PDF417 Encoder

Installation Guide

Version 2.2

(Build 2.1.0.4 or later)

Liability

The information in this manual is subject to change without notice and should not be construed as a commitment by NeoMedia Technologies, Inc. NeoMedia Technologies assumes no responsibility for any errors that might appear in this document.

The software described in this document is furnished under a license and may be used or copied only in accordance with the terms of such license.

© Copyright 1998, 1999, 2000, 2002 NeoMedia Technologies, Inc. All rights reserved.

All products mentioned are trademarks or registered trademarks of their respective owners.

PN: 4511202-00-01-00

Revised 3/1/2002

NeoMedia Technologies, Inc.
2201 Second Street, Suite 600
Fort Myers, Florida 33901
Web Address: http//www.neom.com

Office (239) 337-3434
Fax (239) 337-3668

Tech Support Email: support@neom.com

Table of Contents

OVERVIEW

This document provides the installation instructions for the NeoMedia Technologies PDF417 encoder library. For a detailed description of how to use the encoder, refer to the programmer's manual***, PDF417 Programming Guide***.

The PDF417 encoder library is a powerful set of functions for generating and printing PDF417 barcode symbols in a variety of hardware and software environments. The library is written in ANSI C and is shipped in compiled object form. The library includes interfaces for COBOL and C. While the encoder library does not use any of the standard C library functions, some platforms (e.g., MVS and OS/400) still require that programs calling C modules be linked with the C run-time libraries. Refer to your systems development environment reference for more information.

The encoder consists of the following components:

- The encoder library itself (shipped in compiled, object format for your platform).
- A number of printer fonts for IBM AFP, Xerox, and Hewlett Packard PCL printers. Refer to your NeoMedia Technologies, Inc. PDF417 encoder License Agreement for details on which fonts you have licensed for use.
- A number of sample programs and printer resources.

The specific file names of these components are listed in the platform installation sections.

Windows PC-based Installation

It is highly recommended that the installation of the PDF417 encoder be performed by a skilled System Administrator with intimate knowledge of your Windows environment. The instructions that follow are written in a general sense and will work on most Windows based systems. However, due to the wide and varied PC-based configurations available, it is impossible to provide concise instructions for all possible scenarios. Some of the instructions may therefore require customization.

Requirements

The encoder is supported for use with the following products:

- ACUCOBOL
- C/C++

After completing the installation, the PDF417 encoder will be available as a dynamic link library that can then be linked with your applications. While linking and running the encoder with products other than those listed above may be possible, these types of configurations are not supported by NeoMedia Technologies.

Preparing for Installation

The PDF417 encoder is distributed on diskettes in InstallShield format. The distribution consists of a root directory, `pdf417_32bit or pdf417_16bit`, with the following six sub-directories.

The first directory, **`pdf417_32bit\lib`**, contains the PDF417 encoder dynamic link library (.dll), the link library (.lib)), the static link library (.lib), and C header files which will need to be included in your application.

The second directory, **`pdf417_32bit\sample`**, contains a variety of resources for the programmer, including an Excel spreadsheet PDF417 Calculator, sample programs, callback function definition source files, and C header files.

The third directory, **`pdf417_32bit\acu`**, contains the ACUCOBOL specific files required for building an ACUCOBOL runtime and a batch file to compile a COBOL program.

The fourth directory, **`pdf417_32bit\docs`**, contains the user documentation files in Microsoft Word format.

The fifth directory, **`pdf417_32bit\hpfont`**, contains the font binary files for HP PCL printers.

The sixth directory, **`pdf417_32bit\afpos2`**, contains the font binary files for OS2 AFP printers.

Installing the Encoder

The installation media for Windows is in InstallShield format.

- Insert Disk 1 into floppy drive A.
- Run `A:SETUP`.

For 32-bit installation, this process will create the pdf417 directory structure (pdf417_32bit). For the 16-bit installation, this process will create the top level directory (pdf417_16bit) and copy all files into this directory. For the 32-bit installation, once you have installed the PDF417 encoder, you must copy the `pdfenc_d.dll` into a directory where it can be found at execution time. The Windows operating system searches the `C:\WINDOWS` and `C:\WINDOWS\SYSTEM` directories to resolve the link. It will also search the directory where the executable that is using the library is located. If you know how to link manually this method can be used by making sure the DLL file is copied to the specified directory path.

The following tables list the contents of each of the directories:

Contents of the **`pdf417_32bit\lib`** directory:

File Name	Description
pdfenc_d.dll	Encoder (32-bit) dynamic link library (used at run time). **OR**
nmpdfenc.dll	Encoder (16-bit) dynamic link library (used at run time.
pdfenc_d.lib	Encoder (32-bit) link library (used at link time). **OR**
nmpdfenc.lib	Encoder (16-bit) link library (used at link time).
pdfenc_s.lib	Encoder (32-bit) static link library (used at run time). **OR**
llpdfenc.lib	Encoder (16-bit) large memory module static link library
pdfenc.h	C #include file.
pdfdefs.h	C # include file.
pdfrend.h	C #include file.

Contents of the **`pdf417_32bit\sample`** directory:

Object	Description
Readme32.txt	A standard README file (32-bit installation) **OR**
Readme16.txt	A standard README file (16-bit installation).
samplec.c	Sample C program.
samplec.mak	Makefile for compiling the sample C program.
samplcob.cob	Sample COBOL program.
nmpdf417.cob	COPYLIB for sample COBOL program.
sinkfils.c	Sample file for output callback function to a file.
sinkfile.h	C #include file.
sinkmems.c	Sample file for output callback function to memory.
sinkmem.h	C #include file.
PDF417 Calculator.xls	Excel spreadsheet to calculate byte, codeword, and Font character counts for specific real estate Parameters.

Contents of the **`pdf417_32bit\acu`** directory:

Object	Description
compacu.bat	Batch file to compile samplcob.cob.
direct.c	Source file for creating an ACUCOBOL runtime.
wpruncbl.mak	Makefile for creating an ACUCOBOL runtime with PDF417 subroutine included.

Contents of the `pdf417_32bit\hpFont` directory:

Object	Description
HPPP3309.FLJ	Font file for HP PCL printers, portrait, 300 DPI, 3x9 pixel module size
HPPL3309.FLJ	Font file for HP PCL printers, landscape, 300 DPI, 3x9 pixel module size
HPPP3412.FLJ	Font file for HP PCL printers, portrait, 300 DPI, 4x12 pixel module size
HPPL3412.FLJ	Font file for HP PCL printers, portrait, 300 DPI, 4x12 pixel module size

Contents of the `pdf417_32bit\afpOS2` directory:

Object	Description
C0PD2206.OS2	Font character set for OS2 AFP printers, 240 DPI, 2x6 pixel module size
X0PD2206.OS2	Font code font for OS2 AFP printers, 240 DPI, 2x6 pixel module size
C0PD2309.OS2	Font character set for OS2 AFP printers, 240 DPI, 2x9 pixel module size
X0PD2309.OS2	Font code font for OS2 AFP printers, 240 DPI, 2x9 pixel module size
C0PD3309.OS2	Font character set for OS2 AFP printers, 300 DPI, 3x9 pixel module size
X0PD3309.OS2	Font code font for OS2 AFP printers, 300 DPI, 3x9 pixel module size
C0PD3412.OS2	Font character set for OS2 AFP printers, 300 DPI, 4x12 pixel module size
X0PD3412.OS2	Font code font for OS2 AFP printers, 300 DPI, 4x12 pixel module size
T1PDF417.OS2	Code page for 240 and 30 DPI AFP printers

Contents of the `pdf417_32bit\docs` directory:

Object	Description
pdf417igv2.doc	PDF414 Encoder Installation Guide
pdf417v2.doc	PDF417 Encoder Programming Guide
license.txt	The License Agreement in text format

Linking with the Encoder

The precise syntax for linking with the encoder varies from compiler to compiler. Refer to your compiler's documentation for linking with dynamic link libraries. ***NOTE: The PDF417 encoder is shipped as a dynamic link library (.dll) and as a static link library.***

A makefile (`samplec.mak`) has been provided in the `pdf417_32bit\sample` directory for compiling and linking the sample C program. ***NOTE: This makefile assumes that both `NMAKE` and the standard C compiler (`cl`) are being used.***

C/C^{++}

To compile and link the sample program for 'C':

NOTE: Compiling and testing the C sample is not a required step for installation of the PDF417 Encoder. It is recommended if you will be using the C API.

1. Create a test directory to build the sample program in. **Note: for the 16-bit installation, you can build the sample programs in the installation directory.**
2. Change directory to the sample directory just created (`...\test`) and copy the following files to this directory:

   ```
   ...\test> copy ...\pdf417_32bit\lib\pdfenc_d.dll C:\WINDOWS
   ```

   NOTE: The `DLL` can be copied to the `C:\WINDOWS, C:\WINDOWS\SYSTEM` or the path where the executable will reside which is the current directory (`...\test`).

   ```
   ...\test> copy ...\pdf417_32bit\sample\samplec.mak .
   ...\test> copy ...\pdf417_32bit\sample\samplec.c .
   ...\test> copy ...\pdf417_32bit\sample\sinkfils.c .
   ...\test> copy ...\pdf417_32bit\sample\sinkfile.h .
   ```

3. Edit `samplec.mak` and confirm that the include directory path (-I) is correct for your configuration.

   NOTE: This `.mak` file is a Windows make file utilizing NMAKE. It is not a Microsoft Developer Studio build file. It is a text file and can be edited using any word processor or edit utility.

4. Execute the make file:

   ```
   ...\test> nmake /f samplec.mak
   ```

5. This will compile and link to an executable, samplec.exe. Run the following program:

   ```
   ...\test> samplec
   ```

6. For successful execution, there will be no screen I/O but three files will be created in the current directory:

   ```
   ...\test\nmpdfdib.bmp
   ...\test\nmpdftiff.tif
   ...\test\nmpdffnt
   ```

The files that are created from this test program are DIB, TIFF and font character files which contain a PDF417 symbol with the following encoded:

```
This is a test of the NeoMedia Technologies PDF417 Encoder.  The error
correction will encode at level 3 with ECC padding.
```

Within Windows, you should be able to double click on the DIB and TIFF files created. Microsoft Paint program should be able to understand the .BMP format. Microsoft Imaging should be able to understand the .TIF format. To test the symbol, print the file from within the applications named above and then scan them.

For the font character file, you will have to print the file with the appropriate font definition file loaded. Printing the symbol files is beyond the scope of this document.

ACUCOBOL

To compile and link the sample program for ACUCOBOL:

NOTE: Compiling and testing the COBOL sample is not a required step for installation of the PDF417 Encoder. It is recommended if you will be using the COBOL API.

1. Create a runtime which includes ACUCOBOL and PDF417 encoder library (`wpruncbl.exe`):
   - Change directory to the acu directory in the PDF417 distribution, `...\pdf417_32bit\acu`
   - Make sure that the variables are defined correctly in `wpruncbl.mak`
   - Execute the make file:

     ```
     ...\pdf417_32bit\acu> nmake /f wpruncbl.mak
     ```
2. Create a sample directory to build the sample programs in. **Note: for the 16-bit installation, you can build the sample programs in the installation directory.**
3. Change directory to the test directory just created (`...\test`) and copy the following files to this directory:

   ```
   ...\test> copy ...\pdf417_32bit\sample\samplcob.cob .

   ...\test> copy ...\pdf417_32bit\sample\nmpdf417.cob .

   ...\test> copy ...\pdf417_32bit\acu\compacu.bat .
   ```
4. Compile the test program:

   ```
   ...\test> compacu samplcob
   ```

   This command will create a debuggable object file `samplcob`.
5. Copy the dynamic link library (.dll) to the same directory as the runtime executable or into one of the appropriate places that the system will find it at runtime:

   ```
   ...\test> copy ..\pdf417_32bit\lib\pdfenc_d.dll
   ..\pdf417_32bit\acu\pdfenc_d.dll
   ```

   NOTE: The `DLL` can be copied to the `C:\WINDOWS, C:\WINDOWS\SYSTEM` or the path where the ACUCOBOL runtime executable will reside.
6. Run the new runtime with the COBOL object as a parameter:

   ```
   ...\test> ...\acu\wpruncbl  samplcob
   ```
7. For successful execution, there will be no screen I/O but the following file will be created in the current directory:

   ```
   ...\test\pdffont
   ```

The file that is created from this test program is a font character file which contains an array of font characters that, when printed with the appropriate PDF417 symbol font, will render the PDF417 symbol for the following data:

```
This is a test of the NeoMedia Technologies PDF417 Encoder.  The error
correction will encode at level 3 with ECC padding.
```

UNIX Installation

It is highly recommended that the installation of the PDF417 encoder be performed by a skilled System Administrator with intimate knowledge of your UNIX environment. The instructions that follow are written in a general sense and will work on most UNIX systems. However, due to the wide and varied UNIX configurations available, it is impossible to provide concise instructions for all possible scenarios. Some of the instructions may therefore require customization.

Requirements

The encoder is supported for use with the following products:

- C/C++
- ACUCOBOL
- Micro Focus COBOL

After completing the installation, the PDF417 encoder will be available as a static library that can then be linked with your applications. While linking and running the encoder with products other than those listed above may be possible, these types of configurations are not supported by NeoMedia Technologies.

Preparing for Installation

The PDF417 encoder is distributed as a standard tar archive. The archive consists of a root directory, `pdf417`, with the following five sub-directories.

The first directory, **`pdf417/lib`**, contains the PDF417 encoder static library (.a) and C header files which will need to be included in your application.

The second directory, **`pdf417/sample`**, contains a variety of resources for the programmer, including sample programs, callback function definition source files, and C header files.

The third directory, **`pdf417/acu`**, contains the ACUCOBOL specific files required for building an ACUCOBOL runtime and a shell script to compile a COBOL program.

The fourth directory, **`pdf417/mf`**, contains the Micro Focus COBOL specific files required for building a Micro Focus COBOL runtime and a shell script to compile a COBOL program.

The fifth directory, **`pdf417/fontbin`**, contains a directory **`../hp`** which contains the font binary files for HP PCL printers and the directory**`../afpos2`** which contains the font binary files for OS2 AFP printers.

Installing the Encoder

The installation media for UNIX is in tar archive format. Use tar with the appropriate device for your system to extract the files. The following example is for extracting from a floppy disk on an SCO UNIX system:

```
% umask 0
% tar xvf /dev/rfd0 .
```

The following tables list the contents of each of the directories:

Contents of the `pdf417/lib` directory:

File Name	Description
libpdfenc.a	Encoder static link library.
pdfenc.h	C #include file.
pdfdefs.h	C #include file.
pdfrend.h	C #include file.

Contents of the `pdf417/sample` directory:

Object	Description
readmeunix.txt	A standard README file.
samplec.c	Sample C program.
makesam.umf	Makefile for compiling the sample C program.
samplcob.cob	Sample COBOL program.
nmpdf417.cob	COPYLIB for sample COBOL program.
sinkfils.c	Sample file for output callback function to a file.
sinkfile.h	C #include file.
sinkmems.c	Sample file for output callback function to memory.
sinkmem.h	C #include file.

Contents of the `pdf417/acu` directory:

Object	Description
compacu.sh	Batch file to compile samplcob.cob.
direct.c	Source file for creating an ACUCOBOL runtime.
pruncbl.umf	Makefile for creating an ACUCOBOL runtime with PDF417 subroutines included.
acucobol.include	Makefile include for creating the ACUCOBOL runtime.

Contents of the `pdf417/mf` directory:

Object	Description
compmf.sh	Batch file to compile samplcob.cob.
prunmf.c	Source file for creating a MF COBOL runtime.
prunmf.o	Object to include when building the Micro Focus COBOL runtime.
prunmf.umf	Makefile for creating a Micro Focus COBOL runtime With PDF417 subroutines included.

Contents of the `pdf417/fontbin/hp` directory:

Object	Description
HPPP3309.FLJ	Font file for HP PCL printers, portrait, 300 DPI, 3x9 pixel module size
HPPL3309.FLJ	Font file for HP PCL printers, landscape, 300 DPI, 3x9 pixel module size
HPPP3412.FLJ	Font file for HP PCL printers, portrait, 300 DPI, 4x12 pixel module size
HPPL3412.FLJ	Font file for HP PCL printers, portrait, 300 DPI, 4x12 pixel module size

Contents of the **`pdf417/fontbin/afpos2`** directory:

Object	Description
C0PD2206	Font character set for OS2 AFP printers, 240 DPI, 2x6 pixel module size
X0PD2206	Font code font for OS2 AFP printers, 240 DPI, 2x6 pixel module size
C0PD2309	Font character set for OS2 AFP printers, 240 DPI, 2x9 pixel module size
X0PD2309	Font code font for OS2 AFP printers, 240 DPI, 2x9 pixel module size
C0PD3309	Font character set for OS2 AFP printers, 300 DPI, 3x9 pixel module size
X0PD3309	Font code font for OS2 AFP printers, 300 DPI, 3x9 pixel module size
C0PD3412	Font character set for OS2 AFP printers, 300 DPI, 4x12 pixel module size
X0PD3412	Font code font for OS2 AFP printers, 300 DPI, 4x12 pixel module size
T1PDF417	Code page for 240 and 30 DPI AFP printers

Linking with the Encoder

The precise syntax for linking with the encoder varies from compiler to compiler. Refer to your compiler's documentation for linking with static libraries. ***NOTE: The PDF417 encoder is shipped as a static link library, not as a shared library.***

A makefile (`makesam.umf`) has been provided in the `pdf417/sample` directory for compiling and linking the sample C program. ***NOTE: This makefile assumes that the standard UNIX C compiler (cc) is being used.***

C/C^{++}

To compile and link the sample program for C/C^{++}:

NOTE: Compiling and testing the C sample is not a required step for installation of the PDF417 Encoder. It is recommended if you will be using the C API.

1. Change directory to the sample directory in the PDF417 distribution, `.../sample`.
2. Edit `makesam.umf` and confirm that the include directory path (-I) is correct for your configuration.
3. Execute the following make file:

   ```
   $ make -f makesam.umf
   ```

4. This will compile and link to an executable, samplec. Run the following program:

   ```
   $ samplec
   ```

5. For successful execution, there will be no screen I/O but the following three files will be created in the current directory:

   ```
   .../sample/pdfdib00.bmp
   .../sample/pdftiff00.tif
   .../sample/nmpdffnt
   ```

The files that are created from this test program are DIB, TIFF and font character files which contain a PDF417 symbol with the following encoded:

```
This is a test of the NeoMedia Technologies PDF417 Encoder. The error
correction will encode at level 3 with ECC padding.
```

The files created are Windows based file formats. Export these two files to a Windows environment. Within Windows, you should be able to double click on the files. Microsoft Paint program should be able to understand the .BMP format. Microsoft Imaging should be able to understand the .TIF format. To test the symbol, you can print the file from within the applications named above and then scan them.

For the character font file, you will have to print the file with the appropriate font definition file loaded. Printing the symbol files is beyond the scope of this document.

ACUCOBOL

To compile and link the sample program for ACUCOBOL:

NOTE: Compiling and testing the COBOL sample is not a required step for installation of the PDF417 Encoder. It is recommended if you will be using the COBOL API.

1. Create a runtime which includes ACUCOBOL and PDF417 encoder library (`pruncbl`):
   - Change directory to the acu directory in the PDF417 distribution, `.../pdf417/acu`
   - Make sure that the variables are defined correctly in `pruncbl.umf`
   - Execute the make file:

     ```
     $ make -f pruncbl.umf
     ```
2. Change directory to the sample directory (`.../sample`) and copy the following files to this directory:

   ```
   $ cp …\pdf417\acu\compacu.sh .
   ```
3. Compile the following test program:

   ```
   $ compacu.sh samplcob
   ```

   This command will create a debuggable ACUCOBOL object file `samplcob`.
4. Run the new runtime with the ACUCOBOL object as a parameter:

   ```
   $ …/acu/pruncbl  samplcob
   ```
5. For successful execution, there will be no screen I/O but one file will be created in the current directory:

   ```
   …/sample/pdffont
   ```

The file that is created from this test program is a font character file which contains an array of font characters that when printed with the appropriate PDF417 symbol font will render the PDF417 symbol for the following data:

```
This is a test of the NeoMedia Technologies PDF417 Encoder.  The error
correction will encode at level 3 with ECC padding.
```

Micro Focus COBOL

To compile and link the sample program for Micro Focus COBOL:

NOTE: Compiling and testing the COBOL sample is not a required step for installation of the PDF417 Encoder. It is recommended if you will be using the COBOL API.

1. Create a runtime which includes Micro Focus COBOL and PDF417 encoder library (`prunmf`):
   - Change directory to the mf directory in the PDF417 distribution, `.../pdf417/mf`
   - Make sure that the variables are defined correctly in `prunmf.umf`
   - Execute the make file:

     ```
     $ make -f prunmf.umf
     ```
2. Change directory to the sample directory (`.../sample`) and copy the following files to this directory:

   ```
   $ cp …\pdf417\mf\compmf.sh .
   ```
3. Compile the following test program:

   ```
   $ compmf.sh samplcob
   ```

   This command will create two files: an intermediate code file `samplcob.int` and an animation information file `samplcob.idy`
4. Run the new runtime with the intermediate file as a parameter:

   ```
   $ …/mf/prunmf  samplcob
   ```
5. For successful execution, there will be no screen I/O but one file will be created in the current directory:

   ```
   …/sample/pdffont
   ```

The file that is created from this test program is a font character file which contains an array of font characters that when printed with the appropriate PDF417 symbol font will render the PDF417 symbol for the following data:

```
This is a test of the NeoMedia Technologies PDF417 Encoder.  The error
correction will encode at level 3 with ECC padding.
```

MVS Installation

It is highly recommended that the installation of the PDF417 encoder be performed by a skilled System Administrator with intimate knowledge of your MVS environment. The instructions that follow are written in a general sense and will work on most MVS systems. However, due to the wide and varied MVS configurations available, it is impossible to provide concise instructions for all possible scenarios. Some of the JCL may therefore require customization.

Requirements

Under MVS, the PDF417 encoder requires the C Language run-time libraries. These libraries will be present on your system if you have any of the following products installed:

- LE/370 (Language Environment 370)
- C/370 (C run-time libraries)

The encoder is supported for use with the following products:

- COBOL II
- COBOL 370
- IBM COBOL
- C Language

All of the JCL jobs use an address mode of 31 (`AMODE=31`). You will need to edit the JCL if you are using the older address mode of 24 for the LINK step parameters. It is up to the user to determine the settings for your environment and make the appropriate modifications to the provided JCL.

After completing the installation, the PDF417 encoder routines will be available as distinct object modules in a Partitioned Data Set (PDS) object library. Your object modules will need to be linked with this PDS object. Although linking and running the encoder with products other than those listed above may be possible, these types of configurations are not supported by NeoMedia Technologies.

Preparing for Installation

The PDF417 encoder is distributed in IEBCOPY format containing the following three libraries. ***NOTE: The following reference to Vxxxx is the PDF417 encoder version indicator. Replace the xxxx with the appropriate version of the encoder that can be found on your distribution media.***

The first library, **`NEO.PDFRSC.Vxxxx`**, contains a variety of resources for the programmer, including sample programs, AFP examples, JCL examples, and C header files.

The second library, **`NEO.PDFOBJ.Vxxxx`**, contains the C object modules that must be linked with your own objects before being used on your system. When linking the PDFOBJ objects with your own objects, you can either create a new load library or add the new executables into an existing library. This will depend on your environment's needs and conventions.

The third library, **`NEO.PDFFNT.Vxxxx`**, contains fonts for IBM AFP, Xerox, and HP PCL printers. While all fonts are shipped with the NeoMedia Technologies PDF417 encoder, you may not be licensed to use them. Please refer to your NeoMedia Technologies Standard Software

License for specific details on which fonts have been licensed for use. Use of unlicensed fonts is a violation of the Standard Software License and is strictly prohibited.

Installing the Encoder

The installation media for MVS is in IEBCOPY format. As part of the installation process, three libraries must be copied from tape into three PDS. The following tables list the PDS space and DCB requirements, and each PDS contents. ***NOTE: The Vxxxx is the PDF417 encoder version indicator. The label on your distribution will provide the proper version information for this distribution. For any future installation instructions in this document, fill in the xxxx with the appropriate version.***

- Insert the media into the tape drive.
- Submit the restore media JCL that you have written. (See below for a JCL example.)

The tape media contains the following **STD** files:

Library
NEO.PDFRSC.Vxxxx
NEO.PDFOBJ.Vxxxx
NEO.PDFFNT.Vxxxx

The following **PDS** will be created during installation:

Library	Space
NEOMEDIA.PDF417.LOADLIB	SPACE=(CYL,(1,1,15),RLSE)
NEOMEDIA.PDF417.RSCLIB	SPACE=(CYL,(1,1,15),RLSE)
NEOMEDIA.PDF417.OBJLIB	SPACE=(CYL,(1,1,15),RLSE)
NEOMEDIA.PDF417.FNTLIB	SPACE=(CYL,(1,1,15),RLSE)

DCBs for the above created PDS:

Library	LRECL	BLKSIZE	RECFM	DSORG
NEOMEDIA.PDF417.LOADLIB	N/A	32670	U	PO
NEOMEDIA.PDF417.RSCLIB	80	23440	FB	PO
NEOMEDIA.PDF417.OBJLIB	80	3200	FB	PO
NEOMEDIA.PDF417.FNTLIB	12284	27998	VBM	PO

Contents of the **`NEOMEDIA.PDF417.RSCLIB`**:

Object	Description
README	README text file.
PDFENC	C #include file.
PDFDEFS	C #include file.
PDFREND	C #include file.
IPDFLNK	Sample JCL for linking encoder objects into a load module.

Object	Description
SAMPLEC	Sample C program.
ISAMLEC	Sample JCL for building SAMPLEC.
ISAMPLEC	Sample JCL for building SAMPLEC using the pre-linker.
XSAMPLEC	Sample JCL for executing SAMPLEC.
SINKFILE	C #include file.
SINKFILS	Sample C program.
SINKMEM	C #include file.
SINKMEMS	Sample C program.
SAMPLCOB	Sample COBOL program.
NMPDF417	COPYLIB for sample COBOL program.
ISAMLCB	Sample JCL for building SAMPLCOB.
ISAMPLCB	Sample JCL for building SAMPLCOB using the pre-linker.
XSAMPLCB	Sample JCL for executing SAMPLCOB .

Contents of the **NEOMEDIA.PDF417.OBJLIB**:

Object	Description
PDF417EN	PDF417 Encoder object module.

Contents of the **NEOMEDIA.PDF417.FNTLIB**:

Object	Description
C0PD2206	AFP character set, bounded box, 240 dpi. 2x6 pixel module size
C0PD2309	AFP character set, bounded box, 240 dpi. 3x9 pixel module size
C0PD3309	AFP character set, bounded box, 300 dpi. 3x9 pixel module size
C0PD3412	AFP character set, bounded box, 300 dpi. 4x12 pixel module size
T1PDF417	AFP code page, bounded box.
X0PD2206	AFP coded font, bounded box, 240 dpi. 2x6 pixel module size
X0PD2309	AFP coded font, bounded box, 240 dpi. 3x9 pixel module size
X0PD3309	AFP coded font, bounded box, 300 dpi. 3x9 pixel module size
X0PD3412	AFP coded font, bounded box, 300 dpi. 4x12 pixel module size

The following JCL demonstrates how to restore/copy these files from tape:

NOTE: Change the text in bold/italic to fit your installation.

```
//jobcard here
//INSTALL  PROC V=Vxxxx,TAPEUNIT=TAPE,DISKVOL=SCPMV5
//ALLOC    EXEC PGM=IEFBR14,REGION=4M
//LOADLIB  DD  DSN=NEOMEDIA.PDF417.LOADLIB,DISP=(NEW,CATLG,DELETE),
//   SPACE=(CYL,(1,1,15)),VOL=SER=&DISKVOL,
//   DCB=(RECFM=U,BLKSIZE=32670,DSORG=PO)
//COPYLIB  EXEC PGM=IEBCOPY,REGION=4M,COND=(0,NE,ALLOC)
//SYSOUT   DD SYSOUT=*
```

```
//SYSPRINT DD SYSOUT=*
//SYSUT3   DD SPACE=(CYL,(5,5)),UNIT=SYSDA
//SYSUT4   DD SPACE=(CYL,(5,5)),UNIT=SYSDA
//INDD1    DD DSN=NEO.PDFRSC.&V,UNIT=&TAPEUNIT,DISP=(OLD,KEEP),
//    VOL=(,RETAIN,SER=NEO417),LABEL=(1,SL)
//INDD2    DD DSN=NEO.PDFOBJ.&V,UNIT=&TAPEUNIT,DISP=(OLD,KEEP),
//    VOL=(,RETAIN,SER=NEO417),LABEL=(2,SL)
//INDD3    DD DSN=NEO.PDFFNT.&V,UNIT=&TAPEUNIT,DISP=(OLD,KEEP),
//    VOL=SER=NEO417,LABEL=(3,SL)
//OUTDD1   DD DSN=NEOMEDIA.PDF417.RSCLIB,
//    DISP=(NEW,CATLG,DELETE),
//    VOL=SER=&DISKVOL,SPACE=(CYL,(1,1,15)),
//    DCB=(RECFM=FB,LRECL=80,BLKSIZE=23440,DSORG=PO)
//OUTDD2   DD DSN=NEOMEDIA.PDF417.OBJLIB,
//    DISP=(NEW,CATLG,DELETE),
//    VOL=SER=&DISKVOL,SPACE=(CYL,(1,1,15)),
//    DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200,DSORG=PO)
//OUTDD3   DD DSN=NEOMEDIA.PDF417.FNTLIB,
//    DISP=(NEW,CATLG,DELETE),
//    VOL=SER=&DISKVOL,SPACE=(CYL,(1,1,15)),
//    DCB=(RECFM=VBM,LRECL=12284,BLKSIZE=27998,DSORG=PO)
//SYSIN    DD DUMMY
//         PEND
//INSTALL  EXEC INSTALL,V=Vxxxx,TAPEUNIT=561,DISKVOL=SCPMV5
//COPYLIB.SYSIN DD *
 COPY INDD=INDD1,OUTDD=OUTDD1
 COPY INDD=INDD2,OUTDD=OUTDD2
 COPY INDD=INDD3,OUTDD=OUTDD3
//
```

- The installation of the encoder is now complete. The NEOMEDIA.PDF417.OBJLIB contains the PDF417EN members standing for the ENCODER objects that the sample programs will utilize. These object modules should be made available for the developers so that their programs may be linked with them in order to perform calls to the PDF417 encoder API.

Refer to the section titled *Font Selection* later in this document for information on determining the correct font for your environment.

Linking with the Encoder

The precise syntax for compile, pre-link, and linking with the encoder varies from compiler to compiler. Refer to your system’s documentation for detailed instructions for linking with object PDS.

COBOL

To compile and link the sample program for COBOL:

NOTE: Compiling and testing the COBOL sample is not a required step for installation of the PDF417 Encoder. It is recommended if you will be using the COBOL API.

First, the encoder objects must be linked into a load module. A sample JCL (IPDFLNK) has been provided in the NEOMEDIA.PDF417.RSCLIB PDS for creating the load module PDF417 in NEOMEDIA.PDF417.RSCLIB PDS.

1. Edit 'NEOMEDIA.PDF417.RSCLIB(IPDFLNK)' and SUBMIT it.

Confirm that the JCL is correct for your environment and make any changes as required. Upon completion of this job you will receive a warning: IEW2650I 5102 No module entry point provided. **This is an expected message since the load module has no main()**.

A sample JCL (ISAMLCB) has been provided in the NEOMEDIA.PDF417.RSCLIB PDS for compiling and linking the sample COBOL program SAMPLECOB in the NEOMEDIA.PDF417.RSCLIB PDS.

The ISAMLCB member in the NEOMEDIA.PDF417.RSCLIB PDS uses the IBM supplied cataloged procedure IGYWCL in the step COBCL in order to compile the SAMPLCOB program and link its object with the load module PDF417 and the C run-time library.

The ISAMLCB member in NEOMEDIA.PDF417.RSCLIB is a typical sample that one may adapt to link COBOL programs with the PDF417 load module, and with the C run-time library.

2. Edit 'NEOMEDIA.PDF417.RSCLIB(ISAMLCB)' and SUBMIT it.

Confirm that the JCL is correct for your environment and make any changes as required.

A sample JCL (ISAMPLCB) has been provided in the NEOMEDIA.PDF417.RSCLIB PDS for compiling, pre-linking, and linking the sample COBOL program SAMPLECOB in the NEOMEDIA.PDF417.RSCLIB PDS.

The ISAMPLCB member in the NEOMEDIA.PDF417.RSCLIB PDS uses the IBM supplied cataloged procedure IGYWCPL in the step COBCPL in order to compile the SAMPLCOB program, pre-link and link its object with the PDF417EN and the C run-time library.

The ISAMPLCB member in NEOMEDIA.PDF417.RSCLIB is a typical sample that one may adapt to pre-link and link COBOL programs with the PDF417EN objects, and with the C run-time library.

3. Edit 'NEOMEDIA.PDF417.RSCLIB(ISAMPLCB)' and SUBMIT it.

Confirm that the JCL is correct for your environment and make any changes as required.

Upon completion of the build of SAMPLCOB you can then execute the program.

4. Edit 'NEOMEDIA.PDF417.RSCLIB(XSAMPLCB)' and SUBMIT it.

Confirm that the JCL is correct for your environment and make any changes as required. The XSAMPLCB member in the NEOMEDIA.PDF417.RSCLIB PDS executes the SAMPLCOB executable created above. This program generates an output file via the TPDFFONT DD card.

NOTE: The very first step in the XSAMPLEC has been commented out because there is nothing to be deleted during the first run of this program. If you decided to re-execute this program, then the DELETE step would be needed.

For successful execution, there will be no screen output but one file will be generated as an output file via the TPDFFONT DD card defined in the JCL. The file that is created from this test program is a font character file which contains a PDF417 symbol with the following encoded:

```
This is a test of the NeoMedia Technologies PDF417 Encoder.  The error
correction will encode at level 3 with ECC padding.
```

View this file and confirm that it contains the appropriate font characters. If it does not, you will have to print the file with the appropriate font definition file loaded. Printing the symbol file is beyond the scope of this document.

C Language

To compile and link the sample program for C:

NOTE: Compiling and testing the C sample is not a required step for installation of the PDF417 Encoder. It is recommended if you will be using the C API.

First, the encoder objects must be linked into a load module. A sample JCL (`IPDFLNK`) has been provided in the `NEOMEDIA.PDF417.RSCLIB` PDS for creating the load module `PDF417` in `NEOMEDIA.PDF417.RSCLIB` PDS.

1. Edit ‘`NEOMEDIA.PDF417.RSCLIB(IPDFLNK)'` and SUBMIT it.

Confirm that the JCL is correct for your environment and make any changes as required. Upon completion of this job you will receive a warning: IEW2650I 5102 No module entry point provided. **This is an expected message since the load module has no main()**.

A sample JCL (`ISAMLEC`) has been provided in the `NEOMEDIA.PDF417.RSCLIB` PDS for compiling and linking the sample C program `SAMPLEC` in the `NEOMEDIA.PDF417.RSCLIB` PDS.

The `ISAMLEC` member in the `NEOMEDIA.PDF417.RSCLIB` PDS uses the IBM supplied cataloged procedure EDCC in the steps `SINKFILS` and `SAMPLEC` in order to compile C code and store the resulting objects into an object library. Then, in the LINK step, it uses the IBM supplied cataloged procedure EDCL, in order to link the `SINKFILS` and `SAMPLEC` objects with the `PDF417` load module and the C Run-time library. In most cases the C Run-time library is the SCEELKED.

2. Edit ‘`NEOMEDIA.PDF417.RSCLIB(ISAMLEC)'` and SUBMIT it.

Confirm that the JCL is correct for your environment and make any changes as required.

A sample JCL (`ISAMPLEC`) has also been provided in the `NEOMEDIA.PDF417.RSCLIB` PDS for compiling, pre-linking, and linking the sample C program `SAMPLEC` in the `NEOMEDIA.PDF417.RSCLIB` PDS.

The `ISAMPLEC` member in the `NEOMEDIA.PDF417.RSCLIB` PDS uses the IBM supplied cataloged procedure EDCC in the steps `SINKFILS` and `SAMPLEC` in order to compile C code and store the resulting objects into an object library. Then, in the LINK step, it uses the IBM supplied cataloged procedure EDCPL, in order to pre-link and link the `SINKFILS` and `SAMPLEC` objects with the `PDF417EN` and the C Run-time library. In most cases the C Run-time library is the SCEELKED.

The `ISAMPLEC` member in `NEOMEDIA.PDF417.RSCLIB` is a typical sample that one may adapt to compile pre-link and link C programs with both the `PDF417EN` object and the C run-time library.

NOTE: The COMPILE.STEPLIB has received an override in our example to interact with DSN=SYS1.SCEERUN, and DSN=SYS1.SCBCCMP. It may be the case that your system would not need to override them at all if the name of your libraries match then ones provided in the EDCC procedure.

Upon completion of the build of `SAMPLEC` you can then execute the program.

3. Edit ‘`NEOMEDIA.PDF417.RSCLIB(XSAMPLEC)'` and SUBMIT it.

Confirm that the JCL is correct for your environment and make any changes as required.

The XSAMPLEC member in the NEOMEDIA.PDF417.RSCLIB PDS executes the SAMPLEC executable created above. This program generates an output file via the NMPDFFNT DD card.

NOTE: The very first step in the XSAMPLEC has been commented out because there is nothing to be deleted during the first run of this program. If you decided to re-execute this program, then the DELETE step would be needed.

For successful execution, there will be no screen output but one file will be generated as output via the NMPDFFNT DD card defined in the JCL. The file that is created from this test program is a font character file which contains a PDF417 symbol with the following encoded:

```
This is a test of the NeoMedia Technologies PDF417 Encoder.  The error
correction will encode at level 3 with ECC padding.
```

View this file and confirm that it contains the appropriate font characters. If it does not, you will have to print the file with the appropriate font definition file loaded. Printing the symbol files is beyond the scope of this document.

The following JCL demonstrates how to link the PDF417 encoder with a C program and the C run-time SCEELKED library. If one observes the expansion of this EDCPL, then the SCEELKED library will be noted.

```
//jobcard here
//*
//LINK     EXEC EDCPL,LPARM='MAP,LIST,AMODE=31,RMODE=ANY'
//PLKED.STEPLIB DD DSN=SYS1.SCEERUN,DISP=SHR
//PLKED.SYSIN   DD DSN=NEOMEDIA.PDF417.OBJLIB(YOURPROG),DISP=SHR
//              DD DSN=NEOMEDIA.PDF417.OBJLIB(PDF417EN),DISP=SHR
//LKED.SYSLMOD  DD DSN=YOUR.LOADLIB,DISP=SHR
//LKED.SYSIN    DD *
 ENTRY CEESTART
 NAME YOUREXEC(R)
//
```

Depending on your system configuration you may not even need the above PLKED.STEPLIB override.

This JCL demonstrates how to execute a program that uses the encoder in an AFP printing environment. If your environment is properly configured, the C run-time libraries (e.g., SCEERUN) will be found via the concatenation list:

```
//jobcard here
//AFP   OUTPUT  FORMDEF=PDF417,PAGEDEF=PDF417,
//       PRMODE=PAGE,DATACK=UNBLOCK
//STEP1  EXEC PGM=YOURNAME
//STEPLIB   DD DSN=YOUR.LOADLIB,DISP=SHR
//SYSPRINT  DD SYSOUT=*
//SYSOUT    DD SYSOUT=*
//SYSUDUMP  DD SYSOUT=(F,,71)
//OUTFILE   DD DSN=YOUR.PDFOUT.DATA,DISP=SHR,
//             UNIT=SYSDA,SPACE=(TRK,(5,5),RLSE),
//             DCB=(RECFM=FBA,LRECL=133,BLKSIZE=23408)
//ERRFILE   DD DSN=YOUR.PDFOUT.ERR,DISP=SHR,
//             UNIT=SYSDA,SPACE=(TRK,(5,5),RLSE),
//             DCB=(RECFM=FBA,LRECL=133,BLKSIZE=23408)
//*
//STEP2  EXEC PGM=IEBGENER
//SYSPRINT  DD  SYSOUT=*
//SYSOUT    DD  SYSOUT=*
```

```
//SYSUT1    DD  DSN=YOUR.PDFOUT.DATA,DISP=SHR
//SYSUT2    DD  SYSOUT=*,OUTPUT=*.AFP
//SYSIN     DD  DUMMY
//
```

Execution Errors

Depending on what environment you are installing in, users have encountered some execution problems. If you are getting a `SOC1` or `SOC4` error at execution time, check the concatenation list. The PDF417 Encoder depends on the C runtime being available at runtime. The `SOC1` and `SOC4` error usually indicates a missing `LOADLIB` or runtime system library. Users have encountered this when using the LE370 due to the lack of adding the system libraries to the concatenation list automatically on installation of C. Make sure that `SCEERUN` and `SCEELKED` are included in the system concatenation list. Also confirm that the PDF417 Encoder `LOADLIB`, (`NEOMEDIA.PDF417.LOADLIB)`, is included.

AS/400 Installation

It is highly recommended that the installation of the PDF417 encoder be performed by a skilled System Administrator with intimate knowledge of your AS/400 environment. The following instructions are written in a general sense and will work on most AS/400 systems. However, due to the wide and varied configurations possible, it is impossible to provide concise instructions for all possible scenarios.

Requirements

Under OS/400, the PDF417 encoder requires at a minimum, OS/400 V3R2M0. If you are running an earlier release of OS/400, please contact NeoMedia Technologies.

The encoder is supported for use with the following calling languages:

- COBOL/400 and ILE COBOL
- RPG/400 and ILE RPG
- ILE C
- PL/I

After completing the installation, the PDF417 encoder will be available as a service program for ILE programmers and as a callable program for OPM language programming. While using the encoder with development languages and products other than those listed above may be possible, these types of configurations have not been tested and are not support by NeoMedia Technologies.

Preparing for Installation

The PDF417 is distributed on tape in `SAVLIB` format. The installation library, `PDF`**`Vxxxx`**, contains the object modules that comprise the PDF417 encoder and a number of support files, including fonts. In most cases, the library has been saved as a V3R2M0 save library. Refer to the tape label for the specific `SAVLIB` version that was used to create the media.

The `PDF`**`Vxxxx`** library contains the source files for the AFP fonts. Once the library has been restored, the font resources will need to be compiled on your system.

If you require either Xerox or HP PCL fonts, a separate media (usually a diskette) will have been shipped containing the appropriate fonts. Please refer to your NeoMedia Technologies Standard Software License for specific details on which fonts have been licensed for use. Use of unlicensed fonts is a violation of the Standard Software License and is strictly prohibited.

The library also contains a number of sample programs and encoder support files. These will be discussed in a later section.

Installing the Encoder

STEP 1: Restoring the encoder from tape

The encoder library for OS/400 is in `SAVLIB` format. Refer to the label on the distribution media for the precise `SAVELIB` version that was used. The `RSTLIB` command is used to restore the library:

```
RSTLIB SAVLIB(PDFVxxxx) DEV(TAPxx) VOL(NEO001)
```

The following objects will be restored:

Object	Type	Attribute	Text
PDF**Vxxxx**	*LIB	*TEST	NeoMedia PDF417 Encoder **Vx.x.x.x**
PDFENCOD	*PGM	CLE	PDF417 OPM Interface Program
PDFINITF	*PGM	CLE	PDF417 OPM Interface Program
PDF**Vxx**SRV	*SRVPGM	CLE	PDF417 ILE Service Program
H	*FILE	PF-SRC	PDF417 C Includes
QCBLLESRC	*FILE	PF-SRC	PDF417 ILE COBOL Sample
QCSRC	*FILE	PF-SRC	PDF417 ILE C Sample
QDDSSRC	*FILE	PF-SRC	PDF417 DDS Sample
QFNTRSC	*FILE	PF-DTA	PDF417 Font Source Files
QLBLSRC	*FILE	PF-SRC	PDF417 COBOL/400 Sample
QRPGLESRC	*FILE	PF-SRC	PDF417 ILE RPG Sample
QRPGSRC	*FILE	PF-SRC	PDF417 RPG/400 Sample

STEP 2: Compiling the font resources

If you are using an IPDS printer, you will need to create the PDF417 font resources from the source files provided. Four fonts have been supplied: 2 for 240 DPI printers and 2 for 300, 600, and 1200 DPI printers. NeoMedia recommends that your compile and install both fonts.

PSF/400 searches a default set of single byte character set libraries (`QFNT01` through `QFNT19`) when looking for AFP font resources. NeoMedia recommends that you install our PDF417 fonts in one of these libraries. If you have not installed custom fonts on your system before, you may have to create a library (e.g., `CRTLIB LIB(QFNT01)`). Note that we do not recommend that the fonts be installed in the IBM supplied library `QFNTCPL`.

The following commands demonstrate how to compile the font resources, installing them in the `QFNT01` library. Note that the order in which the files are compiled is important: a coded font's character set and code page must be compiled first (e.g., `C0PD2206` and `T1PDF417` must be compiled before `X0PD2206`). Since all four fonts use the same code page (`T1PDF417`), we only need to compile it once:

```
CRTFNTRSC FNTRSC(QFNT01/T1PDF417) FILE(PDFVxxxx/QFNTRSC)
CRTFNTRSC FNTRSC(QFNT01/C0PD2206) FILE(PDFVxxxx/QFNTRSC)
CRTFNTRSC FNTRSC(QFNT01/X0PD2206) FILE(PDFVxxxx/QFNTRSC)
CRTFNTRSC FNTRSC(QFNT01/C0PD2309) FILE(PDFVxxxx/QFNTRSC)
CRTFNTRSC FNTRSC(QFNT01/X0PD2309) FILE(PDFVxxxx/QFNTRSC)
CRTFNTRSC FNTRSC(QFNT01/C0PD3309) FILE(PDFVxxxx/QFNTRSC)
CRTFNTRSC FNTRSC(QFNT01/X0PD3309) FILE(PDFVxxxx/QFNTRSC)
CRTFNTRSC FNTRSC(QFNT01/C0PD3412) FILE(PDFVxxxx/QFNTRSC)
CRTFNTRSC FNTRSC(QFNT01/X0PD3412) FILE(PDFVxxxx/QFNTRSC)
```

NOTE: these fonts use the standard AFP font naming conventions. Thus, those are zeros in the file names, not O's.

Any printer writers that will be using these fonts should be stopped and restarted to guarantee that they find the new fonts.

Binding with the Encoder

The AS/400 has two distinct development environments: OPM and ILE. The OPM (Original Programming Model) consists of language products like RPG/400, COBOL/400, and PL/I. ILE (Integrated Language Environment) consists of language products like ILE RPG, ILE COBL, and ILE C.

The way the PDF417 encoder library is *used* in these two environments is identical; however, the way the encoder is *called* differs. NeoMedia Technologies' PDF417 Encoder was written in and compiled using ILE C. The software is distributed as an ILE service program. In ILE environments, linking with the encoder is straightforward; it is simply specified as a Bind Service Program to the Create Program command (CRTPGM). For example, if your calling ILE program were named MYLIB/MYPROGRAM, the program would be linked as follows:

```
CRTPGM PGM(MYLIB/MYPROGRAM) BNDSRVPGM(PDFVxxxx/PDFVxxSRV)
```

If the PDF**Vxxxx** library has been added to the library list, you may instead wish to use the following:

```
CRTPGM PGM(MYLIB/MYPROGRAM) BNDSRVPGM(*LIBL/PDFVxxSRV)
```

Using the PDF417 encoder with an OPM language is a little different. An interface program (PDF**Vxxxx**/PDFENCOD) has been provided. This is a "stub" C program which, when invoked, in turn calls the PDF**Vxx**SRV service program (since an OPM program cannot directly call an ILE service program). The PDFENCOD program was linked using the following command:

```
CRTPGM PGM(PDFVxxxx/PDFENCOD)
       MODULE(PDFVxxxx/PDFENCOD)
       BNDSRVPGM(*LIBL/PDFVxxSRV)
       ACTGROUP(*CALLER)
       TGTRLS(V3R2M0)
```

There is a subtle implication here; the PDF**Vxx**SRV service program was linked from the library list (note the BNDSRVPGM parameter). Thus, if you are using an OPM language, the PDF**Vxx**SRV service program **must** appear in the job's library list.

The second interface program, PDFINITF, was linked in the same fashion.

Running the Sample Programs

A number of sample programs have been provided: a COBOL/400 program, an ILE COBOL program, an ILE C program, an RPG/400 program, and an ILE RPG program. For simplicity of compiling and running the samples, you may wish to change your current library to the PDF**Vxxxx** library.

The sample programs use the same printer file, PDF**Vxxxx**/PDF417. Before you can compile any of the sample programs, you must create the printer file the samples will use. Two DDS's have been provided; one for 240 DPI printers and one for the 300 DPI family of printers. These DDS's differ in two ways; each uses a different font and different vertical line spacing. Refer to the *PDF417 Encoder Programming Guide* for more information on font selection and line spacing.

When creating the printer file, you must use a device type of *AFPDS. To create the printer file for a 240 DPI printer, use the following:

```
CRTPRTF FILE(PDFVxxxx/PDF417)
        SRCFILE(PDFVxxxx/QDDSSRC)
        SRCMBR(PDF240)
        DEVTYPE(*AFPDS)
        REPLACE(*YES)
```

To create the printer file for a 300 DPI printer, use the following:

```
CRTPRTF FILE(PDFVxxxx/PDF417)
        SRCFILE(PDFVxxxx/QDDSSRC)
        SRCMBR(PDF300)
        DEVTYPE(*AFPDS)
        REPLACE(*YES)
```

COBOL

To compile and link the sample program for COBOL:

NOTE: Compiling and testing the COBOL sample is not a required step for installation of the PDF417 Encoder. It is recommended if you will be using the COBOL API.

Compiling and Running the sample COBOL/400 Program

The sample COBOL/400 program calls the `PDFVxxxx/PDFENCOD` program. The source for the sample program is in `PDFVxxxx/QLBLSRC(CBLPDF)`. It can be compiled using the following command:

```
CRTCBLPGM PGM(PDFVxxxx/CBLPDF)
          SRCFILE(PDFVxxxx/QLBLSRC)
          SRCMBR(CBLPDF)
          REPLACE(*YES)
```

The program is now ready to run. It will generate a spool file named PDF417 to the default output queue:

```
CALL CBLPDF
```

Compiling and Running the sample ILE COBOL Program

The sample ILE COBOL program links with the `PDFVxxxx/PDFVxx`SRV service program. The source for the sample program is in `PDFVxxxx/QCBLLESRC(CBLPDF)`. You must first compile the COBOL module:

```
CRTCBLMOD MODULE(PDFVxxxx/CBLPDF)
          SRCFILE(PDFVxxxx/QCBLLESRC)
          SRCMBR(CBLPDF)
          REPLACE(*YES)
```

Now create the ILE program (i.e., link it with the PDF417 service program):

```
CRTPGM PGM(PDFVxxxx/ILECBLPDF)
       MODULE(PDFVxxxx/CBLPDF)
       BNDSRVPGM(*LIBL/PDFVxxSRV)
       REPLACE(*YES)
```

The program is now ready to run. It will generate a spool file named PDF417 to the default output queue:

```
CALL ILECBLPDF
```

RPG

To compile and link the sample program for RPG:

NOTE: Compiling and testing the RPG sample is not a required step for installation of the PDF417 Encoder. It is recommended if you will be using the COBOL API.

Compiling and Running the sample RPG/400 Program

The sample RPG/400 program calls the `PDF`**`Vxxxx`**`/PDFENCOD` program. The source for the sample program is in `PDF`**`Vxxxx`**`/QRPGSRC(RPGPDF)`. It can be compiled using the following command:

```
CRTRPGPGM PGM(PDFVxxxx/RPGPDF)
          SRCFILE(PDFVxxxx/QRPGSRC)
          SRCMBR(RPGPDF)
          REPLACE(*YES)
```

The program is now ready to run. It will generate a spool file named PDF417 to the default output queue:

```
CALL RPGPDF
```

Compiling and Running the sample ILE RPG Program

The sample ILE RPG program links with the `PDF`**`Vxxxx`**`/PDF`**`Vxx`**`SRV` service program. The source for the sample program is in `PDF`**`Vxxxx`**`/QRPGLESRC(RPGPDF)`. You must first compile the RPG module:

```
CRTRPGMOD MODULE(PDFVxxxx/RPGPDF)
          SRCFILE(PDFVxxxx/QRPGLESRC)
          SRCMBR(RPGPDF)
          REPLACE(*YES)
```

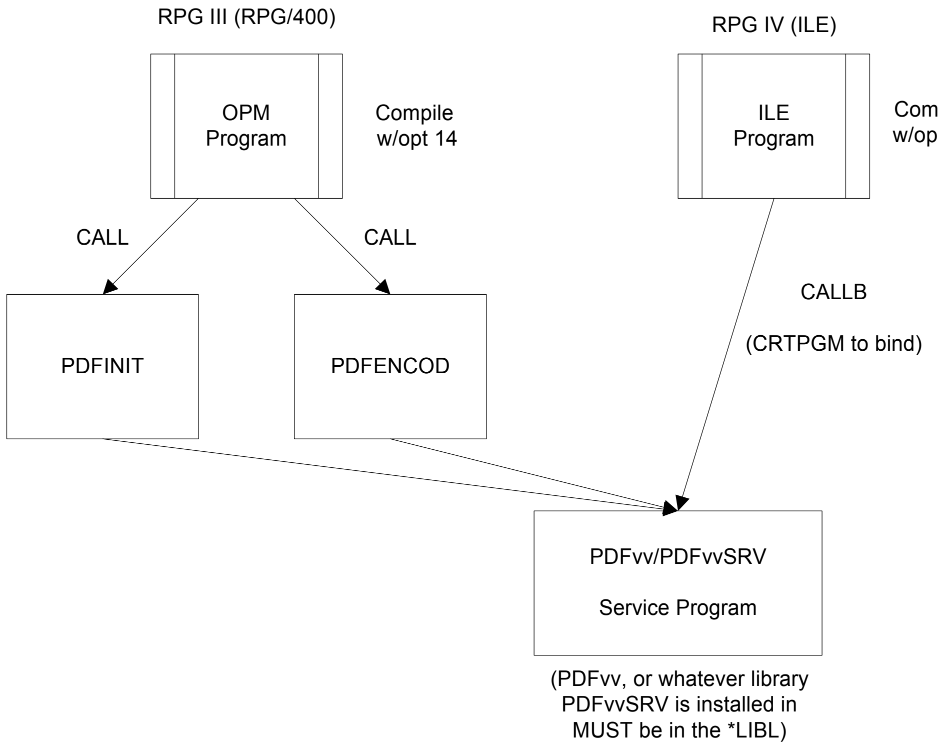
Now create the ILE program (i.e., link it with the PDF417 service program):

```
CRTPGM PGM(PDFVxxxx/ILERPGPDF)
       MODULE(PDFVxxxx/RPGPDF)
       BNDSRVPGM(*LIBL/PDFVxxSRV)
       REPLACE(*YES)
```

The program is now ready to run. It will generate a spool file named PDF417 to the default output queue:

```
CALL ILERPGPDF
```

Following is a pictorial diagram of how the pieces fit together using RPG:



C Language

To compile and link the sample program for C:

NOTE: Compiling and testing the C sample is not a required step for installation of the PDF417 Encoder. It is recommended if you will be using the C API.

Compiling and Running the sample ILE C Program

The sample ILE C program links with the PDF**Vxxxx**/PDF**Vxx**SRV service program. The source for the sample program is in PDF**Vxxxx**/QCSRC(CPDF). You must first compile the C module:

```
CRTCMOD MODULE(PDFVxxxx/CPDF)
        SRCFILE(PDFVxxxx/QCSRC)
        SRCMBR(CPDF)
        REPLACE(*YES)
```

Now create the ILE program (i.e., link it with the PDF417 service program):

```
CRTPGM PGM(PDFVxxxx/CPDF)
       MODULE(PDFVxxxx/CPDF)
       BNDSRVPGM(*LIBL/PDFVxxSRV)
       REPLACE(*YES)
```

The program is now ready to run. It will generate a spool file named PDF417 to the default output queue:

```
CALL CPDF
```

Output of the sample programs

The sample programs generate a single PDF417 symbol to the output spooler. If you view the spooler, you should see several lines of text characters looking something line the following:

```
OOEDGEALCJBNAC LBKC HDNCICJILAOMAD
OOEDGMDCBNLOMMF DOAAADLKOCOEAMOMAD
OOEDFJGLBALHMFHG KLLFGNEHKE OIOMAD
OOEDGMBOKCDBAACMNJALGGKBCJKNOIOMAD
OOEDGELAJOACMODEHOBMHGI ECJNAIOMAD
OOEDGJOBBNBCAOIAJKLHADLFOCJOIAOMAD
OOEDECMNCKCCIMAINHHDAGGFFBILNAOMAD
OOEDGJDHCDCHINHJ NELLGGKNKNIFAOMAD
OOEDECCNBNOGMMLENIDCLFANEKODNMOMAD
OOEDFHNGJ JDAC LBLJF E FCBHLAIOMAD
OOEDGDNAKACJACGM ODHDFBGBCJGAIOMAD
OOEDEA OBHBOACHHFIHDNGCNIJHLOIOMAD
OOEDE DHBLLHAHJL M DCFHIHBHGAMOMAD
OOEDGJB KJGCANOGJLLOADIOCCNHHMOMAD
OOEDFHBOJ ICMKJG IOIMEBGHBEGHAOMAD
OOEDGIFOJG NIKC LNLIHFHGGKBHFAOMAD
OOEDEAOCCJFAALHAJMCI FENCBHOKAOMAD
OOEDDKIOKNDNEGKAHLGLIFCNDKIGMAOMAD
OOEDGKDCJMNCILNJ MK LDBFLCKDFAOMAD
OOEDGM EKLADINKMNOI KFKHDBHGLMOMAD
OOEDGBEOJGHDIHNBNNKNHENCFCBDGMOMAD
OOEDEKHLCEMNALLEHHFGCDBGGCICCMOMAD
```

When sent to the printer, rather than the characters seen above, a PDF417 symbol will be printed.

If you do not get a PDF417 symbol, but rather the rows of characters "all bunched" up, then PSF/400 is performing a font substitution (check the messages for `QSYSOPR`). Possible causes include:

- The PDF417 fonts have not been properly installed in a library that `PSF/400` can find.
- The printer writer was not restarted after the PDF417 fonts were installed.
- The DDS is using the incorrect font for your printer's density.

Font Selection

As mentioned earlier, the PDF417 encoder is shipped with all available printer fonts. Supported platforms include IBM AFP, Xerox, and Hewlett Packard PCL printers.

The following tables list all of the supported platforms and printers. The font name is the name of the font file in the distribution kit. Refer to the *PDF417 Programming Guide* for more information on setting the code points (specifically, the PdfFontInitRender function for C/Setting Parameters for COBOL). Failure to set the code points in the program can cause unpredictable output and can even cause printer reboots.

The information has been broken into two tables; one for UNIX and PC Platforms (ASCII platforms) and a second for IBM S/370, S/390, and AS/400 systems (EBCDIC platforms).

UNIX and PC Platforms:

Printer Family	DPI	Portrait Font	Landscape Font
HP PCL	300	HPPP3309	HPPL3309
HP PCL	300	HPPP3412	HPPL3412

IBM S/370, S/390, AS/400:

Printer Family	DPI	Portrait Font	Landscape Font
IBM AFP	240	X0PD2206	N/A
IBM AFP	240	X0PD2309	N/A
IBM AFP	300	X0PD3309	N/A
IBM AFP	300	X0PD3412	N/A
XEROX AFP – 5 word	300	X5P309	X5L309
XEROX AFP – 5 word	300	X5P310	X5L310
XEROX AFP – 5 word	300	X5P412	X5L412
XEROX AFP – 9700 series	300	X9P309	X9L309
XEROX AFP – 9700 series	300	X9P310	X9L310
XEROX AFP – 9700 series	300	X9P412	X9L412