

---

**Corsair Technology, Inc.**



**SPIN-X/ LPD**  
**Reference Guide**  
**for the SPIN-X/LPD**  
**Version 2R3r**

Document Revision Date January 23, 2001

Corsair Technology Inc.  
3000 Northwoods Pky, Suite 175  
Norcross, Georgia 30071

Document revision date - January 23, 2001

Printed in the United States of America

SPIN-X® is a trademark of  
Corsair Technology Incorporated.

Unisys® is a trademark of Unisys Corporation.

IBM® is a trademark of International Business  
Machines Corporation.

© 2001 Corsair Technology, Inc.

The contents of this document are proprietary to Corsair  
Technology, Inc. and are not to be disclosed to others or used for  
purposes other than intended without  
the written approval of Corsair Technology Inc.

# Table of Contents

<b>1. Introduction</b>	<b>1-1</b>
1.1. SPIN-X/LPD	1-1
1.2. Station Local Queues	1-1
<b>2. Installation</b>	<b>2-1</b>
2.1. SPIN-X/LPD System Requirements	2-1
2.1.1. System Product and Level Requirements	2-1
2.1.1.1. CMS & Network Requirements	2-1
2.1.1.2. TELCON Requirements (If DCP is used)	2-2
2.1.1.3. Other System Requirements	2-2
2.2. Installation from the Release Tape	2-2
2.2.1. Loading the Release Tape	2-3
2.2.2. Setting the LPD Parameters	2-4
2.3. The LPD2200 Parameter File	2-7
2.3.1. The *PARAM. File Structure	2-7
2.3.2. Keyword Notes...	2-10
2.3.2.1. Keyword: LOGFILE	2-11
2.3.2.2. Keyword: LOCALHOST	2-11
2.3.2.3. Keyword: OPTIONS	2-12
2.3.2.4. Keyword: OPERATION	2-13
2.3.2.5. Keyword: CONSOLEKEY	2-14
2.3.2.6. Keyword: BANNER	2-16
2.3.2.7. Keyword: HOLDQUEUE	2-17
2.3.2.8. Keyword: PROCESS	2-18
2.3.2.9. Keyword: HOST	2-19
2.3.2.10. Keyword: LPRALLOW	2-20
2.3.2.11. Keyword: LPRDENY	2-21
2.3.2.12. Keyword: PRINTER	2-22
2.3.2.13. Keyword: MACRO	2-23
2.3.2.14. Keyword: FORMAT	2-26
2.3.2.15. Keyword: QUEUE	2-30
2.3.2.16. Keyword: JOBPOOL	2-33
2.3.2.17. Keyword: DSTMAP	2-36
2.4. Sample LPD Parameter File	2-38
2.5. OS2200 Configuration and TCP/IP Setup	2-44
2.5.1. CMS Parameters	2-44
2.5.2. TELCON Parameters	2-45
2.5.3. Configuring the Exec	2-46
2.5.3.1. Defining Type 2 Symbiont Class Local Stations	2-46
2.5.4. LPD Runstream	2-46

<b>3. LPD Processor Operation</b>	<b>3-1</b>
3.1. Overview	3-1
3.2. Starting the LPD processor	3-2
3.2.1. LPD processor keyins	3-2
<b>4. Printer Installation and Setup</b>	<b>4-1</b>
4.1. Connecting to a Printer	4-1
4.1.1. Locally Attached and LAN Attached Printers	4-1
4.1.2. Network Attached Printers	4-1
4.1.3. Channel Attached Printers	4-1
4.2. Contents of the LPD [ <i>qualifier</i> ]*SETUP file	4-2
4.3. Report Setup and MakeReport	4-3
Report Setup File for Compilation by Makerpt or Makerpt2200	4-4
4.3.1. MACRO Commands to Control Formatting	4-5
4.3.1.1. Variable commands for Banner and Trailer Pages	4-5
4.3.1.2. Example Banner Setup File	4-6
4.3.2. Preformatted Setup Files	4-7
4.3.3. Example Makerpt Execution	4-7
4.4. Using LPR to Access Xpress PC Functionality	4-8
4.4.1. Passing the Format-Name to Xpress with SPIN-X LPR/LPD	4-8
<b>5. Installation Verification &amp; Troubleshooting</b>	<b>5-1</b>
5.1. Installation Verification for LPD Files	5-1

---

# APPENDICES

<b>A. SPIN-X/LPD Error Messages</b>	<b>A-1</b>
A.1. LPD Processor Error Messages	A-1
<b>B. Installation Checklist</b>	<b>B-1</b>
<b>C. Unisys Problem List Entries</b>	<b>C-1</b>
<b>D. Make Report</b>	<b>D-1</b>
D.1. Make Report Source Files	D-1
D.1.1. Specifying Source Files	D-2
D.1.1.1. Make Report Subdirectories	D-2
D.1.2. Sample Execution of Makerpt	D-3
D.1.3. Start up Switches	D-4
D.1.4. Make Report Tokens	D-5
D.1.5. Output Files	D-6
D.1.6. Macro Tokens	D-7
D.1.7. Make Report Primary tokens	D-9
D.1.8. Make Report Procedure Tokens	D-10
D.1.8.1. The !INCLUDE Procedure	D-11
D.1.8.2. The !PERFORM Procedure	D-12
D.1.8.3. The !PAUSE procedure	D-13
D.1.8.4. The !TRANSLATE procedure	D-14
D.1.8.5. The !FCBBUILD procedure	D-23
D.1.9. Sample Script File for Compilation by Makerpt or Makert2200	D-24
<b>E. LPD Processor Utilities</b>	<b>E-1</b>
E.1. CLEANQ	E-1
E.2. PLIST	E-1
E.2.1. Output of PLIST Processor	E-1
Octal Report Options	E-1
Text Report Option	E-1
Other Options	E-1
E.2.2. A Short Description Of Sdf Files.	E-1
Sdf Control Records	E-1
Sdf Data Records	E-1
E.3. MAKERPT2200 Overview	E-1
E.3.1. "F4N" files	E-1
E.3.2. Do I need to edit my DOS based script files for MAKERPT2200?	E-1
E.3.3. Operating tips	E-1
E.3.4. Processor Invocation:	E-1
E.3.5. Options	E-1

---

## 1.1 SPIN-X/LPD

---

SPIN-X/LPD allows a user on a Unisys 1100/2200 to issue an @SYM command to send a print file directly to a UNIX system or any system supporting the LPR-LPD protocol. As well, files on the remote Unix system can be printed on the host printers or routed through the host to another system. The MVS\_DOWNLOAD feature can be used for sending files to an IBM RS6000 running PSF6000. This method is faster than transferring files with the LPR protocol.

With SPIN-X/LPD, all the setup is controlled from the mainframe using a single parameter file. Tailoring of the LPR options can be made on a queue basis. SPIN-X/LPD provides the user with great flexibility by providing optional page setup for the banner page, the report and end-of-job sequences. Besides generating banner and trailer pages containing variable information, page setup can be used to automatically put mainframe printfiles in a format suitable for reading by Web browsers. SPIN-X/LPD also supports PC tools for form generation.

Print queues used by the LPD Server are Station Local Queue names configured in the Operating System of the Unisys host. There are two types of queues used by SPIN-X/LPD. INPUT queues are Station Local print queues from which the LPD Server receives print files to be processed. HOLD queues are used by SPIN-X/LPD to store print jobs that are not in the proper format to process or that could not be delivered due to an error or configuration problem.

In the diagram below, the LPR (SEND mode) portion of the LPD2200 processor is illustrated along side some of the \*PARAM file statements that must be configured.

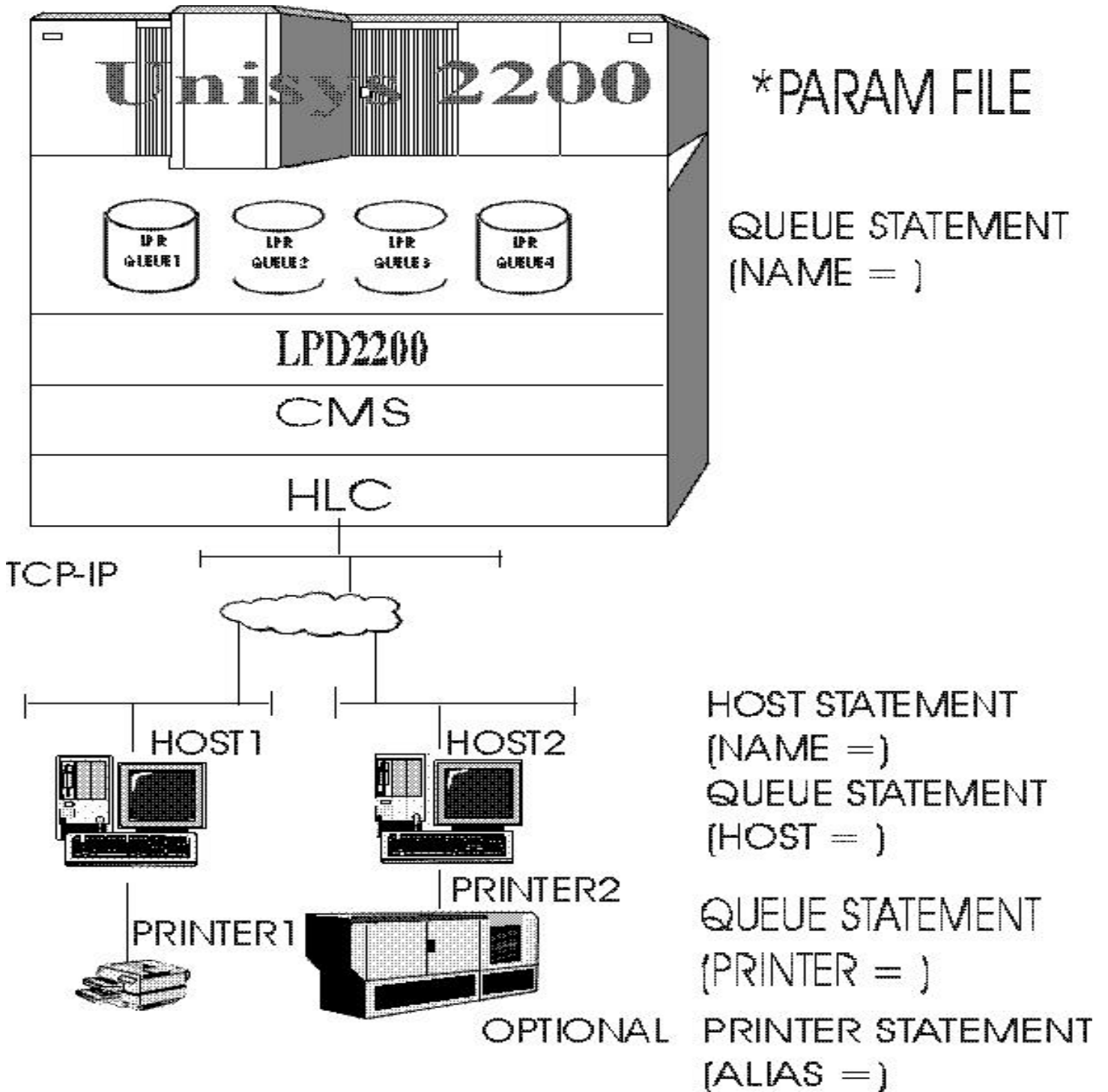


Figure 1

In the diagram below, the LPD (RECEIVE mode) portion of the LPD2200 processor is illustrated along side some of the \*PARAM file statements that must be configured.

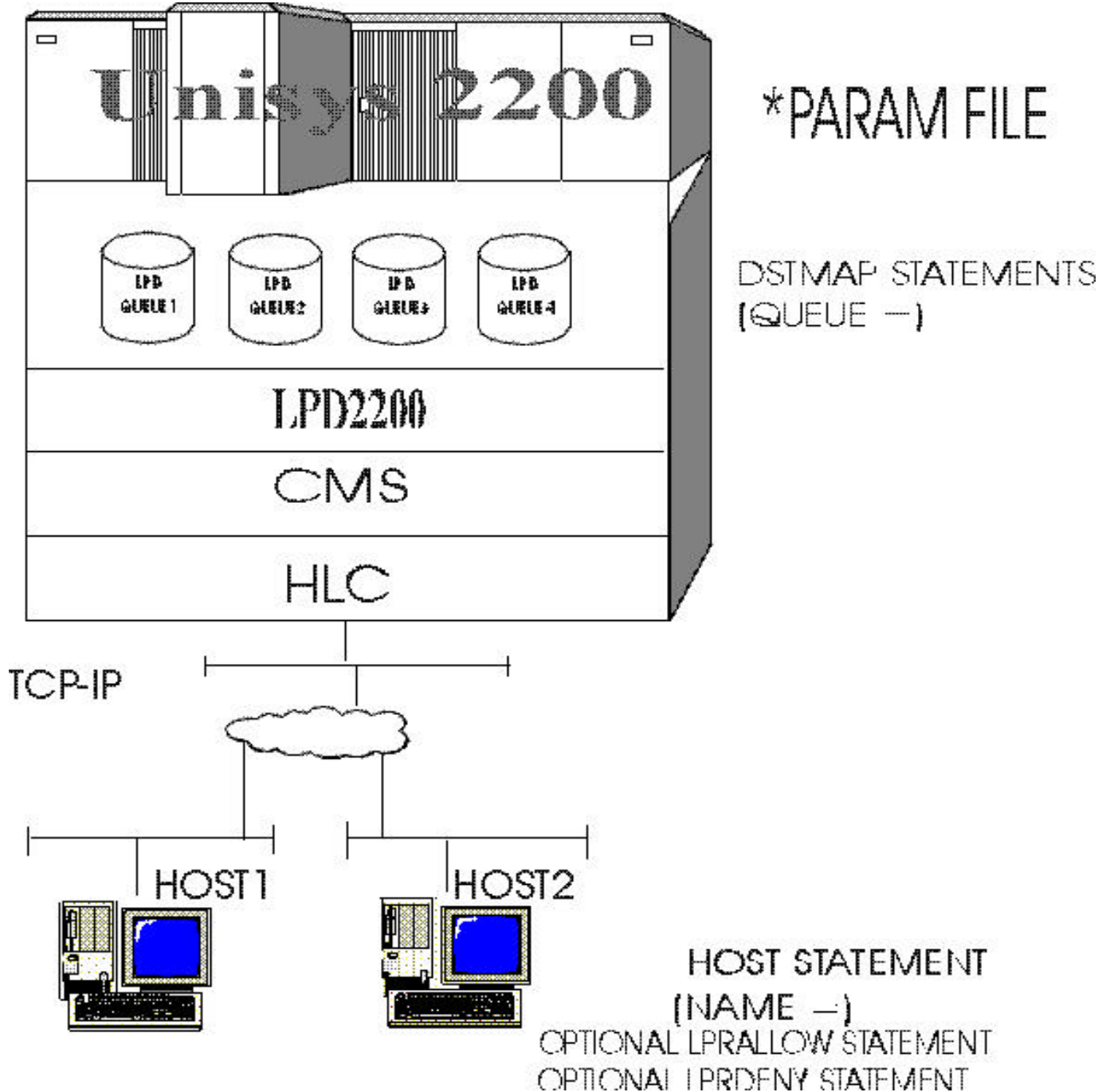


Figure 2

Inbound jobs can be dropped on queues that are for outbound jobs if the DSTMAP "Queue =" field is the same as the "Name =" field of a QUEUE statement. The "Format =" and "Printer =" fields of the DSTMAP statement are overridden by the corresponding fields of the QUEUE statement.

---

## 2.1 SPIN-X/LPD System Requirements

---

### 2.1.1 System Product and Level Requirements

---

The first system levels to support the components required by the LPD feature were as follows:

Operating System	SB4R1
CMS	7R1
TELCON	8R2
TCPIP-STACK	1R2B
DCP LAN	2R1B (If using a DCP not required for an HLC)

*Note: These are the first levels that are known to support the LPD components, the LPD components have not been tested under these levels. CD 5R1 was the first release to support TCP/IP and TSAM*

The SPIN-X/LPR-LPD product was qualified with the following release levels:

Operating System	SB5R2
CMS	7R4
TELCON	9R1B
TCPIP-STACK	1R2B
DCP LAN	2R2B (If using a DCP not required for an HLC)

*Note: Some sites will need to get "fixes" for the Exec from Unisys. Appendix C contains Problem List Entries for the Exec and CMS. If your site experiences one of these problems, the "fix" will be listed in Appendix C.*

#### 2.1.1.1 CMS & Network Requirements

---

CMS PROCESS, IP, and INTERNET-ADR statements are required to define host and DCP addresses. The reader is referred to Section 2.5.2.

*Note: The following fixes to CMS 8R2 (see Appendix C) have been installed at GSU in order to resolve various intermittent problems:*

*Problem Number*  
**16558222**  
**16641201**  
**16665657**  
**16669148**  
**\*16675814**  
**16689017**  
**16689092**  
**16878588**  
**16881899**  
**16902594**  
**16905887**  
**16934194**  
**16941140**  
**16956279**

*\* The fix to problem #16675814 is definitely required if LPD2200 hangs and a 'resume output' message is the last message in the LPDLOG file.*

A Distributed Communications Processor (DCP) or Host LAN Controller (HLC) connected to a TCP/IP network is required.

### **2.1.1.2 TELCON Requirements (If DCP is used)**

---

SUBNET statements for the Host and LAN with IPROUTER=YES are required.

### **2.1.1.3 Other System Requirements**

---

STATION LOCAL queues must be configured as described later in the installation instructions.

UCS run-time libraries are required.

---

## **2.2 Installation from the Release Tape**

---

The installation process will create the files and run-streams required by the LPD processor. Modifications of the runstream, as described later in this chapter, may be made to suit the needs of each site.

The installation process consists of SSG menu screens (shown here in Courier font) which allow the user to enter various configuration parameters required in the creation of the LPD files and run-streams. Processor version numbers and default values for the current release may be different than those shown here.

Our install procedure is intended for initial setup and not for production. Before LPD can be run, it will be necessary to edit the LPD \*PARAM file that results from the install process in order to change queue names and printer names etc. to conform with the ones at your site.

**IMPORTANT NOTE:** LPD users updating to a new version are warned not to overwrite their production version of \*PARAM during the install of the new version. Save it under a different qualifier.

## 2.2.1 Loading the Release Tape

From a demand terminal, enter:

```
@ASG,JT TAPE.,eqp,reel,,NORING
@MOVE TAPE.,1
@ASG,T TEMP.,F///1600
@COPY,G TAPE.,TEMP.
@TEMP.INSTALL
```

Where 'eqp' is the equipment type of your release tape (U47, U9S, U9V, or HICL), and 'reel' is the tape reel number of your release tape.

The following main menu screen is displayed:

```
          SPIN-X/LPD Installation
          Version 2R3
*****
* COPYRIGHT 2001 Corsair Technology, Inc.                                *
*                                                                 *
* SPIN-X, SPIN-X /CENTRAL, SPIN-X/RMS, SPIN-X/LPD, and SPIN-X/XPRESS are *
* trademarks of Corsair Technology, Inc.                               *
*                                                                 *
*                               Corsair Technology, Inc.                *
*                               Norcross, Georgia 30017 U.S.A.         *
*                               All Rights Reserved.                   *
*****
```

- A) Load tape - must be the first option selected
- B) Set SPIN-X/LPD parameters, and install files
- C) Display SPIN-X/LPD parameters
- Q) Exit

Enter the appropriate item letter to proceed with installation

(A,B,C,Q) >A

Option A must be selected initially, to load the release tape files into mass-storage files. You must then select a qualifier for the LPD files that will be created. If you do not provide a qualifier, the default qualifier shown will be used.

**NOTE: This documentation assumes that a qualifier of LPD is used.**

Enter the qualifier to be used for LPD files

The file qualifier specified will be used by the installation process when cataloging the LPD Files.

Xmit blank for default qualifier: (LPD) >

The default version of the LPD processor was created under HMP5.1 of the UNISYS operating system, and is configured to use the standard CMS Processor.

An alternate version of the LPD processor named LPD220/TEST is also available on this release tape. This version is configured to use the CMS TEST version Processor.

Answer "Y" to the following question only if you are using LPD2200 with the CMS TEST version. LPD2200/TEST will be copied to the LPDTST\*.lpd file as LPD2200 with no version so that later it will be recognized by the LPD runstream (section 2.5.4).

Is LPD to be used with the CMS TEST version Processor?  
(Y/<N>)N

The screen below displays the files that will be created during the installation. LPD\*PARAM will only appear if this is the initial installation.

If this is an initial installation of LPD2200, a LPD\*PARAM(+1) sample file is created for you to use as a starting point to modify for your site configuration.

If this is NOT an initial installation, this sample PARAM file will not be created. In any event, all files created will be with a (+1) fcycle to avoid destroying your current files.

Answer Y to the following question only if this is an initial installation and the LPD\*PARAM(+1) is to be created.

Is this an initial installation of LPD2200? (Y/<N>)

The following files will be created:

- LPD\*INSTALL(+)
- LPD\*LPD(+1)
- LPD\*PARAM(+1)
- LPD\*JOBNUM(+1)
- LPD\*SETUP(+1)
- LPD\*UTILITY(+1)

Proceed with cataloging of (+1) cycles? (<Y>,N)

## 2.2.2 Setting the LPD Parameters

---

The main menu is displayed again, but without option A since the tape has already been loaded. Enter the B response.

```

                SPIN-X/LPD Installation
                Version 2R1
*****
* COPYRIGHT 2001 Corsair Technology, Inc.                                *
*                                                                 *
* SPIN-X, SPIN-X /CENTRAL, SPIN-X/RMS, SPIN-X/LPD, and SPIN-X/XPRESS are *
* trademarks of Corsair Technology, Inc.                                *
*                                                                 *
*                Corsair Technology, Inc.                                *
*                Norcross, Georgia 30017 U.S.A.                        *
*                All Rights Reserved.                                    *
*****

```

- B) Set SPIN-X/LPD parameters and install files
- C) Display SPIN-X/LPD parameters
- Q) Exit

Enter the appropriate item letter to proceed with installation

(B,C,Q)>

**Project-id**

Enter the Project-id for the LPD Server @RUN card

This Project-id will be used in the LPD/RUN run-stream in the LPD\*LPD file.

Xmit blank for default Project-id (MYPROJ)>

**Account Number**

Enter the account number for the LPD @RUN cards

This account number will be used in the LPD/RUN run-stream in the LPD\*LPD file

Xmit blank for default (MYACCOUNT) >

**Userid**

```
*****
* NOTE:  The LPD Server interfaces with the EXEC to perform *
*        specialized functions.  The Userid for the Server *
*        run must have SSMOQUE and SCONSOLE privileges    *
*****
```

Enter the Userid for the Server @RUN card

This Userid will be used in the LPD/RUN run-stream in the LPD\*LPD file.

Xmit blank for default (MYUSERID) >

The following screens are then displayed.

The SPIN-X/LPD installation is complete

The LPD\*LPD program file contains:

```
LPD2200    -LPD zoom element
LPD2200/PADS - The LPD zoom element with PADS configured
LPD/RUN    -The LPD Processor batch run-stream
```

If CMS TEST mode was selected the following line appears:

```
The LPD2200 components have been linked with
CMS1100*test$lib.cbep$cms to use the CMS test processor.
```

The batch run stream should be executed from SYS\$LIB\$\*RUN\$ to ensure that it is started with the correct account number and Userid.

The LPD2200/PADS version is included for possible debug analysis, if your site has PADS installed.

Xmit to continue

The LPD\*PARAM file is an SDF file containing a sample LPD configuration. This file is a sample only, and must be edited to define the LPD configuration at your site.

The LPD\*JOBNUM file is a simple SDF file which contains a single image containing the LPD job number. It is used by LPD to assign unique numbers to each print file processed.

The LPD\*SETUP file is a program file containing sample LPD setup elements. The SETUP/DOC element in this file describes the sample SETUP elements included with this release tape.

Xmit to continue

The LPD\*UTILITY program file contains utility programs including

- CLEANQ - Resets the in-progress status of queued print files
- MAKERPT2200 - Creates omnibus setup elements.
- PLIST - Lists/dumps SDF print files.

There are /DOC elements in this file describing these utilities

Xmit to continue

The main menu is displayed again. Entering Q will exit the installation program. If C is entered, the information below is displayed, showing the parameters entered during the installation.

The following is a review of the SPIN-X/LPD configuration parameters

Description	Value
-----	-----
Qualifier for LPD files	LPD
Server Project-id	MYPROJ
Server Account Number	MYACCOUNT
Server Userid	MYUSERID
Initial Installation?	YES
Using CMS Test processor?	NO

Xmit to continue

Control will be returned to the Main Menu.

## 2.3 The LPD2200 Parameter File

The \*PARAM. file is the configuration file used by the OS2200 based Spin-X/LPR-LPD program referred to as LPD2200. This file controls every aspect of LPD2200 operation from the setup of the print queues with their banners and report formats to the determination of remote hosts that can send or receive prints.

### 2.3.1 The \*PARAM. File Structure

The \*PARAM. file is a text based configuration file used by the Spin-X LPD2200 batch program. The \*PARAM file is composed of a series of keyword statements. Each keyword statement starts with one of the keywords defined below and is followed by one or more parameter fields. Each keyword has a unique set of required and optional parameter fields.

The following seventeen \*PARAM. keywords are currently defined:

1. BANNER
2. CONSOLEKEY
3. DEFAULT\_FORMAT
4. DSTMAP
5. FORMAT
6. HOLDQUEUE
7. HOST
8. LOCALHOST
9. LOGFILE
10. LPRALLOW
11. LPRDENY
12. MACRO
13. OPERATION
14. OPTIONS
15. PRINTER
16. PROCESS
17. QUEUE
18. JOBPOOL

Record oriented comments may be freely interspersed within keyword statements. A comment is started whenever a semi-colon (;) character is detected. All data from the semi-colon to the end of the record (end of line) is then ignored.

#### Example:

```
; Production PARAM file for Spin-X/LPR-LPD program (LPD2200).
;
; Update: 05-13-99 (jhy)
; Added new QUEUES for poohbear.
;
; Update: 04-24-98 (whp)
; Added DSTMAP
```

Some keywords statements allow multiple parameter fields. Within a multi-field keyword statement, each parameter field must be separated from the next parameter field by a comma (,) character. Keyword statements may span multiple lines without an explicit continuation character. There is NO comma after the last (or only) parameter field: the keyword statement is implicitly terminated by the absence of a comma ',' character.

Parameter fields are composed of a reserved field name, an equal sign '=' and one or more field values. If more than one field value is presented, then all of the field values for this particular field must be enclosed within parentheses characters '(field1,field2,...,fieldn)' and each individual field value must be separated from the next with comma ',' characters.

**Example keyword statement with two fields:**

```
HOST      NAME=PRINTHOST.GSU.EDU,
          ALIAS=(PRINTHOST, prserver)
```

The above example defines a multi-field "HOST" keyword statement. Two field sequences, "NAME" and "ALIAS", have been defined. The first field, "NAME", has a single field value of "PRINTHOST.GSU.EDU". The second field, "ALIAS", has two field values defined, "PRINTHOST" and "prserver".

**NOTE:** Because multiple field values are defined for the "ALIAS" field name, the field values **MUST** be encapsulated within parentheses characters.

**NOTE2:** An arbitrary number of ASCII SPACE characters are allowed before and after the various elements of a parameter field. For example the following two parameter fields are virtually identical:

```
ALIAS=(NAME1,NAME2)
ALIAS = ( NAME1 , NAME2 )
```

While the keyword names and the field names are **NOT** upper case or lower case specific, the field values **MAY** be. For example the PRINTER statement's NAME field value typically must be defined exactly the way expected on the remote system. The field values "NAME=printer1", "NAME=Printer1" and "NAME=PRINTER1" would represent three different printer names on most typical Unix systems.

**NOTE:** Unix system administrators typically define their printer names as all lower case names.

Certain keyword statements are required within all \*PARAM. files. Other keyword sequences are optional depending on the OPERATION MODE required for LPD2200, (i.e. RECEIVE, SEND or FULL). Certain keyword statements are allowed at most only once (i.e. BANNER, CONSOLEKEY, etc), other keyword statements may be defined as many times as required (i.e. HOST, PRINTER etc). If a particular keyword is defined multiple times (i.e. FORMAT, QUEUE, etc) it's NAME value must be unique. For example it is illegal to have two FORMAT statements with a parameter field of NAME=FMT1.

A List of LPD parameter statements and their fields follows. Where default values are applicable, they are shown in bold type. It should be noted that any names defined by the user can be used in a parameter statement only if the definition of the name precedes that statement in the \*PARAM file. For example, all macros referenced by format statements must precede the format statements which reference them.

<b>LOGFILE</b>	NAME = <i>&lt;Internal-log-filename&gt;</i> ; (max of 64 characters)
<b>OPTIONS</b>	WRITETOCONSOLE = [YES   NO], OUTPUTBLOCKSIZE = <i>&lt;block-size-in-bytes&gt;</i> , ; Default is <b>4096</b> PRINTBANNER = [YES   NO] ALLOW_CONTROL_ERRORS = [YES   <b>NO</b> ], ALLOW_TAPES = [YES   NO], LOGDETAILS = [LOW   <b>HIGH</b>   DEBUG], CONSOLE_MESSAGE_GROUP = <i>&lt;group-name&gt;</i> , MAX_FILES_ACTIVE = [1 - 40] ; Default is <b>10</b>
<b>OPERATION</b>	MODE = [SEND   RECEIVE   <b>FULL</b> ], SEND_PORT_RANGE = <i>&lt;port address range&gt;</i> ; Default is <b>721 - 731</b>
<b>HOST</b>	NAME = <i>&lt;Fully-qualified-domain-name&gt;</i> , ; (max of 80 characters) ADDRESS = <i>&lt;IP-address-in-dotted-notation&gt;</i> , ALIAS = <i>&lt;alias-list&gt;</i> ; (max of 80 characters per alias)
<b>LOCALHOST</b>	NAME = <i>&lt;Unisys-local-host-name&gt;</i> ; (max of 80 characters)
<b>PROCESS</b>	NAME = <i>&lt;CMS-process-name&gt;</i> , ; (max of 6 characters) PASSWORD = <i>&lt;CMS-process-password&gt;</i> ; (max of 32 characters)

<b>HOLDQUEUE</b>	NAME = <Unisys-hold-queue-id> ; (max of 6 characters)
<b>CONSOLEKEY</b>	NAME = <Console-keyin> ; (max of 8 characters)
<b>LPRALLOW</b>	HOST = [<Host_name_list>   \$ALL\$   \$HOSTTABLE\$]
<b>LPRDENY</b>	HOST = [<Host_name_list>   \$ALL\$   \$HOSTTABLE\$   \$NONE\$]
<b>BANNER</b>	MASK = <Bannerid-mask> ; (max of 12 characters)
<b>PRINTER</b>	NAME = <Printer-name>, ; (max of 19 characters) FILETYPE = <ASCII   FL130   XEROXVL   PRISMVL]>, ALIAS = <alias-list> ; (max of 19 characters)
<b>MACRO</b>	NAME = <macro-name>, ; (max of 31 characters) FILENAME = <Fully-qualified-Unisys-element-name>, ; (max of 99 characters) TYPE = [OMNIBUS   SYMBOLIC], CARRIAGE_CONTROL = [ LF   CRLF   ANSI   NONE], KEYWORD = [YES   NO]
<b>DEFAULT_FORMAT</b> or <b>FORMAT</b>	NAME = <Default-format-name>, ; (max of 12 characters) LINES = <Lines-per-page>, ; Default is zero WIDTH = <Characters-per-line>, ; Default is zero TOP_MARGIN = <Top-margin>, ; Default is zero BOTTOM_MARGIN = <Bottom-margin>, ; Default is zero IGNORE_MCONTROL = [YES   NO], LPD_C_OPTION = [<C-option-string>   \$NONE\$], ; (max of 131 characters) BANNER_SETUP = [<Macro-list>   \$NONE\$], ; (max of 31 characters) REPORT_SETUP = [<Macro-list>   \$NONE\$], ; (max of 31 characters) 6LPI_SETUP = [<Macro-list>   \$NONE\$], ; (max of 31 characters) 8LPI_SETUP = [<Macro-list>   \$NONE\$], ; (max of 31 characters) TRAILER_SETUP = [<Macro-list>   \$NONE\$], ; (max of 31 characters) PSF_OPERATOR_MSG = [YES   NO], PSF_CONTROL_FLAG = [YES   NO], PSF_PAGEDDEF = <Pagedef Value>, ; (max of 8 characters) PSF_FORMDEF = <Formdef Value>, ; (max of 8 characters) PSF_CHARS = <Chars Value> ; (max of 4 characters)
<b>QUEUE</b>	NAME = <Unisys-queueid>, ; (max of 6 characters) PHANTOM = [YES NO], GROUP = <Group-name>, ; (max of 31 characters) INIT = [UP DOWN], PRINTER = <Printer-name>, ; (max of 19 characters) FORMAT= <Format-name>, ; (max of 12 characters) TYPE = [TEXT, RAW], SEND_PROTOCOL = [LPR MVS_DOWNLOAD], REMOTE_PORT = <Remote-port-number>, BANNER_MASK = [YES   NO], HOST = <Host-name>, ; (max of 80 characters) CONTROL_SETUP=[<Macro-list> \$NONE\$], SPECIAL_FORM_MSG = [IGNORE   REQUE_MSG   TOHOLDQ], OPERATOR_REMOTE_QUEUE = <Remote-printer-name>, ; (max 19 characters) MAX_ACTIVE = [1-10]
<b>JOBPOOL</b>	NAME = <Unisys-queueid>, ; (max of 6 characters) POOL = (lpd queue1, lpd queue2,...,lpd queue10), THRESHOLD = (jobpool queue threshold, queue1 threshold,...,queue10 threshold), SELECT_FIRST = [YES NO], CARRY_PRINTER = [YES NO], PASS_THRU = [YES NO]

**DSTMAP**    PRINTER = <Printer-name>, ; (max of 19 characters)  
              FORMAT = <Format-name>, ; (max of 12 characters)  
              PRESERVE\_NULLS = [YES|NO],  
              QUEUE = <Unisys-queueid>, ; (max of 6 characters)  
              ALTPRINTER = [<alternate-printer-name. | \$NONE\$], ; (max of 12 characters)  
              ALTFORMAT = [<alternate-printer-name. | \$NONE\$], ; (max of 8 characters)  
              DEFAULT = [YES | NO]

### 2.3.2 Keyword Notes...

---

Included below are notes on the LPD2200's \*PARAM. file keyword statements. Again note that the \*PARAM file cannot have "forward references" in it. For instance, a macro referenced in a format definition cannot appear in the \*PARAM file after that format definition. The KEYWORDS below are introduced in an order in which they could be defined within a valid \*PARAM. file containing no forward references. Therefore the LOGFILE keyword defined below should be the first keyword statement present within the \*PARAM. file, LOCAL-HOST should be next, and so on.

---

### 2.3.2.1 Keyword: LOGFILE

---

The LOGFILE statement is used to define to LPD2200 the @USE name of the log file to be used by the LPD2200 program. The LOGFILE statement must be the first keyword statement in the PARAM file, and the name **must** agree with the @USE name in the LPD2200 run-stream. The LPD2200 log file is a text based file that contains entries in chronological order regarding the actions and activities undertaken by LPD2200.

The following LOGFILE parameter field is required:

```
NAME=logFileUseName
```

... where *logFileUseName* is the @USE name of the log file to be used.

**Example:**

```
LOGFILE NAME=LPDLOG
```

---

### 2.3.2.2 Keyword: LOCALHOST

---

The LOCALHOST statement is used to define an identifier that will be used by LPD2200. The local host name is used as the "originating" host name in outgoing print jobs. This includes the host name component used for naming the "cfA" and "dfA" files, as well as the host name identifier written to the "H" record of the control (i.e. "cfA") file. The local host name is also used in responses generated by LPD2200 to various lpr, lpq, lprm and lpc service requests by remote systems.

The following LOCALHOST parameter field is defined:

```
NAME=localHostName
```

...where *localHostName* is the name to be used as the local host identifier. This name cannot be a fully qualified domain name: it must be a single word, such as an alias in the HOST statement.

If the local host has both an HLC and a DCP and the user wishes to let CMS decide which one to use, then the NAME = field should not match any alias in any HOST statement. This will cause the LPR open request to be resolved by CMS which could result in either the HLC or the DCP being used.

If the user wishes to use only the HLC for outgoing LPR requests, then there must be a HOST statement that gives the fully qualified domain name of the HLC and an alias for it that is the same as the NAME = field of the LOCALHOST statement. For this case the open request will be resolved in favor of the HLC before it gets to CMS. The DCP could be chosen in a like manner by using an alias for the DCP.

**Example:**

```
LOCALHOST NAME=jhyLpd2200
```

### 2.3.2.3 Keyword: OPTIONS

The OPTIONS statement is used to control a number of global parameters affecting LPD2200 behavior. Parameters of the OPTIONS statement are separated by commas.

The following (optional) OPTIONS parameter fields are defined:

WRITETOCONSOLE=[ YES | NO ]

This field defines whether or not critical error messages are written to the system console in addition to the LPD2200 log file. The default, if not specified, is YES.

OUTPUTBLOCKSIZE=[ 0-16380 ]

This field defines the maximum size of each output data block presented to CMS, to be sent out to the peer. Valid values range from 0 to 16380. A value of 0 defaults to a value of 4096. The default, if not specified, is **4096** (4k octets).

PRINTBANNER=[ YES | NO ]

This field defines whether or not LPD2200 should generate LPR control file records which inform the remote host that it should generate a banner page. The default, if not specified, is YES.

**NOTE:** Please do not confuse the ability to enable/disable remote banner generation with LPD2200's ability to locally generate banner pages via the FORMAT statement's BANNER\_SETUP field.

ALLOW\_CONTROL\_ERRORS=[ YES | NO ]

This field defines how LPD2200 should react if an invalid type-060 control image is encountered as when often occurs when processing tape files. If 060 control record errors are allowed (i.e. YES) then the invalid 060 control image is ignored. If control record errors are not allowed (i.e. NO) then an error message is inserted into the outbound print file and subsequent file processing is terminated. The default, if not specified, is NO.

ALLOW\_TAPES=[ YES | NO ]

This field specifies whether or not LPD2200 should ignore @SYM'd tape files. The default, if not specified, is NO.

LOGDETAILS=[ LOW | HIGH | DEBUG ]

This field defines the INITIAL detail level of messages written to the LPD2200 log file. The detail level may be changed at run time via the LPD2200 console keyin command /LD. The following LOGDETAILS values are defined:

LOW : Logs all the print files sent and received.

HIGH : Also logs the TCP Opens and Closes, and more details on the data and control files received on inbound print jobs (i.e. jobs sent from a remote host to LPD2200).

DEBUG: Besides the information logged in the above levels, DEBUG level also enable logging of step in the process of sending and receiving print files via LPD/LPR protocol.

**NOTE:** The processor call options D, H, or L, if specified will override the value specified in LOGDETAILS. Using the call option will enable a high level of log detail while LPD is coming up, even before the PARAM file is read. The console keyin /LD will override the value from this processor call.

**IMPORTANT NOTE:** DEBUG level is not recommended for production systems because of the overhead involved and the amount of information logged. Any serious errors encountered by LPD2200 will be logged regardless of the current LOGDETAILS level. The default, if not specified, is HIGH.

### 2.3.2.4 Keyword: OPERATION

---

The OPERATION statement defines which service(s) the LPD2200 will provide. LPD2200 may be configured only to send OS2200 print files to remote systems, or only to receive print files from remote systems or it may be configured to do both.

The following mandatory OPERATION parameter field is defined:

```
MODE=[ SEND | RECEIVE | FULL ]
```

... where the MODE field defines whether Spin-X/LPD will provide outgoing LPR client services (SEND), incoming LPD server services (RECEIVE) or both (FULL).

SEND: LPD can ONLY initiate transfer of files from the OS2200 host to other hosts using the client side of the LPR/LPD spooling protocol.

RECEIVE : LPD can ONLY receive inbound LPR requests from other hosts.

FULL : (Default) Both SEND and RECEIVE are allowed.

If the SEND mode is enabled, there is a further option:

```
SEND_PORT_RANGE = <starting-port-number - ending-port-number>
```

This parameter allows the site to override the default local port number range of 721 through 731. The smallest starting port number allowed is 256 and the largest ending port number allowed is 1024.

This port number range is used by LPD2200 when making a TCP open request.

**Example:**

```
OPERATION MODE=SEND,  
SEND_PORT_RANGE = 631 - 731
```

### 2.3.2.5 Keyword: CONSOLEKEY

---

The CONSOLEKEY statement is used to define a unique operator keyin with which OS2200 console operators may control LPD2200.

The following (mandatory) consolekey field is defined:

NAME=*consoleKey*

...where *consoleKey* is the unique console keyin prefix required to send LPD2200 commands to this particular instance of LPD2200.

For example if NAME=LPD2200\* then the console command "LPD2200\* /TERM" would be used to initiate shutdown of the LPD2200.

The following LPD2200 commands are currently defined:

- ID            Displays the banner (i.e. version/revision).
- STAT        Displays connection status and information about the queues.
- STAT UP     Displays status of only the unlocked queues
- STAT *queue* Displays queue status of the indicated queue
- TERM        Initiates a graceful termination of LPD2200.
- KILL        Initiates an immediate (ungraceful) termination of LPD 2200.
- INIT        (or /I) Initiates (unlocks) all queues which have been locked.
- INIT *queue* Initiates (unlocks) a queue *queue* which has been locked.
- LD          Displays the current LOGDETAILS value.
- LD LO       Change the current LOGDETAILS value to LOW.
- LD HI       Change the current LOGDETAILS value to HIGH.
- LD D        Change the current LOGDETAILS value to DEBUG.
- LOCK        (or /L) Locks all queues. Files which are still in the process of printing are allowed to finish
- LOCK *queue* Locks the individual queue *queue* .

**Example:**

CONSOLEKEY NAME=LPD\*

For more information about the Console Keyins see Chapter 3 "LPD Processor Operations".

```
CONSOLE_MESSAGE_GROUP = <group-name>
```

where the group-name must be one of the following:

```
SYMSMSG (default)
IOMSG
RSICOM
HSWCON
USER4
USER5
USER6
USER7
```

These group names correspond with the SIMAN User-Id console mode message groups. The name chosen must have been selected as a response console in SIMAN.

This message group affects only the special-forms read-and-reply console messages that are generated when an SDF type-60 or type-61 operator message record is encountered in a print file. This statement is useful for sites where the onsite console operator is not to be responsible for answering these special-forms messages.

See the QUEUE statement's SPECIAL\_FORM\_MSG parameter for the options regarding the handling of SDF type-60 and type-61 operator messages.

```
MAX_FILES_ACTIVE = [ 1-40 ]
```

Specifies the maximum number of print files that can be transmitted at the same time. The maximum is 40, and the default is **10**.

LPD2200 starts a multiple separate activities for each print file being processed. Keeping this number small will limit the amount of memory and processor time used by LPD2200. A larger value may maximize the throughput depending on the number of remote hosts used and their capacity.

Example:

```
OPTIONS WRITETOCONSOLE=YES ,
        OUTPUTBLOCKSIZE=6144 ,
        PRINTBANNER=NO ,
        ALLOW_CONTROL_ERRORS=YES ,
        ALLOW_TAPES=NO ,
        LOGDETAILS=HIGH
```

### 2.3.2.6 Keyword: BANNER

---

The optional BANNER statement is used to define to LPD2200 the bannerid mask to be used for any QUEUE statements with bannerid masking enabled (i.e. BANNER\_MASK=YES).

The following required BANNER field is defined:

```
BANNER MASK=banneridMask
```

The MASK field is used to define how the 12 characters available for the Bannerid field of the @SYM command should be parsed in order to determine appropriate destination information (i.e. HOST, PRINTER and/or FORMAT) to be associated with the current outbound print job. From left to right each character is decoded according to the current MASK value. The 'H' character is used within the MASK to define bannerid character positions reserved for deriving the destination HOST. The 'P' character is used within the MASK to define bannerid character positions reserved for deriving the destination PRINTER. The 'F' character is used within the MASK to define bannerid character positions reserved for deriving the FORMAT. All similar masking characters must be grouped together, i.e. a MASK value of "HHFFPPHHFFPP" is not legal. The supplied mask MAY be less than 12 characters long.

As jobs are retrieved from the various QUEUES that have bannerid masking enabled, LPD2200 will attempt to determine the desired destination information from the bannerid field of the @SYM command. The various components of the bannerid field extracted against the current mask are checked against the various NAMES and/or ALIAS values defined for their respective components.

Examples:

```
BANNER MASK=HHHHPPPPFFFF
```

The first four characters constitute HOST, the middle four characters constitute the PRINTER, and the last four character constitute the FORMAT.

```
BANNER MASK=PPPPPPFFFFFF
```

The first six characters constitute the PRINTER, and the last six characters constitute the FORMAT.

If a host name (or alias) or a format name used in the bannerid field of the SYM statement is not defined in the PARAM file, then the host or format name in the QUEUE statement is used.

If a printer name (or alias) used in the bannerid field of the SYM statement is not defined in the PARAM file, then it is passed on to the remote host on the assumption that the remote host will have a printer of that name.

---

### 2.3.2.7 Keyword: HOLDQUEUE

---

The HOLDQUEUE statement is used to define the name of an OS2200 queue where LPD2200 may requeue files with an incorrect format, or when an error status is received from the remote host.

The following required HOLDQUEUE parameter field is defined:

`NAME=OS2200QueueName`

... where *OS2200QueueName* is the name of an OS2200 queue (typically a STATION LOCAL name).

Example:

```
HOLDQUEUE NAME=LPDERR
```

### 2.3.2.8 Keyword: PROCESS

---

PROCESS statements are used to define the names of a pool of CMS TSAM processes to be used by LPD2200 when communicating with remote hosts. At least one process statement must be defined to LPD2200 regardless of the what mode LPD2200 is currently configured for (SEND, RECEIVE or FULL).

**NOTE:** Since LPD2200 is a multi-activity application, several process names should be configured. LPD2200 will poll each configured processName until it finds one available.

**NOTE2:** The processNames defined to LPD2200 *\*MAY\** also be defined concurrently to other TSAM applications (such as a second copy of LPD2200) assuming that the other applications also acquire and release the CMS processes only when needed.

The following required PROCESS parameter field is defined:

NAME=*processName*

...where *processName* is the name of a CMS process to be used by LPD2200.

The following optional PROCESS parameter field is defined:

PASSWORD=*password*

The optional password configured for this particular processName.

Example:

```
PROCESS NAME=LPD1,  
    PASSWORD=SPINX
```

```
PROCESS NAME=LPD2,  
    PASSWORD=SPINX
```

```
PROCESS NAME=LPD3,  
    PASSWORD=SPINX
```

```
PROCESS NAME=LPD4,  
    PASSWORD=SPINX
```

```
PROCESS NAME=LPD5,  
    PASSWORD=SPINX
```

```
PROCESS NAME=LPD6,  
    PASSWORD=SPINX
```

```
PROCESS NAME=LPD7,  
    PASSWORD=SPINX
```

```
PROCESS NAME=LPD8,  
    PASSWORD=SPINX
```

### 2.3.2.9 Keyword: HOST

HOST statements are used to define a pool of TCP/IP hosts, called the host table. The hosts names defined are used in subsequent LPRALLOW, LPRDENY and QUEUE statements and the alias names may also be used if bannerid masking is enabled or if a LOCALHOST has been defined with the same name as an alias.

The following HOST fields are defined:

ADDRESS=*ipAddress*

... where *ipAddress* is the TCP-IP address of a host specified in standard dotted decimal format (i.e. 131.96.121.5). Use the ADDRESS field to define the IP address of a host when CMS TCPIP-DNR services are not configured or if the host does not have a valid domain name.

NAME=*fullyQualifiedDomainName*

... where *fullyQualifiedDomainName* is a name of a host specified in standard domain name services (DNS) format (i.e. unilpd.gsu.edu). Use this field when CMS TCPIP-DNR services are available. This field value will be resolved by DNS to get the current TCP/IP address for the fullyQualifiedDomainName.

ALIAS=*aliasNames*

This field associates each specific TCP/IP host with one or more alias names that may be used in subsequent LPRALLOW, LPRDENY, QUEUE statements and bannerid masking. The alias name is also used in messages written to the LPD2200 log file. Any of the alias names defined may be used in place of the fully qualified domain name or the ip address. Multiple alias names can be used with Bannerid masking in order to determine the destination host from the Bannerid.

**NOTE:** Use of the NAME and ADDRESS fields are mutually exclusive for any particular HOST statement. Either a NAME or an ADDRESS field, but not both, must be defined for each particular HOST statement.

**NOTE2:** At least one ALIAS value must be defined for each HOST statement. Alternate ALIAS values may be useful if bannerid masking is enabled. (See BANNER and FORMAT keywords for more information on bannerid masking.)

Examples:

```
HOST    NAME=DOCUPRINT.GSU.EDU,
        ALIAS=(DOCUPRINT, dprt)
```

```
HOST    NAME=UNILPD.GSU.EDU,
        ALIAS=UNILPD
```

```
HOST    NAME=HLCLPD.GSU.EDU,
        ALIAS=HLCLPD
```

```
HOST    ADDRESS=131.96.133.33,
        ALIAS=WALTER
```

### 2.3.2.10 Keyword: LPRALLOW

---

The LPRALLOW statement defines a pool of TCP/IP hosts which LPD2200 is allowed to RECEIVE print files from.

The following LPRALLOW field is defined:

```
HOST=[ $ALL$ | $HOSTTABLE$ | hostNameList ]
```

This field defines the list of hosts from which LPD2200 is allowed to accept inbound print jobs. Print requests from all other hosts are rejected. Care should be taken to avoid any conflicts between hosts defined to LPRALLOW and LPRDENY. In case of a conflict, LPRDENY takes precedence. Multiple LPRALLOW statements or multiple host names within one HOST= field are both legal. The following values are defined for the HOST= field:

\$ALL\$ - Inbound print requests from any host is allowed.

\$HOSTTABLE\$ - Inbound print requests only from those hosts defined in the host table (i.e. previously defined hosts in HOST statements above this one are allowed).

*hostNameList* - Inbound print requests are allowed from the host or hosts defined within the list. Host names specified may use alias names defined previously in HOST statements above. If more than one host name is presented, the *hostNameList* must be enclosed within parenthesis characters () and Hostnames be separated from one another with comma characters. These hosts should be defined in the host table using HOST statements.

blank - The LPRALLOW statement is ignored.

**NOTE:** The LPRALLOW statement is only useful if MODE=RECEIVE or MODE=FULL.

Example:

```
LPRALLOW HOST=$HOSTTABLE$
```

### 2.3.2.11 Keyword: LPRDENY

The LPRDENY field defines a pool of TCP/IP hosts which LPD2200 will refuse to RECEIVE print files from. The LPRDENY statement is valid only if the LPRALLOW's "HOST=" value is "\$ALL\$" or "\$HOSTTABLE\$".

The following LPRDENY field is defined:

```
HOST=[ $ALL$ | $NONE$ | $HOSTTABLE$ | hostNameList ]
```

This field defines the list of hosts from which LPD2200 is NOT allowed to accept inbound print jobs. Care should be taken to avoid any conflicts between hosts defined to LPRALLOW and LPRDENY. In case of a conflict, LPRDENY takes precedence. Multiple LPRDENY statements or multiple host names within one "HOST=" field are both legal. The following values are defined for the HOST= field:

\$ALL\$ - All hosts not specifically allowed are denied access.

\$NONE\$ - None of the hosts are denied access.

HostNameList - A subset of the hosts defined within the HOSTTABLE are denied. The HostNameList consists of a list of Host names (aliases may be used) that will not be allowed separated by commas. If more than one Hostname appears, then the list must be contained in parenthesis. These hosts should be defined in the host table using HOST statements above in the PARAM file.

blank - The LPRDENY statement is ignored.

**NOTE:** The LPRDENY statement is only useful if MODE=RECEIVE or MODE=FULL.

Example:

```
LPRDENY HOST=$ALL$
```

### 2.3.2.12 Keyword: PRINTER

---

PRINTER statements are used to define an alias name (useful for bannerid masking), for outbound print processing, or to associate an inbound file to a DSTMAP's statement's PRINTER field so that it can be mapped to the appropriate Unisys queue. The filetype (see below) determines the algorithm used by LPD2200 to convert incoming print files into SDF PRINT\$ format.

The following PRINTER fields are defined:

NAME=*printerName*

...where *printerName* is the name of a printer whose alias name or filetype is to be defined.

ALIAS=*AliasList*

...where *AliasList* is a list of one or more alias names to be assigned to the named printer. If there is more than one alias name provided, the entire alias list must be enclosed within parenthesis. Use of the ALIAS field is optional.

FILETYPE=[**ASCII** | FL130 | XEROXVL | PRISMVL]

This optional field is used to define the parsing algorithm to be used by LPD2200 to transform the received print jobs into Unisys PRINT\$ format.

ASCII - Normal ASCII file (Default).

FL130 - Prism or Solamar fixed length format.

PRISMVL - Prism Variable length format.

XEROXVL - Xerox variable length format.

**NOTE:** The default FILETYPE=ASCII delimits records by ASCII CR, CR/LF, LF and FF characters.

**NOTE2:** ALL inbound files destined for a particular printer name will be processed as though they are in the defined FILETYPE whether or not in fact the file is in the correct format. PRINT\$ files generated by LPD2200 from incorrect input files generally will be unprintable.

Example:

```
PRINTER NAME=SPECIAL1,  
FILETYPE=FL130
```

```
PRINTER NAME=SPECIAL2,  
FILETYPE=XEROXFL
```

```
PRINTER NAME=SYSHPLJ,  
ALIAS=syshp1
```

### 2.3.2.13 Keyword: MACRO

MACRO statements are used to associate auxiliary os2200 file names and processing characteristics with outbound print jobs. The various MACRO names can then be referenced in subsequent FORMAT statements to control the inclusion of optional pre-defined banner, report, trailer and/or control data in outbound file transfers.

A Macro refers to an auxiliary file which may contain data to perform a special function when sent to the printer. The two classes of macro are:

1. Macros with keywords: These macros may include special macro keywords within the macro file which will be replaced dynamically by LPD2200 with appropriate values at the send time. e.g. a banner page which needs to include the userid, account number, date, time, etc. for each print file.
2. Macros with constant data. These macros typically contain constant data that is used to set the printer up in certain ways. For example, these kind of macros may contain a PCL or DJDE sequence to initialize the printer for landscape 132 column duplex printing.

The following MACRO fields are defined:

NAME=*macroName*

...where *macroName* is the name of the macro to be declared.

FILENAME=*os2200fileName.element/version*

...where *os2200FileName.element/version* is a fully qualified name of a 2200 element containing the macro data.

TYPE= [ OMNIBUS | SYMBOLIC ]

This field defines the type of element that this MACRO's FILENAME refers to. The type of element determines how LPD2200's MACRO processing must read data from the element and also determines the type of data that can be inserted within the element. OMNIBUS elements contain an arbitrary length of data. No attempt is made by LPD2200 to impose any type of record structure on OMNIBUS macros. SYMBOLIC macros are inherently record oriented. LPD2200 reads and processes SYMBOLIC macros a record at a time.

**NOTE:** The OMNIBUS elements supported by LPD2200 are a special type we call "F4N" files. F4N files can be created in one of several ways.

1. A macro in the F4N format can be produced by doing a binary mode FTP of a file from a MSDOS PC to the Unisys OS2200 system running Unisys's TAS FTP server.
2. F4N files may also be created by using the supplied MAKERPT2200 utility. MAKERPT2200 is a OS2200 based utility that converts text based input file, called a MAKERPT script, into one or more "F4N" style OMNIBUS elements.

CARRIAGE\_CONTROL = [ LF | **CRLF** | ANSI | NONE ]

This parameter is only used if TYPE = SYMBOLIC (omnibus elements are always output with no carriage control characters). This parameter defines the carriage control character that will precede each symbolic image in the MACRO file/element when it is inserted into the start of the print file. . Valid values (in octal) are:

- LF - A line-feed character (012)
- CRLF - A carriage-return (015) followed by a line-feed (012)
- ANSI - A space character (040)
- NONE - No carriage control character is inserted.

KEYWORD=[ YES | **NO** ]

Flag value indicating whether or not the file referenced by this MACRO should be scanned for possible keyword substitutions. KEYWORD substitution is a special feature that occurs during LPD2200 macro processing. If KEYWORD substitution is enabled, LPD2200 will scan the MACRO for certain defined MACRO KEYWORDS. If a defined MACRO KEYWORD is found, LPD2200 will substitute alternate text in place of the MACRO KEYWORD.

Macro keywords have the following format:

<keywordname> or <#keywordname#>

Surrounding the keyword name with "#" symbols instructs the Macro to replace the regular ASCII letters of the keyword with up to six block letters. These letters are 10 characters wide and 7 characters high. Printing fewer than six block letters will cause the value to be space filled. If the value has more than six characters, it will be truncated after six characters. Thus a value printed in block characters will take up a space on the page of 60 X 7 characters.

Example:

```

SSSSSS  YY    YY    SSSSSS  JJJJJJJJ  HH    HH  YY    YY
SS    SS  YY  YY  SS    SS    JJ    HH    HH  YY  YY
SS          YYYYY  SS          JJ    HH    HH  YYYYY
SSSSSS    YY          SSSSSS    JJ    HHHHHHHH  YY
          SS    YY          SS    JJ    HH    HH  YY
SS    SS  YY          SS    SS  J  JJ    HH    HH  YY
SSSSSS    YY          SSSSSS    JJJ    HH    HH  YY
    
```

The following MACRO keyword names are currently defined:

ACCTID	(12)	- The account number associated with the file.
RUNID	( 6)	- The unique runid associated with the file.
PROJECT	(12)	- The projectid associated with the file.
USERID	(12)	- The userid associated with the file.
INPUTDEVICE	(12)	- The input device name (the STATION LOCAL name from the QUEUE NAME= entry).
DEVICE	(12)	- The device name from the QUEUE PRINTER= entry on which the file is printed.
FILENAME	(30)	- The qualifier*filename(fcycle) of the file.
FORMAT	(12)	- The name of the FORMAT
BANNERID	(12)	- The banner information from the @SYM statement.
PAGES	(6)	- The estimated page count.
COPIES	(4)	- number of copies of the report. This value is passed from SPIN-X RMS
PRTDATE	( 8)	- The date ('mm/dd/yy') the file was processed by .
PRTTIME	( 8)	- The time of day ('hh:mm:ss') the file was processed.
LOCALHOST	(80)	- Name of the Unisys 2200 host as defined in the PARAM file.
C-OPTION	(131)	- C-Option string as defined in the FORMAT statement for this file.
ADDR1thru ADDR6	(64)	- The contents of special RMS-generated user address images
DESC1 thru DESC4	(64)	- The contents of special RMS-generated user description images
PARTNAME	(12)	- The name given to a file on a tape.
PARTNUMBER	(4)	- The reel number of that file.

Examples:

```

NAME=MACRO1,
    FILENAME=LPD2200*MACROS.LS132/PCL,
    TYPE=OMNIBUS,
    KEYWORD=NO

NAME=MACRO2,
    FILENAME=LPD2200*MACROS.PS80/PCL,
    TYPE=OMNIBUS,
    KEYWORD=NO

NAME=BANNER1,
    FILENAME=LPD2200*MACROS.BANNER1/PCL,
    TYPE=OMNIBUS,
    KEYWORD=YES      ; Scan the file for macro keyword
                    ; substitution

```

### 2.3.2.14 Keyword: FORMAT

FORMAT statements are used to associate a number of formatting characteristics to outbound (i.e. SEND) print files. The various FORMAT names can then be referenced in subsequent QUEUE statements.

A special FORMAT called the DEFAULT\_FORMAT may be defined first to define default formatting values for any subsequent FORMAT statements. Fields in FORMAT statements that are not specifically defined will get their values from the corresponding fields of the DEFAULT\_FORMAT. If after processing some formats LPD2200 encounters a second DEFAULT\_FORMAT statement, then the new default values will be applied to formats which follow.

If a DEFAULT\_FORMAT is NOT specified, then any numeric FORMAT field values not explicitly specified default to 0 and any alphanumeric FORMAT fields values default to an empty (i.e. undefined value).

If any field is not defined for a particular format, the default format value is used, so its a good idea to define all fields in the default format and then only define those fields for other formats which are different from the default.

The following FORMAT (and DEFAULT\_FORMAT) fields are defined:

`NAME=formatName`

Name of the format being declared.

`LINES=linesPerPage`

This field defines the default number of lines allowed per page, after which LPD2200 must insert a page break. This value may be superceeded by each print job if the current file retrieved from the OS2200 queue is an SDF PRINT\$ file, and the PRINT\$ contains an 060 control image which specifies an alternate value for lines per page (i.e. an 060 margin control record e.g. M,66,\*,\*). A linesPerPage of value of zero inhibits the insertion of page break characters by LPD2200. The default, if not specified in the FORMAT or DEFAULT\_FORMAT, is 0.

Note: a value of 0 also implies a top margin of 0. H controls (such as from @HDG command) will not print properly unless the top margin is 2 or more.

`TOP_MARGIN=topMarginValue`

This field defines the default top margin in lines reserved per page. This value may be superceeded by each print job if the current file retrieved from the OS2200 queue is an SDF PRINT\$ file, and the PRINT\$ contains an 060 control image which specifies an alternate value for the top margin (i.e. an 060 margin control record e.g. M,\*,5,\*). A topMarginValue must be equal to greater than 2 to allow for the insertion of @HDG text. The default, if not specified in the FORMAT or DEFAULT\_FORMAT, is 0.

`BOTTOM_MARGIN=bottomMarginValue`

This field defines the default bottom margin line reserved per page. This value may be superceeded by each print job if the current file retrieved from the OS2200 queue is an SDF PRINT\$ file, and the PRINT\$ contains an 060 control image which specifies an alternate value for the bottom margin (i.e. an 060 margin control record e.g. M,\*,\*,3). The default, if not specified in the FORMAT or DEFAULT\_FORMAT, is 0.

**NOTE:** The sum of the TOP\_MARGIN and BOTTOM\_MARGIN values are subtracted from the LINES value to define a logical lines-per-page value. The logical lines-per-page value determines how many user data lines will actually be printed per page.

*WIDTH=maxCharactersPerLine*

This field defines the maximum number characters that will be allowed per output record. Any characters in excess of the maximum value will be silently truncated. The default, if not specified in the `FORMAT` or `DEFAULT_FORMAT`, is 0.

`IGNORE_MCONTROL = [ YES/NO ]`

Setting this field to YES will cause LPD2200 not to process any 060 margin controls which may be encountered in the printfile. Any other controls which are concatenated to the mcontrol will be discarded as well.

*LPD\_C\_OPTION=C-CommandText*

This field defines a string of alphanumeric text to be inserted within the control file's "C" record for each outbound print job. The reserved word of \$NONE\$ may be used to inhibit C-CommandText if a `DEFAULT_FORMAT` has been defined and the `DEFAULT_FORMAT` has enabled the `LPD_C_OPTION`.

*BANNER\_SETUP=macroNameList*

This field is used to define one or more MACRO names that should be processed to generate a banner page for outbound print jobs. If more than one macro name is required, separate the names with commas and enclose the entire list within parentheses. The reserved word of \$NONE\$ may be used to inhibit `BANNER_SETUP` processing if a `DEFAULT_FORMAT` has been defined and the `DEFAULT_FORMAT` has enabled `BANNER_SETUP`.

*REPORT\_SETUP=macroNameList*

This field is used to define one or more MACRO names that should be processed to generate report setup data for outbound print jobs. If more than one macro name is required, separate the names with commas and enclose the entire list within parentheses. The reserved word of \$NONE\$ may be used to inhibit `REPORT_SETUP` processing if a `DEFAULT_FORMAT` has been defined and the `DEFAULT_FORMAT` has enabled `REPORT_SETUP`.

*SIXLPI\_SETUP=macroNameList*

This field is used to define one or more MACRO names that should be processed in response to B-Controls calling for a density change to 6lpi in an outbound print job. If more than one macro name is required, separate the names with commas and enclose the entire list within parentheses. The reserved word of \$NONE\$ may be used to inhibit `SIXLPI_SETUP` processing if a `DEFAULT_FORMAT` has been defined and the `DEFAULT_FORMAT` has enabled `SIXLPI_SETUP`.

*EIGHTLPI\_SETUP=macroNameList*

This field is used to define one or more MACRO names that should be processed in response to B-Controls calling for a density change to 8lpi in an outbound print job. Density changes are triggered when 060 B controls are encountered within the report. If more than one macro name is required, separate the names with commas and enclose the entire list within parentheses. The reserved word of \$NONE\$ may be used to inhibit `EIGHTLPI_SETUP` processing if a `DEFAULT_FORMAT` has been defined and the `DEFAULT_FORMAT` has enabled `EIGHTLPI_SETUP`.

*TRAILER\_SETUP=macroNameList*

This field is used to define one or more MACRO names that should be processed to generate trailer data for outbound print jobs. If more than one macro name is required, separate the names with commas and enclose the entire list within parentheses. The reserved word of \$NONE\$ may be used to inhibit `TRAILER_SETUP` processing if a `DEFAULT_FORMAT` has been defined and the `DEFAULT_FORMAT` has enabled `TRAILER_SETUP`.

PSF\_OPERATOR\_MSG=[ YES | **NO** ]

This field defines whether or not SDF PRINT\$ 060/061 special forms control message records will be inserted within the outbound control file. This feature is only useful for remote systems running IBM's PSF6000, an IBM RS6000 AIX based print spooling system. If this feature is enabled (i.e. ON), and a 060 or 061 special forms control is encountered, a record of the form "M-special-forms-message" is inserted within the control file.

PSF\_CONTROL\_FLAG = [ YES | **NO** ]

This feature enables special processing for PSF6000 bound print jobs . This feature if enabled, alters the algorithm used to generate the PSF -o records from values found within type 061 print-control images. For example, when set to YES, a 61 image /subtype 2/operations 1-3 would provide a 4-character value for the Chars (-oCHARS) option, and subtype 17 would modify the default value for the Page Def (-oPAGEDEF) option.

PSF\_PAGEDEF=*pageDefName*

This feature defines a default PAGEDEF value to be inserted within outbound control files. This feature is only useful for remote systems running IBM's PSF6000, an IBM RS6000 AIX based print spooling system. If this feature is enabled, a record of the form "-oPAGEDEF=*pageDefName*" is inserted in the outbound control file. The default value of the *pageDefName* may be superceded by an SDF PRINT\$ 061 control record.

PSF\_FORMDEF=*formDefName*

This feature defines a default FORMDEF value to be inserted within outbound control files. This feature is only useful for remote systems running IBM's PSF6000, an IBM RS6000 AIX based print spooling system. If this feature is enabled, a record of the form "-oFORMDEF=*formDefName*" is inserted in the outbound control file. The default value of the *formDefName* may be superceded by an SDF PRINT\$ 061 control record.

PSF\_CHARS=*charsName*

This feature defines a default CHARS value to be inserted within outbound control files. This feature is only useful for remote systems running IBM's PSF6000, an IBM RS6000 AIX based print spooling system. If this feature is enabled, a record of the form "-oCHARS=*charsName*" is inserted in the outbound control file. The default value of the *charsName* value may be superceded by an SDF PRINT\$ 061 control record.

Examples:

Miminal FORMAT definition

```
FORMAT NAME=MINIMAL
```

Default Format Definition

```
DEFAULT_FORMAT NAME           = STD ,
                LINES          = 66 ,
                WIDTH          = 132 ,
                TOP_MARGIN      = 3 ,
                BOTTOM_MARGIN   = 3 ,
                LPD_C_OPTION     = $NONE$ ,
                BANNER_SETUP    = banner ,
                REPORT_SETUP    = ps80 ,
                TRAILER_SETUP   = $NONE$ ,
                PSF_OPERATOR_MSG = NO
```

PS80 ... HP, Portrait, Simplex, 80 x 66

```
FORMAT NAME           = PS80 ,
                LINES          = 66 ,
                WIDTH          = 80 ,
                BANNER_SETUP    = banner ,
                REPORT_SETUP    = ps80
```

### 2.3.2.15 Keyword: QUEUE

QUEUE statements are used to define the OS2200 queues from which outbound print jobs will be retrieved. QUEUE statements are ONLY needed when Spin-X/LPR-LPD is configured for SEND services. All outbound print jobs are retrieved from these queues for transmission to a remote host. A maximum of 1000 queue statements are allowed.

The following QUEUE statement fields are defined:

NAME=*os2200Queue*

The (required) name of a Unisys OS2200 queue, where users and applications may @SYM their SDF print files to be sent out to a remote host (required). The name "UP" has been reserved for STATUS keyins and may not be used for an LPD Q name.

GROUP = *group-name*

The name of the group to which this queue belongs. If this field is left blank then this queue will default to a group called "default". This feature makes easier the management of queues by enabling an operator to use key-ins such as STAT, INIT, and LOCK on an entire set of queues which have the group-name in common. See Operator Key-ins in Chapter 3.

PRINTER=*printerName*

The (implicitly required) name of the destination printer as defined on a remote host. If bannerid masking is disabled (i.e. BANNER\_MASK=NO) then all print files enqueued to the above queue will be sent with the defined printerName. If bannerid masking is enabled (i.e. BANNER\_MASK=YES) and the current banner mask includes masking for the PRINTER name (i.e. 'PPPP') then the print job's destination printer will be determined from the bannerid's printer name component for each enqueued print job. The printer name component will be tested against the list of ALIAS names defined to the PRINTER statements to determine the actual destination printer name to use.

FORMAT=*formatName*

The (implicitly required) name of a format statement defined previously within the \*PARAM. file. If bannerid masking is disabled (i.e. BANNER\_MASK=NO) then all print files enqueued to the above queue will be formatted with the characteristics of the named FORMAT statement. If bannerid masking is enabled (i.e. BANNER\_MASK=YES) and the current banner mask includes masking for the FORMAT name (i.e. 'FFFF') then the print job's formatting characteristics will be determined from the bannerid's format name component for each enqueued print job. The format name component will be tested against the list of defined FORMAT names to determine the actual formatting characteristics to use.

TYPE = [ **TEXT** / RAW ]

Queues defined as Type "TEXT" (This is the default for SPIN-X LPR/LPD) will have the "f" command in the control (.cfa) file instead of the "l" command. According to the RFC 1179 which has been the basis for the LPR/LPD protocol, the "f" command indicates that the data file will contain only text data while the "l" command indicates binary data. Most LPD daemons will accept binary data in a data (.dfa) file and pass it through unchanged even when the "f" command is in the control file. Some lpd servers, however, including Microsoft NT's TCP/IP Print Service, hold to a stricter interpretation of the RFC 1179. These lpd servers will strip out binary control codes in the data if the control file contains the "f" command. SPIN-X LPR/LPD queues with TYPE = RAW will print to lpd servers such as NT's which require the "l" command for data files which have binary data in them. The "l" command tells the destination lpd server not to filter any control characters but instead to send the raw file to the printer.

SEND\_PROTOCOL = [ **LPR** | MVS\_DOWNLOAD ]

Defines the transmission protocol to be used to send the print files from the Unisys 2200 system across TCP/IP based networks. At present, two protocols are supported:

LPR - the standard LPR/LPD spooling protocol.

MVS\_DOWNLOAD - IBM's protocol to send files to an RS6000 system running PSF6000. This protocol is faster than LPR/LPD.

REMOTE\_PORT = <remote-port-number>

This keyword is used to define the remote port (on the destination host) to which the connection open request is sent. Once the connection is established, all data is sent to this port on the destination host. If the SEND\_PROTOCOL is LPR, then port **515**, used by most Unix machines, is the default. If the SEND\_PROTOCOL is MVS\_DOWNLOAD, this number must match the port number used by the mvsprsd MVS\_DOWNLOAD daemon on the RS6000. The value range is 5001 - 65535. There is no default, so this parameter is mandatory with MVS\_DOWNLOAD.

BANNER\_MASK= [ YES | **NO** ]

Bannerid masking switch (optional, default is NO) indicating whether the destination information, i.e. host name, printer name & format, is to be extracted from the bannerid field or from the HOST, PRINTER and FORMAT fields of this QUEUE statement. If bannerid masking is enabled (i.e. BANNER\_MASK=YES) then the bannerid field of the @SYM command will be used initially to define destination information. If the current banner mask (defined previously with the BANNER statement) does not include a mask for the one or more components of the destination information, then the HOST, PRINTER and/or FORMAT fields of the current QUEUE statement are used, if defined. For example if the current banner mask is defined as PFFFFFFFFF, then the mask contains no host information and by default the destination host name is taken from the HOST field of this QUEUE statement.

HOST=*aliasName*

An (implicitly required) ALIAS name associated with a remote host defined previously in a HOST statement. If bannerid masking is disabled (i.e. BANNER\_MASK=NO) then all print files enqueued to the above queue will be sent to the previously named host. If bannerid masking is enabled (i.e. BANNER\_MASK=YES) and the current banner mask includes masking for the HOST name (i.e. 'HHHH') then the print job's destination host name will be determined from the bannerid's host name component for each enqueued print job. The host name component will be tested against the list of ALIAS names defined to HOST statements to determine the actual destination host.

CONTROL\_SETUP=(*macroname list*)

This field is used to define one or more MACRO names that should be processed in order to generate statements which would be included in the LPD control file for any files sent from this queue. This feature could be used to specify the format in a J string for instance. MACROs used for files on this queue must be of type SYMBOLIC (OMNIBUS MACROs are not valid) and must contain the statement CARRIAGE\_CONTROL=LF. Keywords can be used in the MACROs. If more than one MACRO name is required, separate the MACRO names by commas and enclose the list in parentheses.

SPECIAL\_FORM\_MSG= [ **IGNORE** | REQUE\_MSG | TOHOLDQ ]

Defines how to handle 060 or 061 type print control records having a special forms request message.

Valid values are:

**IGNORE** - (Default) Ignore the special forms message and send the print file to the remote host.

**REQUE\_MSG** - Send the first message encountered in the print file to the operator's console and ask the operator if the job should be sent to the remote host or requeued to a different Unisys queue on the local host. If it is to be requeued to local host, the operator is prompted to enter a Unisys queue name.

**TOHOLDQ** - Move the print file to the LPD hold queue.

OPERATOR\_REMOTE\_QUEUE=*queueName*

Defines an alternate remote queue (also called the printer name) to which to send the print file if a special forms message was encountered in the print file BEFORE the first data image. This allows print files with operator intervention messages to be placed on a different remote queue. If this field is defined and a print file having a special forms message is retrieved from this queue, the destination printer name will be replaced by the OPERATOR\_REMOTE\_QUEUE *queueName*.

MAX\_ACTIVE = [ 1-10 ]

Specifies the maximum number of print files that can be transmitted at the same time for this print queue. The maximum is 10 and the default is 2. If throughput is not an issue for a print queue, set this value to 1 to minimize memory and processor utilization. A larger value may maximize the throughput depending on the capacity of the remote host being used. If the LPDLOG file indicates frequent rejects of open attempts, this number should be reduced to alleviate excessive print file requeuing.

PHANTOM = [ YES | NO ]

If set, this Q is known only to LPD2200 and is not known to the EXEC. This feature can be used only in conjunction with the JOBPOOL feature.

INIT = [ UP | DOWN ]

If set to "UP" this Q will be initiated on startup, regardless of whether the Q option (lock all Qs on startup) has been put on the run card. If "DOWN", the Q will be locked on startup.

NOTIFY= [ YES | NO ]

If set to YES, a console message will be printed when a file begins and ends processing.

Example:

```

QUEUE  NAME           = Q611,
        HOST           = docutech,
        PRINTER        = syshplj,
        FORMAT         = PD80,
        BANNER_MASK    = YES,      ; if yes, mask must be defined
        INIT           = UP

```

### 2.3.2.16 Keyword: JOBPOOL

A Jobpool is a collection of LPD2200 Queues which are accessed via a single Job Pool Queue (JPQ). Depending on Jobpool configuration and the print loads of members, one of the Job Pool Members will be selected by LPD2200 to print a job @sym'd to the named JPQ. The Jobpool can be configured so that the printer selected will be the one LPD2200 believes to be the least busy printer in the pool. Locked queues will not receive print: thus printers in a job pool can be taken down for servicing without interrupting the flow of print or requiring redirection of prnfiles. The JOBPOOL definition must follow the QUEUE statements in the LPD\*PARAM. file.

```
JOBPOOL NAME = lpd_q_name, ; a Unisys queue defined to LPD2200
        POOL = (q1,q2,q3.....,q10), ; more (as many as 10) Qs
        THRESHOLD = (JPQ threshold, q1 threshold, q2.....,q10 threshold),
        SELECT_FIRST = (Y/N),
        CARRY_PRINTER = (Y/N),
        PASS_THRU = (Y/N)
```

```
NAME = lpd_q_name,
```

Files @sym'd to the queue named in the "NAME = " field will be distributed to members of the pool based on how busy the pool members are and whether they are locked or not. We refer to this queue as the Job Pool Queue (JPQ). The PARAM file formatting for the JPQ will be used on the pool member that gets the job and this will be reflected on the banner. The Input Queue name on the banner will also be that of the JPQ. However the host and printer name belonging to the pool member will still be reflected on the banner.

Jobs sent to the Job Pool Q by default will be printed on that Q if it is not busy. If the JPQ is busy the 1st idle Q that is found or else the least busy member of the pool will be selected to print the job instead. However if some application other than LPD2200 prints on this Q, LPD2200 will not notice the extra print traffic. Job Pool logic only applies to LPD2200 Qs.

```
POOL = (q1,q2,q3.....,q10),
```

As few as one or as many as 10 queues may be included in the pool. These must be legal Unisys queues which have been previously configured in the PARAM file. The JPQ can also be included in it's own pool as a member. Other JPQs may be included in the member list but they will be treated as members only and the pools of Qs which belong to them will not be accessed. A reasonable ordering of the pool list would be to put the most efficient printers 1st and set their corresponding thresholds higher than the ones that follow.

```
THRESHOLD = (JPQ threshold, q1 threshold, q2.....,q10 threshold)
```

Threshold values are positive integers which should roughly correspond to the pages per minute ratings of the printers. They can be set to any arbitrary value however depending on the behavior which is desired. The 1st threshold value must belong to the JPQ itself. If zero, the JPQ will not print files but will pass them to the pool. The other thresholds are associated with the pool members in the same order as the members appear in the pool list. The larger the threshold, the more print will be able to pile up in the printer's input buffer. The smaller the threshold, the longer LPD will wait before concluding that the printer is finished printing. In practice, the user will need to experiment with different threshold values in order to get the desired behavior.

```
SELECT_FIRST = (Y/N)
```

If set to Yes, then the default JPQ behavior is changed so that jobs sent to the JPQ will instead always print on the 1st Q in the pool member list. If that Q is locked however, the job will go to a different pool member.

CARRY\_PRINTER = (Y/N)

Some remote systems (such as the Xerox DocuPrint) can associate formatting characteristics with the Printer name that is sent in the control file. If this is the case, then simply swapping the format from the JPQ will not be enough to get the proper formatting. If CARRY\_PRINTER is set to 'Y', the printer name will be carried forward from the JPQ and will appear in the control file. This will only work if the remote system has a printer name which matches the JPQ printer name!

PASS\_THRU = (Y/N)

If set to 'Y' the JPQ can be locked but jobs sent to the JPQ will be distributed instead to unlocked pool members, unless they too are locked.

On the QUEUE statement there is a field called "PHANTOM" which can be set to "YES" or "NO" (the default). Phantom Queues are LPD2200 Qs which have not been configured to the Exec. Phantom Qs can be configured as Pool members if the user wishes these Q names to be accessible only to LPD2200 Job pools and not to the larger community of OS2200 users.

**Example1:**

```
JOBPOOL NAME = UNIS_Q,      ; a legal Unisys Q configured to Exec and LPD
      POOL = (LPD1,LPD2,LPD3), ; if PHANTOM, Exec doesn't know them
      THRESHOLD = (25,25,25,25) ; ppm ratings of printers in pool - the first
      ; member threshold is of the JPQ itself.
```

The above is a "plain vanilla" job pool. A file sym'd to UNIS\_Q will print on one of the printers defined to UNIS\_Q, LPD1, LPD2, or LPD3 with the format defined by UNIS\_Q. LPD2200 thinks that all the pool members are capable of printing 25 pages per minute. The "weighting" algorithm is biased to the front of the pool list - the JPQ should get a bit more print than each of the other pool members. Small adjustments to the threshold values will cause the print distribution to change. The higher the threshold, the more print will go to that pool member. If one pool member is locked, the other pool members pick up the slack. If JPQ is locked, nothing prints.

**Example2:**

```
JOBPOOL NAME = UNIS_Q,
      POOL = (LPD1, LPD2),
      THRESHOLD = (0,100,100),
      SELECT_FIRST = YES,
      CARRY_PRINTER = YES
```

This user has two Docuprints rated at 100 ppm. The Qs LPD1 and LPD2 are phantom Qs (see Q statement) - only LPD2200 can access these "Qs". With SELECT\_FIRST, normally all the print will arrive at the host and printer defined in the Q statement for LPD1. But with CARRY\_PRINTER set to YES, LPD2200 will send the file to the printer defined for UNIS\_Q instead. Setting the JPQ threshold to zero keeps all files from printing on that Q.

Now the Xerox controller on both Docuprint hosts has been configured with the same printer name as for UNIS\_Q. (You must always do this if you use "CARRY\_PRINTER"). Formatting on the Docuprint of files for these "printers" is identical for both machines. Now suppose a 200,000 page report arrives on UNIS\_Q. This file will keep the LPD1 printer busy for 2000 minutes.

The user locks LPD1 with a console keyin. All jobs which arrive on UNIS\_Q will now be automatically switched to LPD2. This Q has a different host (the other Docuprint) but will use the printer from the UNIS\_Q QUEUE definition.

In essence, the JOBPOOL feature is used to make two network printers behave like two channel printers which have been assigned to the same DEVICE in the EXEC. When one is locked, the other prints. If SELECT\_FIRST had not been set, the analogy would be even more perfect, because then when one printer was busy, the LPD2200 would choose the other, just as the EXEC would.

**The following conditions apply to Job Pool Qs.**

If the Job Pool Q is locked, the jobs destined for the locked Q will not be printed by other members of the pool - unless PASS\_THRU= YES.

If the Job Pool Q is not busy, prints will always go to that Q. If it is desired that only the members of the pool receive approximately equal amounts of print, then the Job Pool Q should be locked, PASS\_THRU set to 'Y', and the thresholds of the members set to the same value. Setting the JPQ (1st) threshold to zero will also accomplish this. The MAX\_ACTIVE field of the Q parameter can also affect distribution of jobs when many files arrive on the Job Pool Q at about the same time. LPD2200 can concurrently process more files for a Q when the max\_active parameter is increased.

Enabling banner\_id masking for Job Pool Qs will disable job pooling. Banner masking is still the highest priority and overrides U controls from SPIN-X Report Management System, Unisys print controls in the report data or Param file configurations.

The data structure holding Job Pool data is implemented dynamically as a linked list. This was done to save memory because it is assumed that usually there will not be a lot of job pools. However a user may if he wishes configure as many job pools as there are Qs. The limit is 1000. The limit for pool members has arbitrarily been set to 10.

Since LPD has no way of knowing how busy the printer might be it must estimate how long the printer will take to print a job based on the pages sent and the threshold (pages per minute rating) of the pool member. "Pages sent" is taken from the SMOQUE entry's page count estimate. These "Unisys pages" are rather unpredictable. For instance, an ECL may have 10 lines but Unisys may think there are 300 pages. But straight text without "@" signs goes very evenly. When LPD2200 thinks a printer has been idle for a time equal to 25% of the time taken to print the pages that have been sent to it, then the page count and time elapsed values are reset. If a threshold is set too low, the numbers will not often reset whereas if it is too high, LPD2200 will think the printer is idle when it is really busy and therefor will send the printer more jobs.

**Technical Note:** LPD2200 is a multitasking program. The MAX\_FILES\_ACTIVE field of the OPTIONS parameter can be set as high as 40 (or as low as 1), meaning that 40 files on various Qs can be printed concurrently. Each of these Q activities kicks off several other tasks. Each individual Q parameter has a MAX\_ACTIVE field which determines how many files can concurrently be sent to a Q. These max values primarily impact how much system resources can be soaked up by LPD2200, and not how busy the printer may be. (LPD can of course send print much faster than even the fastest printer can process it). When a Q (but not necessarily a JPQ) is locked, jobs for that Q wait (no SMOQUE get\_file is issued) or if it has reached the limit of activities for that Q the file will also wait.

Normally when a Q is not locked then a file is retrieved from SMOQUE and the Q activity is started. But the activity for a Job Pool Q with PASS\_THRU set must be started even if it is locked in order to retrieve the file from SMOQUE and swap it to one of the Q members. Only the activity count of the Q that actually gets the file will be increased. Internally, it makes absolutely no difference to LPD which Q gets the file.

### 2.3.2.17 Keyword: DSTMAP

DSTMAP statements are used to define "virtual" printer names available to remote users through LPD2200. LPD2200 will refuse to accept inbound print requests unless the printer name requested matches a DSTMAP statement's PRINTER field value. Each unique PRINTER value field defines a "resource" that is available to the world.

DSTMAP statements are only necessary when LPD2200 is configured for "RECEIVE" or "FULL" mode. LPD2200 configured for "SEND" mode does not require DSTMAP statements.

As originally implemented, LPD2200 had no DSTMAP command. The "published" printer names were simply the OS2200 queue names. This meant that users could potentially deposit files on any OS2200 queue (or device) configured. The DSTMAP statements were created to insulate OS2200 queues from remote LPR users who were able to enqueue files to any arbitrary OS2200 queue.

This statement defines a destination Map (DSTMAP) for the inbound print jobs (LPR requests). It maps the inbound printer-format combination to the appropriate Unisys queue on the local host and optionally associates a "virtual" printer name and "format" name with the received print job. This is accomplished by inserting the configured alternate printer and alternate format names into SPIN-X type U-control images within the LPD2200 generated PRINT\$ file, replacing the original destination printer and format names.

The following DSTMAP fields are currently defined:

`PRINTER=printerName`

The printer name is used to define the published (or "public") name that LPD2200 will accept as a valid printer name for inbound print jobs. This name is a required field. If an inbound print job's destination printer name is NOT defined within any DSTMAP statement PRINTER values, the inbound print request is rejected with the message "printer name not defined". The PRINTER field along with the DEFAULT and FORMAT fields are used by LPD2200 to disambiguate (or demultiplex) inbound print requests destined for the same PRINTER name to various OS2200 queues. If a set of DSTMAP statements define the same PRINTER value then one of the DSTMAP commands MUST include the DEFAULT=YES field. Use of multiple DSTMAP commands that define the same printer name is a somewhat sophisticated technique that potentially can have the following benefits:

1. Minimize the number of unique OS2200 queues required to be defined to LPD2200 for depositing incoming print jobs.
2. Minimize the number of "published" printer names required to route jobs to a unique queue.

The DSTMAP statement has the following parameters:

`DEFAULT=[ YES | NO ]`

This is an optional field, which when set to TRUE, indicates that this particular DSTMAP definition is to be used as the default DSTMAP value for all jobs destined to the PRINTER value defined in this default DSTMAP. Only one DSTMAP statement with any particular PRINTER value can include the DEFAULT field.

The first DSTMAP statement defined for each unique PRINTER value is the implied default if no explicit DSTMAP statement includes a DEFAULT=TRUE field.

FORMAT=*formatName*

The FORMAT field is used to disambiguate inbound printer requests with the same printer name to alternate OS2200 print queues. This feature requires that the remote user request that a specially formatted "J" or "job" record having the command -J "FORM=*formatName*" be put into the control file when the sending of the print job is initiated on the OS2200 system. This can be accomplished with the CONTROL\_SETUP field of the QUEUE statement and an appropriate MACRO containing the J record.

If the format on the DSTMAP command matches the format in the "J" record and no ALTFORMAT is specified, the configured FORMAT value is put into the LPD2200 generated PRINT\$ file as a Spin-X type 'U' 060 control image. If no match is found, no format will be put in the 'U' 060 image. There is only one 'U' 060 image in the PRINT\$ file which specifies format. It can have a value or it can be blank.

QUEUE=*os2200QueueName*

The Unisys queue corresponding to the above printer-format combination i.e. an inbound LPR request for the above printer and format would result in the print file being SYM'd to this Unisys queue.

ALTPRINTER=[ *alternatePrinterName* | \$NONE\$ ]

The (optional) ALTPRINTER field specifies an Alternate printer name to override or inhibit the original printer name that would be inserted into the SPIN-X type 'U' 060 control images within the PRINT\$ file that the LPD2200 generates. If this field is \$NONE\$, a blank U-control (no-op) is written. The primary purpose of the PRINTER field is to define a public name by which remote users enqueue files to OS2200 queues. The ALTPRINTER field allows LPD2200 to insert a printer name record that is more appropriate for the final purposes of the print job.

This feature is also particularly useful when LPD2200 is being used as a centralized printer server. In a centralized print server configuration, LPD2200 is set up to relay files received from one host system to another host system. For complex print routing requirements the embedded printer and format names associated with each print job can help minimize the number of unique queues required. SPIN-X/Central, SPIN-X/RMS, SPIN-X/RPF (i.e. Xpress) and Spin-X/LPR-LPD (i.e. LPD2200) all can use the SPIN-X PRINTER and FORMAT type 'U' 060 control images.

ALTFORMAT=[ *alternateFormatName* | \$NONE\$ ]

If the Alternate format name is specified, this name replaces the above original format name to be inserted into the print file's U-control image. If this field is \$NONE\$, a blank U-control (a no-op) is written.

**NOTE:** The value parameters defined to the PRINTER field ARE CASE SPECIFIC. LPD2200 will NOT accept inbound print requests to a printer whose spelling differs only in case.

Examples:

```
DSTMAP PRINTER      = f1130 ,
        FORMAT      = STD ,
        QUEUE       = q624 ,                ; Hold queue
        ALTPRINTER  = $NONE$ ,
        ALTFORMAT   = $NONE$ ,
        DEFAULT    = YES
```

```
DSTMAP PRINTER=RLSR1 ,
        FORMAT=PS80 ,
        QUEUE=RL602 ,
        ALTFORMAT=RL602
        DEFAULT=YES
```

## 2.4 Sample LPD Parameter File

```
;***** [qual]*PARAM. - Sample LPD2200 Parameter File *****
```

```
; This sample PARAM file was extracted from the production PARAM file used
; at Georgia State University. It must be edited for your site configuration
; before attempting to start the LPD2200 server run.
```

```
LOGFILE NAME = LPDLOG
```

```
OPTIONS WRITETOCONSOLE = YES,
        OUTPUTBLOCKSIZE = 6144,
        PRINTBANNER     = NO,
        ALLOW_CONTROL_ERRORS = YES,
        ALLOW_TAPES     = YES,
        LOGDETAILS      = HIGH
```

```
OPERATION MODE = FULL
```

```
HOST NAME = DOCUTECH.GSU.EDU, ALIAS = (DOCUTECH, docu)
HOST NAME = PANTHER.GSU.EDU, ALIAS = (PANTHER, PANT)
HOST NAME = DOCUPRINT.GSU.EDU, ALIAS = (DOCUPRINT, dprt)
HOST NAME = UNILPD.GSU.EDU, ALIAS = UNILPD
HOST NAME = HLCLPD.GSU.EDU, ALIAS = HLCLPD
```

```
HOST ADDRESS=131.96.133.33,
      ALIAS=WALTER
```

```
LOCALHOST NAME = LPD2200
```

```
PROCESS NAME = LPD1, PASSWORD = SPINX
PROCESS NAME = LPD2, PASSWORD = SPINX
PROCESS NAME = LPD3, PASSWORD = SPINX
PROCESS NAME = LPD4, PASSWORD = SPINX
PROCESS NAME = LPD5, PASSWORD = SPINX
PROCESS NAME = LPD6, PASSWORD = SPINX
PROCESS NAME = LPD7, PASSWORD = SPINX
PROCESS NAME = LPD8, PASSWORD = SPINX
```

```
HOLDQUEUE NAME = DSI9
```

```
CONSOLEKEY NAME = LPD*
```

```
LPRALLOW HOST = ($ALL,$)
```

```
LPRDENY HOST = ($NONE$)
```

```
BANNER MASK = HHHHPPPPFFFF
```

```
PRINTER NAME = spinx, ALIAS = spn
PRINTER NAME = docuprint, ALIAS = (dpr, dp)
```

```
MACRO NAME = BANNER,
  FILENAME = LPDPRD*SETUP.BANNER/PCL,
  TYPE = OMNIBUS,
  KEYWORD = YES

MACRO NAME = XGFBANNER,
  FILENAME = LPDPRD*SETUP.BANNER/XGF,
  TYPE = SYMBOLIC,
  KEYWORD = YES

MACRO NAME = PS80,          FILENAME = LPDPRD*SETUP.PS80/PCL,
  TYPE = OMNIBUS

MACRO NAME = PD80,          FILENAME = LPDPRD*SETUP.PD80/PCL,
  TYPE = OMNIBUS

MACRO NAME = LS132,        FILENAME = LPDPRD*SETUP.LS132/PCL,
  TYPE = OMNIBUS

MACRO NAME = LD132,        FILENAME = LPDPRD*SETUP.LD132/PCL,
  TYPE = OMNIBUS

MACRO NAME = LD132G,      FILENAME = LPDPRD*SETUP.LD132G/PCL,
  TYPE = OMNIBUS

MACRO NAME = PS80W,        FILENAME = LPDPRD*SETUP.PS80W/PCL,
  TYPE = OMNIBUS

MACRO NAME = PS80_XGF,    FILENAME = LPDPRD*SETUP.PS80/XGF,
  TYPE = SYMBOLIC

MACRO NAME = PD80_XGF,    FILENAME = LPDPRD*SETUP.PD80/XGF,
  TYPE = SYMBOLIC

MACRO NAME = LS132_XGF,   FILENAME = LPDPRD*SETUP.LS132/XGF,
  TYPE = SYMBOLIC

MACRO NAME = LD132_XGF,   FILENAME = LPDPRD*SETUP.LD132/XGF,
  TYPE = SYMBOLIC

MACRO NAME = GRAYBR_XGF,
  FILENAME = LPDPRD*SETUP.GRAYBR/XGF,
  TYPE = SYMBOLIC

MACRO NAME = PS80A_XGF,   FILENAME = LPDPRD*SETUP.PS80A/XGF,
  TYPE = SYMBOLIC
```

```
DEFAULT_FORMAT NAME = STD, LINES = 66, WIDTH = 132,  
  TOP_MARGIN         = 3, BOTTOM_MARGIN     = 3,  
  LPD_C_OPTION       = $NONE$, BANNER_SETUP = BANNER,  
  REPORT_SETUP       = LS132, TRAILER_SETUP = $NONE$  
  
;  
;  
;           PCL, Portrait, Simplex, 80 x 66  
;  
  
FORMAT NAME = PS80, LINES = 66, WIDTH = 80,  
  REPORT_SETUP = PS80  
  
;  
;  
;           PCL, Portrait, Simplex, 80 x 66  
;  
  
FORMAT NAME = PS80A, LINES = 66, WIDTH = 80,  
  REPORT_SETUP = PS80  
  
;  
;  
;           PCL, Portrait, Duplex, 80 x 66  
;  
  
FORMAT NAME = PD80, LINES = 66, WIDTH = 80,  
  REPORT_SETUP = PD80  
  
;  
;  
;           PCL, Landscape, Simplex, 132 x 66  
;  
  
FORMAT NAME = LS132, REPORT_SETUP = LS132  
  
;  
;  
;           PCL, Landscape, Duplex, 132 x 66  
;  
  
FORMAT NAME = LD132, BANNER_SETUP = BANNER,  
  REPORT_SETUP = LD132  
  
;  
;  
;           PCL, Landscape, Duplex, 132 x 66, Gray bar  
;  
  
FORMAT NAME = LD132G, REPORT_SETUP = LD132G  
  
;  
;  
;           PCL, Portrait, Simplex, 80 x 66, Line Wrap  
;  
  
FORMAT NAME = PS80W, LINES = 66, WIDTH = 998,  
  REPORT_SETUP = PS80W  
  
;  
;  
;           Docuprint XGF, Simplex, 80 x 66  
;  
;
```

```
FORMAT NAME = XPS80, LINES = 66, WIDTH = 80,  
REPORT_SETUP = PS80_XGF,  
BANNER_SETUP = XGFBANNER  
  
;  
;  
; Docuprint XGF, Duplex, 80 x 66  
;  
  
FORMAT NAME = XPD80, LINES = 66, WIDTH = 80,  
LPD_C_OPTION = ""(duplex)"" ,  
BANNER_SETUP = XGFBANNER,  
REPORT_SETUP = PD80_XGF  
  
;  
;  
; Docuprint XGF, Simplex, 132 x 66  
;  
  
FORMAT NAME = XLS132, BANNER_SETUP = XGFBANNER,  
REPORT_SETUP = LS132_XGF  
  
;  
;  
; Docuprint XGF, Duplex, 132 x 66  
;  
  
FORMAT NAME = XLD132, BANNER_SETUP = XGFBANNER,  
REPORT_SETUP = LD132_XGF  
  
;  
;  
; Docuprint XGF, Duplex, 132 x 66 - Gray Bar  
;  
  
FORMAT NAME = XGRAYBR, LPD_C_OPTION = ""(duplex)"" ,  
BANNER_SETUP = XGFBANNER,  
REPORT_SETUP = GRAYBR_XGF  
  
;  
;  
; Docuprint XGF, Simplex, 80 x 66  
;  
  
FORMAT NAME = XPS80A, LINES = 66, WIDTH = 80,  
BANNER_SETUP = XGFBANNER, REPORT_SETUP = PS80A_XGF
```

```
QUEUE NAME = PRTTAA, HOST = docutech, PRINTER = spinx,  
  FORMAT = LS132  
  
QUEUE NAME = PRTTAB, HOST = docutech, PRINTER = spinx,  
  FORMAT = PS80  
  
QUEUE NAME = Q611,   HOST = docutech, PRINTER = spinx,  
  FORMAT = PD80  
  
QUEUE NAME = Q621,   HOST = docutech, PRINTER = spinx,  
  FORMAT = LD132  
  
QUEUE NAME = Q206,   HOST = docutech, PRINTER = spinx,  
  FORMAT = LD132G  
  
QUEUE NAME = Q612,   HOST = docutech, PRINTER = spinx,  
  FORMAT = PS80W  
  
QUEUE NAME = Q819,   HOST = panther,   PRINTER = spinx,  
  FORMAT = PS80A  
  
;  
;   ***** DOCUPRINT Queues *****  
;  
  
QUEUE NAME = LPDT01, HOST = docuprint, PRINTER = docuprint,  
  FORMAT = XPS80  
  
QUEUE NAME = LPDT02, HOST = docuprint, PRINTER = docuprint,  
  FORMAT = XPD80  
  
QUEUE NAME = LPDT03, HOST = docuprint, PRINTER = docuprint,  
  FORMAT = XLS132  
  
QUEUE NAME = LPDT04, HOST = docuprint, PRINTER = docuprint,  
  FORMAT = XLD132  
  
QUEUE NAME = LPDT05, HOST = docuprint, PRINTER = docuprint,  
  FORMAT = XGRAYBR  
  
QUEUE NAME = LPDT06, HOST = docuprint, PRINTER = docuprint,  
  FORMAT = XPS80A
```

---

DSTMAP PRINTER = spinx, FORMAT = LS132, QUEUE = PRTTAA

DSTMAP PRINTER = spinx, FORMAT = PS80, QUEUE = PRTTAB,  
DEFAULT = TRUE

DSTMAP PRINTER = spinx, FORMAT = PS80W, QUEUE = Q612

DSTMAP PRINTER = spinx, FORMAT = PD80, QUEUE = Q611

DSTMAP PRINTER = spinx, FORMAT = LD132, QUEUE = Q621

DSTMAP PRINTER = spinx, FORMAT = PS80A, QUEUE = Q819

DSTMAP PRINTER = spinx, FORMAT = LD132G, QUEUE = Q206

DSTMAP PRINTER = docuprint, FORMAT = XPS80, QUEUE = LPDT01

DSTMAP PRINTER = docuprint, FORMAT = XPD80, QUEUE = LPDT02

DSTMAP PRINTER = docuprint, FORMAT = XLS132, QUEUE = LPDT03

DSTMAP PRINTER = docuprint, FORMAT = XLD132, QUEUE = LPDT04

DSTMAP PRINTER = docuprint, FORMAT = XGRAYBR, QUEUE = LPDT05

DSTMAP PRINTER = docuprint, FORMAT = XPS80A, QUEUE = LPDT06

## 2.5 OS2200 Configuration and TCP-IP Setup

### 2.5.1 CMS Parameters

The following CMS configuration statements are required for LPD print file processing:

Add TSAM PROCESS statements, which are referenced by the LPD PARAM file. A single PROCESS statement is required. If another copy of LPR/LPD is run or if the PROCESS statement can be used by another application it would be better to have more than one PROCESS statements. For sites with requirements for many PROCESS statements, the maximum number of PROCESS statements allowed in CMS may be insufficient, in which case a second CMS may be installed.

```
PROCESS,LPD1 TYPE,TSAM PASSWORD,SPINX INTERNET-ADR,INTH1
PROCESS,LPD2 TYPE,TSAM PASSWORD,SPINX INTERNET-ADR,INTH1
: -Lpd process statements have no T-selector parameter!
PROCESS,LPD10 TYPE,TSAM PASSWORD,SPINX INTERNET-ADR,INTH1
```

The INTERNET-ADR name and address can be used for multiple purposes. For instance it might be shared with TIP. Add the appropriate INTERNET-ADR statement defining the Internet address of this Host. e.g.

```
INTERNET-ADR,INTH1      IP,IP1,132.52.153.80 ; .LPD (DCP)
                        IP,IP2,199.99.12.225 .LPD (HLC)
```

Add the appropriate IP, FEP, and PATH statements linking this Host to the DCP. e.g.

```
IP,IP1      LINK-LAYER,FEP1 ;
            INTERFACE-TYPE,DCP-CHANNEL ;
            GATEWAY,132.52.153.79 ;
            SUBNET,YES,255.255.255.0,1492 ;
            MTU,1492

FEP,FEP1    OWNER,DCP PATH PATH0 LOAD,TELCON*LOAD.GSU$PRC1;
            FRAME-SZ,2400 DUMP,TELCON*DUMPPFILE.
            RETRY-DOWN-MSG,NO ALL-CONS-MSG,YES

PATH,PATH0  INPUT-NODE,FEP1 OUTPUT-NODE,FEP1 OWNER,FEP1;
            STATUS,UP CHECKSUM,OFF TYPE,BLOCK
```

Add IP and LAN statements linking this host to the HLC.

```
IP,IP2      LINK-LAYER,HLCETH ;
            INTERFACE-TYPE,IEEE802.3 ;
            GATEWAY,199.99.12.1 ;
            SUBNET,YES,255.255.255.0,1492 ;
            MTU,1492

LAN,HLCETH  INPUT-NODE,HLTH02 OUTPUT-NODE,HLTH01 ;
            LOCAL-ADR,X08000be00501 ;
            LOAD,SY$LIBS*HLC2-ETHER.
```

An optional TCPIP-DNR statement can be used to relate internet domain names to TCP/IP addresses. If this statement is not used, TCP/IP addresses must be defined in the LPD PARAM file.

```
TCPIP-DNR   LOCAL-SERVER,RSVL.UNISYS.COM,192.61.252.1 ;
            SEARCH-ORDER,SERVERS-ONLY
```

Add TCP/IP to the STATIC-BANKS statement. e.g.

```
STATIC-BANKS TCP/IP
```

## 2.5.2 TELCON Parameters

The following TELCON configuration statements are required for LPD print file processing when using a DCP:

Add a SUBNET statement linking this DCP to the Host, with IPROUTER=YES. e.g.

```
FEPA PRCSR NA = 1;
DCP1 NETADR NA = 1;
CHASUB SUBNET PRCSR=PRC1,TYPE=IPCHAN,;
                IPNETID=(132,52,153,78),;
                CHANNEL=CHAN0,IPROUTER=YES,DGFSIZE=1492,;
                SUBNMASK = (255,255,255,0)
```

Add IPROUTER=YES to the existing LAN SUBNET statement. e.g.

```
LANSUB SUBNET PRCSR=PRC1,TYPE=LAN,IPNETID=(132,52,153),;
                LINE=LINE26,IPROUTER=YES,RIP=YES,DGFSIZE=1492
```

Verify that the DCP IP address matches the Gateway IP address in the CMS configuration. e.g.

```
IPLAN IPADR PRCSR=PRC1,DCAEP=DCP1,;
                IPADDR1=(132,52,153,79,LOCAL)
```

If needed add a LAN Gateway SUBNET statement. e.g.

```
TCPGWY SUBNET PRCSR=PRC1,IPNETID=DEFLTGWY,;
                IPGATEWY=(1,132,52,153,1)
```

## 2.5.3 Configuring the Exec

Print queues serviced by SPIN-X LPR/LPD must be configured to the system before they can be used by LPD2200. Even "phantom" queues which are known only to LPD2200 must ultimately make use of a legal Unisys print queue.

### 2.5.3.1 Defining Type 2 Symbiont Class Local Stations

LPD uses STATION LOCALs for its various print queues as follows:

1. Input queues to which print files are @SYM'd.
2. Hold queues for files that cannot be processed due to improper file format, unknown printer name, etc.

**Station INCL11 Local**

**Station SPXCL1 Local**

**Station HOLDQ1 Local**

## 2.5.4 LPD Runstream

The following sample run-stream is suggested, which should be placed in SYS\$LIB\$\*RUN\$ to ensure that the correct account number and userid is used.

```
@RUN,A/EFG LPDTST,B666030-0845/SYS,SPINX,99999,9999/9999
@ .
@ . ST LPDTST      Execute LPD2200 with no options
@ . ST LPDTST,1    Execute LPD2200 with D option
@ . ST LPDTST,2    Execute LPD2200 with H option
@ . ST LPDTST,3    Execute LPD2200 with L option
@ . ST LPDTST,3    Execute LPD2200 with Q option
@ . ST LPDTST,anyother Shame on you
@ .
@QUAL LPDTST      . Define the default qualifier
@ASG,A *PARAM.
@ASG,A *JOBNUM.
@PRIV
@SYM,DF PRINT$
@CYCLE,C *LPD-PRINT.,5
@CAT,P *LPD-PRINT(+1),F///9999
@BRKPT PRINT$,*LPD-PRINT. . BRKPT the print file to LPD-PRINT
@CYCLE,C *LPDLOG.,5 . Assign a new log file fcycle
@CAT,P *LPDLOG(+1),F///9999
@USE LPDLOG,*LPDLOG.
@CHG,V LPDLOG.
@ASG,A LPDLOG.
@ .
@ . Move the PARAM file into a temporary file so
@ . The LPDTST*PARAM file will be backed up by FAS.
@ .
@ASG,T CORSAIRPARAM.
@COPY *PARAM.,CORSAIRPARAM.
@FREE *PARAM.
@USE PARAM,CORSAIRPARAM.
@ .
@ . Create a new fcycle of the JOBNUM file, so at least the
@ . previous f-cycle will be saved by FAS.
@ .
@CYCLE *JOBNUM.,5
@FREE *JOBNUM.
@CAT *JOBNUM(+1).
@COPY *JOBNUM(-1),*JOBNUM(-0).
```

```
@USE      JOBNUM,*JOBNUM.
@ASG,A    JOBNUM.
@COPY,A   *LPD.LPD2200,TPF$.      . Execute LPD from TPF$.
@FREE     *LPD.
@TEST TE/0          . 0 = NO OPTION
@JUMP TEST1
@.LPD2200
@JUMP DONE
@TEST1:
@TEST TE/1          . 1 = D-OPTION
@JUMP TEST2
@.LPD2200,D
@JUMP DONE
@TEST2:
@TEST TE/2          . 2 = H-OPTION
@JUMP TEST3
@.LPD2200,H
@JUMP DONE
@TEST3:
@TEST TE/3          . 3 = L-OPTION

@JUMP TEST4
@.LPD2200,L
@JUMP DONE
@TEST4:
@TEST TE/4          . 1 = Q-OPTION
@JUMP TEST5
@.LPD2200,Q
@JUMP DONE
@TEST5:
@MSG ***** INVALID SETC VALUE ON 'ST LPDTST' *****
@DONE:
@BRKPT PRINT$
```

---

## 3.1 Overview

---

The LPR/LPD processor is executed from a batch run-stream. The LPR program reads print files from LPR/LPD input queues, passing them concurrently to the appropriate remote hosts. The LPD program can receive files from remote hosts and either enqueue them to the appropriate printers or reroute them to other remote hosts. LPD supports both Unix type lpr/lpd and IBM's MVS Download protocols.

When an SDF file is processed it is converted from the native Systems Data Format to ASCII. Formatting is controlled both by Unisys print controls contained within the file and by the specific formatting associated with each LPD queue in the PARAM. file.

In addition to PARAM file configurations (covered in chapter 2) processor execution is directed by various startup options and operator keyins.

---

## 3.2 Starting the LPD processor

---

The suggested LPD processor batch run-stream is shown in the installation chapter, and should be placed in SYSSLIB\$\*RUN\$ to ensure that it is started with the intended account number and userid. It can then be started with one of the following keyins depending on the desired level of debug messages.

ST LPD - results in standard processor call of @LPD2200. The debug level will be determined when the PARAM file OPTIONS parameter is read. For the options below, the debug level will be known immediately and will override the level given in the PARAM file. **For the options below, log messages will start being generated as soon as the option is read from the run-stream.**

ST LPD,1 - results in processor call of @LPD2200,D which is highest debug level.

ST LPD,2 - results in processor call of @LPD2200,H

ST LPD,3 - results in processor call of @LPD2200,L which is lowest debug level.

ST LPD,4 - results in processor call of @LPD2200,Q which locks all queues on start-up. This option can be overridden on a queue by queue basis by setting INIT = UP in the PARAM file QUEUE statement.

It is important that the run-stream contain the correct @QUAL statement for the qualifier used for the LPD PARAM and JOBNUM files, and that these files are not assigned by anyone else during LPD's initialization phase. Other files which must be @FREE'd before running LPD are the diagnostic files \*LPD-PRINT. and \*LPDLOG.

If the LPD PARAM file configuration is changed, it is necessary to terminate and restart LPD for the changes to take effect.

### 3.2.1 LPD processor keyins

These console keyins are used by the operator or system administrator to control the operation of the LPD Processor.

They are issued by the console command:

LPD\* <command> where LPD\* is the console keyin name specified in the LPD PARAM file. If a different keyin name was selected during the installation process, make the appropriate substitution.

**example:** LPD2R3\* TERM or LPD2R3\* /TERM

Use of the "/" character preceding commands is optional. The use of the "/" character was implemented to make LPD operator commands consistent with the previously existing SPIN-X Central product.

Group operations: Group operations can be performed on named groups of queues (see QUEUE command) or the "default" group (queues which do not have an explicitly named group will belong to the default group). This is accomplished by appending "/G" to the appropriate processor keyin.

**examples:**

LPD\* STAT/G 3HOLE - Show status of queues in 3HOLE group

LPD\* STAT/G - Show status of queues in "Default" group

LPD\* STAT - Show status of all queues

LPD\* STAT Q1 - Show the status of the queue "Q1"

There is one STATUS keyin which displays the status of only the queues which are active (as opposed to locked). This is:

LPD\* STAT UP - Show the status of the "up" queues only

Since this is the same syntax used to display the status of a queue, the result of a STATUS keyin might be unexpected if status of a queue named "UP" were to be displayed. To avoid this confusion it would be better not to use this name for a queue.

Other processor keyins: The keyins INIT, LOCK and STAT can be applied to "groups" by appending "/G" as is demonstrated by the examples above.

- **TERM** Terminates the LPD run after all active print files have been completed.
- **KILL** Immediately terminates the LPD run. Any active print jobs are terminated and re-queued.
- **ID** Identifies LPD version level
- **INIT** (or /I) Initiates (unlocks) all queues which have been locked.
- **INIT *queue*** Initiates (unlocks) a queue *queue* which has been locked.
- **STAT** Prints the number of active and idle tasks for the outbound or client side. Also prints the number of inbound connections that are open and being served by LPD. The status of queues (PRINTING, WAITING, or LOCKED) and gives the HOST, PRINTER and FORMAT for each queue.

- **LD** [*value*]     where LogDetail can be substituted for LD and *value* is Low (or Lo), High (or Hi), or Debug (or D). This key in sets and displays the level of log detail, overriding the level set by either the PARAM file or the start options shown above. For example:
  1. LPD\* LD - displays current level of log detail.
  2. LPD\* LD LO - entries to the log file are kept to a minimum.
  3. LPD\* LD HI - setting the level to high causes more entries to be written to the log file.
  4. LPD\* LD D - Debug causes every detail to be logged. Setting the level to Debug will slow processing.
- **LOCK**     (or /L) Locks all queues. Files which are still in the process of printing are allowed to finish
- **LOCK** *queue* Locks the individual queue *queue* .

---

## 4

# Printer Installation and Set-up

This section first describes the configurations necessary to run the various printer types with SPIN-X LPD. After that, the formatting of banner pages, report bodies and trailer pages will be covered.

---

### 4.1 Connecting to a printer

---

SPIN-X LPR/LPD supports a variety of printers and printer connections, including locally attached parallel and serial printers, LAN-attached printers and network attached printers. The only requirement is an LPD and an IP address. Channel-attached printers require a DataWare board in a SPIN-X Xpress PC.

---

#### 4.1.1 Locally attached and LAN-attached printers

---

SPIN-X LPD can print to locally attached (attached to a PC) centronics parallel printers and serial printers if they are "shared" resources on the NT or Novell network. LAN attached printers, while having their own IP addresses, can also be configured to SPIN-X/LPD in the same way as locally attached printers.

The lpd daemon which drives the printers in the situations above will reside on the Novell, Unix or NT server. It is the IP address of this server which must be configured in the **Host** statement of the SPIN-X/LPD PARAM file. The printer name which this server knows must also be configured in the **Queue** statement. The printer name is case sensitive, so it must match exactly in the PARAM file and on the server.

---

#### 4.1.2 Network attached printers

---

HP printers with a Jet Direct card are IP nodes with their own LPD daemons. The HP daemon conforms exactly with the standard LPD protocol. Very good results can be obtained if the Unisys host is configured with the address of a Jet Direct card. Network attached printers such as the Xerox 6135 are also TCP-IP printers in their own right. The lpd daemon actually resides on the 6135's Sun Sparcstation.

Again, in the case of a network attached printer, the SPIN-X/LPD PARAM file would contain the IP address of the printer itself.

---

#### 4.1.3 Channel-attached printers

---

Printers that require a bus and tag connection are attached through Data/Ware 2000. To connect channel-attached printers, install the board on your PC. Install the Data/Ware device driver CHANNEL.SYS on the PC in the "config.sys" file. See Appendix D of the Xpress manual for details.

Data/Ware supports channel-attached printers that use IBM 3211 protocol; for example, Xerox 4075, Xerox 4050 and StorageTek 5000 line printers. For details on installation and printers supported, see the *Data/Ware 2000 Installation Guide*. (See individual printer manuals for installation instructions and details on set-up.)

Xpress PC is a powerful print spooler that can drive a variety of printers and devices. LPR/LPD is just one of many ways to connect to Xpress. This SPIN-X software is required to drive the Data/Ware card.

With channel printers, formatting of reports is carried out locally. The standard lpd protocol does not provide for sending format names to a printer so a special method has been developed for giving the format name to the Xpress PC. This is discussed in chapter 2 under "Connecting to the Xpress PC".

---

## 4.2 Contents of the LPD [*qualifier*]\*SETUP file

---

During installation of SPIN-X/LPD a \*SETUP file will be created under the LPD qualifier. This file contains sample PCL and XGF setup elements, referenced by the LPD PARAM file for LPD print file processing. They contain banner pages and page formatting commands.

PCL omnibus (ftp binary) elements:

BANNER/PCL - Banner page with keyword replacement for userid, etc.  
 PS80/PCL - Set Portrait, simplex, 80 x 66  
 PD80/PCL - Set Portrait, duplex, 80 x 66  
 PS80W/PCL - Set Portrait, simplex, 80 x 66, line wrap  
 PD80W/PCL - Set Portrait, duplex, 80 x 66, line wrap  
 LS132/PCL - Set Landscape, simplex, 132 x 66  
 LD132/PCL - Set Landscape, duplex, 132 x 66  
 LS132G/PCL - Set Landscape, simplex, 132 x 66, gray-bar  
 LD132G/PCL - Set Landscape, duplex, 132 x 66, gray-bar  
 PD132/PCL - Set Portrait, duplex, 132 x 88

XGF symbolic elements:

BANNER/XGF - Banner page with keyword replacement for userid, etc.  
 GRAYBR/XGF - Set Landscape, duplex, 132 x 66, gray-bar (zebra) form  
 LD132/XGF - Set Landscape, duplex, 132 x 66  
 LS132/XGF - Set Landscape, simplex, 132 x 66  
 PD80/XGF - Set Portrait, duplex, 80 x 66  
 PS80/XGF - Set Portrait, simplex, 80 x 66  
 PS80A/XGF - Set Portrait, simplex, 80 x 66, ANSI text file

HTML symbolic elements:

BANNER/HTML - Preamble to a print file to make it readable by a Web browser.  
 TRAILER/HTML - Required ending to an HTML document.

**Note:** Symbolic elements are created when PC files are simply ftp'd to the host. The use of Makerpt is not required.

This file also contains:

The MAKERPT/EXE omnibus element which can be ftp'd (binary) to a PC directory. This DOS program is used to create the /PCL elements for sites that cannot find what they need in the list above. See the 'Make Report' documentation in the Makerpt appendix of this manual.

MAKERPT2200 is a version of Makerpt which runs on the Unisys 2200. Most of the features of the DOS version have been implemented in this version. .

BANNER/DAT and PCL/DAT are the source elements used by Makerpt to create the PCL elements.

LPS/DAT contains DJDE type setups for all the standard SPIN-X formats. A transform engine (i.e. Solimar) must be used to translate the DJDE setups and associated JDLS, PDLs, and CMEs into Postscript or PCL.

PARAM/GSU - A copy of the PARAM file currently being used at Georgia State University (with actual host addresses removed).

---

## 4.3 Report Setup and MakeReport

---

Formatting of reports from SPIN-X Lpr is controlled by the LPD\*PARAM file. This file contains a number of keyword oriented commands which must appear in a specific order. When LPD2200 reads the PARAM file it allows no "forward references", that is, a thing must be defined before it is used in subsequent statements. The QUEUE statement must appear near the end of the file because it will reference a FORMAT statement which occurs before it. Likewise, the Format statement must not come before MACRO statements referenced by the FORMAT statement.

Simple line printer formatting can be done with FORMAT fields such as LINES, WIDTH, TOP\_MARGIN and BOTTOM\_MARGIN. Further printer specific formatting will be accomplished by the insertion of Setup elements into the print data. Samples of these elements are found in the file LPD\*SETUP. Setup elements can be either fixed text or binary sequences or binary sequences with forms containing a variety of keywords that are replaced dynamically at runtime. The MACRO command is used to identify the setup element and set certain macro processing properties. Then the FORMAT statement will list the MACROs it needs in it's Banner\_setup, Report\_setup and Trailer\_setup fields. Finally a QUEUE statement will reference the FORMAT statement. Any files @SYMmed to a SPIN-X LPR/LPD queue will be given the format associated with the queue in the PARAM file.

A minimal MACRO for a banner might simply reference an OS2200 ascii text file (type SYMBOLIC) that could be created in a text editor. A little more control could be obtained by specifying Keyword Replacement within the MACRO statement. In this way the userID, hostname, accountID or any of the other keywords in the ascii text setup file pointed to by the macro could be dynamically replaced by their actual values at print time.

A minimal macro for a report might reference a PCL binary escape sequence which would be appended to the front of the print file so as to tell an HP printer what printer setup to use. There are many other printers besides the HP which require binary sequences for report setup. Creating these binary sequences can be accomplished with a text editor with the use of a special SPIN-X utility.

On the release tape there is a processor called Makerpt which converts tokenized text files into binary files. Elements in LPD\*SETUP which end in "/DAT" are input files to Makerpt. Elements ending in "/PCL" are output from the PCL/DAT element after processing from Makerpt. If your site only needs one of the already created report formats or banners which are included on the release tape then just point the Setup macro at the LPD\*SETUP file element that is needed - MakeReport is not required. However to take advantage of all the report formatting capabilities available with SPIN-X, the user will need to become acquainted with MakeReport. For custom banners and report formats, consult the Makerpt appendix of this manual (PC version) and the LPD Processor Utilities appendix (Unisys mainframe version).

Both Makerpt versions are included on the LPD release tape: The PC version has a little extra functionality which could save some time on large projects. The output files would need to be ftp'd from the PC to the 2200 as binary, with TAS configured not to prepend the Unisys file header. Makerpt2200 works similarly to the PC version except that procedures have not been implemented. It is documented in the SPIN-X Utilities appendix.

The LPD release tape contains already created elements only for XGF and PCL printers - there are many more already created samples which Makerpt can compile which are available upon request. Many of these are mentioned in the Makerpt appendix.

When customization is required, then the /dat element will have to be modified. Each Sample.dat will contain @name...@end blocks which will cause to be created the various elements corresponding to the @name.

The discussion below will demonstrate how to identify and use these files. A partial Sample.dat file for HP printers (one format only) is shown as an example for the discussion that follows.

### **Report Setup file for compilation by Makerpt or Makerpt2200**

```

; ld132b.dat
; Execute: makerpt ld132b.dat -b -c -n ;
; (will create ld132b.pcl - this filename is indicated by the @ symbol)
; Generate PCL sequence for:
; landscape duplex 132x66 with SEMI_BOLD weight text.
;
;
@ld132b.pcl      ; < Landscape, Duplex, 66 lines/page, 132 columns >
                ; < with 7/16 top margin on front sheet and 7/16  >
                ; < bottom margin on back sheet.                >
                ; < This margin is appropriate for long edge    >

ESC "E"         ; PCL Reset
ESC "&I1S"      ; PCL Duplex Print, long edge binding
ESC "&I1O"      ; PCL Logical Page Orientation (Landscape)

                ; The following command shifts a landscape long-edge
                ; bound logical page left 1/720 of an inch on the
                ; front side of a sheet and right 1/720 of an inch on the
                ; back side of a sheet.

ESC "&I0Z"      ; Top Offset Registration Command: (+/- 1/720 inch)

                ; The following command shifts a landscape long-edge
                ; bound logical page down 90/720 of an inch on the
                ; front side of a sheet and up 90/720 of an inch on
                ; the back side of a sheet.

ESC "&I90U"     ; Left Offset Registration Command: (+/- 1/720 inch)

                ; The following set of commands set the default
                ; font characteristics for the page. These
                ; values determine the specific size of the bit-mapped
                ; font used. The default VMI and HMI characteristics
                ; for this font will be modified in a later set of
                ; commands.

ESC "(10U"      ; PCL Primary Symbol Set = PC-8
ESC "(s0P"      ; PCL Primary Spacing (fixed pitch)
ESC "(s12H"     ; PCL Primary Pitch (CPI)
ESC "(s1B"      ; PCL Primary stroke weight = semi-bold

                ; The following commands allow us to get 132+
                ; characters within each line. If we desire a
                ; wider left hand margin we must increase the
                ; left margin value and reduce the HMI value
                ; appropriately.

ESC "&k9.65H"    ; PCL Horizontal Motion Index (HMI) value (in 1/120ths)
ESC "&a0L"       ; PCL Left Margin: # (in columns)

```

; The following commands allow us to get 66 lines  
 ; per page with a 7/16 inch top margin. If we  
 ; desire a taller top margin then we must increase  
 ; the top margin value and reduce the VMI value  
 ; appropriately. If we desire to longer bottom  
 ; margin then reduce the VMI value appropriately.

; NOTE: To establish a logical page that is  
 ; appropriate for front/back shifting we must  
 ; do two sets VMI/TopMargin commands to center  
 ; the logical page correctly.

ESC "&l18C" ; PCL Vertical Motion index value (in 1/48ths)  
 ESC "&l1E" ; PCL Top Margin = 3 lines down from top  
 ESC "&l5.7C" ; PCL Vertical Motion index value (in 1/48ths)  
 ESC "&l66F" ; PCL Text Length = 66 lines

; The following command insures that a single  
 ; ASCII form-feed character will result in the  
 ; HP restoring the logical cursor to the left  
 ; hand margin after the page eject.

ESC "&k2G" ; PCL Line Termination: FF-CR/FF, LF-CR/LF  
 @end

### 4.3.1 MACRO Commands To Control Formatting

Most editors default to good EOL behavior, usually a carriage return and line feed at the end of each line. If this is not appropriate, use the CARRIAGE\_CONTROL = option to correct the problem. You may for instance want to select LF for a type SYMBOLIC setup element that contains a DJDE.

You can select from the following CARRIAGE\_CONTROL = options to control which character(s) will follow each image in a setup file.

LF Only Line Feeds are sent to the printer.

CRLF Carriage Returns and Line Feeds are sent to the printer

ANSI A space character is sent.

NONE

#### 4.3.1.1 Variable Commands for Banner and Trailer Pages

You can define banner and trailer pages as required by your location. A specific type of printer can have a special banner for a special format. By letting KEYWORD = Y in the MACRO statement the user can place various runtime variables on the banner.

##### Mainframe-defined system variables

Valid variables passed from the host are defined in the MACRO statement of the LPD PARAM file - for instance **USERID**, **ACCTID**, **RUNID**, **FILENAME**, **DEVICE**, **FORMAT**, **PRTDATE**, **PRTTIME**, **DESC1 . . . DESC4**, and **ADDR1 . . . ADDR6** are some of the variables (see MACRO statement) which can be displayed on banners and trailers. This is accomplished by embedding the variables within the banner or trailer setup file as delimited keywords (the variable is enclosed by "<" and ">" symbols). When SPIN-X LPR/LPD parses a KEYWORD mode setup file it will look for delimited keywords. If a delimited keyword is found the variable's current value will be plotted onto the banner page. Any text preceding or following a delimited keyord is preserved during the plotting.

### 4.3.1.2 Banner Setup File Example

#### Example : Laser printer using Keyword option

First the banner template must be edited to include delimited keywords.

Delimited keywords are the various MACRO variables having a "<" character at the beginning and a ">" character at the end. For example to indicate that the "USERID" associated with a print job should be plotted one would insert the delimited keyword <USERID> into the "BANNER/PCL" file.

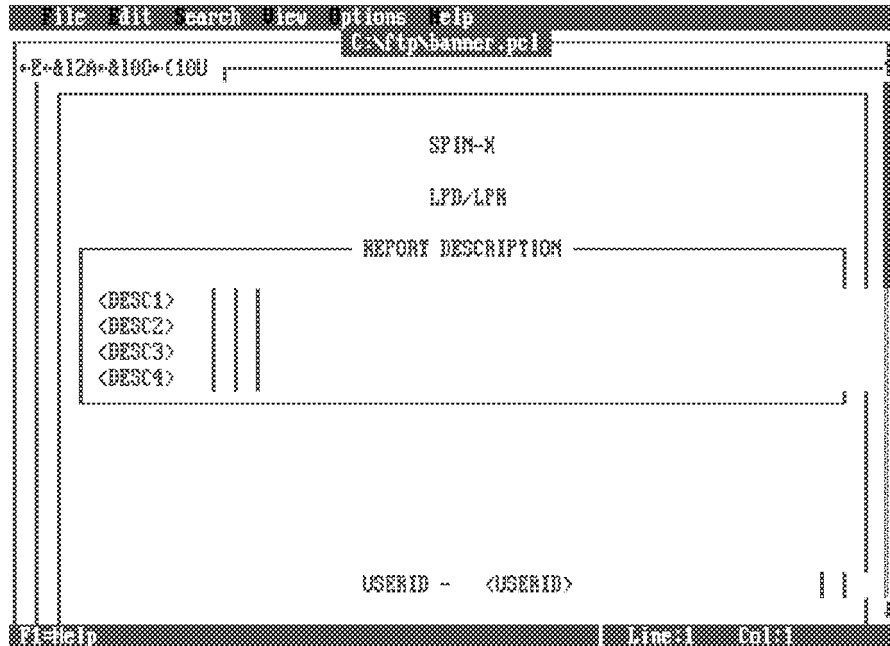
It is possible to plot the same variable more than once or to plot more than one variable on a given line. For example:

```
"USERID - <USERID> RUNID - <RUNID> ACCOUNT - <ACCTID>"
```

Every keyword variable has a default length associated with it. For instance the reference guide indicates that the USERID field is 12 characters. When SPIN-X LPR/LPD parses the "BANNER/PCL" Setup file it will remove the 8 bytes of text that represent the delimited keyword with replace it with 12 bytes of data representing the userid and trailing whitespace. The lines on the template which contain keywords will need to be space filled to the desired length so that the keywords fit properly.

Since keywords are of various lengths, it is necessary to manually adjust the trailing whitespace after each delimited keyword so that the right margin line can be preserved visually when the "BANNER/PCL" file is printed. This is depicted below. The Setup element BANNER/PCL is shown as it appears in MS-DOS Edit. Except for the 1st line, all the extended ASCII border characters will line up after SPIN-X LPR/LPD processes and sends the file. Since the binary escape sequences are not printed, the first line will line up when it finally reaches the printer.

#### Example :



---

### 4.3.2 Preformatted Setup Files

---

For your convenience, several pre-formatted Setup files for both formats and banners were installed during the SPIN-X LPR/LPD installation process. These were mostly created using Makerpt. These files are described previously in this chapter in the section "Contents of the LPD\*SETUP File". The devices supported are HP and XGF printers.

Makerpt was originally designed for a SPIN-X print spooler called Xpress which became the model for the first DEPCON. Xpress supports many other printer and device types. Many of the Setup files output by Makerpt PC for Xpress can be used without modification on the Unisys 2200. The Makerpt PC appendix mentions some of the devices which are supported, and for which there exist .DAT files that Makerpt can compile. Each of these include a pre-formatted banner, several report formats, and a trailer page.

### 4.3.3 Example Makerpt Execution

---

An example of using PC Makerpt in the HP subdirectory follows. The -D:HP is a shortcut from \UTILITY to \UTILITY\DEVTYPE\HP. The -C instructs Makerpt to compile NEWPCL.DAT. This file might be worth reading before you compile it.

```
MAKERPT -D:HP -C NEWPCL.DAT
```

After the compiling of NEWPCL.DAT the HP subdirectory will contain the .pcl files (report formats) for the HP device type.

A binary FTP (where TAS on the 2200 side does **not** use the -b option) of the output to the 2200 would produce the (format C-packed) omnibus elements required for SPIN-X setup elements.

Alternatively, one of the tokenized text files for Makerpt PC could be modified in a text editor and sent (text mode) to the 2200 where Makerpt2200 could compile it. The appendix describing SPIN-X Utilities has the details, but an example of Makerpt2200 execution might be:

```
@LPD*UTILITY.MAKERT2200 LPD*SETUP.BANNER/DAT,LPD*SETUP.
```

---

## 4.4 Using LPR to Access Xpress PC Functionality

---

The LPR protocol (RFC 1179) does not specifically provide for some of the control information which Xpress normally expects to get in an Xpress \$4 control record from the Remote Print Facility on the host. The format of the print especially is not automatically sent. All other control information like Hostname, Filename, Userid, Runid, Device (or Printer) etc. is passed to the remote LPD server. In order for Xpress to do the formatting properly on the PC side, it needs to get the format name from the host side, along with the print file. Since most Xpress machines are already set up with the right formats for the jobs and even custom formats, it is important that Xpress be able to receive the right format names.

The Xpress ASCII input device has a special provision for passing control information, i.e. format names. This method passes information by means of a "signature string". This string has the form:

```
*XPR-DIR-WATCH* control info
```

and as many of these statements as required can sit on top of the data file, each on a line by itself. The signature string and whatever follows is understood by Xpress as control info and is not printed.

The ASCII input device is described in great detail in the Xpress manual along with the other communications options.

No matter what kind of system is sending the LPR request, some provision should probably be made to access the Xpress format with the above method. If the LPR is from SPIN-X LPR/LPD, the following method for sending formats should be used.

---

### 4.4.1 Passing The Format-Name to Xpress with SPIN-X LPR/LPD

---

Now, if we can somehow pass to Xpress the following string, which includes the format-name of the document, we are in good shape:

```
*XPR-DIR-WATCH* FORMAT <format name>
```

...where format-name is the name of the format i.e. PS80 or LS132 etc. The above signature string needs to be placed at the top of the data file, where LPD2XPR has placed the other signature strings with their own control data. We can use the MACRO and BANNER\_SETUP features of SPINX LPR/LPD to easily accomplish this in an automatic fashion, as follows:

1. Create a symbolic (text) element in the LPD setup file (e.g. LPD\*SETUP.XPR/TXT) with just one line of text, exactly as follows:

```
*XPR-DIR-WATCH* FORMAT <FORMAT>
```

2. Insert the following MACRO statement in the LPD Param file:

```
MACRO NAME = XPR_FORMAT,
      FILENAME = LPD*SETUP.XPR/TXT,
      TYPE = SYMBOLIC,
      KEYWORD = YES,
      CARRIAGE_CONTROL = CRLF
```

3. Declare **each** format-name which is being used by Xpress, in the LPD Param file and use the macro created in step 2 as the BANNER\_SETUP macro. For example:

```
FORMAT      NAME = LS132,  
            LINES = 0,  
            BANNER_SETUP = XPR_FORMAT,  
            REPORT_SETUP = $NONE$
```

```
FORMAT      NAME = PS80,  
            LINES = 0,  
            BANNER_SETUP = XPR_FORMAT,  
            REPORT_SETUP = $NONE$
```

In the above example, because of the KEYWORD=YES statement in the XPR\_FORMAT macro, when SPIN-X LPR/LPD sends a file using one of these formats the actual format-name LS132 or PD80 will be substituted for the <FORMAT> keyword in the XPR\_FORMAT macro.

4. Use the above-declared formats in the QUEUE statements. The HOST, which is elsewhere defined in the PARAM file would be the PC running XPRLPD. The PRINTER name would be the same as the one defined in the Xpress XPConfig program. Please note the printer name is case sensitive, so use exactly the same name that is configured in Xpress. For example:

```
QUEUE       NAME = LPDQ1,  
            HOST = XPR_PC,  
            PRINTER = HPLJ,  
            FORMAT = LS132
```

When you SYM a file to LPDQ1, SPIN-X LPR/LPD puts the format-name needed by Xpress at the top of the data file and then LPD2XPR automatically adds the rest of the control information that is included with a standard LPR request.

---

## 5 Installation Verification & Troubleshooting

This section explains how to deal with common problems which may arise while installing LPD. Some useful utility programs are also discussed. If the answer to a specific question is not provided here, feel free to call Customer Support.

---

### 5.1 Installation Verification for LPD Files

---

1. Start the LPD processor as described earlier. From @@CONS mode verify from the console output that the run starts without error.

If the run fails, it's print\$ file (LPD-PRINT) and/or log file (LPD\*LPDLOG.) will indicate the reason for the failure.

2. @SYM a file to a print queue configured as an LPD input queue.

If the file is printed on the remote host printer, the LPD configuration is correct.

If the file does not print on the remote host, analyze the print queues and log files to see where the problem is: some possible actions follow.

1. Do an '@@CONS T D' to ensure that the LPD run is still active.
2. Do an '@@CONS SQ [input-queue] \*' on the queue to which the file was @SYM'd.

a.) If the file is still on the queue:

If the LPD's runid is not appended to the file listing, the queue name is not configured properly in the PARAM file.

If the LPD's runid is appended to the file listing, the file is 'in-progress'. Some internal problem has occurred. Terminate the LPD processor and analyze the print and log files. Run CLEANQ on this input queue. The file can not be printed until CLEANQ has been run.

b.) If the file is no longer on the input queue:

Do an '@@CONS SQ [hold-queue] \*' to see if the file is on the hold queue defined by the PARAM file. If it is, there could be a problem with the file's format. Alternately, the file could be empty or the LPR could have been rejected by the remote host because of an unknown printer name. Analyze the LPD print and log files to determine the cause of the problem.

c.) If the file is not on the LPD hold queue, analyze the LPD print and log files to see if the file was sent to the remote host. If it was, the problem is on the remote host.

If the above steps do not indicate the reason for the failure, back track to find at what point the file was stalled or lost.

---

# A

# SPIN-X/LPD Error Messages

For all error messages, any I/O error codes or ACSF\$ facility status codes are described in the *UNISYS Executive Requests Reference* manual. Basic Service Package (BSP) and SDFIO error codes are described in the *UNISYS SYSLIB Reference* Manual.

All error messages described as 'internal error' indicate that something in LPD processor or operating system has become corrupted.

---

## A.1 LPD Processor Error Messages

---

- ERROR 01:**    **Cannot open Log file [log file name]**
- An attempt to open the log file in the "Write" mode failed. This error appears in the PRINT\$ file. If the log file has been correctly catalogued and assigned as shown in the LPD/RUN run stream, this error should never occur.
- ERROR 02:**    **Cannot open Parameter file [parameter file name]**
- An attempt to open the parameter file in the "Read" mode failed. This error appears in the PRINT\$ file. The parameter file must be assigned with a USE name of PARAM, as shown in the LPD/RUN run stream.
- ERROR 05:**    **Filename [file name] too long. Limit=[file name length] chars**
- ERROR 07:**    **Host name [Host Name] too long. Limit = [number of characters] chars**
- ERROR 10:**    **Input Q name [queue name] too long. Limit =[number of characters] chars**
- The input queue name is longer than maximum allowable by OS1100, in the specified line of parameter file.
- ERROR 12:**    **Hold queue name [queue name] too long. LIMIT = [number of characters] chars**
- The hold queue name is longer than maximum allowable by OS1100, on the HOLDQUEUE statement of parameter file.
- ERROR 13:**    **Banner mask [banner mask string] too long. Limit =[number of characters] chars.**
- ERROR 15:**    **Console keyin [keyword] too long. LIMIT = [number of characters] chars**
- The length of console keyword on the CONSOLEKEY statement is more than the maximum allowed.
- ERROR 17:**    **Process name [process name] too long. LIMIT = [number of characters] chars**
- The length of process name on the specified PROCESSNAME statement is more than the maximum allowed.
- ERROR 18:**    **Password for process name [process name] exceeds [max password length] chars in line [line number]**
- The length of password on the specified PROCESSNAME statement is more than the maximum allowed.
- ERROR 19:**    **Unable to assign input file.**
- ERROR 24:**    **Host alias [alias name] too long. Limit = [number of characters] chars**

**ERROR 36: TSAM status [status code] while trying to Attach**

An attempt to attach to a PROCESS in CMS1100 failed. The TSAM status code gives specific information about the reasons for the failure. The description for the TSAM status codes could be found in the CMS1100 Programming reference manual (Manual # 7831 5827-002), Table 5-13

**ERROR 37: Attach failed because all TSU names are in use**

All TSU names (or process-ids on the PROCESS statement in CMS1100 configuration) valid for the current TSU ( which in this case is SPINX LPR/LPD) are in use, so the attempt to attach failed.

**ERROR 39: Cannot convert host address [dotted IP address] into 4 byte format**

An attempt to convert the specified host address into an internal 4-byted format failed. The reason could be an invalid number (for example more than 255) in the address.

**ERROR 40: Host [domain name/alias] not declared in the Parameter file**

The remote host name/alias specified on an INPUTQUEUE statement in the parameter file is not declared on an ADDRESS or HOSTNAME statement.

**ERROR 41: TSAM status [status code] while trying to do Active-Open**

An attempt to establish an active connection with a host failed. The TSAM status code gives specific information about the reasons for the failure. The description for the TSAM status codes could be found in the CMS1100 Programming reference manual (Manual # 7831 5827-002), Table 5-13

**ERROR 42: TSAM status [status code] while trying to Detach**

An attempt to detach from a PROCESS in CMS1100 failed. The TSAM status code gives specific information about the reasons for the failure. The description for the TSAM status codes could be found in the CMS1100 Programming reference manual (Manual # 7831 5827-002), Table 5-13

**ERROR 43: TSAM status [status code] while trying to Send data**

An attempt to send data to a remote host failed. The TSAM status code gives specific information about the reasons for the failure. The description for the TSAM status codes could be found in the CMS1100 Programming reference manual (Manual # 7831 5827-002), Table 5-13

**ERROR 44: Error indication received from TCP**

A fatal error indication received from TCP which would not allow the communication to go on successfully. More specific information could be gathered by looking at the error reason and error subreason which are listed below:

ERROR reason: Too many transmissions

ERROR reason: Destination unreachable

ERROR subreason: Network unreachable

ERROR subreason: Host unreachable

ERROR subreason: Protocol unreachable

ERROR subreason: Port unreachable

ERROR subreason: Fragmentation needed and don't fragment flag(DF) set

ERROR subreason: Source route failed

ERROR subreason: Destination network unknown

ERROR subreason: Destination host unknown

ERROR subreason: Source host isolated

ERROR subreason: Communication with destination network administratively prohibited.

ERROR subreason: Communication with destination host administratively prohibited.

ERROR subreason: Network unreachable for type of service  
 ERROR subreason: Host unreachable for type of service

ERROR reason: Source Quench

ERROR reason: Time exceeded  
 ERROR subreason: Time to live count exceeded  
 ERROR subreason: Fragment reassembly time exceeded

ERROR reason: Parameter problem

**ERROR 45: Active open rejected**

An attempt to establish an active connection with a remote host failed. The reason could be one of the following:

Request rejected by the remote host.  
 maximum transmissions on active open request.  
 destination unreachable.

**ERROR 46: TSAM status [status code] while trying to Abort**

An attempt to abort a connection failed. The TSAM status code gives specific information about the reasons for the failure. The description for the TSAM status codes could be found in the CMS1100 Programming reference manual (Manual # 7831 5827-002), Table 5-13

**ERROR 47: CMS 1100 not available**

One can get this status at any point of time. This could mean that CMS1100 has not yet initialized, or CMS1100 went down and has not yet recovered.

**ERROR 48: Received TSAM status [status code]**

A TSAM error status was received. The TSAM status code gives specific information about the reasons for the failure. The description for the TSAM status codes could be found in the CMS1100 Programming reference manual (Manual # 7831 5827-002), Table 5-13

**ERROR 49: TSAM status [status code] while trying to Close**

An attempt to close a connection failed. The TSAM status code gives specific information about the reasons for the failure. The description for the TSAM status codes could be found in the CMS1100 Programming reference manual (Manual # 7831 5827-002), Table 5-13

**ERROR 50: Cannot register console keyin [keyin name], status = [KEYIN\$ status code]**

An attempt to register the specified keyin failed. The description for the KEYIN\$ status codes could be found in the Executive Requests Programming reference manual (manual # 7830 7899-002), Table 21-1

**ERROR 51: ER KEYIN\$ status [KEYIN\$ status code] for keyin [keyin name]**

An error status code was received while doing an ER KEYIN\$. The description for the KEYIN\$ status codes could be found in the Executive Requests Programming reference manual (manual # 7830 7899-002), Table 21-1

**ERROR 52: Cannot deregister console keyin [keyin name] status = [KEYIN\$ status code]**

An attempt to deregister the specified keyin failed. The description for the KEYIN\$ status codes could be found in the Executive Requests Programming reference manual (manual # 7830 7899-002), Table 21-1

**ERROR 54: Cannot allocate memory for Receive-activity-handle**

An attempt to allocate memory at run-time for the Receive-activity failed. This is a fatal error which would not allow LPD to function properly and may cause it to abort.

- ERROR 55: Cannot allocate memory for handle for Queue [queue name]**
- An attempt to allocate memory at run-time for the specified queue-activity failed. This is a fatal error which would not allow the current queue activity to function properly and would cause it to abort.
- ERROR 56: TSAM status [status code] while getting local internet addr**
- An attempt to determine the local internet address failed. The TSAM status code gives specific information about the reasons for the failure. The description for the TSAM status codes could be found in the CMS1100 Programming reference manual (Manual # 7831 5827-002), Table 5-13
- ERROR 58: Cannot open job-number file [file name] for reading**
- An attempt to open the job-number file in the "read" mode failed.
- ERROR 59: Cannot open job-number file [file name] for writing**
- An attempt to open the job-number file in the "write" mode failed.
- ERROR 60: LPR rejected by host [domain name/alias] for queue [queue name]**
- An LPR request (Receive print job request) was rejected by the remote host corresponding to the specified queue. There could be many reasons for that. For example, our local host may not be allowed to send print jobs to the remote host or the destination printer/device name on the LPR request may not be configured on the remote host.
- ERROR 61: No local addresses available to satisfy Reserve-Port request**
- A zero was passed for the local internet address. No addresses were available to CMS1100 that would satisfy the request.
- ERROR 62: No local ports available to satisfy Reserve-Port request"**
- A zero was passed for the local port number. No ports were available to CMS1100 that would satisfy the request.
- ERROR 63: TSAM status [status code] while trying to reserve port [port number]**
- An attempt to reserve the specified local port failed. The TSAM status code gives specific information about the reasons for the failure. The description for the TSAM status codes could be found in the CMS1100 Programming reference manual (Manual # 7831 5827-002), Table 5-13
- ERROR 64: Cannot free local port [port number] because it was not reserved**
- An attempt to free the specified local port failed because only the ports which were reserved earlier can be freed.
- ERROR 65: Cannot free local port [port number] because port-address combination invalid**
- An attempt to free the specified local port failed because the port number and internet address combination is invalid.
- ERROR 66: TSAM status [status code] while trying to free port [port number]**
- An attempt to free the specified local port failed. The TSAM status code gives specific information about the reasons for the failure. The description for the TSAM status codes could be found in the CMS1100 Programming reference manual (Manual # 7831 5827-002), Table 5-13
- ERROR 67: Unable to acquire a local port to establish connection**
- An attempt to establish an active connection (by doing an active-open) with a remote host failed because all the valid ports (port numbers 721 thru' 731) are busy.

- ERROR 68: Cannot convert SMOQUE pkt address from EM to BM**
- An error occurred while converting the extended mode virtual address (of the form L, BDI, OFFSET) of SMOQUE packet to a basic mode virtual address (of the form E, BDI, OFFSET)
- ERROR 69: Cannot convert EM address of SMOQUE entry [entry number] to BM**
- An error occurred while converting the extended mode virtual address (of the form L, BDI, OFFSET) of SMOQUE entry to a basic mode virtual address (of the form E, BDI, OFFSET)
- ERROR 70: Cannot retrieve entry from queue [queue name], SMOQUE status: [SMOQUE status code]**
- An attempt to retrieve the specified file from the specified queue failed. The SMOQUE status code gives specific information about the reasons for the failure. The description for the SMOQUE status codes could be found in the Executive Requests Programming reference manual (Manual # 7830 7899-002), Table 13-2.
- ERROR 71: Cannot delete file [qual\*file(cycle)] from queue [queue name], SMOQUE status: [SMOQUE status code]**
- An attempt to delete the specified file from the specified queue failed. The SMOQUE status code gives specific information about the reasons for the failure. The description for the SMOQUE status codes could be found in the Executive Requests Programming reference manual (Manual # 7830 7899-002), Table 13-2.
- ERROR 72: Cannot deactivate queue [queue name], SMOQUE status [SMOQUE status code]**
- An attempt to deactivate the specified queue failed. The SMOQUE status code gives specific information about the reasons for the failure. The description for the SMOQUE status codes could be found in the Executive Requests Programming reference manual (Manual # 7830 7899-002), Table 13-2.
- ERROR 73: Cannot re-activate, SMOQUE status [SMOQUE status code]**
- An attempt to re-activate the deactivated queues failed. The SMOQUE status code gives specific information about the reasons for the failure. The description for the SMOQUE status codes could be found in the Executive Requests Programming reference manual (Manual # 7830 7899-002), Table 13-2.
- ERROR 74: Cannot requeue file [qual\*file(cycle)] to queue [queue name], SMOQUE status: [SMOQUE status code]**
- An attempt to requeue the specified file to the specified queue failed. The SMOQUE status code gives specific information about the reasons for the failure. The description for the SMOQUE status codes could be found in the Executive Requests Programming reference manual (Manual # 7830 7899-002), Table 13-2.
- ERROR 75: Cannot move file [qual\*file(cycle)] to hold-queue [queue name], SMOQUE status: [SMOQUE status code]**
- An attempt to move the specified file to the specified hold queue failed. The SMOQUE status code gives specific information about the reasons for the failure. The description for the SMOQUE status codes could be found in the Executive Requests Programming reference manual (Manual # 7830 7899-002), Table 13-2.
- ERROR 76: Cannot build I/O packet for ER IOW\$**
- An attempt to build an I/O packet for ER IOW\$ in order to read the DIDS file failed.
- ERROR 77: IOW\$ status [I/O status code] while reading file [file name]**
- An attempt to read the specified file using ER IOW\$ failed. The I/O status code gives specific information about the reasons for the failure. The description for the I/O status codes could be found in the Executive Requests Programming reference manual (Manual # 7830 7899-002), Appendix B, Table B-1

- ERROR 78: DNR not available. Cannot resolve domain name: [Domain name]**
- An attempt to resolve the specified domain name failed because the DNR (Domain Name Resolver) is not available. The reason may be that CMS1100 configuration does not contain a TCPIP-DNR statement.
- ERROR 79: TSAM status [status code] while resolving domain name: [domain name]**
- An attempt to resolve the specified domain name failed. The TSAM status code gives specific information about the reasons for the failure. The description for the TSAM status codes could be found in the CMS1100 Programming reference manual (Manual # 7831 5827-002), Table 5-13
- ERROR 100: Insufficient CMS1100 resources. Cannot resolve domain name: [Domain name]**
- The request to resolve the specified domain name cannot be completed due to insufficient CMS1100 resources.
- ERROR 101: Insufficient CMS1100 resources. Cannot resolve address: [IP address]**
- The request to resolve the specified IP address cannot be completed due to insufficient CMS1100 resources.
- ERROR 102: TSAM status [status code] while resolving address: [IP address]**
- An attempt to resolve the specified IP address failed. The TSAM status code gives specific information about the reasons for the failure. The description for the TSAM status codes could be found in the CMS1100 Programming reference manual (Manual # 7831 5827-002), Table 5-13
- ERROR 103: DNR not available. Cannot resolve address: [dotted IP address]**
- An attempt to resolve the specified address failed because the DNR (Domain Name Resolver) is not available. The reason may be that CMS1100 configuration does not contain a TCPIP-DNR statement.
- ERROR 104: Illegal address for host [domain name/alias], Q\_activity for [queue name] not initiated**
- The specified host has an illegal address, maybe because DNR could not resolve the domain name. So the queue activity corresponding to this remote host has not been started.
- ERROR 106: CMS1100 not available. LPD activities terminating...**
- One can get this status at any point of time. This could mean that CMS1100 has not yet initialized, or CMS1100 went down and has not yet recovered. Since LPD cannot function properly without the availability of CMS1100, all activities are terminated and LPD tries to recover by checking periodically whether CMS1100 has become functional.
- ERROR 107: Invalid TSU-Id. Maybe CMS went down. LPD activities terminating...**
- The current TSU-id is not known to CMS1100. The user has detached or the attaching activity has terminated, maybe because CMS1100 is not available. One can get this status at any point of time. This could mean that CMS1100 has not yet initialized, or CMS1100 went down and has not yet recovered. Since LPD cannot function properly without the availability of CMS1100, all activities are terminated and LPD tries to recover by checking periodically whether CMS1100 has become functional.

**ERROR 108: Process no longer attached. Maybe CMS went down. LPD activities terminating...**

The process was detached for one of the following reasons:

the attaching activity terminated  
 CMS1100 terminated or  
 the process was brought down with a CMS1100 DOWN command

One can get this status at any point of time. Since LPD cannot function properly without attaching to a CMS1100 process, all activities are terminated and LPD tries to recover by periodically trying to attach to a process.

**ERROR 109: Not enough space for the control file on host [remote host domain/alias]**

An indication is received from the specified remote host that it does not have enough space to receive the control file, which we are trying to send.

**ERROR 110: Not enough space for the data file on host [remote host domain/alias]**

An indication is received from the specified remote host that it does not have enough space to receive the data file, which we are trying to send.

**ERROR 111: Printer name [Printer Name] too long. Limit = [number of characters] chars****ERROR 113: TSAM status [status code] while trying to do Passive-Open**

An attempt to initiate Passive-open (Listen) failed. The TSAM status code gives specific information about the reasons for the failure. The description for the TSAM status codes could be found in the CMS1100 Programming reference manual.

**ERROR 114: Unable to acquire local port for Passive-open**

The local port was unavailable to initiate a passive-open (Listen).

**ERROR 115: No Open-indication waiting**

Open-indication waiting status was not found.

**ERROR 116: TSAM status [status code] while trying to receive open-indication**

An attempt to receive the active-open request from a peer failed. The TSAM status code gives specific information about the reasons for the failure. The description for the TSAM status codes could be found in the CMS1100 Programming reference manual.

**ERROR 117: Invalid interface level. Memory not allocated**

Invalid interface level for the SLIB memory allocation routines.

**ERROR 118: Invalid locality value. Memory not allocated**

Invalid locality value for the SLIB memory allocation routines.

**ERROR 119: Cannot allocate memory. ELMS StatusId = [ELMS statusid]**

An attempt to allocate memory using SLIB memory allocation routine failed. See SY\$LIB\$\*PROC\$.S\$STORAGE\$D/H for this ELMS statusid.

**ERROR 120: Cannot free memory block. ELMS StatusId = [ELMS statusid]**

An attempt to free memory using SLIB memory deletion routine failed. See SY\$LIB\$\*PROC\$.S\$STORAGE\$D/H for this ELMS statusid.

**ERROR 121: Destination activity not found. Receive status: [TSAM status]**

An attempt to pass the received message to its destination activity failed because the activity could not be found. The TSAM status code gives specific information about the reasons for the failure. The description for the TSAM status codes could be found in the CMS1100 Programming reference manual.

- ERROR 122: Cannot allocate memory to start a listen activity**  
An attempt to allocate memory to start another activity in order to receive an inbound request failed. Maybe system out of memory.
- ERROR 123: Received command has no LF-terminator**  
The command received from the peer has an invalid format. It does not terminate with a Line Feed character.
- ERROR 124: Unable to catalog print file for inbound LPR. FAC: [FAC status code]**  
An attempt to catalog a file, in order to create a print file for the inbound print job, has failed. Executing ECL command @FAC [FAC status code] would give more information about the reason for the failure.
- ERROR 125: Unable to process last key [keyin]**  
Unable to process last key.
- ERROR 126: Unable to assign print file for inbound LPR. FAC:[FAC status code]**  
An attempt to assign the cataloged file, in order to create print file for the inbound print job, has failed. Executing ECL command @FAC [FAC status code] would give more information about the reason for the failure.
- ERROR 127: Unable to assign USE name for print file. FAC: [FAC status code]**  
An attempt to assign a USE name for a cataloged file, in order to create print file for the inbound print job, has failed. Executing ECL command @FAC [FAC status code] would give more information about the reason for the failure.
- ERROR 128: Errors encountered in Parameter file**  
One or more errors were encountered while reading the parameter file. The LPD terminates if any errors are found in the parameter file. Fix the error and restart LPD.
- ERROR 129: (SWRITE) I/O error while writing print file**  
An I/O error occurred while SLIB routine SWRITE was writing to the print file during an inbound LPR activity.
- ERROR 130: (SWRITE) Work area overflow while writing print file**  
Work area for the SLIB routine SWRITE was overflowed while writing to the print file during an inbound LPR activity. A bigger work area is required by SWRITE.
- ERROR 131: (SWRITE) Output print file not assigned**  
An attempt by the SLIB routine SWRITE to write to the output print file failed because it was not assigned to the run.
- ERROR 132: (SWRITE) Invalid control function, while writing print file**  
Invalid value of control function was used while calling SYMBOUT routine. For valid values, see the header file SYMBOUT/H in the LPD source.
- ERROR 133: (SWRITE) Write packet not initialized, while writing print file**  
The write packet for the SLIB routine SWRITE was not initialized. OPEN Function should be called before calling the WRITE Function.
- ERROR 134: (SWRITE) Unexpected symbiont error [SWRITE error code], while writing print file**  
An unexpected error occurred while SLIB routine SWRITE was writing to the print file during an inbound LPR activity. See SYS\$LIB\$\*PROC\$.\$\$\$SWRITE\$D/H for error codes.

- ERROR 135: Timeout while waiting for next input**  
LPD activity timed out after waiting for a specified duration of time for the peer to respond.
- ERROR 136: (SYM) No file name specified on SYM image**  
An attempt to do @SYM of the print file failed because no print file name was specified.
- ERROR 137: (SYM) File name error on SYM image**  
An attempt to do @SYM of the print file failed because the print file name was not valid.
- ERROR 138: (SYM) FAC status error on attempted ASG/USE of SYMmed file**  
A FAC status error occurred when an ASG/USE of the SYMmed print file was done.
- ERROR 139: (SYM) Queue name specified on SYM image is not configured**  
The Unisys queue name to which the print file was SYMmed is not configured.
- ERROR 140: (SYM) Deletion of file that was queued by @SYM illegal**  
The file which was SYMmed cannot be deleted before it gets printed.
- ERROR 141: (SYM) SYM file type does not match configured devices for this group**  
The type of the file which was SYMmed to the specified queue is incompatible with the type of the queue
- ERROR 142: (SYM) File marked to be deleted - SYM illegal**  
The file which is marked to be deleted cannot be SYMmed.
- ERROR 143: (SYM) File was not queued - Print queue was not available**  
An attempt to SYM the print file failed because the print queue was not available.
- ERROR 144: (SYM) @SYM error status code: [error status code]**  
An attempt to SYM the print file failed. The error status code gives specific information about the reasons for the failure. The description for the error status codes could be found in the Executive Request Programming Reference Manual, @SYM status code table of ER CSF\$.
- ERROR 145: Cannot convert SYMINFO\$ pkt address from EM to BM**  
An error occurred while converting ER SYMINFO\$ packet address from extended mode virtual address of the form (L, BDI, OFFSET) to basic mode virtual address of the form (E, BDI, OFFSET). The reason for this error may be that the extended mode BDI value was greater than 4095, which is a basic mode limit.
- ERROR 146: Cannot convert buffer address from EM to BM for ER SYMINFO\$**  
An error occurred while converting buffer address for ER SYMINFO\$ from extended mode virtual address of the form (L, BDI, OFFSET) to basic mode virtual address of the form (E, BDI, OFFSET). The reason for this error may be that the extended mode BDI value was greater than 4095, which is a basic mode limit.
- ERROR 147: Cannot allocate memory for [entry type] entries**
- ERROR 149: Invalid value: [value]**  
Displays the invalid value with line number and column number, keyword and the associated field.

- ERROR 150: Cannot queue file [file name] to [queue name]**  
An attempt to queue the print file to the specified queue failed.
- ERROR 153: Cannot open temporary file [file name]**  
An attempt to open the specified temporary file failed.
- ERROR 157: Print queue [queue name] does not exist**  
The specified queue does not exist on the Unisys host.
- ERROR 160: IOW\$ status [status code] while writing file [file name]**  
An attempt to write to the specified file using ER IOW\$ failed. The status code gives specific information about the reasons for the failure. The description for the status codes could be found in the Executive Request Programming Reference Manual.
- ERROR 161: Cannot allocate memory to modify U-images**  
An attempt to allocate memory for the work area to modify and insert the U-control images at the top of the print file failed. Maybe system out of memory.
- ERROR 165: Invalid source file format. Cannot write SDF**  
The file format of the inbound source file is not supported. SDF file could not be created successfully from the source file during the inbound LPR activity.
- ERROR 166: Invalid Xerox VL format. Cannot write SDF**  
The source file does not have a valid Xerox variable length record format and the printer-form combination on the LPR request is reserved to accept a file only in the Xerox VL format. SDF file could not be created successfully from the source file during the inbound LPR activity.
- ERROR 167: Invalid Xerox VL record length. Cannot write SDF**  
Invalid record length encountered in the header of one of the records. The record length for a record in Xerox VL format cannot be less than 4. SDF file could not be created successfully from the source file during the inbound LPR activity.
- ERROR 168: Cannot allocate memory for SMOQUE entries**  
An attempt to allocate memory to receive SMOQUE entries for the LPQ request failed. Maybe system out of memory.
- ERROR 169: Invalid Prism VL format. Cannot write SDF**  
The source file does not have a valid Prism variable length record format and the printer-form combination on the LPR request is reserved to accept a file only in the Prism VL format. SDF file could not be created successfully from the source file during the inbound LPR activity.
- ERROR 171: Invalid PCC [octal PCC character] for Prism VL format. Cannot write SDF**  
Invalid PCC (carriage control character) encountered for a Prism variable length record. SDF file could not be created successfully from the source file during the inbound LPR activity.
- ERROR 172: Invalid PCC [octal PCC character] for Xerox VL format. Cannot write SDF**  
Invalid PCC (carriage control character) encountered for a Xerox variable length record. SDF file could not be created successfully from the source file during the inbound LPR activity.
- ERROR 173: Invalid PCC [octal PCC character] for FL130 format. Cannot write SDF**  
Invalid PCC (carriage control character) encountered for a Prism /Solamar fixed length record. SDF file could not be created successfully from the source file during the inbound LPR activity.

- ERROR 176:** (SDFI) Exec has returned an I/O error
- ERROR 185:** (SDFI) Unexpected error: [error code], while reading SDF file  
The code is the ELMS diagnostic message code.
- ERROR 186:** Cannot allocate memory for internal use  
NULL returned on a malloc call
- ERROR 189:** Error while reading SDF input file
- ERROR 190:** Mandatory field [keyword field] missing
- ERROR 191:** One of NAME or ADDRESS fields mandatory
- ERROR 192:** Host table full. Cannot process host [hostname]
- ERROR 193:** Invalid FILETYPE value: [filetype]
- ERROR 194:** Format name [format name ] too long. Limit=[number of characters] chars
- ERROR 195:** Invalid banner mask [banner mask string]
- ERROR 196:** [entry name] not found in [entry type] list
- ERROR 197:** [keyword] is not a valid main key
- ERROR 198:** [entry name] not found in [entry type] list
- ERROR 201:** Macro name [macro name] too long. Limit= [number or characters] chars
- ERROR 202:** (F4NI) Cannot parse input name [input name]
- ERROR 203:** (F4NI) F4niOpen() failed on %s in pass [number of passes]
- ERROR 204:** (F4NI) F4niRead() failed on %s in pass [number of passes]
- ERROR 205:** (F4NI) F4niClose() failed on %s in pass [number of passes]
- ERROR 206:** Cannot extract string  
The string specified is not in the right format. It may be a missing opening or closing quotation mark.

- ERROR 207: Mandatory value missing**  
The value for the specified field must be specified.
- ERROR 208: Macro size mismatch. Actual: [actual macro size], Calculated: [calculated macro size]**
- ERROR 209: Datafile size mismatch. Header: [size given in header], Actual: [actual size]**
- ERROR 210: Output buffer overflow during keyword substitution**
- ERROR 211: Output block size [block size] too big. Limit = [max block size]**
- ERROR 212: C-option string too long. Limit= [number of characters] chars**
- ERROR 213: [entry name] too long. Limit= [number of characters] chars**
- ERROR 214: [field name] is not a valid field**
- ERROR 215: No LPD queues configured for Send-Only mode**  
One or more Unisys 2200 queues have to be configured in the QUEUE statements of the PARAM file in order for the SEND mode to work.
- ERROR 216: Duplicate definition for [keyword, value] ]found**
- ERROR 217: Duplicate DSTMAP entry for prn [printer name], form [form name] found**
- ERROR 218: Cannot convert [packet name] address from EM to BM**  
Conversion from an Extended Mode address to a Basic Mode address failed.
- ERROR 219: (SYSLOG) Illegal subtype [subtype code] for log entry**
- ERROR 220: (SYSLOG) SYSLOG\$ packet rejected:**
- ERROR 221: (SYSLOG) Log entry length is less than 1**
- ERROR 222: (SYSLOG) Log entry type [entry code] is in exec range**
- ERROR 223: (SYSLOG) Log entry type [entry code] is in protected range**
- ERROR 224: (SYSLOG) Part of log entry outside callers addressing limits**

- ERROR 225:** (SYSLOG) Invalid BDI specified for the log entry
- ERROR 226:** (SYSLOG) User log entry longer than 100 words
- ERROR 227:** (SYSLOG) Log entry longer than max allowed
- ERROR 228:** (SYSLOG) SYSLOG\$ packet level is invalid
- ERROR 229:** (SYSLOG) SYSLOG\$ packet length is invalid
- ERROR 230:** (SYSLOG) Log entry could not be logged:
- ERROR 231:** (SYSLOG) Log file is disabled, downed or unusable  
?
- ERROR 232:** (SYSLOG) Class specified is not configured to be logged
- ERROR 233:** Invalid CCI\_TABLE pointer for remote host  
The address to the CMS connection ID table for the specified host is not valid.
- ERROR 234:** File assign or free rejected. FAC: [FAC entry code]
- ERROR 235:** reason: Open rejected by peer [peer name]
- ERROR 236:** First data image not found
- ERROR 237:** Could not send checkpoint request
- ERROR 238:** LPR rejected [number of rejections] times
- ERROR 239:** Failed to LPR the file [number of failures] times
- ERROR 240:** (SDFI) Cannot parse input name [input file/element name]
- ERROR 241:** (SDFI) SdfiOpen() failed on [file/element name] in pass %d
- ERROR 242:** (SDFI) SdfiRead() failed on [file/element name] in pass %d
- ERROR 243:** (SDFI) SdfiClose() failed on [file/element name] in pass %d

**ERROR 244: [macro name] not a symbolic SDF macro**

An OMNIBUS macro may have been specified as a SYMBOLIC macro.

**ERROR 244: Illegal local port [port number]. Range [*start-number - end-number*].**

**ERROR 244: Start port [*port number*] greater than end port [*port number*].**

**ERROR 244: Invalid console\_message\_group name.**

The group name specified is not in the list of configured names, as documented in the PARAM OPTIONS statement.

---

# **B**

# **Installation Checklist**

The following pages are a duplicate of the SPIN-X/LPD Installation Checklist supplied in the front pocket of this binder. This duplicate is provided in the case that the other loose copy has become lost and re-installation is required. For convenience, the pages have been printed simplex rather than duplex.

# C

# Unisys Problem List Entries

If you are on System Base Release SB4 and your operating system is Execlevel 43R8 or higher, or on SB5 with Exec level 44R4 or higher, the following PLE's are already resolved. Otherwise, the appropriate PCRs for the following PLEs should be installed.

Problem List Entry (PLE) Time 13:42:41 Date 960111

```
Number: 16336238      Product   : EXEC           Est Comp Date:
Status: CLOSED       Component: X8 CNS         Date Updated  : 950824
Type  : RELEASED     Revision  : 31           Date Prepared : 940906
Form  : TROUBLE REPORT Class    : SOFTWARE
Internal Use :                               Organization : EXECUTIVE-SYSTEMS
Originator  : UNISYS                               Responsibility: UNISYS
Error Source :                               Location    : ROS
Criticality :
Host Processor:
Replaced By  :
Replaces    :
```

## ----- Description Section -----

Headline :

EXERR-410 after termination of a run registered for keyins via ER-KEYIN\$.

Product Level	: 43R5	43R6	43R7	43R7*QS
	43R7X	44R1	44R2	44R3

Keyword: 2200/500 2200/900 CONSOLE  
ER-KEYIN\$ EXERR-410 M-SERIES

Symptoms: Symptoms Updated: 940906  
After entering a termination keyin for a run utilizing ER-KEYIN\$  
keyin registration, the system stops with an EXERR-410.

## Internal Conditions:

Jumpstacks from the dump will indicate that UKEYIN has encountered a class 8 interrupt (reference violation). This occurs in location counter 58 code on a 'BT X7,BPKT,\*A2,BLEAS,0' where BPKT=B8 and BLEAS=B14.

## Technical Explanation:

During the processing of an ER-KEYIN\$ deregistration function UKEYIN will acquire a stack frame and copy the users packet to that stack frame. To accomplish that task the users packet is based on BPKT (B8).

After the packet has been based on B8 the code goes on to process the request. During this processing UKEYIN will call a routine named STORSTAT. STORSTAT will repoint the waiting activity via a CRQRAL call. This is done so that a status of 2000 can be placed in the waiting activities packet. This CRQRAL call executes a LBED instruction which will alter the activity level BDTP (bank descriptor pointer). This altering of the activity level BDTP forces the hardware to scrub the activity level bank base registers B1-B15. This leaves base register B8 for UKEYIN void.

A later reference to B8 to copy the Exec's working copy of the user packet back to the users packet causes the reference violation and the EXERR-410.

## Dependencies:

This is dependent on 2200/900 and 2200/500 hardware, (M-series).

A second dependency is that the run must be utilizing activity level banks.

PLE to UCF : 77983748  
 PLE ref CONTACT : 78143734

----- Workaround Section -----  
 Workaround Description: Workaround Type :  
 ----- Resolution Section -----

Resolution Comments:

Code will be added to UKEYIN to save and restore the necessary  
 VA address and to reload the base register.

Fixed in System : SB4R8 SB5R4

Resolution for Level : 43R5  
 Resolution Status : FIX RELEASED Final Resolution Date: 950331  
 Closure Code : SD  
 PLE to CHG : 00006-62540-PCR  
 Fixed in Release : EXEC-43R8

Resolution for Level : 43R6  
 Resolution Status : FIX RELEASED Final Resolution Date: 950331  
 Closure Code : SD  
 PLE to CHG : 00006-62540-PCR  
 Fixed in Release : EXEC-43R8

Resolution for Level : 43R7  
 Resolution Status : FIX RELEASED Final Resolution Date: 950331  
 Closure Code : SD  
 PLE to CHG : 00006-62540-PCR  
 Fixed in Release : EXEC-43R8

Resolution for Level : 43R7\*QS  
 Resolution Status : FIX RELEASED Final Resolution Date: 950331  
 Closure Code : SD  
 PLE to CHG : 00006-62540-PCR  
 Fixed in Release : EXEC-43R8

Resolution for Level : 43R7X  
 Resolution Status : FIX RELEASED Final Resolution Date: 950331  
 Closure Code : SD  
 PLE to CHG : 00006-62540-PCR  
 Fixed in Release : EXEC-43R8

Resolution for Level : 44R1  
 Resolution Status : FIX RELEASED Final Resolution Date: 950824  
 Closure Code : SD  
 PLE to CHG : 00006-62570-PCR  
 Fixed in Release : EXEC-44R4

Resolution for Level : 44R2  
 Resolution Status : FIX RELEASED Final Resolution Date: 950824  
 Closure Code : SD  
 PLE to CHG : 00006-62570-PCR  
 Fixed in Release : EXEC-44R4

Resolution for Level : 44R3  
 Resolution Status : FIX RELEASED Final Resolution Date: 950824  
 Closure Code : SD  
 PLE to CHG : 00006-62570-PCR  
 Fixed in Release : EXEC-44R4  
 Number : 16336238 ~~~~~

Problem List Entry (PLE) Time 13:42:41 Date 960111

```

Number: 16385719      Product   : EXEC           Est Comp Date:
Status: CLOSED       Component: X8 CNS         Date Updated  : 951212
Type  : RELEASED     Revision  : 40           Date Prepared: 941102
Form  : TROUBLE REPORT Class    : SOFTWARE
Internal Use  :                               Organization : EXECUTIVE-SYSTEMS
Originator   : UNISYS                               Responsibility: UNISYS
Error Source  :                               Location     : ROS
Criticality   :
Host Processor:
Replaced By  :
Replaces     :
    
```

----- Description Section -----

Headline :

EXERR-410 in UKEYIN referencing a non-existent switchlist.

```

Product Level   : 43R5           43R6           43R7           43R7*QS
                  43R7X         44R1           44R2           44R3
Keyword: ER-KEYIN$           EXERR-410           KEYINS
          UKEYIN             UNSOLICITED-KEYIN
Symptoms:                                           Symptoms Updated: 941102
    
```

EXERR-410 in UKEYIN, referencing an activity that is listed in a registration queue buffer as waiting for a console keyin, but has already terminated.

Internal Conditions:  
 Technical Explanation:

A user has registered to receive unsolicited console keyins via ER-KEYIN\$.

The operator enters an unsolicited keyin that is to be handled by this program, but the program has yet to retrieve it.

The program issues an ER-KEYIN\$ with a 'wait' function, specifying the keyins for which it wants to wait.

ER-KEYIN\$ processing searches the registration queue buffer (RQB) for this run. For each keyin indicated, a check is made for an outstanding keyin: if there is one, the user activity is reactivated after being passed the keyin. If there is no keyin outstanding, the user switchlist is indicated as waiting for this keyin.

If, after finding no outstanding messages for several keyins, an outstanding keyin is found \*AND\* this keyin is too large to fit in the user buffer, an error is given to the ER-KEYIN\$ caller but the previously initialised entries, indicating that a switchlist is waiting, are not cleared. If another console keyin, that is to be handled by this same run, is entered, we attempt to reference the switchlist that is indicated as waiting but which has, in the case seen, already terminated. An EXERR-410 was the result, but other symptoms can only be guessed at.

Dependencies:  
 PLE to UCF : 78005728  
 PLE ref CONTACT : 78143734

----- Workaround Section -----

Workaround Description: Workaround Type :

----- Resolution Section -----

Resolution Comments:

If an error occurs when the user issues a 'wait' function and a keyin is found to process, clean up any other entries that have already been initialised in the registration queue buffer.

Also remove an extraneous jump instruction, change a hard-coded wait bit to use the standard definition and correct a range check.

```

Fixed in System      : SB4R8                      SB6

Resolution for Level : 43R5
Resolution Status    : FIX RELEASED                Final Resolution Date: 950331
Closure Code         : SD
PLE to CHG           : 00006-63281-PCR
Fixed in Release     : EXEC-43R8

Resolution for Level : 43R6
Resolution Status    : FIX RELEASED                Final Resolution Date: 950331
Closure Code         : SD
PLE to CHG           : 00006-63281-PCR
Fixed in Release     : EXEC-43R8

Resolution for Level : 43R7
Resolution Status    : FIX RELEASED                Final Resolution Date: 950331
Closure Code         : SD
PLE to CHG           : 00006-63281-PCR
Fixed in Release     : EXEC-43R8

Resolution for Level : 43R7*QS
Resolution Status    : FIX RELEASED                Final Resolution Date: 950331
Closure Code         : SD
PLE to CHG           : 00006-63281-PCR
Fixed in Release     : EXEC-43R8

Resolution for Level : 43R7X
Resolution Status    : FIX RELEASED                Final Resolution Date: 950331
Closure Code         : SD
PLE to CHG           : 00006-63281-PCR
Fixed in Release     : EXEC-43R8

Resolution for Level : 44R1
Resolution Status    : FIX RELEASED                Final Resolution Date: 951211
Closure Code         : SD
PLE to CHG           : 00006-63282-PCR
Fixed in Release     : EXEC-45R1

Resolution for Level : 44R2
Resolution Status    : FIX RELEASED                Final Resolution Date: 951211
Closure Code         : SD
PLE to CHG           : 00006-63282-PCR
Fixed in Release     : EXEC-45R1

Resolution for Level : 44R3
Resolution Status    : FIX RELEASED                Final Resolution Date: 951211
Closure Code         : SD
PLE to CHG           : 00006-63282-PCR
Fixed in Release     : EXEC-45R1
Number              : 16385719 ~~~~~
    
```

Problem List Entry (PLE) Time 08:38:12 Date 960528

```

Number: 16663379      Product   : EXEC           Est Comp Date: 970424
Status: RESOLVED     Component: X8 MSM        Date Updated  : 960429
Type  : RELEASED     Revision  : 68           Date Prepared: 960222
Form  : TROUBLE REPORT Class     : SOFTWARE
Internal Use  :                               Organization : EXECUTIVE-SYSTEMS
Originator   : UNISYS                               Responsibility: UNISYS
Error Source :                               Location     : ROS
Criticality  :
Host Processor:
Replaced By  :
Replaces     :
    
```

----- Description Section -----

```

Headline      :
Programs with extended mode activities are reported artificially big.
Product Level : 43R7X      43R8      43R8AQUAL    44R3
                44R4      44R4AQUAL    45R1      45R1A
                45R1BQUAL  45R2QUAL
Keyword: EXTENDED-MODE  M-SERIES  MEMORY
        MU-KEYIN       NPE        ONMSERIP
        RC-KEYIN       RCS        T-KEYIN
Symptoms:                               Symptoms Updated: 960226
    
```

Program with extended mode activities is reported artificially big.  
Internal Conditions:

In order to correct problems with SCA point during handling of RCS bank expansions, PCR 60562 and friends changed the RCS banks to be allocated the maximum size on M-series systems.

Technical Explanation:

The program size (IK) as reported via the RC, T and MU keyins will thus include the maximum size (262K) of all RCS banks (one for each extended mode activity in the program) even if only 1K is used.

Dependencies:

M-Series systems.

```

PLE to UCF      : 33560057 53008169 53041170 65156911 78082446
PLE ref CONTACT : 34150204
    
```

----- Workaround Section -----

```

Workaround Description:                               Workaround Type      :
    
```

----- Resolution Section -----

Resolution Comments:

Since a user has no control over the allocated size for any of the RCS banks, they are excluded from the IK calculation.

```

Resolution for Level : 43R7X
Resolution Status    : FIX AVAILABLE
Closure Code        : SD
PLE to CHG          : 00006-61192-PCR 00006-62977-PCR 00006-66806-PCR
                    : 00006-67086-PCR 00006-67499-PCR
Fixed in Release    :
    
```

```

Resolution for Level : 43R8
Resolution Status    : FIX AVAILABLE
Closure Code        : SD
PLE to CHG          : 00006-66806-PCR 00006-67086-PCR 00006-67499-PCR
    
```

```

Fixed in Release      :
Resolution for Level : 44R3
Resolution Status    : FIX AVAILABLE
Closure Code        : SD
PLE to CHG          : 00006-65215-PCR  00006-67087-PCR  00006-67500-PCR
Fixed in Release      :

Resolution for Level : 44R4
Resolution Status    : FIX AVAILABLE
Closure Code        : SD
PLE to CHG          : 00006-67087-PCR  00006-67500-PCR
Fixed in Release      :

Resolution for Level : 45R1
Resolution Status    : FIX AVAILABLE
Closure Code        : SD
PLE to CHG          : 00006-67088-PCR  00006-67501-PCR
Fixed in Release      :

Resolution for Level : 45R1A
Resolution Status    : FIX AVAILABLE
Closure Code        : SD
PLE to CHG          : 00006-67088-PCR  00006-67501-PCR
Fixed in Release      :
Number              : 16663379  ~~~~~
    
```

## C.1 PLEs for CMS

The following PLEs for level 8R3 have been shown to affect the operation of SPIN-X LPR/LPD. Therefore the associated PCRs should be installed if your level does not already include the fixes.

2200 PLE ( Problem List Entry ) 1721649 (<http://www.support.unisys.com/ALL/PL>)

Unisys Home Product Support Home ClearPath HMP IX / 2200 Support Home Contact Us  
Marketplace Year 2000 PLE ( Problem List Entry ) 17216490  
Symptoms Internal Conditions Technical Explanation ResolutionDetails

Failed transfers using TCP connection after cm\_tcpsetbuf procedure called  
Product: CMS1100  
Component: TCP Date Prepared: September 21, 1998  
Form: TROUBLE REPORT Date Updated: September 30, 1998  
Affected Level: 8R3 8R3A 8R3B  
Keyword: DATA-CORRUPTION FTP  
INPUT SESSION-CLOSE  
TCP TSAM

### SymptomsSymptoms:

A File transfer that is using the ignore push flag option fails. This happens after another connection that is also running with the ignore push flag option is aborted after it has received a close indication.

### InternalInternal Conditions:

TSAM traces show that more than one TCP connection is using the same tsam\_input\_buffer\_index.

### TechnicalTechnical Explanation:

A staging area is reserved for a TCP connection when procedure cm\_tcpsetbuf is called with the ignore push flag option. The area is freed when a close indication, abort indication, or abort confirm is received. After TCP queues an abort indication or abort confirm to the TSU the TCP connection is released. When a close indication is queued, the state of the connection is changed to "waiting for close request" and the connection is not deleted until the TSU sends a close request. The tsam\_input\_buffer\_index must be equal to zero if an abort indication or abort confirm is queued to the TSU when the connection is in the "waiting for close request" state.

### ResolutionDetailsResolution Details:

Resolution for Level: 8R3  
Resolution Status: FIX AVAILABLE  
Resolved with Change: 00014-20084-PCR \_\_\_\_\_  
Resolution for Level: 8R3A  
Resolution Status: FIX AVAILABLE  
Resolved with Change: 00014-20084-PCR \_\_\_\_\_  
Resolution for Level: 8R3B  
Resolution Status: FIX AVAILABLE  
Resolved with Change: 00014-20084-PCR Return to top of page

### 1999 Unisys Corporation

Component: TSAM Date Prepared: December 8, 1998  
Form: TROUBLE REPORT Date Updated: December 14, 1998  
Affected Level: 8R3 8R3A 8R3B  
Customer Technical Bulletin: ALERT CTB Date Updated: December 11, 1998  
Affected UCFs: 49977693  
Keyword: DATA-CORRUPTION INPUT  
TSAM

### SymptomsSymptoms:

When an application using the TSAM interface receives input data, the input is corrupted, and may actually be the data intended for a different TSAM TSU (transport service user). All of the other parameters on the receive input call are correct.

This problem cannot be detected from CMS 1100 traces.

Another TSAM application, such as cpFTP, is active at the time of the input data corruption.

InternalInternal Conditions:

A TSU using the TSAM interface, such as cpFTP, is receiving input through the TSAM interface. This application uses a special feature in the TSAM subsystem, which causes data to be staged in the TSAM subsystem and passed to the TSU in large blocks.

Two or more other TSAM TSUs or processes are also receiving input while a process such as cpFTP is receiving input. These TSUs are not using the special feature in the TSAM subsystem that causes data to be staged in the TSAM subsystem. These TSUs may experience input data corruption.

TechnicalTechnical Explanation:

To improve the performance of file transfer operations across the TSAM TCP/IP interface, CMS 1100 allows a TSAM TSU to specify an input mode on the "Set TCP Input Buffer Size" procedure call. This feature causes CMS 1100 to accumulate large amounts of input before passing it to the TSU. cpFTP uses this feature.

TSAM TSUs that do not use this feature, (nearly all other TSAM applications), share an input data area through which all input is passed. This input is protected by a field under test-and-set protection. However, a TSU such as cpFTP using the feature described above clears the field protecting the input data area used by other TSAM TSUs. If this field is incorrectly cleared, two or more "typical" TSAM TSUs can use the input data area at the same time, which causes input data corruption.

ResCommentsResolution Comments:

Since the correction applies to an element in the CMS 1100 TSAM subsystem, you must SOLAR install CMS 1100 to incorporate the change into the subsystem.

ResolutionDetailsResolution Details:

```

Resolution for Level: 8R3
Resolution Status:    FIX AVAILABLE
Resolved with Change: 00014-20151-PCR  _____
Resolution for Level: 8R3A
Resolution Status:    FIX AVAILABLE
Resolved with Change: 00014-20151-PCR  _____
Resolution for Level: 8R3B
Resolution Status:    FIX AVAILABLE
Resolved with Change: 00014-20151-PCR  Return to top of page
1999 Unisys Corporation
    
```

2200 PLE ( Problem List Entry ) 17245422

Unisys Home Product Support Home ClearPath HMP IX / 2200 Support Home Contact Us  
 Marketplace Year 2000 PLE ( Problem List Entry ) 17245422  
 Symptoms Internal Conditions Technical Explanation ResolutionComments  
 ResolutionDetails

A TSAM application receives corrupted input data

Product: CMS1100

Problem List Entry (PLE) Time 09:18:41 Date 960912

```

Number: 16558222      Product   : CMS1100      Est Comp Date:
Status: RESOLVED     Component: TCP        Date Updated  : 950810
Type  : RELEASED     Revision  : 7         Date Prepared: 950804
Form  : TROUBLE REPORT Class    : SOFTWARE
Internal Use  :                               Organization : COMMUNICATION DEV.
Originator   : UNISYS                               Responsibility: UNISYS
Error Source  :                               Location      : ROS
Criticality   :
Host Processor:
Replaced By   :
Replaces      :
    
```

----- Description Section -----

```

Headline      :
TCP connections hang
Product Level : 8R1      8R1A      8R1B      8R2
Keyword: CONNECTION-HANG      HANG      PROCESS
          TCP
    
```

Symptoms: Symptoms Updated: 950810  
 CMS 1100 processes or remote user processes that use CMS 1100  
 TCP may experience connection hangs. A CMS 1100 STATUS TCP command  
 shows connections in the FIN-WAIT-2 state.

Internal Conditions:  
 A processed CMS 1100 trace with TCP traces medium shows that a CMS 1100  
 TSU (most likely TAS) receives a TCP open indication, then the TSU  
 immediately issues a CLOSE request for that connection. The trace shows  
 that the TCP connection is in the SYN-RCVD state when the CLOSE is received.  
 The TCPDUMP portion of a processed CMS 1100 dump may show several,  
 or many, TCP connections in the FIN-WAIT-2 state.

Technical Explanation:  
 This problem is caused by CMS 1100 TCP not sending a FIN segment  
 when it should. TCP receives an open (SYN segment) from a peer and notifies  
 the TSU that an open indication is waiting. The TSU receives the open,  
 causing TCP to ACK the SYN and also send a SYN segment. This sequence  
 puts the connection into the SYN-RCVD (SYN received) state. The TSU  
 then immediately issues a CLOSE request, which should cause TCP to send a  
 FIN to the peer, but TCP does not send it. The connection then enters the  
 FIN-WAIT-1 state as if a FIN had been sent. CMS 1100 TCP then receives the  
 ACK for the previous SYN. This appears to TCP to be the ACK for the FIN  
 that was never sent. This causes the connection to enter the FIN-WAIT-2  
 state, waiting for a FIN from the peer. The peer, having never received  
 the FIN, is now in the ESTABLISHED state. If the peer does not have data  
 to send, the connection remains hung with CMS 1100 TCP waiting for the  
 peer's FIN, which will never come.

Dependencies:  
 PLE to UCF : 52990013  
 PLE ref CONTACT :

----- Workaround Section -----

Workaround Description: Workaround Type :





----- Resolution Section -----

Related PLE : 15745665 16453919

Resolution for Level : 7R4  
 Resolution Status : FIX AVAILABLE  
 Closure Code : SD  
 PLE to CHG : 00014-16791-PCR 00014-16879-PCR 00014-16880-PCR  
 00014-16891-PCR 00014-16948-PCR 00014-16951-PCR  
 00014-16976-PCR 00014-17015-PCR 00014-17047-PCR  
 00014-17091-PCR 00014-17099-PCR 00014-17111-PCR  
 00014-17128-PCR 00014-17130-PCR 00014-17133-PCR  
 00014-17182-PCR 00014-17197-PCR 00014-17209-PCR  
 00014-17220-PCR 00014-17254-PCR 00014-17317-PCR  
 00014-17412-PCR 00014-17429-PCR 00014-17445-PCR  
 00014-17458-PCR 00014-17573-PCR 00014-17883-PCR  
 00014-17908-PCR 00014-18309-PCR 00014-18327-PCR  
 00014-18416-PCR 00014-19056-PCR

Fixed in Release :

Resolution for Level : 8R1  
 Resolution Status : FIX AVAILABLE  
 Closure Code : SD  
 PLE to CHG : 00014-17862-PCR 00014-17906-PCR 00014-18417-PCR  
 00014-19057-PCR

Fixed in Release :

Resolution for Level : 8R1A 8R1B  
 Resolution Status : FIX AVAILABLE  
 Closure Code : SD  
 PLE to CHG : 00014-18417-PCR 00014-19057-PCR

Fixed in Release :

Resolution for Level : 8R2  
 Resolution Status : FIX AVAILABLE  
 Closure Code : SD  
 PLE to CHG : 00014-19058-PCR

Fixed in Release :

Number : 16669148 ~~~~~

Problem List Entry (PLE) Time 09:18:42 Date 960912

Number: 16675814 Product : CMS1100 Est Comp Date:  
 Status: RESOLVED Component: TSAM Date Updated : 960322  
 Type : RELEASED Revision : 12 Date Prepared: 960315  
 Form : TROUBLE REPORT Class : SOFTWARE  
 Internal Use : Organization : COMMUNICATION DEV.  
 Originator : UNISYS Responsibility: UNISYS  
 Error Source : Location : ROS  
 Criticality :  
 Host Processor:  
 Replaced By :  
 Replaces :



Problem List Entry (PLE) Time 09:18:42 Date 960912

```

Number: 16689017      Product   : CMS1100      Est Comp Date:
Status: OPEN         Component: TCP        Date Updated  : 960904
Type  : RELEASED     Revision  : 63        Date Prepared: 960410
Form  : TROUBLE REPORT Class    : SOFTWARE
Internal Use  :      Organization : COMMUNICATION DEV.
Originator   : UNISYS      Responsibility: UNISYS
Error Source  :      Location    : ROS
Criticality  :
Host Processor:
Replaced By  :
Replaces     :
    
```

----- Description Section -----

```

Headline      :
TCP connection remains in CLOSE WAIT state indefinitely
Product Level : 7R4      8R1      8R1A      8R1B
                8R2
Keyword: CONNECTION-HANG      DDN1100      FTP
        SESSION-HANG        TCP
    
```

Symptoms: Symptoms Updated: 960412  
 A TCP connection will not close after, for example, a DDN1100 file transfer.  
 Internal Conditions:

The CMS 1100 trace shows a TCP connection in the ESTABLISHED state with more than one segment in the output queue, the last of which has the FIN flag set. CMS 1100 then receives ACK from the peer TCP, acknowledging all of the segments in the output queue, followed by ACK FIN, which puts the connection into the CLOSE\_WAIT state. The connection remains in the CLOSE\_WAIT state.

Technical Explanation:

When the 2200 application requests that a connection be closed, CMS 1100 TCP sends FIN to the peer TCP and puts the connection into the FIN\_WAIT\_1 state. In the case at hand, the FIN was preceded by several output segments that had not yet been acknowledged by the peer TCP. When the retransmission timer expires, CMS 1100 TCP moves the unacknowledged segments from the save queue to the output queue for retransmission, and temporarily changes the connection state from FIN\_WAIT\_1 to ESTABLISHED so that the output routine will resend the output.

If CMS 1100 receives an ACK that acknowledges all of the segments that are in the output queue pending retransmission, CMS 1100 TCP should recognize that both sides have now sent FIN, and the connection should enter the TIME\_WAIT (closed) state. Instead, CMS 1100 TCP "forgets" that it had sent FIN, and sets the connection state to CLOSE\_WAIT, then waits forever for the application to issue a close request, which it has already done.

A similar problem can arise for connections in the LAST\_ACK state since TCP changes the state to CLOSE\_WAIT when it retransmits an output segment.

Dependencies:

```

PLE to UCF      : 65167891
PLE ref CONTACT :
    
```

----- Workaround Section -----

```

Workaround Description:      Workaround Type      :
    
```

## ----- Resolution Section -----

## Resolution Comments:

.  
 One or more of the CHGs needed to resolve this problem adds a new element to the CMS 1100 BASE file. Select KEY=NEW-ELT-CHG to identify those CHGs. You must use the procedure outlined in the Workaround Section of COMUS PLE 10649242 to integrate those CHGs.  
 .

Resolution for Level : 8R1  
 Resolution Status : FIX IN DEVELOPMENT  
 Closure Code : SD  
 Change in Resolution : CHG ADDED Change in Resolution Date : 960725  
 PLE to CHG : 00014-17584-PCR 00014-17611-PCR 00014-17613-PCR  
 00014-17615-PCR 00014-17617-PCR 00014-17629-PCR  
 00014-17653-PCR 00014-17682-PCR 00014-17893-PCR  
 00014-17948-PCR 00014-17956-PCR 00014-17960-PCR  
 00014-18128-PCR 00014-18166-PCR 00014-18181-PCR  
 00014-18197-PCR 00014-18199-PCR 00014-18466-PCR  
 00014-18598-PCR 00014-19031-PCR 00014-19093-PCR  
 00014-19195-PCR 00014-19248-PCR 00014-19251-PCR

Fixed in Release :

Resolution for Level : 8R2  
 Resolution Status : FIX IN DEVELOPMENT  
 Closure Code : SD  
 Change in Resolution : CHG ADDED Change in Resolution Date : 960725  
 PLE to CHG : 00014-19032-PCR 00014-19094-PCR 00014-19196-PCR  
 00014-19249-PCR 00014-19252-PCR

Fixed in Release :

Resolution for Level : 7R4  
 Resolution Status : FIX AVAILABLE  
 Closure Code : SD  
 Change in Resolution : CHG ADDED Change in Resolution Date : 960711  
 PLE to CHG : 00014-19101-PCR 00014-19214-PCR

Fixed in Release :

Resolution for Level : 8R1A 8R1B  
 Resolution Status : FIX IN DEVELOPMENT  
 Closure Code : SD  
 Change in Resolution : STATUS CHANGE Change in Resolution Date : 960807  
 PLE to CHG : 00014-18128-PCR 00014-18166-PCR 00014-18181-PCR  
 00014-18197-PCR 00014-18199-PCR 00014-18466-PCR  
 00014-18598-PCR 00014-19031-PCR 00014-19093-PCR  
 00014-19195-PCR 00014-19248-PCR 00014-19251-PCR

Fixed in Release :

Number : 16689017 ~~~~~

Problem List Entry (PLE) Time 09:18:42 Date 960912

```

Number: 16689092      Product   : CMS1100      Est Comp Date:
Status: RESOLVED     Component: TCP        Date Updated  : 960906
Type  : RELEASED     Revision  : 16        Date Prepared: 960410
Form  : TROUBLE REPORT Class    : SOFTWARE
Internal Use  :                               Organization : COMMUNICATION DEV.
Originator   : UNISYS                               Responsibility: UNISYS
Error Source  :                               Location     : ROS
Criticality  :
Host Processor:
Replaced By  :
Replaces     :
    
```

----- Description Section -----

```

Headline      :
TCP connection is reset (aborted) by the peer TCP.
Product Level : 8R1      8R1A      8R1B      8R2
Keyword: CONNECTION-LOST      FTP      RESET
          TCP
    
```

```

Symptoms:                               Symptoms Updated: 960524
A connection may be aborted soon after CMS 1100 TCP sends a SYN. A file may
contain a small amount of data (up to a segment) but no EOF after an FTP
file transfer terminates.
    
```

```

Internal Conditions:
A LAN trace would show that the peer sent data with the YN.ACK flags.
CMS 1100 TCP would then ACK the YN.ACK with an ACK number one greater
than it should. The peer and CMS 1100 TCP then would get into an extended
exchange of ACKs due to the bad ACK number.
    
```

```

Technical Explanation:
On receipt of the YN.ACK+DATA the next_byte_expected gets incremented
by one due to the SYN flag. CMS 1100 TCP acknowledges the YN.ACK and
then processes the data, increasing the next_byte_expected by the segment
length (data length + 1, per RFC 793). This means that the SYN flag is
counted twice.
    
```

```

Dependencies:
PLE to UCF      : 49566948
PLE ref CONTACT :
    
```

----- Workaround Section -----

```

Workaround Description:                               Workaround Type      :
----- Resolution Section -----
    
```

```

Resolution for Level : 8R1      8R1A      8R1B
Resolution Status    : FIX AVAILABLE
Closure Code         : SD
PLE to CHG           : 00014-19156-PCR
Fixed in Release     :
    
```

```

Resolution for Level : 8R2
Resolution Status    : FIX AVAILABLE
Closure Code         : SD
Change in Resolution: CR      Change in Resolution Date : 960906
PLE to CHG           : 00014-19157-PCR
Fixed in Release     :
Number               : 16689092 ~~~~~
    
```

Problem List Entry (PLE) Time 09:18:42 Date 960912

```

Number: 16905887      Product   : CMS1100      Est Comp Date:
Status: OPEN         Component: TCP         Date Updated  : 960904
Type  : RELEASED     Revision  : 12         Date Prepared: 960820
Form  : TROUBLE REPORT Class    : SOFTWARE
Internal Use  :                               Organization : COMMUNICATION DEV.
Originator   : UNISYS                               Responsibility: UNISYS
Error Source  :                               Location     : ROS
Criticality  :
Host Processor:
Replaced By  :
Replaces     :
    
```

----- Description Section -----

```

Headline      :
Connection hang or abort, or CMS 1100 abort in LANH
Product Level : 8R1      8R1A      8R1B      8R2
Keyword: ABORT      BUFFERS      OUTPUT
      SESSION-HANG      TCP
    
```

Symptoms: Symptoms Updated: 960820

A connection hangs or is aborted, or CMS 1100 aborts in the element LANH.  
 This happens if CMS 1100 CHG 19195 or 19196 has been applied.

Internal Conditions:

The abort is an ER ERR\$ at approximately 010545 in LANH.

Technical Explanation:

The connection can hang or abort because TCP sends output from multiple data buffers without aligning the data on word boundaries. This causes garbage bytes to be sent in the middle of an output.

The LANH abort is due to TCP sending too many bytes of output for the medium, which is a result of not aligning the data on word boundaries.

Dependencies:

PLE ref CONTACT :

----- Workaround Section -----

```

Workaround Description:      Workaround Type      :
    
```

----- Resolution Section -----

Resolution Comments:

.  
 One or more of the CHGs needed to resolve this problem adds a new element to the CMS 1100 BASE file. Select KEY=NEW-ELT-CHG to identify those CHGs. You must use the procedure outlined in the Workaround Section of COMUS PLE 10649242 to integrate those CHGs.

Related PLE : 16881899 16902594

```

Resolution for Level : 8R1
Resolution Status   : FIX IN DEVELOPMENT
Closure Code       : SD
Change in Resolution : STATUS CHANGE      Change in Resolution Date : 960904
PLE to CHG         : 00014-17584-PCR 00014-17611-PCR 00014-17613-PCR
                   : 00014-17615-PCR 00014-17617-PCR 00014-17629-PCR
                   : 00014-17653-PCR 00014-17682-PCR 00014-17893-PCR
                   : 00014-17948-PCR 00014-17956-PCR 00014-17960-PCR
                   : 00014-18041-PCR 00014-18099-PCR 00014-18128-PCR
                   : 00014-18166-PCR 00014-18181-PCR 00014-18197-PCR
                   : 00014-18199-PCR 00014-18466-PCR 00014-18598-PCR
                   : 00014-19031-PCR 00014-19093-PCR 00014-19195-PCR
                   : 00014-19233-PCR 00014-19248-PCR 00014-19251-PCR
    
```

Fixed in Release :

```

Resolution for Level : 8R1A      8R1B
Resolution Status   : FIX IN DEVELOPMENT
Closure Code       : SD
    
```

```

Change in Resolution : STATUS CHANGE          Change in Resolution Date : 960904
PLE to CHG          : 00014-18041-PCR 00014-18099-PCR 00014-18128-PCR
                   : 00014-18166-PCR 00014-18181-PCR 00014-18197-PCR
                   : 00014-18199-PCR 00014-18466-PCR 00014-18598-PCR
                   : 00014-19031-PCR 00014-19093-PCR 00014-19195-PCR
                   : 00014-19233-PCR 00014-19248-PCR 00014-19251-PCR

Fixed in Release    :

Resolution for Level : 8R2
Resolution Status   : FIX IN DEVELOPMENT
Closure Code        : SD
Change in Resolution : STATUS CHANGE          Change in Resolution Date : 960904
PLE to CHG          : 00014-19032-PCR 00014-19094-PCR 00014-19196-PCR
                   : 00014-19249-PCR 00014-19252-PCR

Fixed in Release    :
Number              : 16905887 ~~~~~
    
```

```

Problem List Entry (PLE)                                Time 10:26:07   Date 961216
~~~~~
Number: 16665657          Product   : CMS1100          Est Comp Date:
Status: RESOLVED         Component: TCP            Date Updated  : 961211
----- Description Section -----
Headline      :
CMS 1100 runs out of buffer space.
Product Level : 8R1          8R1A          8R1B          8R2
Keyword: BUFFERS          TCP
Symptoms:                               Symptoms Updated: 960228
  CMS 1100 runs out of buffer space.
Internal Conditions:
  A dump shows many items on the TCP delayed release chain, with the intransit
  flag set in the first item.
    
```

```

Problem List Entry (PLE)                                Time 10:26:08   Date 961216
~~~~~
Number: 16689017          Product   : CMS1100          Est Comp Date:
Status: RESOLVED         Component: TCP            Date Updated  : 961211
----- Description Section -----
Headline      :
TCP connection remains in CLOSE WAIT state indefinitely
Product Level : 7R4          8R1          8R1A          8R1B          8R2
Keyword: CONNECTION-HANG DDN1100          FTP          SESSION-HANG          TCP
Symptoms:                               Symptoms Updated: 960412
  A TCP connection will not close after, for example, a DDN1100 file transfer.
Internal Conditions:
  The CMS 1100 trace shows a TCP connection in the ESTABLISHED state with
  more than one segment in the output queue, the last of which has the FIN
  flag set. CMS 1100 then receives ACK from the peer TCP, acknowledging
  all of the segments in the output queue, followed by ACK FIN, which
  puts the connection into the CLOSE_WAIT state. The connection remains
  in the CLOSE_WAIT state.
    
```

Problem List Entry (PLE) Time 10:26:08 Date 961216

Number: 16878588 Product : CMS1100 Est Comp Date:  
 Status: RESOLVED Component: TCP Date Updated : 961212

----- Description Section -----

Headline :

CMS1100 aborts in BUFGMT

Product Level : 8R1 8R1A 8R1B 8R2

Keyword: ABORT BUFFERS FATALERROR

Symptoms: Symptoms Updated: 960723

CMS1100 terminates due to a bad release of a buffer. The CMS1100 dump top-down scan of the general buffer pool shows unequal control words for a buffer, one control word beginning with 374 and the other with 774. LANH probably logs a CODE=104 INVALID MESSAGE LENGTH error shortly before the abort. This problem occurs on systems that have installed CMS 1100 CHGs 19031 or 19032.

Internal Conditions:

This can occur if CMS 1100 receives a TCP ACK that acknowledges both a segment that had been retransmitted and a following segment that had been sent but not yet retransmitted. The subsequent segment must begin in the data buffer that also contained the end (or all) of the segment that was retransmitted.

Problem List Entry (PLE) Time 10:26:08 Date 961216

Number: 16881899 Product : CMS1100 Est Comp Date:  
 Status: RESOLVED Component: TCP Date Updated : 961211

----- Description Section -----

Headline :

CMS 1100 buffer pool is depleted by output on TCP connections

Product Level : 8R1 8R1A 8R1B 8R2

Keyword: BUFFERS OUTPUT TCP THRESHOLDING

Symptoms: Symptoms Updated: 960627

The CMS 1100 general buffer pool is depleted. Buffer availability may be below 15%. Users cannot open new connections. CMS 1100 may appear hung.

Internal Conditions:

The buffer pool is depleted by output data awaiting acknowledgment by the peer TCP.

Problem List Entry (PLE) Time 10:26:09 Date 961216

Number: 16902594 Product : CMS1100 Est Comp Date:  
 Status: RESOLVED Component: TCP Date Updated : 961211

----- Description Section -----

Headline :

CMS 1100 aborts with a bad buffer release

Product Level : 8R1 8R1A 8R1B 8R2

Keyword: ABORT BUFFERS RECOVERY TCP

Symptoms: Symptoms Updated: 960812

CMS 1100 aborts with a bad buffer release, probably by TCP-SUBS or TCP-EVENTS. This happens if CMS 1100 CHG 19195 or 19196 has been applied. Other symptoms are possible, resulting from premature release and reuse of TCP output buffers.

Internal Conditions:

The buffer being released was released prematurely when a retransmitted segment was acknowledged.



Problem List Entry (PLE) Time 10:26:10 Date 961216

Number: 16956279 Product : CMS1100 Est Comp Date:  
 Status: RESOLVED Component: TCP Date Updated : 961211

----- Description Section -----

Headline :

CMS 1100 aborts in TCP-OUTPUT, or connection hangs instead of sending FIN

Product Level : 8R1 8R1A 8R1B 8R2

Keyword: ABORT CONNECTION-HANG SESSION-CLOSE TCP

Symptoms: Symptoms Updated: 961207

1. CMS 1100 aborts, probably in procedure build\_data\_segment of TCP-OUTPUT.  
 The abort may occur on the exit from this procedure (J 0,X11).
2. A connection hangs after the application gives TCP a close request.  
 If TCP tracing is on, the trace file contains frequent aggregation timer  
 expirations for the connection. In the transport connection table  
 (TCT), the FIN flag is set.

These problems occur in systems that contain the CMS 1100 CHG 19195 and  
 19288 or 19196 and 19289.

Internal Conditions:

In case 1 above, the aborting activity's X1, which points to the PLUS stack,  
 has been corrupted.



MAKERPT.EXE is a utility that facilitates the creation and maintenance of various types of data files used for printing printjobs. The utility, also called Make Report, can be considered a type of compiler. As a compiler, Make Report must be presented with a source file, and, depending on the contents of the source file, it will attempt to generate one or more output files. The contents of the generated output files are defined within the source file using a command syntax unique to Make Report. Make Report's command syntax allows users to define symbolically the contents of practically any arbitrary binary file required.

MAKERPT2200 is a streamlined version of MAKERPT.EXE which runs on the Unisys 2200. It's input is nearly identical to the PC version's input but its output is one or more element/version(s) which corresponds to filename.extension for the PC version. The format of the output file exactly matches the format of a binary file FTP'd to the 2200 from a PC when FTP has been installed on the 2200 **without** the -b option. This utility is discussed in Appendix E, which is devoted to Makerpt2200 and the various other SPIN-X utilities.

---

## **D.1 Make Report Source Files**

---

Make Report source files are simple ASCII text files that can be created and maintained with most standard text editors, including DOS's EDLIN.EXE and EDIT.EXE. While the Make Report compiler can handle line lengths of up to 250 characters, one would typically want to limit source lines to 80 characters or less. There is no limit (except for what your editor might impose) to the number of lines that can be contained within a source file or to the number of output files that can be generated from one Make Report source file.

Each source file may contain comments, macro definitions and one or more output file definition blocks. An output definition block begins with an "@" character followed by a legal DOS filename. This will become the output file name. All output definition blocks must be terminated with an "@end" token. Makerpt will create for each such block one output file having the same name as presented in the *@name*. A sample MAKERPT.DAT source file has been included later in this chapter.

## D.1.1 Specifying Source Files

---

Starting with MAKERPT V2.10, one must explicitly or implicitly specify the source file name as a parameter to MAKERPT. This can be accomplished from the command line in the Xpress directory by one of three methods:

- 1) MAKERPT *source.dat*
- 2) MAKERPT -s:*source.dat*
- 3) MAKERPT -s

The first and second method explicitly define the file name "source.dat" as the input source file. The second and third method use the -s startup switch to define the source file name. Using the -s startup switch with no filename specified implicitly defines the file name "MAKERPT.DAT" as the input source file. At this point the location of the source file has not yet been determined.

The source filename is assumed to have an extension of ".DAT" if no extension is provided. Using the examples above, the user could have supplied name of "source" and Make Report would have used the name "source.dat". If the user wants to use a source file that does not contain an extension, then simply append a trailing "." to the file name, and MAKERPT will use that file name.

### D.1.1.1 Make Report Subdirectories

---

Different printers require different setup to achieve the same result, yet the various format names often are the same regardless of the printer. To prevent confusion, set up subdirectories representing various output device types, i.e. printers like "HP", "LPS", "EPSON", etc. Below is what a typical Makerpt subdirectory might look like:

```
C:\UTILITIES\Makerpt.exe
  \DEVTYPES\HP\HP.DAT
  \DEVTYPES\XGF\XGF.DAT
  \DEVTYPES\IMPACT\IMPACT.DAT
  \DEVTYPES\CPS\CPS.DAT
  \DEVTYPES\LPS\LPS.DAT
  \DEVTYPES\UDK\UDK.DAT
  \DEVTYPES\ MINIMUM\ MINIMUM.DAT
```

Because the contents of a .DAT file are usually printer specific, it is within these device type specific subdirectories that MAKERPT should look to find the .DAT files for each printer.

Now suppose that MAKERPT has just been invoked. Unless MAKERPT was executed with either the startup switch -b or -d, the user will be presented with the following prompt:

**Use DEVTYPES subdirectory? (Y/N)**

If the user answers **Y**, MAKERPT presents the following prompt to the user.

Xpress devtypes:

```
HP
XGF
CPS
LPS
UDK
IMPACT
MINIMUM
```

Please choose from above list ;

The user must select one of the DEVTYPES subdirectories by entering its name on the command line. If he or she answers **N**, then all files are opened relative to the current directory (i.e. "base them here").

The MAKERPT startup switches -d and -b bypass this prompt and imply either a **Y** or **N** response respectively. If the devtypes switch has been presented, then MAKERPT expects one of several things after the switch. If there is a space and a sourcefile name right after the -d switch, then the operator will be presented with the list of subdirectories and will be asked to choose one by typing it on the command line. The sourcefile will then be found or created in the indicated subdirectory off of the subdirectory "DEVTYPES\". Alternatively, a colon and devtype (no spaces) followed by a space and a sourcefile name will locate the source file in the indicated devtype directory without prompting the user with the devtype list.

## D.1.2 Sample Execution of Makerpt

Suppose that you are defining the banner, trailer and report body for the device type 'HP'. The sequence of steps you would go through in order to initialize the control data sent to the printer would be as follows:

1. Go to the DEVTYPES subdirectory HP and copy the HP.DAT file to MAKERPT.DAT, or if this has already been done, copy MAKERPT.DAT to MAKERPT.BAK. Making a backup before making changes is just good policy. Calling the file MAKERPT.DAT is not required if you want to specify, for instance, NEWPCL.DAT when you execute Makerpt. Doing this step simply enables one to specify the MAKERPT.DAT file with "-s". This might be useful if changes are being made and tested often.
2. Use a text editor to make the appropriate changes, if any, in the MAKERPT.DAT file. (You will need to read further to find out what these are.)
3. While still in the 'HP' subdirectory execute Makerpt. For example:

```
C:\UTILITIES\MAKERPT.EXE -B -C -N MAKERPT
```

4. Ftp the resulting Setup files to the 2200 and reference them as described in the SPIN-X LPD Reference Guide.

The execution of Makerpt on the NEWPCL.DAT will result in the creation of the following files in the 'HP' subdirectory. Each file corresponds to an @name... @end output block in the .DAT file. Each @name is a legal DOS name with an extension.

**IMPORTANT NOTE:** The extension **must** be specified to keep Makerpt from defaulting to the behavior expected by Xpress. If there is no extension, Makerpt will create **two** files, a .BIN and a .RPT. The .BIN file will have the Setup sequences.

PS80.PCL	- Portrait Simplex 80 columns
PS80W.PCL	- Portrait Simplex 80 columns, with line wrap
PD80.PCL	- Portrait Duplex 80 columns
PD80W.PCL	- Portrait Duplex 80 columns, with line wrap
LS132.PCL	- Landscape Simplex 132 columns
LS132G.PCL	- Landscape Simplex 132 columns, gray-bar
LD132.PCL	- Landscape Duplex 132 columns
LS132G.PCL	- Landscape Simplex 132 columns, gray-bar
PD132.PCL	- Portrait Duplex 132 columns

**NOTE:** All formats shown above have a default of 66 lines per page.

### D.1.3 Start up Switches

The start up switches discussed above are defined in a help screen which can be accessed by the either of the commands below:

```
makerpt -h
makerpt -?
```

The help screen is shown below.

```
Usage: MAKERPT [switches] scriptName[.ext]
Where scriptName is the primary MAKERPT input script file name.
NOTE: If not specified script files have a default extension of ".DAT".

Switches:

-b          Base Switch: Bypass "Use "DEVTYPE\$\" subdirectory? (Y/N)"
           prompt. Base all files on current directory.

-d[:devType] DevType Switch: Bypass "Use "DEVTYPE\$\" subdirectory? (Y/N)"
           prompt. Base all files on supplied devType subdirectory,
           or present list of valid devtype subdirectories if no
           devtype is supplied.

-@:name     @name Switch: Rebuild only specified name. By default MAKERPT
           attempts to rebuild all @name blocks found in script.

-c          Compile Switch: Bypass the "edit or compile?" prompt.

-e[:editor] Edit Switch: Bypass the "edit or compile?" prompt. If user
           chooses -e then MAKERPT attempts to shell to a text editor.
           The default text editor is "EDIT". To specify an alternate
           editor, provide the name with -e switch or specify it via
           the environment variable "MAKERPT=editor".

-y          Yes Prompt Switch: Bypass the "Confirm overwrites?" prompt.

-n          No Prompt Switch: Bypass the "Confirm overwrites?" prompt.

-s[:scrName] Script File Switch: Use scrName if specified as primary
           script file name or "MAKERPT.DAT" if scrName is not
           specified.

-m:scrName  Macro File Switch: Use scrName to resolve all macro call
           definitions instead of using primary script file.

Startup switches can be presented in any order. If any switches conflict, (-b and
-d for example) then the switch which was presented last will take precedence.
```

## D.1.4 Make Report Tokens

---

As MAKERPT processes a source file, it attempts to discover tokens. Tokens are ASCII character sequences that represent the smallest units of data that MAKERPT comprehends. Macro names, output definition block names and control sequences are all tokens. Valid tokens are created with ASCII characters within the range of an ASCII Exclamation point ('!', x21) to an ASCII TILDE ('~', x7E). Valid tokens can be as small as a single character such as a single ASCII digit that can be used to represent a "Decimal number" token or as large as a 250 character sequence that can be used to represent one of the "Literal" tokens. Tokens must be delimited from one another by one (or more ASCII space), ASCII comma, and/or ASCII carriage return/line feed character sequences.

The following token types are recognized by MAKERPT:

- Comment tokens
- Decimal number tokens
- Hexidecimal number tokens
- Octal number tokens
- ASCII mnemonic name tokens
- Control character tokens
- Quoted literal tokens
- Double quoted literal tokens
- Slashed literal tokens
- Macro name tokens
- Output name tokens
- Procedure Name tokens
- Procedure Specific tokens

Many of these token types are symbolic representations of data while others imply an action. For example, an "Output name" token defines an action that creates a file.

## D.1.5 Output Files

---

As MAKERPT.EXE parses an input source file, it attempts to locate output file name tokens and then create the filenames requested based on the tokens defined within each output definition block. Output files are defined by an '@name' command (for instance '@mydata') followed by a definition block and an '@END' command.

The creation of the output file is initiated when MAKERPT discovers an @name command (i.e. @name.ext). The @name represents the name of the file that is to be created. Each @name parameter should be denoted as a complete name. For example the name of @FORM.PCL would result in the creation of the file "FORM.PCL" One can create files with any particular DOS file name and extension.

Unless the -m switch is presented, MAKERPT will attempt to resolve all macro definitions within the current source file. The user can indicate that a secondary source file will be used for macro name resolution via the -m switch.

**Important Note:** The primary source file and the secondary source file can both contain output file definitions blocks as well as macro definition statements, but if MAKERPT is using a secondary source file for macro resolution, then **all** macro definition statements within the primary source will be ignored as will **all** output file definition blocks within the secondary file.

Unless the -m switch is presented, MAKERPT will attempt to build all output files defined within the primary source file. The contents of each output file is defined by MAKERPT commands within the command block defined by an @name command and its subsequent @END command.

## D.1.6 Macro Tokens

---

A Make Report macro name begins with an ASCII period "." character. The MAKERPT macro name token has two different roles depending on whether the macro name token was found within an output file definition block:

If a macro name token is discovered within an *@name*, @END output definition block, then this is a macro CALL and MAKERPT will scan through the source file from the beginning to discover the definition of the macro. In order for a macro call to be successful, there must be a corresponding macro definition.

If the macro .name token is discovered outside of a output definition block and it is the first token on the line, then it is called a macro DEFINE. Any primary tokens following the macro token determine the macro's definition.

A macro call causes the primary tokens that define the macro to be written to the current output file. Unless the -m switch is presented, MAKERPT will attempt to resolve all macro calls in the primary file. Macro calls are resolved when MAKERPT discovers the first occurrence of .name located outside of an *@name*, @END block.

**Note:** The same macro name can be defined multiple times, but only the **first** occurrence is used to define the contents of a specific macro call. This allows the user to easily experiment with other variations of a macro by simply defining the same macro name more than once. Any .name command located outside of an *@name*, @END block is called a macro definition. Macro definitions can precede or follow any *@name*, @END command blocks.

Unless one wishes to keep the macro source file separate from the output source file, all the macros should be defined either at the beginning or at the end of the MAKERPT.DAT file. If one chooses to provide a secondary source file for macro definitions, then **all** macro resolutions will be attempted against the macro source file. Thus, any macro definitions contained within the primary source file will be ignored.

Macro definitions are built from primary tokens. Output file definitions are built from primary tokens, macro calls, and procedure calls. MAKERPT recognizes several different types of primary tokens (see: MAKERPT Primary Tokens).

Macro definitions (if present) begin with '.' character followed immediately by its name and optionally followed by one or more MAKERPT primary tokens, and optionally a MAKERPT comment. A macro definition is typically used to define a symbolic name for a set of primitive commands that may be used quite often. For example:

```
.reset ESC "E" ; This macro represents the HP reset sequence
```

would define the macro name ".reset" to represent the 2 byte HP LaserJet reset printer sequence. This macro token ".reset" could then be used in place of the two primary tokens ESC and "E" in subsequent output file definition blocks.

Macros have the following characteristics:

- Macros are not case sensitive; the macro defined above would be resolved with any of the following macro calls: ".reset", ".RESET", or even ".rEsEt".
- Macro definitions are currently limited to one line. It is anticipated that a future version of MAKERPT will allow the ASCII backslash '\' character to indicate that a given macro definition is to be continued onto the following line
- Macro nesting is not allowed; one cannot call another macro from within a macro.
- Macros currently cannot support procedure calls (i.e. !name commands).

The macro tokens and primary tokens found within an output definition block determine the contents of the created file. When a macro token is found within an output definition block, MAKERPT writes the contents of the macro's definition to the created file.

## D.1.7 Make Report Primary tokens

Make Report primary tokens represent the most direct method for describing symbolically the contents of an output definition block. These tokens compile directly into predictable binary values.

**Note 1:** All tokens must be separated from each other with white space or commas. White space includes ASCII SPACE (0x20), ASCII Horizontal tab (0x09), ASCII CR (0x13), ASCII LF (0x10) and ASCII FF (0x0A) characters.

- 1) Decimal Number Tokens: one or more decimal digits, 0-9.

0,255,32 ===; 0x00 0xFF 0x20

- 2) Hexidecimal Number Tokens: The letter 'x' or 'X' followed one or two hexadecimal digits, 0-9, A-F or a-f.

x0 ===; 0x00  
XfA ===; 0xFA

- 3) Octal Number tokens: The letter 'o' or 'O' followed by one to three octal digits, 0-7.

o177 ===; 0x7F  
377 ===; 0xFF

- 4) Control Character tokens: The ASCII caret (0x5E) character followed by a single character, A-Z, a-z, @, [, \, ], ^, or \_.

^@ ===; 0x00  
^a ===; 0x01  
^A ===; 0x01  
^\_ ===; 0x31

- 5) Single Quoted Literal Tokens: An ASCII apostrophe (0x27) character followed by one or more displayable ASCII characters followed by a terminating ASCII apostrophe (') character.

'data' ===; 0x64 0x61 0x74 0x6

- 6) Double Quoted Literal Tokens: Same as above except uses ASCII double quote (0x22) character to delimit the data

"data" ===; 0x64 0x61 0x74 0x61

- 7) Literal Data in Slashes: Same as above except uses the ASCII forward slash (0x2F) character as the delimiter.

/data/ ===; 0x64 0x61 0x74 0x61

- 8) ASCII Mnemonic Name Tokens

STX ===; 0x02  
CR ===; 0x0D  
LF ===; 0x0A

**Note 2:** A semicolon character defines the rest of the current line as a comment (unless defined within one of the literal primitives).

## D.1.8 Make Report Procedure Tokens

---

MAKERPT procedure tokens begin with an ASCII exclamation point (!) character. MAKERPT procedure tokens differ from MAKERPT primary tokens in that primary tokens allow one to know ahead of time the exact byte content of the resulting output file. This is possible because primary tokens are exact symbolic representations of the data that is to be written to the output file. Procedure tokens, on the other hand, typically perform some type of operation on an external data file that is somehow used in the process of creating the output file.

Procedures have the following characteristics:

- The procedure name always begins with an ASCII exclamation point (0x21) character.
- Many, but not all, procedures may require parameters.
- Some procedures take a fixed number while others require a variable number of parameters.
- Procedures that take a variable number of parameters have a terminating token to delimit the end of procedure parameters.
- Each procedure may have its own rules concerning how the procedure's parameters (if any) must be formatted (i.e. freeform, column specific, etc.).

The following Make Report procedures are currently defined:

```
!INCLUDE  
!PERFORM  
!PAUSE  
!TRANSLATE  
!FCBBUILD
```

---

### D.1.8.1 The !INCLUDE Procedure

---

The Make Report !INCLUDE procedure simply copies the contents of a supplied file into the current output file. The !INCLUDE procedure requires one parameter, the name of the input file to be copied from. If no pathing components are supplied with the filename, then the file is assumed to be located in the current "default path", otherwise it is opened from the path specified.

```
!INCLUDE filename
```

This procedure requires one parameter, filename. This procedure performs a binary copy of filename into output file. One possible use for this procedure is to include font data contained within an external file into an output file.

```
@DATA1.OUT
.load_font      ; Make Report macro call.
!INCLUDE FONT1.DAT ; Make Report include file.
.activate_font  ; Make Report macro call.
@END
```

The !INCLUDE procedure can also be used to concatenate several files into one. For example:

```
@ALLDATA.OUT
!INCLUDE DATA1.IN
!INCLUDE DATA2.IN
!INCLUDE DATA3.IN
!INCLUDE DATA4.IN
@END
```

**Note:** The filename parameter must be separated from the procedure token by one or more bytes of white space, but it must be on the same line of source as the procedure token.

### **D.1.8.2 The !PERFORM Procedure**

---

This is a special version of the !INCLUDE procedure that is intended for use with HP/PCL print files created by the PerForm form generation utility. This procedure, if successful, copies only a subset of the supplied filename to the output file. The !PERFORM procedure requires one parameter, the name of the input file to be copied from. If no pathing components are supplied with the filename, then the file is assumed to be located in the current "default path", otherwise it is opened from the path specified.

`!PERFORM filename`

**Note:** The filename parameter must be separated from the procedure name by one or more bytes of white space, but it must be on the same line of source as the !PERFORM token.

### **D.1.8.3 The !PAUSE procedure**

---

**!PAUSE**

This procedure causes Make Report to temporarily suspend compiling. A message is output to the screen informing the user that Make Report processing is temporarily stalled. The user is prompted to enter any key to continue processing. The user can optionally press ESC and abort MAKERPT processing.

### D.1.8.4 The !TRANSLATE procedure

!TRANSLATE procedure facilitates the creation and maintenance of various translation table files. Starting with Xpress V1.55h, translate table files can be used (during printing) to alter the normal ASCII character mapping. This Xpress feature is enabled whenever Xpress discovers an ".XLT" file in the appropriate DEVTYPEES directory whose filename matches the current form name. If none is found, then Xpress looks in DEFAULTS for \$DEFAULT.XLT. This translate table will be used by all forms that don't have translate tables bearing their names.

The !TRANSLATE procedure requires a variable number of parameters. The first parameter always represents the translation table type and must be either the token "XPRESS" or "STANDARD". Use the XPRESS token when generating a ".XLT" file to be used during Xpress print processing. The !TRANSLATE procedure is terminated by the token "!END".

**Note:** The STANDARD token can be used to generate 256 byte translate table files. Characters are indexed into the table by the actual hex value of the character and are replaced by whatever is found there. XPRESS type !TRANSLATE files are 512 bytes in length and are not necessarily compatible with ".XLT" files used by other utilities such as Dataware's CHANNEL.SYS. Corresponding to each byte in a standard .XLT table, XPRESS type .XLT tables have an extra byte which supports the 'suppress' command. This is why the XPRESS type table is twice as big.

Both the XPRESS and STANDARD keywords initialize their respective tables, so that all input values pass through unchanged by default. The REASSIGN token can be used to modify a code point from its default. The REASSIGN token requires two parameters: The first is a number that represents the code point to be reassigned, and the second the value that should be output in it's place. These parameters can be represented as either decimal, hexadecimal, or octal tokens.

```
@table1.xlt
;
; The following is a STANDARD (i.e. 256 byte) translate table
; that redefines the ASCII NUL character to an ASCII SPACE.
!translate standard
    reassign x00 x20
!end
@end
```

If the translation table type is type XPRESS, one can also define code points that are to be suppressed during printing. This is accomplished by using the keyword SUPPRESS followed by the number of the code point to be suppressed.

```
@table2.xlt
;
; The following is an XPRESS (i.e. 512 byte) translate table
; that redefines the ASCII NUL character to an ASCII SPACE and
; suppresses any xFF characters.
!translate xpress
    reassign x00 x20
    suppress xff
!end
@end
```

The TRANSLATE feature is enabled by invoking the !TRANSLATE procedure within a MAKERPT source file definition block (i.e. within a @name @END).

The !TRANSLATE procedure requires a minimum of two parameters. The first is the table type keyword, either XPRESS or STANDARD. The XPRESS table type is proprietary to GSU. The STANDARD type generates a standard 256 byte translate table that is compatible with many third party products including Dataware. A !TRANSLATE procedure must always be terminated by a !END keyword.

Between the table type keyword and !END keyword, two keywords, REASSIGN and SUPPRESS (type XPRESS only), can be defined. The REASSIGN keyword requires two parameters: The first represents the code point to be modified, and the second represents the value to be output. The SUPPRESS keyword requires one parameter, the code point that is to be suppressed. Valid code points and values are integer numbers between 0 and 255 inclusive. These integer numbers can be presented to the !TRANSLATE procedure as either decimal, octal or hexadecimal strings (i.e. the strings "REASSIGN 0 32", "REASSIGN o0 o40", or "REASSIGN x0 x20", would all result in a translate table that maps an ASCII NUL character to an ASCII SPACE).

.

**Note:** The keywords are not case specific, and the SUPPRESS keyword is valid only for XPRESS type translate tables.

**!TRANSLATE Examples**

Although not required, for readability it is recommended that each REASSIGN or SUPPRESS command in the !TRANSLATE procedure be assigned to its own line. The following three examples all create identical translate tables, but the readability of the input source is quite obviously different:

**TRANSLATE.DOC: Example 1: "Very compact source, but hard to read."**

```
@test.out
!TRANSLATE XPRESS reassign x7F x08 reassign x00 x20 !END ; Two
reassignments.
@end
```

**TRANSLATE.DOC: Example 2: "Each token on its own line."**

```
@test.out
!TRANSLATE; This table
XPRESS; also
reassign; generates
x7F; two
x08; reassignments
reassign; but since its
x00; spread out so
x20; much, it's
!END; hard to read.
@end
```

**TRANSLATE.DOC: Example 3: "Easy to read."**

```
!translate XPRESS          ; Generate an Xpress type table.
  reassign x7F x08; ASCII DEL become ASCII BS
  reassign x00 x20; ASCII NUL become ASCII SPACE
!end; We're done.
```

**!TRANSLATE.DOC: Example 4**

```
;
; The following !TRANSLATE procedure creates a 256 byte translate
; table that reassigns ASCII control characters (x00-x1F and x7F)
; to the ASCII SPACE character.
; 13t
```

```
!TRANSLATE STANDARD ; We're defining a 256 byte translation table
```

```
reassign x00 x20
reassign x01 x20
reassign x02 x20
reassign x03 x20
reassign x04 x20
reassign x05 x20
reassign x06 x20
reassign x07 x20
reassign x08 x20
reassign x09 x20
reassign x0A x20
reassign x0B x20
reassign x0C x20
reassign x0D x20
reassign x0E x20
reassign x0F x20
reassign x10 x20
reassign x11 x20
reassign x12 x20
reassign x13 x20
reassign x14 x20
reassign x15 x20
reassign x16 x20
reassign x17 x20
reassign x18 x20
reassign x19 x20
reassign x1A x20
reassign x1B x20
reassign x1C x20
reassign x1D x20
reassign x1E x20
reassign x1F x20
reassign x7F x20
```

```
!END ; required procedure end statement
```

```
@end
```

**!TRANSLATE.DOC: Example 5**

```
;  
; The following !TRANSLATE procedure creates a 512 byte translate  
; table that suppresses printing of characters (x00-x1F and x7F).  
;  
!  
!TRANSLATE XPRESS ; We're defining a 512 byte translation table  
  suppress x00  
  suppress x01  
  suppress x02  
  suppress x03  
  suppress x04  
  suppress x05  
  suppress x06  
  suppress x07  
  suppress x08  
  suppress x09  
  suppress x0A  
  suppress x0B  
  suppress x0C  
  suppress x0D  
  suppress x0E  
  suppress x0F  
  suppress x10  
  suppress x11  
  suppress x12  
  suppress x13  
  suppress x14  
  suppress x15  
  suppress x16  
  suppress x17  
  suppress x18  
  suppress x19  
  suppress x1A  
  suppress x1B  
  suppress x1C  
  suppress x1D  
  suppress x1E  
  suppress x1F  
  suppress x7F  
!  
!END ; required procedure end statement  
@end
```

**Makerpt ASCII Mnemonic Name Tokens**

<u>Token</u>	<u>Decimal Value</u>	<u>Hexadecimal Value</u>
NUL	0	x00
SOH	1	x01
STX	2	x02
ETX	3	x03
EOT	4	x04
ENQ	5	x05
ACK	6	x06
BEL	7	x07
BS	8	x08
HT	9	x09
LF	10	x0A\
VT	11	x0B
FF	12	x0C
CR	13	x0D
SO	14	x0E
SI	15	x0F
DLE	16	x10
DC1	17	x11
DC2	18	x12
DC3	19	x13
DC4	20	x14
NAK	21	x15
SYN	22	x16
ETB	23	x17
CAN	24	x18
EM	25	x19
SUB	26	x1A
ESC	27	x1B
FS	28	x1C
GS	29	x1D
RS	30	x1E
US	31	x1F
DEL	127	x7F
EO	255	xFF

**Note:** Case of token does not matter, i.e. NUL, nul, and Nul all are recognized as the same ASCII mnemonic name token.

**Makerpt Control Character Tokens**

<u>Token</u>	<u>Decimal Value</u>	<u>Hexidecimal Value</u>
^@	0	x00
^A or ^a	1	x01
^B or ^b	2	x02
^C or ^c	3	x03
^D or ^d	4	x04
^E or ^e	5	x05
^F or ^f	6	x06
^G or ^g	7	x07
^H or ^h	8	x08
^I or ^i	9	x09
^J or ^j	10	x0A
^K or ^k	11	x0B
^L or ^l	12	x0C
^M or ^m	13	x0D
^N or ^n	14	x0E
^O or ^o	15	x0F
^P or ^p	16	x10
^Q or ^q	17	x11
^R or ^r	18	x12
^S or ^s	19	x13
^T or ^t	20	x14
^U or ^u	21	x15
^V or ^v	22	x16
^W or ^w	23	x17
^X or ^x	24	x18
^Y or ^y	25	x19
^Z or ^z	26	x1A
^[	27	x1B
^\	28	x1C
^]	29	x1D
^^	30	x1E
^_	31	x1F

**Note:** Case of token does not matter, i.e. ^A, and ^a are both recognized as the same control character token.

**MAKERPT.DAT file Example**

```

; MAKERPT.DAT
; 9-18-92 (sysjhy)
;
; This is a sample makerpt.dat file that attempts to demonstrate
; some of the techniques available.
;
;
; MACROS section:
;
; First we define some sample macros.
;
; NOTE: A macro definition must fit on one line.
;

.macro1      STX "MACRO1" x13 x10 ETX      ; 10 Bytes of data
.macro2      x0D x0A CR LF 13 10          ; 6 bytes of data
.macro3      ; 0 bytes of data (i.e. an empty macro definition)

;
; Report section
;

@mydata      ; This starts a report definition for MYDATA.RPT and
; MYDATA.BIN
;
; This report definition uses both primitive
; and macro commands.
;

"A/;Z"      ; Literal data for 5 bytes
'A;"Z'      ; Literal data for 5 bytes
/A;"Z/      ; Literal data for 5 bytes
CR LF ETX STX FF      ; Mnemonic Data for 5 bytes
^@ ^A ^L ^[ ^z      ; Control Data for 5 bytes.
255 255 0 1 43      ; Decimal Data for 5 bytes
x55 xFa X0A x0 x3; Hexidecimal Data for 5 bytes
o177 o377 O4 O21 o256; Octal data for 5 bytes
.macro1      ; Data from macro
.macro2      ; Data from macro
.macro3      ; Data from macro

"END",SP,CR      ; Five bytes comma delimited

@end

; The following report generates an empty MYDATA2.BIN file as well
; as a report file MYDATA2.RPT that calls MYDATA2.BIN.
;
@mydata2     ; This starts a report definition for MYDATA2.RPT and
; MYDATA2.BIN
@end

```

This block will generate ONLY the file MYDATA.BAN because the .ban extension has been specified.

```
@mydata.ban
  "BINARY PERFORM" CR LF ; Generate command to load perform.bin
  "COORDINATES LINES" CR LF
@END
```

To execute MAKERPT, simply type "MAKEPRT *sourceFileName*" at the DOS prompt and follow the various prompts. Execute MAKERPT with the -h switch to get more details on the optional command line switches.

---

### D.1.8.5 The !FCBUILD procedure

---

The Makerpt procedure !FCBUILD is used to create Form Control Buffer (FCB) files that can be used by Xpress during runtime to generate an appropriate 3211 Load FCB command for 3211 channel impact printers attached to Xpress via the Dataware channel board. The FCB files must be installed in the Xpress DEVTYPEES subdirectory, must have the extension ".FCB" to their filenames, and must be enabled for the printer via the XPCONFIG utility.

**Background:** Xpress requires that a driver be installed in CONFIG.SYS to support channel attached printers connected to the Dataware channel card. The driver accepts data presented by DOS and does the necessary conversion to create appropriate Channel Command Word (CCW) commands for sending data to the printer. The driver can accept data in one of several forms, including ASCII data records and EBCDIC Variable Length Record (VLR) data records. One type of CCW command that Dataware can generate is a Load FCB command.

**Note:** ASCII data records cannot be used to load an FCB into a 3211 printer: only the VLR record can be used to load an FCB into a Dataware attached printer. VLR records are padded by Xpress to the exact length which the Dataware driver expects.

Dataware supplies two device drivers with its channel card, CHANNEL.SYS and MULTIPRN.SYS.

- MULTIPRN.SYS must be loaded in CONFIG.SYS if more than one channel printer is to be supported. Up to four copies can be loaded, one for each printer supported. The DOS device name defaults to "CHANnn" where nn is the channel unit address configured with the "c=" parameter in each device driver.
- CHANNEL.SYS (or MULTIPRN.SYS) can be loaded if only one printer is to be supported. CHANNEL.SYS uses slightly less memory than MULTIPRN.SYS and its Dataware printer DOS name defaults to "CHAN".

**Syntax:** The Makerpt !FCBBUILD procedure takes a variable number of parameters and is terminated by the !END token. The following !FCBBUILD procedure commands are defined.

- TYPE=type .

The required TYPE command defines the type of FCB being created. It must be the FIRST command defined after the !FCBBUILD command. Currently the only type defined is "3211".

- LPP=nnn

The required LPP command defines the number of lines per page. This must be an integer value between 2 and 180 inclusive.

- LPI=nn

The required LPI command defines the number of lines per inch. This must be either the integer value 6 or 8.

- ASSIGN=(c,nnn)

The optional ASSIGN command is used to assign a channel, c, to a specific line, nnn. The assign command, if included, cannot be used until the TYPE, LPP and LPI commands have been defined. The channel character, c, must be one of 12 different characters from '1' to '9' or 'A' to 'C'. The line, nnn, must be a value from 1 to 180. The ASSIGN command can be used multiple times, and the same channel can be assigned to multiple lines.

**note1:** Xpress can currently support only a "skip to channel 1". A typical Xpress FCB usually has a single channel '1' defined at line 1 (i.e. "ASSIGN=(1,1)").

**note2:** If no ASSIGN command is defined, then FCBBUILD assumes the user wants channel 1 defined at line 1 (this is normally the case).

**note3:** When dealing with forms such as 1 inch labels that contain 12 labels between folds, one could define either an FCB that contains 6 lines and has channel 1 defined at line 1, or an FCB with 72 lines that defines channel 1 at lines 1, 7, 13, 19, 25, 31, 37, 43, 49, 55, 61, and 67.

**!FCBBUILD Examples****Example 1: for note 3.**

```

@1X6@6.FCB
; This FCB defines a form that is 6 lines long (i.e. 1 inch).
!FCBBUILD
  TYPE=3211
  LPP=6
  LPI=6
  ASSIGN(1,1)
!END
@end

```

**Example 2: for note 3.**

```

@12X6@6.FCB
;
; This FCB defines a form that is 72 lines long (i.e. 12 inches),
; but contains 12 "top-of-forms" every six lines.
;
!FCBBUILD
  TYPE=3211
  LPP=72
  LPI=6
  ASSIGN(1,1)
  ASSIGN(1,7)
  ASSIGN(1,13)
  ASSIGN(1,19)
  ASSIGN(1,25)
  ASSIGN(1,31)
  ASSIGN(1,37)
  ASSIGN(1,43)
  ASSIGN(1,49)
  ASSIGN(1,55)
  ASSIGN(1,61)
  ASSIGN(1,67)
!END
@end

```

**Example 3:**

```

@fcb1.fcb
; Minimal FCBBUILD parameters needed to create a 3211 66X6 FCB with
; Channel 1 defined at line 1.
!FCBBUILD
  TYPE=3211
  LPP=66
  LPI=6
!END
@end

```

**Example 4:**

```
@fcb2.fcb
; Minimal FCBBUILD parameters needed to create a 3211 88x8 FCB with
; Channel 1 defined at line 1.
;
!FCBBUILD
  TYPE=3211
  LPP=88
  LPI=8
!END
@end
```

**Example 5:**

```
@fcb3.fcb
; More explicit FCBBUILD parameters used to create a 3211 66X6 FCB
; with Channel 1 defined at line 1.
; NOTE: This results in an FCB file identical to Example 3.
!FCBBUILD
  TYPE=3211
  LPP=66
  LPI=6
  ASSIGN(1,1)
!END
@end
```

## **Report Setup file for compilation by Makerpt or Makerpt2200**

```

; ld132b.dat
; Execute:  makerpt ld132b.dat -b -c -n ;
; (will create ld132b.pcl - this is indicated by the
; @ symbol)
; Generate PCL sequence for:
; landscape duplex 132x66 with SEMI_BOLD weight text.
;
; The following Macro is defined by placing a dot in front
; of the macro name. When the macro name appears inside an
; @name.. @end block the data in the macro will replace the
; .name

.testmacro1 "This is macro text data" CR LF

@ld132b.pcl                ; Landscape, Duplex, 66
                          ; lines/page, 132 columns
                          ; with 7/16 top margin on front
                          ; sheet and 7/16bottom margin on
                          ; back sheet.
                          ; This margin is appropriate for
                          ; long edge
; .testmacro1  ; this has been commented out because it is
;              ; only for purposes of demonstration. Removing
;              ; the semicolon would cause the text in the
;              ; macro to be printed in front of the report. If
;              ; the macro had Makerpt tokens instead of text,
;              ; the tokens would be compiled.

ESC "E"                  ; PCL Reset
ESC "&l1S"                ; PCL Duplex Print, long edge
                          ; binding
ESC "&l1O"                ; PCL Logical Page Orientation
                          ; (Landscape)

; The following command shifts a landscape long-edge
; bound logical page left 0/720 of an inch on the
; front side of a sheet and right 0/720 of an inch on the
; back side of a sheet.

ESC "&l0Z"                ; Top Offset Registration
                          ; Command: (+/- 1/720 inch)

; The following command shifts a landscape long-edge
; bound logical page down 90/720 of an inch on the
; front side of a sheet and up 90/720 of an inch on
; the back side of a sheet.

ESC "&l90U"               ; Left Offset Registration
                          ; Command: (+/- 1/720 inch)

; The following set of commands set the default
; font characteristics for the page. These
; values determine the specific size of the bit-mapped
; font used. The default VMI and HMI characteristics
; for this font will be modified in a later set of
; commands.

```

```

ESC "(10U"           ; PCL Primary Symbol Set = PC-8
ESC "(s0P"           ; PCL Primary Spacing (fixed
                    ; pitch)
ESC "(s12H"          ; PCL Primary Pitch (CPI)
ESC "(s1B"           ; PCL Primary stroke
                    ; weight = semi-bold

```

; The following commands allow us to get 132+  
; characters within each line. If we desire a  
; wider left hand margin we must increase the  
; left margin value and reduce the HMI value  
; appropriately.

```

ESC "&k9.65H"         ; PCL Horizontal Motion Index
                    ; (HMI) value (in 1/120ths)
ESC "&a0L"             ; PCL Left Margin: # (in
                    ; columns)

```

; The following commands allow us to get 66 lines  
; per page with a 7/16 inch top margin. If we  
; desire a taller top margin then we must increase  
; the top margin value and reduce the VMI value  
; appropriately. If we desire to longer bottom  
; margin then reduce the VMI value appropriately.

; NOTE: To establish a logical page that is  
; appropriate for front/back shifting we must  
; do two sets VMI/TopMargin commands to center  
; the logical page correctly.

```

ESC "&l18C"           ; PCL Vertical Motion index
                    ; value (in 1/48ths)
ESC "&l1E"            ; PCL Top Margin = 3 lines down
                    ; from top
ESC "&l15.7C"         ; PCL Vertical Motion index
                    ; value (in 1/48ths)
ESC "&l66F"           ; PCL Text Length = 66 lines

```

; The following command insures that a single  
; ASCII form-feed character will result in the  
; HP restoring the logical cursor to the left  
; hand margin after the page eject.

```

ESC "&k2G"           ; PCL Line Termination: FF-CR/FF,
                    ; LF-CR/LF

```

@end



---

# E

# LPD Processor Utilities

The following utility programs have been placed into the *[qualifier]\*UTILITY* file during the installation process.

---

## E.1 CLEANQ

---

A privileged-mode program used to reset print queues for contingency reasons. CLEANQ requires the user to have SMOQUE\$ privileges.

CLEANQ should only be executed for print queues that have files left with the in-process status set because of an abnormal LPD processor termination.

CLEANQ will reset the 'in-process' status from all files in a single print queue. It is executed as follows:

```
@[qualifier]*UTILITY.CLEANQ [queue name]
```

Unpredictable results will occur if the in-process flag is reset when the file is being processed by any program.

---

## E.2 PLIST

---

PLIST 2R1 is an OS2200 based Spin-X supplied utility program to help parse a supplied SDF input file and to generate various dump reports from it. Letter options and optional parameters determine which type of PLIST reports will be generated and which records within the sdf-input-file shall be reported on. Specifically this utility can be used to expose either the entire contents of or a subset of the various types of SDF control records (such as the 050, 060 and 061 images) as well expose the contents of the various data records. The PLIST processor accepts an SDF print file or element as input, processes the records and lists out the following information:

1. Control word of the record.
2. Record offset in the source print file.
3. Words in the Record image, excluding the control word.
4. Record image:
  - For 050 type control record: FIELDATA or OCTAL or both.
  - For 060 type control record: ASCII/FIELDATA or OCTAL or both
  - For 061 type control record: Octal dump and the values of all the subtype/fields in the control record.
  - For all other types of control records: Octal dump.
  - For Data records: ASCII or Octal or both.

PLIST can be invoked with the following processor call statement:

```
@[qualifier]*UTILITY.PLIST[,options] sdf-input-file[.element/version],  
[max-records],[start-record],[,output-file]
```

The sdf-input-file parameter is the only required parameter and it should represent the name of the source SDF file for which a report is desired. This parameter must represent the name of an existing SDF data file. (NOTE: Generating reports from elements of a program file is not currently supported). A terminating period character '.' is optional; if one is not specifically provided PLIST will insert one.

**options** A number of different types of reports may be generated depending on the letter-options presented to the processor call line. The defined letter-options include:

- A - Enable octal report for all records.
- B - List omnibus element (default is symbolic)
- C - Enable octal report for all control records.
- D - Enable octal report for all data records.
- L - Enable octal report for label control record.
- N - Generate Narrow (80 column) report.
- O - Include word offset values in standard and octal reports.
- P - Enable octal report for print control records (060, 061).
- Q - Print debug output.
- R - Enable record summary report.
- S - Use space ' ' as word delimiter within text part of octal reports.
- T - Enable text dump report.
- W - Generate wide (132 column) report.
- Z - Generate record count summary only.

The optional max-records and start-record parameters are used to generate a report on a subset of the supplied sdf-input-file.

**max-records** The max-records parameter defines the maximum number of records to be used in generating the report. By default the entire file is dumped if max-records is not specified. If max-records is specified, then PLIST will terminate after the number of records indicated by the max-records parameter is processed. Note: This value is typically used to dump the first several records of a file.

Example: To print just the first 10 records of the sdf-input-file.

```
@PLIST my-print-file.,10
```

**start-record** The optional start-record parameter defines the first record within the supplied sdf-input-file that PLIST process. By default PLIST starts reporting with the first record (i.e. record #1, the SDF 050 label control record) of the sdf-input-file if start-record is not specified. If start-record is specified then record numbers less than the supplied start-record are effectively excluded from the output report. Note: This feature is typically used to generate a report on the last part of a file.

The optional output-file parameter defines the name of an alternate file to which report output should be directed. This feature may be used as an alternative to breakpointing. If the outfile-name specified does not exist, PLIST will create the file as a temporary PCIOS (type 'X') data file. Optionally one may catalog the output-name prior to executing PLIST to create a permanent PCIOS (type 'X') data file.

**NOTE:** If NO sdf-input-file parameter is specified, several usage messages are printed and the program exits.

**NOTE:** PLIST reports can be very verbose (especially when using my preferred call options of 'ANOR'). Breakpointed reports can be extremely long if the input file is large! Remember that there is a max-records field (field 2) to limit PLIST processing to a subset of the input file.

**NOTE:** It is often useful to generate a Z option report first to determine whether or not it is advisable to use the max-records and/or start-record parameters for subsequent reports.

Example: Generate a record count summary report only:

```
@PLIST,Z my-print-file.
```

## E.2.1 Output of PLIST Processor

A number of different reports may be generated based on the letter options specified on the processor call line.

The standard report (i.e. no letter options specified) generates a one line report record for each SDF control record and SDF data record detected. This report record includes a 12 digit octal number representing the control word of the sdf record followed by a message describing the type of SDF record this control word represents: i.e. "Type 'P' (PRINT\$) Label Control Record", "Type 'P' Data Record", "End-of-file Control Record", etc.

### Octal Report Options

The 'A', 'C', 'D', 'L', and 'P' letter options enable the octal report to be generated for various types of control and/or data records. An octal report, if enabled, is generated immediately following the standard report record. An octal report consists of two sections: a left section containing the octal dump and a right section containing a text dump. The octal dump contains 12 digit octal numbers representing the values of the various data words contained with the current SDF record.

- A** An 'A' option enables the generation of octal reports for all records within the file, both control and data.
- C** The 'C' option enables octal reports for all control records.
- D** The 'D' option enables octal reports for all data records.
- L** The 'L' option enables the octal report for the label control record.
- P** The 'P' option enables octal reports for the 060 and 061 print control records.

### Text Report Option

- T** The 'T' letter option causes the descriptive message text to be replaced with a "text" dump of the data contained within a data record: i.e. the message "Type 'X' Data Record" is replaced with records such as "this is a line of data from my data file". An empty record, i.e. one with no data is represented as "". The text dump record may be continued on one or more lines if there is not enough room to fit it on the initial line. The number of words dumped per line depends on the current width, i.e. NARROW or WIDE (see N and W options below).

The text dump may represent either ASCII or FIELDATA characters, with each word of data delimited by either an ASCII pipe, '|', characters or optionally by ASCII space, ' ', characters. The ASCII space delimiter is enabled with the 'S' letter option. An ASCII text dump is indicated by the presence of four characters between each delimiter; a FIELDATA text dump is indicated by the presence of six characters between each delimiter. Unprintable ASCII values are indicated by ASCII question-mark, '?', characters. The number of words dumped per line depends on the current width, i.e. NARROW or WIDE (see N and W options below).

## Other Options

The 'N' and 'W' options are used specify whether a NARROW or WIDE report should be generated. The user may optionally force a desired report width with either the 'N' or the 'W' option. If both the 'N' and 'W' options are specified, the 'W' takes priority.

- N** A narrow report contains records with 79 or less characters and is useful when the report will be viewed directly on a terminal.. By default, PLIST will generate a narrow report if output is to a demand terminal.
- W** A wide report contains records of 132 characters or less and is useful when the report will be printed. If output has been breakpointed, or if PLIST is being executed in batch, a wide report will be generated by default.
- R** The 'R' option will generate a one line record summary report that precedes each record of the standard report. The record summary report includes a #number, a record type description field, a possible spacing field, a word count field, and a code type field.
- O** The 'O' option will enable the printing of offset values for standard report records, and for octal reports. The offset value for the standard report represents the number or words from the beginning of the file that the current control word is located at. The offset values printed within octal reports represent the offsets to the various data words within the current record relative to the first data word of the current record.

The 'O' call option generates VIRTUAL offset values relative to the beginning of the file for each SDF record parsed. PLIST uses the UCS SDFI read routines which "cooks" the SDF records. Generally the OFFSET values reported by PLIST can NOT be used to index into a report generated by "raw" file dumping utilities such as @FANG. (But PLIST OFFSET values may be useful to get one "in the ballpark" when reviewing reports on the same SDF file generated by @FANG .)

- Z** The 'Z' option is a special case report that simply summarizes the count of various types of records contained within the sdf-input-file. This report is often useful to determine whether or not is is practical to generate a report on the entire input file, or whether it is more useful to process just a subset of the input file by specifying the max-records and/or start-record parameters.

A typical usage might be the following

```
@PLIST,ANOR myqual*myfile.,100
```

which will generate a narrow ('N') report for up to the first 100 records encountered. All data and control records ('A') will be dumped with offsets ('O') and with an initial summary record ('R').

## E.2.2 A Short Description Of Sdf Files.

---

PLIST is utility that attempts to read and format the contents of an SDF data file. An SDF data file can contain both control and data records. While it relatively easy for a user to review the contents of data records with various editors such as @ED, @IPF and @CTS, it is much more difficult, if not impossible, to review the contents of the various control records with the same utilities. Other utilities such as @FANG and @FED allow one to examine all words within the file but leave it to the user to determine where one record ends and the next one begins. PLIST by default generates a one line summary report record for each control and data record encountered within the supplied sdf-input-file. Exposure of the control records is often critical in determining why certain files are processed the way they are by the various Spin-X products i.e. Spin-X/Central, Spin-X/RMS, Spin-X/Xpress and Spin-X/LPR-LPD. Specifically, enqueued SDF print files (i.e. Type 'P' (PRINT\$) files) are processed differently than other types of enqueued SDF files. SDF print files may contain various types of SDF control records (i.e. the 060 and 061 records) that dictate printing attributes such as the logical page length, top-margin, bottom-margin, special form request messages, etc. It is the 060 records specifically and the vertical spacing associated with the various type 'P' data records that dictate how the various Spin-X products must paginate and format an SDF print file.

### **Sdf Control Records**

---

Each record contained within an SDF file requires a control word.

Every SDF file begins with a Label Control Record. S1 of the control word is '050', indicating that this record is a Label Control Record.

The SDF control word indicates whether or not the current record is a control record or a data record and whether or not there are any data words associated with the current record.

**Note:** Both control records and data records may contain data words.

S1 of a control word indicates whether the current record is a control record or a data record. If it is a control record, then S2 determines the number of data words associated with the current control record. S3 of a control word in an SDF label control record indicates the type of SDF file, i.e. a FIELDATA 'P' indicates an SDF PRINT\$ (type 'P') file, a FIELDATA 'S' (type 'S') indicates a general symbolic file. The type of file determines the type of data records contained within the file and how the control word of the data records must be interpreted.

### **Sdf Data Records**

---

An SDF data record is presumed when S1 of the control word does not indicate a control record. The number of data words associated with a data record is determined by T1 of the data record's control word. If the SDF file is a type 'P' file, then T2 of the control word indicates the amount of spacing that should be done prior to printing the data, if any, associated with the current data record. Also S6 of the control word of a type 'P' data record indicates the code type of the current record, i.e. ASCII, FILEDATA, EBCDIC, etc. Other types of files have different rules on how to interpret the control word of data records.

## E.3 MAKERPT2200 Overview

---

MAKERPT2200 (pronounced "make-report 2200") is a OS2200 based processor that compiles a text based input script into one or more "F4N" encoded binary OMNIBUS element files that are used by LPD2200 for inserting printer initialization and reset sequences. LPD2200, the non-terminating batch job supplied with Spin-X/LPR-LPD references the "F4N" files created by MAKERPT2200 within MACRO statements in the \*PARAM file.

The MAKERPT2200 processor accepts symbolic images from the runstream or an SDF file or file.element. The images are expected to contain text records oriented to the rules imposed by the DOS based MAKERPT.EXE utility, however MAKERPT2200 does not implement the procedures of the PC Makerpt. The full documentation for the DOS based MAKERPT.EXE is in Appendix D of this manual .

### E.3.1 "F4N" files

---

When a BINARY mode FTP file transfer is made from an MSDOS or UNIX system to the Unisys, an SDF program file or OMNIBUS element is generated by the Unisys TAS FTP server. We call these "F4N" files because they contain two SDF 050 label control records, the second of which contains one word of fielddata text "\*F4N\*". The rest of a "F4N" file then alternates between SDF line number control records and SDF data records. Each SDF data record contains the original MSDOS or UNIX data stored by a technique similar to Unisys's Block Multiplexer Data Format 'C' where nine octets (i.e nine eight-bit bytes) are packed into two 36-bit words. Because "F4N" files represent 8-bit data in a non-traditional way (format 'C' packed) they are usually unusable for typical OS2200 purposes. But the LPD2200 MACRO processing feature understands how to read and extract the 8-bit data contained within the "F4N" formatted OMNIBUS elements.

Although the "F4N" files follow the record oriented structure imposed by SDF files, the data contained within "F4N" files is stored independently of the SDF record structure. A typical "F4N" file may contain any number of "F4N" data records. Although each "F4N" data record typically contains 1600 words of data (each of which represents 7200 octets of packed 8-bit data), it is the SDF control word associated with each data record that actually indicates how many significant octets of data are contained within each SDF data record. This allows the "F4N" file structure to indicate octet level granularity instead of the more traditional word level granularity provided by other SDF file structures.

Because it is often desirable for LPD2200 to insert printer initialization and reset sequences within outbound print jobs, and because these same sequences usually require octet level granularity, and because we had already developed MSDOS based tools (i.e. MAKERPT.EXE) to build these printer initialization and reset sequences, the "F4N" file format became a natural choice for storing files on the OS2200. The MSDOS based MAKERPT.EXE file can be used to generate the desired printer initialization and reset sequences. The files generated by MAKERPT.EXE can then be uploaded to an OS2200 system using a BINARY mode upload. The resulting "F4N" files can then be used directly by LPD220.

We supply MAKERPT2200, an OS2200 based processor that implements a subset of the features available within the MSDOS MAKEPRT.EXE utility for those sites that do not have FTP services available on their OS2200 system or for those that prefer to work with their MAKERPT script files on their OS2200 system instead of depending on access to a MSDOS system to manage their printer initialization and reset sequences.

---

### E.3.2 Do I need to edit my DOS based script files for MAKERPT2200?

---

Generally speaking, the same input script used for the MSDOS MAKERPT.EXE typically may be used by MAKERPT2200. This allows one to use the more convenient text editing tools available on MSDOS systems to generate and edit a script file. The script file can then be tested using MAKERPT.EXE to verify their are no syntax or compilation errors. Once the script file has been edited to one's satisfaction, the it can be uploaded to the OS2200 using any convenient text mode file transfer technique available (i.e. ASCII mode FTP file transfer). Alternately the script file can be created and edited entirely on the OS2200 system using standard OS2200 text editing tools such as @IPF or @CTS.

Because of their rather limited nature, NONE of the MAKERPT "procedures" available for the MSDOS based MAKERPT.EXE have been implemented within MAKERPT2200, this includes the !INCLUDE, !PAUSE, !PERFORM and !TRANSLATE procedures.

---

### E.3.3 Operating tips

---

Usually the MAKERPT2200 processor is executed directly by a user in demand mode. The script file name is presented as the first parameter, and the output file name where the OMNIBUS elements are to be placed is presented as the second parameter. The input script file can be either be a simple OS2200 SDF data file, or a symbolic element from an SDF program file. The output file name must represent a program file. MAKERPT2200 will process each @name..@end block in the input file in order to create each of the the OMNIBUS elements within the output program file. Although the program file name used for the output files can be the same program file that the input script is stored in, it is recommended that the output file be used to contain only the OMNIBUS elements created by MAKERPT2200.

The recommendation to use an alternate program file for the output file has to do with the algorithm used by MAKERPT2200 to generate the OMNIBUS elements. To minimize the mass storage required for for each OMNIBUS element, MAKERPT2200 initially generates each output element as a SYMBOLIC element. After successfully generating a SYMBOLIC element, MAKERPT2200 then adds a OMNIBUS element entry (using PSF\$) to the program file's table of contents. This new OMNIBUS element entry points to same data location as the SYMBOLIC element entry. The SYMBOLIC element entry is then deleted. A report generated by an @PRT,TL of the output program file will illustrate that each OMNIBUS element is preceeded by a deleted SYMBOLIC element entry that points to the same data location. A subsequent @PACK of the output data file will succesfully purge the deleted SYMBOLIC element entries without purging the data whose location is also referenced by undeleted OMNIBUS element entries. Although we have confidence in the technique of initially generating the data as SYMBOLIC we have in rare cases modified the program file's table-of-contents in such a way that the program file could NOT be packed (typically after using @TOCED). In this case we were able to recover the file successfully by doing an @PACK,S or an @PACK,O or by doing an @COPY,S or @COPY,O of the original program file to an alternate program file.

The MAKERPT2200 version has been enhanced so that it will accept file blocks delimited with "#filename"/"#end" tokens in addition to (or instead of) the original "@filename"/"@end" tokens used by the MSDOS based MAKERPT.EXE. The ability to use of the "#" character instead of the "@" character should help users on OS2200 systems where "@" character in column 1 generally has special meaning.

MAKERPT2200 will accept and translate into "element/version" format any filename specified in typical MSDOS "name.ext" format. Use of extended MSDOS style file names (i.e. drive:\path1\path2\name\ext) is not supported and behaviour of MAKERPT2200 when attempting to process such extended style DOS names is undefined. MAKERPT2200 will also accept a subset of what appear to be standard style Unisys file names but will map them to the element/version names as documented below. Extraneous components of Unisys style file names are not allowed and will cause MAKERPT2200 to generate the error "Unexpected components for element name".

filename.	==> element
element	==> element
.element	==> element
filename.element	==> element/version
element/version	==> element/version
.element/version	==> element/version

---

### E.3.4 Processor Invocation:

---

@*[qualifier]*\*UTILITY.MAKERPT2200[*,options*] infilename.elementname, outfilename

..where filename is either an SDF file or a symbolic filename.element or element name.

---

### E.3.5 Options

---

MAKERPT2200 accepts several call options. A short summary of all available call options can be displayed by entering:

```
@LPD2R1*UTILITY.MAKERPT2200,H
```

assuming that LPD2R1 is the qualifier given during the installation process.

Options:

- D Display compile time debugging information.
- I Input is from the runstream; field1 is ignored.
- L A listing of the input file is written to the standard output file.
- S A Syntax check is done of the input stream. No output files are created.
- V Generate "verbose" remark messages to the standard runstream.

Examples:

1. @[qualifier]\*UTILITY.MAKERPT2200 setup.pcl/dot,setup.

The PCL/DAT element in the SETUP file is processed. The resultant omnibus elements are written to the SETUP file.

2. @[qualifier]\*UTILITY.MAKERPT2200,L TPF\$.X

Element TPF\$.X is read by MAKERPT2200. A listing of TPF\$.X is printed.

3. @[qualifier]\*UTILITY.MAKERPT2200,I  
abc  
xyz  
@eof

Images are read from the runstream. No Listing is printed.