precisely

Finalist

Finalist Installation Guide

Version 9.22.0

Table of Contents

7 - Finalist Databases

8 - Auxiliary Databases

9 - Using the Distribution Tool

10 - Glossary of Terms

1 - Installing Finalist on Windows

In this section

Before You Begin the Windows Installation

Before installing a new version of Finalist, we recommend that you uninstall any previous version of Finalist. (See "Uninstalling Finalist for Windows" on page 15 for more information.)

The requirements for installing Finalist on a Windows platform are:

- A supported platform. For a list of supported platforms, see the "Supported Platforms" document available on the Precisely Support site.
- A Pentium or higher processor. Note: Finalist is not supported on devices with ARM processors.
- A minimum of 96 MB RAM. This memory requirement does not include memory for your application code and data.

Hard disk space requirements

- Finalist requires a minimum of 15 MB of free hard disk space to install the product.
- The addressing databases, zip4us.dir and city.dir, require approximately 1.1 GB of space.
- (Optional) PreciselyID database: 3.5 GB to support unique identifiers
- (Optional) Additional hard disk space is required to install United States Postal Service (USPS®) databases for enhanced matching capabilities:
- • **Delivery Point Validation (DPV™)** space requirements depend on the DPV database type:
  - Full: 1.0 GB
  - Split: 1.6 GB
  - Flat: 2.3 GB
  - **LACS$^{Link®}$** database: 375 MB
  - **SuiteLink®** database: 525 MB
  - **Residential Delivery Indicator (RDI™)** database: 48 MB
  - **Early Warning System (EWS)** file: 300 KB
  - **Enhanced Line of Travel (eLOT®)** file: 350 MB

Finalist Keys

Finalist uses a software-based key to license usage. The Finalist key is restricted to your licensed System ID(s). The System ID information must be provided to your Precisely Account Manager

before a software key can be created for your use. If you use Finalist on more than one system, you can provide up to seven System IDs in a single key.

Finding Your System ID

After installing Finalist on your system, to find your System ID:

1. Open a Windows command prompt (cmd.exe).
2. Change to the installed Finalist bin directory.
3. Execute the KeyStore.exe program with no parameters.

   For Program Files:

   - When Finalist is installed, the default Finalist installation directory is C:\Precisely\FinalistXXX.

4. You will see output similar to:

```
cd "\Precisely\FinalistXXX\bin"
C:\Precisely\FinalistXXX\bin>KeyStore.exe
Syntax: keystore key
        where key is the Finalist key.
        A file called keyfile.txt will be generated
        and should be placed in your working folder.
System Tag(s): 10C048,3681D0
C:\Precisely\FinalistXXX\bin>
```

5. The System IDs for Finalist display in the System Tag(s) field. In the example above, the System IDs for Finalist are:

```
10C048
3681D0
```

The System IDs are required to generate your license key. This information is based on your host system. If you are running on a virtual machine (VM) and move your VM to a new host system, you need a new key with new System IDs.

Entering Your Software Key

Use one of these processes to enter your Finalist software key:

- Place the Finalist software key in your source file and enter the Finalist software key in PBFN-GCFG-SOFTWAREKEY (cSoftwareKey).
- Enter the Finalist software key in the pbfn.cfg file.

- Use the KeyStore program to store the Finalist software key in a method available to the Finalist engine. This process makes the Finalist software key automatically available to all programs accessing Finalist.

  Note: The KeyStore program applies to the Finalist software key only. This process does not apply for the LACS^{Link} or DPV keys.

Saving Your Software Key

The Keystore.exe program makes it unnecessary to store your software key in your individual driver code or in individual pbfn.cfg files. KeyStore.exe is a command line program (not a GUI). The syntax of the program is:

```
C:\Precisely\FinalistXXX\bin\keystore.exe <your software key>
```

KeyStore.exe generates a keyfile.txt file to place in the folder where you are running Finalist (not necessarily the Finalist \bin folder). The output will be similar to:

```
C:\Precisely\FinalistXXX\bin\KeyStore.exe <your software key>
Software key: <your software key>
 Finalist version: xx.xx.*
 Platform: Windows
 CASS expires: mm-dd-yyyy
Finalist key {your software key} successfully set.
```

Finalist for Windows Files

Finalist will install both the 32-bit and 64-bit versions. Functionality is identical; only the underlying architecture is different.

Finalist for Windows Library Files

This table describes the Finalist for Windows library files.

File Name	Description
addrscan.dll	AddrScan utility

File Name	Description
addrscan.lib	AddrScan library
dpvdll.dll	Delivery Point Validation (DPV) library
lacsf.dll	LACS^{Link} library
mfwrapmn.dll	Support library
mfwrapmn.lib	Support library
pbelot.dll	Enhanced Line of Travel (eLOT) library
pbfn.dll	Finalist engine
pbfn.lib	Library for Finalist engine
preciselyIDf.dll	PreciselyID library
rdif.dll	Residential Delivery Indicator (RDI) library
stelinkf.dll	SuiteLink library

Finalist for Windows Application Files

File Name	Description
configDriver.exe	Utility for editing the pbfn.cfg configuration file
finalist.exe	Finalist batch driver
keystore.exe	Utility to store software key in product
lookup.exe	Finalist lookup tool
statecut.exe	Utility to create regional bases from national base

File Name	Description
workbench.exe	Graphical user interface to launch Finalist applications

Finalist for Windows Postal Coding Files

The Finalist for Windows software includes the postal coding files and a current copy of the monthly database files. The postal coding files include a static library, header files, and support files for programming and operation of the postal coding library.

Table 1: Finalist Postal Coding Files

File Type	Files
CASS run mode files	xxf.su$ xxl.su$
Database files	city.dir zip4us.dir ewsmmdd.txt elot.dir
Delivery Point Validation (DPV) database files	dpv.db (DPV Flat File) dpvh.db (DPV Full File) dpvs.db (DPV Split File)
Suite^{Link} database files	slk.db
LACS^{Link} database file	llk.db
RDI database file	rdi.db
Log file	log.txt
Configuration file	pbfn.cfg
Includes	C header files

File Type	Files
Includes\copylib	COBOL copybooks
PreciselyID database file	prcsly.db
Documentation files	Documentation files
Samples	Sample Finalist job/def/input files
Samples\C	C sample files
Samples\CPP	C++ sample files
Samples\Java	Java sample files
Samples\VB.NET	Visual Basic .NET sample files

Finalist for Windows Installation Default Directories

Files	Installation Default Directory
.dll, .exe, and pbfn.cfg files	C:\Precisely\FinalistXXX\bin C:\Precisely\FinalistXXX\bin64
Library files	C:\Precisely\FinalistXXX\lib C:\Precisely\FinalistXXX\lib64
Include files	C:\Precisely\FinalistXXX\includes
Sample files	C:\Precisely\FinalistXXX\Samples
Documentation files	C:\Precisely\FinalistXXX\Documentation

- For “FinalistXXX”, “XXX” represents the current version.
- Both 32-bit and 64-bit are installed at the same time.

- The 32-bit executable's and libraries are called bin and lib respectively.
- The 64-bit executable's and libraries are called bin64 and lib64 respectively.

Environment Variables

Finalist does not support the use of environment variables. Finalist searches for components in the current directory (for example, pbfn.dll, finalist.exe, etc.).

Uninstalling Finalist for Windows

Before installing a new version of Finalist, we recommend that you uninstall any previous version of Finalist. To uninstall Finalist for Windows, follow these steps.

1. From the Start menu, select Settings/Control Panel.
2. Double-click on the Add/Remove programs icon.
3. Click on the Finalist list item and click on the Add/Remove button.
4. Click on the Yes button in response to the Confirm File Deletion message.

Installation Steps for Windows

To install Finalist on a Windows platform, follow these steps.

1. Download the software.
2. Use the "Save-As" option to save the .ZIP file to your system.
3. Extract the files.
4. Double click **Setup.exe**.
5. Follow the instructions provided by the installation program for a successful installation.

Installing on Modern Windows Systems

To install Finalist on modern Windows Systems like Windows 10, follow these steps.

1. If Windows starts on the Start screen, click **Desktop**.
2. Ensure that your account has Administrator privileges (required).
3. Click **File Explorer** and navigate to the location of the downloaded Finalist installer.
4. Windows natively supports ZIP files. The ZIP attribute may not appear as the file extension (.zip). You have two options:
   a) Double-click the ZIP file and use File Explorer to browse the ZIP file contents. If you use this option, you must install using the Finalist_Installer.msi file.
   b) Right click the Finalist installer ZIP file. Select **Extract All…** and accept the default options. Click **Extract**.
5. Install the SETUP.exe files or the Finalist_Installer.msi files and follow these steps:
   a) Use File Explorer to navigate to the **C:\Precisely** folder.
   b) Right-click the **FinalistXXX** folder and select **Properties**.
   c) Select the **Security** tab and click **Edit…**.
   d) Select group **Users** and turn on the **Modify** or **Full Control** attribute.

Installing the Finalist Database

This section provides instructions for installing the following Finalist database files:

- city.dir
- zip4us.dir

> **Note:** The same Finalist databases are used for the 32-bit and 64-bit installations of Finalist.

To install the Finalist database files, follow these steps.

1. Download the database(s) via the estore. You will receive an automatic notification via email with special links to the Finalist databases. Click on the desired link and follow the instructions to complete the download. New users to the estore will need to establish a new estore account for the first download transaction.
2. Download the installation .ZIP file.
3. Use the "Save-As" option to save the .ZIP file to your system.

4. Extract the files.
5. Follow the instructions provided by the installation program for a successful database installation.
6. The installation process places the database files in the following locations.

Files	Installation Default Directory
city.dir	C:\Precisely\FinalistXXX\db
zip4us.dir	C:\Precisely\FinalistXXX\db

Installing the Auxiliary Finalist Databases

This section provides instructions for installing the optional Finalist database files in a Windows environment.

Note: The same optional databases are used for the 32-bit and 64-bit installations of Finalist.

Optional Database	Required for CASS Processing?
Early Warning System (EWS)	No
Enhanced Line of Travel (eLOT)	No
Delivery Point Validation (DPV)	Yes
LACS^{Link}	Yes
PreciselyID	No
Residential Delivery Indicator (RDI)	No
SuiteLink	Yes

To install the optional Finalist databases on a Windows platform, follow these steps.

1. Download the database(s) from the estore. You will receive an automatic notification via email with special links to the Finalist databases. Click the desired link and follow the instructions in the FAQ section of the email to complete the download. New users to the estore will need to establish an estore account for the first download transaction.
2. Download the installation .ZIP file.
3. Use the "Save-As" option to save the .ZIP file to your system.
4. Extract the files.
5. Follow the instructions provided by the installation program for a successful database installation.
6. Move the extracted files directly to your database location.

   Note: If you are installing the optional PreciselyID database, it needs to be installed in the same location as the Addressing files (city.dir and zip4us.dir).

Verifying the Windows Installation

To verify your Finalist for Windows installation, follow these steps:

1. Make sure Finalist has access to the database files.
2. Modify the Finalist configuration file pbfn.cfg to indicate the location of the database files. Use an ASCII text editor to modify the lines in the pbfn.cfg to set the City and ZIP+4 file names and the Software Key. For our installation example above, the appropriate lines would be:

```
City Directory Filename = C:\Precisely\Finalist\db\city.dir
ZIP+4 Directory Filename 1 = C:\Precisely\Finalist\db\zip4us.dir
SOFTWARE KEY = software key shipped with the product
```

3. Go to the samples directory and run the batch driver program, using sample.job from the samples directory as the input job file.

```
C:\Precisely\FinalistXXX\Samples> ..\bin\finalist sample.job
```

   Note: In “FinalistXXX” above, “XXX” represents the current version.

4. After the command is completed, the sample.out, sample.val, sample.err, and sample.rpt files are created in the samples directory. The sample.out file contains the output addresses. The output addresses in this file appear in upper case. If the files are generated and the addresses are in upper case, the program ran to completion.

2 - Installing Finalist on Unix or Linux

In this section

Before You Begin the Unix or Linux Installation

The requirements for installing Finalist on a Unix or Linux platform are:

- A supported platform. For a list of supported platforms, see the "Supported Platforms" document available on the Precisely Support site.
- A Pentium or higher processor. Note: Finalist is not supported on devices with ARM processors.
- A minimum of 96 MB RAM. This memory requirement does not include memory for your application code and data.

Hard disk space requirements

- Finalist requires a minimum of 15 MB of free hard disk space to install the product.
- The addressing databases, zip4us.dir and city.dir, require approximately 1.1 GB of space.
- (Optional) PreciselyID database: 3.5 GB to support unique identifiers
- (Optional) Additional hard disk space is required to install United States Postal Service (USPS®) databases for enhanced matching capabilities:
- • **Delivery Point Validation (DPV™)** space requirements depend on the DPV database type:
  - Full: 1.0 GB
  - Split: 1.6 GB
  - Flat: 2.3 GB
- **LACS$^{Link®}$** database: 375 MB
- **SuiteLink®** database: 525 MB
- **Residential Delivery Indicator (RDI™)** database: 48 MB
- **Early Warning System (EWS)** file: 300 KB
- **Enhanced Line of Travel (eLOT®)** file: 350 MB

Finalist Keys

The Finalist key is restricted to your licensed System ID(s). The System ID information must be provided to your Precisely Account Manager before a software key can be created for your use. If you use Finalist on more than one system, you can provide up to seven System IDs in a single key.

> **Note:** When you upgrade your hardware, you must provide the new System ID information to your Precisely Account Manager so a new software key can be generated.

Finding Your System ID

You can run the KeyStore program (refer to the section **"Saving Your Software Key" on page 23**) with no input to display your System ID information. If you do not have access to the KeyStore program, the following method can also be used to obtain the System ID information.

HP-UX

1. From a prompt, issue the following command:

```
uname -i
```

2. The response is similar to:

```
3509210850
```

3. The System ID for Finalist is the last six characters. In the example above, the value is:

```
210850
```

AIX

1. From a prompt, issue the following command:

```
uname -m
```

2. The response is similar to:

```
00C2E8BE4C0
```

3. The System ID for Finalist is in characters 3-8 of the response. In the example above, the value is:

```
C2E8BE
```

SunOS

1. From a prompt, issue the following command:

```
hostid
```

2. The response is similar to:

```
830294e1
```

3. The number returns in a hexadecimal format. Convert the number to decimal. The sample response converts to:

```
2197984481
```

4. The System ID for Finalist is the last six characters. In the example above, the value is:

```
984481
```

Red Hat

1. From a prompt, issue the following command:

```
/sbin/ifconfig
```

2. The response is similar to:

```
eth0      Link encap:Ethernet  HWaddr 00:06:5B:AB:19:68
…
```

3. The System ID for Finalist is the last six actual characters following Ethernet HWaddr. In the example above, the value is:

```
AB1968
```

SuSE

1. From a prompt, issue the following command:

```
cat /proc/cpuinfo
```

2. The response is similar to:

```
vendor_id       : IBM/S390
# processors    : 2
bogomips per cpu: 514.45
processor 0: version = 00,  identification = 000777,  machine = 2096
processor 1: version = 00,  identification = 100777,  machine = 2096...
```

3. The System ID for Finalist is the numbers following identification. In the example above, the two values are:

```
000777
100777
```

Saving Your Software Key

Keystore is an optional program that allows you to avoid storing your software key in your individual driver code or in individual pbfn.cfg files. KeyStore generates a keyfile.txt file that can be placed in the folder where you are running Finalist (not the Finalist /bin folder).

KeyStore is a command line program (not a GUI). The syntax of the program is:

```
./keystore <your software key>
```

KeyStore generates a keyfile.txt file that is to be placed in the folder from which you will run Finalist.

If you run KeyStore without a parameter, KeyStore displays the System ID. It is this System ID that is required to generate your Finalist software key. You can use this method as an alternative to the commands described above.

Finalist for Unix or Linux Files

The Finalist for Unix or Linux installation package includes the postal coding files and a current copy of the monthly database files. The postal coding files include a static library, header files, and support files for programming and operation of the postal coding library.

> **Note:** Finalist installation includes a 32-bit version and a 64-bit version. The 32-bit version of Finalist is located in the finalist/bin and finalist/lib folders. The 64-bit version of Finalist is located in the finalist/bin64 and finalist/lib64 folders. The same samples and databases are used by the 32-bit and 64-bit versions of Finalist.

Table 2: Finalist for Unix or Linux Files

File Type	Files
Binary files	finalist keystore statecut xxf.su$ xxl.su$
Configuration file	pbfn.cfg
Documentation files	Documentation files
Include files	C header files and COBOL copybooks
Library files	libaddrscan.a libpbfn.a
Delivery Point Validation (DPV) database files	dpv.db (DPV Flat file) dpvh.db (DPV Full file) dpvs.db (DPV Split file)
SuiteLink database file	slk.db
LACS^{Link} database file	llk.db
RDI database file	rdi.db
PreciselyID database file	prcsly.db
Samples	Sample Finalist job/def/input files
Samples/C	C sample files
Samples/CPP	C++ sample files
Samples/Java	Java sample files

Environment Variables

Finalist does not support the use of environment variables. Finalist searches for components in the current directory (for example, finalist, statecut, etc.).

Installation Steps for Unix or Linux

To install Finalist on a Unix or Linux platform, follow these steps.

1. Download the software.
2. Use the "Save-As" option to save the file to your system.
3. Extract the files.
4. Use an FTP type of program to binary transfer the .Z file from your Windows machine to your desired platform. You must ensure the filename ends with a capital Z.
5. Install Finalist using the appropriate commands below:

   UNIX

```
%> uncompress FINALIST.Z
%> tar -xvf FINALIST
```

   LINUX

```
%> tar -xvf FINALIST.Z
```

Installing the Finalist Database

This section provides instructions for installing Finalist database files **city.dir** and **zip4us.dir**. The same Finalist databases are used for the 32-bit and 64-bit installations of Finalist.

> **Note:** Linux uses the "Windows" format of the Finalist databases (ASCII, Little Endian). UNIX uses the UNIX version of the Finalist databases (ASCII Big Endian).

To install database files:

1. Download the database(s).

2. Use the "Save-As" option to save the file to your system.
3. Extract the files.
4. Use an FTP type of program to binary transfer the files from your Windows machine to your desired platform.

Installing the Auxiliary Finalist Databases

This section provides instructions for installing the optional Finalist databases in a Unix environment. The same optional databases are used for the 32-bit and 64-bit installations of Finalist.

Table 3: Optional Finalist Databases

Optional Database	Required for CASS Processing?
Early Warning System (EWS)	No
Enhanced Line of Travel (eLOT)	No
Delivery Point Validation (DPV)	Yes
LACS^{Link}	Yes
PreciselyID	No
Residential Delivery Indicator (RDI)	No
SuiteLink	Yes

To install the database files, follow these steps.

1. Download the database(s).
2. Use the "Save-As" option to save the file to your system.
3. Extract the files.
4. Use an FTP type of program to binary transfer the files from your Windows machine to your desired platform.

Verifying the Unix or Linux Installation

To verify your Unix or Linux installation, follow these steps.

1. Make sure Finalist has access to the database files.
2. Modify the Finalist configuration file pbfn.cfg to indicate the location of the database files. Use the Unix text editor vi or another ASCII text editor to modify the lines in the pbfn.cfg to set the City and ZIP+4 file names and the software key. For our installation example above, the appropriate lines would be:

```
City Directory Filename = /precisely/finalist/db/city.dir
ZIP+4 Directory Filename 1 = /precisely/finalist/db/zip4us.dir
SOFTWARE KEY = software key
```

3. Go to the samples directory and run the batch driver program, using sample.job from the samples directory as the input job file.

```
cd /precisely/finalist/samples
../bin/finalist sample.job
```

4. After the command is completed, the sample.out, sample.val, sample.err, and sample.rpt files are created in the samples directory. The sample.out file contains the output addresses. The output addresses in this file appear in upper case. If the files are generated and the addresses are in upper case, the program ran to completion.

Finalist for Unix or Linux Documentation Notes

The Finalist software follows a platform design and programming approach which assures that the same library functionality is available via all Finalist libraries, regardless of operating system. For documentation purposes, the API overviews and descriptions are documented once for all platforms. For detailed descriptions of each API call, refer to your Finalist Reference Guide.

3 - Installing Finalist for z/OS

In this section

Before You Begin the z/OS Installation

The requirements for installing Finalist on a z/OS platform are:

- You must be running on a currently supported IBM operating system.
- See **http://www.ibm.com/software/info/supportlifecycle** for a list of currently supported IBM operating systems.
- For batch processing, you must run in a region that has at least 100M of storage available. Additional options like DPV or LACS^{Link} can significantly increase the storage requirements.
- Finalist provides, as a default option, a version of batch executable compiled with the IBM High Performance Linker option also called XPLINK. XPLINK requires the use of PDSEs. You are not required to use the XPLINK version of Finalist. If you choose not to use the XPLINK version, remove the steps in INSTALLB that begin with XP. If you choose to use the XPLINK version of Finalist batch processing, replace your normal Finalist batch loadlib with the XPLINK Finalist loadlib.
- Finalist provides, as a default option, a version of batch executable compiled for 64-bit z/OS. This version uses the X64 qualifier in the dataset name. You are not required to use the X64 version of Finalist. If you choose not to use the X64 version, remove the steps in INSTALLB that begin with X6. If you choose to use the X64 version of Finalist batch processing, replace your normal Finalist batch loadlib with the X64 Finalist loadlib.
- If you are installing Finalist z/OS for CICS, you must install the batch option first, then move ahead to **Installing Finalist for z/OS CICS** on page 39.
- If you are installing Finalist z/OS for IMS, you must install the batch option first, then move ahead to **Installing Finalist for z/OS IMS** on page 45.

Finalist Keys

Finalist uses a software-based key to license usage. The Finalist key is restricted to your licensed System ID(s). The System ID information must be provided to your Precisely Account Manager before a software key can be created for your use. If you use Finalist on more than one system, you can provide up to seven System IDs in a single key.

> **Note:** When you upgrade your hardware, you must provide the new System ID information to your Precisely Account Manager so a new software key can be generated.

Finding Your System ID

You can run the KeyStore program with no input to display your System ID information. If you do not have access to the KeyStore program, the following methods can also be used to obtain the System ID information.

z/OS

1. Issue the operator command:

```
D M=CPU
```

2. The response is similar to:

```
IEE174I 15.33.37 DISPLAY M 648
PROCESSOR STATUS
ID  CPU                 SERIAL
00  +                   01E3E02096
…
```

3. The System ID for Finalist is the first six characters below the SERIAL field. In the example above, the value is:

```
01E3E0
```

Storing the Finalist Key

PGM=KEYSTORE is a program that allows you to avoid storing your software key in your individual driver code or in individual pbfn.cfg files. The method for storing your software key differs based on whether you are running in a 31-bit mode or a 64-bit mode.

> **Note:** Finalist Compatibility Interface users are required to use PGM=KEYSTORE. CICS and IMS transactions use the Compatibility Interface internally and therefore require the Finalist software key to be stored.

KEYSTORE for 31-Bit

PGM=KEYSTORE generates a PBFNKEYF load module that is linked into your Finalist load library. This is required for CI users and optional for all other users.

> **Note:** Precisely-provided CICS and IMS transactions run with the CI interface. If you are using CICS or IMS, you will need to run PGM=KEYSTORE multiple times, each one targeting the proper LOAD library.

Sample JCL for KeyStore can be found in the FNSOURCE library that is part of the installation. In summary, PGM=KEYSTORE:

- Reads in the key
- Generates an assembler (BAL) program that is compiled and linked and placed into the Finalist LOADLIB. This program only needs to be rerun if you replace your LOADLIB or change your system hardware.

If you run PGM=KEYSTORE without a value in SYSIN, the System ID displays. It is this System ID that is required to generate your Finalist software key. You can use this method as an alternative to the D M=CPU command described above.

KEYSTR64 for 64-Bit

Sample JCL for KEYSTR64 can be found in the FNSOURCE library that is part of the installation. In summary, PGM=KEYSTORE:

1. Reads in the key
2. Generates a C program called PBFNKEYF.
3. PBFNKEYF is compiled into the installed Finalist X64 object library.
4. PBFN is re-linked into the installed X64 LOADLIB using the newly created KEYFILE.

This program only needs to be rerun if you replace your LOADLIB or change your system hardware.

Terminology

While many parts of the Finalist documentation are geared toward Windows and Unix customers, Finalist functions equally well on z/OS operating systems. While some parts of the documentation refer to Windows and Unix style names, z/OS style names can be used. For example, the pbfn.cfg file used on Windows and Unix operating systems is referenced as DD PBFNCFG on z/OS systems. Where generic descriptions are used, Windows and Unix conventions are followed. Where specific detail is provided, z/OS details are given.

Finalist Load Libraries

The Finalist software installation includes the following load libraries.

Table 4: Finalist z/OS Load Libraries

Load Library	Description
31-bit non-XPLINK	Standard load library with full support.
31-bit XPLINK	Higher level of performance compared to the 31-bit non-XPLINK load library; however, the 31-bit XPLINK load library requires XPLINK linkage for entry and exit. Your application must be compiled specifically to support XPLINK. XPLINK provides limited support for the Finalist Compatibility Interface (CI).
64-bit	For pure 64-bit applications. 64-bit is only available with XPLINK. You cannot mix 31-bit and 64-bit processes. As of publish date, IBM COBOL does not support 64-bit. COBOL applications cannot call the 64-bit version of Finalist. C and High Level Assembler (HLASM) are the only IBM compilers currently supported for 64-bit processing. The Finalist batch driver, PGM=FINALIST, fully supports the 64-bit environment when run from the 64-bit load library.

The Finalist Compatibility Interface (CI) remains supported in the 31-bit non-XPLINK environment. The CI (FINAL, FINALOL, LPFN000, etc.) is not supported in the 64-bit environment. There are currently no plans to support the CI in the 64-bit environment due to the inherent complexities of 64-bit and XPLINK processing. Clients planning on migrating beyond 31-bit non-XPLINK or to platforms other than z/OS should include re-writing applications to the Finalist native interface.

Although modified to run natively in a 64-bit environment, Finalist is not fully exploiting 64-bit processing at this time. Planning is underway to more fully exploit 64-bit processing in future releases.

Installation Library

Table 5: Installation Library

Installation Library Member	Contents Description
BALCOPYL	Assembler (BAL) macros and COPY members used to access the Finalist product.
BAOBJLIB	Object members used to create the executable version of Finalist. This library contains both batch and IMS object members.
CINCLUDE	C headers used to access the Finalist product.
CIOBJLIB	Object members used to create the executable version of Finalist for CICS.
CISOURCE	Source to use in conjunction with the Finalist for CICS option. Contains a mixture of Assembler (BAL), COBOL, C, and CICS source.
COBCOPYL	COBOL COPY members used to access the Finalist product.
FNSOURCE	JCL and notes for installing and verifying Finalist.
SAMPLIVP	Sample input stream to test the successful installation of Finalist. This file does not necessarily contain codeable addresses. The purpose of this file is simply to verify that the Finalist installation was successful.
X6OBJLIB	Contains object members used to create the executable version of batch Finalist using 64-bit.
XBOBJLIB	Object members used to create the executable version of batch Finalist using XPLINK, IBM's High Performance Linker.

Installation Steps for z/OS

To install the Finalist for z/OS software, follow these steps.

1. Download the software from the estore to a PC. Unzip the file.
2. Use binary FTP files to send files to your mainframe.
   a) The first set of files is the *.UNLOAD files. These files should be uploaded to an FB 80 sequential dataset using as much as 100 3390 tracks. A specific block size is not required.
   b) The second set of files is the *.DAT files.

      DPVSUD00.DAT is a dataset with a size of 1 track and RECFM=F (not FB) and LRECL=7.

      LLKSUD00.DAT is a dataset with a size of 1 track and RECFM=F (not FB) and LRECL=7.

      SAMPLIVP.DAT is a dataset with a size of 10 3390 tracks and RECFM=FB and LRECL=600.

   c) An uploadz.ftp file has been provided to give approximate file sizes for files to be uploaded to your mainframe. These file sizes are based on 3390 DASD storage. This file is based on standard IBM FTP protocol. The FTP protocol may vary on your system. See your systems programmer for site-specific details.
3. Edit the uploadz.ftp file and insert your mainframe address.
   a) Edit the USERID and PASSWORD values to the appropriate values for your system. You may remove any sections that do not apply to your site. Note that there are separate sections for install component uploading and for sample restore JCL uploading. (You may remove the IMS or CICS sections if you do not use those options.)
   b) Change "hlq." to the appropriate high level qualifier for your installation.
4. The uploadz.bat file has been provided to automatically execute FTP to send the files to your mainframe. From Windows Explorer, double click the uploadz.bat file to send the files to your mainframe.
5. All customers should edit and submit the hlq.RECEIVEB.JCL JCL to create the base (batch) object libraries:

   - BALCOPYL
   - BAOBJLIB
   - CINCLUDE
   - COBCOPYL
   - FNSOURCE
   - X6OBJLIB
   - XBOBJLIB

6. CICS customers should edit and submit the hlq.RECEIVEC.JCL JCL to create the CICS object libraries:

- CIOBJLIB
- CISOURCE

7. IMS customers should edit and submit the hlq.RECEIVEI.JCL JCL to create the IMS object libraries:

   - ACBCNTL
   - DBDSORC
   - GENMAC
   - JCLLIB
   - PSBSORC
   - SORCLIB
   - TFMTSORC

8. Link the Finalist base product.

   Edit and submit member INSTALLB to build the Finalist executable system from the object members. This member creates regular executables, XPLINK (High Performance linker) executables, and 64-bit executables (X64). XPLINK requires z/OS 1.2 and higher for complete support.

   INSTALLB generates PDSE datasets for both object and load libraries. If you do not want to use PDSE datasets, you may manually remove the DSNTYPE=LIBRARY statement from the install member. The XPLINK and X64 executables require PDSE datasets. If you remove PDSE datasets, you must also manually remove the XPLINK and X64 LINK-EDIT steps. All XPLINK and X64 LINK-EDIT step names begin with the letter X.

   Steps LINKLIST and LINKADSC normally display RC=4. Edit member INSTALLB using the comments at the beginning of the member.

9. Store your Finalist key in the Finalist load library.

   If you will be running the Compatibility Interface (CI aka Wrapper), you must also edit and submit member KEYSTORE to store your Finalist key into the Finalist load library. Edit member KEYSTORE using the comments at the beginning of the member.

Using Libraries Created by INSTALLB

The INSTALLB installation process creates:

- hlq.BATCH.IMPORT
- hlq.BATCH.LOAD
- hlq.BATCH.X64.LOAD
- hlq.BATCH.X64.IMPORT
- hlq.BATCH.XPLINK.LOAD
- hlq.BATCH.XPLINK.IMPORT

The hlq.*.LOAD library should be used in your STEPLIB or JOBLIB concatenation to run this version of Finalist.

The hlq.*.IMPORT library should be used as a definition side-deck (DD SYSDEFSD) when compiling your programs that call the Finalist native APIs (for example, PBFNInit, PBFNProcess, PBFNTerminate).

Installing the Finalist Database

After completing the Finalist build, you must load the Finalist databases. This is a separate step because the database load should be rerun each month with current data.

The Finalist database files include data for all 50 states, DC, and all United States territories. If your input file only includes address records from specific states, use the **State Cut** feature to create a database file for just the specific states you need to process your input file. Using the smaller state-specific database files allows Finalist to search through less data during address assignment and to use less processing time. Follow the instructions below to load the full Finalist databases or state-specific databases.

> **Note:** The USPS data varies each month. For information on finding database file information, including record counts and calculating cylinders/tracks, refer to the section **"Finding Database File Information"**.

Loading the Full Finalist Databases

To load the full Finalist databases, follow these steps.

1. Edit and submit member DBREPROD to create and load the Finalist VSAM database files with current data.
2. Edit member DBREPROD using the comments at the beginning of the member.

> **Note:** DBALOCF gets the files from an FTP server. DBREPROD loads the file into the database.

Loading State-Specific Databases

The Finalist City and ZIP+4 database files are both required to use the State Cut feature. The State Cut feature creates new City and ZIP+4 database files containing data for the requested states. These files must be used together to perform address assignment. The newly created City File will

not work with the original Finalist Data File. Also, the newly created Data File will not work with the original City File. You must use both newly created files together to properly perform address assignment.

1. Edit and submit member STATECUT to create and load smaller state-specific Finalist VSAM database files with current data.
2. Edit member STATECUT using the comments at the beginning of the member. The amount of space required when using the STATECUT program depends on the number and/or size of the state(s) selected.

Installing Auxiliary Databases

To load the auxiliary Finalist databases, follow the steps for the appropriate database.

Installing the DPV Database

USPS CASS regulations require DPV processing to generate a USPS Form 3553 (CASS Summary Report).

1. Download the DPV database from the estore.
2. After determining the appropriate DPV database format for your installation site, use the corresponding JCL below to load the DPV database.

Table 6: Delivery Point Validation (DPV) Database Files

File	Description
Flat	Flat builds the DPVDB file. To load the DPV Flat database, use DBALOCD and DPREPROD.
Split	Split builds the DPVSDB file. To load the DPV Split database, use DBALOCDS and DPREPRSD.
Hash (Full)	Hash builds the DPVHDB file. To load the DPV Hash (Full) database, use DBALOCDH and DPREPRHD.

Installing the eLOT Database (Optional)

Download the eLOT database from the estore.

1. To perform eLOT processing, run job LTREPROD to load the eLOT data files.
2. Edit member LTREPROD using the comments at the beginning of the member.

Installing the EWS Database (Optional)

Download the Early Warning System (EWS) database from the estore.

1. To perform EWS processing, run job EWREPRO to load the EWS data files. EWS data can come from two different sources. Finalist provides the most current file each month on tape. However, the USPS updates this file weekly. The JCL is setup to load the data from the Finalist distribution tapes.
2. Build the VSAM EWS databases using member EWREPRO.
3. Edit member EWREPRO using the comments at the beginning of the member.

Installing the LACS^{Link} Database

USPS CASS regulations require LACS^{Link} processing to generate a USPS Form 3553 (CASS Summary Report). Download the LACS^{Link} LLKDB database from the estore.

1. To perform LACS^{Link} processing, run job LLREPROD to load the LACS^{Link} data files.
2. Edit member LLREPROD using the comments at the beginning of the member.

Installing the PreciselyID Database (Optional)

Download the PreciselyID database directly from the estore.

After obtaining the databases, run the PKREPROD (FNPBKJCL for IMS) JCL to make the PreciselyID databases available to Finalist on the mainframe. Mainframe users can access PreciselyID in batch, CICS, and IMS environments.

Installing the RDI Database (Optional)

Download the RDI database directly from the estore.

After obtaining the databases, run the RDREPROD (FNRDJCL for IMS) JCL to make the RDI databases available to Finalist on the mainframe. Mainframe users can access RDI in batch, CICS, and IMS environments.

Installing the SuiteLink Database

USPS CASS regulations require SuiteLink processing to generate a USPS Form 3553 (CASS Summary Report). Download the SuiteLink SLKDB database from the estore.

1. To perform SuiteLink processing, run job SLREPROD to load the SuiteLink data files.
2. Edit member SLREPROD using the comments at the beginning of the member.

Using the EWS Table

For information on using the Finalist Early Warning System (EWS) option, refer to the section **"Using EWS Processing"**.

For z/OS environments, the FNSOURCE PDS includes the LOADEWS JCL. LOADEWS reads the raw EWS file and creates a BAL (assembler) table that contains the same data. This table is compiled and linked under the name PBFNEWS. The PBFNEWS module replaces the default (empty) module that ships with Finalist. Customers choosing to use the PBFNEWS module must re-populate and replace the module every time new EWS data is received. Precisely ships EWS data monthly. The USPS distributes EWS data weekly.

Verifying the Finalist z/OS Installation

To verify your Finalist for z/OS installation, follow these steps.

1. Make sure you have the Finalist databases installed. This may include EWS, eLOT, RDI, DPV, LACS^{Link}, and Suite^{Link} databases depending on your environment.
2. Edit and submit member FINALIST in your FNSOURCE library.

3. Verify that the job ran to a successful completion and produced a valid USPS Form 3553 (CASS Summary Report) and Finalist Batch Report.

Finding Database File Information

You can find information for the Finalist database files, including record counts, in the Finalist Technical Bulletin. To access the Finalist Technical bulletin, go to the estore and log in with your User ID and password. You can use the information in the Finalist Technical Bulletin to calculate the required number of cylinders/tracks for the database files.

Calculating the Number of Cylinders/Tracks

This section provides instructions for calculating the number of cylinders/tracks required for a given Finalist VSAM cluster. The number of cylinders/tracks required varies according to type of DASD, record size, and control interval size. The actual number of records in the Finalist VSAM clusters also varies slightly with each new update or distribution. Refer to the IBM VSAM Administration Guide to find capacity and physical record sizes by device type.

The Technical Bulletin, available on the Precisely Web site, shows the record counts for each of the files. Use the information from the Technical Bulletin and the following formula to calculate the number of cylinders/tracks that a given VSAM cluster requires.

1. Subtract 8 from the control interval size of the file. Divide this value by the record size (LRECL). The result is the number of logical records per VSAM physical record.
2. Multiply the result from step 1 by the number of physical records that fit on one track. The chart in the IBM VSAM Administration Guide (z/OS users) shows the number of physical records that fit on one track. The result is the total number of VSAM physical records in the file.
3. Divide the total number of records in the file by the result from step 2. The result is the number of tracks required for the file. To find the number of cylinders required, divide the number of tracks required by the number of tracks per cylinder.

This sample calculation demonstrates how to calculate the number of cylinders/tracks for a 17,479 record VSAM cluster having a record size of 4088 and a control interval size of 4096 on 3390 DASD.

```
1.   4096  (VSAM control interval size)
  -     8  (VSAM overhead)
     4088  (Number of data bytes per physical record)
  /  4088  (Divide by record size)
        1  (Number of logical records per physical record)

2.      1  (Number of logical records per physical record)
  X    12  (Physical records per track for 3390 DASD)
```

```
       12  (Number of logical records per track for 3390 DASD)

3.  17479  (Total number of records in the file)
  /    12  (Number of logical records per track for 3390 DASD)
    1457  (Total number of 3390 tracks required)
  /    15  (Tracks per cylinder for 3390)
      98  (Number of cylinders required)
```

4 - Installing Finalist for z/OS CICS

In this section

Before You Begin the z/OS CICS Installation

The requirements for installing Finalist in a z/OS CICS environment are:

- You must be running on a currently supported IBM operating system.
- See **http://www.ibm.com/software/info/supportlifecycle** for a list of currently supported IBM operating systems.

Installation Library Description

This table describes the installation library CICS-specific members.

Table 7: Installation Library Description

Installation Library Member	Description
CIOBJLIB	Contains object members used to create the executable version of Finalist for CICS.
CISOURCE	Contains source that may be used in conjunction with the Finalist for CICS option. This contains a mixture of Assembler (BAL), COBOL, C, and CICS source.
COBCOPYL	Contains COBOL COPY members used to access the Finalist product.
FNSOURCE	Contains JCL and notes for installing and verifying Finalist.

Installing Finalist for z/OS CICS

The USPS CASS regulations now require DPV, LACS^{Link}, and Suite^{Link} to run in CASS mode. Other ancillary databases like EWS are not required, but provide additional coding accuracy. While online access is not required to run in CASS mode, consistent results between online and batch can only be achieved if the same process is followed in both cases. To install Finalist z/OS CICS, follow these steps.

1. Follow the steps to install Finalist for z/OS in Chapter 4, Installing Finalist for z/OS.
2. Edit and submit member INSTALLC from the FNSOURCE PDS to build the Finalist CICS system from the object members. Consult your CICS systems programmer for your system details. Edit member INSTALLC using the comments at the beginning of the member.
3. After building the Finalist CICS option, add your Finalist key to the Finalist CICS load library. To do this, edit and submit member KEYSTORE. You can find KEYSTORE in your Finalist FNSOURCE library. Perform the edits using the comments at the beginning of the member.
4. After building the Finalist CICS option, you will need to add the Finalist requirements to your CICS region. For the Finalist CICS option, see PBFN015D in CISOURCE for the RDO entries required. PBFN015D references Finalist and other ancillary database file names. These should be edited for accurate names (change hlq.). Member RDOJCL (in the CISOURCE library) contains starter JCL for running RDO. Consult your CICS systems programmer for your system details.

Updating the EWS File in a z/OS CICS Environment

Finalist reads the Early Warning System (EWS) File into memory only after recycling a CICS region or after issuing a NEWCOPY command to PBFNEWS. This reduces input/output and significantly improves product performance. If you update the EWS File, CBEWS, and do not recycle your CICS region, issue a NEWCOPY command for the module PBFNEWS:

```
CEMT S PROG(PBFNEWS) NEW
```

Using Libraries Created by INSTALLC

The INSTALLC installation process creates:

- hlq.CICS.LOAD
- hlq.CICS.IMPORT

The hlq.CICS.LOAD library should be used in your DFHRPL concatenation for CICS to run this version of Finalist.

The hlq.CICS.IMPORT library should be used as a definition side-deck (DD SYSDEFSD) when compiling your programs that call the Finalist native APIs (for example, PBFNInit, PBFNProcess, PBFNTerminate).

Verifying the Finalist CICS Installation

If you decide to install the CICS component of Finalist, refer to the section “Using Finalist CICS” in your Finalist User’s Guide for information on verifying your Finalist CICS installation using the sample transactions LPCT, PBFN, and LPCF.

Finding Database File Information

You can find information for the Finalist database files, including record counts, in the Finalist Technical Bulletin. To access the Finalist Technical bulletin, go to the estore and log in with your User ID and password. On the left side of the window under "Technical Services" click on My Documentation and then click on Greenbars and Technical Bulletins to access the Finalist Technical Bulletin. You can use the information in the Finalist Technical Bulletin to calculate the required number of cylinders/tracks for the database files.

Calculating the Number of Cylinders/Tracks

This section provides instructions for calculating the number of cylinders/tracks required for a given Finalist VSAM cluster. The number of cylinders/tracks required varies according to type of DASD, record size, and control interval size. The actual number of records in the Finalist VSAM clusters also varies slightly with each new update or distribution. Refer to the IBM VSAM Administration Guide to find capacity and physical record sizes by device type.

The Technical Bulletin, available on the Precisely website, shows the record counts for each of the files. Use the information from the Technical Bulletin and the following formula to calculate the number of cylinders/tracks that a given VSAM cluster requires.

1. Subtract 8 from the control interval size of the file. Divide this value by the record size (LRECL). The result is the number of logical records per VSAM physical record.
2. Multiply the result from step 1 by the number of physical records that fit on one track. The chart in the IBM VSAM Administration Guide (z/OS users) shows the number of physical records that fit on one track. The result is the total number of VSAM physical records in the file.
3. Divide the total number of records in the file by the result from step 2. The result is the number of tracks required for the file. To find the number of cylinders required, divide the number of tracks required by the number of tracks per cylinder.

The following sample calculation demonstrates how to calculate the number of cylinders/tracks for a 17,479 record VSAM cluster having a record size of 4088 and a control interval size of 4096 on 3390 DASD.

```
1.   4096 (VSAM control interval size)
  -     8 (VSAM overhead)
     4088 (Number of data bytes per physical record)
  /  4088 (Divide by record size)
        1 (Number of logical records per physical record)

2.      1 (Number of logical records per physical record)
  X    12 (Physical records per track for 3390 DASD)
       12 (Number of logical records per track for 3390 DASD)

3.  17479 (Total number of records in the file)
  /    12 (Number of logical records per track for 3390 DASD)
     1457 (Total number of 3390 tracks required)
  /    15 (Tracks per cylinder for 3390)
       98 (Number of cylinders required)
```

5 - Installing Finalist for z/OS IMS

In this section

Before You Begin the z/OS IMS Installation

The requirements for installing Finalist IMS:

- You must be running on a currently supported IBM operating system.
- See **http://www.ibm.com/software/info/supportlifecycle** for a list of currently supported IBM operating systems.

Installation Library Description

This table describes the installation library IMS-specific members.

Table 8: Installation Library Description

Installation Library	Description
ACBCNTL	ACB generation parameters for Finalist for IMS
DBDSORC	Database descriptions for Finalist for IMS
GENMAC	Transaction generation descriptions for Finalist for IMS
JCLLIB	Sample JCL for various tasks for Finalist for IMS
PSBSORC	PSB source samples for Finalist for IMS
SORCLIB	SORCLIB Sample source for various tasks for Finalist for IMS
TFMTSORC	Source for Finalist screens for Finalist for IMS

Installing Finalist for z/OS IMS

The USPS CASS regulations now require DPV, LACS^{Link}, and SuiteLink to run in CASS mode. Other ancillary databases, like EWS, are not required, but provide additional coding accuracy. While online access is not required to run in CASS mode, consistent results between online and batch can only be achieved if the same process is followed in both cases. To install Finalist On-Line on a z/OS IMS platform, follow these steps.

1. Follow the steps to install Finalist for z/OS in **Installing Finalist for z/OS**.
2. Link the Finalist IMS product (optional).
3. Edit and submit member INSTALI1 from the FNSOURCE PDS. Edit member INSTALI1 using the comments at the beginning of the member.
4. After you have built the Finalist IMS option, you will need to add your Finalist key to the Finalist IMS load library. To do this, edit and submit member KEYSTORE that is in your Finalist FNSOURCE library. Perform the edits using the comments at the beginning of the member.
5. Edit and submit member INSTALI2 from the FNSOURCE PDS to build the Finalist IMS system from the object members. Edit member INSTALI2 using the comments at the beginning of the member.
6. Finalist IMS requires the use of DL/I data files and provides definitions for using SHISAM databases. Finalist IMS provides sample transactions S56LPCH and S56LPWNH to access Finalist using DL/I processing.
7. For additional IMS installation steps specific to IMS, refer to members MFSGEN, PSBGEN, and ACBGEN in Finalist IMS JCLLIB as samples to run the necessary IMS generations. At time of publication the following are sizes for IMS related files:

Table 9: Finalist IMS File Sizes

IMS File	Size Required
IMS.ACBCNTL	'1,1,15' 3390 tracks
IMS.DBDSORC	'1,1,15' 3390 tracks
IMS.GENMAC	'1,1,15' 3390 tracks
IMS.JCLLIB	'2,1,15' 3390 tracks
IMS.PSBSORC	'1,1,15' 3390 tracks

IMS File	Size Required
IMS.SORCLIB	'5,1,15' 3390 tracks
IMS.TFMTSORC	'5,1,15' 3390 tracks

Using Libraries Created by INSTALI2

The INSTALI2 installation process creates:

- hlq.IMS.LOAD
- hlq.IMS.IMPORT

The hlq.IMS.LOAD library should be used in your STEPLIB or JOBLIB concatenation to run this version of Finalist.

The hlq.IMS.IMPORT library should be used as a definition side-deck (DD SYSDEFSD) when compiling your programs that call the Finalist native APIs (for example, PBFNInit, PBFNProcess, PBFNTerminate).

Completing the Finalist IMS Installation

Use the installation instructions in this section to complete the installation of Finalist for IMS.

DL/I Processing Overview

IMS Finalist requires the use of IMS Data Language/I (DL/I) for all Finalist files. You need to generate Data Base Descriptions (DBDs) to define the database structures for IMS. Source for generating the DBDs is provided. Conversion programs are required to convert some Finalist files into a SHISAM format. JCL is provided in the Finalist IMS JCLLIB to define and populate the IMS SHISAM files. For DL/I processing you must pass the PCB addresses for all Finalist files your application needs. These addresses are available when your application receives control from the IMS DL/I control program. COBOL users must use program LPFNPCB to pass the input PCB address to the Finalist control structure . See the following COBOL source extract. The following is a sample COBOL program called by IMS when calling Finalist.

```
...
COPY LPFNCL01.
...
LINKAGE SECTION.
```

```
*=*============================================================*=*
*=* THIS PROGRAM USES 8 PCBS. -                                 *=*
*=* (1) AN I/O PCB TO COMMUNICATE WITH THE TERMINAL             *=*
*=*     AND RECIEVE MESSAGES FROM THE TERMINAL.                 *=*
*=* (2) A PCB THAT POINTS TO THE DL/I DATAFILE.                 *=*
*=* (3) A PCB THAT POINTS TO THE DL/I CITYFILE.                 *=*
*=* (4-8) PCB'S THAT POINT TO ANCILLARY DATABASES.              *=*
*=*============================================================*=*
COPY LPCFIPCB.
                                                          EJECT
COPY LPCFDPCB.
                                                          EJECT
COPY LPCFCPCB.
                                                          EJECT
=*============================================================*=*
=*                                                           *=*
=* THE LENGTH OF THE FOLLOWING PCB'S ARE NOT EXACT NOR       *=*
=* DO THEY NEED TO BE.                                       *=*
=*                                                           *=*
=*============================================================*=*
01 FNEWS-PCB            PIC  X(42).
01 FNLOT-PCB            PIC  X(42).
01 DPV-PCB              PIC  X(42).
01 LLK-PCB              PIC  X(42).
01 SLK-PCB              PIC  X(42).
01 RDI-PCB              PIC  X(42).
01 PBK-PCB              PIC  X(42).
…
PROCEDURE DIVISION.
     ENTRY 'DLITCBL' USING I-O-PCB, DATAFILE-PCB, CITY-PCB,
                           FNEWS-PCB, FNLOT-PCB,
                           DPV-PCB, LLK-PCB, SLK-PCB, RDI-PCB.
     CALL 'LPFNPCB' USING DATAFILE-PCB IMS-INITDAT.
     CALL 'LPFNPCB' USING CITY-PCB     IMS-INITCITY.
     CALL 'LPFNPCB' USING FNEWS-PCB    PCBFNEWS.
     CALL 'LPFNPCB' USING FNLOT-PCB    PCBFNLOT.
     CALL 'LPFNPCB' USING DPV-PCB      PCBFNDPV.
     CALL 'LPFNPCB' USING LLK-PCB      PCBFNLLK.
     CALL 'LPFNPCB' USING SLK-PCB      PCBFNSLK.
     CALL 'LPFNPCB' USING RDI-PCB      PCBFNRDI.
     CALL 'LPFNPCB' USING PBK-PCB      PCBFNPBK.
…
     MOVE '0' TO FINAL-FUNCTION-CODE
     CALL 'FINALI' USING FINAL-CALL-AREA
```

DL/I Installation Procedures

To install IMS Finalist and convert the Finalist City/State and Data Files to a DL/I format, follow these steps.

1. Perform an MFSGEN for all format members in the hlq.IMS.TFMTSORC library. Use sample MFSGEN in hlq.IMS.JCLLIB as an example for the members you need to generate.
2. If necessary, modify the DBDGENs for all Finalist databases that you require. Use sample DBDGEN in hlq.IMS.JCLLIB to define the databases.
3. Create an application PSB for all application programs that call Finalist. Finalist requires S56LPCH and S56LPWNH for its online applications. Use sample PSBGEN in hlq.IMS.JCLLIB to complete the PSB gen.
4. Use the sample FNDBJCL in hlq.IMS.JCLLIB to create and populate the Finalist DL/I files.
   a) Use the sample FNDPJCL in hlq.IMS.JCLLIB to create and populate the DPV DL/I files.
   b) Use the sample FNLLJCL in hlq.IMS.JCLLIB to create and populate the LACS^{Link} DL/I files.
   c) Use the sample FNEWJCL in hlq.IMS.JCLLIB to create and populate the EWS DL/I files.
   d) Use the sample FNLTJCL in hlq.IMS.JCLLIB to create and populate the eLOT DL/I files.
   e) Use the sample FNRDJCL in hlq.IMS.JCLLIB to create and populate the RDI DL/I files.
   f) Use the sample FNPBKJCL in hlq.IMS.JCLLIB to create and populate the PreciselyID DL/I files.
   g) Use the sample FNSLJCL in hlq.IMS.JCLLIB to create and populate the Suite^{Link} DL/I files.
   h) Use the sample FNSUDJCL in hlq.IMS.JCLLIB to create and populate the DPV and LACS^{Link} security DL/I files.
5. Perform the ACBGEN for the members contained in hlq.IMS.ACBCNTL. Use sample ACBGEN in hlq.IMS.JCLLIB as an example of the members you need to generate.
6. Add the IMS Finalist load library (hlq.IMS.LOAD) to the STEPLIB for the IMS region. Your STEPLIB should be concatenated as shown below.

```
//STEPLIB    DD DISP=SHR,DSN=hlq.IMS.LOAD
```

7. Run a Stage 1 IMSGEN based on the Stage 1 macro supplied in member FINALIST in the hlq.IMS.GENMAC library. A Stage 1 GEN is only required the first time you install IMS Finalist, or when changes are made to the transaction names or the SPA sizes.
8. Edit DFSMDA in hlq.IMS.JCLLIB to reference your database names. For more information, refer to your IBM IMS/VS Utilities Manual. Use sample IMSDALOC in hlq.IMS.JCLLIB to process the DFSMDA statements you just edited.
9. Finalist reads the Early Warning System (EWS) File into memory only after recycling your IMS region or after stopping and restarting program PBFNEWS. This reduces input/output and significantly improves product performance. If you update the EWS File, CBEWS, and do not

recycle your IMS region, issue the following IMS commands (xx is the current message number for your IMS WTOR):

```
/xx/STO PROG PBFNEWS
/xx/STA PROG PBFNEWS
```

10. If your site uses Exceptions File processing, refer to "Exceptions Table Option" in your *Finalist User's Guide* for more information. This step is optional and is the final step in the installation. Use sample PBFNEXTB in hlq.IMS.JCLLIB to process your Exceptions File.
11. Verify your installation. For information on verifying your Finalist IMS installation, refer to "Using Finalist IMS" in your *Finalist User's Guide*.

DL/I Batch Processing

When running Finalist in a batch environment, you can run either as a z/OS batch job or a DL/I batch job.

When running as a z/OS batch job, follow the normal steps for running Finalist.

When running with the IMS DLIBATCH procedure (or executing DFSRRC00 directly), you must concatenate the hlq.IMSBATCH.LOAD library in front of hlq.IMS.LOAD library and the hlq.BATCH.LOAD library when specifying your job's STEPLIB or JOBLIB. For example:

```
//STEP3   EXEC DLIBATCH,MBR=program_name,
//         PSB=program_psb
//G.STEPLIB  DD
//           DD
//           DD DISP=SHR,DSN=hlq.IMSBATCH.LOAD
//           DD DISP=SHR,DSN=hlq.IMS.LOAD
//           DD DISP=SHR,DSN=hlq.BATCH.LOAD
//G.IEFRDER  DD DSN=NULLFILE,UNIT=SYSDA
//G.SYSUDUMP DD SYSOUT=*
//G.SYSOUT   DD SYSOUT=*,
//           DCB=(LRECL=133,BLKSIZE=0,RECFM=FBA)
//G.DFSVSAMP DD *
8192,40
/*
```

> **Note:** Using IMSBATCH.LOAD library allows Finalist to load files into memory for better performance and produce a USPS Form 3553 (CASS Summary Report) and other reports that are turned off for IMS online processing.

You must also create a PSB for the application program. The application program must accept the PCBs passed into the program and pass the PCBs on to Finalist.

When using the DLIBATCH procedure, you use the same DD names as you would for your normal z/OS batch processing but you must point to the DL/I datasets that are used by the IMS On-Line system.

Note: DFSVSAMP must run with a minimum buffer size of 8192.

Verifying the Finalist IMS Batch Installation

A batch driver, FINALSTI, is provided to allow execution of Finalist in a batch environment using SHISAM databases.

After IMS is installed, run the IMS sample JCL FINALSTI to ensure IMS was properly installed in your environment. Please note that the ancillary databases use an alternate DDNAME.

FINALSTI uses PSBs for all of the possible Finalist and ancillary databases. If you do not have an option for an ancillary database, please provide a dummy PSB (duplicate a previously used file) and make sure your applications do not access that file.

Note: If you access the file but your PSB is not pointing to the proper database, unpredictable results will occur which may or may not include z/OS or IMS system ABENDs.

Verifying the Finalist IMS On-Line Installation

If you decide to install the IMS component of Finalist, refer to the section “Using Finalist IMS” in your Finalist User’s Guide for information on verifying your Finalist IMS installation using the sample transactions S56LPCH and S56LPWNH.

Finding Database File Information

You can find information for the Finalist database files, including record counts, in the Finalist Technical Bulletin. To access the Finalist Technical bulletin, go to the estore and log in with your User ID and password. You can use the information in the Finalist Technical Bulletin to calculate the required number of cylinders/tracks for the database files.

Calculating the Number of Cylinders/Tracks

This section provides instructions for calculating the number of cylinders/tracks required for a given Finalist VSAM cluster. The number of cylinders/tracks required varies according to type of DASD, record size, and control interval size. The actual number of records in the Finalist VSAM clusters also varies slightly with each new update or distribution. Refer to the IBM VSAM Administration Guide to find capacity and physical record sizes by device type.

The Technical Bulletin, available on the Precisely website, shows the record counts for each of the files. Use the information from the Technical Bulletin and the following formula to calculate the number of cylinders/tracks that a given VSAM cluster requires.

1. Subtract 8 from the control interval size of the file. Divide this value by the record size (LRECL). The result is the number of logical records per VSAM physical record.
2. Multiply the result from step 1 by the number of physical records that fit on one track. The chart in the IBM VSAM Administration Guide (z/OS users) shows the number of physical records that fit on one track. The result is the total number of VSAM physical records in the file.
3. Divide the total number of records in the file by the result from step 2. The result is the number of tracks required for the file. To find the number of cylinders required, divide the number of tracks required by the number of tracks per cylinder.

The following sample calculation demonstrates how to calculate the number of cylinders/tracks for a 17,479 record VSAM cluster having a record size of 4088 and a control interval size of 4096 on 3390 DASD.

```
1.   4096 (VSAM control interval size)
  -     8 (VSAM overhead)
     4088 (Number of data bytes per physical record)
  /  4088 (Divide by record size)
        1 (Number of logical records per physical record)

2.      1 (Number of logical records per physical record)
  X    12 (Physical records per track for 3390 DASD)
       12 (Number of logical records per track for 3390 DASD)

3.  17479 (Total number of records in the file)
  /    12 (Number of logical records per track for 3390 DASD)
     1457 (Total number of 3390 tracks required)
  /    15 (Tracks per cylinder for 3390)
       98 (Number of cylinders required)
```

6 - Installation Notes and Tips

In this section

CASS vs. Non-CASS Installation

Finalist allows you to turn USPS CASS-certified processing on and off. This feature does not affect the way that addresses are corrected. It simply gives you the option of avoiding some USPS rules which are designed for high-volume mailers. In order to receive postal discounts for a volume mailing, a mailer is required to bring the USPS Form 3553 (CASS Summary Report) that was generated by CASS-certified software to the post office.

Regulations require that current address validation data be used. If CASS-certified processing is turned on, the database files received with Finalist will expire four months after the month listed.

For example, the February database files will expire on June 1. The system will issue an error message when you attempt to initialize Finalist (via calling PBFNInit) if the database has expired.

If you are not processing to achieve postal discounts and do not need the USPS Form 3553 (CASS Summary Report) to submit with your mailing (required in order to receive a postal discount), then you have the option of running in a non-CASS mode and turning off the LACS^{Link}, Suite^{Link}, and DPV options. Your addresses will still be processed using CASS regulations, but Finalist will not generate the USPS Form 3553 (CASS Summary Report). Running in non-CASS mode prevents your application from receiving the error message from PBFNInit indicating the database has expired.

> **Note:** The Finalist product only runs in CASS mode as defined by the USPS. The Finalist product can continue to run in a non-CASS mode outside of the window set by the USPS.

CASS vs. Non-CASS Technical Notes

This information will help you decide whether or not your application should be run in a CASS-certified mode. When the PBFNInit API is called, Finalist verifies CASS mode based on the CASS Flag setting in the pbfn.cfg file. The contents of this file indicates to Finalist whether or not to proceed as CASS-certified software.

This field determines whether Finalist checks expiration dates for bases and engines. For batch processing, cCASSFlag=ON ensures that CASS-required options are turned on including DPV, LACS^{Link}, Suite^{Link}, CASS configuration, Carrier Route (CR), and Delivery Point (DPBC).

The two areas affected are:

- At PBFNInit time, Finalist validates the postal coding database files. The data maintenance dates contained within the files must be within the range indicated by the USPS for CASS certification.
- Finalist only generates the USPS Form 3553 (CASS Summary Report) if the system is running in CASS-certified mode.

Note: If CASS Flag = ON and a conflicting option is encountered (Configuration, Assign CR, Return DPBC, LACS^{Link}=OFF, Suite^{Link}=OFF, or DPV=OFF), a warning message is written to the log file indicating that CASS has been forced off and the reason for CASS being forced off. The message is similar to: Warning Message;CASS forced off: CASS Configuration, Return DPBC, Assign CR, Assign SuiteLink, Assign LACSLink, Assign DPV

Database Compatibility Error Message

If you attempt to use a database that is incompatible with the product version, the following error message is written to the log file.

```
Database and Engine Version differ;DB version (MMMMMM01) != Engine version
 (NNNNNN01)
```

Database Expiration Warning Message

If you are running in CASS-certified mode and the database is within 10 days of expiration (30 days on a mainframe), PBFNInit will return a value of PBFN_HAVE_WARNING.

Engine Expiration Warning Message

The Finalist engine will expire soon. PBFNInit returns the value PBFN_ENGINE_WARNING. Finalist issues the following warning message. This warning message displays on all platforms 1 month before the key expires.

```
CASS ENGINE expires on MM-DD-YYYY
```

Performance Notes

To fine-tune the Finalist system for your environment, keep in mind the following items which may affect performance.

- When running in batch mode, sort your input file based on ZIP Code, then State, then City.
- The system uses the "cache size" parameter in the pbfn.cfg file to determine how much cache to use. This is the maximum number of 4K buffers Finalist can use as internal database cache. If your input file is sorted by ZIP Code, the larger the cache size the faster Finalist processes. If your input file is not sorted, use a small cache size to cut down on input/output time for each ZIP Code change.
- For information on maximizing database performance, refer to **Finalist Databases** on page 57.

Finalist Quick Start

Now that your software is installed, start your work with Finalist as described in these steps.

1. Refer to the Finalist Reference Guide for information on the application program interfaces (APIs) available with the Finalist product.
2. Experiment with the Workbench and the Lookup Tool (not distributed for Unix). This step lets you see how the product's features operate.
3. Design and code your application.
4. Compare the results you achieve with your application to results generated by the tools. For example, if your application cannot postal code an address, use the Lookup Tool to help determine why your application could not code the address.
5. Verify that the pbfn.cfg file contains correct paths to the postal database files city.dir and zip4us.dir.
6. If you are running Finalist under Windows, perform the following steps to become familiar with Finalist features.
   a) Look at the program group created by the Finalist installation procedure.
   b) Click on Lookup.
   c) The first time you run the Lookup application, it will bring you to the configuration dialog. In the Files tab, locate the database files available to your PC, in the Product tab, enter the Software key shipped with the product.
   d) Click OK.
   e) You will see four icons in the main window.
   f) Click Postal Code.
   g) Enter your street address, city and state, and click Code.

If your address is able to be verified against the national address data contained within the database files, your address will be displayed in a standardized format, along with other relevant addressing information, such as ZIP+4, carrier route, etc.

7 - Finalist Databases

In this section

Introduction

This chapter provides information on the Finalist databases including notes and tips to help you maximize your system performance when processing with the Finalist databases.

Finalist Databases

Table 10: Finalist Databases

Database	Required or Recommended for CASS Processing	Description
CITYFILE	Required	Provides basic address matching.
DATAFILE	Required	Provides ZIP + 4 address matching.
EWSFILE	Recommended	The Early Warning System (EWS) database provides early alerts to address changes that could impact your address file. For example, your input file contains the address 123 MAIN ST. The ZIP4 DATABASE only contains MAIN RD. Since MAIN RD is the only entry in the ZIP + 4 database, without EWS processing, the input address would be changed to MAIN RD. Finalist can read the EWS database to determine that a new address MAIN ST has been created. Finalist does not change the input address to MAIN RD ensuring that mail is not delivered to the wrong location. Each monthly database ships with a copy of the EWSFILE database. However, you are encouraged to obtain the most current information available from the USPS web site. Visit **https://postalpro.usps.com/** and look for file EWS002C0.ZIP.
LOTFILE	Recommended	The Enhanced Line Of Travel or eLOT database provides routing information for your coded addresses. While eLOT processing is not required for CASS certification, it is required to obtain discounts. To accomplish this, perform eLOT processing as part of your address hygiene processing or separately as part of your presort processing.

Database	Required or Recommended for CASS Processing	Description
DPVxDB	Required	The Delivery Point Validation (DPV) database provides point specific information about addresses. The ZIP4 databases match addresses to a range. For example, for the address “100-200 N MAIN ST.”, DPV further qualifies the address to identify 101 as a valid delivery point where 103 is not. DPV has three (3) formats of its data: **Full (often called hash)** Use DPV Full when storage of the databases is the most critical factor. The DPV Full database requires about 1 GB of disk storage. **Split** Use DPV Split for a medium storage factor. The DPV Split file requires about 1.6 GB of disk storage. **Flat** Use DPV Flat for the largest storage factor. The DPV Flat file requires about 2.3 GB of disk storage. USPS CASS regulations require DPV processing for CASS certification. If you do not perform DPV processing, Finalist does not generate a USPS Form 3553 (CASS Summary Report).
LLKDB	Required	The LACS^{Link} database provides address conversion. For example, the old style address “RR 1 BOX 123” should be converted to “604 S 450 W” for a more accurate delivery of the mailpiece. This is often referred to as the E911 database since it allows emergency personnel to more accurately identify the address location. USPS CASS regulations require LACS^{Link} processing for CASS certification. If you do not perform LACS^{Link} processing, Finalist does not generate a USPS Form 3553 (CASS Summary Report).
PRCSLYDB	Optional	For each addressable location, PreciselyID processing provides a unique and persistent identifier to reference the address without storing the whole address string. The PreciselyID is further verification an address exists, regardless of DPV matching. Additionally, to link it to other data (i.e., geoenrichment), Precisely has done the complex matching for you. Use the PreciselyID as the gateway to the rest of the Precisely data portfolio. Contact your Sales Representative for more information, based on your specfic needs.
RDIDB	Recommended	The Residential Delivery Indicator (RDI) Option is designed to identify if an address is a residential (RDI=Y) or a business (RDI not equal to Y) address.

Database	Required or Recommended for CASS Processing	Description
SLKDB	Required	The SuiteLink database provides more accurate matching for firms and businesses. For example, WIDGET COMPANY; 2200 WESTERN CT; LISLE IL 60532 is missing the unit (suite) information to accurately deliver the mail. SuiteLink provides the ability to look into the address file and determine that firm WIDGET COMPANY really belongs at a secondary range of STE 100. USPS CASS regulations require SuiteLink processing for CASS certification. If you do not perform SuiteLink processing, Finalist does not generate a USPS Form 3553 (CASS Summary Report).

Maximizing Performance

For best performance in a batch environment, DPV Flat is recommended with a Large memory model. Running with the Flat Large memory model will require about 50MB of additional virtual memory for the processing run. For CICS and IMS processing, Finalist forces a Pico memory model.

For LACS^{Link}, a Medium memory model is recommended if your system can provide the 250MB virtual memory requirement. A Small memory model will require only 35MB of virtual storage with a minimal performance loss. Ultra-Small will require only 1MB of virtual storage, but has significant performance loss.

For Suite^{Link}, a Large memory model will require 50MB of storage and provide excellent performance results. A Medium memory model will require only 7MB of storage and provides good performance results. Small and Ultra-Small will require only 1MB of storage, but have significant performance loss.

File Sizes

This table provides the approximate physical files sizes of the Finalist databases.

Database	Approximate Physical Size
Finalist City file	75 MB

Database	Approximate Physical Size
Finalist Data file	1.0 GB
EWS file	300 KB
eLOT	350 MB
DPV Flat	2.3 GB
DPV Full (hash)	1 GB
DPV Split	1.6 GB
LACS^{Link}	375 MB
PreciselyID file	3.5 GB
RDI file	48 MB
SuiteLink	525 MB

Processing Options

This table provides the recommended Finalist option settings for your platform.

Platform	Recommended Processing Settings
Mainframe Batch	Finalist cache buffers = 30 DPV FLAT with Large Memory Model LACS^{Link} with Small Memory Model Suite^{Link} with Medium Memory Model

Platform	Recommended Processing Settings
Windows and Unix	Finalist cache buffers = 12 DPV FLAT with Large Memory Model LACS^{Link} with Small Memory Model Suite^{Link} with Large Memory Model
CICS and IMS On-Line	Finalist cache buffers = off DPV FLAT with Pico Memory Model LACS^{Link} with Ultra-Small or Pico Memory Model Suite^{Link} with Pico Memory Model

Virtual Memory Requirements

This table provides the approximate virtual memory requirements for the DPV, LACS^{Link}, and Suite^{Link} databases.

Note: Results can vary by month.

Database	Setting	Virtual Memory Required
DPV FLAT NOTE: For Huge and Large settings, subtract: • 4M if not using the DNA table • 4M if not using the NSL table	Huge	2.2GB **Note:** Not recommended.
	Large	79MB
	Medium	51MB
	Small	1MB
	Ultra-Small	1MB
	Pico	0MB **Note:** Not recommended on mainframes.

Database	Setting	Virtual Memory Required
DPV Full	Huge	1003MB
Note: For Huge and Large settings, subtract: • 64M if not using CMRA • 256M if not using the No-Stat table • 64M if not using the Vacant table • 64M if not using the PBSA table • 4M if not using the DNA table • 64M if not using the Throwback table • 4M if not using the NSL table	Large	591MB
	Medium	115MB
	Small	1MB
	Ultra-Small	0MB
	Pico	0MB
DPV Split	Huge	673MB
Note: For Huge and Large settings, subtract: • 64M if not using CMRA • 256M if not using the No-Stat table • 64M if not using the Vacant table • 64M if not using the PBSA table • 4M if not using the DNA table • 64M if not using the Throwback table • 4M if not using the NSL table	Large	607MB
	Medium	67MB
	Small	4MB
	Ultra-Small	30KB
	Pico	0MB **Note:** Pico memory model does not load any files or indexes.
LACS[Link]	Huge	380MB
	Large	290MB
	Medium	250MB
	Small	35MB

Database	Setting	Virtual Memory Required
	Ultra-Small	1MB
	Pico	0MB
SuiteLink	Huge	525MB
	Large	50MB
	Medium	10MB
	Small	0MB
	Ultra-Small	0MB
	Pico	0MB

Note: USPS CASS regulations require DPV, LACS^{Link}, and Suite^{Link} processing for CASS certification. If you do not perform DPV, LACS^{Link}, and Suite^{Link} processing, Finalist does not generate a USPS Form 3553 (CASS Summary Report).

8 - Auxiliary Databases

In this section

What are the Finalist Auxiliary Databases?

Some Finalist options require additional databases and are required for CASS certification and generating the USPS Form 3553 (CASS Summary Report).

Option	Description	CASS Required?
DPV	The Delivery Point Validation (DPV) database contains data that enables you to determine if an address actually exists and whether the USPS actually delivers mail to an address. For information about DPV processing, refer to **Using DPV Processing** on page 78.	Y
EWS	The Early Warning System (EWS) database contains new address information that is in use but not yet available on the ZIP + 4 File. For information about EWS processing, refer to**Using EWS Processing** on page 76.	N
LACS^{Link}	The USPS LACS^{Link} database contains data on address conversions resulting from a 911 emergency response implementation. For information about LACS^{Link} processing, refer to **"Using LACSLink Processing"**.	Y
PreciselyID	The PreciselyID provides a unique identifier for an addressable location. For information about PreciselyID processing, refer to **Using PreciselyID Processing** on page 89.	N
RDI	The Residential Delivery Indicator (RDI) database contains data that indicates whether the USPS identifies addresses as residential or business addresses. For more information about RDI processing, refer to**Using EWS Processing** on page 76.	N
SuiteLink	The USPS SuiteLink database contains data on business addresses that were identified during CASS processing as high-rise default records with associated secondary information. For information about SuiteLink processing, refer to **Using SuiteLink Processing** on page 92.	Y
eLOT	The Enhanced Line of Travel (eLOT) database ensures that Enhanced Carrier Route mailings are closely sorted to the actual delivery sequence. For more information about eLOT processing, refer to **Using Enhanced Line of Travel (eLOT) Processing** on page 77.	N

Installing Auxiliary Databases

For information about installing the Finalist auxiliary databases, refer to the appropriate instructions for your platform.

- **Windows**
- **Unix**
- **Linux**
- **z/OS**

Activating Auxiliary Processing

To activate auxiliary processing, use one of these methods.

Method	Description
pbfn.cfg Configuration File	If you prefer to configure your Finalist installation using global settings, you can define the auxiliary processing in your pbfn.cfg configuration file. For more information about the pbfn.cfg file, refer to your Finalist User's Guide.
PBFNSetupDef Structure	If you prefer to configure your Finalist installation using the Finalist structures, you can define the auxiliary processing fields in the PBFNSetupDef structure. For more information about the PBFNSetupDef structure, refer to your Finalist Reference Guide.
Workbench or Lookup Tool	If you prefer to configure your Finalist installation using the Finalist Workbench or Lookup Tool GUI screens, you can define the auxiliary processing fields in the Product Tab on the PBFN Config Setting dialog box. For more information about the Finalist Workbench and Lookup Tool, refer to your Finalist User's Guide.
Compatibility Interface (CI)	If you prefer to configure your Finalist installation using the CI, you can define the auxiliary processing fields in the Finalist call area. For more information about the CI, refer to your Finalist Reference Guide.

Using the Configuration File to Activate Auxiliary Processing

To activate auxiliary processing using the pbfn.cfg file, complete these fields in your pbfn.cfg file. For more information about the pbfn.cfg file, refer to "Configuring Finalist" in the Finalist User Guide.

Table 11: Configuration File Auxiliary Processing Field Settings

Process	pbfn.cfg Field	Description
DPV	Delivery Point Validation	Indicate whether to perform Delivery Point Validation (DPV) processing.
	DPV Filepath	Define the path to the DPV file.
	DPVKey	Enter the DPV security key.
	DPV Shutdown Indicator	Indicate the action to take when encountering a DPV False Positive (Seed) violation during processing.
	Delivery Point Validation Tie Break	Indicate whether to perform DPV Tie Break processing.
	DPV No-Stat Table	Indicate whether to use the No-Stat Table and return the proper No-Stat code to the output. Indicate whether to load the No-Stat table into memory or use the table outside of memory.
	DPV Vacant Table	Indicate whether to use the Vacant Table and return the proper Vacant code to the output. Indicate whether to load the Vacant table into memory or use the table outside of memory.
	DPV PBSA Table	Indicate whether to use the PBSA Table and return the proper PBSA code to the output. Indicate whether to load the PBSA table into memory or use the table outside of memory.
	DPV DNA Table	Indicate whether to use the Door Not Accessible (DNA) Table and return the proper DNA code to the output. Indicate whether to load the DNA table into memory or use the table outside of memory.
	DPV Throwback Table	Indicate whether to use the DPV P.O. Box Throwback Table and return the proper P.O. Box Throwback code to the output. Indicate whether to load the Throwback table into memory or use the table outside of memory.
	DPV NSL Table	Indicate whether to use the DPV No Secure Location (NSL) Table and return the proper NSL code to the output. Indicate whether to load the NSL table into memory or use the table outside of memory.

Process	pbfn.cfg Field	Description
	DPV Buffer Size	Specify the memory model to use for DPV processing.
	Commercial Mail Validation	Indicate whether to perform Commercial Mail Receiving Agents (CMRA) processing. Indicate whether to load the CMRA table into memory or use the table outside of memory.
LACS^{Link}	LACSLink	Indicate whether to perform LACS^{Link} processing.
	LACSLink Processing	Specify the memory model for LACS^{Link} processing.
	LACSLink Filepath	Define the path to the LACS^{Link} File.
	LACSLink Key	Specify the LACS^{Link} security key.
SuiteLink	SuiteLink	Indicate whether to perform SuiteLink processing.
	SuiteLink Filepath	Define the path to the SuiteLink File.
	SuiteLink Shutdown Indicator	Indicate the action to take when encountering a SuiteLink processing error during the processing run.
	Return SLK Input Secondary	Indicate whether to return input secondary information when SuiteLink returns secondary information.
	SuiteLink Small Memory Flag	Indicate the memory model to use for SuiteLink processing.
EWS	EWS Filename	Specify the EWS file name and path.
	Early Warning System	Indicate whether to perform EWS processing.
RDI	RDI Filepath	Specify the RDI file name and path.
	Residential Delivery Indicator	Indicate whether to perform RDI processing.
eLOT	LOT Filename	Specify the LOT file name and path.
	Assign LOT	Indicates whether to assign LOT codes.
PreciselyID	PreciselyID Processing	Indicates whether to perform PreciselyID processing.

Using PBFNSetupDef to Activate Auxiliary Processing

To activate auxiliary processing using the PBFNSetupDef structure, complete these fields in the PBFNSetupDef structure. For more information about the PBFNSetupDef structure, refer to "Structures and Constants" in your Finalist Reference Guide.

Table 12: PBFNSetupDef Auxiliary Processing Field Settings

Process	PBFNSetupDef Field	Description
DPV	PBFN-GCFG-ASSIGNDPV cAssignDPV	Indicate whether to perform Delivery Point Validation (DPV) processing.
	PBFN-GCFG-DPVFILEPATH cDPVFilePath	Define the path to the DPV file.
	PBFN-GCFG-DPVKEY-NAME cDPVKeyName	Enter the DPV security key.
	PBFN-GCFG-ASSIGNDPVTIE cAssignDPVTie	Indicate whether to perform DPV Tie Break processing.
	PBFN-GCFG-DPVSHUTDOWNINDICATOR cDPVShutdownIndicator	Indicate the action to take when encountering a DPV False Positive (Seed) violation during processing.
	PBFN-GCFG-ASSIGNDPVNOSTAT cAssignDPVNoStat	Indicate whether to use the No-Stat Table and return the proper No-Stat code to the output.
	PBFN-GCFG-ASSIGNDPVVACANT cAssignDPVVacant	Indicate whether to use the Vacant Table and return the proper Vacant code to the output.
	PBFN-GCFG-ASSIGNDPVDNA cAssignDPVDNA	Indicate whether to use the Door Not Accessible (DNA) Table and return the proper DNA code to the output.
	PBFN-GCFG-ASSIGNDPVNSL cAssignDPVNSL	Indicate whether to use the DPV No Secure Location (NSL) Table and return the proper NSL code to the output.
	PBFN-GCFG-ASSIGNDPVTHRWBK cAssignDPVTHRWBK	Indicate whether to use the DPV P.O. Box Throwback Table and return the proper P.O. Box Throwback code to the output.
	PBFN-GCFG-ASSIGNDPVPBSA cAssignDPVPBSA	Indicate whether to use the PBSA Table and return the proper PBSA code to the output.
	PBFN-GCFG-ASSIGNCMRA cAssignCMRA	Indicate whether to perform Commercial Mail Receiving Agents (CMRA) processing.

Process	PBFNSetupDef Field	Description
	PBFN-GCFG-DPVBUFSIZE cDPVBufSize	Specify the memory model to use for DPV processing.
eLOT	PBFN-GCFG-LOTFILE-NAME cLOTFileName	Specify the LOT file name and path.
	PBFN-GCFG-ASSIGNLOT cAssignLOT	Indicate whether to assign LOT codes.
EWS	PBFN-GCFG-EWSFILE-NAME cEWSFileName	Specify the EWS file name and path.
	PBFN-GCFG-ASSIGNEWS cAssignEWS	Indicate whether to perform EWS processing.
LACS^{Link}	PBFN-GCFG-ASSIGNLACSLINK cAssignLACSLink	Indicate whether to perform LACS^{Link} processing.
	PBFN-GCFG-LACSLINKPROCESSING cLACSLinkProcessing	Specify the memory model for LACS^{Link} processing.
	PBFN-GCFG-LACSLINKFILEPATH cLACSLinkFilePath	Define the path to the LACS^{Link} File.
	PBFN-GCFG-LACSLINKKEY cLACSLinkKey	Specify the LACS^{Link} security key.
PreciselyID	PBFN-GCFG-ASSIGNPRECISELYID cAssignPreciselyID	Indicate whether to perform PreciselyID processing.
RDI	PBFN-GCFG-RDIFILEPATH cRDIFilePath	Specify the RDI file name and path.
	PBFN-GCFG-ASSIGNRDI cAssignRDI	Indicate whether to perform RDI processing.
SuiteLink	PBFN-GCFG-ASSIGNSUITELINK cAssignSuiteLink	Indicate whether to perform SuiteLink processing.
	PBFN-GCFG-SUITELINKFILEPATH cSuiteLinkFilePath	Define the path to the SuiteLink File.
	PBFN-GCFG-SUITELINKSHUTDOWN cSuiteLinkShutdown	Indicate the action to take when encountering a SuiteLink processing error during the processing run.
	PBFN-GCFG-RETSLKINPUTSECDRY cRetSLKinputSecdry	Indicate whether to return input secondary information when SuiteLink returns secondary information.
	PBFN-GCFG-SUITELINKSMALLMEM cSuiteLinkSmallMem	Indicate the memory model to use for SuiteLink processing.

Using the Workbench or Lookup Tool to Activate Auxiliary Processing

You can activate auxiliary processing using the Finalist Workbench or Lookup Tool.

To activate auxiliary processing using the Finalist Workbench or the Lookup Tool:

1. Launch the Finalist Workbench or Lookup Tool.
   a) Workbench — From the Tools menu, select PBFN Setup.
   b) Lookup Tool — From the Edit menu, select Config.
2. Select the Product tab on the PBFN Config Setting dialog box.
3. Complete the appropriate fields on the Product tab for the auxiliary processing to perform.

Table 13: Auxiliary Processing Field Settings - Product Tab

Process	Product Tab Field	Description
DPV	DPV Filepath	Define the path to the DPV file.
	Mode	Indicate whether to perform DPV processing:
	Tie Break	Indicate whether to perform DPV Tie Break processing.
	Shutdown Indicator	Indicate the action to take when encountering a DPV False Positive (Seed) violation during processing.
	Buffer Size	Specify the memory model to use for DPV processing.
DPV Tables	CMRA	Indicate whether to perform Commercial Mail Receiving Agents (CMRA) processing. Indicate whether to load the CMRA table into memory or use the table outside of memory.
	Vacant	Indicate whether to use the DPV Vacant Table and return the proper Vacant code to the output. Indicate whether to load the Vacant table into memory or use the table outside of memory.
	DNA	Indicate whether to use the DPV Door Not Accessible (DNA) Table and return the proper DNA code to the output. Indicate whether to load the DNA table into memory or use the table outside of memory.
	Throwback	Indicate whether to use the DPV P.O. Box Throwback Table and return the proper P.O. Box Throwback code to the output. Indicate whether to load the Throwback table into memory or use the table outside of memory.

Process	Product Tab Field	Description
	PBSA	Indicate whether to use the PBSA Table and return the proper PBSA code to the output. Indicate whether to load the PBSA table into memory or use the table outside of memory.
	No Stat	Indicate whether to use the No-Stat Table and return the proper No-Stat code to the output. Indicate whether to load the No-Stat table into memory or use the table outside of memory.
	NSL	Indicate whether to use the DPV No Secure Location (NSL) Table and return the proper NSL code to the output. Indicate whether to load the NSL table into memory or use the table outside of memory.
	DPV Key	Enter the Delivery Point Validation (DPV) security key.
LACS^{Link}	LACS Filepath	Define the path to the LACS^{Link} File.
	LACSLink	Indicate whether to perform LACS^{Link} processing.
	Processing	Specify the memory model for LACS^{Link} processing.
	LACS Key	Specify the LACS^{Link} security key.
SuiteLink	SuiteLink Filepath	Define the path to the SuiteLink File.
	SuiteLink	Indicate whether to perform SuiteLink processing.
	Memory	Indicate the memory model to use for SuiteLink processing.
	Shutdown Indicator	Indicate the action to take when encountering a SuiteLink processing error during the processing run.
EWS	EWS Filename	Specify the EWS file name and path.
	Early Warning System	Indicate whether to perform EWS processing.
RDI	RDI Filepath	Specify the RDI file name and path.
	Residential Delivery Indicator	Indicate whether to perform RDI processing.
eLOT	LOT Filename	Specify the LOT file name and path.

Process	Product Tab Field	Description
PreciselyID	Assign PreciselyID	Indicate whether to perform PreciselyID processing.

4. Select the Process tab on the PBFN Config Setting dialog box.
5. Complete the appropriate fields on the Process tab for the auxiliary processing to perform..

Table 14: Auxiliary Processing Field Settings - Process Tab

Process	Process Tab Field	Description
Suite^{Link}	Return SLK Input Secondary	Indicate whether to return input secondary information when SuiteLink returns secondary information.
eLOT	Assign LOT	Indicates whether to assign LOT codes.

Using the Compatibility Interface (CI) to Activate Auxiliary Processing

To activate auxiliary processing using the CI, define the appropriate field for your platform in the Finalist call area. For more information about the CI, refer to Chapter 3, Using the Compatibility Interface in your Finalist Reference Guide.

Table 15: Compatibility Interface (CI) Auxiliary Processing Field Settings

Process	pbfn.cfg Field	Description
DPV	FINAL-DPV-OPT caDPV	Indicate whether to perform Delivery Point Validation (DPV) processing.
	FINAL-DPV-SI caDPVSDI	Indicate the action to take when encountering a DPV False Positive (Seed) violation during processing.
	FINAL-DPV-BUFFER-SIZE caDPVbuf	Specify the memory model to use for DPV processing.
LACS^{Link}	FINAL-LLK-OPT caLLK	Indicate whether to perform LACS^{Link} processing and the memory model to use.

Process	pbfn.cfg Field	Description
SuiteLink	FINAL-SLK-OPT caSLK	Determines whether Finalist performs SuiteLink processing. If this field contains a "Y", Finalist performs SuiteLink processing.
EWS	FINAL-EWS-OPT caEWS	Determines whether Finalist performs Early Warning System (EWS) processing. If this field contains a "Y", Finalist performs EWS processing.

RDI	FINAL-RDI-OPT caRDI	To activate RDI™ processing for the Compatibility Interface (CI), specify the following when calling the CI:
		`05 FINAL-RDI-OPT PIC X(01).`
		`88 FINALIST-RDI-OFF VALUE 'N'.`
		`88 FINALIST-RDI-ON VALUE 'Y'.`

eLOT	FINAL-LOT-OPT caLOT	Determines whether Finalist performs LOT processing. If this field contains a "Y", the system performs LOT processing.

NOTE: The CI does not support turning on or off these options:

- DPV
  - CMRA processing
  - No Stat processing
  - Tie Break processing
  - Vacant Table processing
  - PBSA Table processing
  - Door Not Accessible (DNA) Table processing
  - DPV P.O. Box Throwback Table processing
  - DPV No Secure Location (NSL) Table processing
  - DPV Key (embedded in the CI)
- SuiteLink
  - Shutdown Indicator
  - Return Input Secondary
  - Small Memory Flag
- PreciselyID
  - PreciselyID processing

Using EWS Processing

New address information that is in use, but not yet available on the ZIP + 4 File, can be found as part of the CASS Department's Early Warning System (EWS). These new or changed addresses can be found at the USPS web site **https://postalpro.usps.com/**. The USPS updates the EWS File weekly. You can download the EWS File from the USPS web site. Precisely includes a monthly update of the EWS File with the database updates.

USPS CASS regulations require all CASS-certified software to be able to read the USPS EWS File. The Finalist EWS Option verifies input addresses that are not found in the current ZIP + 4 File against the USPS EWS File. If an input address is found in the EWS File, the input address is not matched to any similar addresses in the current ZIP + 4 File. Instead, the input address fails and is not coded until the ZIP + 4 File is updated with the correct address from the USPS EWS File.

For example, your input file contains the address "100 S. Bonnie Ct". Finalist does not find an exact match on the current ZIP + 4 File. The current ZIP + 4 File contains the address range 100 - 198 Bonnie Ave. Finalist would normally code the address as "100 Bonnie Ave". If Finalist finds values for "S Bonnie Ct" for the input ZIP Code in the EWS File, CASS regulations require Finalist to fail the address and not update the address until the USPS adds the valid address to the ZIP + 4 File.

How Does EWS Processing Work?

These steps describe the sequence of Finalist EWS processing.

1. During processing of your input file, Finalist identifies an address that is not an exact match to the current ZIP + 4 file.
2. Finalist compares the address to the EWS File.
3. If Finalist finds the inexact address on the EWS File, per CASS regulations, Finalist fails the address and does not update the address until the USPS adds the valid address to the ZIP + 4 File.
4. The number of failed EWS addresses displays on the Finalist Batch Report and the USPS Form 3553 (CASS Summary Report).
5. On the z/OS platform, there are two methods available for working with EWS — the CBEWS file and the PBFNEWS module.

   CBEWS CBEWS is a VSAM file that contains all of the EWS records from the USPS. It should be populated when the USPS releases EWS data (unless you always populate the PBFNEWS module, see below).

   PBFNEWS PBFNEWS is a module that may or may not contain the same EWS records from the USPS. The default configuration shipped with Finalist has no EWS records in

the PBFNEWS module. If you run the LOADEWS JCL (in FNSOURCE), LOADEWS populates the PBFNEWS module.

- If PBFNEWS is NOT populated when Finalist runs, Finalist reads CBEWS and stores the data internally.
- If PBFNEWS is populated when Finalist runs, Finalist does not need to read the CBEWS file. This method saves time and I/O counts, especially critical in an online (CICS or IMS) environment.
- If you are ALWAYS going to populate the PBFNEWS module (LOADEWS), you do not need to load the data into CBEWS. Also, in this case, the CBEWS DD does not need to be part of your JCL stream. NOTE: This method will provide the best performance of EWS processing.
- If you do not want to run LOADEWS, then you must load data into the CBEWS file.

Structures Containing EWS Information

These structures include data to facilitate EWS processing. For more information on these structures, refer to your Finalist Reference Guide.

- PBFN3553Def
- PBFNAddressDataDef
- PBFNIMSSetupDef
- PBFNSetupDef (includes three EWS setup parameters)
- PBFNStatsDef

Using Enhanced Line of Travel (eLOT) Processing

Enhanced Line of Travel (eLOT) sequence is an option for mailers who prepare carrier route mailings other than high-density/125-piece or saturation mailings. eLOT sequencing is required for Basic Enhanced Carrier Route Standard Mail except automation-compatible, letter-size pieces. eLOT sequence is not an exact walk sequence but a sequence of ZIP + 4 Codes arranged in the order that the route is served by a carrier. First the ZIP + 4 groups are sequenced. Then the addresses within each group are identified as being in ascending or descending order.

Finalist releases include a monthly eLOT database. The eLOT database ensures that Enhanced Carrier Route mailings are sorted much closer to the actual delivery sequence. The Finalist database and eLOT database must be in synch (for example, September eLOT data must be processed with

a September Finalist database). If the Finalist database and the eLOT database are not in synch, there may be ZIP + 4 Codes for which eLOT numbers cannot be assigned. The ZIP Code, ZIP + 4 code, carrier route code, and the delivery point of an address must be provided to assign a LOT code.

Assigning Line of Travel (eLOT) Codes

To assign eLOT codes, you must provide the LOT File name and activate the assignment indicator. For more information on activating eLOT processing, please refer to **Activating Auxiliary Processing** on page 67.

During the PBFNInit call, if you have not defined the name of the eLOT File or the open has failed, the PBFN_NOLOT return code is issued. This return code indicates that the Zip4us.dir and the City.dir Files were successfully opened and address processing can continue. However, eLOT codes will not be assigned.

If you did not set the cLOTCode [LOT_LEN] or the LOT pointer is null in the PBFNAddressDataDef structure, eLOT codes will not be assigned and returned for that record. If a record is successfully coded, but the call to the eLOT database failed to return a valid eLOT code, the PBFNProcess call will return PBFN_SUCCESS and issue an error code. For detailed information on eLOT error codes, see "Finalist Error Codes" in your Finalist User's Guide.

LOT Output

PBFNInit may generate this return value.

Return Value	Description
PBFN_NOLOT	The eLOT File has not been defined or the open has failed. The Zip4us.dir and the City.dir Files were successfully opened. Address processing can continue. eLOT codes will not be assigned. The value for this return code is 8.

Using DPV Processing

Finalist uses the Delivery Point Validation (DPV) database to match the addresses in your address file against USPS-provided data. The USPS data consists of low-to-high ranges for streets in the United States. If your input address falls within the low-to-high range for the input address street,

Finalist standardizes the address and assigns the appropriate ZIP Code, ZIP + 4 Code, carrier route code, and delivery point barcode for that range. However, this process does not ensure that the address actually exists or that the USPS actually delivers mail to the address. It only indicates that your input address falls within a known range for the input address street. According to USPS statistics, up to 10% of this coded mail is still undeliverable as addressed.

The DPV Option uses DPV data from the USPS to ensure your input file addresses are actual physical addresses to which the USPS delivers mail. Finalist DPV processing verifies the existence of an address to as fine a level as an apartment or suite.

> **Note:** USPS CASS regulations require Delivery Point Validation (DPV) processing for CASS certification and to generate the USPS Form 3553 (USPS CASS Summary Report).

DPV Structures

This table provides an overview of the DPV structures. For more information on these structures refer to Chapter 2, Structures and Constants in your Finalist Reference Guide.

Structure	Description
PBFNAddressDataDef	Returns DPV False Positive (Seed) Table detail record.
PBFNDPVHdrDef	Returns the Delivery Point Validation (DPV) Option Header record.
PBFNDPVStatsDef	Provides DPV processing statistics.
USPSDetailDef	Returns the detail data required by the USPS for each record creating a DPV False Positive (Seed) Table violation.
USPSDPVHdrDef	Returns the header data required by the USPS for DPV False Positive (Seed) Table violations.

Other Structures Containing DPV Information

These structures include data to facilitate DPV processing. For more information about these structures, refer to your Finalist Reference Guide.

- PBFNAddressDataDef (includes DPV indicators and footnotes)
- PBFNIMSSetupDef

Note: PBFNAddressDataDef includes the DPV indicators and contains the cDPVFootnote and sDPVFootnoteLen fields. These fields define the footnote codes returned during Delivery Point Validation (DPV) Option processing. These codes provide information on your processed input address.

- PBFNSetupDef (includes DPV setup parameters)

DPV Return Information

You can find the DPV processing return codes and footnote codes in the PBFNAddresssDataDef structure. This table displays the DPV return flags, DPV No-Stat indicator, DPV Vacant Table indicator, and DPV PBSA indicator.

Table 16: DPV Return Indicators

Field	Description
cDPVFlags	Character array containing the returned DPV indicators. • Byte 1 (DPV) **N** — Not a valid delivery point. The USPS cannot deliver mail to this address. **Y** — Delivery point validated. Primary range and secondary range (when present) are valid. The USPS can deliver mail to this address. **S** — Valid primary range. Secondary range is present but is not confirmed. The USPS can deliver mail to this address. **D** — Valid primary range. Secondary range is missing. The USPS can deliver mail to this address. • Byte 2 (CMRA) **Y** — The address is a valid Commercial Mail Receiving Agent (CMRA). **N** — The address is a confirmed delivery point but is not a valid CMRA. Blank — This field is blank if the address is not a confirmed delivery point. • Byte 3 (False Positive Flag) **Y** — The address is not a confirmed delivery point and a positive response is received from the False Positive File. **N** — The address is not a confirmed delivery point and a negative response is received from the False Positive File. Blank — The False/Positive Table was not queried. The address is a confirmed delivery point.

Field	Description
cDPVNoStatFound	DPV No-Stat Table status indicator. **Y** Found in the DPV No-Stat Table. **N** Not found in the DPV No-Stat Table.
cDPVPBSAFound	DPV PBSA Table status indicator. **Y** Found in the DPV PBSA Table. **N** Not found in the DPV PBSA Table.
cDPVVacantFound	DPV Vacant Table status indicator. **Y** Found in the DPV Vacant Table. **N** Not found in the DPV Vacant Table.
cDPVDNAFound	DPV Door Not Accessible (DNA) Table status indicator. **Y** Found in the DPV DNA Table. **N** Not found in the DPV DNA Table.
cDPVTHRWBKFound	DPV P.O. Box Throwback Table status indicator. **Y** Found in the DPV P.O. Box Throwback Table. **N** Not found in the DPV P.O. Box Throwback Table.
cDPVNSLFound	DPV No Secure Location (NSL) Table status indicator. **Y** Found in the DPV NSL Table. **N** Not found in the DPV NSL Table.

DPV Footnote Codes

This table displays the DPV footnote codes (cDPVFootnote).

Field	Description	
cDPVFootnote	Field containing the returned Delivery Point Validation (DPV) Option footnote codes. Finalist returns up to six two-byte footnote codes for each address.	
	A1	Input address did not match to the ZIP + 4 File.
	AA	Input address matched to the ZIP + 4 File.
	BB	Input address matched to DPV (all components).
	CC	Input address primary number matched to DPV but secondary number did not match (present but invalid).
	F1	Input address matched to a military ZIP Code.
	G1	Input address matched to a General Delivery address.
	M1	Input address primary number missing.
	M3	Input address primary number is invalid.
	N1	Input address primary number matched to DPV but address is missing secondary number.
	P1	Input address missing PO, RR, or HC Box number.
	P3	Input address PO, RR, or HC box number invalid.
	PB	Input address is a P. O. Box Street Address (PBSA).
	R1	Input address matched to CMRA but secondary number is not present.
	R7	Input address is a Carrier Route R777.
	RR	Input address matched to CMRA.
	U1	Input address matched to a unique ZIP Code.

Using LACS^{Link} Processing

LACS^{Link} processing provides you access to address conversion data resulting from 911 emergency response implementation. The resulting address conversion replaces rural addresses with city format

addresses to ensure emergency responders can locate the address when an emergency situation arises.

For example, an address is converted from a rural route/PO box address format to a number/street address format in order to receive 911 emergency response services. The USPS LACS^{Link} database provides a method for converting the old addresses. The USPS Change of Address (COA) database does not contain these addresses since these changes are actually conversions and not moves resulting in a change of address. Finalist queries the LACSLink database to convert appropriate addresses.

> **Note:** USPS CASS regulations require LACS^{Link} processing for CASS certification and to generate the USPS Form 3553 (USPS CASS Summary Report).

How Does LACS^{Link} Processing Work?

These steps describe the sequence of Finalist LACS^{Link} processing.

1. Finalist codes an address that USPS data indicates has undergone a LACS conversion. In this case, the cLACS field in the PBFNAddressDataDef structure contains "L" indicating a LACS conversion has taken place. Finalist sends the record through LACS^{Link} processing.
2. Finalist encounters a rural route address and cannot match the address to a Box number. Finalist codes the record to the rural route default level. Finalist sends any rural route record that cannot be matched to a Box through LACS^{Link} processing.
3. Finalist sends any address that fails postal coding through LACS^{Link} processing.
4. Finalist then processes the LACS^{Link} returned address through the coding engine using the LACS^{Link} returned address as the input address.

   > **Note:** For LACS converted addresses, the PBFNAddressDataDef fields are based on the LACS returned address as input. The PBFNAddressDataDef fields are not based on the original (pre-LACS) input address.

LACS^{Link} Structures

This table provides an overview of the LACS^{Link} structures. For more information about these structures, refer to Chapter 2, Structures and Constants in your Finalist Reference Guide.

Structure	Description
PBFNRtnLACSStatsDef	To pass LACS^{Link} processing statistics, pass the PBFNRtnLACSStatsDef structure on the PBFNStats or PBFNTerminate call.
PBFNLACSSeedHdrDef	To return LACS^{Link} False Positive violation header information, pass the PBFNLACSSeedHdrDef structure on the PBFNStats call.
USPSPBLACSHdrDef	The USPSPBLACSHdrDef structure returns the LACS^{Link} False Positive violation header data required by the USPS for LACS^{Link} processing in the USPS-required format.
PBFNAddressDataDef	To return the False Positive violation detail information for the USPS, pass the PBFNAddressDataDef structure on the PBFNProcess call.
USPSPBLACSDetDef	The USPSPBLACSDetDef structure returns the LACS^{Link} False Positive violation detail data required by the USPS. For each False Positive (Seed) Table violation, a record is created in the USPS-required format.

Other Structures Containing LACS^{Link} Information

These structures contain data to facilitate LACS^{Link} processing. For more information about these structures, refer to your Finalist Reference Guide.

- PBFNAddressDataDef (includes LACS^{Link} return code)
- PBFNIMSSetupDef
- PBFNInfoDef
- PBFNSetupDef (includes three LACS^{Link} setup parameters)
- PBFNStatsDef

LACS^{Link} Return Codes

If LACS^{Link} processing changes an address, the cLACSRtnCode field in the PBFNAddressDataDef structure contains one of these return codes.

Return Code	Description
A	LACS^{Link} processing successful. Record matched through LACS^{Link} processing.
00	LACS^{Link} processing failed. No matching record found during LACS^{Link} processing.
09	LACS^{Link} processing matched the input address to an older highrise default address. The address has been converted. However, rather than provide an imprecise address, LACS^{Link} processing does not provide a new address.
14	LACS^{Link} processing failed. Match found during LACS^{Link} processing but conversion did not occur due to other USPS regulations.
92	LACS^{Link} processing successful. Record matched through LACS^{Link} processing. Unit number dropped on input.

Resolving LACS^{Link} and DPV False Positives

This section provides information on resolving False-Positive Seed Violations that occur during LACS^{Link} and/or DPV processing.

What is a False-Positive Violation?

The USPS has put security measures in place to ensure mailers using DPV and LACS^{Link} processing do not use these applications to generate mailing lists. False Positive (Seed) records are artificially manufactured addresses provided as part of the DPV and LACS^{Link} options. If the USPS identifies a mailer as repetitively generating False Positive (Seed) violations, the USPS may direct Precisely to invalidate their license.

Towards that end, the USPS monitors addresses that generate a False Positive result. The USPS requires Precisely to report any organization generating a False Positive result during DPV and/or LACS^{Link} processing. If you generate a False Positive result, Finalist generates an error message indicating a False Positive (Seed) violation.

For the job that encountered the False Positive (Seed) violation, the CASS statement will be provided showing the number of records that were confirmed up until the point of the False Positive (Seed) violation. No records will be confirmed after the False Positive (Seed) violation occurred.

For any job set to perform CASS processing and submitted after a the False Positive (Seed) violation occurs, Finalist generates an initialization error and stops processing.

How Do I Know I Have Hit a Seed Violation

This section provides information for determining whether you have hit a seed violation.

Batch Processing

These steps describe the sequence if you encounter a seed violation using the batch application.:

1. The function (DPV or LACS^{Link}) generating the seed violation stops processing.
2. The Finalist job continues to run to completion if the DPV Shutdown Indicator is not set to “S”. Otherwise, a seed violation will stop the Finalist job.
3. An error message is written to the log.

Process	Seed Violation Message
DPV	20125 DPV Processing Error. Seed violation encountered. Call Technical Support.
LACS^{Link}	20152 LACS Processing Error. Seed Violation encountered. Call Technical Support.

4. Finalist generates the USPS Form 3553 (CASS Summary Report) for the job that encountered the False Positive (Seed) violation. It will be reflective of the records that were DPV/LACS^{Link} processed before the False Positive (Seed) violation.
5. Finalist writes the offending record to the SEEDLOG file. For Mainframe environments, your JCL includes a DD statement for SEEDLOG. The seed violation record is written to the location and filename assigned in this DD statement. For Windows and Unix environments, Finalist creates a "seedlog.txt" file in your /bin directory containing the seed violation.
6. In addition to producing a Seed Log, the output file indicates seeds by:
   a) If a DPV seed violation is encountered, the output file contains a "Y" for the DPV Flags False Positive indicator in position 3 of the three-byte output field oDPVPBSA.

b) If a LACS^{Link} seed violation is encountered, the output LACS^{Link} Seed Detail field oLLKSD is populated with a "Y" The oLLKSD field is not available to be posted out with the Workbench for Windows.

7. The function (DPV or LACS^{Link}) generating the seed violation cannot process subsequent jobs until a reactivation key is applied. Any CASS job submitted after the False Positive (Seed) violation will encounter an initialization error.

Calling Finalist

The Finalist engine (PBFN.dll) automatically creates the seedlog file. Calling driver programs are no longer responsible for creating the seedlog file:

Windows and Unix	The seedlog file generated is seedlog.txt.
z/OS	Define the following in your JCL:

```
//SEEDLOG  DD  DSN=hlq.SEEDLOG,
//         DISP=(MOD,CATLG),
//         DCB=(LRECL=180,BLKSIZE=0,RECFM=FB),
//         SPACE=(TRK,(1,1),RLSE)
```

What to do When You Encounter a Seed

If you encounter a seed violation:

1. Review the appropriate USPS information.
   a. For a LACS^{Link} seed violation, review the LACS^{Link} End User Licensee Performance Requirements document at **https://postalpro.usps.com/LL_EU_LPR**.
   b. For a DPV seed violation, review the USPS DPV Product Licensee Performance Requirements document at **https://postalpro.usps.com/DPV_LPR**.
2. Contact Precisely Technical Support via support.precisely.com.

Seed Log File

The Finalist engine (PBFN.dll) automatically creates the Seed Log file. Driver programs are not responsible for creating the Seed Log file. This applies to custom batch drivers calling FINALIST on the mainframe.

- If you perform Finalist processing using the Native or Compatibility Interface batch mode, Finalist generates the Seed Log and writes the Seed Log out to an output file that is defined using the DD SEEDLOG JCL statement on the Mainframe platform:

```
//SEEDLOG  DD  DSN=hlq.SEEDLOG,
//         DISP=(MOD,CATLG),
//         DCB=(LRECL=180,BLKSIZE=0,RECFM=FB),
//         SPACE=(TRK,(1,1),RLSE)
```

- For Windows and Unix, Finalist generates a "seedlog.txt" file automatically and places it in the /bin directory. The same Seed Log output file is used for both DPV and LACS^{Link} seed violations.

The formats for the Header and Detail records are identical for DPV and LACS^{Link}.

Structures Containing False Positive Violation Information

The structures listed below provide information on False Positive results generated during DPV and/or LACS^{Link} processing. For more information on these structures, refer to your Finalist Reference Guide.

Structure	Description
USPSDPVHdrDef	Returns the header data required by the USPS for DPV False Positive (Seed) Table violations in the USPS-required format.
USPSDetailDef	Returns the detail data required by the USPS for each record creating a DPV Option False Positive (Seed) Table violation in the USPS-required format.
USPSPBLACSDetDef	Returns the LACS^{Link} violation detail data required by the USPS. Finalist creates a record in the USPS-required format for each False Positive (Seed) Table violation.
USPSPBLACSHdrDef	Returns the LACS^{Link} False Positive violation header data required by the USPS for LACS^{Link} processing in the USPS-required format.

Using PreciselyID Processing

For each addressable location, PreciselyID processing provides a unique and persistent identifier to reference the address without storing the whole address string. The PreciselyID is further verification an address exists, regardless of DPV matching. If the optional PreciselyID database is installed, processing is available in Batch, IMS, and CICS. For detailed information about Finalist auxiliary processing, please refer to your *Finalist Installation Guide*.

Additionally, to link the PreciselyID to other data (i.e., geoenrichment), Precisely has done the complex matching for you. Use the PreciselyID as the gateway to the rest of the Precisely data portfolio. Contact your Sales Representative for more information, based on your specfic needs.

How Does PreciselyID Processing Work?

These steps describe the sequence of Finalist PreciselyID processing.

1. Finalist validates the addresses in your input file during processing.
2. Finalist looks up the output address in the PreciselyID database.
3. PreciselyID processing returns a flag indicating if a PreciselyID was returned, and if it applied to the full address.

Structures Containing PreciselyID Unique Identifiers

These structures include data to facilitate PreciselyID processing. For more detail, refer to your *Finalist Reference Guide*.

- PBFNAddressDataDef
- PBFNIMSSetupDef
- PBFNSetupDef

PreciselyID Return Information

To post the PreciselyID information using Finalist.exe, define this Definition File layout component keyword in your Definition File.

Definition File Keyword	Description
oPreciselyIDFound=x,y; a[,y]	Identifies the position (x) and length (y) of the indicator identifying if a PreciselyID was found.
oPreciselyID=x,y; a[,y]	Identifies the position (x) and length (y) of the returned PreciselyID.

PreciselyID Output

The PreciselyID output is returned in the PBFNAddressDataDef structure in the cPreciselyIDFound field.

Field	Description	
cPreciselyIDFound	Character identifying if a PreciselyID was found for the address	
	Y	Unique identifier PreciselyID was found for the full address.
	D	Unique identifier PreciselyID was found for the primary address (secondary information was dropped to find a match).
	N	Unique identifier PreciselyID was not found.
	Blank	PreciselyID database was not queried.
cPreciselyID	12-character unique identifier PreciselyID for the addressable location.	

Using RDI Processing

Finalist Residential Delivery Indicator (RDI) processing determines whether an address is a residential or business address.

For example, if a person runs a small business out of their home, RDI would identify that address as a residential address and not a business address. However, if the address is flagged as not residential, then a business is known to operate at this location.

How Does RDI Processing Work?

These steps describe the sequence of Finalist RDI processing.

1. Finalist validates the addresses in your input file during processing.
2. Finalist looks up the coded address in the RDI databases.
3. RDI returns a flag, through Finalist, indicating if this is a residential address.

Structures Containing RDI Information

These structures include data to facilitate RDI processing. For more information about these structures, refer to your Finalist Reference Guide.

- PBFNAddressDataDef
- PBFNIMSSetupDef
- PBFNSetupDef (includes two RDI setup parameters)

RDI Return Information

To post the Residential Delivery Indicator using Finalist.exe, define this Definition File layout component keyword in your Definition File.

Definition File Keyword	Description
oRdi=x,y; a[,y]	Identifies the position (x) and length (y) of the output Residential Delivery Indicator.

RDI Output

The RDI™ output is returned in the PBFNAddressDataDef structure in the cRDI field.

Field	Description	
cRDI	Character defining the Residential Delivery Indicator.	
	Y	Address is a residential delivery.
	N	Address is a business delivery.
	Blank	Failed address lookup (did not return a +4), or RDI was not active.

The CI returns the RDI output in this Finalist Return Area field for function 4, 5, 6, or 7 process calls.

COBOL Field Name/ C Field Name	Description
FINAL-RDI-RETURN-CODE caRDIReturnCode	Contains a "Y" if the matched address is a residential delivery. Contains an "N" if the matched address is a business delivery. This field is blank if the address fails address lookup, or the RDI™ Option is not active.

Using SuiteLink Processing

SuiteLink processing improves your business address information by adding secondary (suite) information to business addresses that CASS processing identified as highrise default records. Finalist uses the USPS SuiteLink database to correct the secondary (suite) information in the business addresses identified in the input file as highrise default records. Records that have been processed through CASS Certified™ ZIP + 4 matching software and identified as highrise defaults with secondary information are potential candidates for SuiteLink processing.

> **Note:** USPS CASS regulations require SuiteLink processing for CASS certification and to generate the USPS Form 3553 (USPS CASS Summary Report).

How Does SuiteLink Processing Work?

These steps describe the sequence of Finalist SuiteLink processing.

1. Finalist calls SuiteLink when the following conditions are met:
   - The Finalist configuration file (pbfn.cfg) indicates SuiteLink = ON and all other required SuiteLink parameters are defined with valid values.
   - Finalist successfully coded the address and the following information exists in the address record:
     - Firm name
     - Valid ZIP Code
     - Valid ZIP + 4 Code
     - Primary number exists
   - A match has been made to a highrise or street default record.
   - The Finalist database is current.
   - The SuiteLink database is current.
2. If SuiteLink returns secondary data, Finalist attempts another match using the corrected data.
3. Finalist checks the Return SLK Input Secondary configuration setting to determine how to return secondary information when SuiteLink secondary information is available.
4. Finalist prints statistics at end of job.

SuiteLink Structure

The following table provides an overview of the SuiteLink structure. For more information on this structure, refer to Chapter 2, Structures and Constants in your Finalist Reference Guide.

Structure	Description
PBFNRtnSuiteLinkStatsDef	To pass SuiteLink processing statistics, pass the PBFNRtnSuiteLinkStatsDef structure on the PBFNStats or PBFNTerminate call.

Other Structures Containing SuiteLink Information

These structures include data to facilitate SuiteLink processing. For more information on these structures, refer to your Finalist Reference Guide.

- PBFN3553Def
- PBFNAddressDataDef (includes SuiteLink return code)
- PBFNInfoDef
- PBFNSetupDef (includes three SuiteLink setup parameters)

SuiteLink Return Codes

If SuiteLink processing changes an address, the cSteLnkRtnCode field in the PBFNAddressDataDef structure contains one of these return codes.

Return Code	Description
A	SuiteLink processing successful. Record matched through SuiteLink processing.
00	SuiteLink processing failed. No matching record found during SuiteLink processing.

9 - Using the Distribution Tool

In this section

What is the Distribution Tool?

You can use the Distribution Tool State Cut feature to build a Finalist database that includes only selected states. These smaller state-specific files could increase the number of records processed per minute resulting in decreased processing time. You can use the State Cut feature from:

- Supplied z/OS JCL on the mainframe
- The Finalist Workbench
- A Windows or Unix platform command line

EXAMPLE

Your database includes data from fifty states, DC, and all United States territories. You only need data from two states to complete a current project. You can use the State Cut Feature to create a database file for just the two states you need on your hard drive. Finalist will have less data to search through during address assignment resulting in less processing time.

> **Note:** The Finalist City and ZIP + 4 database files are both required to use the State Cut feature. The State Cut feature creates new City and ZIP + 4 database files containing data for the requested states. These files must be used together for address assignment. The new zip4us.sc will not work with the original city.dir database file and the new city.sc will not work with the original zip4us.dir database file.

You can use the State Cut feature from the Finalist Workbench, from a Windows or Unix platform command line, or from JCL on the mainframe.

> **Note:** To use the database files generated by the State Cut program, you will need to edit the pbfn.cfg file to point to the new State Cut database files. For more information on the pbfn.cfg file, refer to "Configuring Finalist" in your Finalist User's Guide.

Before Using the State Cut Feature

Before you begin using the Distribution Tool State Cut feature, it is important to note:

- You must first install the Finalist database files onto a local or networked hard drive. If you are using the Finalist database files from a networked hard drive, you must verify that you have a connection to the networked hard drive where the files are located.
- The Finalist City and ZIP + 4 database files are both required to use the State Cut feature. The State Cut feature creates new City and ZIP + 4 database files containing data for the requested states. These files must be used together for address assignment. The new zip4us.sc will not work

with the original city.dir database file and the new city.sc will not work with the original zip4us.dir database file.

- To use the database files generated by the State Cut program, you must edit the pbfn.cfg file to point to the new State Cut database files. For more information on the pbfn.cfg file, refer to Chapter 2, Configuring Finalist in your Finalist User's Guide.

Using the State Cut Feature with z/OS JCL

The FNSOURCE library includes sample JCL to run the State Cut program.

Using the State Cut Feature From the Command Line

The Finalist installation program installs the State Cut program in the bin directory. To run the State Cut program using the default values:

1. Change the directory to the folder containing the Finalist database files (default is C:\Precisely\Finalist\db).

```
cd C:\Precisely\Finalist\db
```

2. Run the State Cut program.

```
"c:\Precisely\Finalistxxx\bin\statecut.exe"
```

3. Enter the state abbreviations for the states for the states to include in the new database files. When entering the commands via standard input, enter a blank line to end the list of states. You can enter state abbreviations in any order.
4. State Cut processing begins.

```
Processing ... please wait.
Press any key to continue
```

State Cut Usage Statement

The usage statement can be displayed by running the State Cut program with the -h or -? options.

```
Usage: StateCut.exe [-s statefile][-l logfile][-f refdb][-g rngdb]
                       [-o refOut][-z rngOut][-v logleve][-i][-h?]
where:
    -s statefile  = File containing list of state abbreviations.
    -l logfile    = Log file name.
    -f refdb      = Reference/City DB file name.
    -g rngdb      = Range/Zip4 DB file name.
    -o refOut     = Output Reference/City DB file name.
    -z rngOut     = Output Range/Zip4 DB file name.
    -v loglevel   = Set log level.
    -i            = Display state name and abbreviation info.
    -h            = Display usage.
    -?            = Display usage.
A list of state abbreviations must be specified either in a file or at
standard input.
The list of state names and abbreviations can be displayed using the -i
 option.
If the reference db and range db are not found, the configuration file
will be used.
There is no default state file name.
The default log file is pbfnlog.txt.
The default ref/city db file is city.dir.
The default range/zip4 db file is zip4us.dir.
The default output ref/city db is city.sc.
The default output range/zip4 db is zip4us.sc.
Options can be in upper case (e.g. -S is the same as -s), but file names
 are case sensitive.
See documentation for MF defaults.
```

State List File

The USPS state abbreviations are used as input to the State Cut program. A list of the state abbreviations and names can be displayed using the -i option at the command line. To perform the same State Cut process each month, a state list file can be created using any text editor or the Finalist Workbench Save button.

To use the state list file at the command line, use the -s statefile option where statefile is the name (and path if necessary) of the file containing the list of states. State abbreviations can be listed in

any order. An example of your input for running the State Cut program from the command line is shown next.

```
"c:\Precisely\Finalistxxx\bin\statecut.exe -s stateslist.txt"
```

The State Cut program status messages display to allow you to monitor processing.

```
Opened state file: stateslist.txt
Processing ... please wait.
Press any key to continue
```

Log File

The State Cut program generates a log file that contains information about the user selections and the run times. This is an example of a State Cut log file.

```
Start Time: 04 14 2016 09:13:57
States read from standard input.
Log file           : pbfnlog.txt
Ref/City db file   : X:\Finalist 9.2 DB\04-04-16 USPS2\city.dir
Range/Zip4 db file : X:\Finalist 9.2 DB\04-04-16 USPS2\zip4us.dir
Output Ref/City db : city.sc
Output Range/Zip4  : zip4us.sc
Log level          : 3
List of States...
IL
End Time: 04 14 2016 09:14:20
```

Log Level

The log level option of the command line (-v loglevel) controls the amount of information dumped to the log file. The log level 3 is the default. Values 0 through 5 are valid, where 5 displays the most information. This is an example of a short segment of additional information dumped at level 5.

```
09:33.30;INFO;dbaccess:0661;2396;0;Info;Opened ref/city db: X:\Finalist
 9.2 DB\04-04-16 USPS2\city.dir
09:33.30;INFO;dbapi:0169;2396;0;Info Message;   Reference Control Record
 - required - 304 bytes
09:33.30;DBG;dbapi:0329;2396;0;Debug Control Record;--- Reference database
 control record ----
09:33.30;DBG;dbapi:0332;2396;0;Debug Control Record;Version and Release
```

```
  : 9.20.00.M.01
09:33.30;DBG;dbapi:0338;2396;0;Debug Control Record;Copyright Date
  : 02-15-2016
09:33.30;DBG;dbapi:0345;2396;0;Debug Control Record;Build Time Stamp
  : 04-09-2016 06:09:00
```

Using the State Cut Feature in a Windows Environment

To create a new Finalist database for selected states, follow these steps:

1. Build a State List containing the names of the states you want to include in your new Finalist database.
2. Define the Path to the Finalist Database Files. The State Cut feature will get the data for your selected states from these database files. You must specify the location for both the City database and the ZIP + 4 database.
3. Define a name and path for the Finalist database files you want to create to contain the data for the states you selected in step 1.
4. Run the State Cut program to build your new database files.

To create a new Finalist database for your selected states, follow these steps.

5. Start the Finalist Workbench.
6. From the Tools menu, click Run Distribution or click the DIST icon on the Workbench toolbar. The State Cut Process dialog box displays.
7. To build a list of states to be included in the new database file, select states for the State List using one of the following options:

   - Click once on the state abbreviation and name in the left-side State List box. Click on the right arrow button.
   - Double click on the state abbreviation and state name in the State List box.
   - With the mouse cursor active in the State List box, type in the state abbreviation of the state you want to select. The state abbreviation and name appears in a highlighted mode. Click on the right arrow button to move the selected state to the Selected State box.

   The selected state appears in the Selected State box for selected states. Repeat for each state you want to include in the new database file to be created.

8. To remove states from the list of selected states in the Selected State box, use one of these options:

   - Click once on the state abbreviation and name in the Selected State box. Click on the left arrow button.

- Double click on the state abbreviation and state name in the Selected State box.
- To empty all selected files from the Selected State box, click Clear.

9. Click Save to save your state list file. The Open dialog pop-up appears. Use the Open dialog pop-up to create a new file or overwrite an existing file.
10. To load the your newly created state list, follow these steps:
    a) From State Cut Process Dialog Box, click Load.
    b) Use the Open dialog box to locate and select the state list file containing the states you want in your new Finalist database file. Your state list file appears in the Selected State box. After loading the file, you may remove or add states and save your new selections before running the state cut program.
11. To define the path to the Finalist database files to use for building your state database, follow these steps:
    a) In the City Base field, use the Browse button to identify the location of the Finalist City database file.
    b) In the Zip4 Base field, use the Browse button to identify the location of the Finalist ZIP + 4 database file.
12. To define a location and file name for the your state database file, follow these steps:
    a) In the New City Base field, use the Browse button to identify the name and location of the Finalist City database file.
    b) In the New Zip4 Base field, use the Browse button to identify the name and location of the Finalist ZIP + 4 database file.
13. After building and loading your state list file, defining the path to the Finalist database files, and defining the name and path for the state database file you want to create, you are ready to run the State Cut program. Click the Run button.

    If you decide not to run the state cut program, click Cancel to return to the Finalist Workbench window.

    Note: If the mouse cursor is active in one of the State List boxes, the run command will also be initiated when you click Enter on the keyboard.

10 - Glossary of Terms

In this section

Terms

3553 Report

USPS CASS Report.

API

Application programming interface (API). A set of routines, protocols, and tools for building software applications.

APO

Army Post Office. Mail for Army personnel is sent to one of several APOs in the United States. Each APO then forwards the mail to military bases throughout the world. Finalist does not process APOs as a conventional address. If Finalist does not find an exact match, the record will fail.

Barcode

An array of rectangular marks and spaces which appear in a predetermined pattern following unambiguous rules in a specific way to represent elements of data which are referred to as characters.

Base Street Name

The base street name is the street name that the USPS actually lists in the ZIP+4 postal file. For CASS certification purposes, your processing job should return the base street name. However, the USPS will accept either the alias street address or the base street address on the mail piece.

Carrier Route Code

A four-position code that designates the appropriate delivery route for a particular address. The USPS establishes carrier route coding schemes. Each scheme is ZIP specific.

CASS

USPS Coding Accuracy Support System.

Check Character

A character included within a symbol with a value used for the purpose of performing a mathematical check to ensure the accuracy of the data.

Check Digit

See Check Character.

CICS

Customer Information Control System.

City Delivery

A combination of delivery methods within a community where all residential and business customers are served according to postal regulations.

City Place Name

The name of a city, place, town, or other name by which a five-digit ZIP Code is commonly known.

CMRA

Private companies offering mailbox rental services to individuals and businesses are Commercial Mail Receiving Agents (CMRA). See also PMB.

Conventional Address

Address in which there is a street name and number. The street direction indicator (if any) in a strictly conventional address is between the street range number and the street name. For example, 1710 N FOREST RD is a typical address line from a conventional address.

Delimiter

A character that marks the beginning or end of a unit of data.

Delivery Point Barcode (DPBC)

A 14-digit barcode consisting of two framing characters, a five-byte ZIP Code, a four-byte +4 code, a two-byte delivery point, and a one-digit modulo check digit. Modulo is a term used to describe several packet-switched network parameters, such as packet number (for example, set to modulo 10, counted from 0 to 9). When the maximum count is exceeded, the counter is reset to 0.

Delivery Point Validation (DPV)

The Finalist Delivery Point Validation (DPV) Option uses DPV data available from the USPS to determine whether an address actually exists. The Delivery Point Validation (DPV) Option can verify the existence of an address to as fine a level as an apartment or suite. Mailers can use the Delivery Point Validation (DPV) Option to ensure the addresses in their address file are actual physical addresses to which the USPS delivers mail.

Deprecated

A term applied to features that are superseded and should be avoided. Although deprecated features remain in the current version, the use of deprecated features generates warning messages. Deprecated features will be removed in the future. Features are deprecated in order to give programmers using the feature time to bring their code into compliance with the new standard.

Directional

NE, West, etc.

DLL

See Dynamic Link Library (DLL).

DPV Door Not Accessible (DNA) Table

The Door Not Accessible (DNA) Table is a DPV® hash table that identifies addresses where carriers cannot knock on the door for mail delivery or where carriers cannot physically access a

residence/building. Some examples include rural/highway contact route (HCR), long driveway, or gated community addresses.

DPV No Secure Location (NSL) Table

DPV processing uses the No Secure Location (NSL) table to identify delivery locations that are not secure. For example, a carrier can access a door but cannot leave a package due to security concerns. The NSL designation alerts mailers to locations where businesses are closed on certain days and locations without mail receptacles (for example, a storefront).

DPV P.O. Box Throwback Table

DPV processing uses the P.O. Box Throwback Table to identify a delivery point that is a street address where mail is not delivered. Instead, delivery is made to the customer's P.O. Box address.

Dual Address

A dual address is an address that contains more than one mailable address (for example, an address that contains both a PO BOX and a street address).

DVD (or DVD-ROM)

DVD-ROM stands for "Digital Versatile Disk" or "Digital Video Disk", read only memory. A DVD holds 4.7 gigabytes of data. You cannot delete or update a file on a DVD-ROM.

Dynamic Link Library (DLL)

Dynamic Link Library called dynamically at execution time.

Early Warning System (EWS)

USPS CASS 2002-2003 regulations require all CASS-certified software to be able to read the USPS Early Warning System (EWS) File. The Finalist Early Warning System (EWS) Option verifies input addresses that are not found in the current ZIP+4 File against the USPS EWS File. If an input address is found in the EWS File, the input address is not matched to any similar addresses in the current ZIP+4 File. Instead, the input address fails and is not coded until the ZIP+4 File is updated with the correct address from the USPS EWS File.

Enhanced Line of Travel (eLOT)

Line of travel sequence is an option for mailers who prepare carrier route mailings other than high-density/125-piece or saturation mailings. eLOT sequencing is required for Basic Enhanced Carrier Route Standard Mail except automation-compatible, letter-size pieces. eLOT sequence is not an exact walk sequence but a sequence of ZIP+4 codes arranged in the order that the route is served by a carrier. First the ZIP+4 groups are sequenced. Then the addresses within each are identified as being in ascending or descending order.

False-Positive Violation

The USPS has put in place security measures to ensure mailers using DPV and LACS^{Link} processing do not use these applications to generate mailing lists. If the USPS identifies a mailer as repetitively generating false-positive violations, the USPS may direct Precisely to invalidate their license. Towards that end, the USPS has created and monitors addresses that generate a false-positive result. The

USPS requires Precisely to report any organization generating a false-positive result during DPV and/or LACS^{Link} processing.

Finance Number

The Finance Number consists of a state code (first two digits) and a postal installation code. There is a unique Finance Number for each post office name.

FIPS Code

Federal Information Processing Standards (FIPS) code. A FIPS code is a two-character state code followed by a three-character county code.

FPO

Fleet Post Office. Mail for Navy personnel is sent to one of several FPOs in the United States. Each FPO then forwards the mail to Navy bases throughout the world. Finalist does not process FPOs as a conventional address. If Finalist does not find an exact match, the record will fail.

Frame Characters

A special barcode character that provides the scanner with the start and stop instructions. Place the frame character at the beginning and ending of the Delivery Point barcode.

HC Highway Contract Route

HC provides for the transportation of mail between post offices or other designated points where mail is received or distributed.

IMS

Information Management System.

LACS^{Link}

The USPS LACS^{Link} database contains data on address conversions.

Last Line Information

The address last line information contains the city, state, and ZIP Code.

Load Library

The load library is the library where the load modules are stored.

Load Module

The load module is the executable program code.

Mailing Statement

A postal service form the mailer fills out, which lists the number of pieces of mail he or she is submitting at discount prices.

Modulo

A term to describe the adjustment value to bring a number up to the next multiple of its base number. For example, 13 modulo 10 has a value of 7. That is, you have to add 7 to 13 to bring it up to the next multiple of 20.

Mother ZIP

The term Mother ZIP is used to refer to the lowest ZIP Code within the finance area.

Nickname Alias Street Name

An alternate street name, maintained at the 5-digit ZIP Code level. It could be a name by which a street was formerly known, or a commonly used nickname for the street.

Non-Deliverable Address

Non-deliverable areas include vacant lots and land that borders railroad tracks, areas to which the USPS does not deliver mail.

Non-Mailing Name

A city name that is recognized by the USPS, but is not the preferred name for the ZIP Code. This is often a vanity name for the area.

Non-Parsed Address

Components of an address are combined into a single field. For example, an address 1 field might contain the Range, Street name, and Street suffix all separated by spaces. A last line address could consist of City, State, and/or ZIP all in the same field. With non-parsed address components, Finalist must spend additional time and resources to determine the individual components.

Parse

To analyze or separate into component parts.

Parsed Address Components

Components of an address are stored independently of each other. Components of a complete street address are some combination of: Range, Pre Directional, Street Name, Post Directional, Street Suffix, Unit Designator, Unit Range, Unit 2 Designator, Unit 2 Range, PMB Designator, PMB Range. Components of a last line are City, State, ZIP, ZIP+4, Delivery Point, Carrier Route, Advanced Bar Code. With Parsed Address components, Finalist does not need to determine the individual components.

PMB

A private mail box.

Point Of Entry

The point (post office) from which mail is entered (submitted).

Post-Directional

A geographic direction which follows the street name.

Pre-Directional

A geographic direction which precedes the street name.

PreciselyID

A unique and persistent identifier to reference the address without storing the whole address string. The PreciselyID is further verification an address exists, regardless of DPV matching.

Preferred Alias Street Name

Street names that are not standardized, that is, those addresses that include directional or suffix words as part of the street name, and not in their own fields. For example, a standardized address such as NE Military Sq would list NE in the pre-directional field, Military in the street name field and SQ in the suffix field. In contrast, the preferred alias street name would list Square in its non-standardized for as part of the street name (for example, Military Square).

Range

Section of a street or road normally identified with a number. For example, 1985 DOWNING LANE uses a number to denote where to deliver the letter or package on Downing Lane. If the address is other than conventional, the range field denotes the pertinent PO box or rural route number.

Region

The first digit of the ZIP Code that indicates the region of the country.

Residential Delivery Indicator

Residential Delivery Indicator (RDI) indicates whether an address is a residential delivery address (not a business delivery addresses).

Return Values

A code that a program or subroutine issues to indicate the status of the processing performed. For example, the subroutine passes a return code to the calling (driver) program to indicate whether to assign a carrier route code to a given address.

SCF

Sectional Center Facility, a major USPS mail collection and distribution center. For multi-ZIP cities, the first three digits of the ZIP Code the city indicate the SCF.

Sector Segment

The ZIP add-on, or, commonly +4 code. See also ZIP Sector Number and ZIP Segment Number.

Street Direction

Refers to the geographical location of any given street address (for example, North, South, East, or West).

Suffix

Normally, a word that follows the street name, indicating the type of street. The following are common suffixes to the street name in a conventional address: Boulevard, Road, Lane, Avenue, Highway, Court, Street, and Drive.

Suite^{Link}

The USPS Suite^{Link} database contains data on business addresses that were identified as high-rise default records during CASS processing. Finalist uses the USPS Suite^{Link} database to append the secondary (suite) information to business addresses identified in the input file as high-rise default records. Records that have been processed through CASS Certified™ ZIP + 4® matching software and identified as high-rise defaults are potential candidates for Suite^{Link} processing.

Three-Digit ZIP Prefix

The first three digits of the ZIP Code. These digits determine the appropriate SCF or postal facility to which the mail should be routed.

Unassigned Address

An address that does not have a sector segment number assigned by the post office.

Undefined Records

Records of varying length that do not contain a record length field. An undefined record is equivalent to a block.

Unique ZIP Code

A ZIP Code that is unique to a building or business.

Unit designator

Type of unit (for example, APT, STE, #, etc.)

URB

See Urbanization.

Urbanization

Denotes a sector, area, or development within a geographic area. Only used in Puerto Rico urban areas.

ZIP Code

The Zoning Improvement Plan (ZIP), established in 1963, is a system of five-digit codes identifying the individual post office or metropolitan area delivery station associated with an address. ZIP Code is a USPS trademark.

ZIP Sector Number

The ZIP sector number forms the first two digits of the ZIP add-on code. Geographically, a ZIP sector is a subdivision of a five-digit ZIP Code area. ZIP sector boundaries do not cross state or county lines.

ZIP Segment Number

The ZIP segment number forms the last two digits of the ZIP add-on code. The ZIP segment is a sub-division of a ZIP sector. Geographically, ZIP segments represent areas such as one side of a

city block between intersections; both sides of a street, including cul-de-sacs; a company or building; a floor or group of floors within a building; a cluster of mailboxes; sections of post office boxes; or other similar delivery groups.

ZIP + 4 code

The nine-digit code, established in 1981 is composed of:

Initial Code	the first five digits identifying the post office or metropolitan area delivery station associated with an address; a hyphen.
Expanded Code	Includes the additional four digits. The first two additional digits designate the sector (a geographic portion of a zone, a portion of rural route, several city blocks or a large building, part of a box section, or an official designation). The last two digits designate the segment (a specific block face, apartment house bank of boxes, a firm, a floor in a large building, or other specific location). ZIP+4 is a USPS trademark.
Zone	Last two digits of the ZIP Code; also, an area defined by the Postal Service for the purpose of establishing mailing rates. Mileage from a central mailing point determines all zones.

2 Blue Hill Plaza, #1563
Pearl River, NY 10965
USA

www.precisely.com