CHAPTER

16

Architectural Compatibility

Precision and Accuracy of Files That Are Moved between Host Architectures **119**
Host Groups **119**
Identical Architectural Groups **120**
Numeric Architectural Groups **120**
Character Architectural Groups **121**
Representing EBCDIC as ASCII or Hexadecimal Data **122**
Interpreting EBCDIC as ASCII Data **122**
Interpreting EBCDIC as Hexadecimal Data **123**

Precision and Accuracy of Files That Are Moved between Host Architectures

Regardless of the method that you use to transport a file that contains numeric data between host architectures, you may encounter problems with the precision of data values on some target hosts. This problem results from binary incompatibilities, such as differing methods for storing floating-point numbers, between host architectures.

Because transport files store numeric data in IBM floating-point format, data files that are created in IEEE-format lose precision when they are translated to transport format. UNIX, Windows, OS/2, and OpenVMS hosts store numeric data in IEEE-format while CMS and OS/390 hosts store numeric data in IBM format.

As an alternative to creating a transport file with a loss of numeric precision, you may prefer to create a CEDA file, whose internal representation does not compromise the precision of numeric data. For details about CEDA, see Chapter 8, "Using Version 8 Cross-Environment Data Access (CEDA)," on page 65. For details about numeric precision, see *SAS Language Reference: Concepts*. If you use SAS/SHARE to access SAS files, read about the topic cross-architecture access in "Managing Incompatible Client/ Server Host Types" on page 58 and *SAS/SHARE User's Guide*.

Furthermore, if you move a SAS file that contains character data from an ASCII-based system to an EBCDIC-based system and then back to the ASCII-based system, some print characters may be translated to values that are different from their original values.

Host Groups

Host types are categorized by identical architectural groups, numeric architectural groups, and character architectural groups. You *do not* need to create a transport file to move between two hosts that share identical architectures. However, you *must* create a transport file to move between two hosts that do not share identical architectures.

Note: This chapter also lists Version 6 hosts for backward compatibility. Hosts that run at Version 6 but not at Version 7 or later are:

- VSE
- Windows 32s
- Macintosh.

△

Identical Architectural Groups

The following host groups contain host types that share a common internal representation for both numeric and character data. Therefore, a data file *is not* translated when moved between hosts that do share a common internal data representation. However, data translation *is* performed on a data file that is moved between hosts that do not share a common internal data representation.

- IBM System/390 architecture hosts
  CMS
  OS/390
  VSE
- UNIX RISC hosts
  AIX for data sets and catalogs

  Note: Version 6 AIX hosts do not support catalogs. △

  HP-UX
  MIPS ABI
  Solaris
  SunOS
- Windows 32-bit hosts
  Windows NT
  Windows 98
  Windows 95
  Windows 32s
- Macintosh hosts
  Macintosh PPC
  Macintosh 68k
- Hosts that are incompatible with all other hosts
  Compaq Tru64 UNIX (formerly Compaq's DIGITAL UNIX)
  Intel ABI
  OpenVMS for Alpha
  OpenVMS for VAX
  OS/2.

Numeric Architectural Groups

The following host groups contain host types that share a common internal numeric representation. Therefore, numeric fields *are not* translated when moved between hosts that share a common internal numeric representation. However, numeric translation *is*

performed on numeric fields that are moved between hosts that do not share a common internal numeric representation.

- IBM System/390 architecture hosts

  CMS

  OS/390

  VSE

- IEEE-format, standard missing-value hosts (big endian)

  Note: Transporting a file from IEEE format to transport format results in loss of precision. For details, see "Precision and Accuracy of Files That Are Moved between Host Architectures" on page 119. △

  AIX

  HP-UX

  Macintosh PPC

  Macintosh 68k

  MIPS ABI

  OpenVMS Alpha

  Solaris

  SunOS

- IEEE-format, standard missing-value hosts (little endian)

  Compaq Tru64 UNIX (formerly Compaq's DIGITAL UNIX)

  Intel ABI

- IEEE-format, PC missing-value hosts

  OS/2

  Windows NT

  Windows 98

  Windows 95

  Windows 32s

- Unique internal representation in a category alone

  OpenVMS VAX.

Character Architectural Groups

The following host groups contain host types that share a common internal character representation. Therefore, character fields *are not* translated when moved between hosts that do share a common internal character representation. However, character translation *is* performed on a character fields that are moved between hosts that do not share a common internal character representation.

- EBCDIC format hosts

  CMS

  OS/390

  VSE

- ASCII-ISO format hosts

  AIX

  Compaq Tru64 UNIX (formerly Compaq's DIGITAL UNIX)

  HP-UX

Intel ABI
MIPS ABI
OpenVMS Alpha
OpenVMS VAX
Solaris
SunOS

- ASCII-ANSI format hosts

Windows NT
Windows 98
Windows 95
Windows 32s

- ASCII-OEM format host

OS/2

- ASCII-MAC hosts

Macintosh PPC
Macintosh 68k.

Representing EBCDIC as ASCII or Hexadecimal Data

Interpreting EBCDIC as ASCII Data

CMS and OS/390 hosts store character data in EBCDIC format. The following example shows SAS code that interprets the first portion of a transport file as ASCII data.

Note: This program does not convert the file to ASCII. It only interprets the first five records in the file as ASCII values and writes them to the SAS log. The transport file remains unchanged. △

Example Code 16.1 Code That Interprets an EBCDIC File as ASCII

```
//PEEK    JOB (,X101),'SMITH,B.',TIME=(,3)
/*JOBPARM FETCH
//STEP1   EXEC SAS
//transport-file DD DSN=USERID.XPT6.FILE,DISP=SHR
//SYSIN DD *
data _null_;
  infile tranfile obs=5;
  input theline $ascii80.;
  put theline;
run;
/*
```

Log output indicates whether the XPORT engine or PROC CPORT created the transport file.

Example Code 16.2 on page 123 shows the first 40 characters in ASCII text of a transport file that the XPORT engine creates.

Example Code 16.2 ASCII Text Displayed by the XPORT Engine

```
HEADER RECORD*******LIBRARY HEADER RECORD!!!!!!!00
```

Example Code 16.3 on page 123 shows the first 40 characters in ASCII text of a transport file that PROC CPORT creates.

Example Code 16.3 ASCII Text Displayed by PROC CPORT

```
**COMPRESSED** **COMPRESSED** **COMPRESSED** **COM
```

Note: If you set the NOCOMPRESS option to PROC CPORT, compression is suppressed, which prevents the display of the preceding text in a transport file. △

For technical details about the transport format that is used for a data set, see Technical Support article TS-140, *The Record Layout of a SAS Transport Data Set*.

Interpreting EBCDIC as Hexadecimal Data

Use an appropriate utility on your host to browse a transport file in hexadecimal format. For OS/390, you can use ISPF (which is an interactive structured programming facility), for browsing a transport file in hexadecimal format.

Alternatively, use the following SAS program to display in hexadecimal format the first twenty 80–byte records of a transport file

Example Code 16.4 Code That Interprets an EBCDIC File as Hexadecimal

```
data _null_;
  infile 'transport-file';
  input;
list;
put '--------------------';
  if _n_ > 20 then stop;
run;
```

Example Code 16.5 on page 123 shows the hexadecimal text of the first 40 characters in a transport file that the XPORT engine creates:

Example Code 16.5 Hexadecimal Translation of Text Created by the XPORT Engine

```
48454144455220524543 4F52442A2A2A2A2A2A2A
4C5920484541444552205245434F524421212121
```

This hexadecimal representation is equivalent to Example Code 16.2 on page 123.

Example Code 16.6 on page 123 shows the hexadecimal text of the first 40 characters in a transport file that PROC CPORT creates.

Example Code 16.6 Hexadecimal Translation of Text Created by PROC CPORT

```
2A2A434F4D505245535345442A2A202A2A434F4D
50442A2A202A2A434F4D505245535345442A2A20
```

This hexadecimal representation is equivalent to Example Code 16.3 on page 123.

The correct bibliographic citation for this manual is as follows: SAS Institute Inc., *Moving and Accessing SAS Files across Operating Environments, Version 8*, Cary, NC: SAS Institute Inc., 1999. 186 pages.

Moving and Accessing SAS Files across Operating Environments, Version 8

ISBN 1-58025-480-2

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513.

1st printing, October 1999