select **PLACE**. Then, select **Label**. The prompt, Label? appears on the prompt line. Type the label, CAS\0, followed by an <ENTER>.

Move the label until it rests on the wire output from pin 15 of the top decoder. This is XY location (3.70, 2.50). Press the left mouse button twice to place it. The prompt, Label? reappears. Press the <ESCAPE> key to return to the main command menu level. To place the remaining three labels we will use the REPEAT command. Now, press <r> three times. Notice that the remaining three label names were numerically incremented and placed in the worksheet.

Action: **Placing a Label on a Bus.** From the main command menu, select **PLACE**. Then, select **Label.** The prompt, Label? appears on the prompt line. Type the label, CAS\[0..3], followed by an <ENTER>.

Move the cursor and place the label on the bus. A good location is (4.30, 2.90).

8.4.7 Placing Power Objects

The lower decoder takes +5V on pin 3; the upper decoder has its pin 3 grounded. Both the ground symbol and the +5V symbol are power objects, even though the ground symbol comes from a library (DEVICE.LIB) and the +5V comes from the PLACE menu.

Action: **Connecting Power Objects.** Place another power object with value VCC at (7.40, 0.70). Also, place a power object with value +5V at (7.90, 0.70). Connect them to each other with a wire.

8.4.8 A Complete Circuit

Now is a good time to update the work file with the Update command in the QUIT menu. Figure 8-3 shows how your worksheet looks so far. You can print it out with the HARDCOPY command.

8.4.9 Placing More Parts

What remains to be done in Module 2 is to add a portion of the circuit that delivers the RAS and CAS\ signals. These derive from the XMEMW\ and XMEMR\ signals (not yet shown). This involves placing three more parts on the worksheet.

Action: **Placing Three More Parts.** Place a 74LS00 at (6.50, 1.60). Place its DeMorgan equivalent at (3.00, 0.90). Then, place the custom part Time Delay from TUTOR.LIB at (4.80, 1.00).

Action: **Drawing More Wires.** Draw two wires, one beginning at (1.00, 1.00) and connecting to the leftmost 74LS00's top input pin. Draw the other wire from (1.00, 1.20), connecting to the leftmost 74LS00's bottom input pin.

Connect the output of the leftmost 74LS00 to pin 1 on the time delay chip. Then, connect pins 8 and 12 to the inputs on the second 74LS00.

Finally, bring a wire from pin 10 of the time delay chip to the right end of the worksheet, vertically even with the end of the bus.

Action: **Placing Module Ports.** Place three module ports. XMEMW\ and XMEMR\ are input to the first 74LS00, and ADDR SEL is output from the time delay chip.

that the user interface to the commands is somewhat different.

Action: **Setting Left Button to YES.** From the main command menu, select **SET**. Then, select **Left Button**, followed by Yes.

Now delete the CAS\ input to the upper decoder. The new CAS\ line will come from the output of the second 74LS00.

Action: **Deleting the CAS\ Line.** Move the cursor to the CAS\ input of the top decoder at (1.10, 3.20). From the main command menu, select **DELETE**. Then, highlight **Object** and press the left button twice. The wire is now deleted.

Put the cursor over the CAS\ module port. Press the left button. The module port is now deleted.

From the main command menu, select **PLACE**, then **Wire**. Place the cursor at pin 5 of the second decoder at (1.50, 3.20). Press the left button and begin drawing in the left direction until (1.10, 3.20). Then, press the left button and go up until (1.10, 2.20). Press the left button and go right until (7.40, 2.20). Press the left button and go up to the output of the NAND gate (7.40, 1.80). Press the left button and hold it down. Move the highlight to **End** and raise the left button.

Notice that drawing lines with Left Button set to YES reduces the number of button presses you must perform. It associates one of the button presses with a button release.

The OrCAD utilities are as follows.

ANNOTATE This utility scans an input file and automatically updates reference designators. This includes updating the corresponding pin numbers that are associated with a particular instance of a device with multiple parts per package. ANNOTATE modifies your work file; but it creates a backup file that contains the original copy of the work file.

BACKANNO Use this utility when you want to update the reference designators after you've run ANNOTATE. For example, assume that you sent your design out for layout; and when it comes back, what you had designated as U1A is now U1B. You can run BACKANNO and supply as input your initial work file and a was/is text file that contains the new translation.

CLEANUP This utility scans a work file and checks for overlapping parts. It removes duplicate or overlapping wires, buses, and junctions. It displays warning messages advising you of duplicate objects. CLEANUP may modify your work file; but it creates a backup file that contains the original copy of your work file.

CLEANUP does not check for objects overlapping part leads and wires overlapping buses (including wire entries of a bus overlapping bus entries to a bus).

ERC This utility performs a classical electrical rules check. It flags unused inputs on parts, unlabeled wires connected to a bus, and invalid connections, such as two outputs wired together.

NETLIST This utility generates a net and wire list in a number of possible formats. The default format is EDIF.

Figure 8-4 shows how the worksheet looks after running ANNOTATE. Notice the updated reference designators on the devices with multiple parts per package. For example the U?A on the 74LS00s connected to the output of the lower decoder changed to U1A, U1B, U1C, and U1D. They are all parts of the same package, and their pin numbers changed accordingly.

NOTE

> This assumes that the objects placed in the TUTOR.SCH schematic, were placed in the order described in this tutorial. If they were not, the schematic may not be annotated in the order illustrated in Figure 8-4.

8.5.2 BACKANNO

Assume that, in Figure 8-4, you really wanted U2B to be designated as U2A. You also wanted U2C to be U2B and U2A to be U2C. You may find that your board lays out better with a pin rearrangement on the 74LS00 designated as U2.

You can construct a was/is file and use the BACKANNO utility. A was/is file is a text file that lists the old reference designator followed by what you want the designator to be. For example, here is a was/is file that redefines the U2 74LS00.

```
U2B  U2A   U2C  U2B   U2A  U2C
```

When BACKANNO changes the reference designators it updates the pin numbers correspondingly. To run BACKANNO on the file TUTOR.SCH, type the following.

C:\ORCAD>BACKANNO TUTOR.SCH WASIS /O

This example assumes that TUTOR.SCH is in the sheet directory and that the wasis file is in the DRAFT directory.

8.5.3 CLEANUP

Typically, CLEANUP is the second utility you run. CLEANUP corrects simple drawing errors that may inadvertently show up as errors in ERC's electrical rules check. Note that CLEANUP will not fix up wires that overlap body object pins. To run CLEANUP on the work file TUTOR.SCH, type the following.

C:\ORCAD>CLEANUP TUTOR.SCH /O

A typical test is to ensure that all inputs to a device are connected. For example, if you neglected to connect pin 3 of the upper decoder to ground, you get the following message.

> WARNING - INPUT has NO Driving Source U3,C

Also, you must label every bus. For example, if in TUTOR.SCH, you did not include the bus label CAS\[0..3], ERC would display the following message.

> <<<WARNING>>> Unconnected MODULE PORT "CAS\[0..3]" at X =8.10 at Y= 2.90

By connecting the two power objects, VCC and +5V, together, you generate the following warning.

> WARNING - POWER Supplies are CONNECTED VCC <-> +5V

This is something you intended. You can safely ignore the warning.

8.5.5 NETLIST

The NETLIST utility generates a netlist in a variety of formats, the default of which is EDIF.

To create a proper net list, you need to show care when dealing with bus labels, module ports, and power objects. Refer to Section 6 for a detailed explanation of these requirements. Outlined below, is a list of some general guidelines.

1. Thoroughly read the discussion on the NETLIST utility located in Section 6.

NETLIST lets you know that it is working by displaying a sequence of asterisks (*) and periods (.). The file containing the CALAY netlist is as follows.

```
/N00001 U2(6) U4(6) U5(1) U2(2);
/N00002 U5(12) U2(10);
/N00003 U5(8) U2(9);
/N00004 U2(8) U3(5);
/N00005 U1(1) U4(11);
/N00006 U1(2) U1(5) U1(10) U1(13) U2(3);
/N00007 U1(4) U4(10);
/N00008 U4(9) U1(9);
/N00009 U4(7) U1(12);
/N00010 U2(4);
/N00011 U2(5);
/N00012 U5(10);
/N00013 U4(1) U3(1);
/N00014 U4(2) U3(2);
/GND    U3(3) U2(7) U5(7) U3(8) U1(7) U4(8);
/N00016 U3(6);
/N00017 U3(4) U4(5);
/N00018 U1(3);
/N00019 U1(6);
/VCC    U4(3) U2(14) U5(14) U3(16) U1(14) U4(16);
/N00021 U1(8);
/N00022 U4(4) U2(1);
/N00023 U1(11);
/N00024 U3(15);
/N00025 U3(14);
/N00026 U3(13);
/N00027 U3(12);
```

The component file NET.CMP is as follows.

```
74LS00       U2          shape     -X-      -Y-      0
74LS08       U1          shape     -X-      -Y-      0
74LS138      U3          shape     -X-      -Y-      0
74LS138      U4          shape     -X-      -Y-      0
TIME DELAY   U5          shape     -X-      -Y-      0
```

8.5.6 PARTLIST

This utility creates a list of all the parts used in the worksheet. You can specify an Include file if you want that list to contain additional information. To run PARTLIST on TUTOR.SCH without an Include file, type the following.

To specify this Include file when you invoke PARTLIST, type the following.

```
C:\ORCAD>PARTLIST  TUTOR.SCH
TUTOR.1ST   TUT.INC /I /O
```

Note that TUTOR.SCH is in the sheet directory and that TUTOR.1ST is generated by PARTLIST and is placed in the ORCAD directory. Finally, the include file TUT.INC should be placed in the ORCAD directory.

The resulting part list now looks as follows.

```
 Memory Decode, Module 2 of Tutorial  Revised:  November 16, 1986 11111
Revision: Bill Of Materials      November 16, 1986      22:18:28  Page 1
Item Qty Ref Part      DESCRIPTION     Part Order Code
____________________________________________________________
1    1   U2     74LS00      TTL Quad Two Input NAND Gate 10003040000
2    1   U1     74LS08      TTL Quad Two Input AND Gate  10003050000
3    2   U3,U4  74LS138     TTL Decoder/Demultiplexer    10003060000
4    1   U5     TIME DELAY Gate Delay 25/125  10006400000
```

8.5.7 PRINTALL, PLOTALL

These utilities permit the printing or plotting of files in batch mode.

With the PRINTALL utility, you can print the schematic file without invoking DRAFT. For example, to print the file TUTOR.SCH, type the following.

```
C:\ORCAD>PRINTALL TUTOR.SCH /O
```

The PRINTALL utility is most useful for hierarchical structures and flat file structures. For example, TUTOR.SCH is a one-sheet schematic that could be a member of a flat file structure. A flat file structure is a text file containing the names of a number of one-sheet schematics. You can print the entire collection by invoking PRINTALL and specifying the flat file filename.

Directory of ASSEMBLY.LIB

08 PIN	14 PIN	16 PIN	24 PIN
28 PIN	40 PIN BOTTOM	40 PIN TOP	CAP 2X7
EDGE CONNECTOR	RESISTOR 1X2	RESISTOR 1X3	RESISTOR 1X5
RESISTOR PACK	TO 220	TO 39	TO 92
XTAL			

Directory of CMOS.LIB

4000	4001	4002	4006	4007	4008	4009	4010	4011
4012	4013	4014	4015	4016	4017	4018	4019	4020
4021	4022	4023	4024	4025	4026	4027	4028	4029
4030	4031	4032	4033	4034	4035	4037	4038	4040
4041	4042	4043	4044	4045	4046	4047	4048	4049
4050	4051	4052	4053	4054	4055	4056	4057	4059
4060	4063	4066	4068	4069	4070	4071	4072	4073
4075	4076	4077	4078	4081	4082	4085	4086	4089
4093	4094	4095	4096	4098	4099	4501	4502	4503
4504	4505	4506	4508	4510	4511	4512	4513	4514
4515	4516	4517	4518	4519	4520	4521	4522	4524
4526	4527	4528	4529	4530	4531	4532	4534	4536
4537	4538	4539	4541	4543	4544	4547	4549	4551
4552	4553	4554	4555	4556	4557	4558	4559	4560
4561	4562	4566	4568	4569	4572	4573	4574	4575
4580	4581	4582	4583	4584	4585	4597	4598	4599
5101	14000	14001	14002	14006	14007	14008	14011	14012
14013	14014	14015	14016	14017	14018	14020	14021	14022
14023	14024	14025	14027	14028	14029	14032	14034	14035
14038	14040	14042	14043	14044	14046	14051	14052	14053
14066	14068	14069	14070	14071	14072	14073	14075	14076
14077	14078	14081	14082	14093	14094	14099	14160	14161
14162	14163	14174	14175	14194	14501	14502	14503	14504
14505	14506	14508	14510	14511	14512	14513	14514	14515
14516	14517	14518	14519	14520	14521	14522	14524	14526
14527	14528	14529	14530	14531	14532	14534	14536	14537
14538	14539	14541	14543	14544	14547	14549	14551	14552
14553	14554	14555	14556	14557	14558	14559	14560	14561
14562	14566	14568	14569	14572	14573	14574	14575	14580
14581	14582	14583	14584	14585	14597	14598	14599	40100
40101	40102	40103	40104	40105	40106	40115	40116	40160
40161	40162	40163	40174	40175	40182	40192	40193	40194
42100	45100	45101	45104	45106	45107	45109	45112	145104
145106	145107	145109	145112					

MC10162	MC10163	MC10164	MC10165	MC10166	MC10168	MC10170
MC10171	MC10172	MC10173	MC10174	MC10175	MC10176	MC10177
MC10178	MC10179	MC10180	MC10181	MC10182	MC10186	MC10188
MC10189	MC10190	MC10191	MC10193	MC10195	MC10197	MC10210
MC10211	MC10212	MC10216	MC10231	F10402	F10414	F10415
F10416	F10422	F10470	F10474	MC10500	MC10501	MC10502
MC10503	MC10504	MC10505	MC10506	MC10509	MC10514	MC10515
MC10516	MC10517	MC10518	MC10519	MC10521	MC10530	MC10531
MC10532	MC10534	MC10535	MC10536	MC10537	MC10538	MC10539
MC10541	MC10558	MC10559	MC10560	MC10561	MC10562	MC10563
MC10564	MC10565	MC10566	MC10568	MC10570	MC10571	MC10572
MC10574	MC10575	MC10576	MC10578	MC10579	MC10580	MC10581
MC10582	MC10586	MC10588	MC10590	MC10591	MC10593	MC10595
MC10597	MC10610	MC10611	MC10612	MC10616	MC10631	F100101
F100102	F100107	F100112	F100113	F100114	F100117	F100118
F10012	F100123	F100124	F100125	F100126	F100130	F100131
F100136	F100141	F100142	F100145	F100150	F100151	F100155
F100156	F100158	F100160	F100163	F100164	F100165	F100166
F100171	F100179	F100180	F100181	F100182	F100183	F100194
F100402	F100414	F100415	F100416	F100422	F100470	F100474
F100170/4						
F100170/8						

Directory of INTEL.LIB

Z80	Z80PIO	8031	80C31	8032	8035	80C35
8039	80C39	8040	8041	8042	8044	8048
80C48	8049	80C49	8050	8051	80C51	8052
8085	8086MAX	8086MIN	8087	8088MAX	8088MIN	8089
8096	8097	8155	8156	8185	8203	8205
8206	8207	8208	8212	8216	8226	8231
8237	8237A	8243	8251	8251A	8253	8254
82C54	8255	82C55	8256	8257	8259A	8272
8272A	8273	8274	8275	8276	8279	8282
8283	8284	8286	8287	8288	8289	8291
8292	8294	8294A	8295	8344	8396	8397
8641	8741	8742	8744	8748	8749	8751
8755	80186	80188	80286	80287	80386	8206-2
82062	82064	82188	82284	82288	82289	82384
82501	82530	82586MA	82586MI	82588HI	82588HM	82720
82731						

Directory of MEMORY.LIB

10H8	10L8	12H6	12L6	14H4	14L4	14L8	16C1
16H2	16L2	16A4	16R4	16RP4	16X4	16L6	16PR6
16R6	16H8	16HD8	16L8	16LD8	16P8	16R8	16RP8
16V8	16Z8	18L4	18P8	20C1	20L2	20R4	20X4
20R6	20L8	20R8	20V8	20X8	315	406	426
12L10	1400	1420	1421	1430	1600	1601	18S42
18S46	18SA46	20L10	20X10	2015	2016	2018	2019

Directory of RF.LIB

AIR INDUCTOR	AIR INDUCTOR VARIABLE	AIR INDUCTOR WIPER
AIR T INDUCTOR	ANTENNA DIPOLE	ANTENNA NETWORK
ANTENNA-1	ANTENNA-2	ARROW
BOX	CAPACITOR FEED	CIRCLE
COAX PLUG	COAX RECEPTACLE	CONNECTOR COAX
CONNECTOR COAX-F	CONNECTOR COAX-M	CRT
ENVELOPE	GROUNDED CAP	HOTLINE JACK
PENTODE	PHASE SHIFTER	PHASE SHIFTER BETA
PHASE SHIFTER THETA	PSB	PST
RF DPDT	RF SPDT	SWITCH NETWORK
TETRODE	TRANSMISSION LINE	TRANSMISSION SEGMENT
TRIODE	TRIODE PCAP	

Directory of SHAPES.LIB

```
CIR01    CIR02    CIR03    CIR04    CIR05    CIR06    CIR07
CIR08
CIR09    CIR10    CIR11    CIR12    DIA06    DIA12    PARALGM
RADI01
RADI02   RADI03   RADI04   RADI05   RADI06   SQ01     SQ02
SQ03
SQ04     SQ05     SQ06     SQ07     SQ08     SQ09     SQ10
SQ11
SQ12
```

Directory of TTL.LIB Continued

	74LS	74S	74ALS	74AS	74HCT	74HC	74ACT	74AC	74F	74
40	**	**	**	..	..	..	..	..	**	**
42	**	..	..	..	**	**	..	..	..	**
43	..	..	..	..	..	..	..	..	..	**
44	..	..	..	..	..	..	..	..	..	**
45	..	..	..	..	..	..	..	..	..	**
46	..	..	..	..	..	..	..	..	..	**
47	**	..	..	..	..	..	..	..	..	**
48	**	..	..	..	..	..	..	..	..	**
49	**	..	..	..	..	..	..	..	..	**
50	..	..	..	..	..	..	..	..	..	**
51	**	..	..	..	..	**	..	..	**	**
54	**	..	..	..	..	..	..	..	..	**
55	**	..	..	..	..	..	..	..	..	..
56	**	..	..	..	..	..	..	..	..	..
57	**	..	..	..	..	..	..	..	..	..
60	..	..	..	..	..	..	..	..	..	**
63	**	..	..	..	..	..	..	..	..	..
64	..	**	..	..	..	..	..	..	**	..
65	..	**	..	..	..	..	..	..	..	..
68	**	..	..	..	..	..	..	..	..	..
69	**	..	..	..	..	..	..	..	..	..
70	..	..	..	..	..	..	..	..	..	**
72	..	..	..	..	..	..	..	..	..	**
73	**	..	..	..	**	**	..	..	..	**
74	**	**	**	**	**	**	**	**	**	**
75	**	..	..	..	**	**	..	..	..	**
76	**	..	..	..	**	**	..	..	..	**
77	**	..	..	..	..	**	..	..	..	**
78	**	..	..	..	..	**	..	..	..	..
80	..	..	..	..	..	..	..	..	..	**
82	..	..	..	..	..	..	..	..	..	**
83	**	..	..	..	..	..	..	..	**	**
85	**	**	..	..	**	**	..	..	**	**
86	**	**	**	..	**	**	**	**	**	**
90	**	..	..	..	..	..	..	..	..	**
91	**	..	..	..	..	..	..	..	..	**
92	**	..	..	..	**	**	..	..	..	**
93	**	..	..	..	**	**	..	..	..	**
94	..	..	..	..	..	..	..	..	..	**
95	**	..	..	**	..	..	..	..	..	**
96	**	..	..	..	..	..	..	..	..	**

Directory of TTL.LIB Continued

	74LS	74S	74ALS	74AS	74HCT	74HC	74ACT	74AC	74F	74
153	**	**	**	**	**	**	**	**	**	**
154	**	..	..	..	**	**	..	..	..	**
155	**	..	..	..	**	**	..	..	..	**
156	**	..	..	..	..	..	..	..	..	**
157	**	**	**	**	**	**	**	**	**	**
158	**	**	**	**	**	**	**	**	**	..
159	..	..	..	..	..	..	..	..	..	**
160	**	..	**	**	**	**	..	..	**	..
161	**	..	**	**	**	**	**	**	**	..
162	**	**	**	**	**	**	..	..	**	..
163	**	**	**	**	**	**	**	**	**	..
164	**	..	**	..	**	**	..	..	**	**
165	**	..	**	..	**	**	..	..	..	**
166	**	..	**	..	**	**	..	..	**	**
167	..	..	..	..	..	..	..	..	..	**
168	**	**	**	**	..	..	..	..	**	..
169	**	**	**	**	**	**	..	..	**	..
170	**	..	..	..	..	..	..	..	..	**
171	**	..	..	..	..	..	..	..	..	..
172	..	..	..	..	..	..	..	..	..	**
173	**	..	..	..	**	**	..	..	..	**
174	**	**	**	**	**	**	..	..	**	**
175	**	**	**	**	**	**	..	..	**	**
176	..	..	..	..	..	..	..	..	..	**
177	..	..	..	..	..	..	..	..	..	**
178	..	..	..	..	..	..	..	..	..	**
179	..	..	..	..	..	..	..	..	..	**
180	..	..	..	..	..	**	..	..	..	**
181	**	**	..	**	**	**	..	..	**	**
182	..	**	..	..	**	**	..	..	**	**
183	**	..	..	..	..	..	..	..	..	..
184	..	..	..	..	..	..	..	..	..	**
185	..	..	..	..	..	..	..	..	..	**
190	**	..	**	..	**	**	..	..	**	**
191	**	..	**	..	**	**	**	**	**	**
192	**	..	**	..	**	**	..	..	**	**
193	**	..	**	..	**	**	**	**	**	**
194	**	**	..	..	**	**	..	..	**	**
195	**	**	..	..	**	**	..	..	**	**
196	**	**	..	..	..	..	..	..	..	**

Directory of TTL.LIB Continued

	74LS	74S	74ALS	74AS	74HCT	74HC	74ACT	74AC	74F	74
293	**	..	..	..	..	..	..	..	..	**
294	**	..	..	..	..	**	..	..	..	..
295	**	..	..	..	..	..	..	..	..	..
298	**	..	..	**	..	**	..	..	**	**
299	**	**	**	..	**	**	**	**	**	..
320	**	..	..	..	..	..	..	..	..	..
321	**	..	..	..	..	..	..	..	..	..
322	**	..	..	..	..	**	..	..	**	..
323	**	..	**	..	**	**	**	**	**	..
347	**	..	..	..	..	..	..	..	..	..
348	**	..	..	..	..	..	..	..	..	..
350	..	**	..	..	..	..	..	..	**	..
352	**	..	**	**	..	**	..	..	**	..
353	**	..	**	**	..	**	..	..	**	..
354	**	..	..	..	**	**	..	..	..	..
355	**	..	..	..	..	..	..	..	..	..
356	**	..	..	..	**	**	..	..	..	..
357	**	..	..	..	..	..	..	..	..	..
365	**	..	**	..	**	**	..	..	**	**
366	**	..	**	..	**	**	..	..	**	**
367	**	..	**	..	**	**	..	..	**	**
368	**	..	**	..	**	**	..	..	**	**
373	**	**	**	**	**	**	**	**	**	..
374	**	**	**	**	**	**	**	**	**	..
375	**	..	..	..	..	**	..	..	..	..
376	..	..	..	..	..	..	..	..	..	**
377	**	..	..	..	**	**	..	..	**	..
378	**	..	..	..	..	**	..	..	**	..
379	**	..	..	..	..	**	..	..	**	..
381	**	**	..	..	..	**	..	..	**	..
382	**	..	..	..	..	**	..	..	**	..
384	**	..	..	..	..	..	..	..	**	..
385	**	..	..	..	..	..	..	..	**	..
386	..	..	..	..	..	**	..	..	..	..
390	**	..	..	..	**	**	..	..	..	..
393	**	..	..	..	**	**	..	..	..	..
395	**	..	..	**	..	..	..	..	**	..
396	**	..	..	..	..	..	..	..	..	..
398	**	..	..	..	..	..	..	..	**	..
399	**	..	..	..	..	..	..	..	**	..

Directory of TTL.LIB Continued

	74LS	74S	74ALS	74AS	74HCT	74HC	74ACT	74AC	74F	74
560	..	..	**	..	..	..	..	..	..	..
561	..	..	**	..	..	..	..	..	..	..
563	..	..	**	..	**	**	..	..	**	..
564	..	..	**	..	**	**	..	..	**	..
568	..	..	**	..	..	..	..	..	**	..
569	..	..	**	..	..	..	..	..	**	..
573	..	..	**	**	**	**	..	..	**	..
574	..	..	**	**	**	**	..	..	**	..
575	..	..	**	**	..	..	..	..	..	..
576	..	..	**	**	..	..	..	..	..	..
577	..	..	**	**	..	..	..	..	..	..
580	..	..	**	**	..	..	..	..	..	..
588	..	..	..	..	..	..	..	..	**	..
589	**	..	..	..	..	**	..	..	..	..
590	**	..	..	..	**	**	..	..	..	..
591	**	..	..	..	..	..	..	..	..	..
592	**	..	..	..	**	**	..	..	..	..
593	**	..	..	..	**	**	..	..	..	..
594	**	..	..	..	..	**	..	..	..	..
595	**	..	..	..	..	**	..	..	**	..
596	**	..	..	..	..	..	..	..	..	..
597	**	..	..	..	**	**	..	..	**	..
598	**	..	..	..	**	**	..	..	..	..
599	**	..	..	..	..	..	..	..	..	..
604	**	..	..	..	..	**	..	..	**	..
605	**	..	..	..	..	..	..	..	**	..
606	**	..	..	..	..	..	..	..	..	..
607	**	..	..	..	..	..	..	..	..	..
608	**	..	..	..	..	..	..	..	..	..
618	**	..	..	..	..	..	..	..	..	..
619	**	..	..	..	..	..	..	..	..	..
620	**	..	**	**	**	**	**	..	**	..
621	**	..	**	**	..	..	..	..	**	..
622	**	..	**	**	..	..	..	..	**	..
623	**	..	**	**	**	**	**	..	**	..
638	**	..	**	**	..	..	..	..	..	..
639	**	..	**	**	..	..	..	..	..	..
640	**	..	**	**	**	**	**	..	..	..
641	**	..	**	**	..	..	..	..	..	..
642	**	..	**	**	..	..	..	..	..	..

Directory of TTL.LIB Continued

	74LS	74S	74ALS	74AS	74HCT	74HC	74ACT	74AC	74F	74
743	..	..	**	..	..	..	..	..	..	..
746	..	..	**	..	..	..	..	..	..	..
747	..	..	**	..	..	..	..	..	..	..
756	..	..	..	**	..	..	..	..	..	..
757	..	..	..	**	..	..	..	..	..	..
758	..	..	..	**	..	..	..	..	..	..
759	..	..	..	**	..	..	..	..	..	..
760	..	..	..	**	..	..	..	..	..	..
800	..	..	..	**	..	..	..	..	..	..
802	..	..	..	**	..	..	..	..	..	..
804	..	..	**	**	..	**	..	..	**	..
805	..	..	**	**	..	**	..	..	**	..
808	..	..	**	**	..	**	..	..	..	..
821	..	..	..	**	..	..	..	..	**	..
822	..	..	..	**	..	..	..	..	**	..
823	..	..	..	**	..	..	..	..	**	..
824	..	..	..	**	..	..	..	..	**	..
825	..	..	..	**	..	..	..	..	**	..
826	..	..	..	**	..	..	..	..	**	..
832	..	..	**	**	..	**	..	..	..	..
841	..	..	..	**	..	..	..	..	**	..
842	..	..	..	**	..	..	..	..	**	..
843	..	..	..	**	..	..	..	..	**	..
844	..	..	..	**	..	..	..	..	**	..
845	..	..	..	**	..	..	..	..	**	..
846	..	..	..	**	..	..	..	..	**	..
850	..	..	..	**	..	..	..	..	..	..
851	..	..	..	**	..	..	..	..	..	..
852	..	..	..	**	..	..	..	..	..	..
856	..	..	..	**	..	..	..	..	..	..
857	..	..	**	**	..	..	..	..	..	..
866	..	..	..	**	..	..	..	..	..	..
867	..	..	..	**	..	..	..	..	..	..
869	..	..	..	**	..	..	..	..	..	..
873	..	..	**	**	..	..	..	..	..	..
874	..	..	**	**	..	..	..	..	..	..
876	..	..	**	**	..	..	..	..	..	..
877	..	..	..	**	..	..	..	..	..	..
878	..	..	**	**	..	..	..	..	..	..
879	..	..	**	**	..	..	..	..	..	..

Directory of TTL.LIB Continued

	74LS	74S	74ALS	74AS	74HCT	74HC	74ACT	74AC	74F	74
2623	..	..	..	**	..	..	..	..	..	..
2640	..	..	..	**	..	..	..	..	..	..
2645	..	..	..	**	..	..	..	..	..	..
8003	..	..	**	..	..	..	..	..	..	..
11000	..	..	..	..	..	..	**	**	..	..
11002	..	..	..	..	..	..	**	**	..	..
11004	..	..	..	..	..	..	**	**	..	..
11008	..	..	..	..	..	..	**	**	..	..
11010	..	..	..	..	..	..	**	**	..	..
11011	..	..	..	..	..	..	**	**	..	..
11013	..	..	..	..	..	..	**	**	..	..
11014	..	..	..	..	..	..	**	**	..	..
11020	..	..	..	..	..	..	**	**	..	..
11021	..	..	..	..	..	..	**	**	..	..
11027	..	..	..	..	..	..	**	**	..	..
11030	..	..	..	..	..	..	**	**	..	..
11032	..	..	..	..	..	..	**	**	..	..
11034	..	..	..	..	..	..	**	**	..	..
11074	..	..	..	..	..	..	**	**	..	..
11109	..	..	..	..	..	..	**	**	..	..
11112	..	..	..	..	..	..	**	**	..	..
11132	..	..	..	..	..	..	**	**	..	..
11138	..	..	..	..	..	..	**	**	..	..
11139	..	..	..	..	..	..	**	**	..	..
11151	..	..	..	..	..	..	**	**	..	..
11153	..	..	..	..	..	..	**	**	..	..
11157	..	..	..	..	..	..	**	**	..	..
11158	..	..	..	..	..	..	**	**	..	..
11160	..	..	..	..	..	..	**	**	..	..
11161	..	..	..	..	..	..	**	**	..	..
11162	..	..	..	..	..	..	**	**	..	..
11163	..	..	..	..	..	..	**	**	..	..
11168	..	..	..	..	..	..	**	**	..	..
11169	..	..	..	..	..	..	**	**	..	..
11174	..	..	..	..	..	..	**	**	..	..
11175	..	..	..	..	..	..	**	**	..	..
11181	..	..	..	..	..	..	**	**	..	..
11190	..	..	..	..	..	..	**	**	..	..
11191	..	..	..	..	..	..	**	**	..	..
11192	..	..	..	..	..	..	**	**	..	..

DISPLAY DRIVERS LISTED IN ALPHABETIC ORDER

Name	Resolution	Colors	Type
ATT16C.DRV	640x400	16	AT&T 6300 DEB
ATT2C.DRV	640x400	2	AT&T 6300
CGA2.DRV	640x200	2	Color Graphics Adapter
CGA4.DRV	320x200	4	Color Graphics Adapter
CPTFULL.DRV	720x728	1	CPT 9000 Full screen
CPTHGC.DRV	720x348	1	CPT 9000 Half Screen
EGA16C1.DRV	320x200	16	EGA standard monitor
EGA16C2.DRV	640x200	16	EGA standard monitor
EGA16E.DRV	640x350	16	EGA Enhanced monitor
EGA2.DRV	640x350	1	EGA Monochrome monitor
EGA4E.DRV	640x350	4	EGA (64K RAM)
EVA480.DRV	640x480	16	TSENG LABS EVA-480
GENOA528.DRV	640x528	16	GENOA SuperEGA HiRes
GENOA640.DRV	640x480	16	GENOA SuperEGA
GENOA752.DRV	752x410	16	GENOA SuperEGA HiRes
GENOA800.DRV	800x600	16	GENOA SuperEGA HiRes
GENOA912.DRV	912x480	16	GENOA SuperEGA HiRes
HGC2.DRV	720x348	1	Hercules Monochrome
HPVECTRA.DRV	640x400	2	HP Vectra Multi-Video
MATROX64.DRV	640x480	16	Matrox PG-640
NCR2.DRV	640x400	2	NCR graphics adapter
NCR4.DRV	640x400	4	NCR graphics adapter
PEGA640.DRV	640x480	16	Paradise Autoswitch
SIGMA400.DRV	640x400	16	Sigma 400
T3100.DRV	640x400	2	Toshiba 3100 Laptop
TANDY2K.DRV	640x400	2	Tandy 2000 Graphics
TELECAT.DRV	640x400	2	Televideo TeleCAT
TGM16I.DRV	640x400	16	Tecmar Graphics Master
TGM16N.DRV	640x200	16	Tecmar Graphics Master
TGM2I.DRV	720x696	1	Tecmar Graphics Master
TGM2N.DRV	720x348	1	Tecmar Graphics Master
TGM4I.DRV	720x400	4	Tecmar Graphics Master
TGM4N.DRV	720x200	4	Tecmar Graphics Master
VECTRXAT.DRV	1024x1024	16	Vectrix PEPE (AT)
VECTRXPC.DRV	1024x1024	16	Vectrix PEPE (PC)
VEGA640.DRV	640x480	16	Video-7 VEGA DELUXE
VEGA752.DRV	752x410	16	Video-7 VEGA DELUXE
WYSE700A.DRV	640x400	2	Wyse-700
WYSE700B.DRV	1280x400	2	Wyse-700
WYSE700C.DRV	1280x800	2	Wyse-700

NOTE ON USING AT&T COLOR GRAPHICS DRIVER

Before this driver can be used, the device driver supplied with the DEB card must be installed. The reference manual from AT&T has all of the needed information. In summary, CONFIG.SYS must contain a line:: DEVICE=DEDRIVER.DEV

The driver file name may vary based on revision. Also, account for the path to the file.

The DEB card requires special modification to the standard monochrome card to be properly set up and to allow color to be

HPLASER3.DRV	HP	LaserJet+,LaserJet II (150 dpi)
HPLASER4.DRV	HP	LaserJet+,LaserJet II (300 dpi)
HPINKJET.DRV	HP	ThinkJet
JDL750.DRV	JDL	750
JDL850.DRV	JDL	850
NECPR201.DRV	NEC	PR-201
NEC8023.DRV	NEC	8023
TI850.DRV	TI	850
TOSHIBA.DRV	Toshiba	P1340/P1350/P1351/P351

PLOTTER DRIVERS

Name	Brand
HP.DRV	Use on all Hewlett Packard
HI.DRV	Use on all Houston Instrument
WG.DRV	Use on all Western Graphtek
CALCOMP1.DRV	Refer to Below
CALCOMP2.DRV	Refer to Below

NOTE ON CALCOMP PLOTTER DRIVERS

Outlined below is a list of the supported Calcomp drivers. Note that you must have one of the listed Controllers (supplied by Calcomp) in order for the plotter to work. Please refer to your Calcomp Appendix for your systems configuration. This is a difficult plotter to support, since there are many modes of operation that are allowed. Be sure you use the correct OrCAD driver for the configuration that you have.

```
        DRAWING MANAGEMENT    ( CONTROL KEYS )
        *************************************

^F1:  List Components        {^F1}=LD{MACROBREAK}{ENTER}{}
^F2:  Place Junction         {^F2}=PJP{ESC}{}
^F3:  Copy Block
{^F3}=BS{MACROBREAK}{ENTER}{MACROBREAK}{MACROBREAK}
                                   BG{}
^F4:  Delete Block
{^F4}=DB{MACROBREAK}{ENTER}{MACROBREAK}{}
^F5:  Move Block
{^F5}=BM{MACROBREAK}{ENTER}{MACROBREAK}{}
^F6:  Edit Part Reference    {^F6}=EERN{MACROBREAK}{ESC}{ESC}{}
^F7:  Edit Title Block
{^F7}=JX450{ENTER}JY450{ENTER}EE{MACROBREAK}{}
^F8:  Change Worksheet Size
{^F8}=ZS{D}{D}{D}{ENTER}{ESC}SW{MACROBREAK}ZS{ENTER}
                                   {ESC}{}
^F9:  Place Text             {^F9}=PT{MACROBREAK}{ENTER}{ESC}{}
^F10: Show Memory            {^F10}=C{}


        ENVIRONMENTAL MANAGEMENT    ( ALTERNATE KEYS )
        *********************************************

\F1:  SET Auto Pan Off       {\F1}=SAN{}
\F2:  SET Auto Pan On        {\F2}=SAY{}
\F3:  Reference Window
{\F3}=ZS{D}{D}{D}{ENTER}{MACROBREAK}P{ENTER}ZS1
                                   {ESC}{}
\F4:  Undo Last Delete       {\F4}=DU{}
\F5:  ZOOM In                {\F5}=ZI{ESC}{}
\F6:  ZOOM Out               {\F6}=ZO{ESC}{}
\F7:  Enter Sheet            {\F7}=QUEE{ESC}{}
\F8:  Leave Sheet            {\F8}=QUL{ESC}{}
\F9:  SET Grid Off           {\F9}=SGVN{ESC}{}
\F10: SET Grid On            {\F10}={ENTER}SGVY{ESC}{}

\I: * GET IOB Symbol         {\I}=BI..\LCALIB2K\IOB.BLK{ENTER}{}
\C: * GET CLB Symbol         {\C}=BI..\LCALIB2K\CLB.BLK{ENTER}{}
\P:   GET Power Symbol       {\P}=PPV{ESC}VVCC{ENTER}TC{}
\G:   GET Ground Symbol      {\G}=PPV{ESC}VGND{ENTER}TAOB{}

* This assumes that you have these objects created in your library


        FILE MANAGEMENT    ( SHIFT KEYS )
        ********************************

SHIFT-F1:   QUIT Abandon Edits     {SHIFT-F1}=QAY{}
SHIFT-F2:   Clear Worksheet        {SHIFT-F2}=DBJX450{ENTER}JY450{ENTER}BJX-
450{ENTER}
                                        JY-450{ENTER}E{ESC}{}
SHIFT-F3:   BLOCK Import           {SHIFT-F3}=BI{MACROBREAK}{}
SHIFT-F4:   Block Export           {SHIFT-F4}=BE{MACROBREAK}{ENTER}{}
SHIFT-F5:   Load Schematic File    {SHIFT-F5}=QI{MACROBREAK}{ESC}{}
SHIFT-F6:   Update Schematic File  {SHIFT-F6}=QU{MACROBREAK}{ESC}{}
SHIFT-F7:   Write To File          {SHIFT-F7}=QW{MACROBREAK}{ESC}{}
SHIFT-F8:   Print Schematic        {SHIFT-F8}=HM{}
```

D.1 PSEUDO CODE SUMMARY OF A PRINTER DRIVER

A printer driver consists of one far procedure called Printer_Driver. The routines within the driver are accessed via jmp instructions. The order of the jmp instructions is very important.

For example, the first jmp is to the routine Initialize_Driver. When the driver is loaded, the first routine in the jmp list is the one that gets executed. Consequently, this routine must be Initialize_Driver.

Here is a pseudo code summary of a printer driver. The driver for the HP Laser Jet Plus is used as an example.

```
Macro definitions
        Define Save_DS
        Define Restore_DS

Error Code definitions
        E_OK                                    equu  0
                                                .
                                                .
                                                .
        E_No_More_Files                         equ  18

STACK segment stack 'STACK'
    Defines a stack segment called STACK. This is
    defined as an empty segment. The calling program
    sets up the stack.
STACK ends

PRINTER segment public 'PRINTER'
    Defines a public segment called PRINTER. This is
    the code and data segment for the printer driver.
Printer_Driver                                  proc  far
        List of jmp instructions to the various routines
```

Data Area

Reserves space for variables and the driver buffer. Also defines variables specific to the printer. For example, the HP Laser Jet Plus requires the variable Enter_Graphics. It is defined as the hexadecimal number 1B 2A 74 37 35 52.

This area also includes the printer information table. Finally, the data area must include the variable End_Of_File_Location. The routine Initialize_Driver uses this variable's location to determine the size of the driver so that it can return unused memory to the system.

PRINTER ends
end

D.2 ASSEMBLY AND LINKAGE

The drivers shown here are intended to be assembled with the Microsoft Macro Assembler and linked with Microsoft Link.

For example, consider the driver for the HP Laser Jet Plus. Assume that its source code resides in a file called HPLASER1.ASM. To assemble HPLASER1.ASM, issue the following command (on one line):

MASM HPLASER1 /N /T /Z, HPLASER1, HPLASER1 /B63;

where:

/N	Suppresses tables in listing file.
/T	Suppresses messages for successful assembly.
/Z	Displays error lines on the screen.

D.3 A MORE DETAILED DESCRIPTION OF A PRINTER DRIVER

D.3.1 Macro Definitions

A printer driver uses two macro definitions. One is called Save_DS, and the other is called Restore_DS. The driver routines listed in the jmp list all begin with Save_DS and end with Restore_DS.

Save_DS pushes the current data segment register (DS) and the current code segment register (CS) onto the stack. It then pops the value of the pushed code segment register into the data segment register. Finally, it tells the assembler (with an assume statement) that the new data segment is the PRINTER segment defined in the driver.

Restore_DS pops the value of the pushed data segment register into the data segment register, thus restoring the value it had before the routine was called. The driver then tells the assembler (with an assume statement) not to expect any particular segment descriptor in DS.

D.3.2 Error Codes

The driver defines a number of symbols to represent the conventional DOS error codes. Note, however, that the driver defines error code 14, normally undefined by DOS. OrCAD defines 14 as the error E_Disk_Full. The driver gets that error if it writes to a file and the number of bytes written is not the same as the number to be written.

D.3.3 Segment Definitions

The driver defines two segments, one called STACK and one called PRINTER. The driver's data and code

First, the routine displays a sign-on message. If you are writing your own driver, you will probably want to make your own sign-on message.

Then, the driver calculates its size. To do this, it makes use of the variable End_Of_File_Location. Consequently, this variable must be the last byte in the program. The driver then returns unneeded memory to DOS.

D.3.6 Get_Printer_Information and Get_Driver_Buffer_Address

The two routines Get_Printer_Information and Get_Driver_Buffer_Address return a pointer in ES:BX.

Get_Printer_Information returns a pointer to the printer information table. This table consists of eight 16-bit words. It must have the following format:

Number of bits per line for narrow paper
Number of bits per line for wide paper
Number of bits per horizontal inch
Number of bits per vertical inch
Number of lines per printhead pass
Number of printhead passes per page
The size of the driver buffer in bytes
The size of the driver buffer row in bytes

If your printer has continuous feed, and you are not trying to skip over perforations, set the number of printhead passes per page to 0. Otherwise, printhead passes per page determines how many times per page Output_Printhead_Pass is called.

Also, be sure to take into account the relationship between the number of lines per printhead pass with the size of the driver buffer and the driver buffer row.

D.3.8 Page_Break

The Page_Break routine sends a formfeed to the printer. The value printhead passes per page determines when this routine is executed.

For example, assume that the HP Laser Jet Plus has 10-inch pages. Then, at 75 bits per vertical inch, it has 750 printhead passes per page. After 750 passes, the Page_Break routine is called.

Note that for the laser printer, graphics mode is exited and then reentered.

D.3.9 Output_Printhead_Pass

The Output_Printhead_Pass routine converts the data in the driver buffer into a format acceptable to the printer. For example, a Toshiba printer requires data sent to it in columns rather than rows. So Output_Printhead_Pass must store enough rows to make up a column and then rotate the data. Consequently, much of what happens inside Output_Printhead_Pass is printer-specific.

For all printers, Output_Printhead_Pass should ensure that any trailing zero bytes are removed from the data. This reduces print time.

When Output_Printhead_Pass is ready to send data to the printer, it calls the routine Write.

D.3.10 Write

The Write routine sends the data to the handle. The handle was passed in via Initialize_Sheet. The Write routine checks that the written count is the same as the to-be-written count. If the number differs, Write returns the error code E_Disk_Full.

```
        jmp     Get_Driver_Buffer_Address
        jmp     Initialize_Sheet
        jmp     Completed_Sheet
        jmp     Page_Break
        jmp     Output_Printhead_Pass

Sign_On_Message  db 'HP LASERJET (75 dpi) Printer Driver  V3.00',0Dh,0Ah,'$'
COPYRIGHT_NOTICE db '(C) Copyright 1987 OrCAD Systems Corporation'
                 db '  ALL RIGHTS RESERVED.','$'

Initialize_Driver:                      ;initialize the interface
        Save_DS
; display the program title, revision and copyright notice
        mov     DX,offset(Sign_On_Message)
        mov     AH,09h
        int     21h                     ;print Sign_On_Message
        lea     DX,Copyright_Notice
        mov     AH,9
        int     21h                     ; print message
; return unused memory to the pool
        mov     AX,CS                   ;segment address of this overlay
        mov     ES,AX                   ;recall segment address
        mov     BX,offset(End_Of_File_Location)
        mov     CL,4
        shr     BX,CL
        add     BX,2
        mov     AH,4Ah                  ;modify allocated memory blocks
        int     21h
        Restore_DS
        ret

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; Get_Printer_Information
;       pass back a pointer to the Printer_Information structure in ES:BX
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
Get_Printer_Information:
        Save_DS
        mov     BX,CS
        mov     ES,BX
        mov     BX,offset(Printer_Information)
        Restore_DS
        ret

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; Get_Driver_Buffer_Address
;       pass back a pointer to the Driver_Buffer in ES:BX to be used for
;       future calls to Output_Printhead_Pass
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
Get_Driver_Buffer_Address:
        Save_DS
        mov     BX,CS
        mov     ES,BX
        mov     BX,offset(Driver_Buffer)
        Restore_DS
        ret

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; Initialize_Sheet
```

```
;       convert the Driver_Buffer into a format that the printer will use
;       remove all trailing zero bytes to reduce print and transfer time
;       return STATUS in AL
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
Output_Printhead_Pass:
        Save_DS
; reduce the transfer count to eliminate trailing zeros
        mov     CX,Driver_Buffer_Size
        mov     AX,DS
        mov     ES,AX
        mov     BX,CX
        dec     BX
        lea     DI,Driver_Buffer[BX]
        xor     AL,AL
        std
        rep     scasb
; convert the value in CX to a printer byte count
        inc     CX
        mov     Row_Count,CX
        mov     AX,CX
        xor     DX,DX
        mov     CL,10
        div     CL
        add     AH,'0'
        mov     Printer_Code[5],AH
        mov     AH,AL
        xor     AH,AH
        div     CL
        add     AH,'0'
        mov     Printer_Code[4],AH
        mov     AH,AL
        xor     AH,AH
        div     CL
        add     AH,'0'
        mov     Printer_Code[3],AH
; transfer the graphics row leadin
        mov     CX,7
        lea     DX,Printer_Code
        call    Write
        jne     OPP_Exit
; transfer the graphics row
        mov     CX,Row_Count
        lea     DX,Driver_Buffer
        call    Write
        jne     OPP_Exit
OPP_Exit:
        Restore_DS
        ret

Printer_Driver  endp

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; Write
;       transfer bytes to the HANDLE
;       pass in address of data in DS:DX
;       pass in byte count in CX
;       return STATUS in AL
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

```
;<<<NOTE>>> set the printhead passes per page to 0 if continuous paper
Driver_Buffer_Size      dw      75              ;Driver_Buffer size (in bytes)
Driver_Buffer_Row_Size  dw      75              ;Driver_Buffer size (in bytes)
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; the following must be the last declaration so that the memory size used by
; the driver can be reduced
End_Of_File_Location    db      ?

PRINTER ends
        end
```

D.5 SOURCE CODE FOR THE TOSHIBA P1350 PRINTER DRIVER

```
PAGE 64,132
NAME Printer_Driver_Toshiba
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; DECLARATIONS
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
Save_DS     MACRO
        push    DS                      ;save old DS
        push    CS
        pop     DS                      ;establish DS addressability
        assume  DS:PRINTER
            ENDM

Restore_DS  MACRO
        pop     DS                      ;restore DS
        assume  DS:nothing
            ENDM

;error codes
        E_OK                    equ  0
        E_Invalid_Function      equ  1
        E_File_Not_Found        equ  2
        E_Path_Not_Found        equ  3
        E_No_Handles_Left       equ  4
        E_Access_Denied         equ  5
        E_Invalid_Handle        equ  6
        E_Bad_Control_Block     equ  7
        E_Memory_Limit          equ  8
        E_Bad_Memory_Address    equ  9
        E_Bad_Environment       equ 10
        E_Bad_Format            equ 11
        E_Bad_Access_Code       equ 12
        E_Bad_Data              equ 13
        E_Disk_Full             equ 14
        E_Bad_Drive_Specified   equ 15
        E_Current_Dir_Remove    equ 16
        E_Not_Same_Device       equ 17
        E_No_More_Files         equ 18

STACK   segment stack   'STACK'
STACK   ends

PRINTER segment public  'PRINTER'
        assume  CS:PRINTER
        assume  DS:nothing
```

```
Get_Driver_Buffer_Address:
        Save_DS
        mov     ES,Driver_Buffer_s
        mov     BX,0
        Restore_DS
        ret

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; Initialize_Sheet
;       pass in HANDLE in BX, destination of this sheet, already opened
;       HANDLE will be checked here to ensure binary mode
;       Allocate memory for arrays
;       pass back STATUS in AL
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
Initialize_Sheet:                       ;initialize the driver for a new sheet
        Save_DS
        mov     Handle,BX               ;save handle for later routines
; check handle
        mov     AX,4400h
        int     21h                     ;get information on handle
        jc      IS_Exit                 ;if error, exit
        test    DL,80h                  ;is handle for a device?
        jz      IS_Allocate             ;if not, skip binary mode set
        or      AX,0000000000100000b    ;force binary mode
        xor     DH,DH
        mov     AX,4401h                ;set device information
        int     21h
IS_Allocate:
        mov     AX,Image_Data_s
        or      AX,AX
        jnz     IS_DB                   ;if segment <> 0 , do not reallocate
        mov     BX,Image_Data_Size
        mov     CL,4
        shr     BX,CL
        inc     BX
        mov     AH,48h
        int     21h
        jc      IS_Exit
        mov     Image_Data_s,AX         ;Image_Data array segment
IS_DB:
        mov     AX,Driver_Buffer_s
        or      AX,AX
        jnz     IS_Exit_OK              ;if segment <> 0 , do not reallocate
        mov     BX,Driver_Buffer_Size
        mov     CL,4
        shr     BX,CL
        inc     BX
        mov     AH,48h
        int     21h
        jc      IS_Exit
        mov     Driver_Buffer_s,AX      ;Driver_Buffer array segment
IS_Exit_OK:
        mov     AL,0                    ;return Status=0
IS_Exit:
        Restore_DS
        ret

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

```
        mov     CX,Image_Data_Size
        shr     CX,1
        cld
        rep     stosw                           ;clear the image data array
; convert the Driver_Buffer into the Image_Data array
; Driver_Buffer is formatted as 24 rows of 2448 bits
; Image_Data is formatted as 2448 groups of 4 bytes
        mov     Printhead_Line,0
        push    DS
        mov     ES,Image_Data_s
        mov     DS,Driver_Buffer_s

        assume  DS:nothing

OPP_Loop:
        mov     AX,CS:Driver_Buffer_Row_Size
        mov     BX,CS:Printhead_Line
        mul     BX
        mov     SI,AX                           ;offset into Driver_Buffer
        mov     AH,CS:Set_Bit[BX]               ;which bit to set for this line
        shl     BX,1                            ;convert to word index
        mov     DI,CS:Which_Byte[BX]            ;which byte to set for this line
        mov     CX,CS:Driver_Buffer_Row_Size
OPP_Line_Loop:
        lodsb                                   ;get byte from Data_Buffer
        or      AL,AL
        jz      OPP_Next_Byte                   ;if = 0 then skip bit checks
OPP_Test_80h:
        test    AL,80h
        jz      OPP_Test_40h
        or      byte ptr ES:[DI   ],AH          ;set bit in Image_Data array
OPP_Test_40h:
        test    AL,40h
        jz      OPP_Test_20h
        or      byte ptr ES:[DI+4 ],AH          ;set bit in Image_Data array
OPP_Test_20h:
        test    AL,20h
        jz      OPP_Test_10h
        or      byte ptr ES:[DI+8 ],AH          ;set bit in Image_Data array
OPP_Test_10h:
        test    AL,10h
        jz      OPP_Test_08h
        or      byte ptr ES:[DI+12],AH          ;set bit in Image_Data array
OPP_Test_08h:
        test    AL,08h
        jz      OPP_Test_04h
        or      byte ptr ES:[DI+16],AH          ;set bit in Image_Data array
OPP_Test_04h:
        test    AL,04h
        jz      OPP_Test_02h
        or      byte ptr ES:[DI+20],AH          ;set bit in Image_Data array
OPP_Test_02h:
        test    AL,02h
        jz      OPP_Test_01h
        or      byte ptr ES:[DI+24],AH          ;set bit in Image_Data array
OPP_Test_01h:
        test    AL,01h
        jz      OPP_Next_Byte
```

```
        assume  DS:nothing

        mov     DX,0
        call    Write
        pop     DS

        assume  DS:PRINTER

        jne     OPP_Exit
; transfer the graphics row end of line
        lea     DX,End_Of_Line
        mov     CX,10
        call    Write
        jne     OPP_Exit
OPP_Exit:
        Restore_DS
        ret

Printer_Driver  endp

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; Write
;       transfer bytes to the HANDLE
;       pass in address of data in DS:DX
;       pass in byte count in CX
;       return STATUS in AL
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
Write           proc near

        assume  DS:nothing

        mov     BX,CS:Handle
        jcxz    Write_Exit
IFDEF NEC
        cmp     BX,4                            ;is the handle the printer?
        je      Write_PRN                       ;if yes, jump
ENDIF
        mov     AH,40h
        push    CX                      ;save count for later check
        int     21h
        pop     CX                      ;recall count
        jc      Write_Exit
        cmp     AX,CX                   ;compare written count
        mov     AL,E_Disk_Full          ;if written count <> write count
        jne     Write_Exit              ;   then disk must be full
        mov     AL,E_OK
Write_Exit:
        ret
IFDEF NEC
Write_PRN:
        push    DS
        mov     BX,DS
        mov     ES,BX                           ;address of buffer is ES
        mov     BX,DX                           ;address of buffer is BX
        mov     AH,30h                          ;block write
        int     1Ah
        shr     AH,1
        shr     AH,1                            ;if error, carry is now set
```

```
        push    DS                      ;save old DS
        push    CS
        pop     DS                      ;establish DS addressability
        assume  DS:PRINTER
            ENDM

Restore_DS  MACRO
        pop     DS                      ;restore DS
        assume  DS:nothing
            ENDM

;error codes
        E_OK                        equ  0
        E_Invalid_Function          equ  1
        E_File_Not_Found            equ  2
        E_Path_Not_Found            equ  3
        E_No_Handles_Left           equ  4
        E_Access_Denied             equ  5
        E_Invalid_Handle            equ  6
        E_Bad_Control_Block         equ  7
        E_Memory_Limit              equ  8
        E_Bad_Memory_Address        equ  9
        E_Bad_Environment           equ 10
        E_Bad_Format                equ 11
        E_Bad_Access_Code           equ 12
        E_Bad_Data                  equ 13
        E_Disk_Full                 equ 14
        E_Bad_Drive_Specified       equ 15
        E_Current_Dir_Remove        equ 16
        E_Not_Same_Device           equ 17
        E_No_More_Files             equ 18

STACK   segment stack   'STACK'
STACK   ends

PRINTER segment public  'PRINTER'
        assume  CS:PRINTER
        assume  DS:nothing

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; Printer_Driver
;       interface to printer driver is made with a series of far calls
;       all parameters are passed via registers, the specific registers are
;       detailed for the sub-command
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
Printer_Driver  proc far
        jmp     Initialize_Driver
        jmp     Get_Printer_Information
        jmp     Get_Driver_Buffer_Address
        jmp     Initialize_Sheet
        jmp     Completed_Sheet
        jmp     Page_Break
        jmp     Output_Printhead_Pass

Sign_On_Message  db 'EPSON FX Printer Driver  V3.00',0Dh,0Ah,'$'
COPYRIGHT_NOTICE db '(C) Copyright 1987 OrCAD Systems Corporation'
                 db '  ALL RIGHTS RESERVED.','$'
```

```
        mul     CX
        mov     Row_Stagger,AX          ;row stagger offset
; check handle
        mov     AX,4400h
        int     21h                     ;get information on handle
        jc      IS_Exit                 ;if error, exit
        test    DL,80h                  ;is handle for a device?
        jz      IS_Allocate             ;if not, skip binary mode set
        or      DX,0000000000100000b    ;force binary mode
        xor     DH,DH
        mov     AX,4401h                ;set device information
        int     21h
IS_Allocate:
        mov     AX,Image_Data_s
        or      AX,AX
        jnz     IS_DB                   ;if segment <> 0 , do not reallocate
        mov     BX,Image_Data_Size
        mov     CL,4
        shr     BX,CL
        inc     BX
        mov     AH,48h
        int     21h
        jc      IS_Exit
        mov     Image_Data_s,AX         ;Image_Data array segment
IS_DB:
        mov     AX,Driver_Buffer_s
        or      AX,AX
        jnz     IS_Exit_OK              ;if segment <> 0 , do not reallocate
        mov     BX,Driver_Buffer_Size
        mov     CL,4
        shr     BX,CL
        inc     BX
        mov     AH,48h
        int     21h
        jc      IS_Exit
        mov     Driver_Buffer_s,AX      ;Driver_Buffer array segment
IS_Exit_OK:
        mov     AL,0                    ;return Status=0
IS_Exit:
        Restore_DS
        ret

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; Completed_Sheet
;       write to HANDLE at the end of the sheet printing
;       free previously allocated memory
;       pass STATUS back in AL
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
Completed_Sheet:
        Save_DS
        mov     AX,Image_Data_s         ;return Image_Data array
        or      AX,AX
        jz      CS_DB                   ;if segment value = 0 then do not free
        mov     Image_Data_s,0
        mov     ES,AX
        mov     AH,49h
        int     21h                     ;function to free memory
        jc      CS_Exit
```

```
OPP_Loop:
        mov     BX,CS:Printhead_Line
        mov     AH,CS:Set_Bit[BX]           ;which bit to set for this line
        shl     BX,1                        ;convert to word index
        mov     DI,CS:Which_Byte[BX]        ;which byte to set for this line
        mov     CX,CS:Driver_Buffer_Row_Size
OPP_Line_Loop:
        lodsb                               ;get byte from Data_Buffer
        or      AL,AL
        jz      OPP_Next_Byte               ;if = 0 then skip bit checks
OPP_Test_80h:
        test    AL,80h
        jz      OPP_Test_40h
        or      byte ptr ES:[DI   ],AH      ;set bit in Image_Data array
OPP_Test_40h:
        test    AL,40h
        jz      OPP_Test_20h
        or      byte ptr ES:[DI+Bytes_Pattern],AH
OPP_Test_20h:
        test    AL,20h
        jz      OPP_Test_10h
        or      byte ptr ES:[DI+Bytes_Pattern*2],AH
OPP_Test_10h:
        test    AL,10h
        jz      OPP_Test_08h
        or      byte ptr ES:[DI+Bytes_Pattern*3],AH
OPP_Test_08h:
        test    AL,08h
        jz      OPP_Test_04h
        or      byte ptr ES:[DI+Bytes_Pattern*4],AH
OPP_Test_04h:
        test    AL,04h
        jz      OPP_Test_02h
        or      byte ptr ES:[DI+Bytes_Pattern*5],AH
OPP_Test_02h:
        test    AL,02h
        jz      OPP_Test_01h
        or      byte ptr ES:[DI+Bytes_Pattern*6],AH
OPP_Test_01h:
        test    AL,01h
        jz      OPP_Next_Byte
        or      byte ptr ES:[DI+Bytes_Pattern*7],AH
OPP_Next_Byte:
        add     DI,Bytes_Pattern*8          ;next 8-block
        loop    OPP_Line_Loop               ;continue for all bytes on line
  start next line
        add     SI,CS:Row_Stagger
        mov     BX,CS:Printhead_Line
        inc     BX
        mov     CS:Printhead_Line,BX        ;save net printhead line number
        mov     AX,CS:Printhead_Line_Count
        mov     CL,Num_Pass                 ;divide by 3
        div     CL
        cmp     BX,AX
        jge     OPP_@@A
        jmp     OPP_Loop                    ;Loop for all printhead lines
OPP_@@A:
        pop     DS
```

```
        Restore_DS
        ret
Printer_Driver  endp

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; Write
;       transfer bytes to the HANDLE
;       pass in address of data in DS:DX
;       pass in byte count in CX
;       return STATUS in AL
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
Write           proc near

        assume  DS:nothing
        mov     BX,CS:Handle
        jcxz    Write_Exit

IFDEF NEC
        cmp     BX,4                            ;is the handle the printer?
        je      Write_PRN                       ;if yes, jump
ENDIF

        mov     AH,40h
        push    CX                              ;save count for later check
        int     21h
        pop     CX                              ;recall count
        jc      Write_Exit
        cmp     AX,CX                           ;compare written count
        mov     AL,E_OK
        je      Write_Exit
        mov     AL,E_Disk_Full                  ;if written count <> write count
        stc
Write_Exit:
        ret

IFDEF NEC
Write_PRN:
        push    DS
        mov     BX,DS
        mov     ES,BX                           ;address of buffer is ES
        mov     BX,DX                           ;address of buffer is BX
        mov     AH,30h                          ;block write
        int     1Ah
        shr     AH,1
        shr     AH,1                            ;if error, carry is now set
        pop     DS
        mov     AL,E_Access_Denied
        jc      Write_Exit
        mov     AL,E_OK
        jmp     Write_Exit
ENDIF

Write           endp

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; DATA AREA
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
Handle                  dw      0
```

HOW TO WRITE A PLOTTER DRIVER

OrCAD provides a number of plotter drivers for use with OrCAD products. If you have a plotter not supported by an OrCAD driver, you may decide to write your own driver. This appendix describes how to do that.

The driver shown here is the generic plotter driver. It is supplied by OrCAD, and its filename is GENERIC.DRV. The generic plotter driver transfers a null-terminated ACSII string to the plotter in place of plotter commands. For example, in place of the command to draw a line, the driver sends the text LINE(. When you write your own plotter driver, you would replace that ASCII string with the appropriate plotter command.

A plotter driver is usually written in assembly language. It must obey certain conventions as described in this appendix.

D.1 PSEUDO CODE SUMMARY OF A PLOTTER DRIVER

A plotter driver consists of one far procedure called Plotter_Driver. The routines within the driver are accessed via jmp instructions. The order of the jmp instructions is very important.

For example, the first jmp is to the routine Initialize_Driver. When the driver is loaded, the first routine in the jmp table is the one that gets executed. Consequently, this routine must be Initialize_Driver.

Here is a pseudo code summary of the generic plotter driver. If your plotter doesn't use a particular routine, you must still provide the routine. In this case, you

Plotter_Driver proc far
List of jmp instructions to the various routines

Initialize_Driver:
Performs any needed initialization. This routine is executed when the driver is loaded. It may contain commands specific to the plotter.

Initialize_Device:
Checks the handle. If the handle is zero, the driver initializes the serial channel. Otherwise, it uses the handle.

Begin_Plot:
The generic plotter outputs the ASCII string BEGIN_PLOT(). A specific driver should output any commands appropriate to the beginning of the sheet.

End_Plot:
The generic plotter outputs the ASCII string END_PLOT(). A specific driver should output any commands appropriate to the end of the sheet.

Set_Velocity:
The generic plotter outputs the ASCII string VELOCITY(). A specific driver should output any commands necessary to change the pen's velocity.

Set_Acceleration:
The generic plotter outputs the ASCII string ACCELERATION(). A specific driver should output any commands necessary to change the pen acceleration.

Set_Force:
The generic plotter outputs the ASCII string FORCE(). A specific driver should output any commands necessary to change the pen force.

one where the pen remains down. For example, the lines making up a box belong to a poly vector stream.

Poly_End:
The generic plotter outputs the ASCII string POLY_END(). The routine marks the end of a poly vector stream.

Output Functions
These are internal routines used by routines in the jmp table, but they don't themselves appear in the jump table.

Data Area
Reserves space for variables and the driver buffer. Also defines variables specific to the plotter. This area also contains the variable End_Of_File_Location. The routine Initialize_Driver uses this variable's location to determine the size of the driver so that it can return unused memory to the system.

PLOTTER ends
end

D.2 ASSEMBLY AND LINKAGE

Plotter drivers are intended to be assembled with the Microsoft Macro Assembler and linked with Microsoft Link.

For example, consider the generic plotter driver. Assume that its source code resides in a file called GENERIC.ASM. To assemble GENERIC.ASM, issue the following command (on one line):

GENERIC.DRV: GENERIC.ASM

For more information about assembling and linking assembly language files, refer to the *Microsoft Macro Assembler User's Guide.*

```
BP              ends

PL          struc                   ;Plot_Line parameters structure
PL_Xs       dw      ?
PL_Ys       dw      ?
PL_Xe       dw      ?
PL_Ye       dw      ?
PL          ends

PE          struc                   ;Plot_Ellipse parameters structure
PE_Xc       dw      ?
PE_Yc       dw      ?
PE_Xr       dw      ?
PE_Yr       dw      ?
PE          ends

PC          struc                   ;Plot_Circle parameters structure
PC_X        dw      ?
PC_Y        dw      ?
PC_Radius   dw      ?
PC          ends

PA          struc                   ;Plot_Arc parameters structure
PA_Xc       dw      ?
PA_Yc       dw      ?
PA_Xe1      dw      ?
PA_Ye1      dw      ?
PA_Xe2      dw      ?
PA_Ye2      dw      ?
PA_Radius   dw      ?
PA          ends

PDL         struc                   ;Plot_Dashed_Line parameters structure
PDL_Xs      dw      ?
PDL_Ys      dw      ?
PDL_Xe      dw      ?
PDL_Ye      dw      ?
PDL         ends

PT          struc                   ;Plot_Text parameter structure
PT_X        dw      ?
PT_Y        dw      ?
PT_Height   dw      ?
PT_String_p dd      ?
PT          ends

STACK   segment stack  'STACK'
STACK   ends

PLOTTER segment public 'PLOTTER'
        assume  CS:PLOTTER
        assume  DS:nothing

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; Plotter_Driver
;       interface to Plotter driver is made with a series of far calls
;       all parameters are passed via registers, the specific registers are
;       detailed for the sub-command
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;   Initialize_Device:
;       initialize the device
;       if the handle is <> 0 then write only to the handle
;       if the handle = 0 then initialize the serial channel
;       pass in via ES:BX a pointer to the PARAMETER structure
;       return STATUS in AL
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
Initialize_Device:
        Save_DS
        xor     AX,AX
        mov     Buffer_Count,AX         ;clear Buffer_Count
        mov     Last_Pen,AX             ;clear Last_Pen
        mov     Last_Velocity,AX        ;clear Last_Velocity
        mov     AX,ES:[BX].P_Handle
        mov     Handle,AX
        or      AX,AX
        jz      ID_Serial
        jmp     ID_Handle
ID_Serial:
        mov     AX,ES:[BX].P_Channel
        dec     AX                      ;convert the channel from 1..2 to 0..1
        mov     Channel,AX
        xor     AL,AL
        mov     CX,ES:[BX].P_Baud_Rate  ;set up AL to reflect the baud rate
        cmp     CX,110
        je      Found_Baud_Rate
        add     AL,00100000b
        cmp     CX,150
        je      Found_Baud_Rate
        add     AL,00100000b
        cmp     CX,300
        je      Found_Baud_Rate
        add     AL,00100000b
        cmp     CX,600
        je      Found_Baud_Rate
        add     AL,00100000b
        cmp     CX,1200
        je      Found_Baud_Rate
        add     AL,00100000b
        cmp     CX,2400
        je      Found_Baud_Rate
        add     AL,00100000b
        cmp     CX,4800
        je      Found_Baud_Rate
        add     AL,00100000b
        cmp     CX,9600
        jne     ID_Error_Exit           ;if not found, exit without changing port
Found_Baud_Rate:
        mov     CL,ES:[BX].P_Parity     ;set up AL to reflect the parity
        cmp     CL,0
        je      Found_Parity
        add     AL,00001000b
        cmp     CL,1
        je      Found_Parity
        add     AL,00010000b
        cmp     CL,2
        jne     ID_Error_Exit           ;if not found, exit without changing port
```

```
;       called at the end of the plot of the current sheet
;       return STATUS in AL
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
End_Plot:
        Save_DS
        mov     SI,offset(ENDPLOT)
        call    Put_Output
        jc      CP_Exit
        call    Flush_Buffer            ;since this is the end of the sheet
CP_Exit:
        Restore_DS
        ret

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;   Set_Velocity:
;       set the plotter current pen to the velocity passed in BX
;       return STATUS in AL
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
Set_Velocity:
        Save_DS
        cmp     BX,Last_Velocity        ;new velocity = last velocity ?
        je      Velocity_Unchanged      ;yes, no change neccessary
        mov     Last_Velocity,BX        ;no, save new velocity
        mov     SI,offset(SETVELOCITY)
        call    Put_Output
        jc      SV_Error_Exit
        mov     AX,Last_Velocity
        call    Put_Word
        jc      SV_Error_Exit
        call    Put_TERM
Velocity_Unchanged:
SV_Error_Exit:
        Restore_DS
        ret

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;   Set_Acceleration:
;       set the plotter current pen to the Acceleration passed in BX
;       return STATUS in AL
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
Set_Acceleration:
        ret

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;   Set_Force:
;       set the plotter current pen to the Force passed in BX
;       return STATUS in AL
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
Set_Force:
        ret

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;   Set_Pen:
;       set the plotter current pen to the Pen passed in BX
;       return STATUS in AL
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
Set_Pen:
        Save_DS
```

```
PL_Error_Exit:
        Restore_DS
        ret

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;   Plot_Circle:
;       plot a Circle
;       X,Y,Radius are pointed to by a pointer passed in ES:BX
;       return STATUS in AL
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
Plot_Circle:
        Save_DS
        mov     word ptr Current_Pointer[0],BX
        mov     word ptr Current_Pointer[2],ES
        call    Put_PENUP               ;raise pen up to center (X,Y)
        jc      PL_Error_Exit
        les     BX,Current_Pointer
        mov     CX,ES:[BX].PC_X
        mov     DX,ES:[BX].PC_Y
        mov     Current_X,CX            ;update current (X,Y)
        mov     Current_Y,DX
        call    Put_XY
        jc      PL_Error_Exit
        call    Put_TERM
        jc      PL_Error_Exit
        mov     SI,offset(CIRCLE)       ;issue a circle command
        call    Put_Output
        jc      PC_Error_Exit
        les     BX,Current_Pointer
        mov     AX,ES:[BX].PC_Radius    ;specify the radius
        call    Make_Plotter_Units
        call    Put_Word
        jc      PC_Error_Exit
        les     BX,Current_Pointer
        mov     AX,ES:[BX].PC_Radius    ;specify chord angle based on radius
        call    Put_CHORD_ANGLE
        jc      PC_Error_Exit
        call    Put_TERM
        jc      PC_Error_Exit
        call    Put_PENUP               ;raise the pen up
        jc      PC_Error_Exit
        call    Put_TERM
PC_Error_Exit:
        Restore_DS
        ret

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;   Plot_Ellipse:
;       plot a Ellipse
;       Xc,Yc,Xr,Yr are pointed to by a pointer passed in ES:BX
;       return STATUS in AL
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
Plot_Ellipse:
        ret

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;   Plot_Arc:
;       plot a Arc
```

```
        les     BX,Current_Pointer
        mov     CX,ES:[BX].PA_Xe2
        mov     DX,ES:[BX].PA_Ye2
        mov     Current_X,CX            ;update current (X,Y)
        mov     Current_Y,DX
PA_Error_Exit:
        Restore_DS
        ret

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;   Plot_Dashed_Line:
;       plot a Dashed_Line
;       Xs,Ys,Xe,Ye are pointed to by a pointer passed in ES:BX
;       return STATUS in AL
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
Plot_Dashed_Line:
        Save_DS
        mov     word ptr Current_Pointer[0],BX
        mov     word ptr Current_Pointer[2],ES
        mov     SI,offset(SETDASHLINE)  ;specify dashline pattern
        call    Put_Output
        jc      PDL_Error_Exit
        call    Put_PENUP               ;raise pen up to (Xs,Ys)
        jc      PDL_Error_Exit
        les     BX, Current_Pointer
        mov     CX,ES:[BX].PDL_Xs
        mov     DX,ES:[BX].PDL_Ys
        call    Put_XY
        jc      PDL_Error_Exit
        call    Put_TERM
        jc      PDL_Error_Exit
        call    Put_PENDOWN             ;let pen down and trace a dashed line
        jc      PDL_Error_Exit
        les     BX, Current_Pointer
        mov     CX,ES:[BX].PDL_Xe
        mov     DX,ES:[BX].PDL_Ye
        mov     Current_X,CX            ;update current (X,Y)
        mov     Current_Y,DX
        call    Put_XY
        jc      PDL_Error_Exit
        call    Put_TERM
        jc      PDL_Error_Exit
        mov     SI,offset(RESETDASHLINE);set back to solid line
        call    Put_Output
        jc      PDL_Error_Exit
        call    Put_TERM
        jc      PDL_Error_Exit
        call    Put_PENUP               ;raise pen up after all
        jc      PDL_Error_Exit
        call    Put_TERM
PDL_Error_Exit:
        Restore_DS
        ret

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;   Plot_Text:
;       plot a Text string
;       String_p,X,Y,Height are pointed to by a pointer passed in ES:BX
```

```
        push    DS
        lds     SI,ES:[BX].PT_String_p  ;DS:SI point to string
        pop     ES
        mov     DI,offset(Text)         ;make ES:DI point to text buffer
        lodsb                           ;get string count
        xor     AH,AH                   ;convert to word
        mov     CX,AX                   ;use as move count
        cld
        rep     movsb
        mov     AL,03h
        stosb                           ;insert a ETX character
        xor     AL,AL
        stosb                           ;make last byte a null
        pop     DS                      ;restore DS
        mov     SI,offset(Text)
        call    Put_Output              ;transfer line
        jc      PT_Error_Exit
        call    Put_PENUP               ;raise pen up after all
        jc      PT_Error_Exit
        call    Put_TERM
PT_Error_Exit:
        Restore_DS
        ret

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;   Poly_Begin:
;       mark the beginning of a poly vector stream
;       pass back STATUS in AL
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
Poly_Begin:
        Save_DS
        mov     Poly_Flag,1             ;set Poly_Flag on
        Restore_DS
        mov     AL,E_OK
        ret

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;   Poly_End:
;       mark the end of a poly vector stream
;       pass back STATUS in AL
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
Poly_End:
        Save_DS
        mov     Poly_Flag,0             ;reset Poly_Flag
        call    Put_PENUP               ;raise the pen up
        jc      PEnd_Error_Exit
        call    Put_TERM
PEnd_Error_Exit:
        Restore_DS
        ret

Plotter_Driver  endp

;<><><><><><><><><><><><><><><<< OUTPUT FUNCTIONS >>><><><><><><><><><><><><><><>

        assume  DS:PLOTTER

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

```
        int     21h                     ;write
        jc      Flush_Buffer_Exit       ;if error, exit now
;check if the write count matches the written count
        cmp     AX,Buffer_Count
        mov     AL,E_OK
        je      Flush_Buffer_Exit       ;if counts match, then E_OK
        mov     AL,E_Disk_Full          ;else the disk is full
        stc                             ;set the carry flag to indicate error
Flush_Buffer_Exit:
        mov     Buffer_Count,0          ;reset Buffer_Count
        ret
Flush_Buffer    endp

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;   Put_Fixed:
;       enter with AX holding a fixed place word
;       convert the fixed place word to an ASCII text stream->  XX.XXX
;       send the text stream to the output
;       return the STATUS in AL with the carry set if <> E_OK
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
Put_Fixed   proc near
;clear ASCII_FIXED
        push    AX                      ;save AX
        cld
        mov     DI,offset ASCII_FIXED
        mov     CX,7
        xor     AX,AX
        rep     stosb
        pop     AX                      ;recall AX
        xor     BX,BX
        cmp     AX,9999
        jle     PF_Thousands
        xor     DX,DX
        mov     CX,10000
        div     CX
        xchg    AX,DX
        add     DL,'0'
        mov     ASCII_FIXED[BX],DL
        inc     BX
PF_Thousands:
        xor     DX,DX
        mov     CX,1000
        div     CX
        xchg    AX,DX
        add     DL,'0'
        mov     ASCII_FIXED[BX],DL
        inc     BX
        mov     ASCII_FIXED[BX],'.'
        inc     BX
        xor     DX,DX
        mov     CX,100
        div     CX
        xchg    AX,DX
        add     DL,'0'
        mov     ASCII_FIXED[BX],DL
        inc     BX
        or      AX,AX
        jz      PF_Null
```

```
        mov     ASCII_Word[BX],DL
        inc     BX
PW_Hundreds:
        xor     DX,DX
        mov     CX,100
        div     CX
        xchg    AX,DX
        add     DL,'0'
        mov     ASCII_Word[BX],DL
        inc     BX
PW_Tens:
        xor     DX,DX
        mov     CX,10
        div     CX
        xchg    AX,DX
        add     DL,'0'
        mov     ASCII_Word[BX],DL
        inc     BX
PW_Units:
        add     AL,'0'
        mov     ASCII_Word[BX],AL
        inc     BX
        mov     ASCII_Word[BX],0
        mov     SI,offset(ASCII_Word)
        test    Negative_Flag,1
        jz      PW_Exit
        mov     Negative_Flag,0
        dec     SI
PW_Exit:
        call    Put_Output
        ret
Put_Word    endp

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;   Put_XY:
;       enter with CX holding X value, DX holding Y value
;       convert to ASCII text streams and send to the output device
;       return the STATUS in AL with carry set if STATUS <> E_OK
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
Put_XY  proc near
        push    DX                      ;save Y in stack
        mov     AX,CX
        call    Make_Plotter_Units
        add     AX,X_Offset             ;offset space of space plot
        call    Put_Word
        jc      Put_XY_Exit
        call    Put_COMMA
        jc      Put_XY_Exit
        pop     AX                      ;recall Y
        call    Make_Plotter_Units
        add     AX,Y_Offset             ;offset space of space plot
        call    Put_Word
Put_XY_Exit:
        ret
Put_XY   endp

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;   Make_Plotter_Units:
```

```
;   Put_CHORD_ANGLE:
;       enter with the radius in AX
;       send the chord angle portion of the command to the output
;       return the STATUS in AL, if <> E_OK then set the carry flag
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
Put_CHORD_ANGLE     proc near
        mov     SI,offset(CHORD_ANGLE)  ;issue the chord angle for a circle
        cmp     AX,100                  ;30 degree max
        jbe     PCA
        add     SI,4                    ;next chord angle
        cmp     AX,250                  ;15 degree max
        jbe     PCA
        add     SI,4                    ;next chord angle
        cmp     AX,500                  ;10 degree max
        jbe     PCA
        add     SI,4                    ;next chord angle
        cmp     AX,1000                 ; 5 degree max
        jbe     PCA
        add     SI,4                    ;next chord angle
        cmp     AX,2000                 ; 2 degree max
        jbe     PCA
        add     SI,4                    ;use 0 degree chord angle
PCA:
        call    Put_Output
        ret
Put_CHORD_ANGLE     endp

;<><><><><><><><><><><><><><><><<< GENERAL FUNCTIONS >>><><><><><><><><><><><><><><>

        assume  DS:PLOTTER

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;   Compare_XY:
;       enter with CX holding X value, DX holding Y value
;       compare X,Y with the current X,Y
;       return with AL=1 if equal, else AL=0
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
Compare_XY   proc near
        mov     AL,1
        cmp     CX,Current_X
        jnz     Compare_XY_Exit
        cmp     DX,Current_Y
        jnz     Compare_XY_Exit
        inc     ax
Compare_XY_Exit:
        dec     ax
        ret
Compare_XY   endp

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;   Find_Arc_Angle:
;       calculate the arc's angle and its direction the plot's pen should move
;       from (Xe1,Ye1) to (Xe2,Ye2)
;       Xc,Yc,Xe1,Ye1,Xe2,Ye2,Radius are referenced the pointer passed in ES:BX
;       return with an angle: negative angle specifies clockwise movement
;                             positive angle specifies counter clockwise movement
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
Find_Arc_Angle proc near
```

```
Calculate4:
        mov     AX,SI                   ;S_Angle - E_Angle
        sub     AX,DI
        ret
Calculate5:
        mov     AX,SI                   ;S_Angle + E_Angle
        add     AX,DI
        ret
Calculate6:
        mov     AX,SI                   ; - (S_Angle + E_Angle)
        add     AX,DI
        neg     AX
        ret
Find_Arc_Angle   endp

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;    Find_Angle:
;       given X-Xo,Y-Yo, find an angle of the line with respect to X-axis
;       enter with SI holding X-distance, DI holding Y-distance
;       return angle in AX
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
Find_Angle  proc near
        or      DI,DI                   ;Y_distance
        jns     Y_Check
        neg     DI
Y_Check:
        or      SI,SI                   ;X_distance
        jns     Compare
        neg     SI
Compare:
        xor     CX,CX
        cmp     SI,DI
        jge     Calculate_Tan           ;X_distance >= Y_distance
        mov     CX,1                    ;cx=1 signals a cotan calculation
        xchg    SI,DI
Calculate_Tan:
        xor     DX,DX
        mov     AX,10000
        mul     DI
        div     SI                      ;tan = Y/X
        xor     BX,BX                   ;index bx is also angle
Lookup_Tan:
        cmp     AX,Tan_Table[BX]
        je      Check_Tan_Cot
        jl      Take_Previous
        add     BX,2
        jmp     Lookup_Tan
Take_Previous:
        sub     BX,2                    ;previous angle
Check_Tan_Cot:
        shr     BX,1                    ;angle in bx
        or      CX,CX
        jz      FA_Exit
        mov     AX,90                   ;angle=90-angle in cotan calculation
        sub     AX,BX
FA_Exit:
        ret
Find_Angle   endp
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
Quadrant_Table          db      4,0,2
                        db      3,0,3
                        db      1,1,2
Calculate_Table         dw      Calculate1,Calculate2,0,Calculate6
                        dw      Calculate3,Calculate4,Calculate5,0
                        dw      0,Calculate6,Calculate1,Calculate2
                        dw      Calculate5,0,Calculate3,Calculate4
Tan_Table               dw      0000,0175,0349,0524,0699,0875,1051,1228,1405,1584
                        dw      1763,1944,2126,2309,2493,2679,2867,3057,3249,3443
                        dw      3640,3839,4040,4245,4452,4663,4877,5095,5317,5543
                        dw      5774,6009,6249,6494,6745,7002,7265,7536,7813,8098
                        dw      8391,8693,9004,9325,9657,10000
Last_Velocity           dw      0
Last_Acceleration       dw      0
Last_Force              dw      0
Last_Pen                dw      0
ASCII_Fixed             db      '00.000',0
Negative_Flag           dw      0
Negative_Sign           db      '-'
ASCII_Word              db      '00000',0
Character               db      ?
Text                    db      256  dup (?)
Buffer                  db      1024 dup (?)
Buffer_Count            dw      0
Buffer_Size             dw      1024
Poly_Flag               dw      ?
X_Offset                dw      ?
Y_Offset                dw      ?
Current_X               dw      ?
Current_Y               dw      ?
Current_Pointer         dd      ?
Channel                 dw      ?
Handle                  dw      ?

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; the following must be the last declaration so that the memory size used by
; the driver can be reduced
End_Of_File_Location    db      ?

PLOTTER ends
        end
```

D.4 SOURCE CODE FOR GENERIC PLOTTER DRIVER

```
PAGE 64,132
NAME Generic_Plotter_Driver
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; DECLARATIONS
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
Save_DS     MACRO
        push    DS                      ;save old DS
        push    CS
        pop     DS                      ;establish DS addressability
        assume  DS:PLOTTER
```

```
PE_Yr         dw      ?
PE            ends

PC            struc
PC_X          dw      ?
PC_Y          dw      ?
PC_Radius     dw      ?
PC            ends

PA            struc
PA_Xc         dw      ?
PA_Yc         dw      ?
PA_Xe1        dw      ?
PA_Ye1        dw      ?
PA_Xe2        dw      ?
PA_Ye2        dw      ?
PA_Radius     dw      ?
PA            ends

PDL           struc
PDL_Xs        dw      ?
PDL_Ys        dw      ?
PDL_Xe        dw      ?
PDL_Ye        dw      ?
PDL           ends

PT            struc
PT_X          dw      ?
PT_Y          dw      ?
PT_Height     dw      ?
PT_String_p   dd      ?
PT            ends

STACK   segment stack   'STACK'
STACK   ends

PLOTTER segment public 'PLOTTER'
        assume  CS:PLOTTER
        assume  DS:nothing

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; Plotter_Driver
;       interface to Plotter driver is made with a series of far calls
;       all parameters are passed via registers, the specific registers are
;       detailed for the sub-command
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
Plotter_Driver  proc FAR
        jmp     Initialize_Driver
        jmp     Initialize_Device
        jmp     Begin_Plot
        jmp     End_Plot
        jmp     Set_Velocity
        jmp     Set_Acceleration
        jmp     Set_Force
        jmp     Set_Pen
        jmp     Plot_Line
        jmp     Plot_Circle
        jmp     Plot_Ellipse
```

```
        mov     AX,ES:[BX].P_Handle
        mov     Handle,AX
        or      AX,AX
        jz      ID_Serial
        jmp     ID_Handle
ID_Serial:
        mov     AX,ES:[BX].P_Channel
        dec     AX                      ;convert the channel from 1..2 to 0..1
        mov     Channel,AX
        xor     AL,AL
        mov     CX,ES:[BX].P_Baud_Rate
        cmp     CX,110
        je      Found_Baud_Rate
        add     AL,00100000b
        cmp     CX,150
        je      Found_Baud_Rate
        add     AL,00100000b
        cmp     CX,300
        je      Found_Baud_Rate
        add     AL,00100000b
        cmp     CX,600
        je      Found_Baud_Rate
        add     AL,00100000b
        cmp     CX,1200
        je      Found_Baud_Rate
        add     AL,00100000b
        cmp     CX,2400
        je      Found_Baud_Rate
        add     AL,00100000b
        cmp     CX,4800
        je      Found_Baud_Rate
        add     AL,00100000b
        cmp     CX,9600
        jne     ID_Error_Exit           ;if not found, exit without changing port
Found_Baud_Rate:
        mov     CL,ES:[BX].P_Parity
        cmp     CL,0
        je      Found_Parity
        add     AL,00001000b
        cmp     CL,1
        je      Found_Parity
        add     AL,00010000b
        cmp     CL,2
        jne     ID_Error_Exit           ;if not found, exit without changing port
Found_Parity:
        mov     CL,ES:[BX].P_Stop_Bits
        cmp     CL,1
        je      Found_Stop_Bits
        add     AL,00000100b
        cmp     CL,2
        jne     ID_Error_Exit           ;if not found, exit without changing port
Found_Stop_Bits:
        add     AL,00000010b
        mov     CL,ES:[BX].P_Word_Length
        cmp     CL,7
        je      Found_Word_Length
        add     AL,00000001b
        cmp     CL,8
```

```
        call    put_Fixed
        jc      BP_Error_Exit
        call    put_TERM
BP_Error_Exit
        Restore_DS
        ret

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;   End_Plot:
;       called at the end of the plot of the current sheet
;       return STATUS in AL
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
End_Plot:
        Save_DS
        mov     SI,offset(EP_String)
        call    Put_Output
        jc      CP_Exit
        call    Flush_Buffer        ;since this is the end of the sheet
CP_Exit:
        Restore_DS
        ret

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;   Set_Velocity:
;       set the plotter current pen to the velocity passed in BX
;       return STATUS in AL
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
Set_Velocity:
        Save_DS
        cmp     BX,Last_Velocity    ;new velocity = last velocity ?
        je      Velocity_Unchanged  ;yes, no change neccessary
        mov     Last_Velocity,BX    ;no, save new velocity
        mov     SI,offset(SV_String)
        call    Put_Output
        jc      SV_Error_Exit
        mov     AX,Last_Velocity
        call    Put_Word
        jc      SV_Error_Exit
        call    Put_TERM
Velocity_Unchanged:
SV_Error_Exit:
        Restore_DS
        ret

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;   Set_Acceleration:
;       set the plotter current pen to the Acceleration passed in BX
;       return STATUS in AL
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
Set_Acceleration:
        Save_DS
        cmp     BX,Last_Acceleration    ;new Acceleration = last Acceleration ?
        je      Acceleration_Unchanged  ;yes, no change neccessary
        mov     Last_Acceleration,BX    ;no, save new Acceleration
        mov     SI,offset(SA_String)
        call    Put_Output
        jc      SA_Error_Exit
        mov     AX,Last_Acceleration
```

```
Plot_Line:
        Save_DS
        mov     word ptr Current_Pointer[0],BX
        mov     word ptr Current_Pointer[2],ES
        mov     SI,offset(PL_String)
        call    Put_Output
        jc      PL_Error_Exit
        les     BX,Current_Pointer
        mov     AX,ES:[BX].PL_Xs
        call    Put_Fixed
        jc      PL_Error_Exit
        call    Put_COMMA
        jc      PL_Error_Exit
        les     BX,Current_Pointer
        mov     AX,ES:[BX].PL_Ys
        call    Put_Fixed
        jc      PL_Error_Exit
        call    Put_COMMA
        jc      PL_Error_Exit
        les     BX,Current_Pointer
        mov     AX,ES:[BX].PL_Xe
        call    Put_Fixed
        jc      PL_Error_Exit
        call    Put_COMMA
        jc      PL_Error_Exit
        les     BX,Current_Pointer
        mov     AX,ES:[BX].PL_Ye
        call    Put_Fixed
        jc      PL_Error_Exit
        call    Put_TERM
PL_Error_Exit:
        Restore_DS
        ret

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;   Plot_Circle:
;       plot a Circle
;       X,Y,Radius are pointed to by a pointer passed in ES:BX
;       return STATUS in AL
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

Plot_Circle:
        Save_DS
        mov     word ptr Current_Pointer[0],BX
        mov     word ptr Current_Pointer[2],ES
        mov     SI,offset(PC_String)
        call    Put_Output
        jc      PC_Error_Exit
        les     BX,Current_Pointer
        mov     AX,ES:[BX].PC_X
        call    Put_Fixed
        jc      PC_Error_Exit
        call    Put_COMMA
        jc      PC_Error_Exit
        les     BX,Current_Pointer
        mov     AX,ES:[BX].PC_Y
        call    Put_Fixed
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

Plot_Arc:
        Save_DS
        mov     word ptr Current_Pointer[0],BX
        mov     word ptr Current_Pointer[2],ES
        mov     SI,offset(PA_String)
        call    Put_Output
        jc      PA_Error_Exit
        les     BX,Current_Pointer
        mov     AX,ES:[BX].PA_Xc
        call    Put_Fixed
        jc      PA_Error_Exit
        call    Put_COMMA
        jc      PA_Error_Exit
        les     BX,Current_Pointer
        mov     AX,ES:[BX].PA_Yc
        call    Put_Fixed
        jc      PA_Error_Exit
        call    Put_COMMA
        jc      PA_Error_Exit
        les     BX,Current_Pointer
        mov     AX,ES:[BX].PA_Xe1
        call    Put_Fixed
        jc      PA_Error_Exit
        call    Put_COMMA
        jc      PA_Error_Exit
        les     BX,Current_Pointer
        mov     AX,ES:[BX].PA_Ye1
        call    Put_Fixed
        jc      PA_Error_Exit
        call    Put_Fixed
        jc      PA_Error_Exit
        les     BX,Current_Pointer
        mov     AX,ES:[BX].PA_Xe2
        call    Put_Fixed
        jc      PA_Error_Exit
        call    Put_COMMA
        jc      PA_Error_Exit
        les     BX,Current_Pointer
        mov     AX,ES:[BX].PA_Ye2
        call    Put_Fixed
        jc      PA_Error_Exit
        call    Put_Fixed
        jc      PA_Error_Exit
        les     BX,Current_Pointer
        mov     AX,ES:[BX].PA_Radius
        call    Put_Fixed
        jc      PA_Error_Exit
        call    Put_TERM
PA_Error_Exit:
        Restore_DS
        ret

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;   Plot_Dashed_Line:
;       plot a Dashed_Line
;       Xs,Ys,Xe,Ye are pointed to by a pointer passed in ES:BX
```

```
        call    Put_COMMA
        jc      PT_Error_Exit
        les     BX,Current_Pointer
        mov     AX,ES:[BX].PT_Y
        call    Put_Fixed
        jc      PT_Error_Exit
        call    Put_COMMA
        jc      PT_Error_Exit
        les     BX,Current_Pointer
        mov     AX,ES:[BX].PT_Height
        call    Put_Fixed
        jc      PT_Error_Exit
        call    Put_COMMA
        jc      PT_Error_Exit
        call    Put_Quote                   ;open string
        jc      PT_Error_Exit
;copy the string from the pointer to the text buffer and convert to null-end
        les     BX,Current_Pointer
        push    DS                          ;save DS
        mov     DI,offset(Text)             ;make ES:DI point to text buffer
        push    DS
        lds     SI,ES:[BX].PT_String_p
        pop     ES
        lodsb                               ;get string count
        xor     AH,AH                       ;convert to word
        mov     CX,AX                       ;use as move count
        cld
        rep     movsb
        xor     AL,AL
        stosb                               ;make last byte a null
        pop     DS                          ;restore DS
        mov     SI,offset(Text)
        call    Put_Output                    ;transfer line
        jc      PT_Error_Exit
        call    Put_Quote                   ;close string
        jc      PT_Error_Exit
        call    Put_TERM
PT_Error_Exit:
        Restore_DS
        ret

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;   Poly_Begin:
;       mark the beginning of a poly vector stream
;       pass back STATUS in AL
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
Poly_Begin:
        Save_DS
        mov     Poly_Flag,0FFh
        mov     SI,offset(P_B_String)
        call    Put_Output
        Restore_DS
        ret

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;   Poly_End:
;       mark the end of a poly vector stream
;       pass back STATUS in AL
```

```
        mov     Buffer[BX],AL           ;save the character in the buffer
        inc     BX
        mov     Buffer_Count,BX         ;save the new last loaded point
        jmp     Get_Character           ;get next character
Put_Output endp

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;   Flush_Buffer:
;       send the entire buffer to the handle
;       return the STATUS in AL with carry set if STATUS <> E_OK
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
Flush_Buffer    proc near
        mov     BX,Handle               ;destination of write
        mov     CX,Buffer_Count         ;how much to write
        mov     DX,offset(Buffer)       ;source of write
        mov     AH,40h
        int     21h                     ;write
        jc      Flush_Buffer_Exit       ;if error, exit now
;check if the write count matches the written count
        cmp     AX,Buffer_Count
        mov     AL,E_OK
        je      Flush_Buffer_Exit       ;if counts match, then E_OK
        mov     AL,E_Disk_Full          ;else the disk is full
        stc                             ;set the carry flag to indicate error
Flush_Buffer_Exit:
        mov     Buffer_Count,0          ;reset Buffer_Count
        ret
Flush_Buffer    endp

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;   Put_COMMA:
;       sent a COMMA to the ouptut device
;       return the STATUS in AL, if <> E_OK then set the carry flag
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
Put_COMMA   proc near
        mov     SI,offset(COMMA)
        call    Put_Output
        ret
Put_COMMA   endp

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;   Put_QUOTE:
;       sent a QUOTE to the ouptut device
;       return the STATUS in AL, if <> E_OK then set the carry flag
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
Put_QUOTE   proc near
        mov     SI,offset(QUOTE)
        call    Put_Output
        ret
Put_QUOTE   endp

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;   Put_TERM:
;       sent a right parenthesis and CrLf to the ouptut device
;       return the STATUS in AL, if <> E_OK then set the carry flag
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
Put_TERM    proc near
        mov     SI,offset(TERM)
```

```
Put_Fixed   endp

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;   Put_Word:
;       enter with AX holding a Word
;       convert the word to an ASCII text stream->  XXXXX
;       send the text stream to the output
;       return the STATUS in AL with the carry set if <> E_OK
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
Put_Word    proc near
        xor     BX,BX
        cmp     AX,9
        jle     PW_Units
        cmp     AX,99
        jle     PW_Tens
        cmp     AX,999
        jle     PW_Hundreds
        cmp     AX,9999
        jle     PW_Thousands
        xor     DX,DX
        mov     CX,10000
        div     CX
        xchg    AX,DX
        add     DL,'0'
        mov     ASCII_Word[BX],DL
        inc     BX
PW_Thousands:
        xor     DX,DX
        mov     CX,1000
        div     CX
        xchg    AX,DX
        add     DL,'0'
        mov     ASCII_Word[BX],DL
        inc     BX
PW_Hundreds:
        xor     DX,DX
        mov     CX,100
        div     CX
        xchg    AX,DX
        add     DL,'0'
        mov     ASCII_Word[BX],DL
        inc     BX
PW_Tens:
        xor     DX,DX
        mov     CX,10
        div     CX
        xchg    AX,DX
        add     DL,'0'
        mov     ASCII_Word[BX],DL
        inc     BX
PW_Units:
        add     AL,'0'
        mov     ASCII_Word[BX],AL
        inc     BX
        mov     ASCII_Word[BX],0
        mov     SI,offset(ASCII_Word)
        call    Put_Output
        ret
```

3. What errors, if any in the manual, need to be corrected?

Page #	Comments
_____	______________________________
_____	______________________________
_____	______________________________
_____	______________________________
_____	______________________________
_____	______________________________
_____	______________________________

4. Please list, or attach a sheet, describing new features that you would like to add to this product.______________________________

Return to:

OrCAD
TECHNICAL SUPPORT DIVISION
3175 NW Aloclek Drive
Hillsboro, OR 97124-7135

Place
Stamp
Here

Return to:

OrCAD
Product Registration
3175 NW Aloclek Drive
Hillsboro, OR 97124-7135

5. Please describe in detail, or enclose an ASCII text file on disk, describing the problem? ____

____

____

____

____

____

____

____

____

____

6. To assist us in our analysis, please check and enclosed a disk with any additional information:

☐ DISK ENCLOSED ☐ ASCII (Problem description) ENCLOSED

☐ SOURCE, OBJECT LIBRARY NAMES____________________

____

☐ Netlist filenames____________________________

☐ Additional files______________________________

____

TO BE COMPLETED BY OrCAD

PRF #______________________ENGINEER__________________________________

Send form and related disks to:

OrCAD
TECHNICAL SUPPORT DIVISION
3175 NW Aloclek Drive
Hillsboro, OR 97124-7135

OrCAD™
Systems Corporation