ROPES TCP/IP Printing Support

Introduction

ROPES Version 7.0 introduced the ability to print ROPES reports directly to network attached printers or print servers using the TCP/IP LPR / LPD (Line Printer Requester/Line Printer Daemon) protocols. Using this capability, your enterprise can now print ROPES reports using your LAN attached devices such as network printers with LPD services built into them, printers attached to print servers such as Hewlett Packard JetDirect devices or server-based print queues on Windows, UNIX or OS platforms - in fact, to any platform that can serve as a print server using the LPR / LPD protocols.

Features

The TCP/IP printing facility in ROPES offers these features:

- Interactive configuration of the ROPES Printer TCP/IP controls.
- Print to IP addresses or use your DNS for address resolution.
- Use well-defined host system TCP/IP services and CICS Sockets facilities.
- Flexible definition of forms overlays and other device-specific controls.
- Easily allows the staged deployment of these features without disrupting your existing production environment.
- Makes the best use of your existing devices and their capabilities.
- Electronic forms images means you don’t print or stock special forms, and you can print a document on it’s “form” at any location, even if the form is not available at that site.

Benefits

Your enterprise can benefit from this ROPES facility in many ways:

- You continue to use your existing applications. Your applications that create data for ROPES to spool and print do not need to be changed in any way. You have no additional investment in application programming or testing to use this feature.
- You can replace your more expensive SNA/SDLC printers, interfaces, controllers and networks with much less expensive devices while, at the same time, add new printers and locations by using the hardware that is already their and already on your LAN.
- You can replace expensive wiring systems with less expense LAN cabling, and in some cases, when appropriate, you can use the public internet backbone for transport of the data.
- You can replace preprinted forms with forms images that ROPES will print, along with your data, on plain paper.
- You can make better use of the capabilities of your printers to save transmission time by having the device perform multiple copy creation from one transmission of the data, or save paper by duplexing your output (print on both sides), or manipulate other functions of the device, as desired.

Setup

In order to use the facilities described in this document you will need to insure that the following minimum configuration is met in your environment.

CICS
CICS Sockets support is required for ROPES LPR support.

TCP/IP
The standard TCP/IP stack is required. If you wish to refer to your printers by a name rather than by their IP address, you will also need to configure a Domain Name Server (DNS) or HOSTS configuration file so that these names may be resolved automatically to the printer IP address.

ROPES
ROPES Version 7.0 is required for minimal LPR support. Version 8.1 (Version 8.0 with cumulative fix 1 applied) is required for the full Extended Device Controls as described here.

MVS, OS/390 and z/OS
No additional operating system services are required.

Defining Printers

ROPES Printers are defined using the ROMT transaction. The transaction allows you to define many of the attributes of the ROPES printer. For TCP/IP support you must specify a Device Block Suffix (DCB) which names a ROPES Device Control Block that has been coded to select LPR device support, as described below. The ROPESDEV macro instruction, used to define the Device Control Blocks, is fully documented in the ROPES Administrator's Guide in the chapter titled "Customizing ROPES for Your Installation."

ROPES Device Control Blocks
The following parameters of the ROPES Device Control Block associated with the printer you are defining are required or suggested for proper operation.

On the ROPESDEV TYPE=CONTROLS macro:
DEVTYPE=LPRD is required to specify LPR device support

The ROPESDEV TYPE=SOCKETS statement is suggested to establish the Sockets time out value to prevent "blocked" read and write requests

If you are using the Extended Device Controls to manage the formatting of reports on your device you will need to code the ROPESDEV TYPE=RULES statement to define the rules to be used. You will also need to code one or more ROPESDEV TYPE=DEVINIT statements to define the data streams to be sent to the printer at the stat of printing or at the start of a report. Likewise, ROPESDEV TYPE=DEVEXIT statements may be needed to specify the data streams to be sent to the printer at the end of a report or at the end of printing. These control statements are discussed in more detail later in this document.

The LPR Options File
This file must be created and populated with the records defining the TCP/IP and LPR options to be used when processing a specific printer, or a specific report on a printer. You create the file (a VSAM KSDS) during ROPES installation, using the instructions provided. The contents of the file are maintained using the maintenance transaction ROLI. The LPR Options File maintenance is described in detail in the ROPES Administrator's Guide, in the "Maintenance" Chapter.

Flexibility has been provided in this file to allow you to define specific entries to control the printing of specific reports at particular printers, and to provide a default action when no report specific settings are required.

Some of the more important options you will be specifying in this file are the values that control the following:

- Flags:
  - Quick Port Release (Y|N)
  - Send CF (Control File) First (Y|N)
  - Delete Source File (Y|N)
  - Print Banner (Y|N)
  - Mail Response (Y|N)
  - Logging (Y|N)
  - Type of LPR Print request (Y|N)
- Values:
  - Banner Class
  - TCP/IP Task Name
  - LPD Port number
  - Log TD Queue Name
  - Maximum Print Width
  - Sending Host name
  - LPD Host At (IP or DNS defining the printer)
  - Printer Queue name
  - Job name
  - Title
  - Source File name
  - Times Roman Font Type and File Name

You will create these entries as required to define the attributes of your ROPES LPR printers in your network.

Defining Reports

ROPES Reports are defined using the ROMT transaction. The transaction allows you to define many of the attributes of the ROPES report. For TCP/IP using the Extended Device Support features you must specify a Forms Control Block Suffix (FCB) which names a ROPES Forms Control Block that has been coded to specify the required Extended Device Controls, as described below. The ROPESFRM macro instruction, used to define the Forms Control Blocks, is fully documented in the ROPES Administrator's Guide in the chapter titled "Customizing ROPES for Your Installation."

ROPES Forms Control Blocks

The following parameters of the ROPES Forms Control Block associated with the report you are defining are required or suggested for proper operation.

If you are using the Extended Device Controls to manage the formatting of reports on your device you will need to code one or more ROPESFRM TYPE=DEVINIT statements to define the data streams to be sent to the printer at the stat of printing or at the start of a report. Likewise, ROPESFRM TYPE=DEVEXIT statements may be needed to specify the data streams to be sent to the printer at the end of a report or at the end of printing. These control statements are discussed in more detail later in this document.

Extended Device Controls Support

The Extended Device Controls Feature permits large or small streams of Printer Control Language device codes to be sent to a ROPES printer. Previously, ROPES had a limited capability to send PCL code strings, and the code sequences could not exceed 254 bytes in length. PCL codes can be sent to ROPES devices using the "X" carriage control character in the first position of the print line. This tells ROPES that the data on the rest of the line is not to be translated, and can be used to send printer control language codes. Another facility for sending special printer codes involves using the ROPES escape character. All data following the escape character up to the next escape character, or for the remainder of the line is sent ASIS.

Using the ROPES Extended Device Controls Feature, the PCL code sequences can be unlimited in size.

Any number of PCL code sequences may be established during device initialization and device reset or termination. Large sequences can be broken down into smaller sequences and labeled and used accordingly. Other compatible devices can reuse PCL sequences.

PCL code sequences are stored in tables. In a future release of this feature, PCL codes may also be stored on one or more “PCL” files. The PCL table is a simple module which contains a fullword pointer to the beginning of the PCL code, a fullword containing the size of the PCL code, a descriptive area which can be variable in size, and the PCL code area which can be virtually unlimited in size. The PCL code table must be generated manually. However, any program capable of producing PCL code, which often includes word processing type programs, may generate the PCL code data itself. The PCL code will then require some reformatting and possibly filtering to put it in PCL code table form.

Individual PCL code tables are identified and loaded based on two new macro statements in the ROPES Device Characteristics Block (DCB). The two new macro statements are TYPE=DEVINIT and TYPE=DEVEXIT. Each DEVINIT and DEVEXIT statement identifies a PCL code table, which must be sent to the printer. Please see the ROPES Administrator's Guide under the section about customizing the ROPES DCB for more information on these new macro statements.

Before a PCL code table can be used, it must be created, assembled and link edited, and then defined to CICS as an assembly language module. A sample PCL code table can be found in the ROPES distribution source library in member name BOX. This member is provided as a sample for examination, and is not really intended for use. However, for purposes of testing the Extended Device Controls feature, it can be used as a test overlay.

The format of a PCL code table can be seen in the sample code below:

```
BOX      CSECT
BOX      AMODE ANY
BOX      RMODE ANY
         DC    AL4(BEGIN)        *POINTER TO BEGINNING OF PCL CODE
         DC    AL4(ENDING-BEGIN) *LENGTH OF PCL CODE DATA
         DC    CL8'BOX     '     *NAME OF PCL CODE TABLE
         DC    C'&SYSDATE'       *DATE OF MODULE ASSEMBLY
BEGIN    EQU   *
         DC    X'1B451B266C32613068316F307331581B'
         DC    X'26663179307830531B2A7230461B391B'
*
*     (PCL code data continues)
*
         DC    X'1B266C3545'
ENDING   EQU   *
         END   BEGIN
```

The first 4 byte (fullword) field of the module points to the beginning of the PCL code data. The second 4 byte (fullword) field contains the length of all the PCL code data. The next two fields are descriptive and provide the module name and assembly date of the PCL code table. The descriptive fields can be variable in size, so more descriptive information or fields may be added before label “BEGIN” if so desired without affecting the functionality of the module.

Generation and creation of PCL code tables is a manual process that must be performed by the user. In some cases, word processing tools may be used to generate PCL code data, which can then be reformatted into the assembler module format described above. In other cases, the PCL code data may have to be created by an entirely manual process.

Forms

A specific use of the PCL code tables you should consider is the preparation of forms overlays to allow you to dynamically print the components of a preprinted standard form onto plain paper at the time that your report is being printed. This capability will allow you to save on the cost of printing and stocking special forms, and give you the operational flexibility to print the reports on any available printer, even if the forms are not stocked at that printer’s location.

We suggest that you create your forms in a word processor or other tool on a personal computer which offers you the ability to print to a PCL device. We have created forms from documents in Microsoft Word, Corel WordPerfect, and from Adobe Acrobat PDF files.

Once the form has been designed, you will need to create the PCL code table. To do this you will follow these steps:

1. Define your output device as a PC file.

2. Define your output device as a printer with PCL support, such as an HP LaserJet or the equivalent.

3. Print your “form” to this file.

4. Convert this file to an “overlay”. An overlay is a specific type of HP PCL macro instruction which causes the same data to be printed on every page. We provide a program that will allow you to do this. You process the file created by your word processor, producing a file which is valid as an overlay in a PCL printer.

5. Convert the overlay file to the appropriate Assembler language source code. We provide a program that will do this for you.

6. Upload the generated assembler source code, compile and link edit this module to a CICS DFHRPL data set and add a PPT entry to define the module to CICS.

You should create the PCL code tables you will be using before you reference them in your ROPES Device Control Blocks and Forms Control Blocks.

In addition to the ability to create long code sequences, the use of PCL code tables allows you to also use the override capabilities of the Extended Device Controls. These overrides allow you to associate certain parameters from the ROPES Forms Control Block (Characters Per Inch, Lines Per Inch, print orientation), and logically apply the Report Copy Count from the report definition or the Copy Count Override from the Printer Definition, and use this information to control the printing of your reports using the printers native capabilities to format and print the data and at the same time allow you adapt the report format and the number of copies to the individual end user location requirements while minimizing the bandwidth demands on your network

The definition and use of Extended Device Controls is documented in great detail in the ROPES Programmer’s Guide in the “Extended Device Controls Feature” chapter.

In the final section of this document we will illustrate, with specific examples, the way that you can use the Extended Device Controls Feature to print forms, control report printing, and use the override capability.

Putting It Together

In this section of the document we will present a series of real world scenarios to illustrate the use of the Extended Device Controls, singly and in combination. We will illustrate these capabilities:

Page orientation
Boxes and Shading
Forms Overlay
Copies Overrides by Location

In this section we will be using the following device control block for all of our examples:

```
①          ROPESID  MODE=ONLINE
```

```
②          ROPESDEV TYPE=INITIAL,NAME=LPRB
③          ROPESDEV TYPE=SPACING,                                          X
                 BUFSIZE=1028,                                             X
                 SINGLE=0D25,NOSPACE=0D
④          ROPESDEV TYPE=CHANNELS,                                         X
                 CH1=0C0D
⑤          ROPESDEV TYPE=CONTROLS,DEVTYPE=LPRD,LINEFMT=V,MAXLINE=255,     X
                 LFATMPP=N,LFATEOT=N,SIGNAL=TS,SCSEJECT=N
⑥          ROPESDEV TYPE=TRANSLATE,TRANTAB=NONE
❶          ROPESDEV TYPE=DEVINIT,DVCNAME=SAMPLAND,DVCKEYN=NONE,           X
                 DVCRTRN=N,DVCSRCE=M,DVCTYPE=OrientSet,DVCKEYN=NONE,      X
                 DVCRULE=ORIENT,DVCOVOF=5,DVCOVLN=1,DVCOVDF=C,            X
                 DVCBUFS=1024,DVCSTAT=Y
❷          ROPESDEV TYPE=DEVINIT,DVCNAME=SAMPLAND,DVCKEYN=NONE,           X
                 DVCRTRN=N,DVCSRCE=M,DVCTYPE=CopySet,DVCKEYN=NONE,        X
                 DVCOVRD=ReportCopies,DVCRULE=CPYRULE,DVCOVOF=24,         X
                 DVCOVLN=2,DVCOVDF=Z,DVCBUFS=1024,DVCSTAT=N
❸          ROPESDEV TYPE=DEVINIT,DVCNAME=LANDBARS,DVCKEYN=NONE,           X
                 DVCRTRN=N,DVCSRCE=M,DVCTYPE=LANDBARS,                    X
                 DVCBUFS=1024,DVCSTAT=N
❹          ROPESDEV TYPE=DEVINIT,DVCNAME=BILLLADG,DVCKEYN=NONE,           X
                 DVCRTRN=N,DVCSRCE=M,DVCTYPE=BILL,                        X
                 DVCBUFS=1024,DVCSTAT=N
❺          ROPESDEV TYPE=DEVINIT,DVCNAME=PORTBARS,DVCKEYN=NONE,           X
                 DVCRTRN=N,DVCSRCE=M,DVCTYPE=PORTBARS,                    X
                 DVCBUFS=1024,DVCSTAT=N
❻          ROPESDEV TYPE=DEVINIT,DVCNAME=PICKLIST,DVCKEYN=NONE,           X
                 DVCRTRN=N,DVCSRCE=M,DVCTYPE=PICK,                        X
                 DVCBUFS=1024,DVCSTAT=N
❼          ROPESDEV TYPE=RULES,RULENAME=ORIENT,RULELST='P,L'
❽          ROPESDEV TYPE=RULES,RULENAME=CPYRULE,                          X
                 RULELST='01,02,03,04,05,06,07,08,09,10'
⑦          ROPESDEV TYPE=FINAL
⑧          END   RO$DLPRB
```

Here is an explaination of each of these statements.

①② These statements are standard in all DCBs. The NAME= value defines the Device Block Suffix, which is appended to RO$D to create the module name used by ROPES. Device Block Suffix is a value on the printer definition panel of the ROMT maintenance transaction.

③④ These statements provide device control values for implementing the ANSI carriage controls for line spacing, channel control spacing, and sets the transmission buffer size for the device.

⑤ This statement indicates that the device is an LPR device (DEVTYPE=LPRD), to be controlled through messages passed in a Temporary Storage Queue (SIGNAL=TS).

⑥ This statement indicates that ROPES is not to try to translate characters which are non-printable on 3270 devices, though data translation from EBCDIC to ASCII will occur prior to transmission.

⑦⑧ These statements are part of the standard termination for a DCB.

```
❶          ROPESDEV TYPE=DEVINIT,DVCNAME=SAMPLAND,DVCKEYN=NONE,           X
                 DVCRTRN=N,DVCSRCE=M,DVCTYPE=OrientSet,DVCKEYN=NONE,      X
                 DVCOVRD=FCBOrient,DVCRULE=ORIENT,DVCOVOF=5,DVCOVLN=1,    X
                 DVCOVDF=C,DVCBUFS=1024,DVCSTAT=N
```

This is a Device Init defintion. It names a device initialization sequence (PCL code table) SAMPLAND (DVCNAME) which is a load module (DVCRSCE) by that name. There is an override rule called ORIENT (DVCRULE), the data field to be overridden occurs in byte 5 (relative to 0) of the data (DVCOVOF), is of length 1 (DVCOVLN), to be controlled by the Orient option in the report's Forms Control Block (DVCOVRD), and is of the type C, to be converted based on the rules for (DVCOVDF), which converts 'P' and 'L' to '0' and '1', respectively. This device code is not used unless the FCB for the report includes a DEVINIT statement naming the DVCTYPE of OrientSet.

❷
```
          ROPESDEV TYPE=DEVINIT,DVCNAME=SAMPLAND,DVCKEYN=NONE,           X
                DVCRTRN=N,DVCSRCE=M,DVCTYPE=CopySet,DVCKEYN=NONE,        X
                DVCOVRD=PrtCopies,DVCRULE=CPYRULE,DVCOVOF=24,            X
                DVCOVLN=2,DVCOVDF=Z,DVCBUFS=1024,DVCSTAT=N
```

This is another Device Init defintion. It names a device initialization sequence (PCL code table) SAMPLAND which is a load module by that name. There is an override rule called CPYRULE, the data field to be overridden occurs in byte 24 (relative to 0) of the data, is of length 2, to be controlled by the Printer Copies override value in the report's printer assignment (making it location dependent), and is of the type Z, to be converted to zoned decimal. This device code is not used unless the FCB for the report includes a DEVINIT statement naming the DVCTYPE of CopySet.

❸
```
          ROPESDEV TYPE=DEVINIT,DVCNAME=LANDBARS,DVCKEYN=NONE,           X
                DVCRTRN=N,DVCSRCE=M,DVCTYPE=LANDBARS,                    X
                DVCBUFS=1024,DVCSTAT=N
```

This is a Device Init definition. It names a device initialization sequence LANDBARS which is a load module by that name. This device code is not used unless the FCB for the report includes a DEVINIT statement naming the DVCTYPE of LANDBARS.

❹
```
          ROPESDEV TYPE=DEVINIT,DVCNAME=BILLLADG,DVCKEYN=NONE,           X
                DVCRTRN=N,DVCSRCE=M,DVCTYPE=BILL,                        X
                DVCBUFS=1024,DVCSTAT=N
```

This is a Device Init definition. It names a device initialization sequence BILLLADG which is a load module by that name. This device code is not used unless the FCB for the report includes a DEVINIT statement naming the DVCTYPE of BILL.

❺
```
          ROPESDEV TYPE=DEVINIT,DVCNAME=PORTBARS,DVCKEYN=NONE,           X
                DVCRTRN=N,DVCSRCE=M,DVCTYPE=PORTBARS,                    X
                DVCBUFS=1024,DVCSTAT=N
```

This is a Device Init definition. It names a device initialization sequence PORTBARS which is a load module by that name. This device code is not used unless the FCB for the report includes a DEVINIT statement naming the DVCTYPE of PORTBARS.

❻
```
          ROPESDEV TYPE=DEVINIT,DVCNAME=PICKLIST,DVCKEYN=NONE,           X
                DVCRTRN=N,DVCSRCE=M,DVCTYPE=PICK,                        X
                DVCBUFS=1024,DVCSTAT=N
```

This is a Device Init definition. It names a device initialization sequence PICKLIST which is a load module by that name. This device code is not used unless the FCB for the report includes a DEVINIT statement naming the DVCTYPE of PICK.

❼
```
          ROPESDEV TYPE=RULES,RULENAME=ORIENT,RULELST='P,L'
```

This is a RULES statement that defines the valid values for the input to the override which names this rule in it's DEVINIT or DEVEXIT statement.

❽
```
          ROPESDEV TYPE=RULES,RULENAME=CPYRULE,                          X
                RULELST='01,02,03,04,05,06,07,08,09,10'
```

This is a RULES statement that defines the valid values for the input to the override which names this rule in it's DEVINIT or DEVEXIT statement. In this case, the rule will apply to the number of copies required, and we have limited the choices to values from 1 -10.

As you can see, you may define as many PCL code tables as you want in the Device Control Block definition. When a printer uses this Device Control Block, all of the PCL code tables are available to any report printed on that printer. Think of it as declaring that, in the case of a form, that the form is available at any printer that uses the Device Control Block.

This is the PCL code table called SAMPLAND which is referenced in the Device Control Block. This is assembled and linked into a load module library in your CICS DFHRPL concatenation, and a CICS PPT (CEDA PROGRAM) entry is required.

The data in this PCL code table consists of five PCL commands:

X'1B45'	= EcE (Escape E) and is the RESET command
X'1B266C314F'	= Ec&l1O and is the Landscape orientation command
X'1B287331302E303048'	= Ec(s10.00H and sets the CPI to 10.00
X'1B266C3844'	= Ec&l8D and set the LPI to 8
X'1B266C303158'	= Ec&l01X and sets the Copy Count to 1

You can find the codes supported by your printer in the appropriate PCL manuals, which are widely available.

```
SAMPLAND CSECT
SAMPLAND AMODE ANY
SAMPLAND RMODE ANY
         DC AL4(BEGIN)              Beginning Of Data Area
         DC AL4(ENDING-BEGIN)       Length Of Data Area
         DC C'SAMPLAND'             Module Name
         DC C'&sysdate'             Module Date
BEGIN    EQU *
         DC X'1B451B266C314F1B287331302E303048'
         DC X'1B266C38441B266C303158'
ENDING   EQU *
         END BEGIN
```

Page Orientation

Page orientation in ROPES can be controlled using the FCB Orient option, which may be assigned the value P(ortrait) or L(andscape).

Now, let's assume that you have coded the PCL code table shown above, which provides the necessary controls for implementing the printer's orientation (and some other settings as well).

This code includes the sequence "1B266C314F" or Ec&l1O, which will set orientation to 1, or landscape. A value of 0 would set orientation to portrait.

We use the Device Block defined above, and we activate this override by specifying it's use in the FCBs which are associated with the reports that require this setting. Here is an example:

```
  RO$FSAMP TITLE 'RO$FSAMP - SAMPLE FORMS CONTROL BLOCK'
           ROPESID MODE=ONLINE
           ROPESFRM TYPE=INITIAL,NAME=SAMP
           ROPESFRM TYPE=SPACING,DEPTH=66,OVFLOW=66
           ROPESFRM TYPE=CONTROL,ORIENT=L
①          ROPESFRM TYPE=DEVINIT,DVCTYPE=OrientSet,DVCUSEF=Y
           ROPESFRM TYPE=FINAL
           END   RO$FSAMP
```

The important line here is ①. On this line, we indicate that the DCB DVCTYPE named OrientSet is to be used with this report (DVCUSEF=Y). The orientation setting for any report using this Forms Control Block, when printed on a printer using the Device Control Block we described above, will be set to the value in this Forms Control Block (ORIENT=L)

Boxes and Shading

You can use the built in features of the PCL language to draw lines, create boxes, and apply shading to areas on your documents.

Now, let's assume that you have coded the following PCL code table, which provides the necessary controls for drawing a box border around the page.

```
BOX      CSECT
BOX      AMODE ANY
BOX      RMODE ANY
         DC AL4(BEGIN)             Beginning Of Data Area
         DC AL4(ENDING-BEGIN)      Length Of Data Area
         DC C'BOX     '            Module Name
         DC C'&sysdate'            Module Date
BEGIN    EQU *
         DC X'1B451B266C32613068316F307331581B'
         DC X'26663179307830531B2A7230461B266C'
         DC X'31653438461B2A74333030521B2A7039'
         DC X'30581B2A70313031591B2A6334411B2A'
         DC X'6332323931421B2A6330501B2A633239'
         DC X'3439411B2A6334421B2A6330501B2A70'
         DC X'33303336581B2A6334411B2A63323239'
         DC X'31421B2A6330501B2A703930581B2A70'
         DC X'32333839591B2A6332393439411B2A63'
         DC X'34421B2A6330501B2838551B28733374'
         DC X'7331306862501B266631733178313078'
         DC X'34581B266C342E30431B287330703073'
         DC X'306231362E366834303939541B266131'
         DC X'304C1B266C313045'
ENDING   EQU *
         END BEGIN
```

Breaking the PCL down is beyond the scope of this document. Besides, we did not code this by hand! This PCL was built by drawing a box in a Corel WordPerfect document and then printing it to a file with a PCL printer definition. Then we converted it to an overlay, and then to assebler code, with software we will provide to you.

We use the Device Block defined above, and we activate this override by specifying it's use in the FCBs which are associated with the reports that require this setting. Here is an example:

```
RO$FSAMP TITLE 'RO$FSAMP - SAMPLE FORMS CONTROL BLOCK'
         ROPESID MODE=ONLINE
         ROPESFRM TYPE=INITIAL,NAME=SAMP
         ROPESFRM TYPE=SPACING,DEPTH=66,OVFLOW=66
         ROPESFRM TYPE=CONTROL,ORIENT=L
         ROPESFRM TYPE=DEVINIT,DVCTYPE=OrientSet,DVCUSEF=Y
①        ROPESFRM TYPE=DEVINIT,DVCTYPE=BOX,DVCUSEF=Y
         ROPESFRM TYPE=FINAL
         END   RO$FSAMP
```

The important line here is ①. On this line, we indicate that the DCB DVCTYPE named BOX is to be used with this report (DVCUSEF=Y). The PCL code table defined as BOX is transmitted to the printer first, and is installed in the printer as an overlay, which means, simply, that it prints on every page. The text of the report is also sent (in landscape orientation) and it appears on the pages which now have a box border. The box and the text will not interfere with each other. There is an example of this output later in this document.

Forms Overlay

You can use the features of sophisticated word processors to produce much more complex forms to be used with your reports.

Now, let's assume that you have built PCL code tables named PICKLIST, BILLLADG and LANDBARS. These are much to large to include here in the text, but they, too, where built using a tool, and not by hand. These were printed from Adobe Acrobat PDF files from formst that were designed and built with a forms design tool.

The following FCB's illustrate how each, in turn, could be invoked by the report that needs them.

```
RO$FPICK TITLE 'RO$FSAMP - PICKLIST FORMS CONTROL BLOCK'
         ROPESID MODE=ONLINE
         ROPESFRM TYPE=INITIAL,NAME=PICK
         ROPESFRM TYPE=SPACING,DEPTH=66,OVFLOW=66
         ROPESFRM TYPE=DEVINIT,DVCTYPE=PICK,DVCUSEF=Y
         ROPESFRM TYPE=FINAL
         END   RO$FPICK

RO$FBILL TITLE 'RO$FSAMP - BILL OF LADING FORMS CONTROL BLOCK'
         ROPESID MODE=ONLINE
         ROPESFRM TYPE=INITIAL,NAME=BILL
         ROPESFRM TYPE=SPACING,DEPTH=66,OVFLOW=66
         ROPESFRM TYPE=DEVINIT,DVCTYPE=BILL,DVCUSEF=Y
         ROPESFRM TYPE=FINAL
         END   RO$FBILL

RO$FLBAR TITLE 'RO$FSAMP - LANDSCAPE GREY BAR FORMS CONTROL BLOCK'
         ROPESID MODE=ONLINE
         ROPESFRM TYPE=INITIAL,NAME=LBAR
         ROPESFRM TYPE=SPACING,DEPTH=66,OVFLOW=66
         ROPESFRM TYPE=DEVINIT,DVCTYPE=SAMPLAND,DVCUSEF=N
         ROPESFRM TYPE=DEVINIT,DVCTYPE=LANDBARS,DVCUSEF=Y
         ROPESFRM TYPE=FINAL
         END   RO$FLBAR
```

You will find samples of the output of these reports later in this document as well.

Copies Overrides by Location

Finally, we will illustrate another very useful feature of the Extended Device Controls feature of ROPES. Let us say, for sake of argument, that the Bill of Lading document we are generating is being printed at more than one location. This is easly done with ROPES. You simply assign the ROPES report to more than one printer.

Now, let us also say, for this example, that at the Shipping Dock, three copies of the Bill of Lading are needed, but only one copy is needed in the Traffic Office.

When you define the report to ROPES you will want to set the report copy count (on the ROMT maintenance panel) to 1. You also assign the report to the printer in the Traffic Office, and to the printer on the Shipping Dock. When you assign the report to the Traffic Office, you do not specifiy a copy override value, but you do specify the value 3 when assigning the report to the Shipping Dock printer.

Using the Device Block we described above, and the following FCB, we can automatically have ROPES send the COPY PCL control value set to 1 for the Traffic Office, so only one copy of the report is printed, on the Bill of Lading form, but 3 copies are printed on the Shipping Dock. Since this is a PCL order to the printer, we only transmit the report and the form data once, and the printer prints each page 3 times.

```
    RO$FBILL TITLE 'RO$FSAMP - BILL OF LADING FORMS CONTROL BLOCK'
             ROPESID MODE=ONLINE
             ROPESFRM TYPE=INITIAL,NAME=BILL
             ROPESFRM TYPE=SPACING,DEPTH=66,OVFLOW=66
①            ROPESFRM TYPE=DEVINIT,DVCTYPE=BILL,DVCUSEF=Y
②            ROPESFRM TYPE=DEVINIT,DVCTYPE=CopySet,DVCUSEF=Y
             ROPESFRM TYPE=FINAL
             END   RO$FBILL
```

In this example, line ① tells ROPES to use the BILL resource for the report, and line ② tells ROPES to use the CopySet resource, which is defined to dynamically set the copy count based on the Printer Copy override value.

Examples of forms

On the following pages, you will see several forms examples. Note that these are scans of documents actually printed by ROPES. The quality of these scans does not reflect the quality of the documents themselves, which are much clearer.

The first example is the BOX overlay, showing line drawing.

The second example shows shading using the LANDBARS overly.

The third example shows an overlay which uses a graphic image (logo), shading and text to create a form for printing Pick Tickets (the PICKLIST overlay).

The fourth example shows an overlay which uses many PCL features to create a complex form. Remember that in this example, the actual report data was blank, so all you are seeing is the form. The actual report data, formatted so that the data is placed in the appropriate boxes, would fill in the form, making the output appear as one complete document. This used to be a preprinted form that had to be loaded into the printer at the appropriate time. This is the BILLLADG overlay.