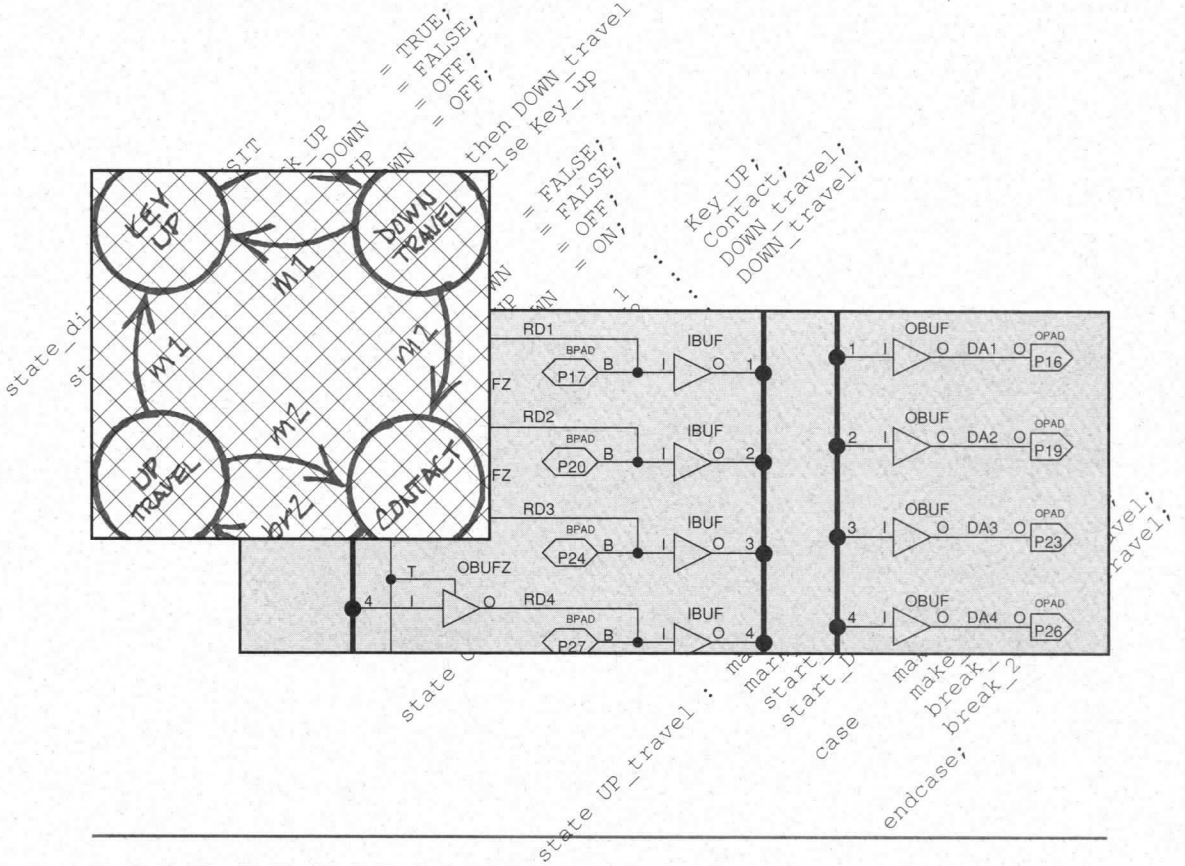
FutureNet®

SCHEMATIC DESIGNER



POST PROCESSING TOOLS

DATA I/O Corporation

981-0249

FutureNet®

Schematic Designer

Post User Manual

DATA I/O Corporation

096-0096-002

August 1991 096-0096-002

Data I/O Corporation
10525 Willows Road N.E., P.O. Box 97046
Redmond, Washington 98073-9746 USA
(206) 881-6444

Acknowledgments:

Table of Contents

5. Compatible Pinlist Generator (PINC)

6. Enhanced Netlist Generator (NET4)

7. Compatible Netlist Generator (NETC)

8. Parts List Generator (PARTS)

9. Messages

Glossary

Index

List of Figures

List of Tables

Preface

The Preface contains details about telephone support, warranty service, the Bulletin Board Service, typographic conventions and more.

Customer Support Offices

United States

For technical assistance, contact

Data I/O Customer Resource Center
Telephone: 800 247-5700
Fax: 206 882-1043

For warranty service, contact your nearest Data I/O Service Center below:

Data I/O Corporate Office
10525 Willows Road N.E.
P.O. Box 97046
Redmond, WA 98073-9746
Telephone: 206 881-6444
Fax: 206 882-1043
Telex: 152167

Data I/O California
1701 Fox Drive
San Jose, CA 95131
Telephone: 408 437-9600
Fax: 408 437-1218

Data I/O Northeastern United States
20 Cotton Road
Nashua, NH 03063
Telephone: 603 889-8511
800 858-5803 (NJ & NY only)
Fax: 603 880-0697

Canada

For technical assistance, contact:

Data I/O Customer Resource Center
Telephone: 800 247-5700
Fax: 206 882-1043

For warranty service, contact:

Data I/O Canada
6725 Airport Road, Suite 302
Mississauga, Ontario, L4V 1V2
Telephone: 416 678-0761
Fax: 416 678-7306

United Kingdom

For technical assistance or warranty service, contact:

Data I/O Limited
660 Eskdale Road
Winnersh, Wokingham
Berkshire RG11 5TS
Telephone: 0734 440011
Fax: 0734 448700

Japan

For technical assistance or warranty service, contact:

Data I/O Japan
Sumitomoseimei Higashishinbashi Bldg. 8F
2-1-7, Higashi-Shinbashi
Minato-Ku, Tokyo 105
Telephone: 03 3432-6991
Fax: 03 3432-6094 (Sales)
03 3432-6093 (Other)
Telex: 2522685 DATAIO J

Germany

For technical assistance or warranty service, contact:

Data I/O Electronic Systems Vertriebs GmbH
Lochhammer Schlag 5a
D-8032 Gräfelfing
Telephone: 089 858580
Fax: 089 8585810

Other European Countries

For technical assistance or warranty service, contact the office below and ask for the number of your local Data I/O representative:

Data I/O Europe
World Trade Center
Strawinskylaan 537
1077 XX Amsterdam, The Netherlands
Telephone: +31 (0)20 6622866
Fax: +31 (0)20 6624427

Other Countries Worldwide

For technical assistance or warranty service, contact the office below and ask for the number of your local Data I/O representative:

Data I/O Intercontinental
10525 Willows Road N.E.
P.O. Box 97046
Redmond, WA USA 98073-9746
Telephone: 206 881-6444
Fax: 206 882-1043
Telex: 4740166

Technical Assistance

Calling

To help us provide quick and accurate assistance, please be at your programmer or computer when you call, and have the following ready:

- Product version number
- Product serial number (if available)
- Detailed description of the problem you are experiencing
- Error messages (if any)
- Device manufacturer and part number (if device-related)
- Product manual

For technical assistance, contact the appropriate Customer Support office listed at the front of the Preface.

Electronic Mail

You can also reach Data I/O via electronic mail (e-mail). To help us provide quick and accurate assistance, please include the information listed above. Also, include your name, phone number, and e-mail address in your message, and send it to one of the following addresses:

```
techhelp@Data-IO.COM
```

or

```
{apple|decwrl|rutgers|gatech|uunet}!pilchuck!techhelp
```

Note: Select one of the five addresses listed above in braces. For example, you might send e-mail to the following address: `uunet!pilchuck!techhelp`.

See your system administrator if you need more information on which address to use.

Warranty Information

Data I/O Corporation warrants this product against defects in materials and workmanship at the time of delivery and thereafter for a period of ninety (90) days.

The foregoing warranty and the manufacturers' warranties, if any, are in lieu of all other warranties, expressed, implied or arising under law, including, but not limitied to, the implied warranties of merchantability and fitness for a particular purpose.

Data I/O maintains customer service offices throughout the world, each staffed with factory-trained technicians to provide prompt, quality service. For warranty service, contact the appropriate Customer Support office listed at the front of the Preface.

Bulletin Board Service

From the Data I/O Bulletin Board System (BBS) you can obtain a wide range of information on Data I/O products, including current product descriptions, new revision information, technical support information, helpful application notes, and other miscellaneous information. In addition, the BBS has a large collection of useful DOS utilities you can download.

The BBS message facility allows you to leave messages for the BBS system operator, Customer Support personnel, or other customers. Functions are available to provide device support information or to request support for a particular device.

To learn more about the U.S. Data I/O BBS, call 1-206-882-3211. Multiple lines are available, all supporting 1200/2400/9600/19200 baud with U.S. Robotics Courier HST modems with V.42bis compatibility, set to 8 data bits, 1 stop bit and no parity. Call 1-206-861-6959 to contact the BBS connected to a U.S. Robotics Dual/HST modem supporting 1200/2400/9600/19200 baud with V.32bis/V.42bis compatibility. Online help files provide more information about the BBS and its capabilities.

For your nearest Data I/O Bulletin Board System outside the U.S., contact the appropriate Customer Support office listed at the front of the Preface.

End User Registration and Address Change

If the end user for this product or your address has changed since the Registration Card was mailed, please notify your nearest Customer Support office as listed at the front of the Preface. This ensures that you receive information about product enhancements. Be sure to include the product serial number, if available.

Typographic Conventions

Throughout this manual different typographic conventions represent different cases of input and output.

Keyboard Keys

Keyboard keys may be shown in boxes (for example, [Q]) or as bolded text.

The **Enter** key (or on some keyboards, the **Return** key) is represented by this symbol: [↵].

Key Combinations

Key combinations, such as **Control-Z**, are shown as two key boxes separated by a dash; for example, [Ctrl] – [Z].

A key combination like [Esc] [Ctrl] – [T] means to press and release [Esc], then press [Ctrl] and [T] at the same time.

Variable Input

Variable input is italicized and should be replaced with the requested information. For example, enter **copy** ***filename.hex*** means type **copy** just as you see it and replace ***filename.hex*** with the name of your file.

Optional Input

Optional items of a command are shown in brackets; for example

[*option1*] [*option2*]...[*optionn*]

Items separated by a vertical bar (for example, OR | OR | ...) are mutually exclusive; that is, only one of the options listed can be specified.

Displayed Messages

Text that appears on the screen will be displayed in a typewriter-like typeface; for example,

```
You will see this text displayed on the screen.
```

1 Post Overview

The FutureNet Post package processes drawing files produced by the FutureNet Schematic Designer (referred to as FutureNet in this manual). The processing performed by Post software allows a variety of CAE tools to use FutureNet drawing files. The Post package consists of the following tools:

- Drawing Preprocessor
- Design Rule Check
- Pinlist Generators
- Netlist Generators
- Parts List Generator
- EDIF 2 0 0 Netlist Writer

Each of these tools is outlined below, and discussed in detail in its own chapter. (The Netlist Writer is described in a separate manual.)

Post Tools

Drawing Preprocessor

The Drawing Preprocessor combines a set of FutureNet drawing files into a single design database: the Drawing Connectivity Model (DCM). First, local connection data is determined for each FutureNet drawing and appended to the drawing file. Then this local connection data is used to create a global model of the design. The resulting DCM file is a netlist/pinlist data structure, which post processors use to selectively report information about the design.

Design Rule Check

Design Rule Check uses the DCM file to perform basic wiring checks for problems such as missing or duplicated reference designators, unconnected pins, and signals that have only one connection. A report is generated for each type of check requested. The reports aid in checking the schematic drawings for correctness before proceeding further with the design.

Design Rule Check performs some checks against basic electrical wiring rules, but does not perform any functional verification of the design.

Pinlist Generator

The Pinlist Generator uses the DCM file to produce a pinlist file in ASCII format that lists the symbols for each drawing, the pins on the symbols, and the signals to which the pins are connected. This symbol-oriented information can be used by other programs, such as simulators.

Two different pinlist programs are included with Post: an enhanced version that takes advantage of the latest features of FutureNet, including property sheet generation using layered text; and a version that provides backward compatibility by generating pinlists similar to those generated by older versions of FutureNet, which are used by some existing post processing tools.

Netlist Writer

The Netlist Writer uses the DCM file to produce a netlist file in ASCII format that lists the names of the signals in the design and the pins to which they are connected. This signal-oriented information can be used by other programs, such as printed circuit board layout and wire wrapping tools.

As with the Pinlist Generator, two separate programs are provided with the Netlist Generator, offering enhanced and backwardly-compatible netlists.

Parts List Generator

The Parts List Generator uses the DCM file to produce a summary of all the parts used in a design. The listing gives you information on the quantity of each unique part needed to implement the design.

EDIF 2 0 0 Netlist Writer

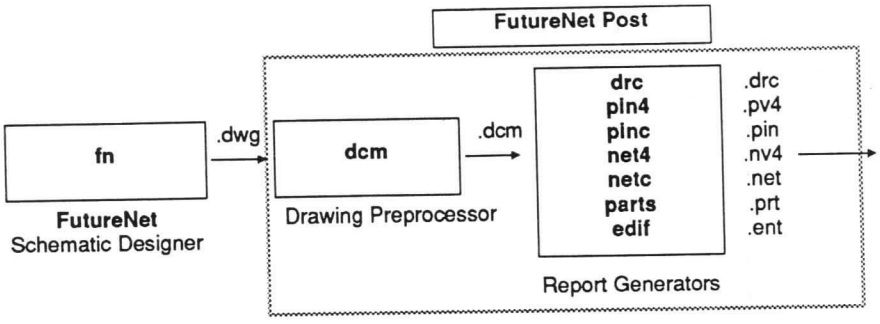
The EDIF 2 0 0 Netlist Writer uses the DCM file to produce a standard EDIF 2 0 0 netlist. This Netlist Writer takes full advantage of the latest features of FutureNet, including property sheet generation using layered text. This netlist can be used as input to other computer-aided engineering and design tools.

Using Post

Figure 1-1 shows the relationship of FutureNet to Post. It identifies the process, the name used to invoke the process, and the default file extension assigned to the various output files.

The Post programs can be run from the operating system command line or in FutureNet from the **Export** menus. This manual explains the functions and operation of each program; for information on basic operation for the command line and menus, see the *FutureNet User Manual*. For more information on the Export menus, see the next section, "Export Menu."

Figure 1–1
Post Process Overview



Export Menu

The **Export** menu (Figure 1–2) is accessed by entering **EXPORT** on the FutureNet command line, or from the Command menu by selecting **EXPORT Generate Reports**. You can run each Post program either as a stand-alone menu selection or as part of a consecutive processing operation with the Export Manager.

Figure 1–2
Export Menu

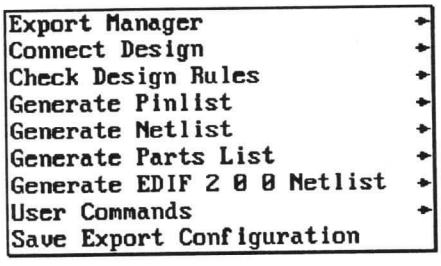


Table 1–1 summarizes the **Export** submenu selections. The submenu selections **Export Manager, User Commands,** and **Save Export Configuration** are covered in this chapter. The other submenu selections are covered in the chapters for that program.

Table 1–1
EXPORT Generate Reports Menu

Submenu Selection	Function	Program Name	Chapter
Export Manager	Specify processing order for programs and run consecutively	N/A	current
Connect Design	Create drawing connectivity model	dcm	"The Drawing Preprocessor"
Check Design Rules	Check drawing against design rules	drc	"Design Rule Check"
Generate Pinlist	Generate a pinlist in enhanced or compatible format	pin4 pinc	"Enhanced Pinlist Generator" "Compatible Pinlist Generatorr"
Generate Netlist	Generate a netlist in enhanced or compatible format	net4 netc	"Enhanced Netlist Generator" "Compatible Netlist Generator"
Generate Parts List	Generate a parts list	parts	"Parts List Generator"

Submenu Selection	Function	Program Name	Chapter
Generate EDIF 2 0 0Netlist	Generate a netlist in EDIF 2 0 0 format	edif	*FutureNet EDIF 2 0 0 Netlist Writer User Manual*
User Commands	Define your own programs, specify processing order for programs, and run consecutively	N/A	current
Save Export Configuration	Save state of Export menus as default setup	N/A	current

Export Manager

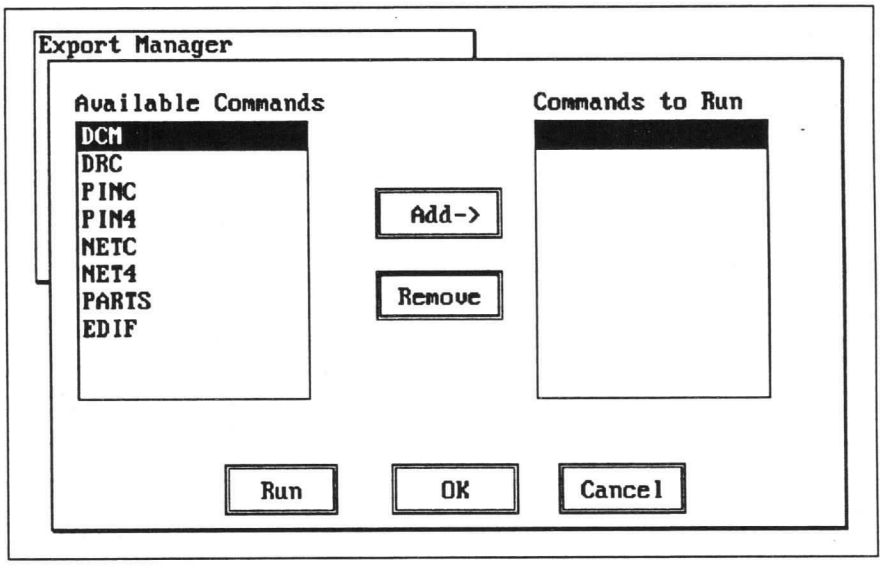
The Export Manager allows one-step processing of FutureNet drawings, from creation of the drawing connectivity model to generation of the Post reports you select.

In either case, you must choose the options for the individual Post programs you want to run using the option selection dialog box for that program. To run several programs consecutively, do **not** select the program button (for example, DCM); instead, select OK to return to the main menu, and select options for the next program. Once all options have been entered, select **Export Manager**.

Export Manager Dialog Box

When you select **Export Manager**, the dialog box shown in Figure 1-3 appears.

Figure 1–3
Export Manager Dialog Box



The dialog box contains two data boxes. The **Available Commands** data box lists the post processing programs and any user commands defined in the User Commands dialog box. The **Commands to Run** data box lists the programs selected for consecutive processing.

Adding Programs

Add the Post programs you want processed consecutively to the **Commands to Run** data box by selecting them from the **Available Commands** data box as follows:

1. Highlight the program name in the **Available Commands** data box.
2. Select **Add->**.

The selected program name is entered above the highlight bar in the **Commands to Run** data box, and is removed from the Available Commands data box.

Removing Programs

Remove items from the **Commands to Run** list as follows:

1. Highlight the program name in the **Commands to Run** data box
2. Select **Remove**.

The selected program name is removed from the **Commands to Run** list and restored to the **Available Commands** list.

Running the Consecutive Process

Select the **Run** button in the **Export Manager** dialog box to run the commands in the **Commands to Run** data box. The commands will run in the order given by the Commands to Run data box.

On the Sun, the processes will run in the window that was used to run FutureNet. Output from the processes will be put into that window. For this reason, it is recommended that FutureNet be run from a Command Tool window to ease review of the output from the processes.

On the PC, the screen will clear as FutureNet is temporarily closed and a DOS shell is opened. Output of the commands will appear on the screen. After the commands have completed, or if an error is detected in one of the processes, the message

```
Press any key to continue
```

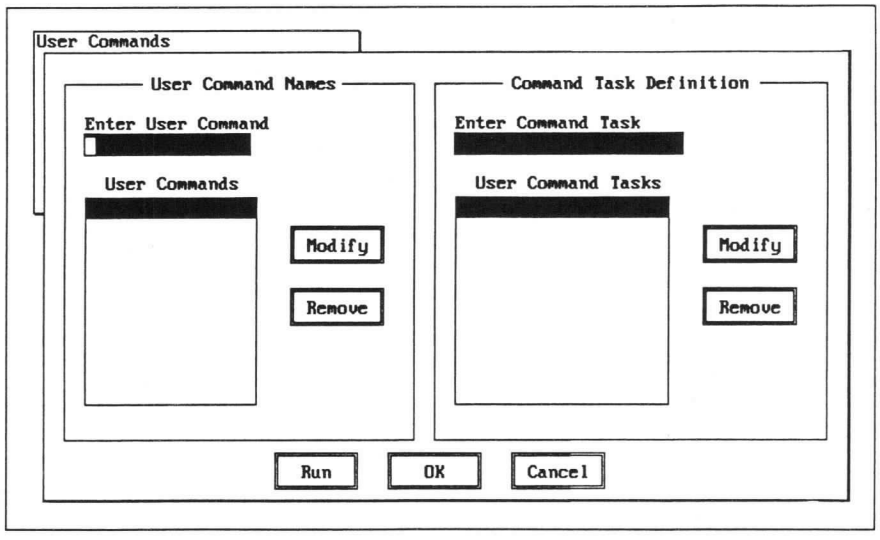
is displayed. Press any key to return to FutureNet.

Note: Avoid using the Ctrl-C or Ctrl-Break key sequences. These may cause FutureNet to fail, with possible loss of data.

Output files from the commands that were run can be reviewed by doing the following.

1. Use the DOS command to open a DOS shell.
2. Use any normal DOS tools to look at the files.

Figure 1–4
User Commands Dialog Box



User Commands

The **User Commands** dialog box allows you to create your own programs to add to the **Available Commands** list in the Export Manager dialog box. When you select **User Commands**, the dialog box shown in Figure 1-4 appears.

The dialog box contains two sections: the **User Command Names** section lists the command names, and the **Command Task Definition** section defines the tasks required to accomplish the commands. The names can be any unique name that you want. The tasks must be valid command line operations for your operating system. Any number of tasks may be specified for a particular user command. The tasks do not need to be unique. When a command is selected in the **User Commands** data box, the corresponding tasks will appear in the **User Command Tasks** data box.

The procedures for entering, modifying and removing command names and tasks are the same.

Entering Command Names and Tasks

Enter a command name or task as follows:

1. Highlight (select) an entry in the data box (if previous entries exist). The entry that you enter (in step 4 below) will appear directly above this highlighted entry; if no entry is highlighted, the entry will appear at the end of the list.
2. Select the **Enter User Command** or **Enter Command Task** entry field.
3. Type in the desired command name or task.
4. Press ⏎ to enter the text into the **User Commands** or **User Command Tasks** data box.

Modifying Command Names and Tasks

Modify a command name or task as follows:

1. Highlight the command name or task.
2. Select **Modify**. The selected command name or task is copied to the corresponding entry field.
3. Edit the command. Refer to the *FutureNet User Manual* for instructions on editing in entry fields.
4. Press ↵, and the command name or task is corrected in the data box.

Removing Command Names or Tasks

Remove command names or tasks as follows:

1. Highlight the program name in the data box.
2. Select **Remove**.

Note: When a command name is removed, all tasks that are associated with that name are also removed.

Running a User Command

Run a command in the **User Commands** dialog box as follows:

1. Select a command in the **User Command** data box.
2. Select **RUN** to run the highlighted command.

After processing for the command is complete, you are returned to the FutureNet workspace. Exit FutureNet to examine the generated reports.

Note: If you select **OK** *to accept changes made in the* ***User Commands*** *dialog box, the changes will appear in the* ***Export Manager*** *dialog box.*

From the ***Export Manager*** *dialog box, commands can also be run.*

Save Export Configuration

The settings of the Export menu can be saved for future use by using the Save Export Configuration command from the Export menu. This command will save all settings in the Export menu to a configuration file, **fn.cfg**. If the file was detected when FutureNet is initialized, then that file will be updated; otherwise, the file will be created in the current directory.

When FutureNet is initiated, it searches for the **fn.cfg** file in the following directories, in the following order:

1. Current directory
2. Directory from which FutureNet was run from
3. The directories on the system path—using the PATH environment variable.

2 *Drawing Preprocessor (DCM)*

The Drawing Preprocessor reads the top-level drawings associated with a design and builds a drawing connectivity model (DCM) of the design. Lower-level drawing files can be incorporated into the DCM. The DCM is saved as a file (.dcm) for use by post processors.

DCM processing begins by searching for the drawing set specified by the root and lower-level drawing files. Each drawing file is checked to ensure that current connectivity data is appended to the end of the file. If a drawing has been edited since the Drawing Preprocessor was last run, new connectivity data is generated for the drawing and appended to the end of the file.

Processing continues by tracing all wires and buses in the drawings and resolving their implicit and explicit connections.* A connectivity model is built with this information, and the results are written to a file with a .dcm extension.

The drawing connectivity model can only be generated from drawings in the DASH-4 (any version) or later file format. Refer to the appendix of the *FutureNet User Manual* that discusses conversion of DASH-2 or DASH-3 drawings to the DASH-4 or later format.

Running the Drawing Preprocessor

The Drawing Preprocessor can be run from the Export menu in FutureNet, or from the operating system command line. Basic menu and command line operation is explained in the *FutureNet User Manual*.

The Drawing Preprocessor uses FutureNet drawings as input files, with the default drawing extension .dwg. You do not need to specify extensions for input files with .dwg extensions (for example, an input file named test.dwg can be specified as test). Extensions different from the default FutureNet drawing extension must be specified.

* *The conventions used to determine connection are described in the chapter titled "Understanding FutureNet" in the* FutureNet User Manual.

The output file containing the drawing connectivity model defaults to the name of the first root drawing file with a .dcm extension.

Command Line

The Drawing Preprocessor is started using the following command:

dcm ***input_filenames options***

An input filename is required. If no input filename is given, the program prompts for information.

In addition to the options described here for the Drawing Preprocessor, there are some options that are common to FutureNet and the post processors. These options are described in detail, along with information on placing command line options in command files, in the *FutureNet User Manual.*

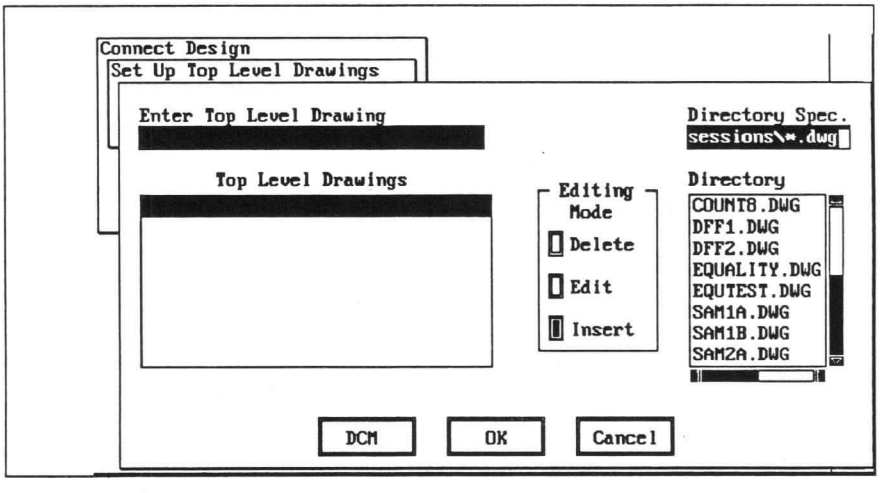
Menus

The Drawing Preprocessor is run from FutureNet by using the Export menus. Enter EXPORT on the FutureNet command line or select **EXPORT Generate Reports** from the command menu, then select **Connect Design.**

This section covers how to use the dialog boxes in the Connect Design menu to set up options for the Drawing Preprocessor. Basic menu operations (for example, editing in entry fields, or selecting menus and check boxes) are discussed in the *FutureNet User Manual.*

The following discussions refer to Figure 2-1. Figure 2-1 is an example of a **Connect Design** dialog box; other dialog boxes in the Connect Design menu will operate similarly.

Figure 2–1
Typical Dialog Box



Entering List Items

In the **Connect Design** dialog box, use the **Enter** entry field (shown as **Enter Top Level Drawing** in the typical **Connect Design** dialog box) to enter an item into the **List** data box beneath it (shown as **Top Level Drawings**) as follows:

1. Select the **Insert** action button in the **Editing Mode** box.
2. Highlight (select) an item in the **List** data box (if previous entries exist). The item that you enter (in step 5 on the next page) will appear directly above this highlighted item; if no item is highlighted, the item will appear at the end of the list.

3. Select the **Enter** entry field with the mouse.
4. Type in the data needed.
5. Press ↵ to enter the data into the data box.

Note: You can also enter drawing files using the **Directory** *data box as described in the following section.*

Selecting Items from the Directory Data Box

You can enter a filename from the **Directory** data box into the **List** data box as follows:

1. Select the **Insert** action button in the **Editing Mode** box.
2. Highlight (select) an item in the **List** data box (if previous entries exist). The item that you select (in step 3 below) will appear directly above this highlighted item; if no item is highlighted, the item will appear at the end of the list.
3. Select (click on) the desired item in the **Directory** data box. The item is copied to the **List** data box.

Editing List Items

Edit a list item as follows:

1. Select the **Edit** action button in the **Editing Mode** box.
2. Select (click on) the item to be edited in the **List** data box. The highlighted item is copied to the **Enter** entry field.
3. Edit the item (see the *FutureNet User Manual* for instructions on editing in entry fields).
4. Press ↵ to enter the corrections in the data box.

Deleting List Items

Delete items from the **List** data box as follows:

1. Select the **Delete** action button in the **Editing Mode** box.
2. Select (click on) the item to be deleted in the **List** data box. The highlighted item is deleted.

Note: After deleting items, immediately select a different action button (either **Edit** *or* **Insert***) from the* **Editing Mode** *box. Leaving the* **Delete** *action button highlighted may cause unwanted deletions.*

Specifying a Directory and Filename Filter

The **Directory Spec.** entry field allows you to enter a directory and file specification that filters the information in the **Directory** data box beneath it. (You can also enter files from the **Directory** data box into the **List** data box. See earlier in this chapter.)

1. Select the **Directory Spec.** entry field with the mouse.
2. Type in a directory and file specification (wildcards are allowed).
3. Press ↵ to filter the data into the data box.

Option Summary

Table 2-1 summarizes the Drawing Preprocessor options. Each option is discussed further in the paragraphs that follow.

Table 2–1
Drawing Preprocessor Options

Task	Connect Design Menu Selection	Command Line
Include top-level drawing file	Set Up Top-level Drawings	-i*filename*
Include lower-level drawing file	Set Up Lower-level Drawings	-l*filename*
Search directory for drawings	Set Up Design Directories	-p*directory*
Suppress functional block warnings	Set Up Options	-t
Suppress all messages	Set Up Options	-w
Include all lower-level drawing files	Set Up Lower-level Drawings	-x
Include all except specified lower-level drawing files	N/A	-x*filename*

Drawing Preprocessor Options

The options specific to the Drawing Preprocessor are discussed below. The menu selection is given on the left; and the command line equivalent is on the right, with the option being discussed in bold text.

Specify Directories to Search

Set Up Design Directories

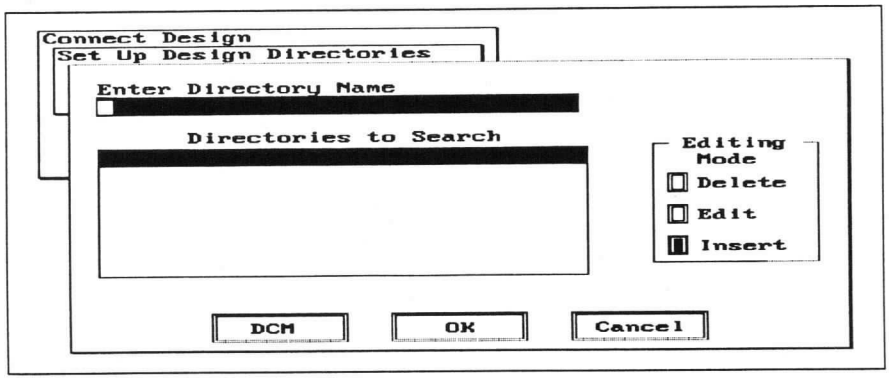
dcm -i*filename(s)* **-p*path+...***

Use this option to specify a directory search path for drawing files. These directories are searched if a drawing file is not in the current directory. Directories are searched in the order specified. The default search path is the current directory.

Note: On the command line, specify all directories before specifying the drawing files. If a large number of directories and drawings are involved, you may want to use a command file as described in the FutureNet User Manual.

The **Set Up Design Directories** dialog box is shown in Figure 2-2.

Figure 2–2
Set Up Design Directories Dialog Box



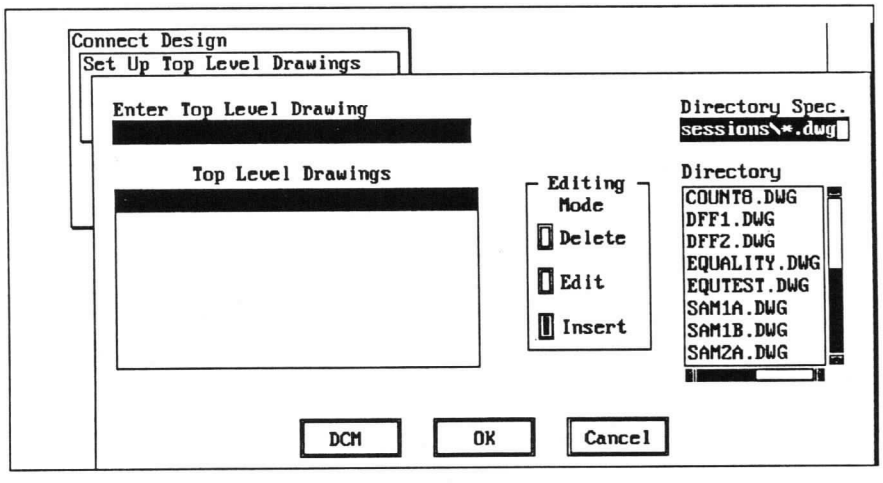
Include and Exclude Drawing Files

Set Up Top Level Drawings

dcm -i*filename+...*

Use this option to specify the top-level drawing files to include in the design. At least one top-level drawing file must be specified, and drawing files are included sequentially. The default filename extension is .dwg if none is specified. The **Set Up Top Level Drawings** dialog box is shown in Figure 2-3.

Figure 2–3
Set Up Top Level Drawings Dialog Box



Set Up Lower Level Drawings

dcm -i*filename(s)* **-l***filename+...*

Use this option to include specific lower-level drawings that are referenced, using functional blocks, by a top-level drawing. The **Set Up Lower Level Drawings** dialog box is shown in Figure 2-4 on the next page.

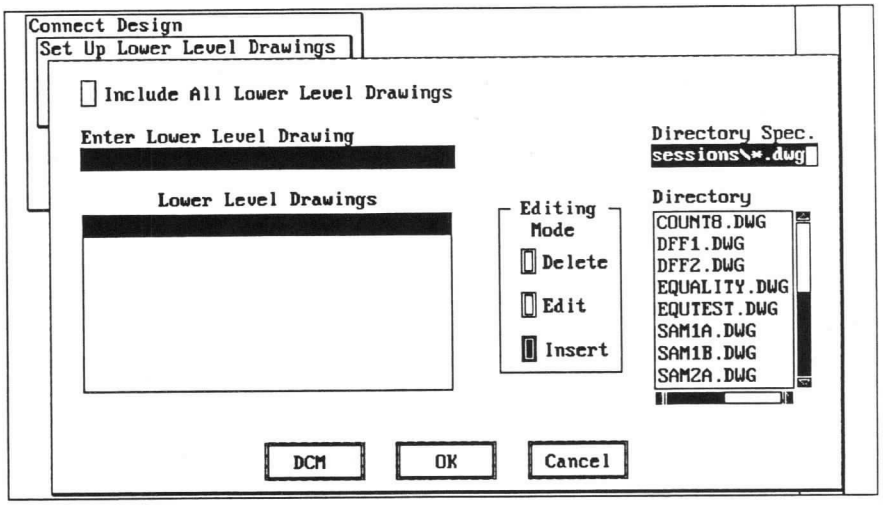
Set Up Lower Level Drawings

☐ **Include All Lower Level Drawings**

dcm -i*filename(s)* **-x[***filename+...***]**

Use this option to include all lower-level drawings that are referenced, using functional blocks, by a top-level drawing.

Figure 2–4
Set Up Lower Level Drawings Dialog Box



To include all lower-level drawing files in the design, check the **Include All Lower Level Drawings** box, or use -x with no filenames. To include all but specified files, use -x*filename*.

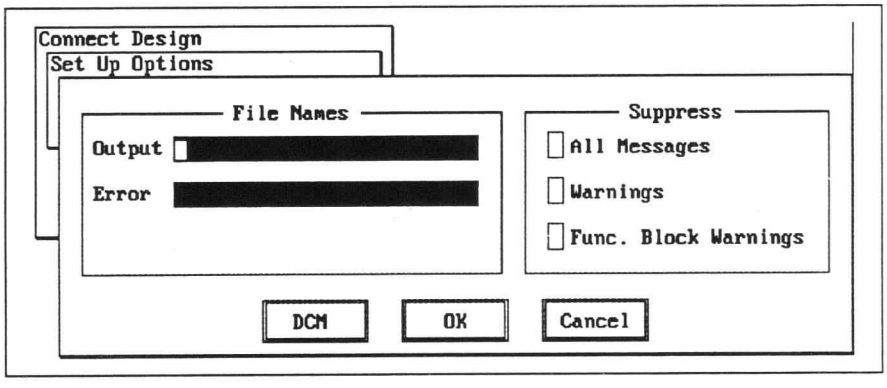
Note: If it is easier to list the files you want to include, use the -lfilename option.

Set Up Drawing Preprocessor Options

Set Up Options

Drawing Preprocessor options are selected from the **Set Up Options** dialog box shown in Figure 2-5.

Figure 2–5
Set Up Options Dialog Box



Specify Filenames

Output ▮__

dcm -i*filename(s)* **-o*filename***

Use this option to specify a name for the output file. The default filename is the first top-level drawing filename with a .dcm extension.

Error ▮__

dcm -i*filename(s)* **-e*filename***

Use this option to specify a name for the error file. Error messages are generated for all top- and lower-level drawing files specified. The default filename extension is .der.

Suppress Status Messages

☐ **Suppress All Messages**

dcm -i*filename(s)* **-s**

Use this option to omit all messages from the display and error file. Use +s to reset the option so that all messages will be sent to the display and error file. (This might be useful for overriding the command file settings.)

☐ **Suppress Warnings**

dcm -i*filename(s)* **-w**

Use this option to omit all warning messages from the display and error file. Use +w to reset the option so that all warning messages are sent to the display and error file.

☐ **Suppress Func. Block Warnings**

dcm -i*filename(s)* **-t**

Use this option to suppress warning messages about functional blocks that cannot be found or that are present but unconnected. Use +t to reset the option so that functional blocks that cannot be found or that are unconnected are flagged as errors.

Run Options

DCM — Selecting DCM runs the Drawing Preprocessor with the options selected.

Ok — Selecting OK saves the entered options and exits the dialog box.

Cancel — Selecting Cancel discards any changes and exits the dialog box.

Running the Local Subprocess

Occasionally, an error message which indicates that **local** must be run separately from the normal Drawing Preprocessor execution may be encountered. **local** generates a connectivity model for a specific drawing and appends this information to the drawing file. This local connectivity model is then combined with others to create the DCM file. This section gives information on how to run local.

Note: It is not necessary to run **local** *separately unless you receive an error message that indicates you should run* **local**.

If you receive an error message that indicates you should run **local** separately, start local from the command line by typing

local *filename*

The following options are valid with local. See the *FutureNet User Manual* for complete descriptions.

-i	Input file
-e	Error file (default extension is .der)
-s	Silent mode
-q	Query mode

After processing the indicated files with local, rerun the Drawing Preprocessor using the revised input files generated by local.

Design Structure and Name Assignment

Attributes for Controlling Structured Connections

The Drawing Preprocessor uses displayed text with FILE and FILN attributes to establish structured design connections. These connections are described in detail in the chapter titled "Understanding FutureNet" in the *FutureNet User Manual.*

FILE and FILN are used to identify functional block drawing files used in the design.

Assigning Signal and Bus Names

A name is associated with every net in the design. The net name consists of a bus name (if the signal is associated with a bus), a signal name, and a drawing reference number. Explicit names are obtained from user-specified text fields in the drawing. Names are internally generated for any unnamed wires and buses.

Bus names are assigned according to the conventions specified in the chapter titled "Understanding FutureNet" in the *FutureNet User Manual.* Non-bused signals, packaged signals, and universal signals are not associated with a bus.

The drawing reference number assigned to a net depends on the signal type. In general, signals are associated with the first drawing in the design tree in which they occur. Universal signals are not associated with a drawing.

User-specified Names

User-specified names will be used, if provided, overriding the internal generation of names. Explicit and implicit user-specified signal names are propagated through every line segment in the wiring net. In a hierarchical design, signal names from higher level drawings replace lower-level functional block signal names.

Internally Generated Signal and Bus Names

Internally generated names are supplied for all unnamed wires and buses.

Internally generated names have the form

****xxxyyyrrr***

or

****xxxxyyyyrrr***

where *** is used to indicate an internally generated name, *xxx* and *yyy* (0 - 999) and *xxxx* and *yyyy* (1000 - 9999) are the x,y coordinates for a unique point on the unnamed line segment, and *rrr* is the drawing reference number. The value of an internally generated name is constant, provided the wiring net is not modified.

Features

Partial Link

A partial link is the omission of one or more functional blocks from a design. A partial link occurs automatically whenever a drawing file in a functional block drawing set cannot be found.

A partial link can be generated using the - l*filename* and -x*filename* options. This can be useful for simulating part of a circuit or where functional blocks describe elements that will not be expanded for circuit board layout, such as PLDs or FPGAs.

3 Design Rule Check (DRC)

The Design Rule Check (DRC) program checks a FutureNet design for compliance with certain electrical design rules. The program generates a report that contains a section for each rule which was checked, so that potential problems in the design can be resolved before further processing is done.

The Design Rule Check program checks a schematic drawing set for items such as missing or duplicated designators and signals that have only one connection, but does not perform any functional checks (verification) of your design.

You can choose for the report file to list

- Unconnected pins
- Multiple appearance of reference designators and missing reference designators
- Signals with only one connection
- Incorrect nets based on electrical rules (further described in "Design Rule Check Output File Format")
- Signal names sorted alphanumerically

If you are checking only a portion of a larger design, you can tell the program to bypass checks for signals with no source, no destination, or with only one connection.

Depending on your particular application, the design errors reported may or may not be significant. For example, you may have drawn a schematic diagram that has incorrect attributes assigned to the text fields. If the schematic is to be used only as a printed drawing, the reported errors are of no significance since they do not alter the printed copy. However, incorrect attributes in drawing files that are to be passed to the post processors are considered drawing errors.

Running the Design Rule Check Program

The Design Rule Check program can be run from the export menus in FutureNet or from the operating system command line. Basic menu and command line operation is explained in the *FutureNet User Manual*. Operations specific to the Design Rule Check program are explained in this section. For consecutive processing of the Design Rule Check program with other Post processors, see Chapter 1, "Post Overview."

The Design Rule Check program offers a number of types of checks for drawings. All requested design check reports are written to the same output file, but are placed on separate pages within the file.

Command Line

Run the Design Rule Check program from the command line by entering

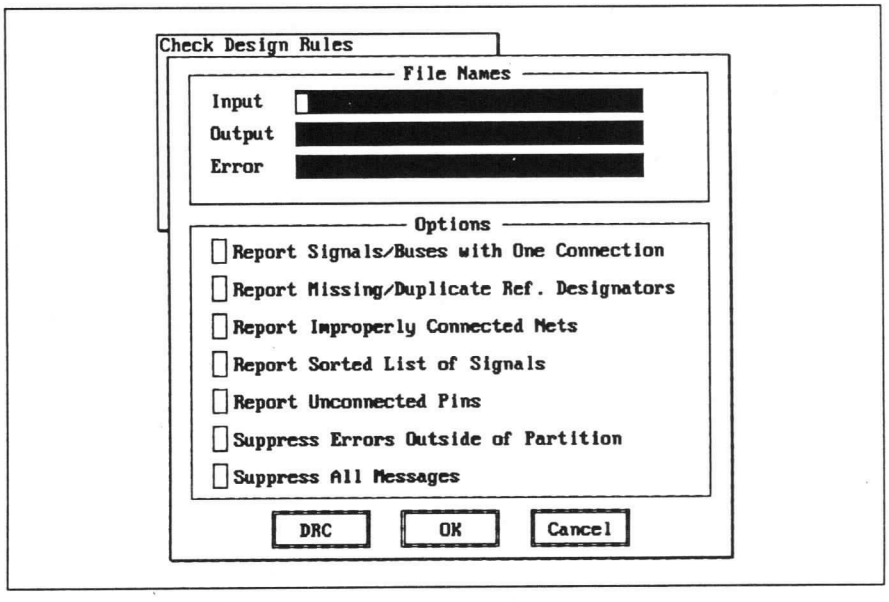
drc

In addition to the options described for the Design Rule Check program, there are some options that are common to FutureNet and the post processors. These options are described in detail, along with information on placing command line options in command files, in the *FutureNet User Manual*.

Menus

The Design Rule Check program is started from the FutureNet command menu by selecting **EXPORT Generate Reports**, then **Check Design Rules**. Figure 3-1 shows the **Check Design Rules** dialog box.

Figure 3–1
Check Design Rules Dialog Box



Design Rule Check Options

The options specific to the Design Rule Check program are discussed on the next page. The menu selection is given on the left; the command line equivalent is on the right, with the option in bold text being discussed.

Specify Filenames

Input ▮__

drc -i*dcm_file*

Use this option to specify the input .dcm file. If no extension is specified for the input file, the input filename extension .dcm is used as the default. Files with a different extension need the extension specified.

If the input file is not specified, the Design Rule Check program will prompt you for the .dcm filename and for other options.

Output ▮__

drc -i*dcm_file* -o*filename*

Use this option to specify a name for the output file. The output file, which contains the results of the checks, defaults to the name of the input file with a **.drc** extension in the current directory. The content of the output file depends upon the options chosen in the export menu or on the command line.

Error ▮__

drc -i*dcm_file* -e*filename*

Use this option to specify a name for the error file. The default filename extension is .der.

Select Report Contents

☐ Report Signals/Buses with One Connection

drc -i*dcm_file* -c

Use this option to report signals or buses that are connected to only one pin. A signal is considered to be connected to only one pin when the line is not drawn or connected by its assigned name to another symbol pin.

If an unconnected signal intentionally appears in a drawing, use the N/C signal name. Signal lines that are named N/C are ignored during this check and are not listed on the Signals With Only One Connection report.

Refer to the "Design Rule Check Output File Format" section in this chapter for a description of the Signals With Only One Connection report **.drc** output file.

Several types of signals can be connected to only one pin and are not flagged. These non-flagged signals include any that have the following alphanumeric text fields attached to them by the point of effect:

VCC
+5V
+12V
-12V
VEE
GND

In addition, any alphanumeric field with the following attributes will not be flagged:

PWRS (156)
GNDS (157)
PRLS (167)
GDLS (168)

☐ **Report Missing/Dupl. Ref. Designators**

drc -*idcm_file* **-l**

Use this option to report:

- Circuit symbols that have reference designators that are assigned to one or more other symbols in the drawing, and
- Circuit symbols that have no reference designator assigned

The attribute assigned to a reference designator can be LOC (2) or GATE (155), where the designator is checked for uniqueness throughout the entire drawing set, or LOCL (146), where the designator is checked for uniqueness within the drawing file only. A drawing with more than one symbol having the designator R1 and attribute LOCL will result in a report that lists each symbol number that has the designator R1.

This option also checks for missing designators so that designator usage can be accurately determined. Any part that contains pins (that is, has one or more text fields with a pin-related attribute that assigns a pin identifier) is checked for a reference (circuit) designator. Symbols used as drawing title blocks are not checked for the presence of reference designators.

A functional block is not checked for the presence of reference designators if **all** of the following are true:

- The functional block has a text field with a filename pointer attribute (that is, FILE or FILN), **and**
- A drawing file exists in the design with the same name as the text field, **and**
- The drawing connectivity model includes connectivity data for the drawing file.

Refer to "Design Rule Check Output File Format" section in this chapter for a description of the Missing/Duplicate Reference Designators in the **.drc** output file.

☐ **Report Improperly Connected Nets**

drc -*idcm_file* **-v**

Use this option to report possible errors that result from improper connection of nets. This report results from an examination of the **.dcm** file that checks for:

- Nets with more than one active pull-up
- Nets connected to only output pins (no destination)
- Nets connected to only input pins (no source)

- Nets connected to an active pull-up and a tristate or open collector pin
- Nets that have an active pull-up and an open collector
- Nets connected to an active pull-up and a bidirectional pin
- Nets that are connected to power or ground except for those connected to pins having a PIN or PINN attribute, or the power/ground pins (attributes GND, +5V, +12V, -12V, and VEE) of the symbol.

☐ **Report Sorted List of Signals**

drc -*idcm_file* **-n**

Use this option to generate a list of signal names, sorted alphanumerically for each drawing file and for each bus in the drawing files. Signals are not sorted across the entire design.

---

*Note: If a signal or bus line has no name, it is sorted by the name internally generated by the Drawing Preprocessor (of the form ***xxxxyyyyrrr). See "The Drawing Preprocessor" in this manual for a description of these generated names.*

To locate an unnamed net in a particular drawing, you can enter the coordinates in FutureNet (CURSOR X,Y command), or you can cross reference the signal to the netlist.

☐ **Report Unconnected Pins**

drc -*idcm_file* **-p**

Use this option to report pins that have no connections. A pin is defined as any alphanumeric field that has a pin name attribute assigned to its displayed name or number. An unconnected pin is a pin that has no signal line connected to it. Refer to the chapter titled "Understanding FutureNet" in the *FutureNet User Manual* for a list of pin-related attributes that can be used with Post Processor programs.

If a pin is connected to a signal or bus line that connects to no other pin, the pin is considered to be a connected pin and is not flagged by this function. However, the unconnected signal or bus line will be flagged as a net with only one connection. Refer to "Design Rule Check Output File Format" section in this chapter for a description of the Unconnected Pins Report in the **.drc** output file.

☐ **Suppress Errors Outside of Partition**

drc -*idcm_file* **-m**

Use this option to suppress reporting of certain signals that are normally reported as:

- Signal with only one connection
- Signal with no source
- Signal with no destination

This option is useful when a design is partitioned into smaller modules that are designed and tested separately, and then later brought together as a complete design.

This option prevents all signals that have attributes SIGI, SIGO, and SIGB from being reported if they do not connect to another symbol pin.

☐ **Suppress All Messages**

dcm -i*filename(s)* **-s**

Use this option to omit all messages from the display and error file.

Output File Format

The reports produced by Design Rule Check are written in ASCII format. The default output file generated by Design Rule Check has the same name as the input file with a **.drc** file extension.

The following information is listed on the top of every page of the Design Rule Check report:

- The name of the program (DESIGN RULE CHECK)
- The input filename with the date and time of creation
- The output filename with the data and time creation

The report file always contains a drawing tree listing all of the drawings in the design, and can contain one or more of the following reports. The command line option is given in parentheses for reference.

- Unconnected Pins (-p)
- Missing/Duplicate Designators (-l)
- Signals/Buses with One Connection (-c)
- Improperly Connected Nets (-v)
- Sorted Signal List (-n)

The following paragraphs describe each of the reports that can be generated by Design Rule Check. The sample schematic drawing shown in Figures 3-2 and 3-3 was used to generate the sample reports. Later in this chapter, the schematic errors are fixed, and the resulting reports are shown for the altered schematic.

Figure 3–2
Example Schematic (1 of 2)

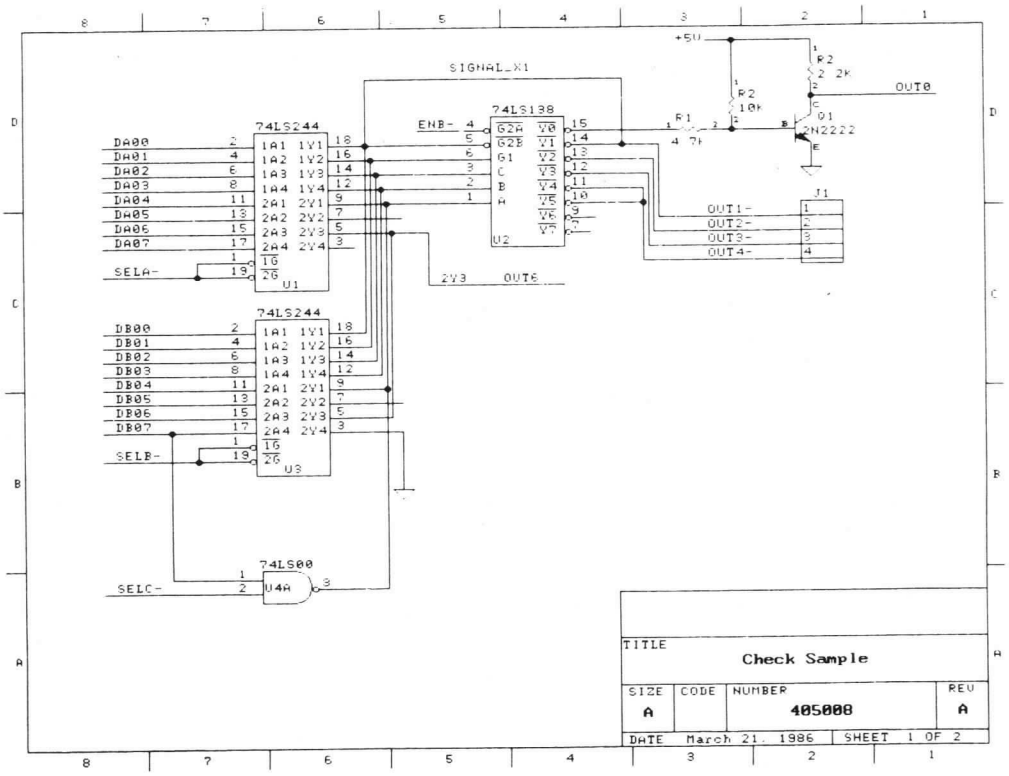
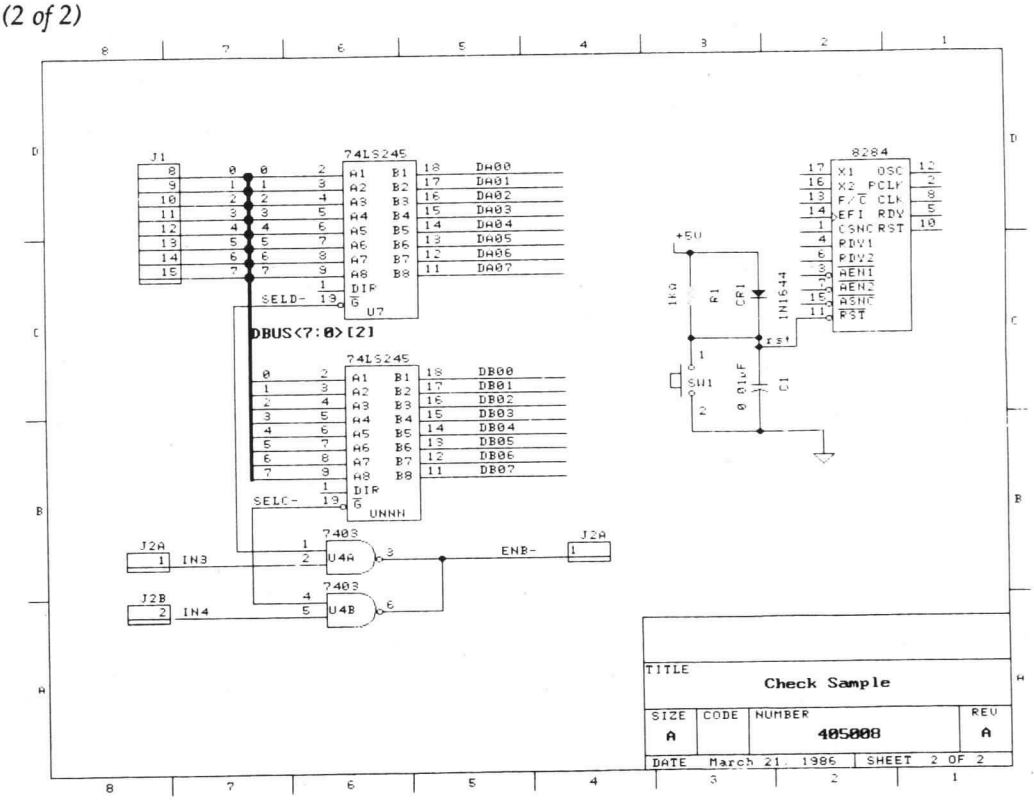


Figure 3–3
Example Schematic (2 of 2)



Drawing Tree Information

The first sheet of a report lists each of the drawings contained in the **.dcm** file. A sample drawing tree is shown in Figure 3-4. The drawings are listed in numerical order based on the drawing occurrence count assigned by the Drawing Preprocessor in the **.dcm** file.

Drawings on the **Drawing Trees** report are listed in the following syntax:

```
File: path\filename,##-##
```

path\filename is the search path and name of the FutureNet drawing file, including the extension. The two numbers separated by a hyphen are reference numbers: the first number is the drawing number assigned by the Drawing Preprocessor, and the second number is the occurrence of this drawing in the design. In the example Drawing Tree report, **sam1a.dwg** is drawing number 1 and its first occurrence in the design, and **sam2a.dwg** is drawing number 2 and its first occurrence.

The reference numbering scheme *does not* relate to any drawing and sheet numbers located on the drawing and contained in the drawing file. The drawing/sheet information (usually contained in the title block of the drawing and assigned an attribute of DPAG) is copied directly from the drawing file to the **Drawing, Revision** and **Page No** fields on the Drawing Trees report.

Figure 3–4
Drawing Tree Information

```
------------------------------------------------------------
DESIGN RULE CHECK                                    page:   1

Input File:   sam1a.dcm       Date: Wed Jan 20 11:09:56 1988
Output File:  sam1a.drc       Date: Wed Jan 20 11:11:06 1988
------------------------------------------------------------
Drawing Trees
------------------------------------------------------------
File:  \deschk\sam1a.dwg,1-1
   TITLE:  Check Sample
   Drawing:  405008
   Revision:  A
   Page No:  1 OF 2
   Date:  January 14, 1988

File:  \deschk\sam2a.dwg,2-1
   TITLE:  Check Sample
   Drawing:  405008
   Revision:  A
   Page No:  2 OF 2
   Date:  January 14, 1988
```

Unconnected Pins Report

This report lists the pins in the drawing(s) that have no signal or bus lines connected. A sample Unconnected Pin Report is shown in Figure 3-5, and was obtained by running the schematic in Figures 3-2 and 3-3 through the Drawing Preprocessor and then the Design Rule Check program.

The first column of the report lists the drawing and occurrence count number of the drawing in which the unconnected pin appears. The "Sym.Ref." column identifies the symbol associated with the unconnected pin(s). The number in this column is the symbol reference number, always located in the upper left corner of the symbol. The symbol reference number is useful in cases where there is no reference designator assigned to the symbol, or if duplicate designators have been assigned.

The "Desig.-Pin No." column lists the pin by reference designator of the symbol and the pin number. The "Description" column lists the type of pin, as determined from the attribute assigned to the pin name.

The first unconnected pin in the list is pin 3 of U1. Figure 3-2 lists this pin as on drawing 1 and in symbol 2. Drawing 2 contains several unconnected pins, some of which are identified only by the pin and symbol reference number 22. Entries with ??? in place of the reference designator indicate a missing reference designator.

Figure 3–5
Unconnected Pins Report

```
------------------------------------------------------------
DESIGN RULE CHECK                                 page:   2
Input File:     sam1a.dcm       Date: Wed Jan 20 11:09:56 1988
Output File:    sam1a.drc       Date: Wed Jan 20 11:11:06 1988
------------------------------------------------------------
UNCONNECTED PINS
Dwg-Occ Sym.Ref.        Desig.-Pin no.  Description
------------------------------------------------------------
1-1             2       U1-3            Tri-state or open collector
1-1             6       U2-4            Input
1-1             6       U2-7            Output (Active Pull-Up)
2-1             14      U7-1            Input
2-1             15      UNNN-1          Input
2-1             16      U4A-2           Input
2-1             19      J2B-2           Input connector pin
2-1             22      ???-1           Input
2-1             22      ???-2           Output (Active Pull-Up)
2-1             22      ???-3           Input
2-1             22      ???-4           Input
2-1             22      ???-5           Output (Active Pull-Up)
2-1             22      ???-6           Input
2-1             22      ???-7           Input
2-1             22      ???-8           Output (Active Pull-Up)
2-1             22      ???-10          Output (Active Pull-Up)
2-1             22      ???-12          Output (Active Pull-Up)
2-1             22      ???-13          Input
2-1             22      ???-14          Input
2-1             22      ???-15          Input
2-1             22      ???-16          Input
2-1             22      ???-17          Input
```

Missing/Duplicate Reference Designators Report

This report lists the symbols that have the same reference designator as one or more other symbols in the same drawing, and also symbols that have no reference designator. An example Missing/Duplicate Reference Designators Report is shown in Figure 3-6, and was obtained by running the partial schematic in Figures 3-2 and 3-3 through the Drawing Preprocessor and then the Design Rule Check program. The report indicates that the schematic in Figures 3-2 and 3-3 has one missing reference designator and eight symbols that share four designators.

The "Desig." column lists the reference designator(s) that appear in more than one symbol. This column is blank if a reference designator is missing. The "Dwg-Occ" column identifies the drawing in which the problem was found; the "Sym.Ref." column lists the symbol reference number. The first entry on the report in Figure 3-6 lists symbol number 22 on sheet 2 which has no reference designator. The remaining entries list J1, J2A, R1, R2, and U4A. Each of these reference designators are assigned to two different symbols, and in the case of J2A and R2, are on the same sheet.

Figure 3–6
Missing/Duplicate Reference Designators Report

```
------------------------------------------------------------------
DESIGN RULE CHECK                                    page:   3
Input File:     sam1a.dcm       Date: Wed Jan 20 11:09:56 1988
Output File:    sam1a.drc       Date: Wed Jan 20 11:11:06 1988
------------------------------------------------------------------

MISSING/DUPLICATE REFERENCE DESIGNATORS
Desig.  Dwg-Occ Sym.Ref.
------------------------------------------------------------------

        2-1             22      Missing Reference Designator
J1      1-1             12      Duplicate Reference Designator
        2-1             13
J2A     2-1             18      Duplicate Reference Designator
        2-1             27
R1      1-1             7       Duplicate Reference Designator
        2-1             23
R2      1-1             8       Duplicate Reference Designator
        1-1             10
U4A     1-1             4       Duplicate Reference Designator
        2-1             16
```

Signals with Only One Connection Report

This report lists signals that have only one connection to a symbol pin. An example Signals With Only One Connection Report is shown in Figure 3-7 , and was obtained by running the partial schematic in Figures 3-2 and 3-3 through the Drawing Preprocessor and then the Design Rule Check program. The schematic in Figure 3-3 has two signal lines that are only connected to one pin; one named IN3 and one named IN4. An examination of Figure 3-2 shows that these signal lines have small breaks in them at one end.

Figure 3–7
Signals with Only One Connection Report

```
------------------------------------------------------------------
DESIGN RULE CHECK                                    page:   4
Input File:     sam1a.dcm       Date: Wed Jan 20 11:09:56 1988
Output File:    sam1a.drc       Date: Wed Jan 20 11:11:06 1988
------------------------------------------------------------------
SIGNALS WITH ONLY ONE CONNECTION
Dwg-Occ Bus Name        Signal Name
                Sym.Ref.        Desig.-Pin No.  Description
------------------------------------------------------------------
2-1                     IN3
2-1             18      J2A-1(24)               Input connector pin
2-1                     IN4
2-1             17      U4B-5(23)               Input
```

Improperly Connected Nets Report

This report lists all signal nets that may have an error based on an examination of signal and/or bus name attributes. The Design Rule Check program checks the nets and lists the following *possible* drawing errors:

- Nets with more than one active pull-up
- Nets connected to only output pins (no destination)
- Nets connected to only input pins (no source)
- Nets connected to an active pull-up and a tristate or open collector pin
- Nets connected to an active pull-up and a bidirectional pin
- Nets that are connected to power or ground except for those connected to pins having a PIN (1) attribute, or the power/ground pins of the symbol
- Nets connected to an active pull-up and an open collector

An example report is shown in Figure 3-8. The format of the Improperly Connected Nets Report is similar to that described for the Signals with Only One Connection Report. Each net listed on the report has its associated symbol pins listed directly under it in the third column. *xy* coordinates preceded by *** are listed for signal lines that have no name assigned. The last entry for a listed net gives the reason it is on the report.

The signals listed in the report shown in Figure 3-8 are not necessarily errors. For example, nets SELA-, SELB-, SELC-, and SELD- are reported as having only one connection, no destination, or no source. In the overall design, these nets may connect to symbols which are in drawings that were not included in the **.dcm** file used to generate this report; in this case, their lack of connections would not constitute a design error. You could suppress them from the report as described in "Running the Design Rule Check Program."

Figure 3–8
Improperly Connected Nets

```
------------------------------------------------------------------
DESIGN RULE CHECK                                    page:   5
Input   File:   sam1a.dcm       Date: Sun Jan 17 13:24:10 1988
Output  File:   sam1a.drc       Date: Tue Jan 19 13:46:22 1988
------------------------------------------------------------------
IMPROPERLY CONNECTED NETS
Dwg-Occ Bus Name        Signal Name
                Sym.Ref.        Desig.-Pin No.  Description
------------------------------------------------------------------
1-1                        ***067051
1-1             2       U1-9(20)            Tri-state or open collector
1-1             6       U2-1(23)            Input
1-1             4       U4A-3(21)           Output (Active Pull-Up)
1-1             3       U3-9(20)            Tri-state or open collector
Net has both an active pull-up and a tri-state/open collector

1-1                        OUT6
1-1             2       U1-5(20)            Tri-state or open collector
1-1             3       U3-5(20)            Tri-state or open collector
Net has no destination

1-1                        ENB-
2-1             16      U4A-3(27)           Open collector pin
2-1             27      J2A-1(24)           Input connector pin
2-1             17      U4B-6(27)           Open collector pin
Net has no destination

1-1                        GND
1-1             3       U3-3(20)            Tri-state or open collector
Net has 1 signal pin(s) tied to Power or Ground (Warning)

1-1                        OUT1-
1-1             2       U1-18(20)           Tri-state or open collector
1-1             6       U2-5(23)            Input
1-1             6       U2-14(21)           Output (Active Pull-Up)
1-1             12      J1-1(25)            Output connector pin
1-1             3       U3-18(20)           Tri-state or open collector
Net has both an active pull-up and a tri-state/open collector

1-1                        OUT4-
1-1             6       U2-11(21)           Output (Active Pull-Up)
1-1             12      J1-4(25)            Output connector pin
1-1             6       U2-10(21)           Output (Active Pull-Up)
```

```
------------------------------------------------------------------
DESIGN RULE CHECK                                    page:   6
Input   File: sam1a.dcm Date: Sun Jan 17 13:24:10 1988
Output  File: sam1a.drc Date: Tue Jan 19 13:46:22 1988
------------------------------------------------------------------
IMPROPERLY CONNECTED NETS
Dwg-Occ Bus Name        Signal Name
              Sym.Ref.        Desig.-Pin No.  Description
------------------------------------------------------------------
Net has more than one active pull-up

1-1                       SELA-
1-1             2       U1-19(23)               Input
1-1             2       U1-1(23)                Input
Net has no source

1-1                       SELB-
1-1             3       U3-19(23)               Input
1-1             3       U3-1(23)                Input
Net has no source

1-1                       SELC-
1-1             4       U4A-2(23)               Input
2-1             15      UNNN-19(23)     Input
2-1             17      U4B-4(23)               Input
Net has no source

2-1                       IN3
2-1             18      J2A-1(24)               Input connector pin
Net has no destination
Net has one connection

2-1                       IN4
2-1             7       U4B-5(23)               Input
Net has no source
Net has one connection

2-1                       SELD-
2-1             14      U7-19(23)               Input
2-1             16      U4A-1(23)               Input
Net has no source
```

Sorted Signal List

This report lists the signals in alphanumeric order for

- Each drawing file, independent of other drawings in the design. Figure 3-9 shows the list of signals taken from the schematic diagram of Figures 3-2 and 3-3 and indicates the drawing occurrence for the signals.
- Each bus in the design. Figure 3-10 shows the report that lists the signals contained in the buses of the schematic in Figures 3-2 and 3-3.

Figure 3–9
Sorted Signal List

```
-------------------------------------------------------------------
DESIGN RULE CHECK                                    page:   6
Input   File:   samla.dcm       Date: Wed Jan 20 11:09:56 1988
Output  File:   samla.drc       Date: Wed Jan 20 11:11:06 1988
-------------------------------------------------------------------
THE SORTED SIGNALS:
Signal Name     Dwg-Occ
-------------------------------------------------------------------
Drawing Path: 1-1
***067042               1-1
***067045               1-1
***067048               1-1
***067051               1-1
***112036               1-1
***139035               1-1
+5V                     1-1     2-1
OUT6                    ** alias **  2Y3
                        1-1
DA00                    1-1     2-1
DA01                    1-1     2-1
DA02                    1-1     2-1
DA03                    1-1     2-1
DA04                    1-1     2-1
DA05                    1-1     2-1
DA06                    1-1     2-1
DA07                    1-1     2-1
DB00                    1-1     2-1
DB01                    1-1     2-1
DB02                    1-1     2-1
DB03                    1-1     2-1
DB04                    1-1     2-1
DB05                    1-1     2-1
DB06                    1-1     2-1
DB07                    1-1     2-1


-------------------------------------------------------------------
DESIGN RULE CHECK                                    page:   7
Input   File:   samla.dcm       Date: Wed Jan 20 11:09:56 1988
Output  File:   samla.drc       Date: Wed Jan 20 11:11:06 1988
-------------------------------------------------------------------
THE SORTED SIGNALS:
Signal Name     Dwg-Occ
-------------------------------------------------------------------
ENB-                    1-1     2-1
GND                     1-1     2-1
OUT0                    1-1
OUT1-                   1-1
OUT2-                   1-1
OUT3-                   1-1
OUT4-                   1-1
SELA-                   1-1
SELB-                   1-1
SELC-                   1-1     2-1

Drawing Path:  2-1
***147070               2-1
IN3                     2-1
IN4                     2-1
SELD-                   2-1
```

Figure 3–10
Sorted Bus Signal List

```
------------------------------------------------------------------
DESIGN RULE CHECK                                      page:   8
Input   File: saml1a.dcm Date: Wed Jan 20 11:09:56 1988
Output  File: saml1a.drc Date: Wed Jan 20 11:11:06 1988
------------------------------------------------------------------
THE SORTED BUS/SIGNALS:
Bus                     Signal Name     Dwg-Occ
------------------------------------------------------------------
Drawing Path:  2-1
DBUS                    0               2-1
DBUS                    1               2-1
DBUS                    2               2-1
DBUS                    3               2-1
DBUS                    4               2-1
DBUS                    5               2-1
DBUS                    6               2-1
DBUS                    7               2-1
```

Sorted Signal List, Special Conditions

There are two special characters that can appear adjacent to the drawing occurrence count field on a sorted signal list. These characters are * and @. The * is placed adjacent to a drawing occurrence count to signify that the signal passes through a functional block on that drawing to some real connections under it. The @ is placed adjacent to a drawing occurrence count to signify that the signal name appears at more than one location in the design. The multiple appearance of the signal name is due to multiple usage of the functional block in which it is used. The multiple appearances of a signal name in this case do not indicate that they are connected together.

Figures 3-11 through 3-13 provide an example of how the * and @ are used in the sorted signal list to indicate the above signal conditions. Figure 3-11 shows the sorted signal list for the drawing shown in Figure 3-12. In Figure 3-12, the two symbols labeled ADD1U are functional blocks. Functional block ADD1U is expanded in Figure 3-13 to show the internal signals and connections.

In Figure 3-11, signal CD has an * beside the 1-1 drawing occurrence to indicate that it passes into a functional block on this drawing (shown in Figure 3-12). The real connection of CD is to pin CD of LD3 in drawing occurrence numbers 2-1 and 2-2 (that is, the expanded block diagram).

Signal SO has an @ beside the 2-2 drawing occurrence to indicate that its usage has been repeated (Figure 3-12 shows that ADD1U is used twice) although no real connection is implied. Signals ABUS0, ABUS1, BBUS0, and BBUS1 each have * entries to indicate that they are just passing through functional blocks in the drawing with the 1-1 occurrence count.

Figure 3–11
Sorted Signals List with Special Characters

```
------------------------------------------------------------------
DESIGN RULE CHECK                                    page:   1
Input   File: adderu.dcm        Date: Fri Jan 15 13:25:30 1988
Output  File: adderu.drc        Date: Fri Jan 15 13:25:42 1988
------------------------------------------------------------------
Drawing Trees
------------------------------------------------------------------
        File:  \deschk\adderu.dwg,1-1<R>

        File:  \deschk\addlu.dwg,2-1<R>
        Path:  \deschk\adderu.dwg,1,X0<R>

        File:  \deschk\addlu.dwg,2-2<R>
        Path:  \deschk\adderu.dwg,2,X1
------------------------------------------------------------------
DESIGN RULE CHECK                                    page:   6
Input   File: adderu.dcm        Date: Fri Jan 15 13:25:30 1988
Output  File: adderu.drc        Date: Fri Jan 15 13:25:42 1988
------------------------------------------------------------------
THE SORTED SIGNALS:
Signal Name     Dwg-Occ
------------------------------------------------------------------
Dwg-Occ : 1-1
CD                    * 1-1     2-1           2-2
Dwg-Occ : 2-1
S0                      2-1
                      @ 2-2

------------------------------------------------------------------
DESIGN RULE CHECK                                    page:   7
Input   File: adderu.dcm        Date: Fri Jan 15 13:25:30 1988
Output  File: adderu.drc        Date: Fri Jan 15 13:25:42 1988
------------------------------------------------------------------
THE SORTED BUS/SIGNALS:
Bus                     Signal Name      Dwg-Occ
------------------------------------------------------------------
Dwg-Occ : 1-1
ABUS                      0       *           1-1                   2-1
ABUS                      1       *           1-1                   2-2
BBUS                      0       *           1-1                   2-1
BBUS                      1       *           1-1                   2-2
```

Figure 3–12
Drawing with Identical Functional Blocks and Occurrence Count 1-1

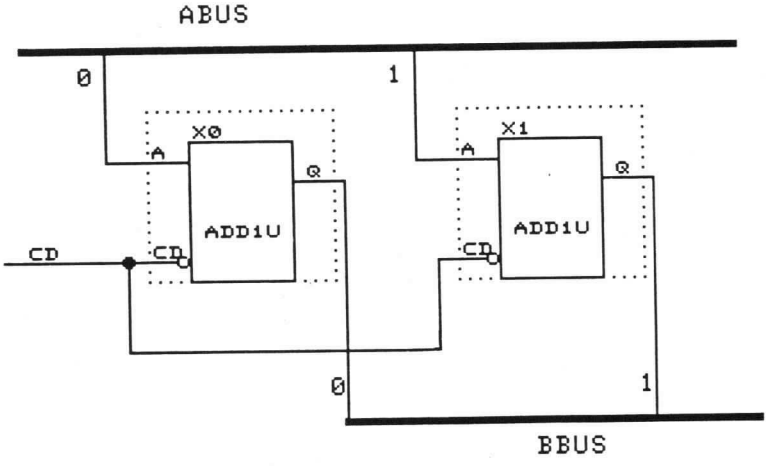
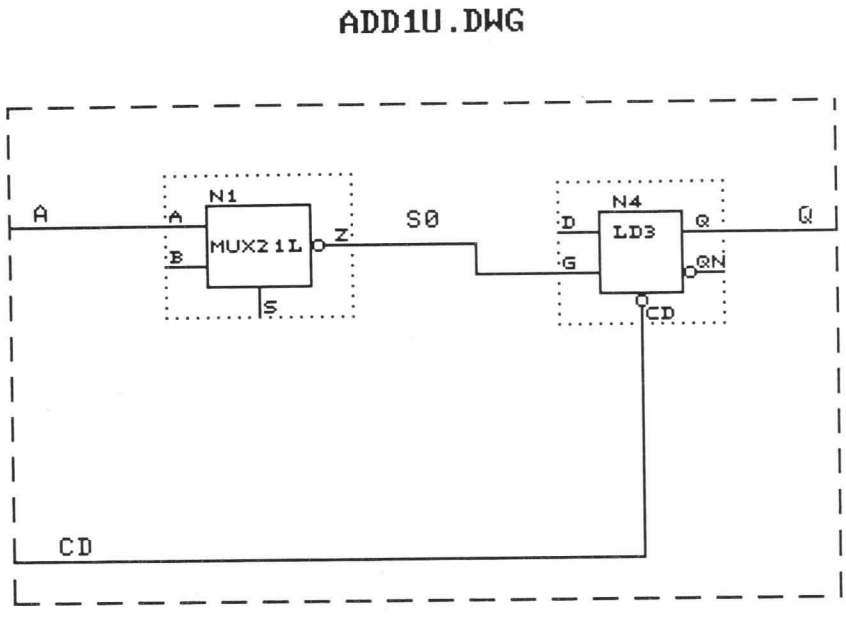


Figure 3–13
Expanded Functional Block add1u.dwg with Drawing Occurrence Counts 2-1 and 2-2



Revising the Example Schematic Diagram

Figures 3-14 and 3-15 show the same schematic diagram as shown in Figures 3-2 and 3-3 , but with corrections and alterations made. The reports shown in Figures 3-16 through 3-20 show the result of running the corrected drawing through the Drawing Preprocessor and Design Rule Check programs.

Figure 3–14
Revised Schematic (1 of 2)

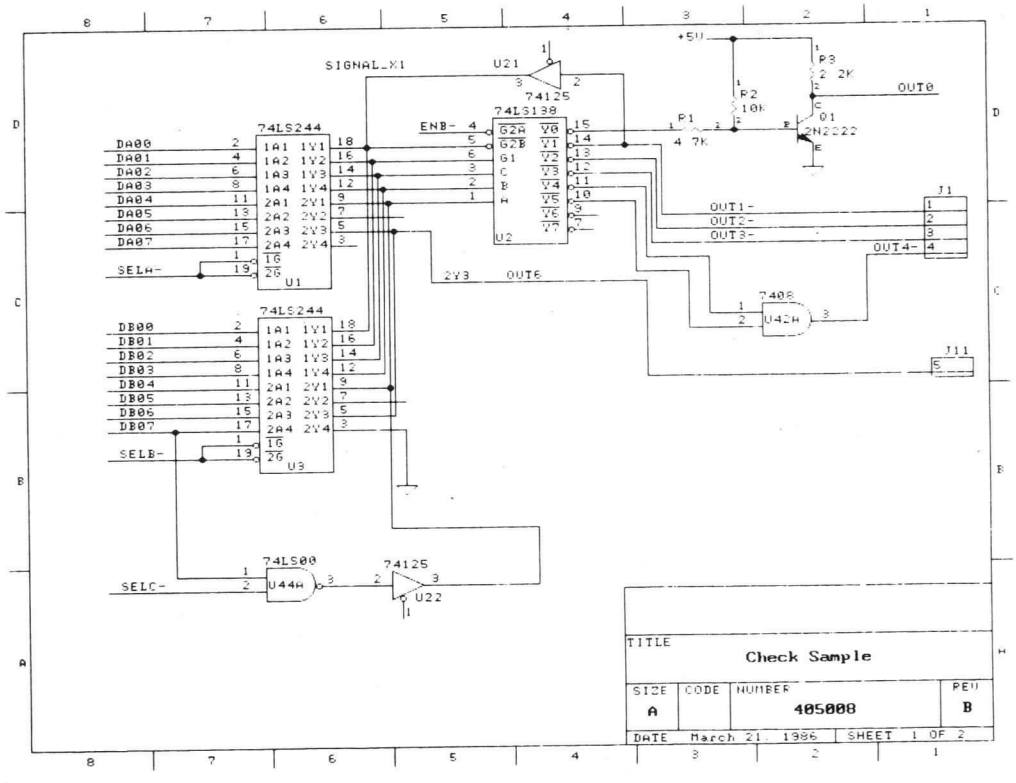


Figure 3–15
Revised Schematic (2 of 2)

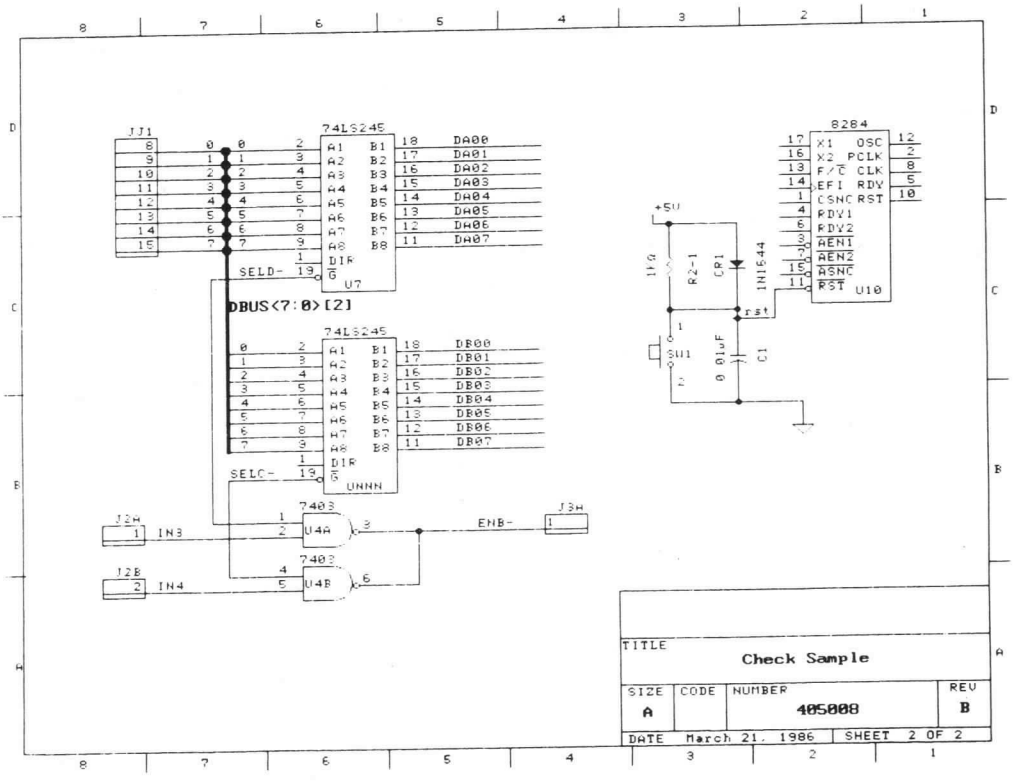


Figure 3–16
Revised Unconnected Pins Report

```
------------------------------------------------------------------
DESIGN RULE CHECK                                    page:   2
Input   File: samlb.dcm Date: Wed Jan 20 11:10:18 1988
Output  File: samlb.drc Date: Wed Jan 20 11:12:10 1988
------------------------------------------------------------------
UNCONNECTED PINS
Dwg-Occ Sym.Ref Desig.-Pin no.  Description
------------------------------------------------------------------
1-1             2       U1-3            Tri-state/open collector
1-1             6       U2-7            Output (Active Pull-Up)
1-1             15      U21-1           Input
1-1             17      U22-1           Input
2-1             14      U7-1            Input
2-1             15      UNNN-1          Input
2-1             22      U10-1           Input
2-1             22      U10-2           Output (Active Pull-Up)
2-1             22      U10-3           Input
2-1             22      U10-4           Input
2-1             22      U10-5           Output (Active Pull-Up)
2-1             22      U10-6           Input
2-1             22      U10-7           Input
2-1             22      U10-8           Output (Active Pull-Up)
2-1             22      U10-10          Output (Active Pull-Up)
2-1             22      U10-12          Output (Active Pull-Up)
2-1             22      U10-13          Input
2-1             22      U10-14          Input
2-1             22      U10-15          Input
2-1             22      U10-16          Input
2-1             22      U10-17          Input
```

Figure 3–17
Revised Missing/Duplicate
Reference Designators Report

```
------------------------------------------------------------------
DESIGN RULE CHECK                                    page:   3
Input   File: samlb.dcm Date: Wed Jan 20 11:10:18 1988
Output  File: samlb.drc Date: Wed Jan 20 11:12:10 1988
------------------------------------------------------------------
MISSING/DUPLICATE REFERENCE DESIGNATORS
Desig.          <N>   Dwg-Occ   Sym.Ref.
------------------------------------------------------------------
******* NO DUPLICATE REFERENCE DES. DETECTED ******
```

Figure 3–18
Revised Nets with Only One
Connection Report

```
------------------------------------------------------------------
DESIGN RULE CHECK                                    page:   4
Input   File: samlb.dcm Date: Wed Jan 20 11:10:18 1988
Output  File: samlb.drc Date: Wed Jan 20 11:12:10 1988
------------------------------------------------------------------
SIGNALS WITH ONLY ONE CONNECTION
Dwg-Occ Bus Name        Signal Name
                Sym.Ref.        Desig.-Pin No.  Description
------------------------------------------------------------------
******* NO ONE CONNECTION BUS/SIGNAL FOUND ******
```

Figure 3–19
Revised Improperly Connected Nets Report

```
------------------------------------------------------------------
DESIGN RULE CHECK                                      page:   5
Input   File: samlb.dcm Date: Wed Jan 20 11:10:18 1988
Output  File: samlb.drc Date: Wed Jan 20 11:12:10 1988
------------------------------------------------------------------
IMPROPERLY CONNECTED NETS
Dwg-Occ Bus Name        Signal Name
                Sym.Ref.        Desig.-Pin No.  Description
------------------------------------------------------------------
1-1                       GND
1-1             3       U3-3(20)              Tri-state or open collector
Net has 1 signal pin(s) tied to Power or Ground (Warning)<R>

1-1                       SELA-
1-1             2       U1-19(23)             Input
1-1             2       U1-1(23)              Input
Net has no source

1-1                       SELB-
1-1             3       U3-19(23)             Input
1-1             3       U3-1(23)              Input
Net has no source

1-1                       SELC-
1-1             4       U44A-2(23)            Input
2-1             15      UNNN-19(23)           Input
2-1             17      U4B-4(23)             Input
Net has no source

2-1                       SELD-
2-1             14      U7-19(23)             Input
2-1             16      U4A-1(23)             Input
Net has no source
```

Figure 3–20
Revised Sorted Signals List
(partial)

```
------------------------------------------------------------------
DESIGN RULE CHECK                                      page:   6
Input   File: samlb.dcm Date: Wed Jan 20 11:10:18 1988
Output  File: samlb.drc Date: Wed Jan 20 11:12:10 1988
------------------------------------------------------------------
THE SORTED SIGNALS:
Signal Name      Dwg-Occ
------------------------------------------------------------------
Dwg-Occ : 1-1
***064132                1-1
***067042                1-1
***067045                1-1
***067048                1-1
***067051                1-1
***112036                1-1
***112048                1-1
***112051                1-1
***139035                1-1
+5V                      1-1     2-1
OUT6                     ** alias **  2Y3
                         1-1
DA00                     1-1     2-1
DA01                     1-1     2-1
DA02                     1-1     2-1
DA03                     1-1     2-1
DA04                     1-1     2-1
DA05                     1-1     2-1
DA06                     1-1     2-1
DA07                     1-1     2-1
DB00                     1-1     2-1
DB01                     1-1     2-1
DB02                     1-1     2-1
DB03                     1-1     2-1
DB04                     1-1     2-1
```

4 Enhanced Pinlist Generator (PIN4)

The Enhanced Pinlist Generator analyzes the graphic and textual information contained in a drawing connectivity model to determine connections for each symbol pin. The Enhanced Pinlist Generator produces a pinlist in ASCII format consisting of the names of the pins of all symbols in the entire design and the names of the signals each pin connects to. This circuit connectivity information is used by other printed circuit board layout and logic simulation programs.

Enhanced and Compatible Pinlists

The Post package contains two pinlist generators. The Enhanced Pinlist Generator takes full advantage of advanced features such as layered text. The Compatible Pinlist Generator provides backward compatibility for existing products that do not support the new features.

This chapter provides detailed information about how to use the Enhanced Pinlist Generator. You will also find a detailed explanation of the resulting pinlist format in the section, "Enhanced Pinlist Output File Format." Because this new format is designed to make maximum use of layered text, it is quite different from older style pinlists. Operation of the Compatible Pinlist Generator is in the chapter, "Compatible Pinlist—Generator (PINC)."

Both of the pinlist formats described above can only be generated from drawings in the DASH-4 or later file format. Refer to the appendix in the *FutureNet User Manual* that discusses conversion of DASH-2 or DASH-3 drawings to the DASH-4 format.

Running the Enhanced Pinlist Generator

The Enhanced Pinlist Generator can be run from the FutureNet export menus or from the command line. Basic menu and command line operation is explained in the *FutureNet User Manual*. Operations specific to the Enhanced Pinlist Generator are explained in this document.

The Pinlist Generator requires a .dcm input file.

Note: For the Enhanced Pinlist Generator to function correctly, the drawing files must be available and in the same location as they were during the creation of the **.dcm** *file.*

Command Line

Run the Enhanced Pinlist Generator from the command line by entering

pin4

In addition to the options described for the Enhanced Pinlist Generator, there are some options that are common to FutureNet and the post processors. These options are described in detail, along with information on placing command line options in command files, in the *FutureNet User Manual.*

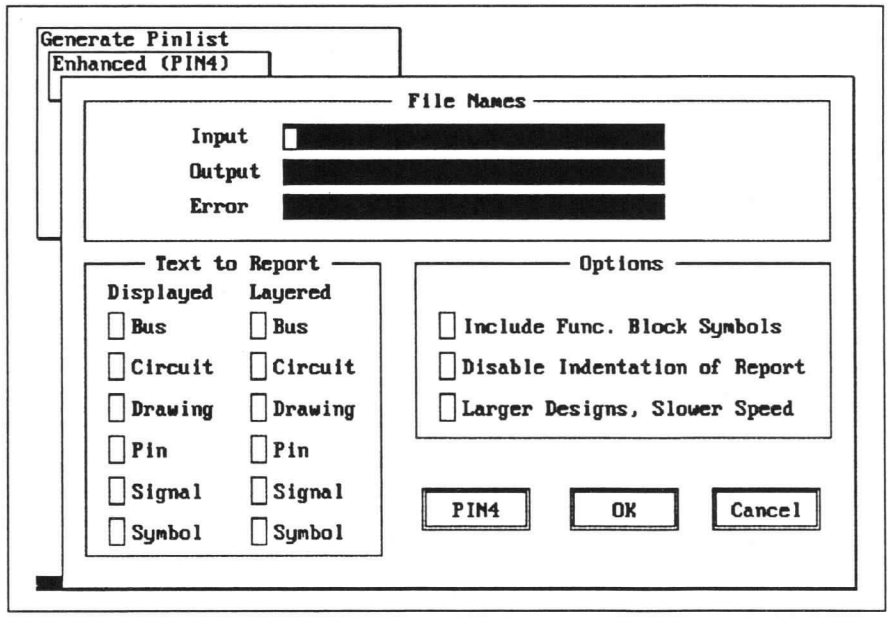
Menus

You can run the Enhanced Pinlist Generator from the FutureNet Export menus either separately or as part of a consecutive process. To run the program separately, access the Enhanced Pinlist dialog box by making the following menu selections:

EXPORT Generate Reports
Generate Pinlist
Enhanced (PIN4)

Figure 4-1 shows the dialog box for selecting Enhanced Pinlist Generator options.

Figure 4–1
Enhanced Pinlist Generator Dialog Box



After selecting your options, select the **PIN4** action button to run the program.

To run the program as part of a consecutive process, refer to the section "Export Manager" in Chapter 1, "Post Overview."

Enhanced Pinlist Generator Options

The options specific to the Enhanced Pinlist Generator are discussed below. The menu selection is given on the left; the command line equivalent is on the right, with the option in bold text being discussed.

Specify Filenames

Input

pin4 **-i***dcm_file*

Enter the .dcm file created by the Drawing Preprocessor.

In the menus, if you run the Pinlist Generator in the same session that you run the Drawing Preprocessor, the input filename is entered for you. When you enter the **.dcm** filename and press ↵, the default values for the output and error filename entry fields are completed for you.

Output

pin4 -*idcm_file* **-o***filename*

The default file name of the Enhanced Pinlist output file is *dcm_file*.**pv4**.

In the menus, the default is entered automatically when you enter an input filename. You may use the mouse to select and change the filename shown, but normally the default is satisfactory.

Error

pin4 -*idcm_file* **-e***filename*

The default name for an error file created during Enhanced Pinlist processing is *dcm_file*.**per**. This error file is created if any errors are encountered during processing.

In the menus, the default is entered automatically when you enter an input filename. You may use the mouse to select and change the filename shown, but normally the default is satisfactory.

Select Displayed and Layered Text to Include

Text to Report: Displayed

pin4 -*idcm_file* **-d***mode*

By default, all display text is included in a pinlist. Use this option to selectively suppress categories of display text from your pinlist. The *mode* argument can consist of one or more characters.

Categories of display text to be included are specified as described below:

Menu	Command Line	Description
☐ Bus	-db	Bus-related
☐ Circuit	-dc	Circuit-related
☐ Drawing	-dd	Drawing-related
☐ Pin	-dp	Pin-related
☐ Signal	-ds	Signal-related
☐ Symbol	-dy	Symbol-related

The **-d***mode* option causes the Enhanced Pinlist Generator to suppress all but specified categories of display text in the pinlist output. The **+d** option causes the Enhanced Pinlist Generator to use the default output style of including all display text.

Text to Report: Layered

pin4 -*idcm_file* **-l***mode*

By default, all layered text is included in a pinlist. Use this option to select certain categories of layered text to include in your pinlist. The *mode* argument can consist of one or more characters.

Note: You can only list categories of layered text that are also selected for display text. For example, if you exclude circuit-related display text from the Display Text Mode, you cannot include circuit-related layered text because one is dependent upon the other.

Categories of layered text to be included are specified as described below:

Menu	Command Line	Description
☐ Bus	-lb	Bus-related
☐ Circuit	-lc	Circuit-related
☐ Drawing	-ld	Drawing-related
☐ Pin	-lp	Pin-related
☐ Signal	-ls	Signal-related
☐ Symbol	-ly	Symbol-related

The **-l***mode* option causes the Enhanced Pinlist Generator to suppress all but specified categories of layered text in the pinlist output. The +l option causes the Enhanced Pinlist Generator to use the default output style that includes all layered text.

Other Options

☐ Include Func. Block Symbols

pin4 -*idcm_file* **-f**

Functional block symbols within individual drawings are not normally included in the symbol groups of a pinlist. Use this option to include functional block symbols in the pinlist.

☐ Disable Indentation of Report

pin4 -*idcm_file* **-u**

Use this option to disable pinlist formatting. By default, the pinlist is formatted by indenting successive subheadings.

☐ **Allocate Memory for a Large Design**

pin4 -*idcm_file* -m

By default, the Enhanced Pinlist Generator uses all available memory, leaving a fixed amount of buffer space for drawing data. This reduces disk accesses and significantly improves processing speed. Some large drawing files can require more than the normal amount of buffer space, and can only be processed if the buffer space is increased. The **Allocate Memory for a Large Design** option increases the available memory to process large designs. Since this option results in slower processing, you should only use this option when you receive an out-of-memory error message when generating a pinlist for a large design.

Enhanced Pinlist Output File Format

The pinlist file produced by the Enhanced Pinlist Generator is written in standard text file format. This format makes it possible for the file to be listed on the screen, edited, or processed by other programs that accept standard text input.

The file is in ASCII format and contains a fixed number of fields. These fields are separated by commas and may contain up to 255 characters each. A carriage return (Hex 0D) and line feed character (Hex 0A) appear at the end of each record.

Structure

The pinlist output is organized into nested groups. These groups consist of records of related data. Table 4–1 illustrates the nested relationship of the various groups found in a typical pinlist output.

Table 4–1
Pinlist Group Nesting

Group	Description
PINLIST,4	Header Record
DATE,...	Run Date
DCM,...	DCM File Definition Group
DATE,...	DCM File Creation Date
SET,...	Drawing Set Definition Group
FILE,...	Drawing Number and Filename
DATE,...	Drawing File Creation Date
D,...	Design Data Record
T,...	Design-Related Text

Group	Description
DWG,...	Drawing Group
PATH,...	Path Record
D,...	Drawing Data Record
T,...	Drawing-Related Text Record
SYM,...	Symbol Record
D,...	Symbol Data Record
T,...	Symbol-Related Text Record
PIN,...	Pin Record
D,...	Pin Data Record
T,...	Pin-Related Text Record
BUS,...	Bus Record
D,...	Bus Data Record
T,...	Bus-Related Text Record
BSIG,...	Bused Signal Record
EQU,...	Equate Record
D,...	Bused Signal Data Record
T,...	Bused Signal-Related Text Record
SIG,...	Signal Record
EQU,...	Equate Record
D,...	Signal Data Record
T,...	Signal-Related Text Record
DWG,...	Drawing Group

Header Group

The first group in a pinlist is the header group. It contains a header record, a run-time date record, DCM file definition group, set definition groups, and design data groups.

Header Record

The header record specifies the pinlist output format type. The header record has the following form:

PINLIST, 4

PINLIST is the record identifier.

The number 4 refers to the current pinlist version type.

Run-Time Date Record

The header record is followed by a date record which specifies the date on which the pinlist report was generated.

Date records are output throughout the pinlist report to document files and report creation dates.

All date records (regardless of where they appear) have the following form:

DATE,*ddd mmm dd hh:mm:ss yyyy*

In the above format:

DATE is the record identifier.

ddd is an abbreviation for the day of the week (i.e., Mon, Tue, etc.).

mmm is an abbreviation for the month (i.e., Jan, Mar, Dec).

dd is the numerical day of the month (i.e., 02, 15, 31).

hh is the hour of the day based on a 24 hour clock (i.e. 3:00 PM is 15:00 in 24-hour format).

mm is the minutes.

ss is the seconds.

yyyy is the complete year designation.

DCM File Definition Group

The run-time date record is followed by a DCM file definition group. The DCM group contains two records: a DCM file record and a file creation date record.

The DCM file record has the following form:

DCM,*file,num*

DCM is the record identifier.

file is the name of the **dcm** file used to generate the pinlist.

num is the number of drawing files used to create the drawing connection model.

The DCM file record is followed by a date record which specifies the file creation date of the **dcm** file.

Set Definition Group

The DCM file definition group is followed by a series of set definition groups that document the drawing sets that were used in creation of the **dcm** file.

A set definition group consists of a set record followed by a series of file and file date record pairs.

The set definition record has the following form:

SET,*num*

SET is the record identifier.

num is the number of drawing files in the set.

The file definition record has the following form:

FILE,*dwgnum,filename*

FILE is the record identifier.

dwgnum is a unique number assigned to the specific drawing (named in *filename*) in the drawing connectivity model (dcm).

filename is the name of the FutureNet drawing file (including directory path).

The file record is followed by a date record specifying the creation date of the drawing file.

Design Data Groups

The set definition group is followed by a number of data groups that document the contents of the design-related alpha fields contained within the design.

Data groups are output throughout the pinlist to document displayed alpha fields. Each data group consists of a data record followed by text records.

Data records have the following form:

D,*attr,text*

In the format shown above:

D is the record identifier.

attr is the attribute number assigned to the field.

text is the displayed text.

In the following example, 151 is the attribute number of the data field and VIDEO FRAME GRABBER is the alphanumeric contents of the data field.

D,151,VIDEO FRAME GRABBER

There is one text record for each line of layered text associated with the alpha fields.

Text records have the form:

T,*data*

T is the record identifier.

data is a line of layered text.

The example shown below illustrates:

T,COM=GATES DESIGN EXAMPLE

COM=GATES DESIGN EXAMPLE is layered text that appears under the alpha field described above.

Drawing Groups

Drawing groups contain all of the data that relates to an instance of a particular drawing file in the design. In structured designs, drawing groups may be generated several times for the same drawing if that drawing file is used multiple times in the structured design.

Drawing groups appear in the pinlist in numerical order, based upon the drawing occurrence count assigned by the drawing connectivity model.

Six types of records can appear in drawing groups. The first record, which is always present, appears as follows:

DWG,*filename,dwg-occ-cnt,ref-num*

DWG is the record identifier.

filename is the name of a FutureNet drawing file (excluding the directory path information).

dwg-occ-cnt is the drawing occurrence count. The first number in this field represents the drawing number as specified in the FILE record; the second number represents the occurrence of a drawing file in a design.

ref-num is a unique number assigned to the drawing occurrence. It is used to make unnamed or local signal names unique to the drawing occurrence.

In the following example, the drawing group contains the pinlist data for the second occurrence of the third drawing. This drawing occurrence is assigned a reference number of 6. The name of the third drawing is **frram.dwg**.

DRAWING,frram.dwg,3-2,6

The remainder of the drawing group consists of the following groups:

Path group	Bus groups
Drawing Data groups	Signal groups
Symbol groups	

Path Group and Path Records

The drawing paths used to move through the design from the root to the current drawing can be traced through the path records. Under the drawing record that names the current drawing, there is one path record for each drawing file in the design hierarchy, beginning at a root drawing and moving down to the current drawing. The path group is a series of path records as shown below:

PATH,*filename,dwg-occ-cnt,symbol-ref,loc*

PATH is the record identifier.

filename is a file along the current drawing path.

dwg-occ-cnt is the drawing occurrence count.

symbol-ref is the symbol reference number assigned by FutureNet. This number is unique for each symbol on a drawing. In FutureNet, the number is in the upper-left corner of each symbol cell.

loc is the user assigned location designator (attribute LOC/2), if any, of the symbol.

The following example shows that the current drawing was referenced from the eleventh symbol in the fourth occurrence of the second drawing. The drawing filename was **addr.dwg** and the symbol has an assigned location of U20.

PATH,addr.dwg,2-4,11,U20

Drawing record groups for root drawings do not contain path records.

Drawing Data Group

After the path group, a drawing data group is output. The data group records contain all the drawing-related alpha fields and associated layered text for the drawing file.

Symbol Groups

A symbol data group is generated for each symbol used on the drawing. Each symbol group contains a symbol record, followed by data groups and pin groups.

Symbol Records

The symbol record appears as:

SYM,*symbol-ref,loc*

SYM is the record identifier.

symbol-ref is the symbol reference number of the symbol in the filename that contains a pointer down the named drawing path.

loc is the user assigned location designator (attribute LOC/2).

In the following example, the seventh symbol in the current drawing with an assigned location of U002 is referenced.

SYM,7,U002

Symbols which have no pins and consist only of comment fields (attributes COM and NULL) are not output in the pinlist.

Symbol Data Groups

The symbol data groups show the contents of symbol-related alpha fields associated with the named symbol.

Pin Groups

Symbol data groups are followed by a series of pin groups that document the contents of all pins contained in the symbol. Pin groups contain a pin record followed by pin data groups.

Pin Records:

Pin records appear as:

PIN, *bus,bus attr,bus dwg-occ-cnt,*
sig,sig attr,sig dwg-occ-cnt,
pin attr,pin

In the above format:

PIN is the record identifier.

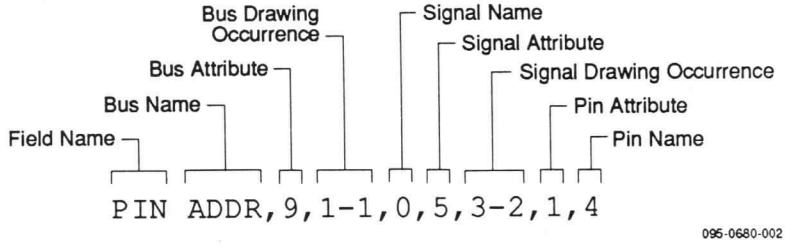
bus is the name of the bus (if any) connected to the signal line that is attached to the named pin. If a *bus* exists, the *bus attr* and *bus dwg-occ-cnt* of the substituted bus name are specified. Otherwise, all three bus fields are empty.

sig is the name of the signal line (if any) attached to the named pin. If a signal is attached to the pin, the *sig attr* and *sig dwg-occ-cnt* of the substituted signal name are specified, otherwise all three signal fields are empty.

pin attr is the attribute of the pin, and *pin* is the name of the pin.

In Figure 4–2, pin 4 with attribute 1 is connected to signal 0 with attribute 5, which, in turn, is part of bus ADDR with attribute 9. The signal name (0) was obtained from the second instance of the third drawing (3-2). The bus name (ADDR) was obtained from the first occurrence of the first root level drawing (1-1).

Figure 4–2
Sample Pin Record



Pin Data Groups:

The pin data groups show the contents of pin-related alpha fields associated with the named pin.

Bus Group

In the pinlist, signals connected to a bus are listed prior to signals not connected to a bus. Bused signals are listed in a bus group. A bus group consists of a bus record, bus data groups, and bused signal groups.

Bus Records

Bus records have the following appearance:

BUS,*bus,attr,bus dwg-occ-cnt*

BUS is the record identifier.

bus is the name of the bus line.

attr is the attribute number of the bus field used to name the bus.

dwg-occ-cnt is the drawing occurrence count.

In the following example, the bus BDATA is the first instance of drawing 2. The bus has attribute 9.

BUS,BDATA,9,2-1

When a bus is substituted from a higher level of the design, the substituted bus name, attribute, and drawing occurrence count are used in the fields of this record.

Bus Data Groups

The bus data groups document the contents of any bus-related alpha fields associated with the bus in the drawing.

Bused Signal Groups

A signal group is nested within each bus group for every signal line that connects to the bus. These bused signal groups contain a bused signal record, equate records, and bused signal data groups.

Bused Signal Record:

Bused signal records have the following form:

BSIG,*sig,attr,sig dwg-occ-cnt*

BSIG is the record identifier.

sig is the name given to the signal line.

attr is the attribute number of the named signal field.

sig dwg-occ-cnt is the drawing occurrence count.

When a signal is substituted from a higher level of the design, the substituted signal name, drawing occurrence count number, and attribute are used.

Equate Groups and Records:

When more than one name is assigned to a signal, an equate group is generated. An equate group is a series of equate records. There is one record for each additional signal name. Equate records take the form:

EQU,*bus,alias,dwg-num*

EQU is the record identifier.

bus is the bus name.

alias is another name for the signal in the primary group.

dwg-num is the drawing number.

In the example shown below, the signal in the first occurrence of the second drawing has the name EnaAD. Its alias, as shown, is E1.

SIG, EnaAD,5,2-1
EQU,,E1,2

Bused Signal Data Groups:

The bused signal data groups document the contents of any signal-related alpha fields associated with the bused signal.

Signal Group

The last groups in the pinlist output are the non-bused signal groups. A signal group consists of a signal record, equate records, and signal data groups.

Signal Record

Signal records have the following form:

SIG,*sig,attr,sig dwg-occ-cnt*

SIG is the record identifier.

sig is the name given to the signal line.

attr is the attribute number of the named signal field.

sig dwg-occ-cnt is the drawing occurrence count.

When a signal is substituted from a higher level of the design, the substituted signal name, drawing occurrence count, and attribute are used.

Equate Groups and Records

When more than one name is assigned to a signal, an equate group is generated. An equate group is a series of equate records. There is one record for each additional signal name.

Equate records take the form:

EQU,,*alias,dwg-num*

EQU is the record identifier.

alias is another name for the signal in the primary group.

dwg-num is the drawing number.

In the example shown below, the signal in the first occurrence of the second drawing has the name EnaAD. Its alias, as shown, is E1 in drawing 2.

SIG, EnaAD,5,2-1
EQU,,E1,2

Signal Data Groups

The signal data groups document the contents of any signal-related alpha fields associated with the signal.

5 Compatible Pinlist Generator (PINC)

The Compatible Pinlist Generator produces a pinlist file in the format described later in this chapter.

Note: Attributes introduced with Pinlist Generator 4.0 and above will be mapped into backwardly-compatible attribute numbers. In addition, the BUS 9 attribute is mapped to the SIG 5 attribute for version 2.0 post processors. Some variations to the output format may be obtained through the use of options that specify deviation from the pinlist format that was standard prior to version 4.0.

Enhanced and Compatible Pinlists

The Post package contains two pinlist generators. The Enhanced Pinlist Generator takes full advantage of advanced features such as layered text. The Compatible Pinlist Generator provides backward compatibility for existing products that do not support the new features.

This chapter provides detailed information about how to use the Compatible Pinlist Generator. You will also find a detailed explanation of the resulting pinlist format in the section, "Compatible Pinlist Output File Format." Operation of the Enhanced Pinlist Generator is in Chapter 4, "Enhanced Pinlist Generation (PIN4)."

The Compatible Pinlist Generator generates a pinlist that is the same format as the older standard FutureNet pinlist. You can use this format with existing post processing tools that depend upon the older format. The Compatible Pinlist Generator provides backward compatibility for drawings that have been created using the FutureNet Version 4.x and later design and drawing features. Layered text can be used, but it will not appear in compatible pinlists because it was not supported in earlier pinlist formats.

Both of the pinlist formats described above can only be generated from drawings in the DASH-4 or later file format. Refer to the appendix of the *FutureNet User Manual* that discusses conversion of DASH-2 or DASH-3 drawings to the DASH-4 format.

Running the Compatible Pinlist Generator

The Compatible Pinlist Generator can be run from the FutureNet export menus or from the command line. Basic menu and command line operation is explained in the *FutureNet User Manual*. Operations specific to the Compatible Pinlist Generator are explained in this document.

The Pinlist Generator requires a .dcm input file (a drawing connectivity model created by the Drawing Preprocessor).

Note: For the Compatible Pinlist Generator to function correctly, the drawing files must be available and in the same location as they were during the creation of the **.dcm** *file.*

Command Line

Run the Compatible Pinlist Generator from the command line by entering

pinc

In addition to the options described here for the Compatible Pinlist Generator, there are some options that are common to FutureNet and the post processors. These options are described in detail, along with information on placing command line options in command files, in the *FutureNet User Manual*.

Menus

You can run the Compatible Pinlist Generator from the FutureNet export menus either separately or as part of a consecutive process. To run the program separately, access the Compatible Pinlist dialog box by making the following menu selections:

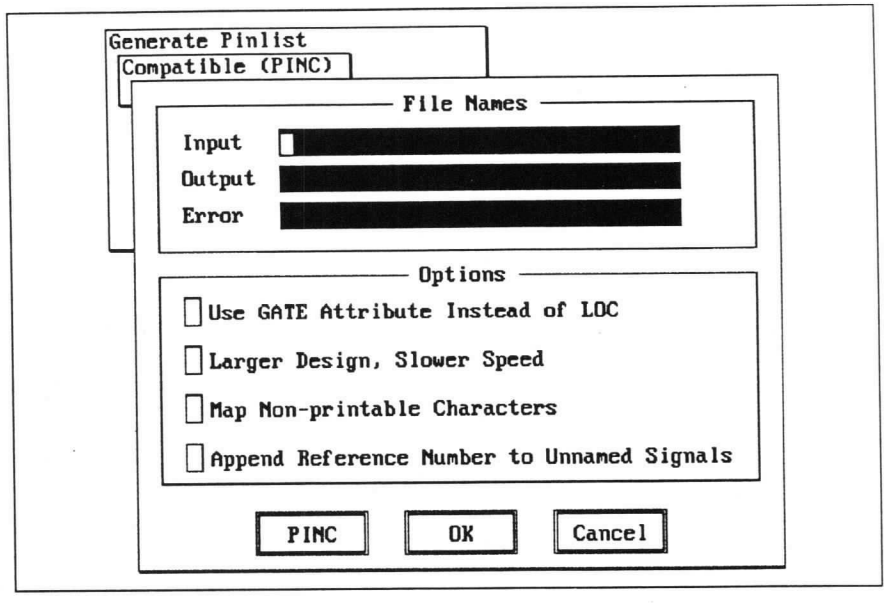
EXPORT Generate Reports
Generate Pinlist
Compatible (PINC)

Figure 5-1 shows the dialog box for selecting Compatible Pinlist Generator options.

After selecting your options, select the **PINC** action button to run the program.

To run the program as part of a consecutive process, refer to the section "Export Manager" in Chapter 1, "Post Overview."

Figure 5–1
Compatible Pinlist Generator Dialog Box



Compatible Pinlist Generator Options

The options specific to the Compatible Pinlist Generator are discussed below. The menu selection is given on the left; the command line equivalent is on the right, with the option being discussed in bold text.

Specify Filenames

Input ▮__

pinc **-i***dcm_file*

Enter the .dcm file created by the Drawing Preprocessor.

In the menus, if you run the Pinlist Generator in the same session that you run the Drawing Preprocessor, the input filename is entered for you. When you enter the **.dcm** filename and press ↵, the default values for the output and error filename entry fields are completed for you.

Output ▮__

pinc -i*dcm_file* **-o***filename*

The default file name of the Compatible Pinlist output file is *dcm_file*.**pin**.

In the menus, the default is entered automatically when you enter an input filename. You may use the mouse to select and change the filename shown, but normally the default is satisfactory.

Error ▮__

pinc -i*dcm_file* **-e***filename*

The default name for an error file created during Compatible Pinlist processing is *dcm_file*.**pcr**. This error file is created if any errors are encountered during processing.

In the menus the default is entered automatically when you enter an input filename. You may use the mouse to select and change the filename shown, but normally the default is satisfactory.

Other Options

☐ **Ignore GATE Attribute**

pinc -*idcm_file* **-g**

In earlier-format pinlists, gate information for multi-gate or multi-section parts was appended to reference designators. For example, U001B has the gate designator B appended to the basic reference designator (U001) indicating the second, or B, section of the part. FutureNet now has a specific attribute (GATE) to designate gate information. The Compatible Pinlist Generator by default appends display text with the GATE attribute to the reference designator. This option allows you to control this function.

The **-g** option indicates the GATE field display text **should be ignored and not** appended to the end of the LOC field display text.

The **+g** option resets to the default to append the GATE field display text to the LOC field display text in the pinlist output.

☐ **Allocate Memory for a Large Design**

pinc -*idcm_file* **-m**

By default, the Compatible Pinlist Generator uses all available memory, leaving a fixed amount of buffer space for drawing data. This reduces disk accesses and significantly improves processing speed. Occasionally, large drawing files require more than the normal amount of buffer space, and can only be processed if the buffer space is increased. The **Allocate Memory for a Large Design** option increases the available memory to process large designs. Since this option does result in slower processing speed, you should only use this option when you receive an out-of-memory message when generating a pinlist for a large design.

☐ **Map Non-printable Characters**

pinc -*idcm_file* **-p**

Some characters allowed by FutureNet (version 4.0 and later) are not printable on some printers. The normal range of printable characters is ASCII character code 20H (space) through 7EH (tilde~). Use this option to map non-printable characters to their octal value. This value is a multi-character sequence of *~ddd* in the pinlist output. The tilde character (~) is mapped to a double tilde (~~).

☐ **Append Reference Number to Unnamed Signals**

pinc -*idcm_file* **-r**

Earlier-format pinlists had processor-assigned names for unnamed signals. The assigned names were of the form ******xxxyyy*** or ******xxxxyyyy*** where *** appears as shown and *xxxyyy* or *xxxxyyyy* is a unique number derived from the placement of the wiring net in the schematic.

The new format (enhanced) pinlist appends a unique drawing reference number assigned by the processor to the end of the name. This reference number remains constant for all unnamed signals within the same drawing. Use this option to choose whether or not to use this feature with the earlier-format pinlist.

Compatible Pinlist Output File Format

The pinlist file produced by the Compatible Pinlist Generator is in ASCII format, and contains a fixed number of fields. These fields are separated by commas and may contain up to 255 characters each. A carriage return (Hex 0D) and line feed character (Hex 0A) appear at the end of each record.

Structure

The pinlist output is organized into nested groups. These groups consist of records of related data.

Each group begins with an open-parenthesis "(" at the front of the first record, and end with a record that contains only a closed-parenthesis ")" and a carriage return and line feed.

In the pinlist, most types of record groups are nested within other record groups. The one group around which all other data is nested is the drawing data group.

Header Data

The first record in a pinlist looks like this:

PINLIST,2

In this record, the number 2 refers to the current pinlist version type.

Drawing Data

Drawing data record groups contain all of the data that relates to an instance of a particular sheet (drawing file) in the design. Each sheet has such a group associated with it. In structured designs, a drawing data group is generated several times for the same sheet if that drawing file is used multiple times in the structure.

Drawing data groups appear in the pinlist in numerical order, based upon the path reference number. This number, assigned by the Pinlist Processor, is the combination of the drawing reference number and the sheet number, separated by a hyphen. It is used to reference the specific sheet of a design that a symbol or signal appears on.

Six types of records can appear in drawing data groups. The first record, which is always present, takes the form:

(DRAWING, ***filename, path ref***

filename is the name of a FutureNet or structured design drawing file (including DOS directory information and filename extension).

path ref is the path reference number.

The remainder of each DRAWING group consists of some combination of path data, symbol data, global drawing data, bus line data, and named signal data.

Path Data

If you are doing structured design, the DRAWING record is normally followed by a series of PATH records, so that the drawing path used to move through the structure to the current drawing file can be traced back to the root. These records take the form:

PATH, ***filename,symbol ref,ref. des***

filename is a file along the current drawing path (complete with DOS directory information and *filename* extension).

symbol ref is the reference number of the symbol in *filename* that contains a pointer down that drawing path.

ref. des is the contents of the reference designator (LOC 2) field of that symbol.

DRAWING record groups for sheets within the root drawing set do not contain PATH records.

Symbol Data

A symbol data group is generated for each symbol used on the sheet. Each symbol group contains three types of records.

The SYM Record

The first record takes the form:

(SYM, *symbol ref*

symbol ref is the symbol reference number assigned by FutureNet. This number is unique for each symbol on a sheet and is located in the upper-left corner of each symbol cell.

The DATA Record

The SYM record is followed by a number of DATA records which document the contents of the non-pin-type attribute fields contained within the symbol. These records take the form:

DATA,*attribute,text*

attribute is the attribute number of the data field.

text is the alphanumeric contents of that field.

The PIN Record

DATA records are then followed by a series of PIN records which document the contents of all pin-type attribute fields contained within the symbol. These records take the form:

PIN, *bus,signal,ref, signal attribute,pin attribute,text*

bus is the name of the bus (if any) connected to the signal line attached to the pin.

signal is the name given to that signal line. If you did not specify a name, then the Compatible Pinlist Generator generates one in the following format:

***xxxyyy**

****xxxyyy*** is a six-digit integer, derived from the position of the signal on the sheet.

ref is a system-generated field that indicates the sheets the signal can be found on. When the signal name has been specified by the user, the drawing reference number is used for the *ref* field. This is because the signal line is allowed to cross over onto any sheet of that drawing set.

When the signal name is system generated, however, the signal is, and must be unique to the specific sheet upon which it appears. For these signal lines, the path reference number appears in the *ref* field.

signal attribute and *pin attribute* are the attribute numbers associated with the signal and pin name fields, respectively.

text is the contents of the pin name field.

A record containing only a closed-parenthesis symbol ")" follows the last PIN record in each group.

Global Drawing Data

After all symbol data groups have been generated for this drawing group, a series of records are output which show the contents of all non-symbol and non-signal related alphanumeric attribute fields. These records each take the form:

DATA,*attribute,text*

attribute is the attribute number of the alphanumeric field.

text is the contents of that field.

Bus Line Data

The BUS Record

In the pinlist, signals that are connected to a bus are listed prior to signals that are not connected to a bus. Such signals are listed in a bus line data group, which starts with the record:

(BUS,*bus,ref,attribute,text*

bus is the name of the bus line.

ref is a program generated field which denotes the location of the bus in the design.

attribute is the signal attribute number of the bus field.

text is the contents of that field, usually the same as the bus name.

When no bus is specified, the Compatible Pinlist Generator generates a name of the form:

*****xxxyyy**

****xxxyyy* is a unique six number. Whenever such names are generated, the ref field is equal to the path reference number. Otherwise, the contents of the ref field is equal to the drawing reference number.

When a bus name is substituted from a higher node of the design, the attribute number is not substituted. If the bus line is named in the current drawing file, then the attribute of that field is used in the BUS record. If the bus line is either unnamed, or does not appear at this node (only the bus-signal line appears) then the attribute field is blank.

The DATA Record

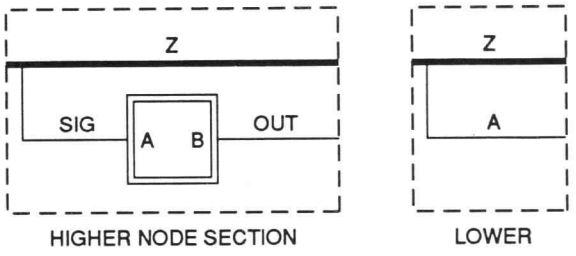
If a bus line contains two or more bus name fields with identical names but different attribute numbers or different additional data, one field is randomly chosen as the field to appear in the BUS statement, while the others appear in DATA statements. These statements take the form:

DATA,*attribute,alpha text*

attribute is the attribute number of the alternate bus name field.

alpha text is the contents of that field.

Bus name substitution can also generate DATA records. If a bus signal on a lower node is attached to an identically named bus line, the Compatible Pinlist Generator generates a BUS statement with no attribute number and a DATA statement containing the attribute for the lower node bus name. For example:



Assume the above buses and signals all have an attribute of 5. Because SIG in the higher node (path reference 1-1) attaches to pin A of the functional block, when the lower node is pinlisted, signal line A is replaced by the signal descriptor Z,SIG. Because Z is also present in the lower node, a DATA record is generated:

(BUS,Z.1-1,,Z
DATA,5,Z

The SIG Record

A signal data group is also nested within each bus data group for every line that connects to the bus. These signal data groups can contain three types of records. The first record type are SIG records in the format:

(SIG,*signal,ref,attribute,text*

signal is the name given to the signal line.

ref is the reference number assigned to the bus in the BUS record.

attribute is the attribute number of the signal name field.

text is the contents of that field.

The DATA Record

Like the bus lines themselves, if a signal line contains two or more signal name fields with identical names but different attribute numbers or different additional data, one field is randomly chosen as the field to appear in the SIG statement, while the others appear in DATA statements. The DATA records are in the format:

DATA,*attribute,alpha text*

attribute is the attribute number of the alternate signal name field.

alpha text is the contents of that field.

The EQU Record

Signals that attach to buses must always have user-supplied names. Where more than one name is assigned to a signal, a SIG record group is generated for each. However, in each group except the first (or primary) SIG group, an "equate" record is generated. This record tells the user or post-processor that the related signal data refers to the same signal line as the record group for the signal identified in the equate statement. Equate records take the form:

EQU,*bus,signal,ref*

bus is the name of the bus line of the primary group.

signal is the name of the signal in the primary group.

ref is the reference number assigned to that bus line.

A record containing only a closed-parenthesis ")" appears after the last record in each SIG group. A second closed-parenthesis record appears after the last SIG group in each BUS group.

Named Signal Data

The SIG Record

The final sub-group nested within the DRAWING group contains named signal data. A signal record is generated for each signal that had a user-specified name. Signal names that were replaced in a structured design due to signal name substitution do not appear in the pinlist.

The first record of a signal group takes the form:

SIG,*signal,ref,attribute,text*

signal is the name assigned to the signal line.

ref is a number assigned by the Compatible Pinlist Generator to reference the location of the signal in the design. Because all signals in this section gave user-supplied names, *ref* is always equal to the drawing reference number.

attribute is the attribute number associated with the signal name field.

text is the contents of that field (usually the same as the signal name).

The DATA Record

As with bus-signals, if a signal line contains two or more signal name fields with identical names but different attribute numbers or different additional data, one field is randomly chosen as the field to appear in the SIG statement, while the others appear in DATA statements. These data records are in the following format:

DATA,*attribute,alpha text*

attribute is the attribute number of the alternate signal name field.

alpha text is the contents of that field.

The EQU Record

SIG and DATA records may be followed by an equate record. An equate record appears when a signal line is assigned two or more different signal names. This record tells the user or post processor that the related signal data refers to the same signal line as the record group for the signal identified in the equate statement. Equate records take the form:

EQU,*bus,signal,ref*

bus is the bus name of the primary SIG group (if any).

signal is the signal name of the primary SIG group.

ref is the system assigned reference number. If the signal with the equated name is attached to a bus, then the reference number is equal to the reference number assigned to that bus. Otherwise, *ref* is equal to the drawing reference number of the signal.

A record containing only a closed-parenthesis ")" appears after the last record in each SIG group. A second record containing only a closed-parenthesis character appears after the last SIG group in the DRAWING group.

When all DRAWING record groups have been generated, a <Ctrl–Z> code (X'1A') is generated to signify end of file.

6 Enhanced Netlist Generator (NET4)

The Enhanced Netlist Generator analyzes the graphic and textual information contained in schematic drawings to determine connections for each signal. The netlist generator produces a netlist in ASCII format consisting of the names of the signals in the entire design and the names of the pins each signal connects to. This circuit connectivity information is used by printed circuit board layout and logic simulation programs.

Enhanced and Compatible Netlists

The Post package contains two netlist generators. The Enhanced Netlist Generator takes full advantage of advanced features such as layered text. The Compatible Netlist Generator provides backward compatibility for existing products that do not support the new features.

This chapter provides detailed information about how to use the Enhanced Netlist Generator. You will also find a detailed explanation of the resulting netlist format in the section, "Enhanced Netlist Output File Format." Because this new format is designed to make maximum use of layered text, it is quite different from older style netlists. Operation of the Compatible Netlist Generator is explained in Chapter 7, "Compatible Netlist Generator (NETC)."

Both of the netlist formats described above can only be generated from drawings in the DASH-4 or later file format. Refer to the appendix in the *FutureNet User Manual* that discusses conversion of DASH-2 or DASH-3 drawings to the DASH-4 format.

Running the Enhanced Netlist Generator

The Enhanced Netlist Generator can be run from the FutureNet export menus or from the command line. Basic menu and command line operation is explained in the *FutureNet User Manual*. Operations specific to the Enhanced Netlist Generator are explained in this document.

The Netlist Generator requires a .dcm input file (a drawing connectivity model created by the Drawing Preprocessor).

Note: For the Enhanced Netlist Generator to function correctly, the drawing files must be available and in the same location as they were during the creation of the **.dcm** *file.*

Command Line

Run the Enhanced Netlist Generator from the command line by entering

net4

In addition to the options described below for the Enhanced Netlist Generator, there are some options common to FutureNet and the post processors. These options are described in detail, along with information on placing command line options in command files, in the *FutureNet User Manual.*

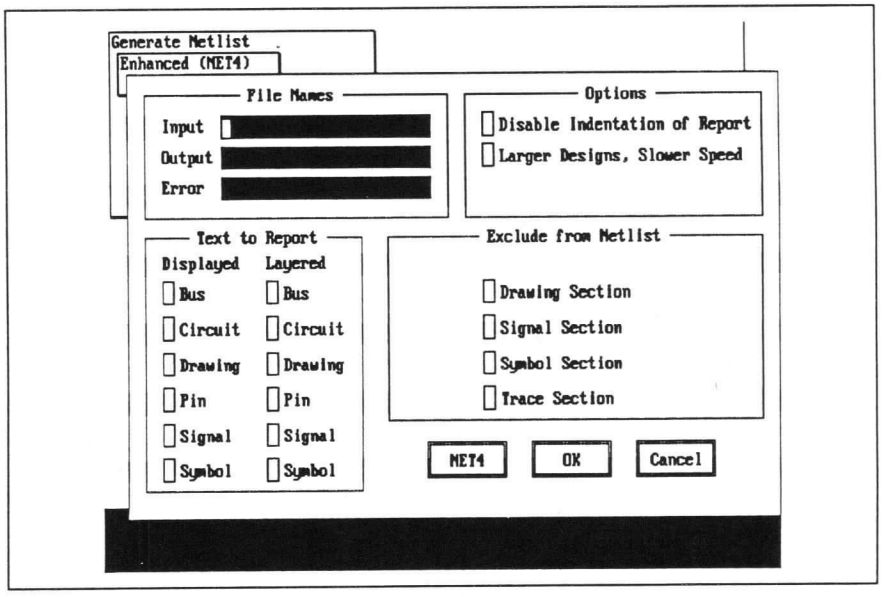
Menus

You can run the Enhanced Netlist Generator from the FutureNet export menus either separately or as part of a consecutive process. To run the program separately, access the Enhanced Netlist dialog box by making the following menu selections:

EXPORT Generate Reports
Generate Netlist
Enhanced (NET4)

Figure 6-1 shows the dialog box for selecting Enhanced Netlist Generator options.

Figure 6–1
Enhanced Netlist Generator Dialog Box



After selecting your options, select the **NET4** action button to run the program.

To run the program as part of a consecutive process, refer to the section "Export Manager" in Chapter 1, "Post Overview."

Enhanced Netlist Generator Options

The options specific to the Enhanced Netlist Generator are discussed below. The menu selection is given on the left; the command line equivalent is on the right, with the option being discussed in bold text.

Specify Filenames

Input ▮__

net4 **-i***dcm_file*

Enter the .dcm file created by the Drawing Preprocessor.

In the menus, if you run the Netlist Generator in the same session that you run the Drawing Preprocessor, the input filename is entered for you. When you entered the **.dcm** filename and press ↵, the default values for the output and error filename entry fields are completed for you.

Output ▮__

net4 -i*dcm_file* **-o***filename*

The default filename of the enhanced netlist output file is *dcm_file*.**nv4**.

In the menus, the default is entered automatically when you enter an input filename. You may use the mouse to select and change the filename shown, but normally the default is satisfactory.

Error ▮__

net4 -i*dcm_file* **-e***filename*

The default name for an error file created during enhanced netlist processing is *dcm_file*.**ner**. This error file is created if any errors are encountered during processing.

In the menus, the default is entered automatically when you enter an input filename. You may use the mouse to select and change the filename shown, but normally the default is satisfactory.

Select Displayed and Layered Text to Include

Text to Report: Displayed

net4 -i*dcm_file* **-d***mode*

By default, all display text is included in a netlist. Use this option to selectively suppress categories of display text from your netlist.

Categories of display text to be included are specified as described below:

Menu	Command Line	Description
☐ Bus	-db	Bus-related
☐ Circuit	-dc	Circuit-related
☐ Drawing	-dd	Drawing-related
☐ Pin	-dp	Pin-related
☐ Signal	-ds	Signal-related
☐ Symbol	-dy	Symbol-related

The **-d***mode* option causes the Enhanced Netlist Generator to suppress all but specified categories of display text in the netlist output. The **+d** option causes the Enhanced Netlist Generator to use the default output style of including all display text.

On the command line, you can list one or more arguments. For example, **-dbsp** includes bus-, signal- and pin-related text in the netlist.

Text to Report: Layered

net4 -i*dcm_file* **-l***mode*

By default, all layered text is included in a netlist. Use this option to select certain categories of layered text to include in your netlist.

Note: You can only list categories of layered text that are also selected for display text. For example, if you exclude circuit-related display text from the Display Text Mode, you cannot include circuit-related layered text because one is dependent upon the other.

Categories of layered text to be included are specified as described below:

Menu	Command Line	Description
☐ Bus	-lb	Bus-related
☐ Circuit	-lc	Circuit-related
☐ Drawing	-ld	Drawing-related
☐ Pin	-lp	Pin-related
☐ Signal	-ls	Signal-related
☐ Symbol	-ly	Symbol-related

The **-l***mode* option causes the Enhanced Netlist Generator to suppress all but specified categories of layered text in the netlist output. The **+l** option causes the Enhanced Netlist Generator to use the default output style that includes all layered text.

On the command line, you can list one or more arguments. For example, **-lbsp** includes bus-, signal- and pin-related text in the netlist.

Other Options

☐ Disable Indentation of Report

net4 -i*dcm_file* **-u**

Use this option to disable netlist formatting. By default, the netlist is formatted by indenting successive subheadings.

☐ Allocate Memory for a Large Design

net4 -i*dcm_file* **-m**

By default, the Enhanced Netlist Generator uses all available memory, leaving a fixed amount of buffer space for drawing data. This reduces disk accesses and significantly improves processing speed. Occasionally, large drawing files require more than the normal amount of buffer space, and can only be processed if the buffer space is increased. The **Allocate Memory for a Large Design** option increases the available memory to process large designs. Since this option does result in slower processing speed, you should only use this option when you receive an out-of-memory error message when generating a netlist for a large design.

Exclude from Netlist

net4 -i*dcm_file* -x*mode*

Use this option to exclude specific sections of the netlist. The *mode* argument can consist of one or more characters.

Sections of the netlist to be excluded are specified as described below:

Menu	Command Line	Description
☐ Drawing Section	-xd	Drawing-related
☐ Signal Section	-xs	Signal-related
☐ Symbol Section	-xy	Symbol-related
☐ Trace Section	-xt	Signal and bus substitution trace[1]

[1] *BUS-SUB and SIG-SUB groups are not included. This section can be very large.*

Enhanced Netlist Output File Format

The netlist file produced by the Enhanced Netlist Generator is written in ASCII format, and contains a fixed number of fields. These fields are separated by commas and may contain up to 255 characters each. A carriage return (Hex 0D) and line feed character (Hex 0A) appear at the end of each record.

Structure

The netlist output is organized into sections. These sections consist of groups and records of related data. Table 6-1 illustrates the nested relationship of the various sections found in a typical netlist output.

Table 6–1
Netlist Group Nesting

Group	Description
NETLIST,4	Header Record
DATE,...	Run Date Record
DCM,...	DCM File Definition Group
DATE,...	DCM File Creation Date
SET,...	Drawing Set Definition Group
FILE,...	Drawing Number and Filename
DATE,.	Drawing File Creation Date
D,...	Design Data Record
T,...	Design-Related Text
DWG,...	Drawing Section
PATH,...	Path Record
D,...	Drawing Data Record
T,...	Drawing-Related Text Record
SYM,...	Symbol Section
D,...	Symbol Data Record

T,...	Symbol-Related Text Record
SYM,...	Symbol Section
EQU,...	Equate Record
SIG,...	Signal Section
EQU,...	Equate Record
PIN,...	Pin Record
D,...	Bused Signal Data Record
T,...	Bused Signal Text Record
BUS_SUB,...	Bus Substitution Record
D,...	Bus Data Record
T,...	Bus-Related Text Record
SIG_SUB,...	Signal Substitution Record
D,...	Signal Data Record
T,...	Signal-Related Text Record

Header Section

The first section in a netlist is the header section. It contains a header record, a run-time date record, DCM file definition group, set definition groups, and design data groups.

Header Record

The header record specifies the netlist output format type. The header record has the following form:

NETLIST, 4

NETLIST is the record identifier.

The number 4 refers to the current netlist version type.

Run-Time Date Record

The header record is followed by a date record which specifies the date on which the netlist report was generated.

Date records are output throughout the netlist report to document files and report creation dates.

All date records (regardless of where they appear) have the following form:

DATE,*ddd mmm dd hh:mm:ss yyyy*

DATE is the record identifier.

In the above format:

ddd is an abbreviation for the day of the week (i.e., Mon, Tue, etc.)

mmm is an abbreviation for the month (i.e., Jan, Mar, Dec)

dd is the numerical day of the month (i.e., 02, 15, 31)

hh is the hour of the day based on a 24 hour clock (i.e. 3:00 PM is 15:00 in 24 hour format)

mm is the minutes

ss is the seconds

yyyy is the complete year designation

DCM File Definition Group

The run-time date record is followed by a DCM file definition group. The DCM group contains two records: a DCM file record and a file creation date record.

The DCM file record has the following form:

DCM,*file ,num*

DCM is the record identifier.

file is the name of the **dcm** file used to generate the netlist.

num is the number of drawing files used to create the drawing connection model.

The DCM file record is followed by a date record which specifies the file creation date of the **dcm** file.

Set Definition Group

The DCM file definition group is followed by a series of set definition groups that document the drawing sets that were used in creation of the **dcm** file.

A set definition group consists of a set record followed by a series of file and file date record pairs.

The set definition record has the following form:

SET,*num*

SET is the record identifier.

num is the number of drawing files in the set.

The file definition record has the following form:

FILE,*dwgnumfilename*

FILE is the record identifier.

dwgnum is a unique number assigned to the specific drawing (named in *filename*) in the drawing connectivity model (**dcm**).

filename is the name of the FutureNet drawing file (including directory path).

The file record is followed by a date record specifying the creation date of the drawing file.

Design Data Groups

The set definition group is followed by a number of data groups that document the contents of the design-related alpha fields contained within the design.

Data groups are output throughout the netlist to document displayed alpha fields. Each data group consists of a data record followed by text records.

Data records have the following form:

D,*attr* ,*text*

In the format shown above:

D is the record identifier.

attr is the attribute number assigned to the field.

text is the displayed text.

In the following example, 151 is the attribute number of the data field and VIDEO FRAME GRABBER is the alphanumeric contents of the data field.

D,151,VIDEO FRAME GRABBER

There is one text record for each line of layered text associated with the alpha fields.

Text records have the form:

T,*data*

T is the record identifier.

data is a line of layered text.

The example shown below illustrates:

T,COM=GATES DESIGN EXAMPLE

COM=GATES DESIGN EXAMPLE is layered text that appears under the date record alpha field, described above.

Drawing Section

Drawing groups contain all of the data that relates to an instance of a particular drawing file in the design. In structured designs, drawing groups may be generated several times for the same drawing if that drawing file is used multiple times in the structured design.

Drawing groups appear in the netlist in numerical order, based upon the drawing occurrence count assigned by the drawing connectivity model.

A drawing group consists of a drawing record, a series of path records, and data records. The first record, which is always present, appears as follows:

DWG,*filename,dwg-occ-cnt ,ref-num*

DWG is the record identifier.

filename is the name of a FutureNet drawing file (excluding the directory path information).

dwg-occ-cnt is the drawing occurrence count. The first number in this field represents the drawing number as specified in the FILE record; the second number represents the occurrence of a drawing file in a design.

ref-num is a unique number assigned to the drawing occurrence. It is used to make unnamed or local signal names unique to the drawing occurrence.

In the following example, the drawing group contains the netlist data for the second occurrence of the third drawing. This drawing occurrence is assigned a reference number of 6. The name of the third drawing is **frram.dwg**.

DRAWING,frram.dwg,3-2,6

The drawing record is followed by a path record.

Path Group and Path Records

The drawing paths used to move through the design from the root to the current drawing, can be traced through the path records. Under the drawing record that names the current drawing, there is one path record for each drawing file in the design hierarchy beginning at a root drawing and moving down to the current drawing. The path group is a series of path records as shown below:

PATH,*filename ,dwg-occ- cnt ,symbol-ref ,loc*

PATH is the record identifier.

filename is a file along the current drawing path.

dwg-occ-cnt is the drawing occurrence count.

symbol-ref is the symbol reference number assigned by FutureNet. This number is unique for each symbol on a drawing. While in FutureNet, the number can be seen in the upper-left corner of each symbol cell.

loc is the user assigned location designator (attribute LOC/2), if any, of the symbol.

The following example shows that the current drawing was referenced from the second drawing, fourth occurrence of the eleventh symbol. The drawing filename was **addr.dwg** and the symbol has an assigned location of U20.

PATH,addr.dwg,2-4,11,U20

Drawing groups for root drawings do not contain path records.

Drawing Data Group

After the path group, a drawing data group is output. The data group records contain all the drawing-related alpha fields and associated layered text for the drawing file.

Symbol Section

The symbol section consists of a series of symbol groups. There is one group for each symbol in the design. Each symbol group contains a symbol record, followed by data groups or equate records.

Symbol Records

The symbol record appears as:

SYM,*dwg-occ-cnt ,symbol-ref ,loc*

SYM is the record identifier.

dwg-occ-cnt denotes the drawing occurrence count.

symbol-ref is the symbol reference number assigned by FutureNet. The symbol reference number is unique for each symbol on a drawing. While in FutureNet, the number can be seen in the upper-left corner of each symbol cell.

loc is the user assigned location designator (attribute LOC/2), if any, of the symbol.

In the following example, the seventh symbol, with an assigned location of U002, in the second occurrence of the third drawing is referenced. The assigned symbol location is U002.

SYM,3-2,7,U002

Symbol Data Groups

The symbol data groups document the contents of symbol-related and pin-related alpha fields associated with the named symbol. Data groups are output for the first drawing occurrence of a symbol. If the symbol is in a functional block drawing that is referenced more than once in the design, all additional references to the symbol are followed by equate records.

Symbol Equate Record

The symbol equate record indicates that there are multiple occurrences of the symbol because it is in a functional block drawing that is referenced more than once. The equate record has the form:

EQU,*dwg-occ-cnt ,symbol-ref*

EQU is the record identifier.

dwg-occ-cnt denotes the first occurrence of the symbol where the symbol data group is fully documented in the netlist output.

sym-ref is the symbol reference number assigned by FutureNet.

Signal Section

The signal section consists of a series of signal groups. There is one group for each signal network in the design.

Signal Group

A signal group consists of a signal record, equate records, and a series of pin groups.

Signal records have the following form:

SIG,*bus ,bus attr ,bus dwg-occ- cnt , sig ,*
sig attr ,sig dwg-occ-cnt

In the above record:

SIG is the record identifier.

bus is the name of the bus line.

bus attr is the attribute number of the bus field used to name the bus.

bus dwg-occ-cnt is the drawing occurrence count.

sig is the name given to the signal line.

sig attr is the attribute number of the named signal field.

sig dwg-occ-cnt is the drawing occurrence count.

When a bus is substituted from a higher level of the design, the substituted bus name, attribute, and drawing occurrence count are used in the fields of this record.

When a signal is substituted from a higher level of the design, the substituted signal name, attribute, and drawing occurrence number are used.

Equate Groups and Records

When more than one name is assigned to a signal, an equate group is generated. An equate group is a series of equate records. There is one record for each additional signal name.

Equate records take the form:

EQU,*bus ,alias ,dwg-num*

EQU is the record identifier.

bus is the bus identifier, if any.

alias is another name for the signal in the primary group.

dwg-num is the drawing number where the alias originated.

In the example shown below, the signal in the first occurrence of the second drawing has the name EnnAD. Its alias, as shown, is E1. E1 originated in the third drawing.

```
SIG,,,EnnAD,5,2-1
   EQU,,E1,3
```

Pin Groups

A pin group consists of a pin record, pin data groups, bus substitution groups and signal substitution groups.

Pin records have the following form:

PIN,*pin dwg-occ-cnt ,sym-ref ,loc , pin attr ,pin*

In the above record, the fields are defined as follows:

PIN is the record identifier.

pin dwg-occ-cnt is the drawing occurrence count.

sym-ref is the symbol reference number of the symbol of which the pin is a part.

loc is the user assigned location designator (attribute LOC/2), if any, of the symbol.

pin attr is the attribute number of the pin field used to name the pin.

pin is the name of the pin.

Pin Data Groups

The pin data groups document the contents of pin-related alpha fields associated with the name pin.

Bus Substitution Group

If the signal attached to the named pin is a bused signal, the pin record is followed by one or more bus substitution groups. There is a bus substitution group for each occurrence of the bus in the hierarchy beginning where the bus originated and continuing down to the point where the bused signal is defined.

Within the bus substitution group are optional bus data groups. Bus data groups are output for each alpha field at the specified level of substitution for the bus. Within a signal group, the bus substitution for a given drawing occurrence is only output the first time the drawing is encountered. Subsequent references will only consist of the bus substitution record.

Bus substitution records have the following form:

BUS_SUB,*bus dwg-occ-cnt*

In the record shown above:

BUS_SUB is the record identifier.

bus dwg-occ-cnt is the drawing occurrence count in this bus substitution hierarchy. If it is the first definition of the substitution for this drawing occurrence count, the record is followed by bus data groups for the bus related alpha fields.

Signal Substitution Group

The pin record is also followed by one or more signal substitution groups. There is a signal substitution group for each occurrence of the signal in the hierarchy beginning where the signal originated and continuing down to the pin.

Within the signal substitution group are optional signal data groups. Signal data groups are output for each alpha field at the specified level of substitution for the signal. Signal data groups are not output if a data group for the same drawing occurrence has already been output earlier in the signal group. Again, only the first occurrence will have a data group output.

Signal substitution records have the following form:

SIG_SUB,*sig dwg-occ-cnt*

In the above record:

SIG_SUB is the record identifier.

sig dwg-occ-cnt is the drawing occurrence count in this signal substitution hierarchy. If it is the first definition of the substitution for this drawing occurrence count, the record is followed by signal data groups for the signal related alpha fields.

Signal Group Example

The following is an example of a signal group:

```
SIG,AB,5,1-1,4,5,2-1
    PIN,3-1,3,N4,21,Q
        D,21,Q
        BUS_SUB,1-1
            D,5,AB[0:3]
        BUS_SUB,2-1
            D,5,R
        SIG_SUB,2-1
            D,5,0
        SIG_SUB,3-1
            D,5,Q1
    PIN,3-2,8,N6,21,S
        D,21,S
        BUS_SUB,1-1
        BUS_SUB,2-1
        SIG_SUB,2-1
        SIG_SUB,3-2
            D,5,S1
```

In the example shown, signal 4 on bus AB is connected to two pins; Q and S. Bus AB is first named in drawing 1-1 with a SIG/5 attribute. Bus AB is passed via a functional block to drawing 2-1. In drawing 2-1, the bus is named R with a SIG/5 attribute. In drawing 2-1, signal 4 (with a SIG/5 attribute) is attached to the bus. Signal 4 is passed via functional blocks to drawing 3-1, where pin Q is connected, and to drawing 3-2, where pin S is connected. The signal is named Q1 in drawing 3-1 and S1 in drawing 3-2.

7 Compatible Netlist Generator (NETC)

The Compatible Netlist Generator produces a netlist file in the format described later in this chapter.

Note: Attributes introduced with Netlist Generator 4.0 and above will be mapped into backward compatible attribute numbers. In addition, the BUS 9 attribute is mapped to the SIG 5 attribute for version 2.0 post processors. Some variations to the output format may be obtained through the use of options that specify deviation from the netlist format that was standard prior to version 4.0.

Enhanced and Compatible Netlists

The Post package contains two netlist generators. The Enhanced Netlist Generator takes full advantage of advanced features such as layered text. The Compatible Netlist Generator provides backward compatibility for existing products that do not support the new features.

This chapter provides detailed information about how to use the Compatible Netlist Generator. You will also find a detailed explanation of the resulting netlist format later in this chapter. Operation of the Enhanced Netlist Generator is in the chapter, "Enhanced Netlist (NET4)."

The Compatible Netlist Generator generates a netlist that is the same format as the older standard netlist. You can use this format with existing post processing tools that depend upon the older format. The Compatible Netlist Generator provides backward compatibility for drawings that have been created using the FutureNet Version 4.x and later design and drawing features. Layered text can be used, but it will not appear in compatible netlists because it was not supported in earlier netlist formats.

Both of the netlist formats described above can only be generated from drawings in the DASH-4 or later file format. Refer to the appendix in the *FutureNet User Manual* that discusses conversion of DASH-2 or DASH-3 drawings to the DASH-4 format.

Running the Compatible Netlist Generator

The Compatible Netlist Generator can be run from the FutureNet menus or from the command line. Basic menu and command line operation is explained in the *FutureNet User Manual*. Operations specific to the Compatible Netlist Generator are explained in this document.

The Netlist Generator requires a .dcm input file.

Note: For the Compatible Netlist Generator to function correctly, the drawing files must be available and in the same location as they were during the creation of the **.dcm** *file.*

Command Line

Run the Compatible Netlist Generator from the command line by entering

netc

In addition to the options described here for the Compatible Netlist Generator, there are some options that are common to FutureNet and the post processors. Information on these options, and on placing command line options in command files are in the *FutureNet User Manual*.

Menus

You can run the Compatible Netlist Generator from the FutureNet export menus either separately or as part of a consecutive process. To run the program separately, access the Compatible Netlist dialog box by making the following menu selections:

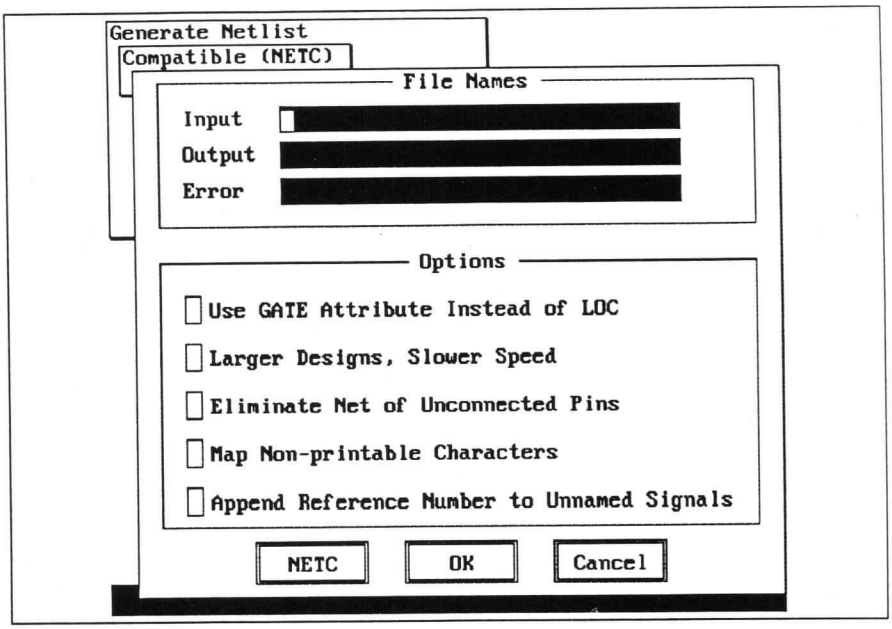
EXPORT Generate Reports
Generate Netlist
Compatible (NETC)

After selecting your options, select the **NETC** action button to run the program.

To run the program as part of a consecutive process, refer to the section "Export Manager" in Chapter 1, "Post Overview."

Figure 7-1 on the facing page shows the dialog box for selecting Compatible Netlist Generator options.

Figure 7–1
Compatible Net List Generator Dialog Box



Compatible Netlist Generator Options

The options specific to the Compatible Netlist Generator are discussed below. The menu selection is given on the left; the command line equivalent is on the right, with the option being discussed in bold text.

Specify Filenames

Input ▮__

netc **-i***dcm_file*

Enter the .dcm file created by the Drawing Preprocessor.

In the menus, if you run the Netlist Generator in the same session that you run the Drawing Preprocessor, the input filename is entered for you. When you entered the **.dcm** filename and press ↵, the default values for the output and error filename entry fields are completed for you.

Output ▮__

netc -i*dcm_file* **-o***filename*

The default file name of the Compatible Netlist output file is *dcm_file*.**net**.

In the menus, the default is entered automatically when you enter an input filename. You may use the mouse to select and change the filename shown, but normally the default is satisfactory.

Error ▮__

netc -i*dcm_file* **-e***filename*

The default name for an error file created during Compatible Netlist processing is *dcm_file*.**ncr**. This error file is created if any errors are encountered during processing.

In the menus, the default is entered automatically when you enter an input filename. You may use the mouse to select and change the filename shown, but normally the default is satisfactory.

Other Options

☐ **Ignore GATE Attribute**

netc -*idcm_file* **-g**

In earlier-format netlists, gate information for multi-gate or multi-section parts was appended to reference dsignators. For example, U001B has the gate designator B appended to the basic reference designator (U001) indicating the second, or B, section of the part. FutureNet now has a specific attribute (GATE) to designate gate information. The Compatible Netlist Generator by default appends display text with the GATE attribute to the reference designator. This option allows you to control this function.

The **-g** option indicates the GATE field display text **should be ignored and not** be appended to the end of the LOC field display text.

The **+g** option resets to the default to append the GATE field display text to the LOC field display text in the netlist output.

☐ **Allocate Memory for a Large Design**

netc -*idcm_file* **-m**

By default, the Compatible Netlist Generator uses all available memory, leaving a fixed amount of buffer space for drawing data. This reduces disk accesses and significantly improves processing speed. Occasionally, large drawing files require more than the normal amount of buffer space, and can only be processed if the buffer space is increased. The **Allocate Memory for a Large Design** option increases the available memory to process large designs. Since this option does result in slower processing speed, you should only use this option when you receive an out-of-memory error message when generating a netlist for a large design.

☐ **Exclude Net of Unconnected Pins**

netc -*idcm_file* **-n**

The earlier-format netlist contained a pseudo-net for unconnected pins. This net consisted of all unconnected pins in the design. The signal N/C is also a reserved signal name for a network of unconnected pins. Use this option to exclude this net from the output file.

☐ **Map Non-printable Characters**

netc -*idcm_file* **-p**

Some characters allowed by FutureNet (version 4.0 and later) are not printable on some printers. The normal range of printable characters is ASCII character code 20H (space) through 7EH (tilde~). Use this option to map non-printable characters to their octal value. This value is a multi-character sequence of *~ddd* in the netlist output. The tilde character (~) is mapped to a double tilde (~~).

☐ **Append Reference Number to Unnamed Signals**

netc -*idcm_file* **-r**

Earlier-format netlists had processor-assigned names for unnamed signals. The assigned names were of the form ******xxxyyy*** or ******xxxxyyyy*** where *** appears as shown and *xxxyyy* or *xxxxyyyy* is a unique number derived from the placement of the wiring net in the schematic.

The new format (Enhanced) netlist appends a unique drawing reference number assigned by the processor to the end of the name. This reference number remains constant for all unnamed signals within the same drawing. Use this option to choose whether or not to use this feature with the earlier-format netlist.

(Exclude Bus Name in Output)

netc -*idcm_file* **-b**

This option is not available from the menus. Use this command line option to exclude the bus name from the output signal records for compatibility with some earlier translators.

Compatible Netlist Output File Format

The netlist file produced by the Compatible Netlist Generator is written in ASCII format, and contains a fixed number of fields. These fields are separated by commas and may contain up to 255 characters each. A carriage return (Hex 0D) and line feed character (Hex 0A) appear at the end of each record.

Structure

Records are output in sets of structured groups that contain one or more lines of related data. For example, one group might contain a signal name record followed by records for the pins that connect to it.

Groups begin with an open-parenthesis "(" at the front of the first record, and end with a record that contains only a closed-parenthesis ")", carriage return and line feed.

The first record in a netlist looks like this:

NETLIST,2

In this record, the number 2 refers to the current netlist version type.

This record is then followed by three sections of information: drawing data, symbol data, and signal data.

Drawing Data

A drawing data group is generated for each drawing file in the design. In structured designs, a drawing data group is generated each time a file is used in the structure. Drawing data groups appear in the netlist in numerical order, based upon the path reference number.

Three types of records can appear in drawing data groups. The first record, which is always present, takes the form:

(DRAWING, ***filename, path ref***

filename is the name of a FutureNet or structured design drawing file (including DOS directory information and filename extension).

path ref is the generator-assigned path reference number.

The PATH Record

The second record type, PATH, only appears in netlists of structured designs. This record traces the drawing path used to move through the structure to the current drawing file, and should not be confused with DOS directory paths.

A record is generated for each file along the drawing path. These records take the form:

PATH, ***filename,symbol ref,circuit des***

filename is a file along the current drawing path (complete with DOS directory information and *filename* extension).

symbol ref is the reference number of the symbol in *filename* that contains a pointer down that drawing path.

circuit des is the contents of the circuit designator (LOC 2) field of that symbol.

The DATA Record

The third record type in this section is the drawing data itself. A DATA record is generated for each alphanumeric field on the sheet that has an attribute number between 50 and 54, as well as any field that is not contained within a symbol, or associated with a signal line.

These records take the form:

DATA,*attribute,alpha text*

attribute is the attribute number of the data field.

alpha text is the alphanumeric contents of that field.

A record containing only a closed-parenthesis symbol ")" follows the last DATA record in each group.

Symbol Data

A symbol data group is generated for each symbol used in the design. In structured designs, a group is generated several times for the same symbol if the file that contains that symbol is used multiple times in the structure. Each symbol group contains two types of records.

The SYM Record

The first record takes the form:

(SYM,*path ref,symbol ref*

path ref is the Compatible Netlist Generator-assigned path reference number.

symbol ref is the symbol reference number assigned by the Schematic Designer.

The DATA Record

The SYM record is then followed by a number of DATA records which document the contents of all attribute fields contained within the symbol. These records take the form:

DATA,*attribute,alpha text*

attribute is the attribute number of the data field.

alpha text is the alphanumeric contents of that field.

A record containing only a closed-parenthesis symbol ")" follows the last DATA record in each group.

Sorting SYM Records

Symbol record groups appear in the netlist in an order based on a comparison of a number of fields. In this comparison, if any of these fields are empty, they are placed in the netlist prior to records in which that particular field is not empty.

Records are first sorted in terms of part number. If two records share the same part number, then the value (attribute 4) fields are compared. Where these fields are also equal, tolerance (attribute 6), stress (attribute 7), and drawing reference number, location (attribute 2), drawing sheet number, and symbol reference number are compared in their respective order.

Fields are sorted in alphanumeric order, with numbers first.

Signal Data

The SIG Record

The third section in the netlist contains signal information. A signal (or SIG) record group is generated for each signal that appears in the input DCM file. Signal names that were replaced in the pin lists of structured design drawings through signal name substitution do not cause record groups to be generated.

Sorting SIG Records

SIG record groups appear in the netlist in a specific order determined by comparing several attribute fields. These fields, in the order in which they are compared, are bus name, signal name, drawing reference number, and sheet number.

In this comparison, if a field is empty, the record associated with it is placed in the netlist prior to records in which that particular field is not empty. Otherwise, records are sorted in alphanumeric order, with numbers first, by comparing one character of a field at a time.

The first record of a signal group takes the form:

SIG,*bus,signal,ref,attribute,text*

bus is the name of the bus (if any) which the signal is attached to. In the case of a signal that is not connected to a bus, the *bus* field is empty, although the commas are still included in the record as placeholders.

signal is the name assigned to the signal line.

ref is a number assigned by the FutureNet Compatible Netlist Generator to reference the sheet or sheets of a design on which the signal appears. When the signal name is user specified, the associated signal line may appear on any or all sheets of its drawing set. Because of this, the *ref* field contains the drawing reference number. When the signal name is instead generated by the FutureNet Compatible Netlist Generator, the signal can only be local to a single drawing file. This is because the processor has no data with which to associate it to a net on another sheet. In these cases, the path reference number is used for *ref*.

attribute is the attribute number associated with the signal name field.

text is the contents of that field (usually the same as the signal name).

The DATA Record

Occasionally, SIG records are followed by DATA records. This type of record appears when the signal line is associated with more than one alphanumeric field that has the same signal name, but different attribute numbers or additional text. The format of these records is:

DATA,*attribute,alpha text*

attribute is the attribute number of the text field.

alpha text is the contents of that field.

The EQU Record

The remainder of the SIG record group may contain one of two types of records: EQU (equate) records and PIN records.

Equate records appear when a signal line is assigned two or more different signal names. This record tells the user or post-processor products that the pin information for the signal line can be found under the equated signal name. Equate records take the form:

EQU,*bus,signal,ref*

bus is the bus name (if any).

signal is the signal name.

ref is the drawing or path reference number assigned by the Compatible Netlist Generator.

The PIN Record

Pin records appear in all other cases, and contain the information that determines the specific pins of specific symbols to which the signal line attaches. These records take the form:

PIN,*path ref,symbol ref,circuit des,attribute,text*

path ref is the path reference number of the sheet that contains the pin to which the signal connects.

symbol ref is the reference number of the symbol of which the pin is a part.

circuit des is the contents of the symbol's circuit designator (LOC 2) field.

attribute is the attribute number of the pin field.

text is the pin name found within that pin field.

Sorting PIN Records

PIN records appear in the netlist in an order based on a comparison of several alphanumeric fields. The first of these fields is drawing reference number. For records that share the same drawing reference number, the circuit designator (LOC 2) fields are then compared. Where these fields are also equal, sheet number, symbol reference number and pin name fields are compared in their respective order.

In this comparison, if a field being compared is empty, the associated record is placed in the netlist prior to records in which that particular field is not empty. Otherwise, records are sorted in alphanumeric order, with numbers first, by comparing one character in a field at a time.

A closed-parenthesis symbol ")" record, on its own line, follows the last EQU or PIN record in each SIG group.

When all SIG record groups have been generated, a <Ctrl–Z> code (X'1A') is generated to signify end of file.

8 Parts List Generator (PARTS)

The Parts List program traverses a design, drawing by drawing, and generates a list of parts needed to implement the design. The output file format is covered in the section, "Parts List Output File Format."

Part uniqueness is determined by the following properties:

- PART — part designator
- VAL — part value
- TOL — part tolerance
- STR — part stress
- PCHR — part characteristics

For the Parts List program, a part on a schematic diagram is defined as any symbol which has

- Connected pins
- Text fields with symbol-related attributes assigned to it

Running the Parts List Program

The Parts List program can be run from the FutureNet export menus or from the command line. Basic menu and command line operation is explained in the *FutureNet User Manual*. Operations specific to the Parts List program are explained in this document.

The Parts List program requires a .dcm input file.

Note: For the Parts List program to function correctly, the drawing files must be available and in the same location as they were during the creation of the **.dcm** *file.*

Command Line

Run the Parts List program from the command line by entering

parts

In addition to the options described here for the Parts List, there are some options that are common to FutureNet and the post processors. These options are described in detail, along with information on placing command line options in command files, in the *FutureNet User Manual.*

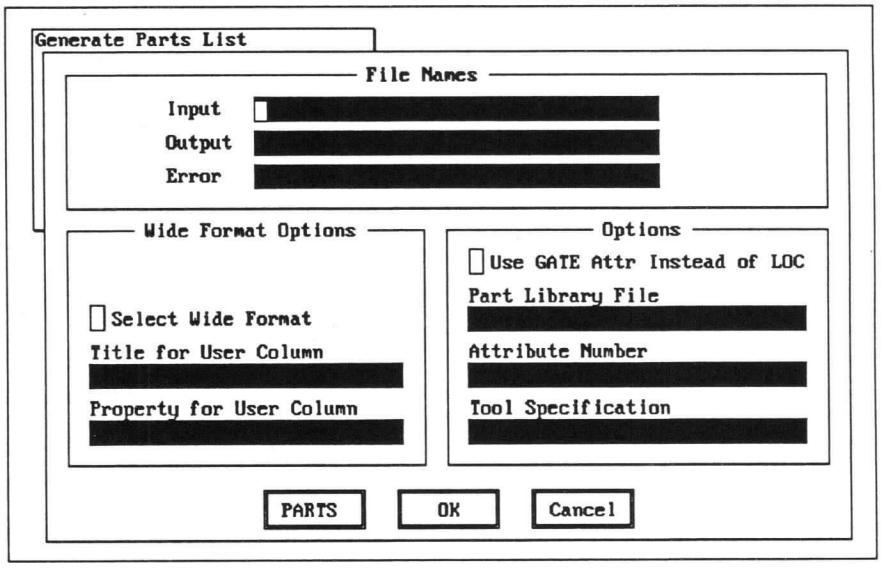
Menus

You can run the Parts List program from the FutureNet export menus either separately or as part of a consecutive process. To run the program separately, access the Parts List dialog box by making the following menu selections:

EXPORT Generate Reports
Generate Parts List

Figure 8-1 shows the dialog box for selecting Parts List program options.

Figure 8–1
Parts List Program Dialog Box



After selecting your options, select the **PARTS** action button to run the program.

To run the program as part of a consecutive process, refer to the section "Export Manager" in Chapter 1, "Post Overview."

Parts List Program Options

The options specific to the Parts List program are discussed below. The menu selection is given on the left; the command line equivalent is on the right, with the option being discussed in bold text.

Specify Filenames

Input ▮__

parts **-i***dcm_file*

Enter the .dcm file created by the Drawing Preprocessor.

In the menus, if you run the Parts List program in the same session that you run the Drawing Preprocessor, the input filename is entered for you. When you enter the **.dcm** filename and press ↵, the default values for the output and error filename entry fields are completed for you.

Output ▮__

parts -*idcm_file* **-o***filename*

The default filename of the Parts List output file is *dcm_file*.**prt**.

In the menus, the default is entered automatically when you enter an input filename. You may use the mouse to select and change the filename shown, but normally the default is satisfactory.

Error ▮__

parts -*idcm_file* **-e***filename*

The default name for an error file created during Parts List processing is *dcm_file*.**pxr**. This error file is created if any errors are encountered during processing.

In the menus, the default is entered automatically when you enter an input filename. You may use the mouse to select and change the filename shown, but normally the default is satisfactory.

Add Columns to Report

☐ Select Wide Format

parts -*idcm_file* **-w**

Use this option to specify 132 column output allowing an additional column of data in the Parts List report. You can then define a title for the column and a property to report.

Title for User Column ▮__

parts -*idcm_file* **-c***string*

Use this option to specify the title for the added column. If no title is specified, the default header DESCRIPTION is used. Choosing a header for this column automatically selects 132-column format.

Note: The title (string) cannot include any spaces. The Parts List program will consider the title to end when the first space (if any) is encountered in the string. Any word that follows the first space is interpreted as an input file, causing an error.

Property for User Column ▮__

parts -*idcm_file* **-p***property*

Use this option to specify the contents of the added column. Selecting a property for the user column automatically selects 132-column format.

If your drawing contains a part with the property VENDOR=xxx, selecting the property VENDOR (for example, -pVENDOR) writes xxx in the user column for that part. Refer to the chapter titled "Understanding FutureNet" in the *FutureNet User Manual* for methods of assigning properties to alphanumeric fields.

If no property is specified, no output will be listed in this column. If the indicated property is not specified for a given part, the column for that part will be blank.

Only one piece of data per part per drawing is included in the report, so parts used multiple times in the design should have the same value for the property specified for the User Column. The Parts List program records the first value it encounters for the property for a part and ignores the rest. Note that the drawings and the parts within a drawing are not processed in a specific order.

Other Options

☐ Use GATE Attribute to Select Gates

parts -*idcm_file* **-g**

Use Gate Attribute to Select Gates.

FutureNet allows two methods of identifying gates in a design. How the Parts List report uses these methods is controlled by this option. The two methods for gate identification are the following:

- Use of the GATE property, either using display text or by using layered text under one of the symbol related text fields in the part.
- Use of the LOC property to specify both the part reference designator and the gate in the designated part. For example, if the reference designator is U001a, then the actual reference designator of the part is U001 and the gate is "a."

Using the GATE property is the preferred method, unless strict backward compatibility with older translators is needed. For more information on properties see "Understanding FutureNet" in the *FutureNet User Manual.*

Normally, both the GATE attribute and the LOC attribute will be checked to identify gates, with the GATE attribute having precedence. However, when the **Use GATE Attribute to Select Gates** option is used, only the GATE attribute will be used to identify gates. The LOC property will be used verbatim—no attempt will be made to strip a gate designator from the text field.

When a gate is identified in a drawing, the output format of the **REFERENCE DESIGNATOR** column of the Parts List will be modified using the following rules:

- A part is only counted once for each set of gates encountered.
- An asterisk (*) is printed after the reference designator of the second and succeeding occurrence of gates using that designator.

- A comment prints on the bottom of the page indicating that the designator has been previously counted and is not included in the parts count.
- The reference designator will be followed by the gate, in brackets, as in U001[a].

Note: No checking for duplicate reference designators will be performed. Parts List will tally duplicate occurrences of a reference designator, counting all occurrences as unique. Use the Design Rule Check program to check for duplicate reference designators.

The following examples should clarify how the number of parts is computed, especially when gated parts are involved.

With the following reference designator combinations, the part count is:

U1 U1	part count = 2
U1a U1b	part count = 1
U1a U1b U1a U1b	part count = 2
U1 U1a	part count = 2
U1 U1a U1a	part count = 3
U1 U1a U1a U1b	part count = 3

The above examples are valid regardless of the status of this option.

(Suppress Date/Time Data)

parts -*idcm_file* **-z**

This option is not available from the menus. Use **-z** on the command line to suppress the printing of the date and time at the top of the output file.

(Redirect Parts List Output)

parts -*idcm_file* **-u**

This option is not available from the menus. Use **-u** on the command line to redirect the contents of the output file to standard output (stdout). By default, standard output is the screen. See your DOS manual for further information on standard output.

Specify Part Library File

Part Library File

parts -i*dcm_file* -l*part_file*

Allows specification of a parts file from which properties may be retrieved. These properties can be used to identify the parts which will be listed. Also, the property which is used in the user column can be put in this file. For more information on parts file see the *Companion Programs and Reference User Manual.*

Specify Attribute Number

Attribute Number

parts -i*dcm_file* -a*num*

The symbol-related attribute listed here are the legal attributes available for use with the -a option:

3
4
6
7
55-99
109-113
116-127
131
135
140
142-144

Attribute numbers above 99 are non-printing, displayed text attributes; they are probably the best choice since they do not print on the schematic but do appear on screen. Other symbol-related attributes will work, but we recommend that you use the attributes listed here.

Specify Tool Property

Tool Specification

parts -i*dcm_file* -t

Use this option to specify tool-specific properties. If a tool is specified for parts, properties which are tool-specific will override generic properties. The default tool is PRT. Specifying another tool causes PRT-specific properties to be ignored.

From the command line, enter **-tNONE** on the command line to specify that only generic properties be used.

Parts List Output File Format

A Part is defined as any symbol which has connected pins, or any symbol which has alpha fields with symbol related attributes. The following attributes are symbol related:

COM (0)
LOC (2)
PART (3)
VAL (4)
TOL (6)
STR (7)
SYRL (138)
LOCL (146)
GATE (155)
SYLT (161)
LOGC (165)
Reserved attributes (55-99, 110-113, 116-127, 133, 135, 140, 142, 143)

The data from the alpha fields including the displayed and layered text are processed to obtain the PART, VAL, STR, TOL, PCHR, LOC, GATE, and a possible user-specified property for generation of the parts list report.

Properties

Refer to the chapter titled "Understanding FutureNet" in the *FutureNet User Manual* for more information on properties. Circuit properties are obtained from displayed or layered circuit-related text.

Attribute	Property
CNUM (148)	CNUM = display text
CREV (149)	CREV = display text
CDAT (150)	CDAT = display text
CTIT (151)	CTIT = display text
CREL (152)	CREL = display text
CENG (153)	CENG = display text
CRLT (158)	display text is a property specification
CRRL (163)	none

Drawing properties are obtained from displayed or layered drawing-related text.

Attribute	Property
COM (0)	none
TITL (50)	TITL = display text
DNUM (51)	DNUM = display text
DREV (52)	DREV = display text
DPAG (53)	DPAG = display text
DATE (54)	DATE = display text
DENG (154)	DENG = display text
DWRL (129)	none
DWLT (159)	display text is a property specification
(55-135, 140, 142, 143)	none

Part properties are obtained from displayed or layered symbol–related text.

Attribute	Property
COM (0)	none
LOC (2)	LOC = display text
PART (3)	PART = display text
VAL (4)	VAL = display text
TOL (6)	TOL = display text
STR (7)	STR = display text
SYRL (138)	none
LOCL (146)	LOC = display text
GATE (155)	GATE = display text
SYLT (161)	display text is a property specification
(55-135, 140, 142, 143)	none
PCHR	none

The properties PART, VAL, STRESS, TOL, and PCHR are used to identify individual parts. The Parts List program uses the property PCHR for part characteristics. The value of the PCHR property can be used to refine the uniqueness of the parts. It can be used for part packaging (socketed, slim, etc), composition (ceramic, etc), vendor, etc. For example, a capacitor may have the associated line of displayed text:

```
PCHR=Mylar
```

which has been given the attribute SYLT (161). The characteristic Mylar will then be included in the PART/VAL/STRESS/TOL/PCHR column, and will distinguish that capacitor from a similar value ceramic capacitor.

Another property, defined by the user, may also be provided in layered or displayed text for a description column in the output. See the section titled Property for User Column for details on selection of the user-defined property.

The displayed text of the CRLT (158), DWLT (159), and SYLT (161) attributed alpha fields must have the following format:

[tool:] property=[value] or
[tool:] property#[value]

Layered and displayed text will be processed to obtain values for properties CNUM, CREV, CDAT, CTIT, CREL, CENG, TITL, DNUM, DREV, DPAG, DATE, DENG, PART, VAL, STR, TOL, LOC, GATE, PCHR and any user-specified property. Properties may be tool-specific or generic. The default tool for Parts List is PRT. See the section titled Tool Specification for information on making tool specifications.

A generic property should be specified only once either through displayed or layered text. A tool specific property for a given tool should be specified only once.

Part List Output Reports

The output from Parts List contains two reports: Files and Parts.

The first report is a reference listing of the input and output files including the drawings used to generate the Parts List. Parts List will detect and output Circuit Title, Circuit Number, Circuit Revision, Circuit Release, Circuit Date, Circuit Engineer, Drawing Title, Drawing Engineer, Drawing Number, Drawing Revision, Drawing Page, and Drawing Date information if present. These are determined by the properties CTIT, CNUM, CREV, CREL, CDAT, CENG, TITL, DENG, DNUM, DREV, DPAG, and DATE respectively.

The second report is a list of the parts used in the design. The first column, **QTY**, lists the total quantity in the design of the particular part referred to.

The second column, **PART/VAL/STRESS/TOL/PCHR**, lists the data associated with

- The PART property (for example, 74LS00)
- The VAL property (for example, 2.2K)
- The STR property (for example, 1/4w)
- The TOL property (for example, 1%)
- The PCHR property (for example, ceramic)

The primary sort within this report is an alphanumeric sort on the concatenation of the PART, VAL, STR, TOL, and PCHR data (for example, CAP, 0.1uF, 1/4w, 5%, ceramic).

This column is optionally followed by a description column if the wide format is selected. The header for this column is **DESCRIPTION** unless a different header has been selected (see the section Header for User Column). The column will be left blank unless the property specified for Parts List has been found associated with the part.

The next column, **DWG/CNT**, lists first the drawing number (which refers back to the first report) and the count within that drawing of the given part. This is useful in the case of multiple LOCs, as there will be only one unique REFERENCE DESIGNATOR listed no matter how many times a particular LOC is encountered within a drawing (associated with a unique PART). Each set of REFERENCE DESIGNATORS within a PART is sorted by drawing.

The last column, **REFERENCE DESIGNATORS**, is a listing of the LOC data which were found associated with each PART. An entry of **???** in this column indicates that the PART referred to had no LOC data.

Sample Parts List

The following pages show a sample Parts List Generator output report generated from the schematic sheets that follow the report. The report is generated from the command line as follows:

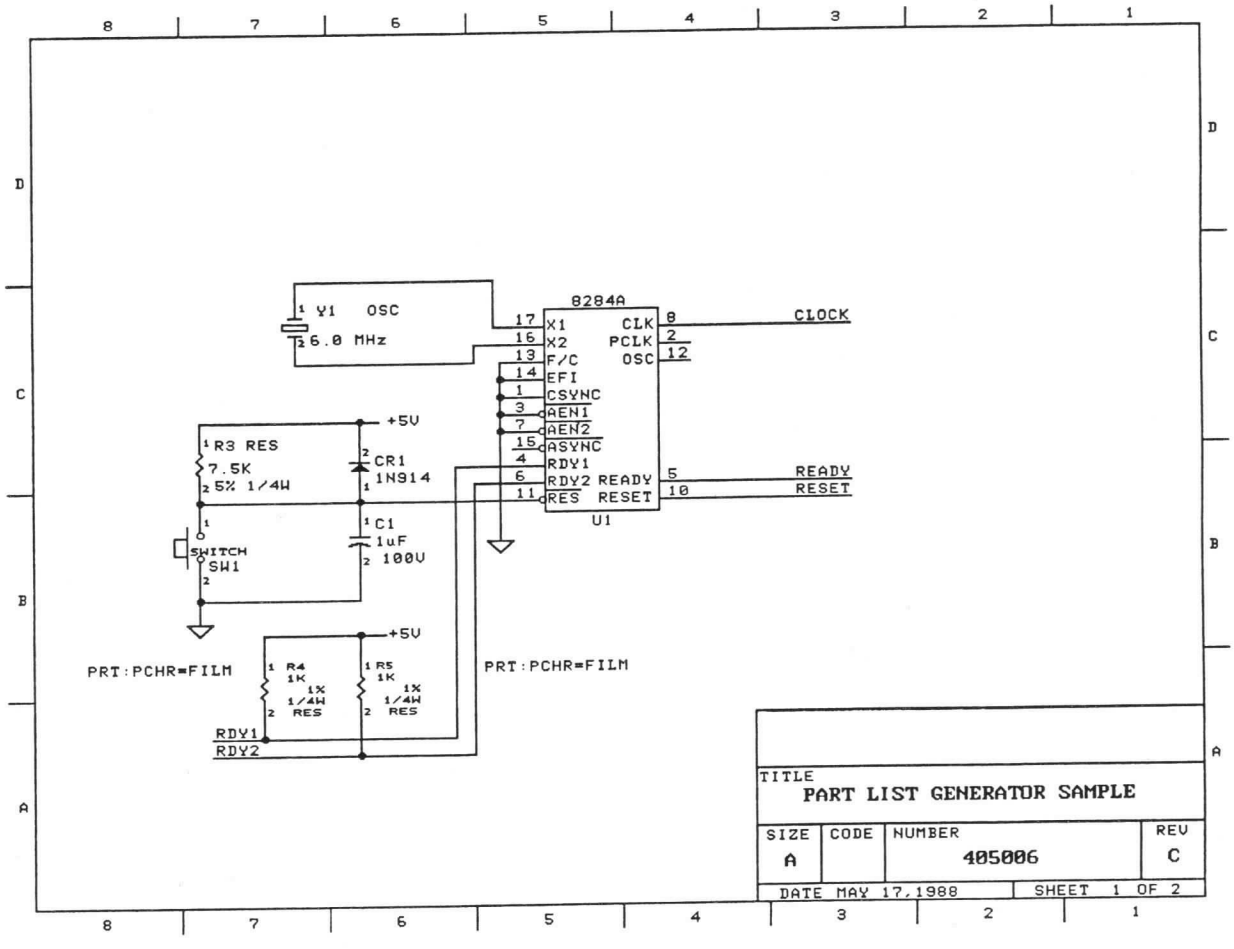
parts sampl1 sampl2

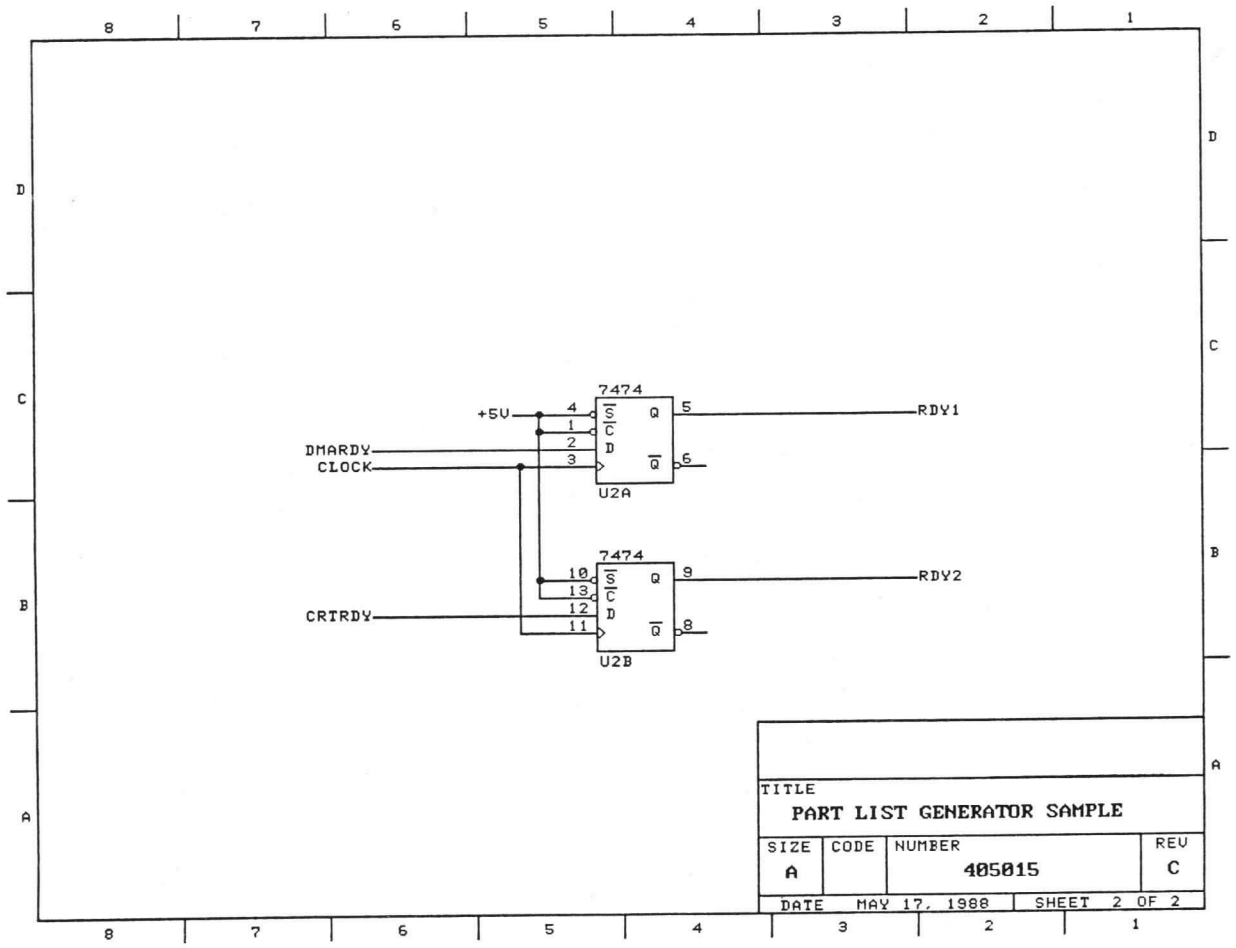
Note: The PCHR (part characteristic) field (that is, TANT, PLASTIC, CERAMIC, COMP, and N/O) in page 2 of the report is contained in layered text on the schematic diagram.

Figure 8–2
Sample Parts List

```
--------------------------------------------------------------------------------

PART LIST GENERATION                                                    page:   1
Input  File: sampl1.dcm       Date: Tue Jul 26 09:26:10 1988
Output File: sampl1.prt       Date: Tue Jul 26 09:26:30 1988

--------------------------------------------------------------------------------
Drawing Files
--------------------------------------------------------------------------------

Drawing #1:  \POST\SAMPL1.DWG
                TITLE: PARTLIST GENERATOR SAMPLE
                Drawings: 405006
                Revision: C
                Page No: 1
                Date: MAY 17,1988

Drawing #2:  \POST\SAMPL2.DWG
                TITLE: List of Materials Example
                Drawings: 405015
                Revision: C
                Page No: 2
                Date: July 3, 1985



--------------------------------------------------------------------------------

PART LIST GENERATION                                                    page:   2
Input  File: sampl1.dcm       Date: Tue Jul 26 09:26:10 1988
Output File: sampl1.prt       Date: Tue Jul 26 09:26:30 1988

--------------------------------------------------------------------------------
| QTY  |  PART/VAL/STRESS/TOL/PCHR   | DWG/CNT |     REFERENCE DESIGNATORS      |
|------|-----------------------------|---------|--------------------------------|
|    1 | 1N914                       |   1/1   | CR1                            |
|    1 | 1uF, 100V, TANT             |   1/1   | C1                             |
|    1 | 7474, PLASTIC               |   2/1   | U2[A], U2[B]*                  |
|    1 | 8284A, CERAMIC              |   1/1   | U1                             |
|    1 | OSC, 6.0 MHz                |   1/1   | Y1                             |
|    2 | RES, 1K, 1/4W, 1%, FILM     |   1/2   | R4, R5                         |
|    1 | RES, 7.5K, 1/4W, 5%, COMP   |   1/1   | R3                             |
|    1 | SWITCH, N/O                 |   1/1   | SW1                            |

  * Gated designator counted elsewhere - not included in this count
```

```
-------------------------------------------------------------------------------

PART LIST GENERATION                                              page:    3
Input  File: sampl1.dcm      Date: Tue Jul 26 09:26:10 1988
Output File: sampl1.prt      Date: Tue Jul 26 09:26:30 1988

-------------------------------------------------------------------------------

Total Parts List Time: 2 sec
```




8
7
6
5
4
3
2
1
D
C
B
A
7474
+5V
DMARDY
CLOCK
CRTRDY
RDY1
RDY2
U2A
U2B
TITLE
PART LIST GENERATOR SAMPLE
SIZE
A
CODE
NUMBER
405015
REV
C
DATE MAY 17, 1988
SHEET 2 OF 2

9 *Messages*

The following messages have been divided into two major sections. The first section contains messages specific to the Drawing Preprocessor. The second section contains general messages which can occur in the other Post programs.

Drawing Preprocessor Messages

The Drawing Preprocessor provides four levels of messages: fatal, error, warning, and general message. Errors are listed here by group, and alphabetically within groups. Words in italics, such as *name* and *number*, represent information provided by the Drawing Preprocessor as part of the error message to help locate and correct errors, for example, filenames and symbol numbers.

Fatal Errors

Fatal errors terminate the program and provide information about the problem that caused the error. No data is lost with fatal errors.

Errors

Errors do not terminate the program. The Drawing Preprocessor will continue to process the drawing files, but no model will be created. Because errors do not terminate the program, they provide a way of finding design errors that need to be corrected before a model can be built. Once the errors have been corrected, run the Drawing Preprocessor again.

Warnings

Warnings indicate that design problems may be present or that the design may not be what you are expecting, but it will not prevent a model from being created. For example, warnings can occur when two conflicting attributes refer to the same signal, for example, universal and normal assigned to the same signal.

Messages

Messages report program status.

Fatal Errors

Could not read record *number*

This error indicates that the file may have been corrupted. Read the file back into FutureNet, save it, and then try running the Drawing Preprocessor again.

Error reading drawing file *name*

Error in drawing file format, that is, the file has been corrupted. Read the file back into FutureNet, save it, and then try running the Drawing Preprocessor again.

Fatal error(s) found; no connection data appended to file

This error indicates that local, a subprocess of the Drawing Preprocessor, has found errors in the drawing file. These errors will prevent the connectivity model from being built and connection data from being appended to the end of the drawing file. Correct all the listed errors (see the error file) and then run the Drawing Preprocessor again.

File *name* **didn't open for read**

File has been read-protected.

File *name* **didn't open for write**

Unable to append local connection data to end of file. File has been write-protected.

File *name* **is not a DASH-4 (or later) drawing file**

This drawing is not in the DASH-4 file format. Only DASH-4 and later drawing files are supported. Go back into the schematic editor, load the drawing into DASH-4, save it, and then try running the Drawing Preprocessor again. Loading a DASH-3 drawing into DASH-4 automatically translates the DASH-3 drawing into the DASH-4 format.

File *name* **is not a drawing file**

The specified file is not a DASH-4 drawing file.

Illegal search path *path*

The search path has been incorrectly specified.

Input file not specified

An input file was not specified or was specified incorrectly.

Ran out of memory.....Run "local filename", then run DCM

During normal operation, the local and dcm subprocesses are transparent: the drawing preprocessor uses the drawing files (.dwg) to generate a drawing connectivity model file (.dcm). This message indicates that there was insufficient memory to complete the connectivity model for a single drawing file, so you need to run the local process separately. Refer to "Running the Local Subprocess" in the chapter titled "Using the Drawing Preprocessor."

Another option is to try partitioning the drawing file named on the screen or in the error file and running the Drawing Preprocessor again. Clear out any RAM resident programs to free up memory.

Ran out of space writing connection data

Your disk is full. Make space on the disk or go to a different output device.

Unable to close file *name*
Disk space is insufficient to close file. Make space on the disk and try again.

Unable to connect file *name*
This error occurs when local connection, a subprocess of the Drawing Preprocessor, has uncovered fatal errors. Correct the errors listed in the error file and try again.

Unable to create virtual memory file *name*
Disk space is insufficient to save file. Make space on the disk and try again.

Unable to open drawing file *name*
Drawing file is not in current directory and optional search path has not been specified, an illegal search path was specified, or the drawing is read-protected.

Unable to open file *name*
Drawing file is not in current directory and optional search path has not been specified, the drawing file was not found in any of the search paths specified, or the drawing is read-protected.

Unable to open virtual memory file *name*
Disk space is insufficient to open file, or specified drive is not ready.

Unknown record type *number*
This error indicates that the file may have been corrupted. Read the file back into FutureNet, save it, and then try running the Drawing Preprocessor again.

Virtual memory allocation error
Disk space is insufficient to store drawing connectivity model. Remove any unnecessary files from the disk and try running the Drawing Preprocessor again.

Virtual memory error
No .dcm file has been created. Post has encountered a problem while trying to resolve connectivity for the circuit. Call Customer Support at one of the telephone numbers listed in the Preface.

Errors

Ambiguous pin *name* **at** *xxxx, yyyy*
The position of the point of effect and its display text associate a pin with a symbol. A pin must relate to precisely one symbol. In this case, the pin at the specified coordinates cannot be precisely related to one symbol.

Attribute conflict on signal *name* **at** *xxxx, yyyy*
This error occurs when two signals have the same name and one of the signals has been assigned a universal attribute and the other a non-universal attribute.

Bus *name* **in file** *name* **has an unnamed signal**
Unnamed signals on a bus are not allowed.

Duplicate functional block pin name due to signal aliasing in functional block *name*
Two input pin names have been assigned to the same signal inside the functional block.

Duplicate pin name *name* **in symbol** *number* **in file** *name*	There are two pins with the same name on the same functional block.
Files *name* **and** *name* **reference each other through functional blocks**	A loop has been created between the indicated files in which a lower level drawing calls a higher level drawing. The second file listed in the error message contains the functional block that references the first file in the error message.
Functional blocks: *name* : **are not used consistently in** **symbol** *number* **in file** *name* :	A drawing file has been specified in two different drawing sets or the functional blocks are not identical (that is, they have a different number of pins).
Illegal attribute *name* **on bus** *name* **in file** *name*	A package signal attribute, for example, SIGP or SGLP, is assigned to a bus.
Illegal attribute *name* **on signal** *name* **in file** *name*	The attribute type BUSS or BUSN is assigned to a signal.
Illegal bus to signal junction at *xxxx, yyyy*	An illegal bus to signal junction has been made. This type of error can be difficult to locate. For example, it can be hidden from view under an interconnect dot.
Illegal line found starting at *xxxx, yyyy*	Overlapping or duplicate lines. Reload into FutureNet for automatic correction.
Illegal pin at *name* **at** *xxxx, yyyy*	The point of effect for this pin is shared by more than two symbols.
Multiple bus name on bus *name* **in file** *name*	Bus aliasing is not allowed. Load the indicated drawing, locate the indicated bus, and check the display text to highlight the names to the bus.
Multiple pin name on pin *name* **on symbol** *number* **in file** *name*	Two pin names within the same symbol have been assigned the same point of effect. Load the indicated drawing, locate the point of effect for the indicated pin, and check the highlighted display text for multiple pin names.
Signal *name* **not found in functional block** *name* **used in file** *name*	The indicated signal does not exist within the named functional block.
Unnamed signal on bus at *xxxx, yyyy.*	All signals connecting to a bus must be named.
Universal signal *name* **used in connection of functional block** *name*	Universal signals cannot be used to connect functional blocks.

Warnings

Ambiguous signal name *name* **at** *xxxx, yyyy*

The point of effect for the indicated signal name is incorrectly placed. It is not possible to determine which signal the point of effect applies to. Load the drawing into FutureNet and reposition the point of effect correctly.

Ambiguous symbol text *name* **at** *xxxx, yyyy*

The point of effect for the indicated symbol text has been incorrectly placed. It is not possible to determine which symbol the point of effect applies to. Load the drawing into FutureNet and reposition the point of effect correctly.

Attribute mismatch on signal *name* **in file** *name*

Conflicting attributes have been assigned to the indicated signal.

Bus pin *name* **terminates at symbol** *number* **in file** *name*

This message indicates that a bus pin terminates at an unexpanded functional block or a primitive symbol. There is no method provided for mapping the signals on the bus further. This is a condition that should probably be corrected if the design is to be transferred to PCB layout.

Drawing connection data is from a previous version of FutureNet and may be obsolete, reLOAD and SAVE each drawing in DASH-5

Connection data appended by the Drawing Preprocessor is not current. Enter the editor, load and save the drawing, and then run the Drawing Preprocessor again.

Duplicate pin *name* **at** *xxxx, yyyy*

There are two pins with the same name on the same symbol at the specified coordinates.

Functional block *name* **in symbol** *number* **in file** *name* **does not have any pins**

The indicated functional block is isolated from the rest of the circuit.

Multiple LOC fields on symbol *number* **in file** *name*

The indicated symbol has more than one location designator. This is not necessarily an error.

Pin *name* **in symbol** *number* **in file** *name* **is not connected**

There is an unconnected pin in the symbol and file indicated.

Pin *name* **not found on functional block** *name* **in file** *name*

There is a signal in the indicated functional block for which there is no pin. This is not necessarily an error.

Possible unexpected pin connection at *xxxx, yyyy*

A signal has been placed along the boundary or a portion of the boundary of a symbol, touching some or all of the pins along that boundary. All pins touched by that signal are considered connected.

Scope mismatch with signal *name* **and bus** *name.*

The scope (that is, local, set, or universal attribute assignments) of signals and buses that are connected must match. For more information on signal and bus scope, see the chapter titled "Understanding FutureNet" in the *FutureNet User Manual.*

Signal *name* at *xxxx, yyyy* may conflict with universal signals

One of the special signal names (+5V, +12V, -12V, GND, or VEE) has been assigned to a signal that requires but does not have a universal signal attribute (for example, SIGU or SGNU) assigned to it. Special signal names can only be used with signals that have been assigned one of the universal signal attributes.

Signal *name* at *xxxx, yyyy* not on line

The point of effect for the indicated signal is not attached to a line. Load the drawing into FutureNet and reposition the point of effect for the indicated signal.

Symbol *number* in file *name* is unconnected

There is an isolated circuit in the design.

Temporary line found at *xxxx, yyyy*

There is a temporary line in the drawing with an end point at the listed coordinates. Temporary lines can be connected in ways you may not be expecting. Make temporary lines permanent with the /P command.

Unable to open drawing file *name*; functional block ignored

A drawing specified in a functional block has not been found; the drawing is read-protected, or the model is created without the drawing.

Unrecognized option *name* ignored

An option that does not match the list of supported options has been encountered.

Messages

Connection data at end of file *name*

This file has not been edited since last run through the Drawing Preprocessor. Connection data is current.

Global connection complete; created file *name*

The drawing connectivity model has been created.

Building connection data at end of file *name*

This message indicates that local connection has not yet been run on the specified file. Local connection runs automatically when the Drawing Preprocessor is invoked.

Checking drawings for connection data

Reading local connectivity data.

General Messages

The following messages may be generated by the Post programs. A brief explanation of what has occurred and possible corrective action accompanies each message. Other messages may be generated by the operating system or the user interface. Refer to the *FutureNet User Manual* for clarification of messages generated by the user interface.

Can't open .DRC report file *name*

The Design Rule Check program was unable to create the output file *name*. Possible reasons for this failure are lack of sufficient disk space or a write-protected file with the same name as the file *name*.

Drawing file *name* **not used to create the named model**

The named drawing file was not used to create the drawing connectivity model. Possibly a filename has been changed. Try rerunning **dcm**.

File *name* **is not a valid DCM file**

The file *name* may have the correct name, but it is not the correct file format for a **dcm** input file. You must provide a valid **dcm** file for input.

Out of memory

The program was unable to allocate sufficient memory. For Pinlist and Netlist Generator, try again with the **-m** option to run with the minimum memory requirement for the design.

Unable to open DCM file *name*

The program was unable to open the **dcm** input file *name*. It is possible that the named file did not exist or was not in the current directory or specified directory path. Check your file to ensure that it exists in a directory where the program can find it.

Unable to open drawing file *name*

The program was unable to open the drawing file *name*. It is possible that the named file did not exist, was not in the current directory or directory path, or is read-protected. Check your file to ensure that it exists in a directory where the program can find it.

Unable to open error file *name*

The program encountered an error but was unable to open the output error file. Error messages will be written to **stderr**.

Unable to open netlist output file *name*

The Netlist Generator was unable to create the output file *name*. Possible reasons for this failure are lack of sufficient disk space or a write-protected file with the same name.

Unable to open pinlist output file *name*

The Pinlist Generator was unable to create the output file *name*. Possible reasons for this failure are lack of sufficient disk space or a write-protected file with the same name.

Unable to open temporary file *name*

The Design Rule Check program or the Compatible Net List program was unable to open the drawing file *name*. It is possible that there is insufficient disk space for the program to open an additional file.

Unable to read drawing file *name*

The program was unable to read the drawing file *name*. It is possible that there was a disk access problem or the file was read-protected.

File *name* **does not exist**

In Design Rule Check, the program cannot find *name*. In the other programs, the file has not been properly specified or has not been created.

Glossary

This chapter describes terms that are specific to the Post package.

dcm

The drawing preprocessor uses two subprocesses, called local and dcm (or global), to generate a drawing connectivity model. Dcm reads the individual connectivity data appended by local to each drawing file, and generates a drawing connectivity model for the entire design.

design tree

A **design tree** is the hierarchical relationship of drawings in a design. The term takes its name from the branching relationship of the drawing sets when a structured design is created.

drawing connection model

A **drawing connection model** is a binary file representing the design created by establishing connectivity between drawing files.

drawing file

A **drawing file** is a single sheet that is output by FutureNet. This sheet may or may not comprise the entire design. A drawing file can only be used in one drawing set in a given design.

drawing number

The **drawing number** is the unique number assigned automatically to each drawing file in a design. Drawing numbers begin at one and increase sequentially by one for each drawing file in a design. In structured designs, root level drawings are assigned numbers first.

drawing occurrence count

The **drawing occurrence count** identifies a particular occurrence of a drawing file in a design. It consists of a drawing number followed by an occurrence count, separated by a hyphen. Each drawing is assigned a unique number. The occurrence count begins at one and is increased sequentially by one each time a particular drawing file is used in the design.

drawing path

The **drawing path** is an ordered list of the drawings and block symbols traversed in structured designs from a root drawing file to a lower level drawing file. It consists of one or more groups of filenames, functional block symbol reference numbers, and functional block circuit designators.

drawing reference number

The **drawing reference number** is a unique number assigned to each drawing occurrence in the design. A drawing reference number is used to make internally generated names unique.

drawing set

A **drawing set** consists of one or more related drawing files. Drawing sets contain either root level drawings or lower level drawings. Drawing sets that contain two or more drawing files will result in signal connectivity between the files when the Drawing Preprocessor is run. A drawing set can be used more than once in a design.

functional block symbol

A **functional block symbol** is a type of symbol used in structured designs to represent part of a circuit. The composition of this circuit is then detailed in the drawing set name inside the functional block.

global

The drawing preprocessor uses two subprocesses, called local and dcm (or global), to generate a drawing connectivity model. Dcm reads the individual connectivity data appended by local to each drawing file, and generates a drawing connectivity model for the entire design.

internally generated name

Internally generated names are names generated by the drawing connectivity modeler for any wires or buses that are not assigned a name by the user.

local

The drawing preprocessor uses two subprocesses, called local and dcm (or global), to generate a drawing connectivity model. Drawing files entered into the drawing preprocessor are first run through local, and individual connectivity data is generated and appended to the file. Dcm then reads the connectivity data in each drawing file, and generates a drawing connectivity model for the entire design.

lower level drawing set

A **lower level drawing set** consists of the drawings named in a functional block symbol.

root level drawing set

A **root level drawing set** consists of the drawings at the highest level of the design tree, that is, the drawings specified for the Drawing Preprocessor.

structured design

A **structured design** is a hierarchical design consisting of a root level drawing set and one or more lower level drawing sets.

substitution

Substitution refers to the process where signal names, bus names, and attributes in lower level drawings of the design tree are replaced by signal names, bus names, and attributes from higher level drawings.

Index

E

N

O

P

R

S

3 *Running PREPLOT*

PREPLOT is a utility that sorts the data in FutureNet drawing files, resulting in quicker output on vector-type devices (such as most plotters). Files created by this utility can be specified at the DPLOT **Drawing Name** prompt.

It is not mandatory that you run PREPLOT before plotting a drawing, although it is strongly recommended.

Note: ***preplot.exe*** *is useful only for drawings that are destined for plotter-type (vector) devices. Do not run PREPLOT on drawings for output to a printer-type (raster) device.*

Starting the Program

To start the PREPLOT program, type

preplot [*inputfile*] [*outputfile*]

input file — the name of a FutureNet drawing file. If you do not specify a filename extension, the program assumes the extension **.dwg**.

output file — the name you wish to assign to the sorted drawing file. Do not specify a filename extension; the program automatically adds the extension **.dpl**.

Program Prompts and Messages

Once entered, the PREPLOT program responds with:

```
FutureNet Plot Preprocessor
Version x.xx Copyright 1986 FutureNet
```

If you did not enter an input file on the command line, you are prompted:

```
Enter drawing file [.dwg]:
```

Enter the filename of your FutureNet drawing. PREPLOT assumes a file extension **.dwg**, unless otherwise specified.

If you did not enter an output file on the command line, you are prompted:

```
Enter  plot  file [inputfile.dpl]:
```

Enter the filename that you wish to assign to the output file. PREPLOT assumes an output file extension **.dpl**, unless otherwise specified. Also, DPLOT automatically checks for **.dpl** files if no extension is given for DPLOT drawing files.

4 *Troubleshooting*

DPLOT problems usually fall into one of two categories:

- Cable problems
- Computer configuration problems

These problems and their solutions are discussed below.

Troubleshooting for the PC

Cable Problems

The most common DPLOT problem is incorrect cabling. Cable problems may result from

- Using the wrong cable
- Plugging the cable into the wrong port on the back of your computer

Determining the Right Cable

The cables required by some printers and plotters are not always the standard parallel or serial cable that you can buy "off the shelf." See the appendixes for special cable requirements for Hewlett-Packard HP-GL plotters and Houston Instrument DMP plotters. If you use a different plotter, see the Driver Supplement manual that came with the DPLOT driver.

Determining the Right Port

Frequently, computers are configured with two serial ports (COM1/COM2) and two parallel ports (LPT1/LPT2). First determine which port is which. Then, look at the OUTPUT statement in your Device Definition **.dev** file to determine where DPLOT output is going. Some **.dev** files send output to a specific port (LPT1, LPT2, COM1, COM2) while others output to a special device driver (SER1, SER2, C9071, C9072).

If OUTPUT is sent to a specific port, connect your cable to that port.

If OUTPUT is sent to driver SER1 or C9071, connect your cable to COM1.

If OUTPUT is sent to driver SER2 or C9072, connect your cable to COM2.

Computer Configuration Problems (PC Only)

The Configuration Check Utility

DPLOT requires your computer to be configured in a specific way. The CONCHECK utility, included with your DPLOT package, displays your current system configuration so that you can locate problems and make any necessary adjustments.

When using CONCHECK, keep your DPLOT Driver Supplement(s) on hand for reference.

To start the CONCHECK program, change to the DPLOT directory of your fixed disk. Then enter **concheck**.

The program responds with

```
FutureNet Configuration Check Program
Version X.XXx  Copyright 1986 FutureNet
```

It then presents you with information on the following items, relevant to using DPLOT:

- DOS Version
- DOS Path
- Math Co-processor
- DPLOT Version
- Device Definition Files
- Serial Port Configuration
- Serial Port Drivers
- DEVICE Commands
- MODE Commands

DOS Version

This is the version of DOS installed on your machine. To use DPLOT, you must have DOS 3.10 or later. If not, upgrade your DOS before using DPLOT.

DOS Path

The DOS Path is the list of directories your computer searches to find programs that are not in the current directory. These directories are initially specified in the PATH statement of your **autoexec.bat** file.

If the directory DPLOT is not included in your DOS path, it must be added.

Math Co-Processor

Some DPLOT drivers require an 8087, 80287, 80387, or 80487 Math Co-processor chip to be installed in your system. If the message

```
Co-Processor NOT installed
```

appears, you may need to purchase the appropriate Co-processor for your computer and install it before using DPLOT.

Serial Port Drivers

Some serial output devices need special device drivers to communicate through the COM ports. For these devices, Data I/O supplies special device drivers SER1, SER2, C9071, and C9072. If your Device Definition **.dev** file specifies output to one of these device drivers, that driver must be installed.

If CONCHECK tells you that this driver is NOT installed, one of the following two situations has occurred:

1. You have not installed the driver.
2. DPLOT does not know where the driver is.

If you have not installed the driver, follow the installation instruction in your DPLOT Driver Supplement.

If the driver is installed, look at the following section on DEVICE Commands to determine where DPLOT is looking for the driver.

DEVICE Commands

Device drivers for DPLOT should be loaded automatically into your machine by the system configuration file, **config.sys**. This **CONCHECK** section lists the drivers that **config.sys** loads.

For Output To	This Line Should Appear
SER1	DEVICE=\DPLOT\SERIAL1.SYS
SER2	DEVICE=\DPLOT\SERIAL2.SYS
C9071	DEVICE=\DPLOT\SERCAL1.SYS
C9072	DEVICE=\DPLOT\SERCAL2.SYS

If the correct driver is not present, edit your config.sys file to include the driver and reboot your computer. See the *FutureNet Installation Guide.*

DPLOT Version

The line

```
DPLOT.EXE Version: XX.Xx
```

tells you which version of DPLOT you are using. If you need assistance from Data I/O, the customer support engineers will need this information.

Device Definition File

The names and locations of all Device Definition **.dev** files installed in the current directory or **DPLOT** directory (if any) are listed directly under the DPLOT version.

If no **.dev** files appear, follow the instructions in the DPLOT Driver Supplement for installing the Device Definition file for your specific printer or plotter.

Serial Port Configuration

Next **CONCHECK** displays the configuration for the serial ports, COM1 and COM2. This information specifies the communications parameters: baud rate, parity type, number of data bits, and number of stop bits.

If your output device is a serial printer or plotter, the parameters for the port your printer or plotter is connected to must be set up using a DOS **MODE** command, as specified in your DPLOT Driver Supplement.

Check if the **DEVICE** Command for your driver is present, and specified as shown in the previous paragraphs.

If not present, install your device drivers by following the instructions in your DPLOT Driver Supplement.

If present, but not specified correctly, edit your **config.sys** file using a text editor or word processor. Then reboot your system before running DPLOT.

If present and specified correctly, make sure that the appropriate driver file (serial1.sys, serial2.sys, sercal1.sys, and/or sercal2.sys) was copied into the **DPLOT** directory.

MODE Commands

When outputting to serial devices, a **MODE** command is sometimes needed in your **autoexec.bat** file to configure your serial ports. Check your DPLOT Driver Supplement to see if your output device requires a specially configured serial port. If so, add a **MODE** statement to **autoexec.bat** as specified in your DPLOT Driver Supplement.

If the proper **MODE** command already appears in your **autoexec.bat** file, but the COM ports are configured incorrectly, your COM ports may be altered by other programs. See the previous section, "Serial Port Configuration." To correct this problem, enter a new **MODE** command at a DOS prompt, prior to running DPLOT (or create a batch file that does this).

Installation Troubleshooting for the Sun

Determining the Right Cable

The cables required by some printers and plotters are not always the standard parallel or serial cable that you can buy "off the shelf." The DPLOT Driver Supplement for your printer or plotter identifies special cable requirements (if any). Make sure that you are using the correct cable for your printer or plotter.

Determining the Right Port

The ports used by DPLOT are serial port A and serial port B. The output is directed to serial port A by the name **/dev/ttya**; the output is directed to serial port B by the name **/dev/ttyb**. Be sure that the cable is connected to the correct serial port as determined by the **/dev/ttya** or **/dev/ttyb** statement in the **.dev** file you are using.

Configuring the Serial Port

Check the file **/etc/ttys** to see if the first character of the plotter port is a 0 or a 1. A value of 0 here disables the **getty** (login) process; if a 1 is present, change it to a 0. For example, if the file **/etc/ttys** contains **12ttya**, change it to **02ttya**.

A DPLOT Parameter Keywords

The following keywords and parameter values should be used when specifying DPLOT parameters on the command line or in the Settings File, **dplot.set**. All possible parameter values are not available with every output device.

For a complete discussion of DPLOT parameters and values, see Chapter 2.

Parameter/Keyword	**Possible Values**
Device name/device	Any available .dev file
Paper size/paper	NARROW, N, WIDE, W, A, B, C, D, E, A0, A1, A2, A3, A4, A5, *hzXvt*
Resolution/res	HIGH, LOW, H, L 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
Scaling/scale	TOFIT, NONE, FULL, A, B, C, D, E, *hzXvt*, percentage
Placement/place	LT, LC, LB, CT, CC, CB, RT, RC, RB
Aspect ratio/aspect	Percentage
Strip to print/strip	ALL, strip number
Options/opts	B, D, R, O

B Device Files

Each device file, **.dev**, contains the valid parameter values and default settings for a specific printer or plotter. This chapter gives a general description of device files. The supplemental driver information for your printer or plotter describes the specific contents of the .dev file you received.

You can edit these files, or create new ones, in order to alter the file's default settings or to configure the file so that DPLOT output is sent to another device. To create or edit device files, use any text editor capable of producing unformatted ASCII text.

Data Fields in the .dev File

Device files contain the following statement keywords and associated values:

- NAME
- NOTE (0 or more)
- DRIVER
- OUTPUT
- PAPER
- BORDER (not always present)
- DIRECT (not always present)
- CHARACTER (not always present)
- CIRCLE (not always present)
- DOT (not always present)
- BUS (not always present)
- ARC (not always present)
- ROLL/CUT/AUTO
- GRID

Each statement must be placed on a separate line. The values associated with the statements are used by DPLOT to determine program prompts, defaults, and the set of communications parameters needed to drive your printer or plotter. These values must be separated from the statement keyword by at least one blank space.

Name

This statement contains the name of your printer/plotter, for example,

NAME Houston Instruments DMP Plotters

Although not used by DPLOT, this required field serves to improve readability of the device file.

Note

This statement contains miscellaneous notes. A device file can contain up to eight NOTE statements. Data I/O includes the version of the device file in one of these statements as an aid to troubleshooting, for example,

NOTE Version 2.00A
NOTE Supports models 41, 42, 51, 52

Driver

This statement specifies the filename of the driver needed to control the printer or plotter. The filename extension **.drv** must be specified, for example,

DRIVER dmp.drv

Output

This statement specifies the system device where printer/plotter output is normally sent, for example,

OUTPUT ser1

The supplemental driver information identifies the devices that may be used with your specific printer or plotter.

Alternately, output may be sent to a file for later printing (see "Using Spooled Output" in Chapter 2), for example:

OUTPUT dplot.out

When outputting to a file, you may also use the special output character * (plus a unique filename extension) to assign the drawing filename, for example,

OUTPUT *.out

Using the above example, if the drawing file is named **dsample.dwg**, the output will be spooled to a file named **dsample.out**.

Paper

This statement specifies valid paper sizes (ie, NARROW, ARCH, C, A5) for your printer or plotter. Each value must be separated by a comma, for example,

PAPER E,D

DPLOT uses the first value in the statement as the default. Therefore, in the above example, **E** would be the DPLOT default paper size.

Border

This statement tells DPLOT what portion of the paper cannot be plotted. This is generally true on pinch roller-type plotters because the pen cannot draw on the paper in the area under the rollers. The border area for both dimensions must be specified in inches in the form

BORDER ***horiz x vert***

horiz is the sum of the margins along the long edge of the paper; ***vert*** is the sum of the margins along the short edge.

For example, the following specifies a 1-inch border around the entire sheet:

BORDER 2.0 x 2.0

This statement does not appear in the device file unless borders are required.

Character

This statement tells DPLOT to use the plotter's built-in character generator instead of DPLOT's fonts; there are no parameters.

CHARACTER

This statement is used with vector-type devices only, and does not appear in the device file when DPLOT's default fonts are to be used. If this statement does appear, the presence of the **DIRECT** statement is implied, whether specified or not. As a result, if you remove this statement from a device file, you must replace it with a **DIRECT** statement.

Circle

This statement uses the plotter's built-in circle generator to draw circles; there are no parameters.

CIRCLE

This statement is used with vector-type devices only, and does not appear in the device file when DPLOT's circle generator is to be used.

If this statement does appear, the presence of the **DIRECT** statement is implied, whether specified or not. If you remove a **CIRCLE** statement from a device file, you must replace it with a **DIRECT** statement.

Note: If the plot is rotated, or if the aspect ratio is altered, this statement is ignored in favor of DPLOT's circle generator.

Dot

This statement uses the plotter's built-in circle generator to draw connection dots; there are no parameters.

DOT

This statement is used with vector-type devices only, and does not appear in the device file when DPLOT's built-in circle generator is to be used.

If a DOT statement does appear, the presence of the **DIRECT** statement is implied, whether specified or not. If you remove a **DOT** statement from a device file, you must replace it with a **DIRECT** statement.

Note: If the plot is rotated, this statement is ignored in favor of DPLOT's circle generator.

Bus

This statement uses the plotter's built-in "rectangle fill" function to draw solid bus lines. This is useful in plotting scaled-up drawings, as DPLOT's standard bus lines (composed of three closely-spaced parallel lines) tend to separate at large scale sizes. The function does, however, take more plotting time than the standard DPLOT algorithm.

There are no parameters.

BUS

This statement is used with certain plotters only. The DPLOT Driver Supplement for your plotter describes this statement.

If this statement does appear, the presence of the **DIRECT** statement is implied, whether specified or not. As a result, if you remove the **BUS** statement from a device file, you must replace it with a **DIRECT** statement.

Note: If the plot is rotated, this statement is ignored in favor of DPLOT's circle generator.

Arc

This statement uses the plotter's built-in arc generator to draw arcs. There are no parameters.

ARC

This statement is used with certain vector-type devices only, and does not appear in the device file when DPLOT's arc generator is to be used.

If this statement does appear, the presence of the **DIRECT** statement is implied, whether specified or not. If you remove an ARC statement from a device file, you must replace it with a **DIRECT** statement.

Note: If the plot is rotated, or if the aspect ratio is altered, this statement is ignored in favor of DPLOT's arc generator.

Roll/Cut/Auto

The choice of one of these statements identifies the paper feed type: continuous, single cut sheet, or automatic single sheet feed, respectively. There are no parameters.

ROLL

indicates the output device uses either continuous roll or form fed paper. Therefore, DPLOT does not have to pause between sheets when printing multiple strips.

Grid

This statement specifies the LOW to HIGH resolution grid sizes available for your output device. There can be up to 10 values in this field, providing support for printers with up to 10 dot density resolutions. These values are separated by commas, and ordered from lowest to highest resolution, for example,

GRID 360,540,1080

When running DPLOT, the first value in the **GRID** statement (which is the default value) corresponds to the LOW resolution parameter; the last value is the HIGH resolution parameter.

The device file accompanying your DPLOT Driver provides the acceptable Grid values for that device. In general, the higher the Grid value, the more accurate the reproduction.

Device File Examples

The following pages contain examples of device files for the

- Epson FX, MX, and RX series printers
- CalComp 960 plotter
- QMS Laser printer
- Houston Instruments DMP plotter

These files are for illustration only, and may differ from the actual device files used for your printer or plotter.

EPSON

NAME	Epson MX, FX, RX printers
NOTE	Version 2.10A
DRIVER	\DPLOT\EPSON.DRV
OUTPUT	PRN
PAPER	W (Wide), N (Narrow)
ROLL	
GRID	360, 540

CAL960.DEV

NAME	CalComp 960 Plotter
NOTE	Version 2.10A
DRIVER	\DPLOT\CAL960.DRV
OUTPUT	SER1
PAPER	E, D, C, B, A
DIRECT	
CUT	
GRID	400

QMS.DEV

NAME	QMS Laser printers
NOTE	Version 2.10A
DRIVER	\DPLOT\QMS.DRV
OUTPUT	PRN
PAPER	A, B, N
AUTO	
GRID	300

DMP.DEV

NAME	DMP 40/50 Series Plotters
NOTE	Version 2.10B
NOTE	Supports models 41, 42, 51, 52
DRIVER	\DPLOT\DMP.DRV
OUTPUT	SER1
PAPER	D, C, E
BORDER	2.0 X 2.0
CHARACTER	
DOT	
CUT	
GRID	250

C Error Messages

The following is an alphabetical list of the error messages generated by DPLOT. Also included is an explanation of and corrective action for each message.

Arc vector not set in Driver, remove from .dev file.

The device driver is not set up to do built-in arc generation. Remove the ARC entry from the device file. Replace it with DIRECT if CHARACTER, CIRCLE, DOT, or BUS is not specified.

Bad jump address driver cannot be used as DIRECT.

The driver selected is for a raster-type device and cannot use the DIRECT option. Remove DIRECT, CHARACTER, CIRCLE, DOT, BUS, and ARC from the Device file.

Bus vector not set in Driver, remove from .dev file.

The device driver is not set up to do built-in box fill operations. Remove the BUS entry from the device file. Replace it with DIRECT if CHARACTER, CIRCLE, DOT, ARC, or BUS is not specified.

Cannot access specified Driver *drivername.*

The driver **.drv** file specified in the selected device file was not found on the system path.

1. Make sure the driver is on the system path.
2. Verify that the device file contains a valid driver name.

Cannot find *filename*.**dev.**

Check your entry and try again.

Cannot open option file.

The system-generated **.opt** file could not be created. Make sure there is space on the current file system.

Character vector not set in Driver, remove from .dev file.

The device driver is not set up to do built-in character generation. Remove the CHARACTER entry from the device file. Replace it with DIRECT if CIRCLE, DOT, BUS, or ARC is not specified.

Circle vector not set in Driver, remove from .dev file.	The device driver is not set up to do built-in circle generation. Remove the CIRCLE entry from the device file. Replace it with DIRECT if CHARACTER, DOT, BUS, or ARC is not specified.
Device file missing driver specification	A DRIVER value is not present in the device file. Add the appropriate driver to the file.
Device file missing grid specification.	A GRID value is not present in the device file. Add an appropriate grid value to the file.
Device file missing output specification.	A device name or filename is missing from the OUTPUT field in the device file. Add a valid name for OUTPUT.
Device file missing roll, cut or auto specification.	The ROLL, CUT or AUTO entry is missing from the paper feed field in the device file. Add ROLL, CUT or AUTO to the feed field.
Dot vector not set in Driver, remove from .dev file.	The device driver is not set up to do built-in dot generation. Remove the **DOT** entry from device file. Replace it with DIRECT if CHARACTER, CIRCLE, BUS, or ARC is not specified.
Fatal memory overflow error.	The program memory is full. The program cancels the command being executed. Make sure that your system has the required 256K of memory. If it does and you still get this message, contact Data I/O.
File filename is not a drawing file.	The **.dwg** filename is not a FutureNet drawing file. Check your entry and try again.
Font vector not set in Driver, remove from .dev file.	The device driver is not set up to do built-in character generation. Remove the CHARACTER entry from the device file. Replace it with DIRECT if CIRCLE, DOT, BUS, or ARC is not specified.
Invalid BORDER entry in .dev file.	The BORDER entry was not specified properly. Respecify the BORDER entry in the form *horiz/vert* where *horiz* is the sum of the left and right border, and *vert* is the sum of the top and bottom border.
Invalid entry.	A parameter value that you specified is not valid for your output device. Enter a different value, or add the new value to the device file.
Invalid use of +	When entering multiple drawing names, you used the plus sign incorrectly. Check your entry and try again. There may be no blank spaces between the filenames and the plus signs.
Multiple drawing files not connected by +	When entering multiple drawing names, you did not separate the names with the plus sign. Check your entry and try again.

Paper, grid overflow!! Lower grid value in device file.

The number of dots required along the X axis has exceeded 32,768, or the number of dots required along the Y axis has exceeded 16,384. Since the resolution is already at its lowest when this message occurs, there are two possible solutions:

- Change to a smaller paper size.
- Decrease the GRID value in the device file.

Specified Driver is not an .exe file.

The driver specified in the device file cannot be loaded as a driver. Check a for bad entry in the device file. Recopy the driver from the backup floppy.

Resolution, paper, grid overflow!! Try lower resolution.

The number of dots required along the X axis has exceeded 32,768, or the number of dots required along the Y axis has exceeded 16,384. The three possible solutions are

- Change to a lower resolution, if available.
- Change to a smaller paper size.
- Decrease the GRID value in the device file.

Too many characters in command stream.

The combination of the settings file and the command line has too many characters. Remove duplicate keyword entries.

Too many GRID values in device file.

There are more than 10 (0 to 9) values in the GRID entry of the device file. Remove the inappropriate entries.

Unable to create intermediate file.

Your file system is full. Make sure there is enough space on your file system and try again.

Unable to open device file.

The device file cannot be opened. Check your entry and try again.

Unknown input parameter.

A parameter on the command line is not recognized by DPLOT or is invalid. Check your entry and try again.

D Hewlett-Packard HP-GL Plotters

This appendix describes the hardware requirements, installation instructions and device file contents for the DPLOT program. These drivers enable DPLOT to control the Hewlett-Packard 7400 and 7500 series plotters, as well as devices that support the Hewlett-Packard Graphics Language (HP-GL).

Hardware Requirements

To use DPLOT with an HP-GL plotter, you must have

- An asynchronous communications adapter in your computer.
- An HP 7400 series, 7500 series, or other HP-GL compatible plotter.
- A specially configured RS-232 cable. The wiring diagram for this cable is described later in this appendix.

Please check your hardware documentation for any additional hardware requirements, installation procedures, or pre-operational setup procedures.

Installing the Drivers

HP-GL Files

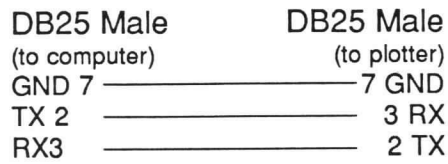
The Hewlett-Packard HP-GL Plotter requires these files:

hp1.drv	This driver provides the control codes to operate HP-GL plotters with A- and B-size paper.
hp2.drv	This driver provides the control codes to operate HP-GL plotters with C-, D-, and E-size paper.
hp1.dev	This device file specifies valid user options and DPLOT default values for HP-GL plotters that use A and B-size paper.
hp2.dev	This device file specifies valid user options and DPLOT default values for HP-GL plotters that use C, D, and E-size paper.
serial1.sys (PC only)	This special serial port driver controls the flow of data through your system's primary asynchronous communications port (COM1).
serial2.sys (PC only)	This special serial port driver controls the flow of data through your secondary asynchronous communications port (COM2).
hpinit.fn1	This user-configurable file contains the initialization sequence that is sent to the plotter prior to sending drawing data (for the HP 7400 series).
hpterm.fn1	This user-configurable file contains the termination sequence that is sent to the plotter after all drawing data has been sent (for the HP 7400 series).
hpinit.fn2	This user-configurable file contains the initialization sequence that is sent to the plotter prior to sending drawing data (for the HP 7500 series).
hpterm.fn2	This user-configurable file contains the termination sequence that is sent to the plotter after all drawing data has been sent (for the HP 7500 series).

Note: Choose the **hpinit** *and* **hpterm** *program sets and copy them to* **hpinit.fn** *and hpterm.fn, respectively.*

Sun Cabling to the HP 7400 and 7500 Series Plotters

The Hewlett-Packard plotters require the following cable:



When you build the cable, label each end appropriately (either "To Computer" or "To Plotter").

PC Cabling to the HP 7400 and 7500 Series Plotters

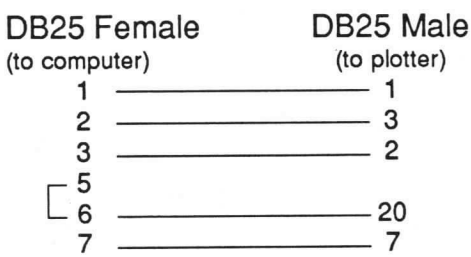
There are three types of asynchronous communications adapters used with IBM-compatible personal computers:

- A DB25 male 25 pin connector
- A DB25 female 25 pin connector
- A DB9 male 9 pin connector (AT-compatible)

Each adapter requires a different type of connecting cable.

Male DB25 Adapters

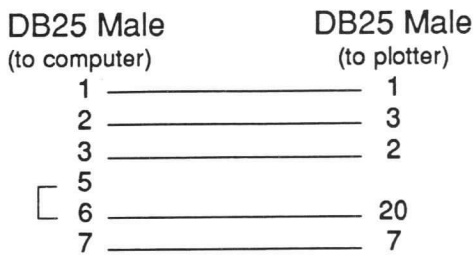
Adapters that use a male DB25 connector require a female connector on the computer end of the cable.



This cable is asymmetrical. Attach the female DB25 to the asynchronous serial connector of your computer. Attach the male DB25 connector to the plotter.

Female DB25 Adapters

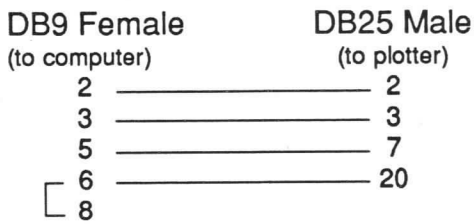
Adapters that use a female DB25 connector require the following cable:



When you build the cable, label each end appropriately (either "To Computer" or "To Plotter").

Male DB9 Adapters

Parallel/Serial cards for PCs have a DB9 connector for the serial port and require the following cable:



This cable is asymmetrical. Attach the 9-pin DB9 connector to the asynchronous communications port on your computer. Attach the 25-pin DB25 connector to the plotter.

HP Device Files

The device file for HP-GL plotters that accept A- and B-size paper is called **hp1.dev**. The device file for HP-GL plotters that accept C-, D-, and E-size paper is called **hp2.dev**. These files contain plotter-relevant information for the HP 7400 and 7500 series-compatible plotters. The format of these files is discussed in Appendix B.

Under most circumstances, these files require no user modification. However, if you wish to change the entries in these files, use an ASCII text editor to modify the information.

hp1.dev

A listing of **hp1.dev** is shown below.

```
NAME            HP1 for A and B size HP Plotters
NOTE            Version X.XXx
DRIVER (PC)     \dplot\hp1.drv
       (Sun)    hp1.drv
OUTPUT (PC)     ser1
       (Sun)    /dev/ttyb
PAPER           B, A
BORDER          1.0X0.85
CHARACTER
CIRCLE
ARC
CUT
GRID            1016
```

NAME Identifies the intended output device as HP-GL plotters that support A- and B-size paper.

NOTE Contains the device file version number.

DRIVER (PC) Specifies the DOS directory in which the DPLOT driver is located (\DPLOT) for the PC.

(Sun) Specifies the name of the DPLOT driver that controls the HP plotter (hp1.drv) for the Sun.

OUTPUT (PC) Specifies that SER1 is the DOS device to which output is sent to the PC. (This is a special driver that outputs data through the primary serial port (COM1). If you want output sent to the alternate serial port (COM2), you must edit the device file, substituting SER2 for SER1.

(Sun) Specifies that **/dev/ttyb** is the device to which output is sent to the Sun. (This is a special driver that outputs data through the serial port. If you want output sent to serial port **/dev/ttya**, you must edit the device file, substituting **/dev/ttya** for **/dev/ttyb.**)

PAPER Supplies DPLOT with the allowable paper sizes for these HP plotters: A and B. Because it is specified first, B-size is used as the default size.

BORDER Specifies that area of the paper that cannot be plotted upon, due to the placement of the paper rollers. 1.0X0.85 specifies a border of .5 inches on the horizontal edges and .425 inches on the vertical edges.

CHARACTER Tells DPLOT to use the HP plotter's internal character generator, rather than the DPLOT fonts.

CIRCLE Tells DPLOT to use the HP plotter's internal circle generator, rather than the DPLOT circle drawing routine. This command is ignored if the plot is rotated or if the aspect ratio of the plot is not 100% (1:1).

ARC Tells DPLOT to use the HP plotter's internal arc generator, rather than the DPLOT arc drawing routine. This command is ignored if the plot is rotated or if the aspect ratio of the plot is not 100% (1:1).

CUT Tells DPLOT that the HP plotter takes cut sheet paper, rather than continuous roll or form fed paper.

GRID Specifies the resolution grid size for the vectors that are output to the plotter.

Note: If you significantly scale up your plots (and do not use rotation), adding the BUS command to the end of this device file improves the look of your bus lines. This command tells DPLOT to draw bus lines using the HP plotter's rectangle fill command, rather than drawing three closely-spaced parallel lines. Please note that plotting is slowed down when drawing lines use BUS.

hp2.dev

Below is a listing of **hp2.dev**:

```
NAME            HP2 for C, D, and E size HP Plotters
NOTE            Version X.XXx
DRIVER  (PC)    \DPLOT\hp2.drv
        (Sun)   hp2.drv
OUTPUT  (PC)    ser1
        (Sun)   /dev/ttyb
PAPER           D, E, C, B, A
BORDER          0.7X2.75
CHARACTER
CIRCLE
ARC
CUT
GRID            254
```

NAME Identifies the intended output device as HP-GL plotters that support C-, D-, and E-size paper.

NOTE Contains the device file version number.

DRIVER (PC) Specifies the DOS directory in which the DPLOT driver is located (\DPLOT) for the PC.

(Sun) Specifies the name of the DPLOT driver that controls the HP plotter (hp2.drv) for the Sun.

OUTPUT (PC) Specifies that SER1 is the DOS device to which output is sent for the PC. (This is a special driver that outputs data through the primary serial port (COM1). If you want output sent to the alternate serial prot (COM2), you must edit the device file, substituting **SER2** for **SER1**.)

(Sun) Specifies that **/dev/ttyb** is the device to which output is sent for the Sun. (This is a special driver that outputs data through the serial port. If you want output sent to the alternate serial port **/dev/ttya**, you must edit the device file, substituting **/dev/ttya** for **/dev/ttyb.)**

PAPER	Supplies DPLOT with the allowable paper sizes for these HP plotters: A, B, C, D, and E. Because it is specified first, D-size is used as the default size.
BORDER	Specifies that area of the paper that cannot be plotted upon, due to the placement of the paper rollers. 0.7X2.75 specifies a border of .35 inches on each horizontal edge and 1.375 inches on each vertical edge.
CHARACTER	Tells DPLOT to use the HP plotter's internal character generator, rather than the DPLOT fonts.
CIRCLE	Tells DPLOT to use the HP plotter's internal circle generator, rather than the DPLOT circle drawing routine. This command is ignored if the plot is rotated or if the aspect ratio of the plot is not 100% (1:1).
ARC	Tells DPLOT to use the HP plotter's internal arc generator, rather than the DPLOT arc drawing routine. This command is ignored if the plot is rotated or if the aspect ratio of the plot is not 100% (1:1).
CUT	Tells DPLOT that the HP plotter takes cut sheet paper, rather than continuous roll or form fed paper.
GRID	Specifies the resolution grid size for the vectors that are output to the plotter.

Note: If you significantly scale up your plots (and do not use rotation), adding the BUS command to the end of this device file improves the look of your bus lines. This command tells DPLOT to draw bus lines using the HP plotter's rectangle fill command, rather than drawing three closely-spaced parallel lines. Please note that plotting is slowed down when drawing lines use BUS.

PC Initialization and Termination Files

Two special files are used by DPLOT to initialize your printer before sending your drawing data, and to reset your plotter after the plot is finished. These files are **hpinit.fn** and **hpterm.fn**. Two versions of these files are installed in the DPLOT directory. **hpinit.fn1** and **hpterm.fn1** support the 7400 Series of plotters. **hpinit.fn2** and **hpterm.fn2** support the 7500 Series. Copy the appropriate set to **hpinit.fn** and **hpterm.fn**.

hpinit.fn

This file contains an initialization sequence for your plotter consisting of

- Hardware handshaking information
- Plotter initialization information
- Scaling information

The contents of this file are sent to the plotter prior to sending the drawing data.

hpterm.fn

This file contains a termination sequence for your plotter consisting of

- Plotter reset information
- Frame advance information (for roll plotters)

The contents of this file are sent to the plotter after all drawing data has been plotted.

Customizing the Files

The files are in ASCII format, but because they contain non-printable control codes, they are not reproduced here. If desired, they can be user-modified using a text editor or word processor (in non-document or unformatted mode) to instruct your plotter to automatically perform additional functions.

CAUTION: ***Although an incorrectly altered initialization or termination file is unlikely to damage your plotter, it can cause your plotter to behave erratically, producing undesirable results. Therefore, a backup copy is strongly recommended when altering these files.***

You may keep two (or more) versions of these files by placing them in separate directories on your hard disk. When plotting, DPLOT first looks for these files in the current directory. If not found, it looks for the file in the DPLOT directory. If not found in either directory, DPLOT uses its internal defaults.

For information on the codes required to control your plotter, consult the *Hewlett Packard Interfacing and Programming Manual.*

E Houston Instrument DMP Plotters

This appendix describes the hardware requirements and device file contents for the DPLOT driver which enables DPLOT to control the Houston Instrument DMP Plotter models 41, 42, 51 and 52.

Hardware Requirements

To use DPLOT with a DMP plotter, you must have

- An asynchronous communications adapter in your computer
- A Houston Instrument DMP-41, 42, 51 or 52 plotter
- A specially configured RS-232 cable. The wiring diagram for this cable is described later in this appendix.

Please check your hardware documentation for any additional hardware requirements, installation procedures, or pre-operational setup procedures.

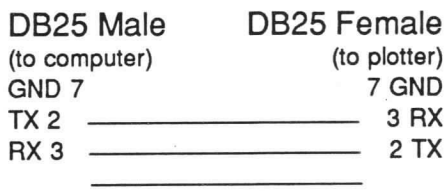
Installing the Drivers

The Houston Instrument DMP Plotter requires these files:

dmp.drv	Provides DPLOT with the control codes that operate the Houston Instrument DMP Plotter.
dmp.dev	The device file, used to specify valid user options and DPLOT default values.
serial1.sys (PC only)	A special serial port driver, used to control the flow of data through your system's primary asynchronous communications port (COM1).
serial2.sys (PC only)	A special serial port driver, used to control the flow of data through your secondary asynchronous communications port (COM2).

Sun Cabling to the DMP Plotter

The Sun serial adapter cable uses a female DB25 connector and requires the following cable.



The Type B cable is asymmetrical. Attach the male DB25 connector to the asynchronous serial connector of your computer. Attach the female DB25 connector to the plotter.

PC Cabling to the DMP Plotter

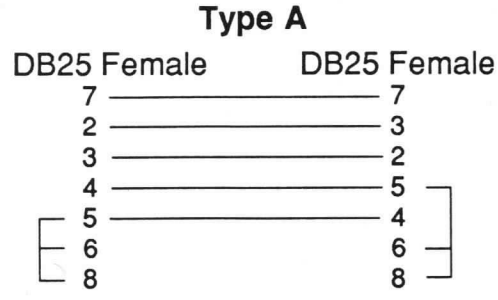
There are three types of asynchronous communications adapters used with IBM-compatible personal computers:

- A DB25 male 25 pin connector
- A DB25 female 25 pin connector
- A DB9 male 9 pin connector (AT-compatible)

Each adapter requires a different type of connecting cable.

Male DB25 Adapters

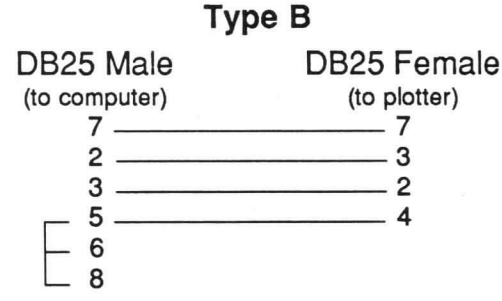
Adapters that use a male DB25 connector require a Type A cable:



This cable is symmetrical. Attach either end to the plotter, and the other end to the asynchronous communications adapter on your computer.

Female DB25 Adapters

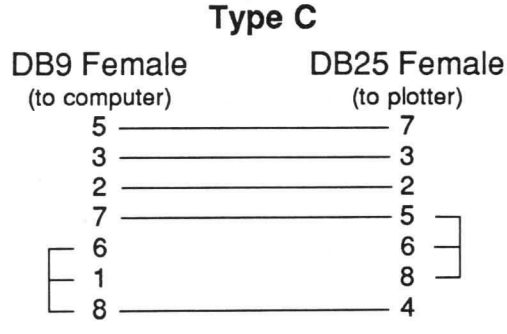
Adapters that use a female DB25 connector require a Type B cable:



The Type B cable is asymmetrical. Attach the male DB25 connector to the asynchronous serial connector of your computer. Attach the female DB25 connector to the plotter.

Male DB9 Adapters

AT-compatible Parallel/Serial cards have a DB9 connector for the serial port and, thus, require a Type C cable:



The Type C cable is asymmetrical. Attach the nine pin DB9 connector to the asynchronous communications port of your computer. Attach the 25 pin DB25 connector to the plotter.

DMP Device File

The device file for the DMP plotters is called **dmp.dev**. This file (included with this driver package) contains plotter-relevant information for the DMP plotter models 41, 42, 51 and 52. The format of this file is discussed in Appendix B.

Under most circumstances, this file requires no user modification. However, if you wish to change the entries in this file, use a text editor or a word processor in unformatted mode to add the new information.

Below is a listing of **dmp.dev:**

```
NAME            DMP 40/50 Series Plotters
NOTE            Version X.XXx
                Supports models 41, 42, 51, 52
DRIVER  (PC)    \dplot\dmp.drv
        (Sun)   dmp.drv
OUTPUT  (PC)    ser1
        (Sun)   /dev/ttyb
PAPER           D,C,E
BORDER          2.0X2.0
CHARACTER
DOT
CUT
GRID            250
```

NAME Identifies the intended output device as DMP plotters.

NOTE Contains the device file version number and the DMP plotter models supported by this version.

DRIVER (PC) Specifies the DOS directory where that driver is located (\dplot) for the PC.

(Sun) Specifies the name of the DPLOT driver that controls the DMP plotter (dmp.drv) for the Sun.

OUTPUT (PC) Specifies that SER1 is the DOS device to which plotting data is sent for the PC. (This is a special driver that outputs data through the primary serial port (COM1), If you want output sent to the secondary serial port (COM2), you must edit the device file, substituting **SER2** for **SER1**.

(SUN) Specifies that **/dev/ttyb** is the DOS device to which plotting data is sent for the Sun. (This is a special driver that outputs data through serial port A. If you want output sent to serial port A, you must edit the device file, substituting **/dev/ttya** for **/dev/ttyb**.)

PAPER Supplies DPLOT with the allowable paper sizes for DMP plotters: C, D, and E. Because it is specified first, D-size is used as the default size.

BORDER Specifies that area of the paper that cannot be plotted upon, due to the placement of the paper rollers. 2.0X2.0 specifies a border of 1 inch on each edge.

CHARACTER	Tells DPLOT to use the DMP plotter's internal character generator, rather than the DPLOT fonts. This occurs when plotting non-rotated drawings only. When plotting rotated drawings, the standard DPLOT fonts are used.
DOT	Tells DPLOT to use the DMP plotter's internal circle generator to draw connection dots, rather than the DPLOT connection dot generator. This occurs when plotting non-rotated drawings only. When plotting rotated drawings, the standard DPLOT dot generator is used.
CUT	Tells DPLOT that the DMP plotter takes cut sheet paper, rather than continuous roll or form fed paper.
GRID	Specifies the resolution grid size for the vectors that are output to the plotter.

FutureNet®

Schematic Designer

DPLOT User Manual

DATA I/O Corporation

096-0005-003

August 1991 096-0005-003

Data I/O Corporation
10525 Willows Road N.E., P.O. Box 97046
Redmond, Washington 98073-9746 USA
(206) 881-6444

Acknowledgments:

Table of Contents

1 Introduction to DPLOT

The FutureNet® DPLOT program is an enhanced alternative to the FutureNet PRINT feature. DPLOT allows you to output FutureNet drawings to a variety of printers and plotters, and provides full control over

- Paper size
- Resolution (printer dot density)
- Scale
- Aspect ratio
- Positioning on the page
- Inclusion/suppression of alpha field boundaries, attributes, and symbol reference numbers.

In addition, you can modify many DPLOT default settings so that you can custom configure DPLOT for your specific needs.

If you purchased a DPLOT package for use with plotters (or other vector output devices) you also received the PREPLOT utility. This utility presorts the drawing data, resulting in improved plotting speeds.

2 *Running DPLOT*

Starting the Program

To begin the DPLOT process, enter **dplot** on the command line. The program responds with

```
FutureNet DPlot Program
Version x.xx Copyright 1985 FutureNet
```

You are then prompted to enter the following parameters:

```
Drawing Name
Device Name
Paper Size
Resolution
Scaling
Placement
Aspect Ratio
Strip to Print
Other Options
```

At each prompt, you may

- Press [↵] to accept the DPLOT default value.
- Enter a new value.
- Press [?] to see a help message.
- Press [Esc] to return to the previous prompt.
- Press [;] to accept all remaining defaults.

If you are using a PC, you may also

- Enter [Ctrl] - [Break] to exit the program.
- Enter DOS to get a DOS shell. To return from the DOS shell to DPLOT, type **exit**.

DPLOT Parameters

Drawing Name

You are prompted

```
Please enter drawing file [.DWG]: (PC)
```

```
Please enter drawing file [.dwg]: (SUN)
```

Enter the name of the FutureNet drawing file you wish to print/plot. If you enter a filename without specifying a filename extension, DPLOT first checks for a file of that name with a **.dpl** extension. (**.dpl** files are created by the Preplot Sorting Utility described in Chapter 3.) If the **.dpl** extension is not found, DPLOT then looks for a file with the default FutureNet drawing file extension, **.dwg**.

To print more than one drawing at a time, separate the drawing names with a plus sign (+). For example, to specify two drawing files, **file1.dwg** and **file2.dwg**, you would enter **file1+file2.** When specifying multiple files, each is printed on its own sheet of paper.

Pressing ↵ at this point will exit DPLOT and return you to the operating system.

---

*Note: If a drawing previously run through the Preplot Sorting Utility is updated but not rerun through Preplot, the **.dpl** file will represent the old drawing and will plot if DPLOT is run before Preplot.*

Device Name

You are prompted

```
Please enter device name [\DPLOT\*.dev]: (PC)
```

```
Please enter device name [*.dev]: (SUN)
```

Enter the device filename you wish to use for this print/plot. This is the **.dev** file that tells DPLOT what printer or plotter you are using, what port it's connected to, and what options are available. To find out the Device File contents for your particular plotter or printer, consult the supplemental driver information for that device.

If you specify a filename without a filename extension, the program assumes **.dev**. If you do not specify a path, the program looks for that file first in the current directory. If the device file is not in the current directory, then the program looks for the file in either

- The DPLOT directory (PC), or
- The directory from which DPLOT was run (SUN)

Listing Available Device Names

If you do not know the name of the device file that you want to use, enter a device name of

***path**.dev** (PC)

***path*/*.dev** (SUN)

where *path* is the path name to the directory containing your device files.

DPLOT then searches the specified directory. If only one device file is found, DPLOT uses it. If more than one is found, DPLOT lists all files and asks you to select one.

On the Sun, if you press ↵ without specifying anything in response to the Device name prompt, DPLOT presents a list of device files that reside in the directory from which DPLOT was run.

Redirecting Output

The **.dev** files specify the system device to which the output will be sent. To send your output to a file (for spooling), or to a different device, enter your Device Name as follows:

***devicename*=output**

where device name is the name of the desired .dev file and output is the desired device or filename. For example, if your device file was **dmp.dev** and you wished to redirect output to **ser2**, you would enter:

dmp=ser2.

The asterisk wildcard character *, followed by a unique filename extension, assigns the drawing filename to the output file. For example, if the drawing filename is **dsample.dwg** and the device file you want to use is **hp2.dev**, assign the name **dsample.out** to the output file by entering

hp2=*.out

For more information on spooling output to a file for later printing/plotting, see "Using Spooled Output" in this chapter.

Paper Size

You are prompted with

```
Please enter paper size [size]:
```

The paper size refers to your choice of paper size for this drawing. The default value [size] is determined by the contents of the selected **.dev file**. Paper size may be specified as follows:

NARROW (N)	Largest possible paper size for 80 Column printers.
WIDE (W)	Largest possible paper size for 132 Column printers.
A through E	American plotter paper sizes A, B, C, D, and E.
A0 through A5	European plotter paper sizes A0, A1, A2, A3, A4, and A5.
*hz*X*vt* [M][+][-]	Horizontal and vertical dimensions, where *hz* is the width of the paper and *vt* is the height of the paper.

M is an optional field indicating that the specified values are in millimeters, rather than inches.

+ overrides the **.dev** file default paper type to specify continuous form or roll-type paper.

- overrides the **.dev** file default paper type to specify single cut sheet paper.

Resolution (Dot Density)

You are prompted

```
Please enter resolution [LOW]:
```

Resolution refers to the horizontal (X-axis) and vertical (Y-axis) dot density of your output device. These options are listed below.

LOW (default)	The lowest available resolution, as specified in the **.dev** file.
HIGH	The highest available resolution, as specified in the **.dev** file.
0 through 9	The **.dev** file **GRID** statement identifies the possible resolutions for your printer or plotter. Entering 0 selects the first value specified in that statement, 1 selects the second, etc. Note that not all drivers have 10 levels of resolution.

Appendix B describes the format of the GRID statement. For resolution information specific to your printer or plotter, consult the supplemental driver information for that device.

Note: The largest number of points along the X-axis that DPLOT can output is 32,767; the largest along the Y-axis is 16,383. Because higher resolutions mean more points per inch, it may not be possible to print large drawings (e.g. E size) at your output device's high resolution setting.

Scaling

You are prompted

```
Please enter scale [TOFIT]:
```

Scaling can be specified as follows:

TOFIT (default)	The drawing will be scaled down to fit on the paper (if necessary) but not scaled up.
NONE	No scaling.
FULL	The drawing is scaled either up or down (as necessary) to fit on the paper. Aspect ratio is maintained.
FILL	The drawing size is stretched to fit the full size of the paper (minus any specified border area). Aspect Ratio is not maintained. Values assigned at the "Aspect Ratio" prompt (described later in this section) are ignored.
A through E	The drawing is scaled to fit on an A, B, C, D, or E sized sheet with no border. If your printer/plotter cannot print to the edge of the paper and you have not made allowances in your drawing, these options produce undesirable results.

***hzXvt* [M]**	Horizontal and vertical dimensions, where ***hz*** is the width of the drawing and ***vt*** is the height of the drawing. **M**, if used, indicates that the specified values are in millimeters instead of inches.
percentage	A percentage value, either absolute, as in 100% or 75%, or relative to 100%, as in +10% or -25%. The percent sign (%) is optional, but relative values must be preceded by a plus (+) or minus (-) sign.

Placement

You are prompted

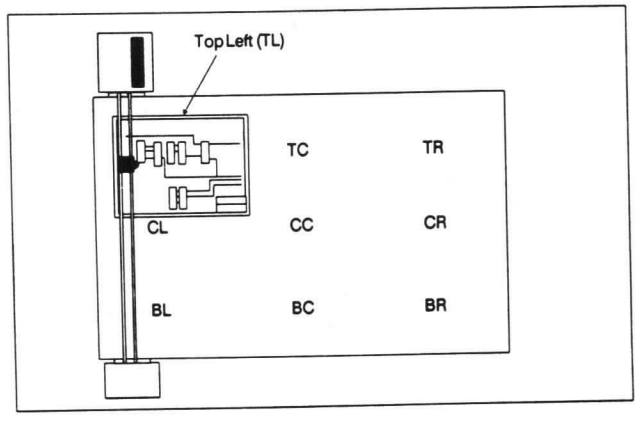
```
Please enter placement [CC]:
```

Placement is the combination of horizontal and vertical values that tell DPLOT where to place the drawing on the paper. These values assume that the left edge of the paper is the area loaded into your printer/plotter. The options are as follows:

TL	Top Left corner (shown in Figure 2-1)
CL	Centered top to bottom, left side
BL	Bottom left corner
TC	Top half of sheet, centered left and right
CC	Center of sheet (default)
BC	Bottom half of sheet, centered left and right
TR	Top right corner
CR	Centered top to bottom, right side
BR	Bottom right corner

Append / to the value to rotate the plot 90 degrees. If you wish to use the default placement with rotation, you must enter the two-letter code followed by a /. A / alone is erroneous input.

Figure 2-1
Drawing Placement Locations



Aspect Ratio

The DPLOT prompt for aspect ratio is

```
Please enter aspect ratio [100%]:
```

Normally, the height and width of drawings are kept in proportion to each other, even when scaled to different sizes. Changing the Aspect Ratio allows you to alter these proportions so that drawings will fill an entire sheet.

When altering the Aspect Ratio, the width of the drawing does not change; only the height is affected. This change is specified as a percentage of the original size.

This percentage can be either positive or negative. It can be an absolute value, as in 100% or 75%, or it can be a value relative to 100%, as in +10% or -25%. The percent sign % is optional, but relative values must always be preceded by a plus (+) or minus (-) sign.

After the prompt, either enter a percentage value, or press ↵ to assign 100%. (100% means there is no adjustment to the X axis data.)

---

*Note: If **FILL** was selected as a scaling option, the Aspect Ratio parameter is ignored.*

Strip to Print

You are prompted

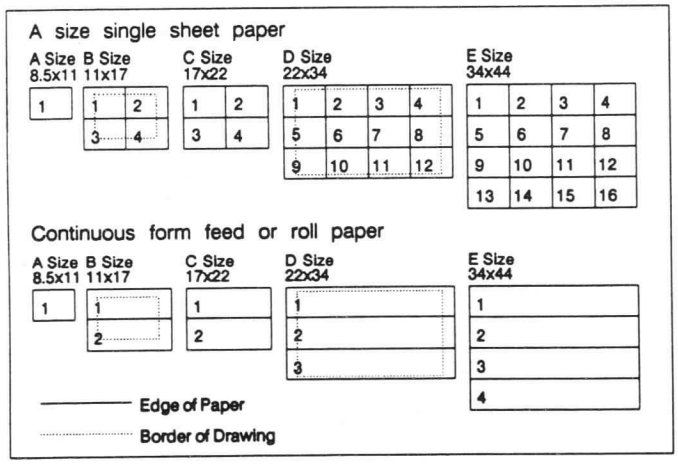
```
Please enter strip number [ALL]:
```

Drawings sometimes require larger paper sizes than many printers or plotters can accommodate. In these instances, DPLOT divides the drawing into a set of horizontal strips, which may be pasted together later to form the full-size drawing. For printers that use single cut sheet paper, each strip is subdivided into individual sheets.

As a default, DPLOT prints all strips that make up your drawing. To accept this default, press ↵ .

To print a specific strip only, determine the strip number using the diagram below, and then enter that number at the prompt.

Figure 2-2
Strip Number Determination Diagram Options



Options

You are prompted

```
Please enter sym ref nums (R), boundary (B), disp
attribute (D), overscore (O)[ ]:
```

Any, all, or none of the following may be specified:

R Print/plot symbol reference numbers.

B Print/plot alphanumeric field boundaries.

D Print/plot the display attribute of the alphanumeric fields instead of the actual text.

O Print/plot overscores down one dot unit on the plot.

Alternate Methods of Parameter Entry

In the previous sections of this chapter, you learned how to enter DPLOT parameters by responding to a set of prompts. This section discusses two alternative methods of entering parameters that can make it easier to use the program. They are as follows:

- Customize DPLOT (using a Settings file) so that you are not prompted for parameters that rarely change.
- Type parameters directly on the command line.

Customizing DPLOT With a Settings File

Many users consistently give the same response to one or more of the DPLOT prompts. Through the use of a Settings File, you can customize DPLOT so that you do not have to be prompted for these items.

You may create a file named **dplot.set**, which stores standard responses to prompts. DPLOT looks for this file first in the current directory, and then in the directories listed in the system path (in the order specified). DPLOT reads the contents of this file, if found, and then prompts you for only those parameters not specified by the file.

The Settings File may contain any or all parameters except Drawing Name. You must manually enter the drawing name each time you use DPLOT.

Create your Settings file **dplot.set** using an ASCII text editor.

The syntax for parameters in **dplot.set** is: ***keyword=value***. The parameters and corresponding **dplot.set** keywords are:

Parameter	**Keyword**
Device name	DEVICE
Paper size	PAPER
Resolution	RES
Scaling	SCALE
Placement	PLACE
Aspect ratio	ASPECT
Strip to print	STRIP
Options	OPTS

Each **keyword=value** statement must be on its own line and should contain no blank spaces. Be careful to spell keywords correctly to avoid misinterpretation.

The value assigned to **device** may be the name of any available device **.dev** file. Valid values for the remaining keywords are determined by the contents of the device file that you choose. For a list of these values, consult the supplemental driver information for that device.

The following **dplot.set** file would bypass all program prompts except Drawing Name:

```
DEVICE=hp2.dev
PAPER=b
RES=high
SCALE=full
PLACE=cc
ASPECT=100
STRIP=all
OPTS=
```

Thus, with this file, all you would need to do is enter the command **dplot** and then specify a drawing name when prompted.

Entering Parameters on the Command Line

If you want to create a drawing using parameters with values that differ from those in **dplot.set**, you can enter those parameters directly on the command line, overriding the entries in **dplot.set**. To enter parameters from the command line, use the following form:

dplot ***[drawing] [,parms][;]***

drawing — the drawing name

parms — one or more DPLOT parameters, specified in the form keyword=value. Multiple parameters may be entered in any order, and must be separated by commas or single spaces. No blank spaces may separate a keyword and value, only the equals sign. Parameters that are not specified default to the value assigned in **dplot.set**. See Appendix A for a list of valid keywords and parameter values.

; — the semicolon instructs DPLOT to use the system default value for any parameter not specified on the command line or in the **dplot.set** file. Otherwise, you are prompted for unspecified parameters.

The following DPLOT example uses a FutureNet drawing named **myfile** and command line parameters:

dplot myfile,PAPER=b,PLACE=tl/,OPTS=b

Entering Command Line Parameters (Sun Only)

Due to a characteristic of the UNIX operating system, the semicolon must be preceded by an ESC character (\) as in the example below:

dplot myfile, DEVICE = dmp.dev \;

This prints/plots the drawing **myfile** using all **dplot.set** parameters or system defaults, except those for paper size, placement, and options.

Run Time Messages

Output Creation Messages

After all parameters are entered, plotting begins and the following message appears, informing you what DPLOT process is taking place:

```
Creating Output...number% Complete
```

After plotting finishes, system control returns to the operating system.

Cut Sheet Paper Message

If you are using cut sheet paper to print multiple drawings or multi-page drawings, DPLOT waits after printing each page and displays

```
Please change paper for next plot.
Press <Esc> to abort, any other key to continue plotting.
```

After you have inserted a new sheet of paper, press ↵ to continue or Esc to abort the plot.

Using Spooled Output

Why Spool Output?

Plotting and printing large jobs takes time, tying up the workstation doing the job. Users often want to either output from a different workstation, or have their workstation machine plot/print in the background while they continue using the workstation in the foreground.

Both of these options are possible by spooling your DPLOT output to a file.

Creating a Spooled File

When the DPLOT program requests a device name, answer in the form

device=filename

where *device* is the name of the desired device file, and *filename* is the name you wish to assign to a spooled output file. Any valid filename may be used.

Then answer all other DPLOT prompts normally. When DPLOT executes, output is sent to the spooled output file instead of the specified printer or plotter. This spooled output file contains all of the control codes necessary for the plotter/printer to create the drawing.

When working with drawings that must be output in strips, one file is created for each drawing strip. The last letter of the filename extension is altered to represent the strip number. The first nine strips are labeled 1 through 9, while additional strips are labeled A through Z.

For example, if output for a two-strip drawing is spooled to **dplot.out**, the following files are created:

dplot.ou1
dplot.ou2

Printing/Plotting the Spooled File for the PC

Once spooled, a file may be output at any time using the DOS **COPY** command with the **/b** parameter. This parameter tells DOS that you are copying a binary file.

For example, to print the spooled file **myplot.out** (created using JDL.DEV) on a JDL-750 printer connected to device LPT1 (the primary parallel port), enter **copy /b myplot lpt1**.

---

*Note: On HP-GL and DMP plotters, spooled files may be plotted in background mode, using the DOS **PRINT** command. This mode enables you to use your computer for another task while plotting at the same time. For more information on the use of this command, consult your DOS manual.*

Printing/Plotting the Spooled File for the Sun

To print/plot a spooled file use the utility program **dplotspl**. To plot a spooled file on a plotter that is attached to your machine, use the following command:

dplotspl file.spl /dev/ttyb

This will direct **file.spl** to serial port **b**.

To plot to a remote machine, you must specify the remote machine name. For the example above, with the plotter attached to machine "remote", use the following command:

dplotspl file.spl remote:/dev/ttyb

Note: There is no protection against two users accessing the plotter at the same time.

Batch File Processing

The following paragraphs describe a process that you can use to create multiple output files from DPLOT and then at a later time print or plot these files. This is not the only way to accomplish this, and it is not meant to be an all-inclusive strategy.

1. Change the OUTPUT specification in your **.dev** file to ***.out**.
2. Create a **dplot.set** file containing all user interface entries except the drawing name.
3. Create a batch file with one line per drawing you wish to batch. The line must include the DPLOT command, the name of the drawing file, and any parameter for this drawing which does not match the parameter in the **dplot.set** file.
4. Run the batch file.
5. Move the **.out** files to the proper devices (PRN, SER1, C9071, etc.) one at a time or all at once if your paper is **AUTO**, or **ROLL**.

Example #1 Outputting to a Plotter

These are the files to plot:

dr1.dwg
sc2.dwg
dr3.dwg

This is the standard device file for the DMP-41 Plotter:

NAME	**DMP 40/50 Series Plotters**
NOTE	**Version 2.10B**
NOTE	**Supports models 41, 42, 51, 52**
DRIVER	**\DPLOT\DMP.DRV**
OUTPUT	**SER1**
PAPER	**D, C, E**
BORDER	**2.0 X 2.0**
CHARACTER	
CIRCLE	
CUT	
GRID	**250**

Applying the steps mentioned above, take the following actions:

1. Change the OUTPUT specification from SER1 to ***.out**.
2. Create the file **dplot.set** with the following lines:

   DEVICE=DMP
   PAPER=D
   RES=LOW
   SCALE=TOFIT
   PLACE=CC
   ASPECT=100
   STRIP=ALL
   OPTS=

3. Create a batch file called **d.bat**, with one line per drawing. (Parameters shown here are just for this example. In general, use the parameters which you need that don't match the **dplot.set** file). Each line ends with a **;**.

   DPLOT DR1 ASPECT=90;
   DPLOT SC2:
   DPLOT DR3 RES=LOW PLACE=TL OPTS=B

4. At the DOS prompt, type **D** ↵ to execute the batch file.
5. If your plotter uses **CUT** paper, enter

   copy /b dr1.out ser1(change the paper)
   copy /b sc2.out ser1(change the paper
   copy /b dr3.out ser1

   If your plotter use **AUTO** or **ROLL**, enter

   copy /b *.out ser1

Example #2 Outputting to a Printer

These are the files to plot:

dr1.dwg
scat2.dwg
draw3.dwg

This is the standard device file for plotting to an EPSON printer:

NAME	**Epson MX, FX, RX Printers**
NOTE	**Version 2.10A**
DRIVER	**\DPLOT\EPSON.DRV**
OUTPUT	**PRN**
PAPER	**W (Wide, N (Narrow)**
ROLL	
GRID	**360, 540**

Applying the steps mentioned above, you would take the following actions:

1. Change the OUTPUT specification from **PRN** to ***.out**.
2. Create the file **dplot.set** with the following lines:

   DEVICE=EPSON
   PAPER=W
   RES=HIGH
   SCALE=TOFIT
   PLACE=CC
   ASPECT=100
   STRIP=ALL
   OPTS=

3. Create a batch file called **e.bat**, with one line per drawing. (Parameters shown here are just for this example. In general, use the parameters which you need that don't match the **dplot.set** file.) Each line ends with a **;**.

   DPLOT DRAW1 PLACE=TL SCALE=NONE;
   DPLOT SCAT2 PAPER=N
   DPLOT DR3;

4. At the DOS prompt, type: **E** ↵ to execute the batch file.
5. The EPSON printer uses **ROLL**, therefore:

   COPY /B *.OUT PRN

FutureNet®

Companion Programs and Reference

User Manual

DATA I/O Corporation

096-0087-002

September 1991 096-0087-002

Data I/O Corporation
10525 Willows Road N.E., P.O. Box 97046
Redmond, Washington 98073-9746 USA
(206) 881-6444

Acknowledgments:

Table of Contents

1. Overview

2. Part File Syntax

3. FutureNet Part File Utility

4. Part Library Utility

5. FutureNet Symbol Library Utility

6. FutureNet Symbol Identifier Utility

7. Error Messages

List of Figures

1 Overview

This manual describes utilities that enhance FutureNet by translating FutureNet designs to other CAD/CAE systems and back again.

- FutureNet Part File Utility — Creates part files for FutureNet utilities/translators.
- FutureNet Part Library Utility — Translates and combines one or more part files into a single library.
- FutureNet Symbol Library Utility — Documents each symbol in a FutureNet symbol library in the form of a FutureNet drawing.
- FutureNet Symbol Identifier Utility — Documents the symbol identifiers used in a design.

Read the "Understanding FutureNet" chapter of the *FutureNet User Manual* before reading this manual.

The following five figures illustrate the FutureNet design development processes and the relationships of the utilities. Reviewing these processes and relationships will help you move easily from one design phase to another.

- Figure 1-1 shows an overview of the FutureNet Part File concept.
- Figure 1-2 shows the relationship between the FutureNet Part File symbol relationships. It's labeled "The Mapping Concept".
- Figure 1-3 shows how a custom translator can use the Part File.
- Figure 1-4 shows the process for generating a Part File.
- Figure 1-5 shows where each of the FutureNet Utilities fits into the FutureNet design task.

Figure 1-1
The Part File Concept

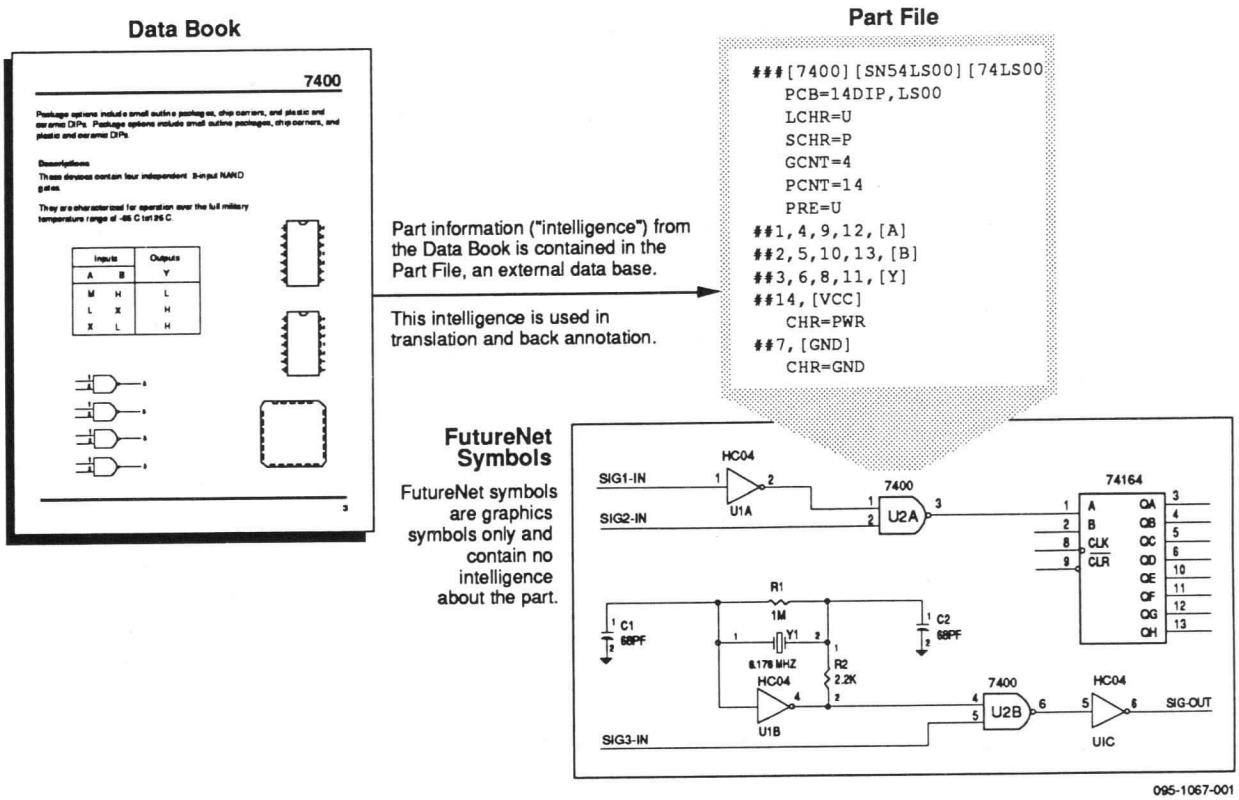
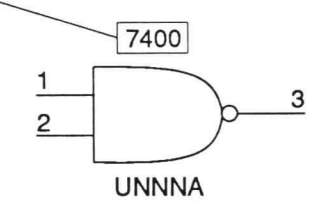


Figure 1-2
The Mapping Concept

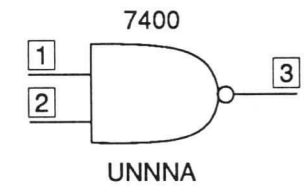
Part Name

```
###[7400][SN54LS00][74LS00]
   PCB=14DIP,LS00
   LCHR=U
   SCHR=P
   GCNT=4
   PCNT=14
   PRE=U
##1,4,9,12,[A]
##2,5,10,13,[B]
##3,6,8,11,[Y]
##14,[VCC]
   CHR=PWR
##7,[GND]
   CHR=GND
```



Pins

```
###[7400][SN54LS00][74LS00]
   PCB=14DIP,LS00
   LCHR=U
   SCHR=P
   GCNT=4
   PCNT=14
   PRE=U
##1,4,9,12,[A]
##2,5,10,13,[B]
##3,6,8,11,[Y]
##14,[VCC]
   CHR=PWR
##7,[GND]
   CHR=GND
```



Gate

```
###[7400][SN54LS00][74LS00]
   PCB=14DIP,LS00
   LCHR=U
   SCHR=P
   GCNT=4
   PCNT=14
   PRE=U
##1,4,9,12,[A]
##2,5,10,13,[B]
##3,6,8,11,[Y]
##14,[VCC]
   CHR=PWR
##7,[GND]
   CHR=GND
```

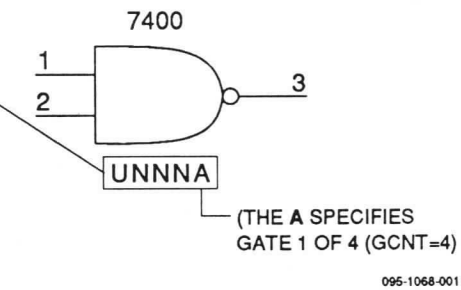


Figure 1-3
Custom Translators

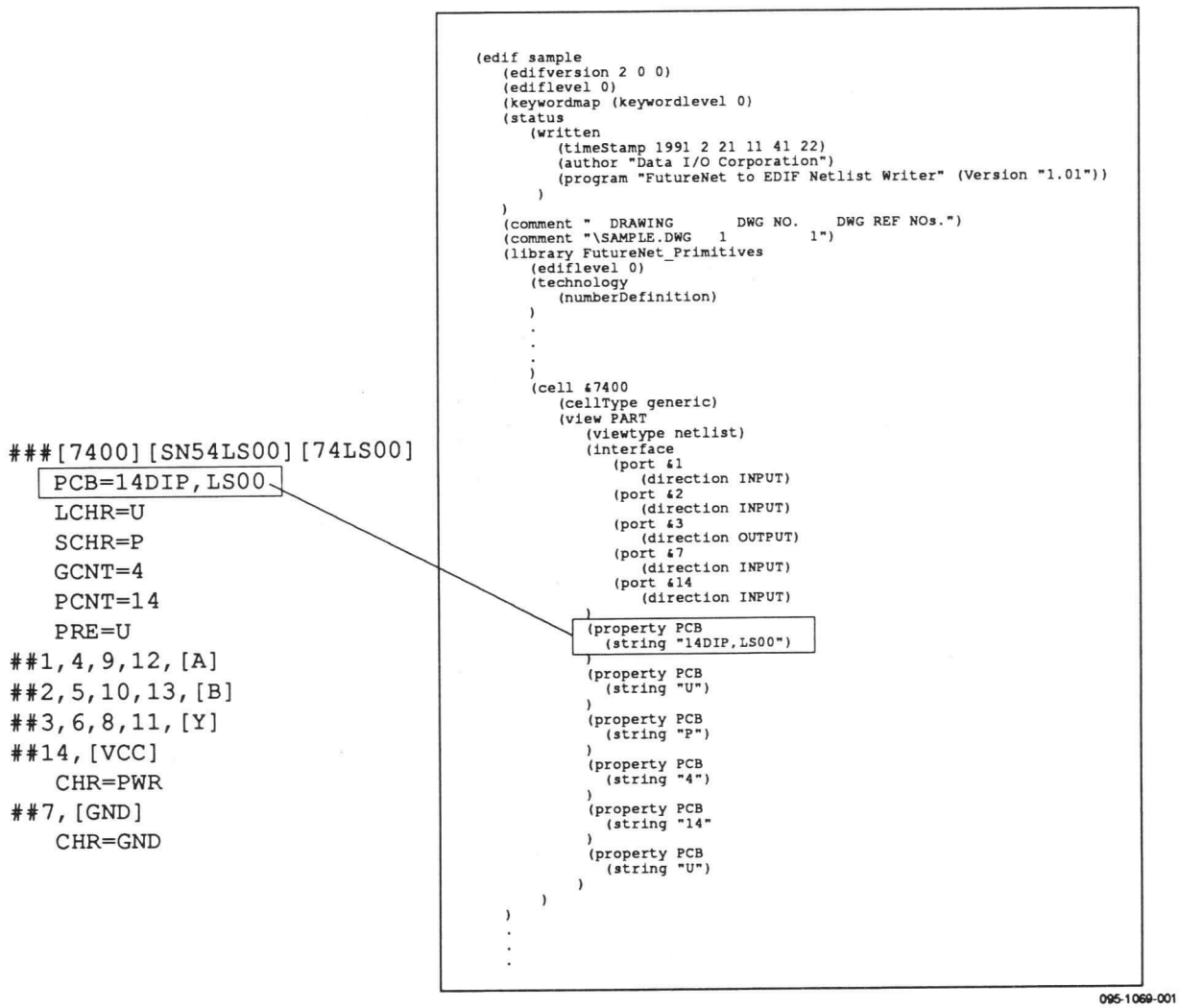


Figure 1-4
Generating a Part File

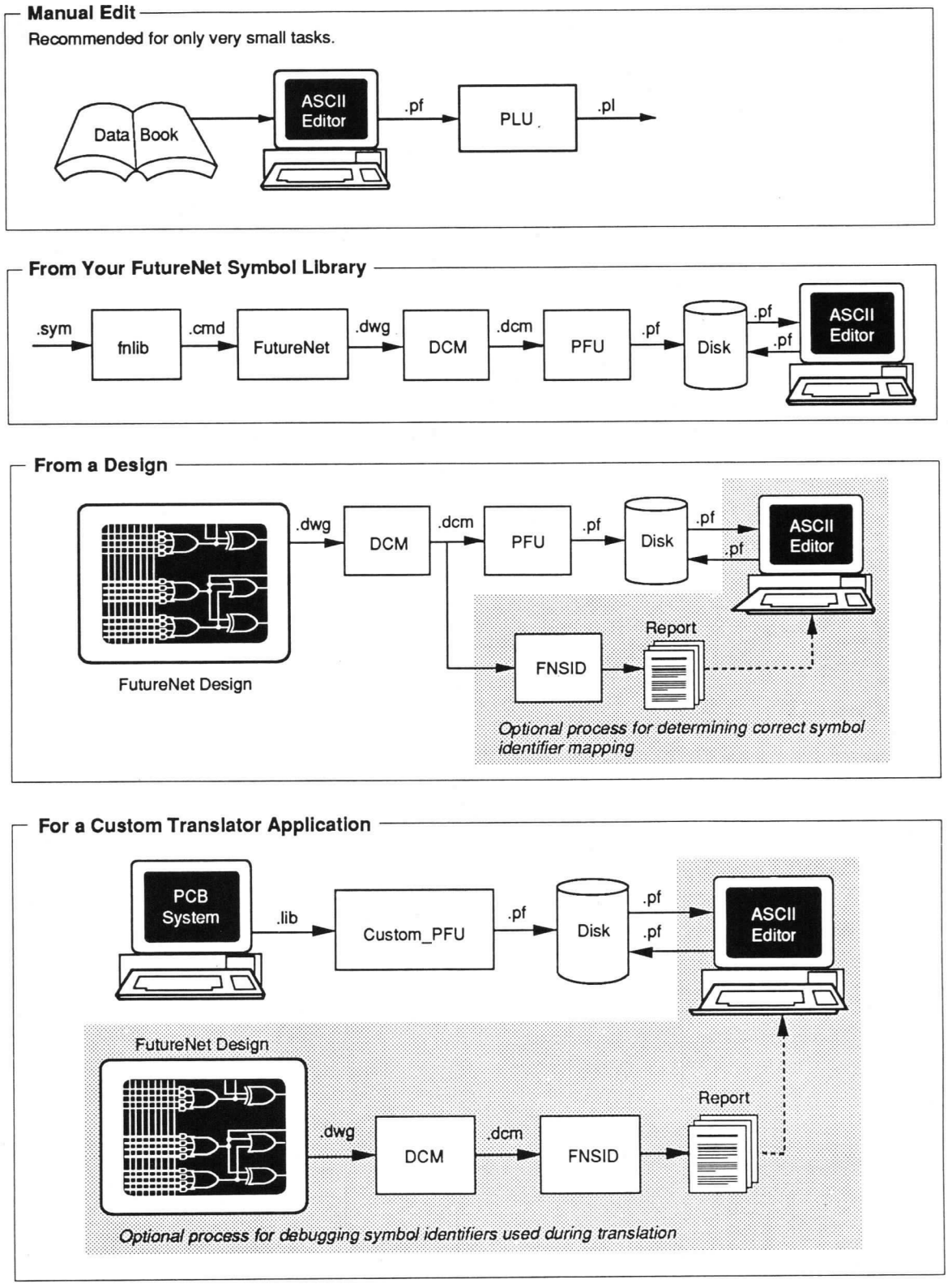
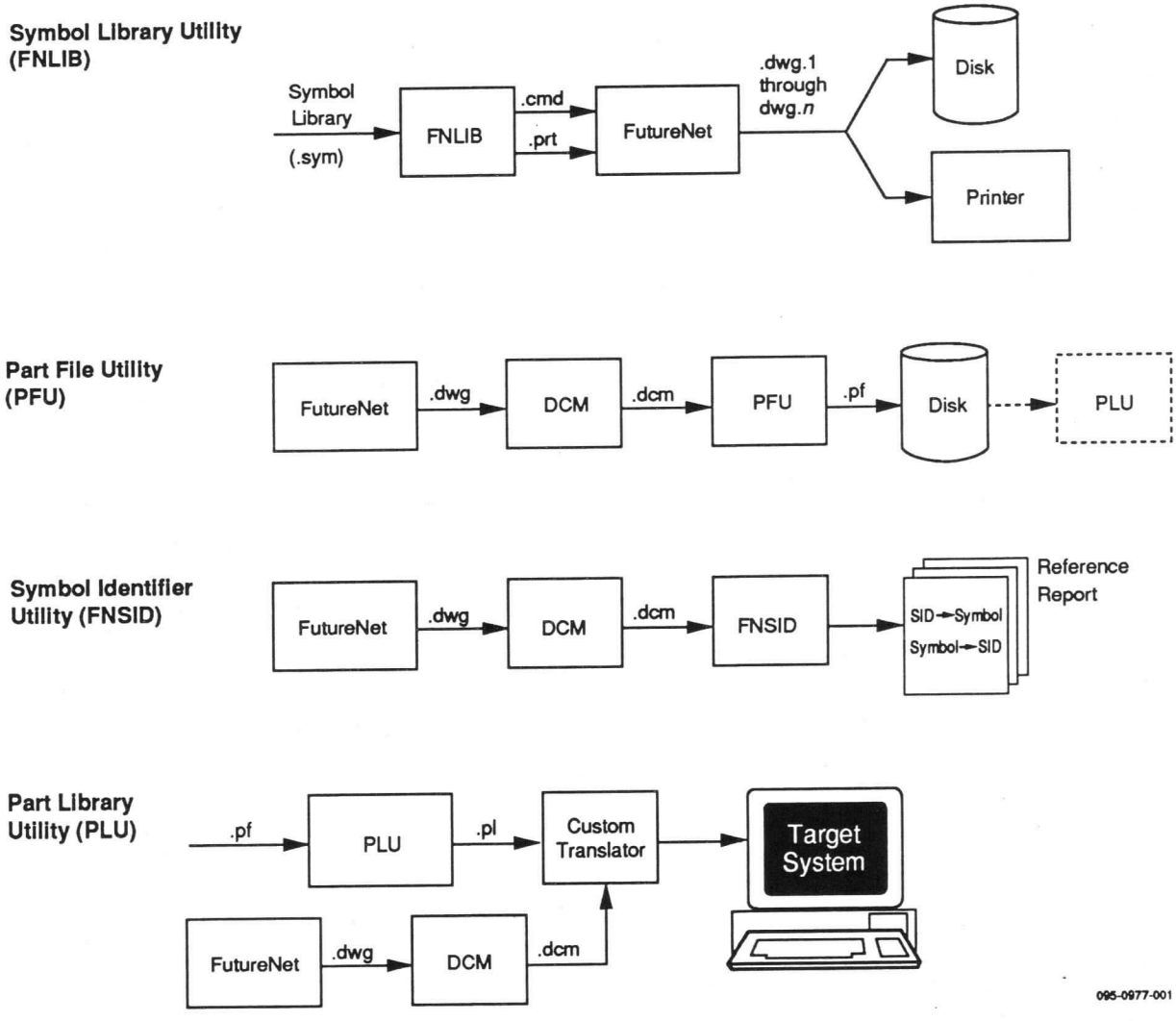


Figure 1-5
FutureNet Utilities Flow Chart



Definitions

Following is a list of terms you should become familiar with before using the utilities.

Alphanumeric Field — A displayed line of text (of a given font) on a drawing, with an associated attribute, layered text, justification, orientation, and point of effect.

Attribute — A predefined property represented by a number and name (mnemonic) that is assigned to an alphanumeric field. The property is characteristic of the schematic element with which the alphanumeric field is associated.

Attribute Mnemonic — A 3- or 4-character name which identifies an attribute.

Attribute Number — A number that identifies an attribute.

Device — An electrical component that is used on a Printed Circuit Board (PCB).

Displayed Text — The single line of text in an alphanumeric field that is visible on the drawing.

Drawing Set — A set of drawings at the top of a hierarchy or the drawing(s) named in a functional block.

Function — An electrical operation performed by a set of pins in a part.

Functional Block — A symbol that is used as a placeholder in a higher level drawing to specify a drawing set at a lower level in the design.

Functionally Defined Part — A part that contains more than one slot.

Global Property — A property assigned to a part that occurs only in a part file and applies to all instances of that part in a design.

Instance-Specific Property — A property assigned to a symbol in FutureNet Schematic Designer using attributes or layered text. This can be used to override a global property.

Layered Text — The text in an alphanumeric field that is not visible on the drawing. Layered text is used to create property sheets to supplement the alphanumeric field attribute.

Like — Equivalent.

Multiple Slots — Like Function — A part that contains more than one slot with each slot having equivalent functions.

Multiple Slots — Mixed Function — A part that contains more than one slot with the slots having different functions.

No Slots — Single Function — A part that contains no slots and has either a single function or no function.

No Slots — Mixed Function — A part that contains no slots but has different functions represented by two or more symbols. This is a special case of the multiple slot — mixed function part.

Part — A reference to a single symbol or a set of symbols that represent an electrical device.

Part File — A database of part information. This information is not present in the schematic symbols of the design and aids in determining the number of gates in a part, the correct pins for each gate, the number of pins of a part, etc.

Physically Defined Part — A part that is not functionally defined and is defined only by the number of pins for that device.

Pin Identifier — The displayed text (pin name or number) used to establish connectivity.

Point of Effect — A movable x,y coordinate that is used to associate an alphanumeric field with a schematic element.

Property — A single characteristic of a schematic element.

Property Name — The name of a particular characteristic of a schematic element.

Property Assignment Statement — The syntax required to assign a property to a schematic element (used in layered text or in a part file):

[tool]:property=value.

Schematic Element — Any design element that belongs to one of the following conceptual areas in the design: circuit, drawing, bus, signal, pin, and symbol.

Scope — The extent of the connectivity that signals with the same name have within a design.

Signal Aliasing — A signal that has been assigned more than one name (bus aliasing is not allowed.)

Slot — A specific position for a function within a part. For example, pins 1, 2, and 3 of a 7400 part would be slot 1; pins 4, 5, and 6 would be slot 2, etc.

Symbol — A single graphic representation of an entire electronic device or a portion of one.

Tool — A specific post processor or translator.

Tool Identifier — A name that associates a property with a specific tool.

Value — The data assigned to a property.

FutureNet Properties

It is especially important to understand how properties are derived from attributes and layered text. A brief explanation of properties is given here, but detailed information is provided in the *FutureNet User Manual.* There is no limit to the number of properties that can be specified in a design, but there are a basic set of properties that are common to the utility programs. Target-specific translators might use additional properties or extended definitions of the base set.

A property denotes a single characteristic of a schematic element. All schematic elements have at least one property, most have more. Properties identify things about an element such as:

Pin type, for example

tristate, output, or input

Signal/pin name, for example

GND
tsv
sig01

Location reference designator, for example

U21
U0001
U15

Part

and/2
80486
244

All property information must be available before any post processing tools or translators can interpret and process your design.

Property Assignment

Properties are assigned using FutureNet attributes or property assignment statements. These statements take the following form:

[*tool:*]*property=value.*

There are two ways to assign properties:

- Global property assignment using part files
- Instance-specific property assignment in the FutureNet program

Global Property Assignment

Global property assignment statements must be placed in a part file. (See Chapter 2, Part File Syntax.) These properties are specified on a per part and per pin basis. When you add a property to a part in the part file, the FutureNet program adds that property to all parts of that type.

Instance-specific Property Assignment

The FutureNet program assigns instance-specific properties by attributes or by layering property assignment statements under a symbol, displaying them with the symbol on the drawing, or using a combination of layered and displayed text. Attributes and property assignment statements are associated with an alphanumeric field. Attributes are provided for the most common properties required in a design, for quick and easy property assignment.

Instance-specific properties apply only to the symbol they are assigned to and take precedence over properties assigned in part files. The FutureNet editor does not allow you to add global property information.

Symbol-related Properties

The properties associated with a part are used to determine if a part is functionally or physically defined.

Symbol-related properties apply to the symbol to which they are assigned. Properties that relate to a part are global properties; these properties are consistent for all occurrences of a part, such as number of pins, number of gates, or swap characteristics. Properties that relate to an occurrence of a part are instance-specific properties and these vary for each occurrence of a part in the design, such as location designator, gate designator, or board location.

The following are definitions of the most common FutureNet symbol-related properties.

GATE=gate_id (Instance-specific, attribute GATE, 155)

The GATE property identifies a symbol as one of the symbols of a multi-symbol part. For a functionally defined part, the gate_id specifies which slot the symbol should occupy. For a physically defined part, the gate_id specifies which pin set in the part the symbol should occupy. For example

GATE=C

specifies that the symbol this property assignment statement applies to, occupies the third pin set in a multi-symbol part.

GATE=32

specifies that the symbol occupies the thirty-second pin set in a multi-symbol part. The gate property can also be obtained from the LOC property. See the LOC property description for details.

GCNT=n (Global, no attribute)

The GCNT (gate count) property is for functionally-defined parts and defines the number of symbols (gates) that make up a part, where *n* is always an integer. This property is used when:

- Multiple FutureNet symbols are used to represent a single part.
- The part is functionally defined.
- Each symbol that makes up the part must be a single gate.

The GCNT property value is used to verify that you have not exceeded the specified gate count. For example, if you tried to use the LOCs U1A, U1B, and U1C to represent three gates in one package, but assigned the part a gate count of 2, GCNT=2, the translator would generate an error message.

Note, however, that if your drawing uses fewer symbols than are specified by GCNT, no error will be detected. Ensure that you have specified the correct number of symbols for a complete part.

See the SCNT property description for using multi-symbol parts in FutureNet that are physically-defined.

LOC=location value (Instance-specific, attribute LOC, 2)

LOC is the circuit location or reference designator for a symbol, such as, U21, R18, and C81. Adding a letter suffix to the LOC identifies the symbol as one of the symbols in a multi-symbol part. The letter suffix is treated as the GATE property unless you specify otherwise. For example, U1C could indicate the symbol occupies the third slot in a device.

Parts with GCNT=1 do not need the letter suffix "A".

For multi-symbol parts, see the GATE, GCNT, and SCNT property descriptions.

PART=*part_name*
VAL=*part_value*
STR=*part_stress*
TOL=*part_tolerance*
PCHR=*part_characteristics*

(Instance-specific, attributes PART, 3; VAL, 4; STR, 7; TOL, 6; PCHR property)

These five properties identify the part depicted by a symbol. The **symbol identifier** for a symbol is derived from the concatenation of these five properties. At least one of these five properties must be present on a symbol for a part (the PART property is the most common). These properties are most commonly assigned via FutureNet attributes.

The symbol identifier is used to access the part file for global property information. See the "Symbol Identifier" entry section in Chapter 2 for more information. You can create unique symbol identifiers by changing one of the five properties. For example, you could have two different 10KW resistors by specifying two different tolerances with the TOL (tolerance) property as follows:

VAL=10K
TOL=5%

VAL=10K
TOL=1%

PCNT=n (Global)

The PCNT property specifies the number of pins for the part. The number of pins is *n*. The count includes implicit power and ground pins.

PRE=val (Global)

The PRE property specifies the prefix that should be used on location designators. For example, PRE=U implies that the preferred prefix for this part is U.

LCHR=U | A (Global)

The LCHR Location Designator (reference designator) characteristics property specifies that a part is physically defined. If this property is present or no SCHR property is present, the part is physically defined. The value of the property is instance-specific and specifies whether the reference designator for the associated symbol is assigned or unassigned. The possible values are:

U The location designator is not assigned. The LOC property on the schematic is ignored. (It is similar to SCHR=P for functionally defined parts.)

A The gate is assigned a location designator. The LOC property is the location designator the the part. (It is similar to SCHR=N for functionally defined parts.)

Normally the location designator characteristics default to unassigned (LCHR=U or not SCHR or LCHR) in the part library. You may override this in the drawing for parts which have been assigned (LCHR=U) by the property assignment statement LCHR=A.

Note: The gate designator on multi-symbol, physically defined parts, must be specified even if the location designator is not assigned. LCHR=U on a symbol of a multi-symbol part implies the symbol is not assigned, but it does represent a specific pin set of the part as designated by the GATE property.

SCHR=N | P (Global)

The SCHR (swap characteristics) property specifies that a part is functionally defined. If this property is absent, the part is physically defined. The value of the property is instance-specific and specifies whether the associated symbol can be swapped with another part. The possible values are:

N The gate cannot be swapped. The LOC and physical pin numbers on the schematic are the assigned gate. The gate is already packaged in the part. (It is similar to LCHR=A for physically defined parts.)

P The gate can be swapped with any other gate in any identical part in the design. The pin numbers on the gate are any valid gate of the part. The part location is unassigned and the LOC property is ignored. (It is similar to LCHR=U for physically defined parts.)

Normally the swap characteristics are defaulted to (swappable) (SCHR=P) in the part library. You may override this in the drawing for parts which have already been packaged (SCHR=N). SCHR=N in the part file will specify that a part is functionally defined but all the packaging in the drawing is fixed.

Note: When you use parts that have been assigned the SCHR property, the parts cannot have unconnected pins. Unconnected pins must be connected to a signal called N/C.

SCNT=n (Global)

The SCNT (symbol count) property is for physically defined parts and specifies the number of discrete FutureNet symbols that make up the part. Use this property for multi-symbol parts that are physically defined. The SCNT property is used to verify that the correct number of symbols are present for a part by comparing the SCNT property value with the number of GATES for a LOC.

For example, you could use a transistor symbol and an LED symbol to build an optoisolator part. The part would have SCNT=2 because there are two symbols in the part. If you used the part on a schematic, two symbols with GATE properties 1 and 2 would be required for each occurrence of the part.

Pin-Related Properties

Pin-related properties affect the pin to which they are assigned. They can be global or instance-specific. The pin identifier assigned to a pin in the drawing is used to map the pin into the part file for global pin properties.

CHR=PWR | GND | COM (Global)

The CHR (characteristics) property specifies the characteristic of a pin as either power, ground, or common. See your specific translator manual for more information.

PNUM=pin_number (Global)

The PNUM (pin number) property specifies physical pin number. It is required for all pins, but defaults to the pin identifier if that identifier is numeric. For example, if the pin identifiers for a transistor symbol are **E** for the emitter pin, **B** for the base and **C** for the collector, the pin numbers 1, 3, and 2 can be established in the part file using the PNUM property. (See the "Mapping Pin Names and Numbers" section in Chapter 2.)

PNAM=pin_name (Global)

The PNAM (pin name) property specifies the pin name. The pin name defaults to the pin identifier if that identifier is alphanumeric.

2 *Part File Syntax*

A part file is a database of symbol and pin properties for FutureNet symbols or for the symbols in target system libraries. There is one entry in the file for each symbol type in the library or design. The property data assigned to the symbols and pins in a part file applies globally to all symbols and pins of a given part type, unless overridden by drawing level property data. In addition to accepting global property data, part files are used to map parts and their pins between systems.

Symbols are listed in the part file by a symbol identifier, which is derived from the concatenation of the PART, VAL, STR, TOL, and PCHR properties or from user-defined identifiers specified by the attribute option. Note that a single symbol identifier may apply to a number of different symbols. For example, the FutureNet symbols 00A (2-input NAND, pins 1, 2, and 3) and 00B (2-input NAND, pins 4, 5, and 6) have the same symbol identifier, 7400. Each symbol entry includes the pins for that symbol, listed by a pin identifier.

Part files can be created automatically using one of the custom part file utilities available with the FutureNet translators or with the part file utility (PFU) that generates a part file from a design. All part file utilities conform to the syntax described here, while the data required from the user to complete the file varies between utilities.

Part files can be created manually using any ASCII editor. Obviously, part files created manually contain only the data you put in them, though they still must conform to part file syntax. Creating part files manually is practical for only a few parts because of the amount of data required.

Part Types

There are four part classifications. Depending on the part type, a part can be functionally or physically defined with single or multiple FutureNet symbols. The next four sections describe the part classifications: Multiple Slots — Like Functions, Multiple Slots — Mixed Functions, No Slots — Single Function, No Slots — Mixed Functions.

The definitions below are important to your understanding of the part type discussion that follows.

Function — An electrical operation performed by a set of pins in a part.

Functionally-defined part — A part that contains more than one slot.

Physically-defined part — A part that is not functionally defined. It is defined only by the number of pins for that particular device.

Slot — A specific position for a function within a part. For example, pins 1, 2, and 3 of a 7400 part would be slot 1; pins 4, 5, and 6 would be slot 2, and so on.

FutureNet Symbols

The FutureNet symbols that you use to represent a functionally-defined part must match the definition of the part in the part file. For example, when you are using parts with multiple slots, each slot must be represented by a separate FutureNet symbol. A connector functionally-defined as 32 1-pin gates must be represented by 32 separate symbols. Parts that are physically-defined can be represented by multiple symbols, but they will not have the swap characteristics of a functionally-defined part. Each symbol of a physically-defined part reflects some set of pins in the part.

If you are using FutureNet as an interface to another system, such as a PCB layout system, the definition in the part file and corresponding FutureNet symbols must match the part definition in the target system.

Multiple Slots — Like Functions

This part type contains more than one slot with each slot having equivalent functions. This part type can be functionally or physically defined and it is the only part type that can be functionally defined.

If the part is functionally-defined in the part file, then the part has swap characteristics (SCHR=N or SCHR=P). You should use this approach with common TTL devices such as the 7400, 7404, and 7406.

If the part is physically defined in the part file, it may be represented by multiple symbols, but the part would not have swap characteristics. You might use this approach with resistor packs or a 7474 flip-flop.

Multiple Slots — Mixed Functions

This part type contains more than one slot and each slot has different functions. Currently it must be physically defined (LCHR=U or LCHR=A) and the gates within the part cannot be swapped. This part type can be defined with multiple FutureNet symbols. The SN74265, which contains two buffers and two NAND gates, is an example of this part type.

No Slots — Single Function

This part type contains no slots and has either a single function or no function. It does not contain gates and must be physically defined (LCHR=U or LCHR=A). It is usually represented by a single FutureNet symbol. The MC68000 microprocessor IC, a 16R8 PAL IC, and a custom gate array are examples of this part type.

No Slots — Mixed Functions

This part type contains no slots but has different functions represented by two or more symbols. This part is a special case of the multiple slot — mixed function part. This part type must be physically defined and could consist of multiple FutureNet symbols which are combined to form one part. For example, you could use a transistor symbol and an LED symbol to create an optoisolator part. Another example of this part type is a multi-symbol connector.

Part File Example

The part file example on the following page shows a typical FutureNet database of symbol and pin properties. Following the example are detailed descriptions of the elements in the part file.

```
###[4164]
  PCNT=16
  PRE=U


###[2N2222A]
  PCNT=3
  PRE=Q
##1,[E]
##2,[B]
##3,[C]


###[1N4148}
  PCNT=2
  PRE=CR


###[,10UF]
  PRE=C
##1,[A]
##2,[B]

###[7400]
  SCHR=P
  GCNT=4
  PCNT=14
  PRE=U
##1,4,9,12
##2,5,10,13
##3,6,8,11
##7
  CHR=GND
##14
  CHR=PWR


###[7404]
  SCHR=P
  GCNT=6
  PCNT=14
  PRE=U
##1,3,5,9,11,13
##2,4,6,8,10,12
##7
  CHR=GND
##14
  CHR=PWR
```

Syntax

The part file is made up of part identifier entries and their associated pin identifiers. Property information can be added to the part and pin identifiers using property assignment statements of the form:

property=value

or

tool:property=value

Tool and property values are not case-sensitive.

Comment lines, which begin with a single #, can be placed anywhere. Blank lines are permitted.

Lines may be continued by using a back slash (\) before the new line.

Syntax Conventions

Part file syntax uses the following conventions:

{ }	optional entry
{ }	zero or more repetitions of an optional entry
# and [ ]	literals

#	comment
###	symbol identifier
##	pin identifier

The stylized part file example below demonstrates part file syntax,

```
# This is a comment
###[symbol_identifier] {[symbol_identifier]}...
{property_assignment_statement}
...
##pin_identifier {[,pin_identifier]}...
{property_assignment_statement}
...
##pin_identifier {[,pin_identifier]}...
{property_assignment_statement}
...
# next part entry
###[symbol_identifier] {[symbol_identifier]}...
...
```

Symbol Identifier Entry

Symbol identifier entries are derived from the concatenation of the PART, VAL, STR, TOL, and PCHR properties or from user-defined identifiers as specified by the attribute option. Symbol identifiers are unique. They are used to identify parts in the part file and should not be confused with the part property on a FutureNet symbol, which is only one of the five properties comprising the symbol identifier. FutureNet symbols must have at least one of these properties or a user-defined identifier to be included in the part file. The symbol identifier takes the following form:

{part_prop}{,val_prop}{,str_prop}{,tol_prop}{,pchr_prop}

where part_prop, val_prop, str_prop, tol_prop, and pchr_prop are the property values for the PART(2), VAL(3), STR(7), TOL(6), and PCHR (no current attribute) properties from the symbol in the FutureNet drawing. These properties cannot have embedded commas.

Missing leading entries must be indicated with a comma. For example, symbols identified by VAL and PCHR would be listed as:

###[,1K,,,CARBON]

Symbols identified by PART and PCHR would be listed as:

###[8088,,,,INTEL]

Missing trailing entries do not use a comma. Symbols identified by VAL and TOL would be listed as:

###[,10K,,5%]

Symbols identified by VAL, STR, TOL, and PCHR would be listed as:

###[,10uF,50v,10%,TANT]

A symbol identified by VAL, only the symbol identifier would have the following form:

###[,10K]

WARNING: The -a*num* option (attribute number) is not supported by the FutureNet to Edif Netlist Writer. See Chapter 3, the "Command Line Options" section, for a description of -a*num*. Alternatives to -a*num* include changing the symbol identifier for a symbol in the FutureNet program or using the PID property in a part file to specify a different identifier for a symbol.

Pin Identifier Entry

The pin identifier lists the pins on a part. Pin identifiers are derived from displayed alphanumeric fields that have been assigned one of the pin attributes used to assign an identifier.

Pin identifiers are either numeric or alphanumeric. In most cases, they will be a number. Alphanumeric pin identifiers must be enclosed in brackets [ ].

For many physically defined parts, pin entries are not required if the FutureNet symbol contains valid numbers for the pin identifier.

Pin identifier entries are listed in the part file under a symbol identifier entry.

Functionally Defined Parts

Pin identifiers for functionally defined parts form a matrix in which columns correspond to a specific gate symbol and rows correspond to a single input or output pin. The last column on the right may be the logical pin identifier (alphanumeric character) and does not correspond to a specific gate.

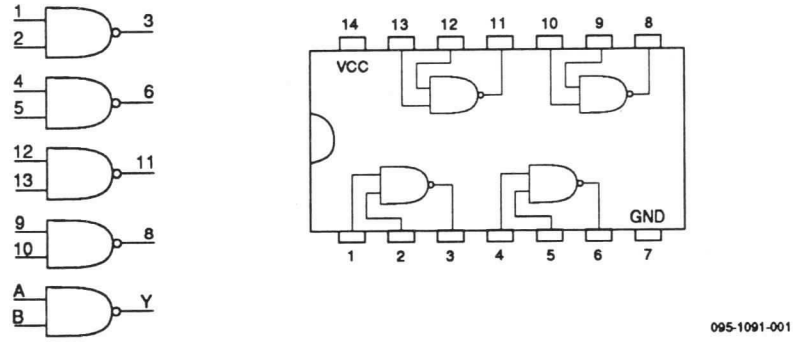
Note: The order in which pins appear is important and must match the order specified in the data book. If this order is not adhered to, packaging or back annotation may not be accomplished correctly.

After the gate pins are defined, power and ground pins for the part can be specified.

Symbol and pin properties are added to the part file as property assignment statements. Symbol properties follow the ### symbol identifiers line entry. Pin properties follow the ## pin identifiers line entry.

Figure 2-1 shows a 7400 package and the individual symbols that can be used to represent it. Notice that the last gate in the column on the left is a logical representation of a gate and does not correspond to a specific gate in the 7400.

Figure 2-1
7400 Part



Multiple gate parts like the 7400 are listed in the part file with pins in rows and gates in columns. The first entry is the symbol identifier, described earlier. Pin identifiers are listed after the symbol identifier, with inputs first (pins 1 and 2) and outputs last (pin 3). There is one row for each pin on the symbol. All of the pins on a symbol are listed in one column, which corresponds to the gate. Since each of the gates in a 7400 has three pins, there are three rows. Since there are four gates there are four columns, in gate order, and one extra column to list logical pin identifiers. The global properties required are SCHR and GCNT. A complete entry for a 7400 looks like this:

```
###   [7400]
                    SCHR=P
                    GCNT=4
                    ##   1,4,9,12,[A]
                    ##   2,5,10,13,[B]
                    ##   3,6,8,11,[Y]
##14
                    CHR=PWR
##7
                    CHR=GND
```

Alphanumeric Pin Identifier

Note that numeric pin identifiers are separated by commas and that alphanumeric pin identifiers, in addition to being separated by commas, are also enclosed in brackets. Pin identifiers enclosed in brackets are treated as names and must therefore have an alpha component. The example below is not legal because the pin identifier enclosed in brackets does not have the required alpha component.

```
##1,[2]                                    INCORRECT
```

Multiple Power and Ground

Duplicate pin identifiers are not allowed. Under a part, a given pin identifier can be specified only once.

However, the same values for the PNAM and PNUM properties can appear under different pin identifiers. The following is not be a legal method of associating ground with two pins.

```
##1,[GND]                                  INCORRECT
##7,[GND]
```

The following example shows the correct way to specify two ground pins using the PNAM property and property assignment statements.

```
##1
   PNAM=GND                               CORRECT
##7
   PNAM=GND
```

Common Pins

Parts that use common pins cannot be functionally defined using gates. The following error is reported if you try to define the part using gates, because the common pin is detected as a duplicate pin.

ERROR - File *filename*, line *linenumber*
Duplicate pin specification.

For example, an SN74LS244 is defined with the following pin entries for each gate:

Gate 1: 1,2,18
Gate 2: 1,4,16
Gate 3: 1,6,14
Gate 4: 1,8,12

Pin number 1 is common to all four gates.

The following is not a correct part file definition for this part.

```
###[LS244]
    ##1,1,1,1,[G]
    ##2,4,6,8,[A]
    ##18,16,14,12,[Y]                    INCORRECT
    ##10,[GND]
    ##20,Vcc]
```

The correct part file definition for this part would be:

```
###[LS244]
    ##1[G]
    ##2[A]
                                         CORRECT
.
.
.
##20[Vcc]
```

Mapping Pin Names and Numbers

The pin identifier on the FutureNet symbol can specify only a PNAM or PNUM property. If a pin name (or number) is missing or the pin name (or number) on the symbol is not valid for a utility, PNAM (or PNUM) properties can be added to the part file. For example, suppose a transistor in FutureNet uses pin number 1 or pin name **E** for the emitter, pin number 2 or pin name **B** for the base, and pin number 3 or pin name **C** for the collector. The following statements in the part file cause the translator to substitute pin number 1 for the FutureNet pin **E** or 1, pin number 3 for the FutureNet pin **B** or 2, and pin number 2 for the FutureNet pin **C** or 3. Note that for the emitter pin a PNUM=1 is not required because the Part Library Utility provides a default PNUM property of PNUM=1.

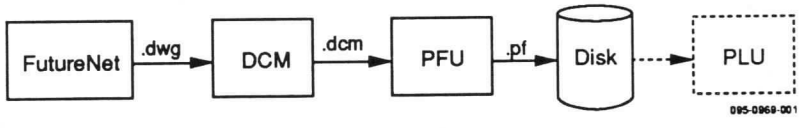
```
##1,[E[
##2,[B]
    PNUM=3                               CORRECT
##3,[C]
    PNUM=2
```

3 FutureNet Part File Utility

The FutureNet Part File Utility (PFU) is used to create part files for FutureNet utilities/translators. Part files are used as input to the Part Library Utility (PLU), converted, and then used by the target utilities or translators. See Figure 3-1 for a flow chart of the Part File Utility.

The FutureNet Part File Utility takes the Drawing Preprocessor .dcm file as input and creates a part file template of the FutureNet design.

Figure 3-1
Part File Utility Flow Chart



Note: The part file created by the FutureNet Part File Utility is only a template of the symbols in a FutureNet design. You must edit the file to include symbol and pin property data relevant to your system and design. See Chapter 2 for more information on part file conventions and syntax.

FutureNet to EDIF Netlist Writer

If you intend to use the FutureNet to EDIF Netlist Writer and want to create an optional part file, use the FutureNet Part File Utility to create the .pf file. Use that .pf file as input to the Part Library Utility. The Part Library Utility creates a .pl file as an output. Then run the EDIF translator using the .dcm and .pl files as input.

Other Translators and Utilities

If you intend using other FutureNet translators, create the required part file from the target system library(s) using the part file utility described in your translator manual.

Optionally, you can use a part (.pf) file created from your FutureNet design using the FutureNet Part File Utility or you can create it manually using an ASCII editor (suitable for small designs or a few parts).

Create a part library using the part (.pf) files as input to the Part Library Utility. The Part Library Utility creates a .pl file.

Run the translator/utility using the .dcm and .pl files as input.

The FutureNet Part File Utility uses FutureNet drawings as input and creates a template of the parts in a FutureNet design. Translator-specific part files are created from target system library(s) and create a template of the parts in the library(s).

Conventions

Symbol and Pin Identifiers

Symbol identifier entries are derived from the concatenation of the PART, VAL, STR, TOL, and PCHR properties or from user-defined identifiers as specified by the -anum option.

Two symbols can have the same symbol identifier but different sets of pin identifiers. The part file utility assumes these are symbols for a functionally-defined part. For example, the standard FutureNet parts 7400A and 7400B have the same identifier (7400), but the pin identifiers are (1,2,3) for the 7400A and (4,5,6) for the 7400B. When this occurs, pins with the same location relative to the symbol origin are grouped together in the output as shown below.

```
### [7400]
    ## 1, 4
    ## 2, 5
    ## 3, 6
```

This part file definition is incomplete because gates 3 and 4 are missing, as well as power and ground pins.

Editing Requirements

If the part file is being generated from a drawing set where the pin ordering does not match the data book, the part file entry will be erroneous. This can occur as a result of pins being swapped on a symbol. If all gates of a part are not present in a drawing, you must edit the part file to add the missing gate information.

Symbols That Do Not Have Identifiers

Symbols must have at least one of the five properties that make up the symbol identifier or a user-defined attribute (via the -anum option) in order to appear in the part file. See the -anum description under Utility-specific Options. Symbols that do not meet this requirement, like title blocks, are ignored.

Properties

The Part File Utility outputs the symbol properties GCNT, PCNT, PRE, SCHR, SCNT, and the pin property CHR based on the data provided in the design. Option -x is provided to exclude these default property definitions. GCNT, PCNT and SCNT have values of the number of gates, pins, or symbols encountered in the design. SCHR has a value of P and PRE a value of U.

Running the Utility

To create a part file, the minimum command line entry is:

pfu *dcmfile*

Command Line Options

The Part File Utility uses the basic command line options in addition to several system-specific options.

Basic Command Line Options

Basic command line options are summarized below.

Option	Definition
-i*filename*	Specifies input file(s) (.dcm)
-o*filename*	Specifies output file name (.pf)
+o	Resets output file name to default.
-e*filename*	Specifies output file for errors (.pfe).
+e	Resets error file name to default.
-s	Sets program to silent mode.
+s	Resets program to non-silent mode.
-q	Sets program to query mode.
+q	Resets program to non-query mode.

Utility-Specific Options

Option	Definition
-a*num*	Attribute Number — The **-a***num* option allows you to specify your own symbol identifier for a symbol by instructing the utility to take the identifier for the symbol from the displayed text field with the specified attribute number.

Option	Definition
-anum	Symbol identifier entries are by default derived from the concatenation of the PART, VAL, STR, TOL, and PCHR properties and can be overridden using this option. When using this option, use the same attribute number for all symbols. For example, if you want a symbol to be identified by your company part number, then use FutureNet to create an alphanumeric field containing the company part number in the symbol and assign the attribute number to the field. The symbol-related attributes listed here are the legal attributes available for use with the -a option: 3 4 6 7 55-99 109-113 116-127 131 135 140 142-144 Attribute numbers above 99 are non-printing, displayed text attributes; they are probably the best choice since they do not print on the schematic buc do appear on screen. Other symbol-related attributes will work, but we recommend that you use the attributes listed here.

Note: We recommend that you do not use attributes 3, 4, 6, and 7 since they already have specific meanings to FutureNet translators. Using them for any other than their defined purpose constitutes a non-standard use and may cause problems with other FutureNet translators. Use any of the other attributes listed instead.

Note: If you already have libraries that use these attributes, you do not need to change them, but realize that they may not work properly with other translators.

CAUTION: The *-anum* option is not supported by the FutureNet to Edif Netlist Writer. Alternatives to *-anum* include changing the symbol identifier for a symbol in the FutureNet program or using the PID property in a part file to specify a different identifier for a symbol.

Option	Description
-n	Symbol Identifier Only — The **-n** option causes the Part File Utility to list only symbol identifiers and properties in the part file. No pins will be output.
-g	Generate Power and Ground Pins — The **-g** causes the Part File Utility to include power and ground pins in the part file. Note that this option is NOT compatible with -n.

4 Part Library Utility

The Part Library Utility (PLU) translates and combines one or more part files (.pf) into a single library (-pl) file. Figure 4-1 shows the relationship of the Part Library Utility to the FutureNet design process. The part library file is a non-text file and cannot be edited.

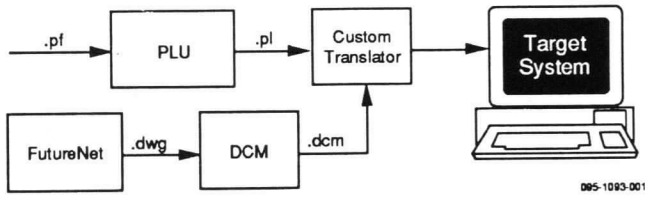


Figure 4-1
Part Library Utility Flowchart

Running the Part Library Utility

To create a part library, the minimum command line entry is:

plu *partfile*

Multiple input files can be specified if you want to combine a number of part files in a single part library.

plu *partfile1 partfile2 partfile3...partfilen*

Note: If there is a duplicate definition of a part in the part file, PLU issues a warning and uses the first definition, discarding the others. The .pf input files are processed in the order specified on the command line.

Command Line Options

The Part Library Utility uses the basic command line options in addition to several system-specific options.

Basic Command Line Options

Basic command line options are summarized below.

Option	Definition
-i*filename*	Specifies input file(s) (.pf)
-o*filename*	Specifies output file name (.pl)
+o	Resets output file name to default.
-e*filename*	Specifies output file for errors (.ple).
+e	Resets error file name to default.
-s	Sets program to silent mode.
+s	Resets program to non-silent mode.
-q	Sets program to query mode.
+q	Resets program to non-query mode.

Utility-specific Options

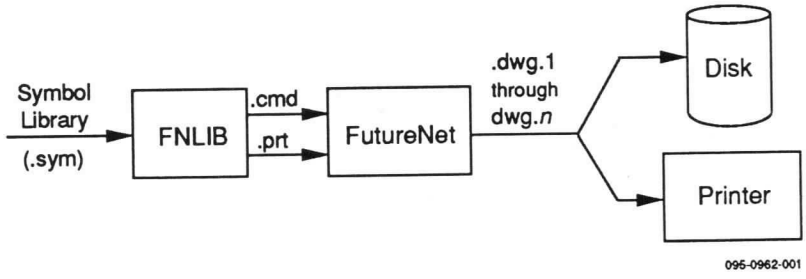
Two additional options are available for exclusive use with the Part Library Utility. The options are specified on the command line and explained below.

Option	Definition
-p	Path — The **-p***path* option specifies the path to search for .pf files when your files are not all in the current directory. The current directory, which is the default search directory, will always be searched.
-v	Verbose — Selecting verbose mode with the -v option causes the utility to display the name of the part file it is reading and the name of each part as it is added to the part library. Note that these messages are always placed in the error (.ple) file.

5 FutureNet Symbol Library Utility

The FutureNet Symbol Library Utility (FNLIB) documents each symbol in a FutureNet symbol library in the form of a FutureNet drawing. See Figure 5-1 for a flow chart description of the Symbol Library Utility.

Figure 5-1
Symbol Library Utility Flowchart



The symbols from the specified library are loaded into FutureNet drawing sheets. Included in the documentation are the name of the symbol in the FutureNet library, the symbol identifier for the part library, and the symbol-related and pin-related alphanumeric fields with associated layered text. Drawing sheets are assigned page numbers, and indexes by symbol library name and by symbol identifier are created on the last drawing sheets. The symbol identifier is derived from the concatenation of the following five properties: PART, VAL, STR, TOL, and PCHR.

The output of the FutureNet Symbol Library Utility consists of two FutureNet command files: *outputfile*.cmd and *outputfile*.prt. When the FutureNet command script, *outputfile*.cmd, is run through FutureNet Schematic Designer, a set of drawings documenting the symbol library is created. The drawings are consecutively named *output_file.number* for sheets with symbols and *output_file.inumber* for the indexes. The *output_file.prt* is a FutureNet command script that loads and prints the drawing sheets. See the FutureNet AUTO and EXEC commands and information on command files in your *FutureNet User Manual*.

Running the Utility

To create symbol library documentation, the minimum command line is:

fnlib *libraryname*

Note: If the fnlib environment variable is set (set fnlib=e:\dataio\fnlib), the fnlib executable only looks there for the library name. It checks the current directory only if the fnlib or dashlib environment variable is not set. If the fnlib environment variable is not set, the fnlib executable defaults to the old dashlib environment variable. If neither environment variable is set, the fnlib executable uses the current directory.

Command Line Options

The Symbol Library Utility uses the basic command line options in addition to several utility-specific options.

Basic Command Line Options

Basic command line options are summarized below.

Option	Definition
-i*filename*	Specifies input library (.sym). If a fully qualified file name is not given, the FNLIB environment variable specifies the directory path for the library.
-o*filename*	Specifies output file name (.cmd)
+o	Resets output file name to default.
-e*filename*	Specifies output file for errors (.lbe).
+e	Resets error file name to default.
-s	Sets program to silent mode.
+s	Resets program to non-silent mode.
-q	Sets program to query mode.
+q	Resets program to non-query mode.

Utility-specific Options

Option	Definition
-dsize	Drawing Size — By default, symbols are loaded onto a D-size drawing sheet. The *-dsize* option allows you to specify a different drawing sheet size. The *size* specification is a letter A through E, for drawing sheet size A through E. For example, if you want symbols documented on a C-size drawing sheet, specify **-dC** on the command line.

Note: ***A****-size drawings are often too small to allow documentation of the alphanumeric symbol and pin text fields.*

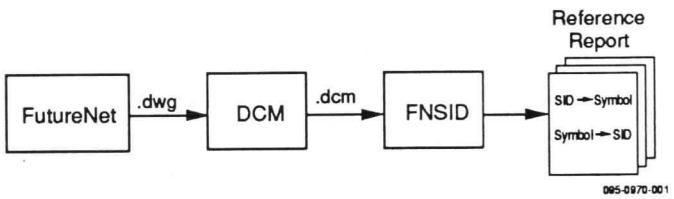
-t[s | p | n])	Text — The -t option allows you to choose which type of alphanumeric field text is documented, as shown below.

Option	Definition
-tn	No alphanumeric text documented.
-ts	Symbol-related text only.
-tp	Pin-related text only.
-tsp	Symbol-related and pin-related. The default mode is to document symbol-related and pin-related alphanumeric text.
-v	View — The -v option allows you to watch the documentation of the symbols as they are loaded onto the FutureNet drawing sheets.

6 FutureNet Symbol Identifier Utility

The FutureNet Symbol Identifier Utility (FNSID) documents the symbol identifiers used in a design. You can use this document to verify that all the symbols used in a design have an entry in your part file. See Figure 6-1 for a flow chart of the Symbol Identifier Utility.

Figure 6-1
Symbol Identifier Utility Flowchart



Running the Utility

To create a symbol identifier listing, the minimum command line entry is:

fnsid *dcmfile*

Command Line Options

The Symbol Identifier Utility uses the basic command line options in addition to several utility-specific options.

Basic Command Line Options

Basic command line options are summarized below.

Option	Definition
-i*filename*	Specifies the input library (.dcm)
-o*filename*	Specifies output file name (.sid)
+o	Resets output file name to default.
-e*filename*	Specifies output file for errors (.sde).
+e	Resets error file name to default.
-s	Sets program to silent mode.
+s	Resets program to non-silent mode.
-q	Sets program to query mode.
+q	Resets program to non-query mode.

Utility-specific Options

Option	Description
-a*num*	Attribute Number — The **-a***num* option allows you to specify your own symbol identifier for a symbol by instructing the utility to take the identifier for the symbol from the displayed text field with the specified attribute number. Symbol identifier entries are by default derived from the concatenation of the PART, VAL, STR, TOL, and PCHR properties and can be overridden using this option. When using this option, use the same attribute number for all symbols. For example, if you want a symbol to be identified by your company part number, then you would use the FutureNet Schematic Designer to create an alphanumeric field containing the company part number in the symbol and then assign the attribute number to the field.

Option	Description
-a*num*	The symbol-related attribute listed here are the legal attributes available for use with the -a option: 3 4 6 7 55-99 109-113 116-127 131 135 140 142-144 Attribute numbers above 99 are non-printing, displayed text attributes; they are probably the best choice since they do not print on the schematic but do appear on screen. Other symbol-related attributes will work, but we recommend that you use the attributes listed here.

Note: Attributes 3, 4, 6, and 7 already have specific meanings to FutureNet translators. We recommend that you not use them since their use for any other than their defined purpose constitutes a non-standard use and may cause problems with other FutureNet translators. Use any of the other attributes listed instead.

If you already have libraries that use these attributes, you do not need to change them, but realize that they may not work properly with other translators.

WARNING: The -a*num* option is not supported by the FutureNet to Edif Netlist Writer.

Option	Description
-t	Tool — The -t *tool* option specifies which properties for the specified tool will be used. For example, if a symbol has the PART attribute on display text **7400** and layered text **PCB:PART=SN7400**, with the option **-t**, the value **SN7400** would be used as the symbol identifier instead of **7400.**

Note: You may specify only one tool with the -t option.

7 *Error Messages*

The error messages listed here are for all utilities described in the manual. There are three levels of error messages; Fatal, Error, and Warning. Errors are listed here by level, and then alphabetically within levels. Words in italics, such as *filename*, represent information supplied by the utility. Messages that are self-explanatory are listed but not described.

Fatal Errors Terminate the program. The message provides information about the problem that caused the error. No data is lost when fatal errors occur.

Errors Do not terminate the program. The translator continues to run, but no files are created. Because errors do not terminate the program, they help you find design problems that need to be corrected before continuing. Once the errors have been corrected, run the translator again.

Warnings Indicate that design problems may be present or that your design is not what you expected it to be. For example, warnings can occur when two conflicting attributes refer to the same signal. Warnings do not prevent output files from being created.

Fatal Errors

Attribute *x* is not a symbol attribute	You have specified a valid attribute, but not a symbol attribute.
Attribute *attributename* is not a valid attribute	You have specified an attribute that does not exist in the FutureNet system.
File *filename* is not a valid DCM file.	The specified file is not a valid DCM file.
Invalid part library *filename*	The specified file is not a valid part library file.
Options -n (part names only and -g (power/ground pins) are not compatible	You have specified two mutually exclusive command line options.
Unable to open DCM file *filename*.	The indicated file is not in the current directory or on the path, the file is read-protected, or the file is corrupted.
Unable to open error file *filename*	The indicated file is not in the current directory or on the path, the file is read protected, or the file is corrupted.
Unable to open input file *filename*	The indicated file is not in the current directory or on the path, the file is read protected, or the file is corrupted.
Unable to open output file *filename*	The indicated file is not in the current directory or on the path, the file is read protected, or the file is corrupted.
Unable to open part library *filename*	The indicated file is not in the current directory or on the path, the file is read protected, or the file is corrupted.
Unable to create part library *filename* due to errors.	Errors occurred during processing that prevented creation of the part file.

Errors

File *filename*, line *linenumber*: Duplicate property specification *propertyname* ignored	While running plu, the same property was specified twice for a symbol or pin identifier.
File *filename*, line *linenumber*: Maximum number of symbol identifiers exceeded.	Maximum is 1000.
File *filename*, line *linenumber*: Maximum number of pin identifiers exceeded.	Maximum is 1000.
File *filename*, line *linenumber*: Missing ']' for *line*	There is a missing bracket] in the indicated line in the part file.

File *filename*, **line** *linenumber*: **No pin identifier after '##'**	## was found, but no pin identifier was specified.
File *filename*, **line** *linenumber*: **Property data too large, property** *propertyname* **ignored**	The properties data specified for the indicated symbol or pin identifier exceeded the maximum allowed. Property data for a given part or pin (tool:property=value) is limited to 4096 characters.
File *filename*, **line** *linenumber*: **Property exceeded 80 characters**	The indicated property entry exceeded the 80 character limit.
File *filename*, **line** *linenumber*: **Syntax error**	There is a syntax error in the indicated line.
File *filename*, **line** *linenumber*: **Syntax error property** *propertyname*	Can't parse property entry for indicated line. Check for missing equal (=) sign.
File *filename*, **line** *linenumber*: **Syntax error, too many #'s for line.**	Maximum is 3.
Syntax error for plu part property *propertyname*	Can't parse property entry for indicated line. Check for missing equal (=) sign.
Tool list exceeded limit of 50 tools	
Unable to open part file *filename*	The indicated file is not in the current directory or on the path, the file is read protected, or the file is corrupted.

Warnings

File *filename*, **line** *linenumber*: **'[ ]' found with no symbol identifier present.**	
File *filename*, **line** *linenumber*: **Ignoring duplicate part** *partname*	The same symbol identifier is specified in different part files.
File *filename*, **line** *linenumber*: **Duplicate pin specification**	Two identical pin identifiers were specified for the same symbol identifier.
File *filename*, **line** *linenumber*, **Duplicate specification for property** *propertyname*	While running pfu, the same property was specified twice for a symbol or pin identifier.
File *filename*, **line** *linenumber*: **Duplicate specification property** *propertyname* **tool** *toolname*	The same property was specified twice for a symbol or pin identifier and a particular tool.
File *filename*, **line** *linenumber*: **Duplicate specification property** *propertyname* **tool** *toolname* **symbol** *symbol reference number*	The same property was specified twice for a symbol or pin identifier and a particular tool for the designated symbol reference number.

File *filename,* **line** *linenumber*: **No symbol identifier after '###'**	A ### was found, but no symbol identifier was specified
File *filename,* **line** *linenumber*: **Syntax error property** *propertyname* **tool** *toolname*	Can't parse property entry for indicated line. Check for missing equal (=) sign.
More than one alpha field with attribute number X found	The **-a***num* option has been used to assign the same attribute to more than one alpha field on a single symbol. The translator uses the first text found. See the Command Line Options section of the Part File Utility for details on the **-a***num* option.
Syntax error property *propertyname* **symbol** *symbol reference number*	There is a syntax error in a property entry for the indicated symbol.

Index

D

E

F

G

FutureNet® to EDIF 200 Netlist Writer

User Manual

DATA I/O Corporation

096-0089-002

August 1991 096-0089-002

Data I/O Corporation
10525 Willows Road N.E., P.O. Box 97046
Redmond, Washington 98073-9746 USA
(206) 881-6444

Acknowledgments:

Table of Contents

5. Error Messages

Index

1 Introduction

The Electronic Design Interchange Format (EDIF) has been developed for users of computer-aided engineering and design tools to facilitate the transfer of data about a design between electronic design systems.

The FutureNet® to EDIF Netlist Writer (referred to as the Netlist Writer) accepts FutureNet drawings that have been processed by the Drawing Preprocessor and translates the resulting drawing connectivity model (.dcm file) into an EDIF version 2 0 0 netlist file.

This document describes the operation of the Netlist Writer: Keyword level 0, EDIF level 0.

EDIF Statement of Intent

The Netlist Writer is intended to produce a valid EDIF netlist that correctly describes the connectivity of the FutureNet drawings from which it was produced. No assumptions about target system requirements, configuration, or interpretation of the EDIF file are made.

Drawing Prerequisites

Requirements

PID (Part Identification) Property

The PID property is used to map part names between FutureNet and the target system. When a part in FutureNet has a different part name than the same part has in the target system, the PID property can be used to map the part names. The PID property can be specified in a part file or using property assignment statements in FutureNet, depending on which is easier.

When the PID property exists for a part, that property will be used as the EDIF part (cell) name.

Signals that Go On and Off the Design

Signals in hierarchical designs that are used as design ports, that is, signals that go on or off the design, must appear in the top level drawing(s) and must be assigned one of the following FutureNet attributes:

SIGI
SIGO
SIGU
SGNU
SIGB
PWRS
GND

Signals assigned these attributes are recognized as signals that enter and/or exit the design. The netlist writer creates logical design pins (ports in EDIF terminology) for these signals.

The use of these attributes must conform to the rules outlined in the chapter "Understanding FutureNet" in the *FutureNet User Manual*.

Symbol Use Restrictions

Before discussing use restrictions, a brief description of the information in the EDIF file is needed.

The first section in the file is a definition section. Each uniquely named part (symbol) in the design is listed in the definition section. An occurrence section follows the definition section and contains one entry for each occurrence (instance) of the parts defined in the definition section. An occurrence definition references the definition section for a description of the part, but adds information specific to the occurrence, like pin numbers and pin names. Obviously, you would expect to find more occurrences than definitions.

Restrictions

Symbol Compatibility

Since the translator uses the first occurrence of a uniquely named symbol (part) as the definition symbol, all occurrences that reference this definition must have the same number of pins and they must be in the same location.

Hierarchical Designs

The Netlist Writer accepts both hierarchical and flat designs as input, but creates only flat netlists.

Properties and Nets

Properties are not generated for NETS. Refer to the chapter "Understanding FutureNet" in the *FutureNet User Manual* for information on how properties are used in FutureNet.

Bus and Signal Names

Bus and signal names of the form **unsig*k*** and **unbus*k***, where ***k*** is an integer, are system generated names used to identify unnamed signals and buses. Do not use names of this form for signals and buses.

2 *Using the Netlist Writer*

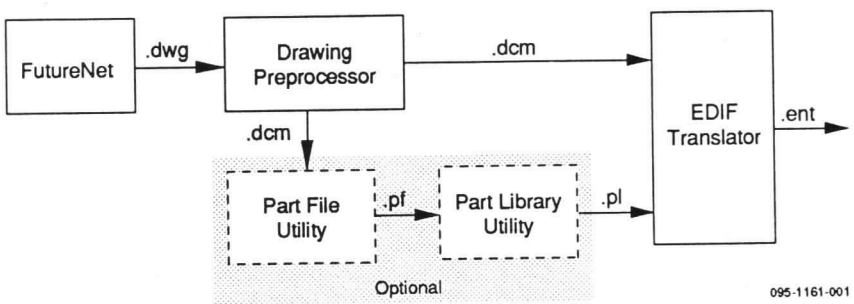
Before You Begin

Note that the Netlist Writer takes the following files as input:

1. Drawing Preprocessor **.dcm** file.
2. Optional Part Library Generator **.pl** file. The **.pl** file can be created using the FutureNet Part File Utility or it can be created manually . See the *FutureNet Companion Programs and Reference* manual for details.

System flow is outlined in Figure 2-1.

Figure 2-1
Netlist Writer System Flow



Running the Translator

The EDIF netlist writer is run from the system command line. It takes a .dcm file as input and is invoked by entering:

edif ***inputfile*** **[-t***toolname***] [-a***authorname***] [-p***filename***]**

followed by the appropriate command line options. The minimum entries are:

edif ***inputfile***

The translator uses two sets of command line options. The first set consists of the basic command line options available to all FutureNet Post processing tools. The second set consists of translator-specific options.

Basic Command Line Options

Basic command line options are summarized here.

Option	Definition
-i*filename*	Specifies input file, the design to translate (.dcm)
-o*filename*	Specifies output filename (.ent).
+o	Resets output filename to default.
-e*filename*	Specifies output file for errors (.eer).
+e	Resets error filename to default.
-s	Sets program to silent mode.
+s	Resets program to non-silent mode.
-q	Sets program to query mode.
+q	Resets program to non-query mode.

Translator-specific Options

Additional options available for exclusive use with the translator are explained below, along with instructions on how to activate or deactivate the option. These options are specified on the command line.

Tool Option (-t*toolname*)

-t specifies the name of the tool for which properties are to be generated in the EDIF file, in addition to the generic properties. No tool-specific properties are translated by default. Refer to the chapter "Understanding FutureNet" in the *FutureNet User Manual* for information on how properties are used and assigned in FutureNet.

Author Option (-a*authorname*)

-a specifies the author string to be used for the EDIF author construct. The default author string is **Data I/O Corporation**.

Part Library Option (-p*filename*)

-p specifies the filename for a part library (**.pl**) file. The default is no part library file.

You can specify a file that exists in the current directory or give a fully qualified file path name to a file in a different directory.

Pinpointing Problems

The flat EDIF netlist makes it difficult to tell in which drawing of the design an error occurred, or where a particular instance or net was defined. Nevertheless, the target system reading the EDIF file may generate warnings that may require you to find the relevant drawing.

To help you locate a particular instance of a part, a drawing cross-reference table is generated and placed in the error (**.eer**) file and also as comments near the top of the EDIF (**.ent**) file. The cross-reference table lists the drawing names, their drawing number, and their drawing reference number(s). Since all instances of a part include the FutureNet drawing reference number in their names, this table helps you locate the correct drawing.

3 Output File Format

The netlist file is an ASCII file that conforms to EDIF standard 2 0 0.

EDIF File Format

The following is a stylized example of the EDIF file format with descriptions. Note that this example is abbreviated and that not all levels of nesting are shown.

Note that property names are output in the EDIF netlist in uppercase, irrespective of how they were specified. Property entries are not case-sensitive in FutureNet.

EDIF File Format	Description
`(edif`	
`  (status`	
`  (comment "reference data`	Drawing cross-reference table
`  (library`	
`    (cell`	Cell (part) definition
`      (port`	Cell port definitions
`      (port`	
`      (port`	
`    (cell`	
`      (port`	
`      (port`	
`      (port`	
`    (cell`	Design cell
`      (instance`	Instance definition
`        (portRef`	Instance ports
`        (portRef`	
`        (portRef`	
`      (instance`	
`        (portRef`	
`        (portRef`	
`        (portRef`	
`      (net`	Netlist
`        (portRef`	Connected ports
`        (portRef`	
`          (instRef`	Instance reference
`        (portRef`	
`          (instRef`	
`    )`	
`      (net`	Netlist
`        (portRef`	Connected ports
`        (portRef`	
`          (instRef`	Instance reference
`        (portRef`	
`          (instRef`	
`    )`	
`  (design ...`	Design construct
`)`	

General Terms

Cell

This is the EDIF name for a symbol or part. Also, a single cell is used to represent the entire design.

Port

This is the EDIF name for a pin.

Property

Properties for instances and their pins are generated purely from properties in the drawing; the parts library file is not used. Note that, in EDIF, instance property values override cell property values.

Properties for the cell and port definitions come exclusively from the part library; properties in the drawing are not considered.

File Format Terms

Drawing Cross-reference Table

This table cross-references drawing names to drawing numbers and drawing reference numbers.

Cell Definitions

This section defines all of the unique cells (parts) used in the design. The cell name is the Post symbol identifier, which is the concatenation of the following properties: part, val, str, tol, and pchr.

Many symbols can have the same symbol identifier. When generating the cell definition, each symbol in the design is looked at by the translator; each time a symbol is found with a symbol identifier that has not been seen previously, a cell definition is created.

Note that when two different symbols have the same symbol identifier, they are treated as the same cell by the translator, as in a 7400 NAND and its Demorgan equivalent, which looks like an OR gate but has inverted inputs.

For symbols that do not have a symbol identifier, the translator generates a name. The generated name takes the form **part_*symbol reference number_drawing number*.** For example, a symbol with symbol reference number 3 on drawing 6 would be named **part_3_6**. Note that the drawing number referred to here is a unique number that is assigned by the Drawing Preprocessor to each drawing file in a design. In structured designs, root level drawings are assigned numbers first. Compare this to the drawing reference number in the Instance Definition entry of this section. Refer to the *FutureNet Post User Manual* for more information on the Drawing Preprocessor.

Properties for the cell and port definitions come exclusively from the part library; properties in the drawing are not considered.

Cell Port Definitions

A cell definition is generated each time the translator finds a symbol with a unique symbol identifier. This symbol is also used to generate the port (pin) definitions for the cell. This means that the pin identifiers from that symbol are used to name the cell ports.

Note that not all symbols with the same symbol identifier will have the same pin identifiers.

Design Cell

The actual design is described in a cell construct at the same level as all the other cells. The design cell is always the last cell.

The design cell will contain instances, which reference cells from the cell definition section, followed by nets, which reference the instances' ports.

Instance Definitions

Each symbol in the design will cause an instance to appear in this section. The instance will reference the cell definition that has its symbol identifier.

The instance name is the location designator (LOC property) followed by the symbol reference number and the drawing reference number. For example, a symbol with location designator U22, symbol reference number 5, and drawing reference number 9 would be named **U22_5_9**.

For symbols that do not have a LOC property, the translator generates an identifier. The generated identifier takes the form **inst_*symbol reference number_drawing reference number***. A symbol with no location designator, a symbol reference number of 6, and drawing reference number of 5 would be named **inst_6_5**. Note that the drawing reference number referred to here is a unique number assigned to each drawing occurrence in the design. Compare this to the drawing number in the Cell Definition entry of this section. Refer to the *FutureNet Post User Manual* for more information on the Drawing Preprocessor.

Properties for instances and their ports come exclusively from the drawing; properties from the parts library are not considered. Note, however, that according to the rules of EDIF, the instance inherits all of the properties of the cell definition. If the same property is specified in both places, the instance property overrides the cells.

The LOC property, if it exists, is output as an EDIF designator construct, not as a property.

Instance Ports

An instance port is a pin on a particular instance of a symbol. Recall that two symbols with the same symbol identifier may have different pin identifiers. When this occurs, the instance's pin identifier is output in an EDIF designator construct. To EDIF, the pin identifier is the pin identifier specified in the cell definition, not the pin identifier specified on the instance.

Note: The instance symbol must have the same number of pins and the pins must be in the same place as the symbol used to generate the cell definition. If not, an error will be generated.

Netlist

All nets (signals) are listed here, including a list of all the instances on each net.

The form for net names varies, depending on whether the net is named or unnamed, or whether it is on a named or unnamed bus. The four forms for net names are:

Nets

- Named Nets *netname_d*
- Unnamed Nets unsig_*k*

Nets on Buses

- Named Buses *busname_d_netname*
- Unnamed Buses unbus_*k_netname*

where *d* is the drawing reference number and *k* is an integer beginning at 1 and increasing by 1 with each new signal or bus found.

Connected Ports

This section refers to all of the ports (pins) on a net by name. If the port being referenced is on an instance rather than a top-level (design) port, then an instanceRef construct will be included so you can identify which instance it is that owns the port. Refer to the description of Instance Ports above for more information.

Instance Reference	This construct refers to an instance by name. Refer to the section on Instance Definitions above for more information on instance names.
Design Construct	The design construct simply states which cell represents the design. Design level properties are output here.

Implementation Conventions

Name Mapping

EDIF restricts the characters that can be used in names, for example, in cell, instance, and port names. When an illegal name is encountered, a new name is generated. All further reference to the object must use the new legal EDIF name. For a complete description of legal EDIF names, refer to the EDIF standard.

Legal names are generated by mapping illegal characters as follows:

Illegal Character	Mapped To
+	P
-	N
, (comma)	_ (underscore)

All other illegal characters are mapped to their hexadecimal representation of the ASCII code, and appear in the form *_dd*, where *dd* is a hexadecimal value.

Overbars

Pin and signal names with overbars will have a hyphen (-) appended to them by the Drawing Preprocessor. When the hyphenated names are run through EDIF, the hyphen will be mapped to an N, as described in the preceding section.

String Mapping

EDIF strings can contain any ASCII characters except the percent sign (%) and double quotes ("). When these are encountered, they are converted to *%dd%*, where *dd* is the hexadecimal value of the ASCII character codes.

Non-printable characters, like the ohm symbol (Ω), are mapped to *~ddd*, where *ddd* is the octal code for the character. If this string is then used as an EDIF name, the tilde (~) will be mapped as an ASCII character.

4 File Formatter

EDIF File Formatter Utility

The default output from the netlist writer is a human readable netlist file, that is, a formatted and indented file. These files can be large, requiring considerable storage space and processing time. The File Formatter can be used with the -p option to pack the file by removing white space and carriage returns.

Compressed files can be reformatted by running them through the File Formatter without specifying the -p option.

To compress a file, enter:

ef ***inputfile outputfile*** **-p**

If you are using the default filename extensions, **.ent** for input files and **.for** for output files, you will need to specify a different input or output name or extension when you unpack a file. If you do not make one or the other names different, the formatter will be trying to write to the same file it is reading from. To reformat a packed file, enter:

ef ***inputfile outputfile***

Command Line Options

The File Formatter uses the standard FutureNet user interface. You must specify an input file or you will be prompted for one. The output filename defaults to the input file with a **.for** (for format) extension. An error file (if needed) also uses the input filename with **.efe** extension. Basic command line options for the interface are:

Option	Definition
-i*filename*	Specifies input file (**.ent**).
-o*filename*	Specifies output filename (**.for**).
+o	Resets output filename to default.
-e*filename*	Specifies output file for errors (**.efe**).
+e	Resets error filename to default.
-s	Sets program to silent mode.
+s	Resets program to non-silent mode.
-q	Sets program to query mode.
+q	Resets program to non-query mode.

Formatter-specific Options

Pack (-p)

The -p option packs the netlist file by removing white space and carriage returns.

5 *Error Messages*

There are three levels of error message: Fatal, Error, and Warning. Errors are listed here by level, and then alphabetically within levels. Words in italics, such as *filename*, represent information supplied by the program.

- **Fatal Errors** — Terminate the program. The message provides information about the problem that caused the error. No data is lost when fatal errors occur.
- **Errors** — Do not terminate the program The translator will continue to run, but no files are created. Because errors do not terminate the program, they help you find design problems that need to be corrected before PCB layout. Once the errors have been corrected, run the translator again.
- **Warnings** — Indicate that design problems may be present or that your design is not what you expected it to be. For example, warnings can occur when two conflicting attributes refer to the same signal. Warnings will not prevent output files from being created.

Fatal

Drawing file *filename* not used to create the named model.

The named drawing file was not used to create the drawing connectivity model.

Error(s) detected in drawing. No EDIF file generated!

If errors are detected during EDIF processing, this message is generated just prior to program termination; the output file is deleted.

Fatal format error detected.

File *filename* may not be a legal EDIF file!

An attempt to format a non-EDIF file was made. Specifically, the parentheses in the file did not follow legal EDIF nesting.

File *filename* does not exist.

The named file has not been properly specified or has not been created.

File *filename* is not a valid DCM file.

The named file may have the correct name, but is not of the correct format for a DCM input file. You must provide a valid .dcm file for input.

Illegal file names, input and output files must be different.

Part library *filename* obsolete, rerun part library utility.

Out of memory.

The Netlist Writer was unable to allocate sufficient memory.

Unable to open DCM file *filename*.

The Netlist Writer was unable to open the specified .dcm input file. It is possible that the named file did not exist, was not in the current directory or specified directory path, or has the wrong protection.

Unable to open drawing file *filename*.

The Netlist Writer was unable to open the specified drawing file. It is possible that the named file did not exist, was not in the current directory or specified directory path, or has the wrong protection.

Unable to open error file *filename*.

The Netlist Writer was unable to open the specified error file. It is possible that the named file did not exist, was not in the current directory or specified directory path, or has the wrong protection.

Unable to open input file *filename*.

The Netlist Writer was unable to open the specified input file. It is possible that the named file did not exist, was not in the current directory or specified directory path, or has the wrong protection.

Unable to open output file *filename*.

The Netlist Writer was unable to open the specified output file. It is possible that the named file did not exist, was not in the current directory or specified directory path, or has the wrong protection.

Unable to read drawing file *filename*.

The Netlist Writer was unable to open the specified drawing file. It is possible that the named file did not exist, was not in the current directory or specified directory path, or has the wrong protection.

Error

Instance *instancename* has fewer pins than cell defintion.
Instance *instancename* was used to generate the definition.

Symbols with the same part name are required to have the same number of pins. The symbol used to generate the cell definition and the instance symbol had a different number of pins.

Pin *pinnumber* on instance *instancename* not found in cell definition.
Instance *instancename* was used to generate the cell definition.

Symbols with the same part name are required to have their pins in the same relative location.* The instance symbol had at least one pin in a location where there was no pin on the defining symbol (the symbol used to make the cell difinition).

* *The "same relative location" means the POE of the pin name must be in the same location relative to the symbol origin.*

Warning

Duplicate specification for property *propertyname*

Duplicate specification for property *propertyname* tool *toolname*

Duplicate specification for property *propertyname* tool *toolname* symbol *symbol reference*

Duplicate specification for property *propertyname* tool *toolname* symbol *symbol reference* pin *pinidentifier*

You have tried to specify two values for a property on a given object, for example, two PART properties on a given symbol. Note that this can also occur when the same *tool:property* combination is specified more than once.

No properties found for tool *toolname*

You have specified a tool with the -t option, but no properties have been specified for that tool.

Syntax error property *propertyname*

Property specification syntax is *property=value* or *tool:property=value.*

Refer to your FutureNet documentation for information on specifying properties.

Syntax error property *propertyname* **tool** *toolname*

Property specification syntax is *property=value* or *tool:property=value.*

Refer to your FutureNet documentation for information on specifying properties.

Syntax error property *propertyname* **tool** *toolname* **symbol** *symbol reference*

Property specification syntax is *property=value* or *tool:property=value.*

Refer to your FutureNet documentation for information on specifying properties.

Syntax error property *propertyname* **tool** *toolname* **symbol** *symbol reference* **pin** *pinidentifier*

Property specification syntax is *property=value* or *tool:property=value.*

Refer to your FutureNet documentation for information on specifying properties.

Index

E

F

P

S

T