Introduction

In general, informats are completely portable. Only the informats that have aspects specific to OS/390 are documented in this chapter.

All informats are described in SAS Language Reference: Dictionary; that information is not repeated here. Instead, you are given details on how the informat behaves under OS/390, and then you are referred to SAS Language Reference: Dictionary for further details.

Considerations for Using Informats under OS/390

EBCDIC and Character Data

The following character informats produce different results on different computing platforms, depending on which character-encoding system the platform uses. Because OS/390 uses the EBCDIC character-encoding system, all of the following informats convert data to EBCDIC.
These informats are not discussed in detail in this chapter because the EBCDIC character-encoding system is their only host-specific aspect.

$ASCIIw.

converts ASCII character data to EBCDIC character data.

$BINARYw.

converts binary values to EBCDIC character data.

$CHARZBw.

reads character data and converts any byte that contains a binary zero to a blank.

$EBCDICw.

converts character data to EBCDIC. Under OS/390, $EBCDIC and $CHAR are equivalent.

$HEXw.

converts hexadecimal data to EBCDIC character data.

$OCTALw.

converts octal data to EBCDIC character data.

$PHEXw.

converts packed hexadecimal data to EBCDIC character data.

w.d

reads standard numeric data.

All the information that you need in order to use these informats under OS/390 is in SAS Language Reference: Dictionary.

---

**Floating-Point Number Format and Portability**

The manner in which OS/390 stores floating-point numbers can affect your data. See “Representation of Floating-Point Numbers” on page 139 for details.

---

**Reading Binary Data**

If a SAS program that reads and writes binary data is run on only one type of machine, you can use the following native-mode*informats:

IBw.d  
reads integer binary (fixed-point) values, including negative values, that are represented in two's complement notation

PDw.d  
reads data that are stored in IBM packed decimal format

PIBw.d  
reads positive integer binary (fixed-point) values

RBw.d  
reads real binary (floating-point) data

* Native-mode means that these informats use the byte-ordering system that is standard for the machine.
If you want to write SAS programs that can be run on multiple machines that use different byte-storage systems, use the following IBM 370 informats:

S370FFw.d
is used on other computer systems to read EBCDIC data.

S370FIBw.d
reads integer binary data.

S370FIBUw.d
reads unsigned integer binary data.

S370FPDw.d
reads packed decimal data.

S370FPDUw.d
reads unsigned packed decimal data.

S370FPIBw.d
reads positive integer binary data.

S370FRBw.d
reads real binary data.

S370FZDw.d
reads zoned decimal data.

S370FZDLw.d
reads zoned decimal leading sign data.

S370FZDSw.d
reads zoned decimal separate leading sign data.

S370FZDTw.d
reads zoned decimal separate trailing sign data.

S370FZDUw.d
reads unsigned zoned decimal data.

These IBM 370 informats enable you to write SAS programs that can be run in any SAS environment, regardless of the standard for storing numeric data. They also enhance your ability to port raw data between host operating environments.

For more information about the IBM 370 informats, see SAS Language Reference: Dictionary.

---

**Date and Time Informats**

Several informats are designed to read time and date stamps that have been written by the System Management Facility (SMF) or by the Resource Measurement Facility (RMF). SMF and RMF are standard features of the OS/390 operating environment.
They record information about each job that is processed. The following informats are used to read time and date stamps that are generated by SMF and RMF:

- **PDTIME**w.
  - reads the packed decimal time of SMF and RMF records.

- **RMFDUR**.
  - reads the duration values of RMF records.

- **RMFSTAMP**w.
  - reads the time and date fields of RMF records.

- **SMFSTAMP**w.
  - reads the time and date of SMF records.

- **TODSTAMP**.
  - reads the 8-byte time-of-day stamp.

- **TU**w.
  - reads the Timer Unit.

In order to facilitate the portability of SAS programs, these informats may be used with any operating environment that is supported by the SAS System; therefore, they are documented in SAS Language Reference: Dictionary.

---

**Dictionary**

**Ew.d**

- Reads numeric values that are stored in scientific notation
- Numeric
  - **Width range:** 7-32 bytes
  - **Default width:** 12
  - **Decimal range:** 0-31
  - **OS/390 specifics:** interprets input as EBCDIC, minimum and maximum values

**Details**

Numbers are interpreted using the EBCDIC character-encoding system, with one digit per byte. The range of the magnitude of acceptable values is from $5.4 \times 10^{-79}$ to $7.2 \times 10^{75}$. Any number outside this range causes an overflow error.

The following examples illustrate the use of the informat.
Informats

HEX

w.

Data Line  | Informat  | Value  
----------|-----------|--------
1.230E+02  | e10.      | 123    
-1.230E+02 | e10.      | -123   
1.230E+01  | e10.      | 12.3   
1.235E+08  | e10.      | 123,500,000

Note: In these examples, Data Line shows what the input looks like when viewed from a text editor. Value is the number that is used by SAS after the data pattern has been read using the corresponding informat.

See Also

- Informat: E in SAS Language Reference Dictionary
- Format: “Ew.” on page 168

HEXw.

Converts hexadecimal positive binary values to either integer (fixed-point) or real (floating-point) binary values

Numeric

Width range: 1-16 bytes
Default width: 8
OS/390 specifics: interprets input as EBCDIC, IBM floating-point format

Details

Under OS/390, each hexadecimal digit that is read by the HEX informat must be represented using the EBCDIC code, with one digit per byte. For example, the hexadecimal number '3B'x is actually stored in the external file as the bit pattern represented by 'F3C2'x, which is the EBCDIC code for 3B. (See Table 9.1 on page 142 for a table of commonly used EBCDIC characters.)

The format of floating-point numbers is host specific. See “Representation of Floating-Point Numbers” on page 139 for a description of the IBM floating-point format that is used under OS/390.

The w value of the HEX informat specifies the field width of the input value. It also specifies whether the final value is an integer binary (fixed-point) value or a real binary (floating-point) value. When you specify a width value of 1 through 15, the input hexadecimal number represents an integer binary number. When you specify a width of 16, SAS interprets the input hexadecimal number as a representation of a floating-point number.

The following examples illustrate the use of HEXw.d under OS/390.
### See Also

- Informat: HEXw.d in SAS Language Reference Dictionary
- Format: “HEXw.” on page 169
- “Representation of Numeric Variables” on page 139

### IBw.d

Reads integer binary (fixed-point) values, including negative values

**Numeric**

- **Width range:** 1-8 bytes
- **Default width:** 4
- **Decimal range:** 0-10
- **OS/390 specifics:** two's complement notation

**Details**

On an IBM mainframe system, integer values are represented in two's complement notation. If the informat specification includes a d value, the number is divided by $10^d$.

Here are several examples of the IBw.d informat:

<table>
<thead>
<tr>
<th>Data Line (Hex)</th>
<th>Informat</th>
<th>Value</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>FFFFFFFB2E</td>
<td>ib4.</td>
<td>-1234</td>
<td></td>
</tr>
<tr>
<td>000000003034</td>
<td>ib6.2</td>
<td>123.4</td>
<td></td>
</tr>
<tr>
<td>00000001E208</td>
<td>ib6.2</td>
<td>1234</td>
<td></td>
</tr>
</tbody>
</table>

Note: In these examples, Data Line (Hex) represents the bit pattern stored, which is the value you see if you view it in a text editor that displays values in hexadecimal representation. Value is the number that is used by SAS after the data pattern has been read using the corresponding informat.
representation. Value is the number that is used by SAS after the data pattern has been read using the corresponding informat.

See Also

- Informats: IBw.d, S370FIBw.d, and S370FPIBw.d in SAS Language Reference Dictionary
- Format: “IBw.d” on page 170

PDw.d

Reads data that are stored in IBM packed decimal format

Numeric

Width range: 1-16 bytes
Default width: 1
Decimal range: 0-31
OS/390 specifics: IBM packed decimal format

Details

The w value specifies the number of bytes, not the number of digits. If the informat specification includes a d value, the number is divided by 10^d.

In packed decimal format, each byte except for the last byte represents two decimal digits. (The last byte represents one digit and the sign.) An IBM packed decimal number consists of a sign and up to 31 digits, thus giving a range from -10^{31} + 1 to 10^{31} – 1. The sign is written in the rightmost nibble. (A nibble is 4 bits or half a byte.) A hexadecimal C indicates a plus sign, and a hexadecimal D indicates a minus sign. The rest of the nibbles to the left of the sign nibble represent decimal digits. The hexadecimal values of these digit nibbles correspond to decimal values; therefore, only values between '0'x and '9'x can be used in the digit positions.

Here are several examples of how data is read using the PDw.d informat:

<table>
<thead>
<tr>
<th>Data Line (Hex)</th>
<th>Informat</th>
<th>Value</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>01234D</td>
<td>pd3.</td>
<td>-1234</td>
<td></td>
</tr>
<tr>
<td>0123400C</td>
<td>pd4.2</td>
<td>1234</td>
<td>the d value of 2 causes the number to be divided by 10^2</td>
</tr>
</tbody>
</table>

Note: In these examples, Data Line (Hex) represents the bit pattern stored, which is the value you see if you view it in a text editor that displays values in hexadecimal representation. Value is the number that is used by SAS after the data pattern has been read using the corresponding informat.
See Also

- Informats: PDw.d and S370FPDw.d in SAS Language Reference: Dictionary
- Format: “PDw.d” on page 171

RBw.d

Reads numeric data that are stored in real binary (floating-point) notation

Numeric

Width range: 2-8 bytes
Default width: 4
Decimal range: 0-10
OS/390 specifics: IBM floating-point format

Details

The w value specifies the number of bytes, not the number of digits. If the informat specification includes a d value, the number is divided by 10^d.

The format of floating-point numbers is host-specific. See “Representation of Floating-Point Numbers” on page 139 for a description of the IBM floating-point format that is used under OS/390.

The following examples show how data that represent decimal numbers are read as floating-point numbers using the RBw.d informat:

<table>
<thead>
<tr>
<th>Data Line (Hex)</th>
<th>Informat</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>434CE00000000000</td>
<td>rb8.1</td>
<td>123</td>
</tr>
<tr>
<td>44300C0000000000</td>
<td>rb8.2</td>
<td>123</td>
</tr>
<tr>
<td>C27B000000000000</td>
<td>rb8.</td>
<td>-123</td>
</tr>
<tr>
<td>434D200000000000</td>
<td>rb8.</td>
<td>1234</td>
</tr>
<tr>
<td>41C4000000000000</td>
<td>rb8.</td>
<td>12.25</td>
</tr>
</tbody>
</table>

Note: In these examples, Data Line (Hex) represents the bit pattern stored, which is the value you see if you view it in a text editor that displays values in hexadecimal representation. Value is the number that is used by SAS after the data pattern has been read using the corresponding informat.
See Also

- Informats: RBw.d and S370FRBw.d in SAS Language Reference: Dictionary
- Format: “RBw.d” on page 172
- “Representation of Numeric Variables” on page 139

ZDw.d

Reads zoned decimal data

Numeric
Width range: 1-32 bytes
Decimal range: 0-32
OS/390 specifics: IBM zoned decimal format

Details

Like numbers that are stored in standard format, zoned decimal digits are represented in EBCDIC code. Each digit requires one byte of storage space. The low-order, or rightmost, byte represents both the least significant digit and the sign of the number. Digits to the left of the least significant digit are represented in EBCDIC code as ‘F0’x through ‘F9’x. The character that is printed for the least significant digit depends on the sign of the number. In EBCDIC code, negative numbers are represented as ‘D0’x through ‘D9’x in the least significant digit position; positive numbers are represented as ‘C0’x through ‘C9’x.

The following examples illustrate the use of the ZDw.d informat:

<table>
<thead>
<tr>
<th>Data Line (Hex)</th>
<th>Informat</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>F0F0F0F1F2F3F0C0</td>
<td>zd8.2</td>
<td>123</td>
</tr>
<tr>
<td>F0F0F0F0F1F2D3</td>
<td>zd8.</td>
<td>-123</td>
</tr>
<tr>
<td>F0F0F0F0F1F2F3C0</td>
<td>zd8.6</td>
<td>0.00123</td>
</tr>
<tr>
<td>F0F0F0F0F0F0F0C1</td>
<td>zd8.6</td>
<td>1E-6</td>
</tr>
</tbody>
</table>

Note: In these examples, Data Line (Hex) represents the bit pattern stored, which is the value you see if you view it in a text editor that displays values in hexadecimal representation. Value is the number that is used by SAS after the data pattern has been read using the corresponding informat. See Table 9.1 on page 142 for a table of commonly used EBCDIC characters.


See Also

- Informats: ZDw.d and S370FZDw.d, S370FZDLw.d, S370FZDSw.d, S370FZDTw.d, and S370FZDUw.d in SAS Language Reference: Dictionary and “ZDBw.d” on page 212
- Format: “ZDw.d” on page 174

ZDBw.d

Reads zoned decimal data in which zeros have been left blank

Numeric
Width range: 1-32 bytes
Decimal range: 0-32
OS/390 specifics: used on IBM 1410, 1401, and 1620

Details

As previously described for the ZDw.d informat, each digit is represented as an EBCDIC character, and the low-order, or rightmost, byte represents both the sign and the least significant digit. The only difference between the two informats is the way in which zeros are represented. The ZDBw.d informat treats EBCDIC blanks (’40’x) as zeros. (EBCDIC zeros are also read as zeros.)

The following examples show how the ZDBw.d informat reads data:

<table>
<thead>
<tr>
<th>Data Line (Hex)</th>
<th>Informat</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>40400400F14040C0</td>
<td>zdb8.</td>
<td>1000</td>
</tr>
<tr>
<td>40400400F1F2D3</td>
<td>zdb8.</td>
<td>-123</td>
</tr>
<tr>
<td>40400400F1F2C3</td>
<td>zdb8.</td>
<td>123</td>
</tr>
</tbody>
</table>

Note: In these examples, Data Line (Hex) represents the bit pattern stored, which is the value you see if you view it in a text editor that displays values in hexadecimal representation. Value is the number that is used by SAS after the data pattern has been read using the corresponding informat. See Table 9.1 on page 142 for a table of commonly used EBCDIC characters.
See Also

- Informats:
  - ZDB.w.d in SAS Language Reference Dictionary
  - “ZD.w.d” on page 211
- Format: “ZD.w.d” on page 174