the tool for software designers

PDL/81™

Installation and Tailoring Guide

(Version 2.0)

Caine, Farber & Gordon, Inc.　　Warren Point International Ltd.

RESTRICTED RIGHTS LEGEND

Comments or questions relating to this manual or to the subject software are welcomed and should be addressed to:

In North America:

Caine, Farber & Gordon, Inc.
1010 East Union Street
Pasadena, CA 91106
USA

Tel: (800) 424-3070 or
(818) 449-3070

Fax: (818) 440-1742

In the Rest of the World:

Warren Point International Ltd.
Babbage Road
Stevenage, Herts SG1 2EQ
England

Tel: 0438 316311

Fax: 0227 86521

Form Number: 9102-18

1 August 1988
4 January 1989
1 February 1989
18 July 1990
1 December 1991

Contents

Appendices

1. Introduction

This manual describes the method of installing the PDL/81 processor, Version 2.0, Edit Code 908 or later, on various common machines and operating systems. These include:

- UNIX™ systems (Chapter 2 and Chapter 3);
- Digital Equipment Corporation VAX™ computers under the VMS™ operating system (Chapter 4); and
- IBM PC computers under the PC-DOS operating system (Chapter 5).

Following these are chapters devoted to topics which are common to all of the versions.

CAUTION

> All of the installation discusions in this part are intended to be read and followed by experienced users who are familiar with the problems of software installation on the particular machine.

1.1 Related Publications

Other publications relating to the use of PDL/81 are:

- *PDL/81 Introduction and Invocation Guide* – a guide to invoking PDL/81 under various operating environments
- *PDL/81 Design Language Reference Guide* – a guide to using PDL/81 for producing software design documents
- *PDL/81 Ada Design Language Reference Guide* – a guide to using PDL/81 for Ada program design
- *PDL/81 Document Language Reference Guide* – a guide to using PDL/81 for producing various documents such as manuals and reports

- *PDL/81 Format Designers Guide* – A guide to developing new types of PDL/81 design and document styles

2. Binary Installation on UNIX Systems

This chapter discusses the installation of binary versions of the PDL/81 processor under the UNIX operating system. If you are installing a UNIX source version of the processor, see secref{UNIXS}.

This discussion assumes that the installation is being performed by an experienced UNIX user or systems programmer.

2.1 Supported Operating Environment

Generally, the UNIX version of PDL/81 requires a machine which runs a modern version of UNIX such as System V, Ultrix, or Berkeley 4.[23]bsd. On some such machines and UNIX variants, PDL/81 is supported in binary form.

2.2 Hardware Prerequisites

Approximately 500,000 bytes of disk are required to host the PDL/81 processor.

2.3 Binary Installation

The delivery medium for a binary installation will normally be a 9-track magnetic tape. In some cases, however, it may be one or more diskettes or other special types of recording media.

In general, installation is performed by logging in as *root*, mounting the delivery tape, and entering

```
cd /
tar xv
```

If other methods are required, a memorandum describing them will accompany the delivery.

The result of the installation will be to install the PDL/81 processor in */usr/bin*, the manual pages in */usr/man/man1*, and the style library in */usr/lib/pdl81*.

2.4 Modifying the Style Library

Several files in the style library directory (/usr/lib/pdl81) contain site dependent information. These files should be examined and the indicated modifications performed.

2.4.1 Defining Various Notices

A number of library files contain text which *must* be modified. The files are:

clphn.lib definitions to handle the fourth argument of the .Reverse command of the “manual” style

cpyrt.lib definitions to produce a copyright notice for the “manual” style

rstrct.lib definition of the Restricted Rights notice for the “manual” style

uid.lib definition of the user id string for the “design” and “design74” styles

uttl.lib definition of the name and address block for the “design” style

WARNING

THESE FILES ARE PROVIDED AS EXAMPLES ONLY. SEVERAL OF THEM CONTAIN TEXT WHICH MAY HAVE LEGAL IMPLICATIONS. NO LEGAL ADVICE IS INTENDED BY THEIR INCLUSION. IT IS STRONGLY SUGGESTED THAT COMPETENT LEGAL ADVICE BE OBTAINED BEFORE ADOPTING YOUR OWN CONTENTS FOR SUCH FILES.

2.4.2 Style and Device Tailoring

The distributed style and device files will usually produce the desired results. However, as you gain experience with PDL/81, you may want to tailor style and device definitions to meet specific device and site requirements. Style tailoring is discussed in Chapter 7 and device tailoring is discussed in Chapter 6.

2.4.2.1 Changing the Standard Page Depth

All distributed device files define a page depth of 66 lines. This may be changed, for each device, by changing the definition of the “pdepth” number register in the appropriate device file. For example, the page depth supported by the “pr10” device (*/usr/lib/pdl81/pr10.d*) may be changed from 66 lines to 64 lines by changing

```
#{nr;pdepth;66}
```

to

```
#{nr;pdepth;64}
```

3. Source Installation on UNIX Systems

This chapter describes the procedures for installing a source version of PDL/81 under the UNIX operating system. It should operate under most current versions, such as System V, Ultrix, and 4.[23] bsd. However, since local modifications are frequently made in these operating systems, an experienced UNIX system programmer should be consulted prior to performing the installation and if difficulties are encountered during the installation.

Before installing PDL/81, this manual should be read and understood in its entirety. It will almost certainly be necessary to modify various portions of the "makefile" to reflect local conditions.

3.1 Software Prerequisites

PDL/81 is intended to be installed using the "make" utility. It is written entirely in the C programming language.

3.2 Hardware Prerequisites

Approximately 2,000,000 bytes of disk are required during the installation process. After installation, about 750,000 bytes of disk will be required.

3.3 A Note to the Reader

This manual is intended for an experienced UNIX user who is familiar with the "make" utility, shell procedures, the C compiler and C programming language, the file system, and any local operating system modifications. A good grounding in UNIX terminology is also assumed.

WARNING

EXCEPT WHERE EXPLICITLY STATED TO THE CONTRARY, THE OPERATIONS DESCRIBED IN THIS MANUAL SHOULD NOT BE PERFORMED WHILE LOGGED IN (OR su'ed) AS root, bin, OR ANY OTHER PRIVILEGED USER.

3.4 Restoring the Distribution Tape

The first file of the distribution tape contains the entire source of the PDL/81 processor and is recorded in *tar* format. To restore it, mount the tape, position to a directory in which you want to place the source and do

```
tar xv
```

if you mounted the tape on your system's default drive. If you mounted it on some other drive, say |/dev/rmt1h|, you will need to specify the drive as in

```
tar xvf /dev/rmt1h
```

The following directory structure will be created

```
pdl81.dir         (root directory for PDL/81)
   source.d       (source directory)
      lib         (style library)
      man         (manual pages)
      samples     (sample PDL/81 input)
```

3.5 What To Do If Tar Fails

There are still a surprising number of incompatible *tar* formats in current UNIX usage. If you attempt to restore the *tar* file as above and find that you cannot, you should still be able to restore the PDL/81 processor if it was delivered on 9-track magnetic tape.

The second and third files on a 9-track magnetic tape delivery contain the source for the *sarin* archive restoration program and all of the PDL/81 source in the form of a *sarin* archive. These may be restored as described in Appendix A.

3.6 Generating PDL/81 With the “Make” Utility

The PDL/81 distribution is tailored for generating PDL/81 using the “make” utility as described in this chapter.

All examples in this chapter assume that you have restored the distribution tape and extracted the PDL/81 files as described in Section 3.4. It is also assumed that your working directory is “pdl81.dir/source.d”.

This section is best read with a listing of the “make” description file in hand. This file is named “makefile” and resides in the “source.d” directory.

3.6.1 Setting Config.h Configuration Definitions

Before generating PDL/81 for the first time, the *config.h* file should be examined and modified as discussed in this section.

3.6.1.1 Specifying the PDL/81 Style Library Directory

The symbol DEFPATH should be set to the path to the directory that will hold the PDL/81 style library after installation. Since this string will be prepended to the name of any library file, it should have a trailing solidus (/). As distributed, the symbol is defined as

```
#define   DEFPATH    "/usr/lib/pdl81/"
```

If DEFPATH is changed, the LIB make symbol (Section 3.6.2.3) should be changed to correspond.

3.6.1.2 Specifying the Temporary Files Directory

The symbol TMPDIR should be set to the name of the directory in which PDL/81 should create temporary files. As distributed, the symbol is defined as

```
#define   TMPDIR    ""
```

Since this string will be prepended to the name of any temporary file, it should have a trailing solidus (/).

3.6.1.3 Specifying User Style Library Options

If the symbol UDEFOK is set to zero, PDL/81 will not accept the -u or -b invocation option. If it is set to one, these options are allowed. Refer to the *PDL/81 Format Designers Guide* for a description of these options. As distributed, UDEFOK is defined as

```
#define   UDEFOK    1
```

3.6.2 Setting Makefile Configuration Definitions

Before generating PDL/81 for the first time, the “Configuration Definitions” portion of *makefile* should be examined and modified as discussed in this section.

3.6.2.1 Specifying C Compiler Options

This section describes the various symbols which are used to supply options to the C compiler when compiling PDL/81.

3.6.2.1.1 Name of the C Compiler

The definition of the symbol CC should be set to the local name of the C compiler. This is usually “cc”. The distributed version will have

```
CC = cc
```

3.6.2.1.2 C Compiler Optimization

As distributed, the makefile contains the line

```
COPT = -O
```

which causes the C compiler optimizer to be executed. If trouble is encountered in building or testing PDL/81, it might be a good idea to comment out this line and rebuild PDL/81 without optimization.

3.6.2.1.3 Other C Compiler Options

The CFLAGS symbol, distributed as

```
CFLAGS =
```

may be defined to contain any other desired C compiler options of local interest (usually none).

3.6.2.2 Specifying Linker Options

This section describes symbols which are used to supply options to the UNIX linker (ld) when linking PDL/81.

3.6.2.2.1 Split I/D Linking

The SPLITID symbol is distributed as

```
SPLITID = -i
```

which will cause the linker to produce an executable image (text file) using separate code and data spaces. This is required for proper operation of PDL/81 on many systems – particularly 16-bit ones. Some UNIX systems, such as Berkeley 4.x bsd, use a demand paging mechanism and the split i/d concept is not relevant to that environment. When generating PDL/81 under such systems, comment out this line as

```
#SPLITID = -i
```

so that the “-i” option will not be passed to the linker.

3.6.2.2.2 Other Linker Options

The LDFLAGS symbol, distributed as

```
LDFLAGS =
```

may be set to any other desired linker options of local interest (usually, none).

3.6.2.3 Specifying the PDL/81 Style Library Directory

The LIB symbol defines the name of the directory to hold the PDL/81 document style library. The distributed version has the definition

```
LIB = /usr/lib/pdl81
```

If you change this definition, be sure to change the DEFPATH symbol in the *config.h* file to correspond (Section 3.6.1.1).

3.6.2.4 Specifying the PDL/81 Execution Directory

The symbol BIN defines the name of the directory to hold the executable PDL/81 file. The distributed version has the definition

```
BIN = /usr/bin
```

3.6.2.5 Specifying Manual Page Installation

The file “man/pdl81.man” contains the “nroff” source of the UNIX manual pages for PDL/81. These are written using the “man” macro set. The file “man/pdl81.cat” contains the manual pages after processing by “nroff” and is suitable for display with “cat”.

To cause one of these to be installed, the symbol MAN must be defined to perform the installation. As distributed, the following line appears:

```
MAN  = cp man/pdl81.man /usr/man/man1
```

This should be examined, and possibly changed, to reflect local manual page location and naming requirements. If it is not desired to install the manual pages, comment out the line.

3.6.3 Modifying the Style Library

Several files in the style library directory (pdl81.dir/source.d/lib) contain site dependent information. These files should be examined and the indicated modifications performed.

3.6.3.1 Defining Various Notices

A number of library files contain text which *must* be modified. The files are:

lib/clphn.lib	definitions to handle the fourth argument of the .Reverse command of the “manual” style
lib/cpyrt.lib	definitions to produce a copyright notice for the “manual” style
lib/rstrct.lib	definition of the Restricted Rights notice for the “manual” style
lib/uid.lib	definition of the user id string for the “design” and “design74” styles
lib/uttl.lib	definition of the name and address block for the “design” and “design74” styles

WARNING

THESE FILES ARE PROVIDED AS EXAMPLES ONLY. SEVERAL OF THEM CONTAIN TEXT WHICH MAY HAVE LEGAL IMPLICATIONS. NO LEGAL ADVICE IS INTENDED BY THEIR INCLUSION. IT IS STRONGLY SUGGESTED THAT COMPETENT LEGAL ADVICE BE OBTAINED BEFORE ADOPTING YOUR OWN CONTENTS FOR SUCH FILES.

3.6.3.2 Style and Device Tailoring

The distributed style and device files will usually produce the desired results. However, as you gain experience with PDL/81, you may want to tailor style and device definitions to meet specific device and site requirements. Style tailoring is discussed in Chapter 7 and device tailoring is discussed in Chapter 6.

3.6.3.2.1 Changing the Standard Page Depth

All distributed device files define a page depth of 66 lines. This may be changed, for each device, by changing the definition of the “pdepth” number register in the appropriate device file. For example, the page depth supported by the “pr10” device may be changed from 66 lines to 64 lines by changing

```
    #{nr;pdepth;66}
```

to

```
    #{nr;pdepth;64}
```

in the “pr10.d” file in the “lib” directory.

3.6.4 Compiling and Linking PDL/81

The command

```
make
```

will compile those modules which have changed since the last time “make” was used (or which depend on files that have changed) and will link PDL/81 placing the executable module in the file “pdl81”. Some utility routines will also be generated and placed in the “lib” directory from which they will be later moved to the library directory.

A full compilation of PDL/81 can be forced by

```
make fullbuild
```

3.6.5 Testing PDL/81

Prior to installation, the newly built PDL/81 should be tested by using the shell procedure “xpdl81” in the form

```
xpdl81 [options, etc.]
```

Before using “xpdl81” for the first time, it must be made executable by a command such as

```
chmod 700 xpdl81
```

The “xpdl81” command will execute the version of PDL/81 found in the current directory and will use the subdirectory “./lib” as the location of the style library.

The “pdl81.dir/source.d/samples” directory contains sample input to PDL/81. For example, the first design found in the *PDL/81 Design Language Reference Guide* can be produced by

```
xpdl81 samples/design1 >output-file
```

3.6.6 Installing PDL/81

The compiled, linked, and tested PDL/81 may be installed for general use by the command

```
make install
```

This should be run while logged on as the owner of the directories defined by the

LIB (see Section 3.6.2.3) and BIN (see Section 3.6.2.4) configuration symbols. At most sites, this will require logging on as “bin”.

WARNING

THIS COMMAND WILL CAUSE REPLACEMENT OF THE FILE NAMED pdl81 IN THE DIRECTORY DEFINED BY THE BIN CONFIGURATION SYMBOL AND OF FILES IN THE DIRECTORY DEFINED BY THE LIB CONFIGURATION SYMBOL. BE SURE THAT YOU INTEND THIS TO HAPPEN BEFORE EXECUTING THIS COMMAND.

During execution of “make install”, an attempt will be made to create the directory defined by the LIB configuration symbol. If this attempt fails with an error number of 17, it is probably all right since that means the directory already exists (e.g., from a previous execution of “make install”).

4. Installation on VMS Systems

There are two methods for installing the PDL/81 processor under VMS – one, described in this chapter, uses the VMSINSTAL procedure. The other, described in Appendix B, is more difficult and time consuming but requires neither VMSINSTAL nor SYSTEM privileges.

We recommend the use of VMSINSTAL. Installing PDL/81 with VMSINSTAL results in a processor which is better matched to your system than would be the case if the alternate installation method is used. Installation is much simpler and should require less than one hour.

4.1 Hardware and Software Prerequisites

This version of PDL/81 will run on Version 4.7 or later of VMS. It will probably also run on earlier versions but this installation method may not be compatible with earlier version of VMSINSTAL. If you experience problems with this method, you should install using the alternate method described in Appendix B.

The installation requires at least 3,000 blocks of disk. There are no other hardware requirements that are unique to PDL/81.

CAUTION

> This installation method must be run from a privileged account – typically SYSTEM. It will modify the system files
>
> ```
> SYS$LIBRARY:DCLTABLES.EXE
> SYS$HELP:HELPLIB.HLB
> ```
>
> and will create the files and directories
>
> ```
> SYS$SYSTEM:PD8_V20.EXE
> SYS$SYSROOT:[SYSLIB.PD8_V20_LIB]
> SYS$SYSROOT:[SYSTEST.PD8_V20_SAMPLES]
> ```
>
> If you do not want to do this, do not install by this method. The method described in Appendix B provides for a non-privileged, but less

convenient, installation.

4.2 Performing the Installation

This section discusses the steps required in using VMSINSTAL to install PDL/81. A transcript of an actual installation session is used to illustrate the steps. In the transcript, *italics* are used to show material entered by the user. The symbol *<CR>* stands for the carriage return key.

4.2.1 Preparation

Before you begin the installation you should be sure that your system is adequately backed up in case any severe problems occur. Since the installation will modify the DCL tables, the current set should be saved by a command such as

```
$ copy sys$library:dcltables.exe  sys$library:dcltables_old
```

4.2.2 Getting Started

After logging in as SYSTEM, the default device and directory should be set to SYS$UPDATE. Then, VMSINSTAL should be invoked, giving it the product name to be installed and the device on which the distribution tape is to be mounted. The product name for this release of PDL/81 is PD8020. Instead of giving the name, you may just give an asterisk. After starting up, VMSINSTAL will ask you if you are satisfied with the state of your system backup. If you are not, reply *no* and the process will be terminated. If you reply *yes*, processing will continue.

```
$  set def sys$update
$  @vmsinstal * msa0

        VAX/VMS Software Product Installation Procedure V5.0-2

It is 11-MAR-1990 at 11:31.
Enter a question mark (?) at any time for help.

* Are you satisfied with the backup of your system disk [YES]? <CR>
```

4.2.3 Mounting the Distribution Tape

You will next be asked to mount the tape. Do so, and then enter *y*.

```
Please mount the first volume of the set on  MSA0:.
* Are you ready? y <CR>
%MOUNT-I-MOUNTED, PDL81 mounted on _MSA0:

The following products will be processed:

  PD8 V2.0


        Beginning installation of PD8 V2.0 at 11:32

%VMSINSTAL-I-RESTORE, Restoring product saveset A ...
```

4.2.4 Display of Notices

Several notices will now be displayed. On completion, you will be asked if you wish to continue. If you are satisfied with the restrictions presented in the notices, enter *y* (or just hit return). If not, enter *n* and processing will be terminated.

```
***** Begin Installation of PDL/81 V2.0.908 *****

The PDL/81 processor and its associated source code, source listings,
and License Key are confidential and proprietary information of Caine,
Farber, & Gordon, Inc.  Use, duplication, and disclosure may only be made
in accordance with your Agreement with Caine, Farber & Gordon, Inc.

PDL/81 is an Unpublished Work.  Copyright (C) 1981, 1987, 1990 by Caine,
Farber & Gordon, Inc.  This copyright notice and any others appearing on the
work do not evidence any actual or intended publication of such work.

The following statement is required by the Commercial Software --
Restricted Rights clause at FAR 52.227-19:  Unpublished -- rights reserved
under the copyright laws of the United States.

RESTRICTED RIGHTS LEGEND
------------------------

Use, duplication, or disclosure by the Government of the United States is
subject to restrictions as set forth in subparagraph (c)(1) of the
Commercial Computer Software -- Restricted Rights clause at FAR 52.227-19 or
subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer
Software clause at DFARS 252.227-7013.  Contractor/Manufacturer is Caine,
Farber & Gordon, Inc., 1010 East Union St., Pasadena, CA 91106.  (818)
449-3070.

* Do you wish to continue [YES]?<CR>
```

4.2.5 Naming the PDL/81 Command

The normal name of the command used to run PDL/81 is PDL81. If, for some reason, you need another name, you can provide it now. We suggest, however, that you just enter return when asked.

```
    By default, PDL/81 will be installed with "PDL81" as the DCL verb used to
    invoke the processor.  You may, however, specify a different verb.
    Remember that all verbs in your DCL tables must be unique in the first
    four characters!

* Verb to use for PDL/81 processor [PDL81]: <CR>
* You chose PDL81 as the verb.  Is that ok [YES]? <CR>
```

4.2.6 Choosing the Run-Time Library

The executable PDL/81 image should normally be built using your system's shareable C library images. You may, however, use the C object library, instead. You are now given the choice.

```
    PDL/81 will normally be built using the RTL (Run Time Library)
    shareable images.  You may, however, link with the RTL object
    library.

* Do you wish to use the RTL shareable images [YES]? <CR>
```

4.2.7 Acknowledging Required Changes

You will now be reminded that after installation is complete you must read and modify certain files in the library. These files all contain information that should be changed to reflect the requirements of your company.

```
    WARNING:  AFTER INSTALLATION IS COMPLETE, YOU *MUST* READ AND MODIFY
    THE CONTENTS OF THE FOLLOWING FILES IN SYS$SYSROOT:[SYSLIB.PD8_V20_LIB]:

        CLPHN.LIB  CPYRT.LIB  RSTRCT.LIB  UID.LIB  UTTL.LIB

    SEE THE CHAPTER ON VMS INSTALLATION IN THE "PDL/81 INSTALLATION AND
    TAILORING GUIDE".

* Do you wish to continue [YES]? <CR>
```

4.2.8 Selecting Installation Verification

You will now be asked if you wish to run the installation verification procedure when installation is finished. You should probably elect to do so.

```
    This kit has an installation verification procedure (IVP).  After
    installation, it will be left in

        SYS$SYSROOT:[SYSTEST.PD8_V20_SAMPLES]PD8_IVP.COM

    You may invoke this at any time to verify the the installation.

* Do you want to run the IVP after the installation [YES]? <CR>
```

4.2.9 Purging of Files

During installation, any replaced files will normally be purged. You are now given a chance to keep them.

```
* Do you want to purge files replaced by this installation [YES]? <CR>
```

4.2.10 Remainder of the Installation

At this point all questions have been asked and installation will proceed automatically.

```
    This ends the conversational part of the PDL/81 installation.  The
    remainder of the installation process should not require attention.

Linking PDL/81 with the shared RTL ...
Establishing DCL entry and help file ...
Establishing PDL/81 executable image ...
Creating PDL/81 style library ...
%VMSINSTAL-I-SYSDIR, This product creates system disk directory
VMI$ROOT:[SYSLIB.PD8_V20_LIB].
Creating PDL/81 samples ...
%VMSINSTAL-I-SYSDIR, This product creates system disk directory
VMI$ROOT:[SYSTEST.PD8_V20_SAMPLES].
%VMSINSTAL-I-MOVEFILES, Files will now be moved to their target directories...
```

4.2.11 Installation Verification

If you asked that the installation verification procedure be run, you should see

```
    Start Installation Verification Procedure for PDL/81

        This procedure will run the installed PDL/81 on the file
        sys$common:[systest.pdl81]design3.pdl and will place the
        output in design3.lis in that directory.  The output should
        look like that in design3.lis-delivered.

    Executing: PDL81/DEVICE=PR10N  DESIGN3


PDL/81 V2.0(vb).908 (5332-PD8)
[style: DESIGN]
[device: PR10N]
[pass 1]
1 [source: VMI$ROOT:[SYSTEST.PD8_V20_SAMPLES]DESIGN3.PDL;1]
2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
[pass 2]
1 .1 [source: VMI$ROOT:[SYSTEST.PD8_V20_SAMPLES]DESIGN3.PDL;1]
2 3 .1 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
[trees]
21 .1
[dindex]
22 .1 .2 .3 .4
[sindex]
23 .1 .2 .3
[rindex]
24 .1
[cic]
25 .1 .2
[stats]
26

    Verification Procedure executed successfully.


        Installation of PD8 V2.0 completed at 11:53


        VMSINSTAL procedure done at 11:55
```

4.3 Post-Installation Tailoring

At this point, a few modifications should be made to files in the

```
SYS$SYSROOT:[SYSLIB.PD8_V20_LIB]
```

directory. Most of these changes are optional. However, the ones relating to notices should be read and modified.

4.3.1 Defining the Notices

A number of files contain text which must be modified before PDL/81 is used for actual production work:

`CLPHN.LIB` definitions to handle the fourth argument of the .*Reverse* command of the *manual* style.

`CPYRT.LIB` definitions to produce a copyright notice for the *manual* style.

`RSTRCT.LIB` definition of the Restricted Rights Notice for the *manual* style.

`UID.LIB` definition of the identification string for the *design* and *ada* styles.

`UTTL.LIB` definition of the name and address block for the *design* and *ada* styles.

NOTE

> These files are provided as examples only. Several of them contain text which may have legal implications. No legal advice is intended by their inclusion. It is strongly suggested that competent legal advice be obtained before adopting your own contents for such files.

4.3.2 Defining the Default Style

As distributed, PDL/81 will process with the *design* style unless the /STYLE qualifier is used. This default can be changed by modifying the file named DEFAULT. It's distributed contents will be something like:

```
#{lib;DESIGN}
```

For example, if you wish to have the *manual* style as your default, change the definition line to

```
#{lib;MANUAL}
```

4.3.3 Defining the Default Output Device

As distributed, PDL/81 will process with the *pr10* output device definition unless the /DEVICE qualifier is used. This default can be changed by modifying the file named DEFAULT.D. It's distributed contents will be something like:

```
#{lib;PR10.D}
```

For example, if you wish to have the *hp12* output device as your default, change the definition line to

```
#{lib;HP12.D}
```

4.3.4 Style and Device Tailoring

The distributed style and device files will usually produce the desired results. However, as you gain experience with PDL/81 you may want to tailor style and device definitions to meet specific device and site requirements. Style tailoring is discussed in Chapter 7 and device tailoring is discussed in Chapter 6.

NOTE

> Some VMS printer devices may not support underscoring as defined in the distributed device definitions. See Section 6.2 for a discussion. Also, because some devices will not properly support form feeds or horizontal tabs, you may have to modify the settings of the *.noff* or *.notab* parameters as discussed in Section 6.1.

4.3.5 Changing the Standard Page Depth

All distributed device files define a page depth of 66 lines. This may be changed for a particular device by changing the definition of the *pdepth* number register in the device definition file for that device. For example, the page depth supported by the PR10 device may be changed to 60 lines per inch by changing the definition line in the PR10.D file from

```
#{nr;pdepth;66}
```

to

```
#{nr;pdepth;60}
```

4.4 The PDL/81 Sample Directory

The directory

```
SYS$SYSROOT:[SYSTEST.PD8_V20_SAMPLES]
```

contains a number of PDL/81 design source files that may be useful in understanding how to use the PDL/81 processor. This directory also contains the Installation Validation Procedure (PD8_IVP.COM) and a copy of the Command Language Definition file (PD8.CLD).

4.5 Removing PDL/81

To remove the PDL/81 processor from your system, delete the the following files and directories:

- `SYS$SYSROOT:[SYSEXE]PD8_V20`
- `SYS$SYSROOT:[SYSLIB.PD8_V20_LIB]*.*`
- `SYS$SYSROOT:[SYSLIB.PD8_V20_LIB]`
- `SYS$SYSROOT:[SYSTEST.PD8_V20_SAMPLES]*.*`
- `SYS$SYSROOT:[SYSTEST.PD8_V20_SAMPLES]`

This will delete all of PDL/81 except for the command definition in the system DCL tables. You may leave the command definition installed, if you wish, since attempting to use it will only result in an error message. However, if you wish to remove that last vestige of PDL/81, you may do so by using the /DELETE qualifier of the Command Definition Utility (SET COMMAND).

WARNING

Do not do this unless you know what you are doing! If you damage the system DCL tables you may find it difficult to recover them.

5. Installation on DOS Systems

This chapter describes the procedures for installing PDL/81 under the DOS operating system. It should operate under Release 3.10 and later versions of DOS. The installation of PDL/81 should be performed by someone who is familiar with installing software under DOS.

Before installing PDL/81, this manual should be read and understood in its entirety.

5.1 Software Prerequisites

Release 3.10 of DOS, or a later compatible version, is required for proper installation and operation of his version of PDL/81.

5.2 Hardware Prerequisites

An IBM PC, PC-XT, or PC-AT computer with at least 512K bytes of RAM is required for proper installation and use of PDL/81.

5.3 Restoring the Distribution Diskettes

The DOS version of PDL/81 is distributed on one or more diskettes. To restore them, first create a directory to hold the PDL/81 files. It is typically named `pdl81lib`. For example, to create this on your `c` drive, do

```
mkdir c:\pdl81lib
```

The, copy the contents of the distributions diskette(s) to this directory. For example, to copy the contents to the directory created in the previous step, do

```
copy a:*.*  c:\pdl81lib
```

for each of the diskettes in the distribution.

The directory will contain the PDL/81 processor, `pdl81.exe`, several sample input files, `*.pdl`, and the various style files.

5.4 Installing the Processor

After the diskettes have been restored, a few additional installation steps are necessary.

5.4.1 Installing the Processor Executable

If you have a standard directory to hold executable programs (e.g., c:\bin), move the PDL/81 executable file, pdl81.exe, to that directory. Otherwise, place the style library directory path (e.g., c:\pdl81lib) into your path environment variable.

5.4.2 Defining the Style Library

The PDL/81 processor must be able to locate the style library. Place the definition

```
set pdl81lib=path\
```

into your autoexec.bat file. Note that the trailing "\" is required. For example, if the directory is c:\pdl81lib, use the definition

```
set pdl81lib=c:\pdl81lib\
```

Strictly speaking, if the directory is named pdl81lib and is located on your default drive, the definition is not required. It is good practice to define it anyway in case you start using other default drives in the future.

5.4.3 Specifying the Directory for Temporary Files

By default, the PDL/81 processor creates temporaries in the current directory.

An environment symbol may also be used to specify the location. If present, the processor will use the value of pdl81tmp to locate the directory for temporary files. For example, executing the command

```
set pdl81tmp=b:\tmp\
```

will cause PDL/81 to place its temporary files in directory tmp on device b. An assignment such as this could be placed in the autoexec.bat file.

5.4.4 Saving Disk Space

If disk space is at a premium, it is usually possible to delete some of the files in the library directory. If you are only using the PDL/81 DESIGN style, you only need the following style library files:

```
default  default.d  design  uid.lib  uttl.lib
```

plus the .d file(s) corresponding to the devices which you wish to use.

5.4.5 DOS Configuration Changes

Some of the PDL/81 styles require a number of simultaneously open files. For correct operation, DOS should be configured to allow at least twenty such files. This may be accomplished by placing

```
files=20
```

in the DOS `config.sys` file.

Because of the amount of input and output performed by PDL/81, a reasonable number of disk buffers should be configured. We suggest the value of 12. This may be accomplished by placing

```
buffers=12
```

in the DOS `config.sys` file.

5.4.6 Testing PDL/81

The library directory contains several files with the extention of `pdl` which may be used to test the installation. For example. the command

```
pdl81 design3.pdl >d3.lis
```

will create a design output similar to the first sample design shown in Appendix E of the *Design Language Reference Guide*.

5.5 Making Site-Dependent Modifications

Before using PDL/81 in production, various site-dependent modifications must be made.

5.5.1 Specifying Default Output Device Type

As distributed, the default output device is “pr10n”. This is defined by the line

```
#{lib;pr10n.D}
```

in the *PDL81LIB\default.D* file. This line may be changed to reflect the desired default output device type. For example, to specify “hp12” as the default device type, the line

```
#{lib;hp12.D}
```

might be used.

5.5.2 Specifying Default Style

As distributed, the default style is “design”. This is defined by the line

```
#{lib;design}
```

in the *PDL81LIB\default* file. This may be changed to reflect the desired default style. For example, to specify “manual” as the default style, the line

```
#{lib;manual}
```

might be used.

5.5.3 Other Style Library Modifications

Several other files in the library directory PDL81LIB contain site dependent information. These files should be examined and the indicated modifications performed.

5.5.3.1 Defining Various Notices

A number of library files contain text which *must* be modified. The files are:

clphn.lib definitions to handle the fourth argument of the .Reverse command of the “manual” style

cpyrt.lib definitions to produce a copyright notice for the “manual” style

rstrct.lib definition of the Restricted Rights notice for the “manual” style

uid.lib definition of the user id string for the “design” style

uttl.lib definition of the name and address block for the “design” style

WARNING

THESE FILES ARE PROVIDED AS EXAMPLES ONLY. SEVERAL OF THEM CONTAIN TEXT WHICH MAY HAVE LEGAL IMPLICATIONS. NO LEGAL ADVICE IS INTENDED BY THEIR INCLUSION. IT IS STRONGLY SUGGESTED THAT COMPETENT LEGAL ADVICE BE OBTAINED BEFORE ADOPTING YOUR OWN CONTENTS FOR SUCH FILES.

5.5.3.2 Style and Device Tailoring

The distributed style and device files will usually produce the desired results. However, as you gain experience with PDL/81, you may want to tailor style and device definitions to meet specific device and site requirements. Style tailoring is discussed in Chapter 7 and device tailoring is discussed in Chapter 6.

5.5.3.2.1 Changing the Standard Page Depth

All distributed device files define a page depth of 66 lines. This may be changed, for each device, by changing the definition of the “pdepth” number register in the appropriate device file. For example, the page depth supported by the “printer” device may be changed from 66 lines to 64 lines by changing

```
    #{nr;pdepth;66}
```

to

```
    #{nr;pdepth;64}
```

in the “pr10n.D” file in the PDL81LIB directory.

6. Device Tailoring

PDL/81 supports a number of device types which may be used with the “design”, “letter”, “manual”, “memo”, and “text” styles. These device types, which are communicated to PDL/81 by the “-d” invocation option under UNIX and DOS and by the /DEVICE qualifier under VMS, are:

diablo10 a Diablo 1620 printer operating at 10 characters per inch on 8.5 by 11 inch paper

diablo12 a Diablo 1620 printer operating at 12 characters per inch on 8.5 by 11 inch paper

diablo a generic Diablo 1620 printer (distributed to be the same as “diablo12”)

hp10 an HP Laserjet printer operating at 10 characters per inch.

hp12 an HP Laserjet printer operating at 12 characters per inch and using the 9288G (Legal Elite) font.

pr10 a normal printer operating at 10 characters per inch on 14 by 11 inch paper for designs and 8.5 by 11 inch paper for documents

pr10n a normal printer operating at 10 characters per inch on 8.5 by 11 inch paper for both designs and documents

pr12 a normal printer operating at 12 characters per inch on 8.5 by 11 inch paper for both designs and documents

pr16 a normal printer operating at 16 (or similar) characters per inch on 8.5 by 11 inch paper for both designs and documents (useful for printing wide designs on narrow paper and for printing drafts of documents to be later printed at 12 characters per inch)

printer a generic printer device (distributed to be the same as “pr10”)

default device to be used if a “-d” invocation option (or /DEVICE qualifier) is not given (distributed to be the same as “printer”)

6.1 Tailoring the Device Specifications

Each device described above is defined by the contents of a file of the given name with “.d” appended. These device files may be found in the style library library directory.

Each file defines number registers and strings which specify the characteristics of that device. The following attributes are required for all styles:

cpi characters per inch (10, 12, 16)

lpi lines per inch (6 or 8)

pdepth page depth in lines (e.g., 66 for 11 inch paper at 6 lpi)

pwidth page width in characters. This should be the maximum width paper you plan to use in the printer. The actual printing width is controlled by other attributes (e.g., rmarg for documents and boxrm for designs).

.po page offset in characters (how far to shift each output line to the right before printing it)

.cwidth character width in basic units. For a device with constant width spaces (e.g., a line printer), this should be “one”. For a device with variable width spaces (e.g., a Diablo), this should be the number of horizontal raster units (of the size resulting from the *gon specification, below) which make up a character.

*gon if the device supports variable width spaces and the “.cwidth” number register is set to other than “one”, this string must be the escape sequence to turn on graphics mode

*goff if the *gon string is used, this must be the escape sequence to turn off graphics mode

*eol sequence of characters to output at end of each printed line. If empty, a newline is postpended to each line.

*bol sequence of characters to be output at the beginning of each printed line. If empty, nothing special is output.

$$devinit a string to be sent to the standard output before any other output is done. Typically, this string would set special device modes.

$$devterm a string to be sent to the standard output after all other output is done. Typically, this string would restore a device to the installation standard configuration.

*bu string used to print the bullet character (must result in net horizontal motion of +*1* character and net vertical motion of 0).

*f1b, *f1, *f1e font control strings for producing underscored output.

*f2b, *f2, *f2e font control strings for producing bold face output.

*f3b, *f3, *f3e font control strings for producing underscored, bold face output.

.noff 0 = PDL/81 may generate form feed characters in the output; 1 = PDL/81 may not generate form feed characters in the output.

.notab 0 = PDL/81 may generate horizontal tab characters in the output; 1 = PDL/81 may not generate horizontal tab characters in the output.

The following attributes are required by the Document Language styles:

lmarg	left margin of printed line of body text. This should normally be "1". Use ".po" (or adjust the printer) to leave an empty left margin.
rmarg	right margin of printed line for body text (must be less than or equal to "pwidth").
mcpos	position on line to print a marginal character (must be less than "lmarg" or greater than "rmarg").

The following attribute is required by the "design" style:

boxrm	column number in which to print the right edge of a segment box.

6.2 Underscores and Overstriking

PDL/81 indicates underscoring and overstriking by appropriate use of the backspace character. However, some printer spoolers do not properly handle this character for devices which do not support its use. In such cases, the ".nobs" number register can be set to have a value of "1" which will cause PDL/81 to convert uses of the backspace character to an appropriate number of over-printed lines.

This problem is compounded by the fact that certain types of printers can only handle certain specific over-printing sequences. For example, the basic underscoring sequence in the distributed PDL/81 is "underscore", "backspace", "character" which allows display of the output on a normal CRT terminal without loss of actual characters. Some printers, however, require the sequence "character", "backspace", "underscore" if the characters are, in fact, to be underscored on the output. These sequences are controlled in the distributed PDL/81 device definitions by the font definition strings "*f1", "*f2", and "*f3". If the printed results are not what you expect, you may need to change the output order in these strings. The general scheme in these strings is:

`'_`	means "underscore"
`^h`	means "backspace"
`'#`	means the character to be underscored

Finally, some types of printers cannot overstrike except to underscore. For these, you will need to redefine the "*bu" string to output only a single character.

6.3 Device Attributes for Each Device

The table below shows the distributed values for the device attributes for each of the above devices. Entries consisting of a "-" indicate that the item is defined as null for that device. The parenthesized numbers refer to the following notes:

1 outputs `^[3` to place the Diablo in graphics mode.

2 outputs `^[4` to place the Diablo in normal mode.

3 places Diablo in 10-cpi mode with left margin at column 10.

4 places Diablo in 12-cpi mode with left margin at column 12.

5 returns Diablo to 10-cpi mode with left margin at column 10.

6 bullet defined to be o, “backspace”, +.

7 font 1 (underscore) is “underscore”, “backspace”, “character”.

8 font 2 (bold face) for printers is same as the base font.

9 font 2 (bold face) for Diablos done by overstriking four times.

10 font 3 (bold face, underscored) is font 2 preceded by “underscore”, “backspace”.

11 mcpos is set to three columns right of the right margin (rmarg).

12 sets horizontal motion index to 1/60th inch.

13 restores horizontal motion index.

14 resets printer, initializes font and mode, and sets a page offset.

15 resets printer.

16 outputs *start underscore* escape sequence.

17 outputs *stop underscore* escape sequence.

18 produces bold face by overstriking with a slight offset.

The *hp12* style also supports two special fonts: 13, which access the special characters such as “trademark” and “copyright”; and 14, which access the line drawing set. Details of these fonts can be found by examining the `hp12.d` file.

	hp10	hp12	pr10	pr10n	pr12	pr16	diablo10	diablo12
cpi	10	12	10	10	12	16	10	12
lpi	6	6	6	6	6	6	6	6
pdepth	63	63	66	66	66	66	66	66
pwidth	120	120	120	120	120	120	120	120
.po	0	0	0	0	0	0	0	0
.cwidth	6	5	1	1	1	1	6	5
*gon	(12)	(12)	–	–	–	–	(1)	(1)
*goff	(13)	(13)	–	–	–	–	(2)	(2)
$$devinit	(14)	(14)	–	–	–	–	(3)	(4)
$$devterm	(15)	(15)	–	–	–	–	(5)	(5)
*bu	*	*	(6)	(6)	(6)	(6)	(6)	(6)
*f1b	(16)	(16)	–	–	–	–	–	–
*f1	–	–	(7)	(7)	(7)	(7)	(7)	(7)
*f1e	(17)	(17)	–	–	–	–	–	–
*f2b	–	–	–	–	–	–	–	–
*f2	(8)	(18)	(8)	(8)	(8)	(8)	(9)	(9)
*f2e	–	–	–	–	–	–	–	–
*f3b	–	(16)	–	–	–	–	–	–
*f3	–	(18)	(10)	(10)	(10)	(10)	(10)	(10)
*f3e	–	(17)	–	–	–	–	–	–
.noff	0	0	0	0	0	0	0	0
.notab	1	1	0	0	0	0	0	0
lmarg	1	1	1	1	1	1	1	1
rmarg	65	78	65	65	78	78	65	78
mcpos	(11)	(11)	(11)	(11)	(11)	(11)	(11)	(11)
boxrm	72	83	106	72	83	106	72	83

7. Style Tailoring

The distributed styles have a large number of parameters which may be changed to tailor the styles for particular uses and devices. This chapter discusses the procedures for performing this tailoring. Note that gross changes, such as page size and layout, should be performed by modifying the device definitions as described in Chapter 6.

7.1 Tailoring the “Design” and “Ada” Styles

The “design” and “ada” styles provide for easy modification of the user ID field, the title page name and address block, and the various default settings.

7.1.1 Changing the User ID Field

The user ID field is a short string which is printed at the far left of the header line on each page of the design (other than the cover page). The definition is contained in the “uid.lib” file in the PDL/81 style library directory. As distributed, the operative portion of this file is

```
#{ds;user_id;CFG, INC.}
```

The first fourteen characters of this string (“CFG, INC.” as distributed) constitute the user ID field. If desired, this string may be changed to provide local identification.

7.1.2 Changing the Cover Page Name and Address Block

The name and address block, as printed at the top of the cover page for a design, is contained in the “uttl.lib” file in the PDL/81 style library directory. As distributed, the operative portion of this file is

```
#{ds;user_title;{\
    #{ce;CAINE, FARBER & GORDON, INC.}\
    #{ce;1010 EAST UNION STREET}\
    #{ce;PASADENA, CALIFORNIA 91106}\
}}
```

Each call on the "ce" primitive causes its argument to be printed centered on a line. These lines may be changed (and new ones added) to supply the proper local name and address.

7.1.3 Common Default Parameters

Near the beginning of each design style is a block of definitions labeled *CONFIGURATION DEFINITIONS*. These may be modified to establish new defaults:

show set to 1 to report progress on the standard error file; set to 0 to suppress the progress report.

security the default security classification of all designs processed with this style. If set to NONE, security banners will not be provided by default.

.stree set to 0 to provide long trees; set to 1 to provide short trees.

_trees set to 1 to produce a listing of the calling trees; set to 0 to suppress the listing.

_dindex set to 1 to produce a data index; set to 0 to suppress the data index.

_sindex set to 1 to produce a segment index; set to 0 to suppress the segment index.

dbchar set to the single character to use as the box character for data segments.

ebchar set to the single character to use as the box character for external segments.

sbchar set to the single character to use as the box character for flow segments.

tbchar set to the single character to use as the box character for text segments.

sbox set to 1 to produce "short" segment boxes (no right edges); set to 0 to produce "long" segment boxes.

_kwcase set to "off" to print keywords in the same case as entered; set to "upper" to print keywords in upper case; set to "lower" to print keywords in lower case.

_kwfont set to the number of the font to be used to print keywords.

_lblindent
set to 1 for inline labels; set to 0 for flush left labels (as in PDL/74).

li_indent set to indentation amount for itemized and enumerated lists.

vi_indent set to indentation amount for verb lists.

The distributed definitions for these parameters are:

Item	Setting
show	1
security	NONE
.stree	1
_trees	1
_dindex	1
_sindex	1
dbchar	D
ebchar	X
sbchar	*
tbchar	#
sbox	0
_kwcase	upper
_kwfont	0
_lblindent	1
li_indent	8
vi_indent	16

7.1.4 Defaults for Complexity Measurement

The defaults for complexity measurement are established in the style files and may be changed there.

Whether or not complexity is measured by default is controlled by the “_cmplx” number register. If set to 0, the default is “NoComplexity”, otherwise it is “Complexity”. As distributed, the default is “NoComplexity”.

The default maximum allowable segment complexity is established by setting the “_cmplx_warn” number register. As distributed, the value is *6*.

7.1.5 Defaults for Requirements Tracking

The installation default for showing requirements is controlled by the “ShowReq” number register. If the value is 0, requirements will not be displayed with the individual flow segments. If the value is 1, requirements will be shown. As distributed, the value is 1.

The installation default for the Requirements Index is controlled by the “_rindex” number register in the *design* and *ada* styles. If the value is 1, a requirements index is printed; if the value is 0, a requirements index is not printed. As distributed, the value is 1.

7.1.6 Defaults for Consistency Checking

The installation default for the calls-in-context list is controlled by the “_cic” number register. If it has the value 0, a calls-in-context list will not be printed. If it has the value 1, a list will be printed. As distributed, the value is 0.

7.1.7 Defaults for Flow Figure Checking

Keyword checking is controlled by the “_kwnesterr” number register. The file contains the definition

```
#{nr;_kwnesterr;n}
```

where *n* may be 0 to disable checking; 1 to enable checking with error messages directed to the design document; or 2 to enable checking with error messages directed to both the standard error file and the design document. As distributed, the value is 2.

7.1.8 Defaults for Flow Figure Enhancement

The default for flow figure enhancement is controlled by the “_kwmatch” number register. A value of 0 means do not enhance flow figures; a value of 1 means enhance flow figures.

The character to be used for flow figure enhancement is set in both the style files and, optionally, in the device definition files. The following three definitions are used in the style files:

_kwvchar this string is set to the character to use for flow figure enhancement. A null or blank means that the device-specific character should be used or, in the absence of such, the character in the “_kwvdflt” string. As distributed, this string is null.

_kwvfont this number register specifies the font to use in the absence of a device-specific definition. As distributed, the value is 0 (use base font).

_kwvdflt this string contains the character to use in the absence of both a “_kwvchar” and a device-specific definition. As distributed, it is defined to be “|”.

The device-specific enhancement definitions may be used in cases where the device has special characteristics, such as a line-drawing character set, that may be used to improve flow figure enhancement. The definitions are

kw_v this string contains the character to be used for enhancement.

kw_vfont this number register specifies the font to be used for enhancement.

As distributed, the *hp12* device definition specifies these to use the line drawing character set of the *Legal Elite* font cartridge.

7.1.9 Defaults for Design Index

Several tailoring options have been added to allow changes in the format of the various design indexes:

- If the number register _sxlcol is set to 1, the segment index will have the column of line numbers; if set to 0, it will not.
- If the number register _sxlref is set to 1, the segment index will display line numbers in the cross-references; if set to 0, it will not.

- If the number register _dxlcol is set to 1, the data index will have the column of line numbers; if set to 0, it will not.
- If the number register _dxlref is set to 1, the data index will display line numbers in the cross-references; if set to 0, it will not.
- If the number register _ciclcol is set to 1, the calls-in-context index will have the column of line numbers; if set to 0, it will not.

7.1.10 Defaults for Security Banners

The default security banner style is controlled by the "_secstyle" number register which may be set to the values:

0 Center the security classification with the project name on the left and the sheet count on the right.

1 Center the security classification with the project name on the left and the sheet count on the right for odd-numbered sheets; project name on the right and the sheet count on the left for even-numbered pages.

2 Center the project name with the security classification on the left at the top of the page and on the right at the bottom of the page.

The distributed value is 0.

7.1.11 Changing Keywords and Adding New Ones

At the end of the *CONFIGURATION DEFINITIONS* section are the definitions of the primary and secondary keywords. These may be modified and new ones may be added.

7.1.11.1 Defining Primary Keywords

The "kw" function defines new keywords and has the form

```
#{kw;name;pre;post;complexity;class;code;flags}
```

where the arguments are

name the keyword name.

pre optionally signed integer giving the number of indentation stops to indent prior to printing the line which begins with the keyword.

post optionally signed integer giving the number of indentation stops to indent after printing the line which begins with the keyword.

complexity integer specifying the cyclomatic complexity associated with this keyword.

class used in flow figure checking (Section 7.1.11.3).

code used in flow figure checking (Section 7.1.11.3).

flags flags to associate with the keyword. The only currently defined flags is "1" which suppresses reference collection for the remainder of the line which starts with the keyword.

For example, the function

```
#{kw;case;-1;1}
```

will define “case” to be a keyword which will be printed one stop to the left of the current position. Following lines will be printed one stop to the right of the position in which the keyword is printed. With this definition, the sequence

```
do case type
case open:
...
case close:
...
case delete:
...
enddo
```

will result in

```
DO CASE type
CASE open:
    ...
CASE close:
    ...
CASE delete:
    ...
ENDDO
```

7.1.11.2 Defining Secondary Keywords

Secondary keywords are defined by the “skw” function which has the form

```
#{skw;name;complexity}
```

where the arguments are

name the keyword name.

complexity integer specifying the cyclomatic complexity associated with this keyword.

For example

```
#{skw;loop}
```

will define “loop” to be a secondary keyword. Note that a word may be both a keyword and a secondary keyword.

7.1.11.3 Keyword Classes and Codes

Flow figure checking is controlled by *class/code* pairs which are associated with each keyword. These pairs are defined using the “kw” text function.

The classes are used to distinguish between flow figures and are positive numbers. The codes are used to distinguish among types of keywords within a given flow figure and are in the range 0–5, inclusive. The codes are used to access the state table

	0	**1**	**2**	**3**	**4**	**5**
0	0	1	4	5	5	5
1	0	1	2	3	3	3
2	–	–	–	–	–	–
3	0	1	2	6	3	3
4	0	1	2	6	3	3
5	0	1	2	6	6	6

where the column headings correspond to the *code* of an incoming keyword, the row headings correspond to the *code* at the top of the *keyword stack*, and the table entries are *actions* to be performed.

The *codes* may be thought of as corresponding to

0 As an incoming keyword, this is one that doesn’t make any change in the structure (such as *RETURN* or *UNDO*). Since these are never pushed on the stack, a *code* of 0 on the stack means the stack is empty.

1 A figure-opening keyword (such as *DO* or *IF*).

2 A figure-ending keyword (such as *ENDDO* or *ENDIF*).

3 An intermediate keyword that may only occur once following a figure-opening keyword.

4 An intermediate keyword that may occur any number of times after a *code* 2 or 3 keyword. *ELSEIF* is an example.

5 An intermediate keyword that may occur once after a *code* 2, 3, or 4 keyword and may not be followed by a *code* 3 or 4 keyword. *ELSE* is an example.

A separate *class* should be assigned to each flow figure. For example, the *IF...ENDIF* figure might be *class* 1 and the *DO...ENDDO* figure might be *class* 2.

The *actions* are

0 Do nothing.

1 Push the *class* and *code* of the incoming keyword onto the stack.

2 If the incoming *class* matches that on the top of the stack, pop the stack. Otherwise, issue an error message.

3 If the incoming *class* matches that on the top of the stack, pop the stack and push the *class* and *code* of the incoming keyword onto the stack. Otherwise, issue an error message.

4 Issue an error message.

5 Issue an error message.

6 Issue an error message.

7.2 Tailoring the Document Language Styles

The “letter”, “manual”, “memo”, and “text” document language styles have a number of tailoring parameters. They can be divided into those which appear in each of the styles and those which appear in only one or a few styles.

7.2.1 Common Tailoring Parameters

The tailoring parameters which appear in each of the document language styles are:

Show	set to 1 to report progress on the standard error file; set to 0 to suppress the progress report.
FgnStyle	set to 0 to number figures and tables sequentially within the whole document; set to 1 to restart numbering for each level-one heading.
CapLevel	highest numbered heading level to capitalize.
DS_Indent	indentation for displays.
EjLevel	highest level numbered heading to start a new page.
Li_Indent	indentation for itemized and enumerated lists.
Nt_Indent	indentation for “notes”, “Warnings”, and “Cautions”.
OddChap	set to 0 if level-one headings which cause an eject should not be forced to be on an odd numbered page; set to 1 if the skipped even numbered page should *not* be printed; set to 2 if the skipped even numbered page should be printed with headings and footings; set to 3 if the skipped even numbered page should be printed without headings and footings.
Pp_Indent	indentation for first line of paragraphs.
Ri_Level	lowest numbered heading level to be run-in.
Vi_Indent	indentation for verb lists.
H_Start	number of first page to receive page headings.
F_Start	number of first page to receive page footers.
_Just	set to 0 if default for body justification is “no”; set to 1 if default for body justification is “yes”.
_Els	amount of extra line spacing to be used in filled text.
PH_Font	base font for page headers.
PF_Font	base font for page footers.
FgN_Font	base font for number portion of a figure or table caption.
FgT_Font	base font for text portion of a figure or table caption.
H1_Font	base font for level 1 headings.

H2_Font base font for level 2 headings.

H3_Font base font for level 3 headings.

H4_Font base font for level 4 headings.

H5_Font base font for level 5 headings.

H6_Font base font for level 6 headings.

H7_Font base font for level 7 headings.

The table below shows the settings of these parameters for the various document language styles. Entries indicated as "–" are not defined in the specified style.

	letter	manual	memo	text
Show	1	1	1	1
FgnStyle	0	1	0	0
CapLevel	1	2	1	2
Ds_Indent	8	8	8	8
EjLevel	0	1	0	1
Li_Indent	8	8	8	8
Nt_Indent	8	8	8	8
OddChap	0	1	0	0
Pp_Indent	0	5	0	5
RiLevel	3	4	3	4
Vi_Indent	16	16	16	16
H_Start	2	–	2	1
F_Start	2	–	2	1
_Just	0	1	1	1
_Els	0	0	0	0
PH_Font	0	0	0	0
PF_Font	0	0	0	0
FgN_Font	0	0	0	0
FgT_Font	0	0	0	0
H1_Font	-1	-1	-1	-1
H2_Font	-1	-1	-1	-1
H3_Font	-1	-1	-1	-1
H4_Font	-1	-1	-1	-1
H5_Font	-1	-1	-1	-1
H6_Font	-1	-1	-1	-1
H7_Font	-1	-1	-1	-1

7.2.2 Tailoring Parameters for the “Letter” Style

The “letter” style has several parameters which are set in the {it DEVICE DEPENDENT DEFINITIONS} section of the style file:

Sig_Pos number of characters to the left of the right margin to start the signature block.

Ra_Pos vertical position to start return address (or date if there is no return address).

Ad_Pos vertical position to start the address block.

The table below shows the distributed settings of these parameters.

	10 cpi	**12 cpi**	**6 lpi**	**8 lpi**
Sig_Pos	30	30		
Ra_Pos			8	11
Ad_Pos			19	26

7.2.3 Tailoring Parameters for the “Manual” Style

The “manual” style has several parameters which are set in the {it DEVICE DEPENDENT DEFINITIONS} section of the style file:

Fn_VPos vertical position of form number (must be before title block window).

Fn_HPos horizontal position of form number.

Tb_Top vertical position of first line in title block window.

Tb_Depth depth of title block window.

Tb_Rm right margin of title block window.

The table below shows the distributed settings of these parameters.

	10 cpi	**12 cpi**	**6 lpi**	**8 lpi**
Fn_VPos			7	9
Fn_HPos	60	73		
Tb_Top			14	18
Tb_Depth			14	18
Tb_Rm	46	56		

There are also a number of other parameters which are specific to the “manual” style:

Security default security classification for documents processed by this style. If set to “NONE”, security banners will not be provided by default.

TtpSec set to 0 if security banners are not to appear on title page or reverse; set to 1 if security banners are to so appear.

SecStyle set to 0 for centered classification, project on left, sheet number on left; set to 1 for same as 0 but swap project and sheet number on alternate pages; set to 2 for centered project with classification on left at the top and right at the bottom.

SecPfx a sequence of characters which will *not* appear as the first characters in any output line when security banners are in effect.

TtpStyle set to 0 for title page intended for a cut-out cover; set to 1, otherwise.

TocStyle set to 0 for non-hierarchical format; set to 1 for hierarchical format.

FglLevel set to 0 if a list of figures is not to be produced; set to 1 if this list is to be produced.

TblLevel set to 0 if a list of tables is not to be produced; set to 1 if this list is to be produced.

Indexing set to 0 if an index is not be produced; set to 1 if it is to be produced.

IToc set to 0 if the table of contents and lists of figures and tables are to be printed at the end of the document (saves processing time); set to 1 if they are to be produced in-line following the title page reverse.

OddChapNotice
set to 0 if skipped, even-numbered pages (under control of “OddChap”) are not to carry the “OddChapStr” notice; set to 1 if they are to carry the notice.

OddChapStr
string to be printed as a notice under control of “OddChapNotice”.

OddList set to 0 if lists of figures and tables are not to be forced to start on an odd numbered page; set to 1 if they are to be so forced (under control of “OddChap”).

TocLevel highest numbered heading level to be placed in the table of contents.

The distributed settings of these parameters are shown below.

Item	Setting
Security	NONE
TtpSec	1
SecStyle	0
SecPfx	~??%@;
TtpStyle	0
TocStyle	0
FglLevel	1
TblLevel	1
Indexing	1
IToc	0
OddChapNotice	0
OddChapStr	[THIS PAGE INTENTIONALLY LEFT BLANK]
OddList	1
TocLevel	4

7.2.4 Tailoring of “Manual” Style Notices

The following files contain templates for the notices which may be printed on the title page reverse:

clphn.lib a notice describing how the document was printed and referencing any trademarks that are used.

cpyrt.lib a copyright notice and optional permission to copy.

rstrct.lib a “Restricted Rights Legend”.

These files *must* be examined and modified to reflect local requirements.

WARNING

> These files are provided as examples only. No legal advice is intended by their inclusion. It is suggested that competent legal advice be obtained before these local modifications are made.

7.2.5 Tailoring Parameters for the “Memo” Style

The “memo” style has several parameters which are set in the “DEVICE DEPENDENT DEFINITIONS” section of the style file:

Hb_VPos vertical position of start of header block.

Hb_HPos horizontal position of start of header block.

To_Pos vertical position to print the “To” line.

The table below shows the distributed settings of these parameters.

	10 cpi	12 cpi	6 lpi	8 lpi
Hb_VPos			5	7
Hb_HPos	31	38		
To_Pos			19	26

7.2.6 Tailoring Parameters for the “Text” Style

The tailoring parameters specific to the “text” style are:

Security
: default security classification for documents processed by this style. If set to “NONE”, security banners will not be provided by default.

SecPfx
: a sequence of characters which will *not* appear as the first characters in any output line when security banners are in effect.

OddChapNotice
: set to 0 if skipped, even-numbered pages (under control of “OddChap”) are not to carry the “OddChapStr” notice; set to 1 if they are to carry the notice.

OddChapStr
: string to be printed as a notice under control of “OddChapNotice”.

The distributed settings of these parameters are:

Item	**Setting**
Security	NONE
SecPfx	~??%@;
OddChapNotice	0
OddChapStr	[THIS PAGE INTENTIONALLY LEFT BLANK]

A. Alternate UNIX Tape Restoring

There are still a number of incompatible *tar* formats in current UNIX usage. If you attempt to restore the *tar* file as described in Chapter 3 and find that you cannot, you should still be able to restore the PDL/81 processor if it was delivered on 9-track magnetic tape.

The second and third files on a 9-track magnetic tape delivery contain the source for the *sarin* archive restoration program and all of the PDL/81 source in the form of a *sarin* archive. These may be restored as described in this Appendix.

A.1 Restoring the Distribution Tape

PDL/81 is distributed in a manner intended to be very portable. It does not depend upon the availability of any particular tape archiving program. This chapter describes the format of the distribution tape and discusses the steps required for restoring a tape and preparing the restored files for generation of a PDL/81.

A.1.1 Tape Format

PDL/81 is distributed on one reel of 9-track, standard, magnetic tape containing two files of information:

1. A small C source program, named “sarin”, which may be used to extract the PDL/81 files from the archive contained in the second tape file; and
2. An archive file containing the PDL/81 source.

A.1.1.1 Structure of the PDL/81 Source Archive

The archives consist of a sequence of lines, each of which is terminated by a newline. Lines whose first three characters are $ are *control lines*. The possible control lines are:

```
{$}C commentary
```

which causes the commentary to be displayed when the archive is processed;

```
{$}D directory-name
```

which causes the named directory to be created by the “mkdir” command;

```
{$}F file-name
```

which causes the named file to be created;

```
{$}E
```

which causes the file named in the last `$F` control line to be closed; and

```
{$}Z
```

which terminates the archive. Any lines which are not control lines are just source and are written to the file named in the last “{$}F” control line.

A.1.2 Restoring the Magnetic Tape

A distribution in the form of an industry-standard magnetic tape will have the format:

```
source for sarin utility
<EOF>
PDL/81 source archive
<EOF>
<EOF>
```

All data blocks on the tape are 512 bytes long.

The information files may be restored by a shell sequence such as:

```
(dd of=junk;
 dd of=sarin.c;
 dd of=sarc) </dev/rmt0h
```

where *rmt0h* is the raw interface to the tape drive that you are using. The `junk` file will contain the *tar* image and should be removed.

A.1.3 Compiling the Sarin Utility

The sarin utility may be used to extract the PDL/81 source files from the archive. It should be compiled by

```
cc sarin.c -o sarin
```

which will leave an executable version of the utility in the file “sarin”.

Because all tape blocks are 512 bytes long, it is possible that the last block of the sarin.c file will contain one or more trailing ASCII nulls. This should not cause any problems. However, if the C compiler complains about illegal characters which appear to be at the end of the file, the shell sequence

```
ed sarin.c
w
q
```

should remove the trailing nulls.

A.1.4 Extracting PDL/81 Files From the Archives

Once the sarin utility has been compiled, the commands

```
sarin -t <sarc
```

will display a listing of all archived files and directories. The listing will be written on the standard error file.

Actual extraction of the PDL/81 files may be performed by the commands

```
sarin <sarc
```

These will create the directory “pdl81.dir” and restore all of the PDL/81 files and subdirectories into it. A progress log will also be displayed on the standard error file.

Restoring the archives will create the following directory structure:

```
pdl81.dir          (root directory for PDL/81)
    source.d       (source directory)
        lib        (PDL/81 style library)
        man        (UNIX Manual pages)
        samples    (sample PDL/81 input)
```

Refer to the file “files.pr” in the “source.d” directory for a description of these directories and their contents.

During extraction, warning messages will be issued if the “pdl81.dir” directory or any of its subdirectories already exist but extraction will proceed using the existing directories.

WARNING

SARIN WILL OVERWRITE FILES OF THE SAME NAME WHICH ALREADY EXIST. IF THE pdl81.dir DIRECTORY EXISTS, BE SURE THAT IT DOES NOT CONTAIN FILES YOU WANT TO SAVE BEFORE YOU EXECUTE SARIN.

B. Alternate VMS Installation Method

This section describes an alternate procedure for installing PDL/81 under the VMS operating system. The primary installation procedure is described in Chapter 4 and requires installation on a release 4.7 or later of VMS. The alternate procedure, described in this Appendix B, may also operate under earlier versions of VMS.

The installation of PDL/81 should be performed by an experienced VMS system programmer.

Before installing PDL/81, this Section should be read and understood in its entirety.

B.1 Software Prerequisites

Release 4.7 of VMS, or a later compatible version, is required for proper installation and operation of this version of PDL/81.

B.2 Hardware Prerequisites

Any VAX computer capable of supporting Release 4.7 of VMS is required for proper installation and operation of this version of PDL/81.

Approximately 3,000 disk blocks are required during the installation process. After installation, the only permanently required space should be approximately 350 disk blocks for the executable image of PDL/81 and approximately 500 disk blocks for the distributed document style library.

B.3 A Note to the Reader

This manual is intended for an experienced VMS system programmer who is familiar with installing programs under VMS.

B.4 The Distribution Tape

The distribution tape contains all of the files necessary to install PDL/81 under VMS. This section describes the procedures for restoring the tape and the contents of the restored directories.

B.4.1 Restoring the Tape

The distribution tape was created by the VMS BACKUP utility. There are two save sets and the one used for installation as described in this Section is the *second* of the two.

The save set is named PDL81.BKP and may be restored by:

1. Create a directory, say PDL81, to hold the restored files:

```
$ CREATE/DIR  [.PDL81]
```

2. Allocate a tape drive and mount the tape:

```
$ ALLOCATE MT TAPE
$ MOUNT/FOREIGN TAPE
```

3. Restore the tape to the PDL81 directory:

```
$ BACKUP/LOG TAPE:PDL81.BKP/SEL=[PDL81DLV...] -
     [.PDL81...]
```

B.4.2 Contents of the PDL81 Directory

After restoring the tape, the PDL81 directory will contain:

`README`	Notices and restricted rights legend
`PD8_V20.EXE`	The executable image of PDL81.
`PDL81.CLD`	The Command Language Definition file for PDL/81.
`PDL81.HLP`	The help file for PDL/81.
`LIB.DIR`	The directory containing the PDL/81 style library (The “style” library).
`SAMPLES.DIR`	The directory containing sample PDL/81 input for testing the processor.
`SOURCE.DIR`	The directory containing the PDL/81 source and object files.
`TPDL.CLD`	A CLD file that may be used for testing PDL/81 prior to installation.
`TPDL.COM`	A command procedure which will temporarily define TPDL81 as a test version of PDL/81.

B.5 Making Site-Dependent Modifications

After the distribution tape has been restored, various site-dependent modifications must be made to PDL/81.

B.5.1 Modifying the Command Language Definition File

The PDL81.CLD file contains the input to the Command Language Definition utility. Changes can be made in this file to reflect the location of the installed PDL81 executable image, the location of the installed style library, the default output device type, and the default style.

B.5.1.1 Specifying Location of Installed PDL/81 Image

As distributed, the PDL81.CLD file assumes that the PD8_V20.EXE image will be placed in SYS$SYSTEM. Thus, it contains the line

```
    image pd8_v20
```

This line should be changed if another location is desired for the image. For example, if the image is to reside in SYS$TEST:[SPECIAL], the line

```
    image SYS$TEST:[SPECIAL]pd8_v20
```

might be used.

B.5.1.2 Specifying Location of Installed Style Library

As distributed, the PDL81.CLD file assumes that the style library will be located in SYS$SYSROOT:[SYSLIB.PD8_V20_LIB]. Thus, the definition for the /LIBRARY qualifier contains the line

```
    value=(default=SYS$SYSROOT:[SYSLIB.PD8_V20_LIB])
```

This line should be changed if another location is desired for the library. For example, if the library is to reside in SYS$TEST:[PDLLIB], the line might be changed to

```
    value=(default=SYS$TEST:[PDLLIB])
```

might be used.

B.5.2 Modifying the Style Library

Several files in the library directory [.PDL81.LIB] contain site dependent information. These files should be examined and the indicated modifications performed.

B.5.2.1 Defining Various Notices

A number of library files contain text which *must* be modified. The files are:

`[.LIB]CLPHN.LIB` definitions to handle the fourth argument of the *.Reverse* command of the *manual* style

`[.LIB]CPYRT.LIB` definitions to produce a copyright notice for the *manual* style

`[.LIB]RSTRCT.LIB`	definition of the Restricted Rights notice for the *manual* style
`[.LIB]UID.LIB`	definition of the user id string for the *design* and *ada* styles
`[.LIB]UTTL.LIB`	definition of the name and address block for the *design* and *ada* styles

NOTE

> These files are provided as examples only. Several of them contain text which may have legal implications. No legal advice is intended by their inclusion. It is strongly suggested that competent legal advice be obtained before adopting your own contents for such files.

B.5.2.2 Defining the Default Style

As distributed, PDL/81 will process with the *design* style unless the /STYLE qualifier is used. This default can be changed by modifying the file named DEFAULT. It's distributed contents will be something like:

```
#{lib;DESIGN}
```

For example, if you wish to have the *manual* style as your default, change the definition line to

```
#{lib;MANUAL}
```

B.5.2.3 Defining the Default Output Device

As distributed, PDL/81 will process with the *pr10* output device definition unless the /DEVICE qualifier is used. This default can be changed by modifying the file named DEFAULT.D. It's distributed contents will be something like:

```
#{lib;PR10.D}
```

For example, if you wish to have the *hp12* output device as your default, change the definition line to

```
#{lib;HP12.D}
```

B.5.2.4 Style and Device Tailoring

The distributed style and device files will usually produce the desired results. However, as you gain experience with PDL/81, you may want to tailor style and device definitions to meet specific device and site requirements. Style tailoring is discussed in Chapter 7 and device tailoring is discussed in Chapter 6.

NOTE

> Some VMS printer devices may not support underscoring as defined in the distributed device definitions. See Section 6.2 for a discussion.

B.5.2.5 Changing the Standard Page Depth

All distributed device files define a page depth of 66 lines. This may be changed, for each device, by changing the definition of the *pdepth* number register in the appropriate device file. For example, the page depth supported by the PR10 device may be changed from 66 lines to 60 lines by changing

```
#{nr;pdepth;66}
```

to

```
#{nr;pdepth;60}
```

in the PR10.D file in the [.LIB] directory.

B.6 Installing and Testing PDL/81

After the site-dependent modifications have been made, PDL/81 may be installed and tested as described in this section.

NOTE

> The procedures described in this chapter will modify the contents of various system directories and files. They must be performed by a privileged user. This chapter should be read and understood in its entirety before actually performing any of the procedures. Changes in these procedures may be necessary due to local requirements or policies.

B.6.1 Moving the Executable Image

The PD8_V20.EXE file should be moved to its installed location. This must be the same location defined in the PDL81.CLD file. For example, the image may be moved to the default location by

```
$ COPY PD8_V20.EXE  SYS$SYSTEM
```

After moving the image, the file protection must be set by

```
$ SET PROT=W:E SYS$SYSTEM:PDL81.EXE
```

B.6.2 Moving the Style Library

The style library should be moved to its installed location. This must be the same location defined in the PDL81.CLD file. For example, the library may be moved to the default location by

```
    $ CREATE/DIR SYS$SYSROOT:[SYSLIB.PD8_V20_LIB]
    $ COPY [.LIB]*.* SYS$SYSROOT:[SYSLIB.PDL81]
```

After moving the library, the protection on the directory and its contents must be set by:

```
    $ SET PROT=W:RE SYS$SYSROOT:[SYSLIB]PDL81.DIR
    $ SET PROT=W:RE SYS$SYSROOT:[SYSLIB.PDL81]*.*
```

B.6.3 Installing the PDL/81 Command

PDL/81 may be installed as a system-wide command or may be installed on a process-local basis.

B.6.3.1 System-Wide Installation

PDL/81 may be installed as a system-wide DCL command by editing the DCL command tables to include the PDL/81 Command Language Definition. Before performing the edit, the standard command tables should be saved by

```
    $ COPY SYS$LIBRARY:DCLTABLES.EXE -
           SYS$LIBRARY:DCLTABLES.OLD
```

Then, the tables may be edited by

```
    $ SET COMMAND /TABLES=SYS$LIBRARY:DCLTABLES -
           /OUTPUT=SYS$LIBRARY:DCLTABLES  PDL81
```

Following this, VMS must be rebooted in order to establish these tables as the active DCL tables.

B.6.3.2 Process-Local Installation

If installation policies preclude modifying the standard DCL command tables on a system-wide basis, process-local definition of the PDL81 command may be made by including the command

```
    SET COMMAND SYS$LIBRARY:PDL81
```

in either the standard initial LOGIN.COM file or in the LOGIN.COM file of each user who will be executing PDL/81. Note that this assumes that the PDL81.CLD file has been moved to the SYS$LIBRARY directory.

B.6.4 Installing the PDL/81 HELP File

The PDL/81 help module (file PDL81.HLP) may be installed in the standard system help library by

```
    $ LIBRARY/INSERT/HELP SYS$HELP:HELPLIB PDL81
```

B.6.5 Optional Installation of PDL/81 as a Known Image

If PDL/81 is to be run frequently at your installation, you may see a performance improvement by installing PDL/81 as a *known image*. This may be accomplished by inserting the following line at the appropriate point in your STARTUP.COM or local SYSTARTUP.COM files as dictated by installation policies.

```
PDL81 /OPEN /HEADER /SHARE
```

B.6.6 Compiling and Linking PDL/81

You do not normally need to compile the PDL/81 processor. However, if you have the PDL/81 source code and access to the VAX-11 C compiler, the entire PDL/81 processor may be compiled by setting your default to the [.SOURCE] directory and executing

```
@CCPDL
```

If you desire to recompile only some of the source modules, the command

```
CC  file[,...]
```

may be used.

The PDL/81 processor may be relinked by executing

```
@LINKPDL
```

which will link with the C object library or by

```
@LINKPDLS
```

which will link with the C library shareable image.

Index