Observations®
THE TECHNICAL JOURNAL
for SAS Software Users
www.sas.com/obs

# SAS® System Support for Asian Languages
# SAS® システムにおけるアジア言語サポート
# SAS® 系統對亞洲語言的支援
# SAS® 系统对亚洲语言的支援
# 아시아 언어를 위한 SAS® 시스템 지원

*Lee Richardson and Shin Kayano*

Lee Richardson is vice president of Asia/Pacific Latin America at SAS Institute. Lee has a BA degree from Duke University and an MA degree from the University of North Carolina at Chapel Hill. His areas of expertise include Asian language support, decision support systems, and balanced business scorecards. Lee has been a SAS software user for 20 years.

Shin Kayano is general manager of Asia/Pacific Research & Development at SAS Institute Japan Ltd. Shin supports double-byte character sets and internationalizing products. He has been using SAS software for 17 years.

## Abstract

Release 6.07 and later of the SAS System supports many Asian languages such as Japanese, Korean, simplified Chinese (used in mainland China and Singaport), and traditional or complex Chinese (used in Taiwan and Hong Kong) by generally supporting double-byte character sets, or DBCS. This article introduces DBCS systems and gives an overview of how SAS System products support these systems on various platforms and in various languages.

## Contents

# Introduction

This article is intended for SAS software users who need to use Asian languages with the SAS System. For example, an American pharmaceutical company may want its Japanese subsidiary to use a SAS application in English that the US headquarters developed. How can the application be converted into Japanese and what are the implications?

In another example, an employee of a Japanese company in the US may be responsible for a SAS application developed in Japan. How can the application be used by Japanese staff in the US, in a different hardware environment than in Japan? Can the application be ported to a different machine?

What about an American semiconductor manufacturer that wants its offices in Japan, Taiwan, and Korea to use a SAS/QC application that it has developed? How can the application be converted into the local language on a variety of hardware platforms?

Addressed here are the issues involved with DBCS applications and an illustration of how the SAS System supports DBCS. By *support*, we mean the ability of SAS software to interface naturally with vendors' implementations of Asian language systems. Such support includes

- using DBCS in the SAS Display Manager System and with other windowing products
- using DBCS in the SAS DATA step and SAS procedures
- using DBCS with SAS/GRAPH software and graphical procedures
- porting SAS software DBCS applications across platforms.

Support for DBCS in general should not be confused with the translation of the SAS System itself into various Asian languages. The Institute  has translated the windows and messages of several products such as SAS/ASSIST, SAS/AF, and SAS/FSP software into Japanese, and other translation projects are ongoing. These local language versions of SAS software products are available through local Institute subsidiaries.

However, such translations are not a prerequisite for using Asian languages with SAS software products. For example, a user in Japan can use the Japanese language with SAS/INSIGHT software, although SAS/INSIGHT software itself has not been translated into Japanese, as illustrated in Figure 1:

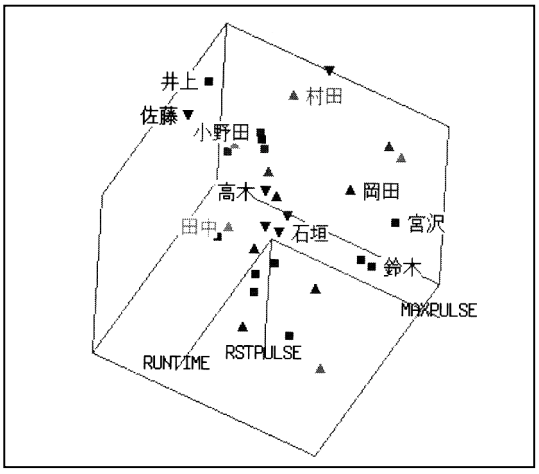


**Figure 1  SASUSER.FITNESS Data Set with Japanese Names Added**

As another example, even though the windows and messages of SAS/FSP software have not been translated into Chinese, a Chinese-speaking user could create a data entry application in Chinese that would accept

Chinese input. Or a Korean-speaking user could build Korean-language menus with SAS/AF software, even though SAS/AF windows and messages have not been translated into Korean.

The SAS System has no artificial intelligence capability that performs automatic translation between English (or any other Western language) and Japanese, Chinese, or Korean; you cannot use the SAS System to automatically translate your application into Japanese, Chinese, or Korean. Such automatic or *machine* translation, while possible, is beyond our scope here. This article focuses on how to use Asian languages with the SAS System.

The reader should have general knowledge of the SAS System and specific knowledge about the DATA step, windowing products, and SAS/GRAPH software. Knowledge about porting SAS applications across operating systems is helpful to understand parts of the article.

## What Is DBCS?

Western alphabets and special characters found on standard keyboards outside of Asia are represented by a single byte in either EBCDIC or ASCII. However, because of the sheer number of Chinese characters in Asian languages (Japanese and Korean both use Chinese characters, called *Kanji* in Japan and *Hanja* in Korea), 2 bytes are required to have enough binary combinations to represent them all. Japanese, for example, requires a minimum of around 2,000 binary combinations, and traditional Chinese requires up to 14,000.

Because 2 bytes are required to represent each character, these systems are referred to as double-byte character sets, or DBCS. Figure 2 illustrates the different hexadecimal codes:


**Figure 2**

Before Japanese, Chinese, or Korean can be used with SAS software, the underlying hardware and systems software must be in place. You cannot use these languages on an English or other European language system. All of the major hardware vendors such as IBM Corporation, Fujitsu, Inc., Hitachi, Digital Equipment Corporation, Sun Microsystems, Inc., and Hewlett-Packard Co. provide computer systems that support DBCS. Outside of Asia, such systems are available on a limited or special-order basis from the major hardware vendors.

These systems provide various methods to input DBCS characters into the computer. Generally speaking, the most common methods are phonetically based. Here, the user inputs a pronunciation of a Chinese character into the computer, often using English letters. The computer then pops up a menu of similar-sounding characters, and the user chooses one, with a single key stroke.

DBCS is usually implemented horizontally and left-to-right, like English and other European languages. The vertical, right-to-left style of writing used in traditional Japanese, Chinese, and Korean is not supported by the SAS System. In Asia, generally only desktop publishing software supports this traditional writing style.

On mainframe systems, DBCS support is intertwined with the 3270-style data stream. The DBCS character string is surrounded by escape codes called *Shift Out/Shift In, or SO/SI*. These codes originated from the

need, on old-style printers, to shift out from EBCDIC to the DBCS print train and to shift in to EBCDIC from the DBCS print train.

SO/SI surrounds the DBCS string, padding it with at least 1 byte on either side of the character string. This padding can cause problems, which we discuss later in this article.

The key mainframe vendors, IBM Corp., Fujitsu, Inc., and Hitachi, all have different encoding schemes for DBCS and for SO/SI. Hitachi, for example, uses 2 bytes each for SO/SI, for a total of 4 bytes.

Minicomputers, workstations, and PCs do not have SO/SI. However, like mainframe systems, the encoding schemes differ here among different platforms and operating systems. Generally, input is performed using the windowing systems such as Motif, DECwindows, OS/2 Presentation Manager, and MS Windows.

Differences exist among DBCS encoding schemes for various languages, such as Japanese, simplified Chinese, traditional or complex Chinese, and Korean, as well as among operating systems.

## The SAS System Approach to DBCS Support

The SAS System approach has been to support the most popular DBCS systems on the most popular platforms for a particular language. This support is accomplished by several system options and windowing and graphics drivers that support DBCS.

The three key system options used for general DBCS support are DBCS/NODBCS, DBCSTYPE=, and DBCSLANG=, all set at SAS System invocation.

### The DBCS/NODBCS System Option

DBCS turns on general DBCS handling, in batch or interactive mode, as well as for other products such as SAS/CONNECT software, discussed later.

### The DBCSTYPE= System Option

DBCSTYPE= specifies the encoding system in use. Few standards exist for DBCS encoding; most vendors have their own encoding scheme. Thus, the same character has a different hexadecimal code under each operating system, as shown in Figure 3:


**Figure 3**

Some possible values for DBCSTYPE= are shown in the following table:

| Platform | DBCSTYPE= Value |
|---|---|
| mainframe | IBM |
| | FUJITSU (or FACOM) |
| | HITACHI (or HITAC) |
| minicomputers | DEC and DG |
| UNIX systems: | EUC (for Sun, MIPS, NeXT, and SR10) |
| | DEC (for ULTRIX) |
| | DG (for DG/UX) |
| | HP15 (for HP-UX) |
| | PCIBM or SJIS (for AIX/6000) |
| PC DOS, Windows, OS/2 | SJIS (for all Japanese-language PCs) |
| | PCIBM (for IBM PCs) |
| | PCFCM (for Fujitsu PCs) |
| | PCHTC (for Hitachi PCs) |
| | PCMS (for PCs using Microsoft encoding) |

The values for DBCSTYPE= depend upon the environment in which the SAS System is executing.

DBCSTYPE= also provides the SAS System the flexibility of cross-system support, for example, using an X-terminalwith HP15 on a Sun server. Refer to the SAS Companion for each environment to determine the appropriate value for DBCSTYPE=.

## The DBCSLANG= System Option

The DBCSLANG= option provides support for different Asian languages that use DBCS. Possible values for DBCSLANG are

JAPANESE Japanese DBCS and single-byte English lowercase

KATAKANA Japanese DBCS and single-byte Katakana(a Japanese alphabet)

KOREAN Korean Hanja and English lowercase

HANGUL Korean Hanja and Hangul (the Korean syllabary)

CHINESE simplified Chinese

TAIWANESE traditional or complex Chinese

The differences among the DBCS languages are mainly the starting and ending hexadecimal code points for the DBCS character sets. For example, on IBM mainframe systems, DBCS (Japanese) begins at '4140'x and ends at '68FF'x, while DBCS (Chinese) begins at '4140'x, stops at '46FF'x, resumes at '4840'x, and ends at '6FFF'x.

Additionally, no common encoding scheme for Chinese characters has been agreed upon by the countries that use DBCS. Thus, the same character for "big" on IBM mainframe systems has a different hexadecimal encoding depending on whether the language in use is Japanese, Chinese, or Korean, as illustrated in Figure 4:


**Figure 4**

Here are some examples of these options used in various environments:

```
options dbcs                              /* turn on DBCS */
        dbcstype=IBM        /* IBM mainframe environment */
        dbcslang=JAPANESE;       /* using Japanese Kanji */

options dbcs                              /* turn on DBCS */
        dbcstype=HP15             /* HP UNIX environment */
        dbcslang=KOREAN;          /* using Korean Hangul */

options dbcs                              /* turn on DBCS */
        dbcstype=dec       /* DEC VMS/ULTRIX environment */
        dbcslang=TAIWANESE; /* using traditional Chinese */
```

## General Areas of DBCS Support

Among SAS software products, DBCS support can be divided into several areas:

- the DATA step and batch-oriented procedures
- windowing capabilities
- cross-system connectivity and compatibility
- access to databases
- graphics

## The DATA Step and Procedures

In the DATA step and batch-oriented procedures, DBCS can be used wherever a text string within quotes is allowed. Variable values, variable labels, and data set labels can all be in DBCS. Titles and footnotes may also be in DBCS. DBCS can also be used as input data and with range and label specifications in the FORMAT procedure.

**Note:** Variable names cannot be in DBCS; they must conform to standard SAS naming conventions.

The WHERE clause, which can be used in the DATA step and procedures, supports DBCS wherever quoted strings are allowed.

Here is an example of using DBCS in a batch-oriented SAS program:

```
   /* DBCS を用いたプログラムの例                    */
data sales(label='売り上げデータ');
   input region dept sales;
   label region='地区'
         dept='部門'
         sales='売り上げ';
   cards;
1 1 1200
1 2 340
1 2 860
2 1 830
2 2 1000
;

proc format;
   value regfmt 1='東京'
                2='大阪'
                3='名古屋';
run;

proc print label;
   title '地区別売り上げデータ';
   format region regfmt.;
run;
```

DBCS support at this level is available interactively or in batch mode.

On mainframe systems that use SO/SI, the SO/SI is stored as part of the data when input with the $CHARn. informat. Thus, the length of the character variable should be at least two bytes longer than the character string to allow space for the SO/SI.

As part of the SAS System support for DBCS, the SO/SI surrounding character strings is ignored when processing WHERE clauses. Thus, searches for embedded DBCS text are possible.

## DATA Step DBCS Functions, Formats, and Informats

Several DATA step functions, all prefixed with the letter K, have been developed to support the peculiarities of DBCS. KSUBSTR, KSTRCAT, and KTRUNCATE perform substringing, concatenation, and truncation of DBCS strings correctly, as shown in the following example:

```
data _null_;
   city='東京';              /* "Tokyo" in DBCS          */
      /* get first char of DBCS string                  */
   frstchar=ksubstr(city,1,1);
   put frstchar=;
run;
```

After you submit this code, the following line appears in the SAS LOG window:


Using the standard SUBSTR function would permit possible splitting of a DBCS character, which is undesirable. Using KSUBSTR and other DBCS-enabled functions ensures correct DBCS handling.

The following DBCS-enabled functions are available and correspond to the standard functions available in the DATA step:

| | |
|---|---|
| KCOMPRESS | KCOUNT |
| KINDEX | KLEFT |
| KLENGTH | KREVERSE |
| KRIGHT | KSCAN |
| KTRANSLATE | KSUBSTR |
| KUPDATE | KTRIM |
| KVERIFY | KUPCASE |

KSTRCAT performs the DBCS equivalent of the | | concatenation operator.

For cross-platform DBCS data conversion, such as converting a DBCS character string from IBM mainframe encoding to DEC encoding, the KCVT function is useful:

```
data null;
   infile ibm;
   file dec;
   input string $80.;
   decstr=kcvt(string,'IBM','DEC');
   put decstr;
run;
```

This code converts an 80-byte DBCS string from IBM encoding to DEC encoding.

In addition, several SAS formats and informats have been developed to handle DBCS on SO/SI systems more conveniently. Two of the most useful are $KANJI. and $KANJIX.:

| | |
|---|---|
| **Informats** | $KANJIX. adds SO/SI to DBCS |
| | $KANJI. removes SO/SI from DBCS |
| **formats** | $KANJI. adds SO/SI to DBCS |
| | $KANJIX. removes SO/SI from DBCS |

Since DBCS must have SO/SI to display correctly, the KANJI. and KANJIX. formats and informats are useful for storing large amounts of DBCS data without attaching the SO/SI.

## Handling Character Strings in Procedures

At the procedure level, care must be taken if character strings can be split across lines because the procedure might inadvertently split a DBCS character. The PRINT, REPORT, TABULATE, and FREQ procedures are particularly likely to split strings.

If undesired string splitting occurs, you should add spaces to strings or labels to move the split to a better location. With PROC REPORT and PROC TABULATE, the SPLIT= option may be used to force string splitting at the desired location.

The customization capabilities with PROC REPORT make it an excellent procedure to use with DBCS. The following example uses PROC REPORT and DBCS data to produce the results shown in Output 1:

```
proc report data=sql.union nowindows;
   title1 '1989-90 年度台灣、中國大陸、及香港地區 ';
   title2 '進出口貿易狀況分析報告I';

   column country item year,(import export netexp)
          import export totnet z;

   define country / group format=$char10. width=10 '地區';
   define item / group format=$char10. width=12
                 '進出口項目';
   define year / across format=foryr10. width=10 '- 年度 -';
   define netexp / computed format=dollar16. width=12
                   '淨出口額';
   define import / sum noprint;
   define export / sum noprint;
   define totnet / computed format=dollar16. width=16
                   '總出超金額';
   define z / computed format=$1. width=1 spacing=0 ' ';

   more SAS statements

run;
```

**Note:** Concerning the REPORT procedure, the default SPLIT character '/' is '61'x. Unfortunately, '61'x is used as part of many DBCS characters on mainframes. Thus, when using the SPLIT= option on mainframe systems, choose a character other than '/'. The blank character ('40'x) generally works without problems.

## Windowing Applications

The capabilities of the DATA step and most procedures described earlier are batch-oriented; however, the SAS System provides DBCS support for windowing and interactive capabilities such as the SAS Display Manager System (DMS), and SAS/FSP, SAS/AF, SAS/CALC, SAS/INSIGHT, and SAS/EIS software.

Users must specify the DBCS DBCSLANG= and DBCSTYPE= options at SAS System invocation to be able to use DBCS in interactive mode.

Figure 5 is an example of a windowing application developed using SAS/AF software. The menu is created just as an English-language menu would be created, only using DBCS input capabilities.

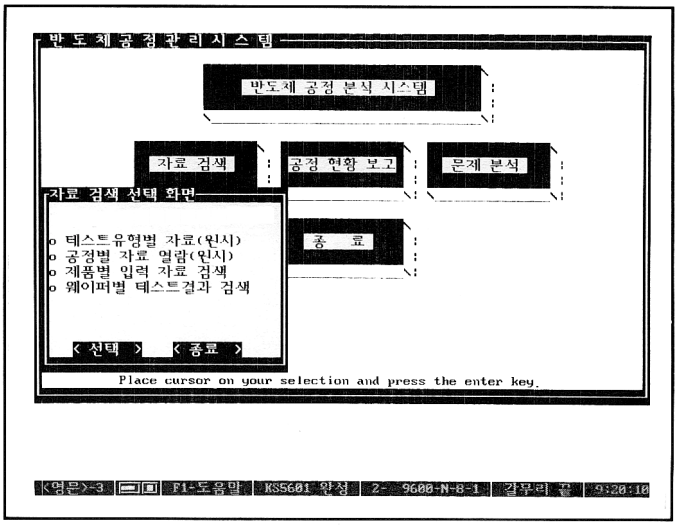


**Figure 5 Korean Language Used in a SAS/AF Application**

1989-90 年度台灣、中國大陸、及香港地區
進出口貿易狀況分析報告 I

| 地區 | 進出口項目 | 1989 淨出口額 | 1990 淨出口額 | 總出超金額 |
|---|---|---|---|---|
| | | — 年度 — | | |
| 中國大陸 | 五金 | $4,802,218 | $11,166,214 | $15,968,432* |
| | 汽車 | $-3,190,215 | $-1,559,002 | $-4,749,216 |
| | 其他 | $28,502,642 | $-34,227,074 | $-5,724,433 |
| | 紡織品 | $16,712,534 | $57,393,427 | $74,105,961* |
| | 農產品 | $46,022,257 | $32,768,999 | $78,791,256* |
| | 電子 | $-4,622,403 | $-8,569,438 | $-13,191,841 |
| | 電腦 | $-86,241,015 | $-20,187,773 | $-106,428,788 |
| | 電器用品 | $-3,163,190 | $1,164,248 | $-1,998,942 |
| | 鞋類 | $21,248,184 | $48,654,715 | $69,902,899* |
| 中國大陸 | | $20,071,013 | $86,604,316 | $106,675,329* |

**Output 1 Using DBCS Data with PROC REPORT**

# MVA Cross-System Support and DBCS Conversion

The SAS System provides DBCS conversion capabilities across hardware and operating systems. This conversion capability allows DBCS users to take advantage of the SAS System's MultiVendor Architecture (MVA) and application portability.

You can think of this support as an extension of the ASCII-EBCDIC conversion that occurs transparently when moving data across operating systems.

## The KCVT Function

First, the KCVT function described earlier allows users to convert data from one encoding scheme to another. KCVT could be used, for example, to convert DBCS data in an IBM DB2 database to DEC Rdb/VMS encoding to transport the data from an IBM to a DEC platform. Such data would normally be incompatible on another system. However, with the KCVT function, the SAS System could be viewed as a utility or bridge to convert data from one platform to another.

## The CPORT and CIMPORT Procedures

The SAS System also provides cross-system DBCS conversion capabilities via the CPORT and CIMPORT procedures.

In the following example, a SAS/AF application is converted from DEC DBCS encoding to IBM DBCS encoding:

```
proc cport catalog=my.lib.afsys.catalog
           intype=dec
           outtype=ibm;
run;
```

You cannot convert across languages. For example, it is meaningless to convert a Japanese DBCS application to a Korean DBCS application. As stated previously, the SAS System contains no artificial intelligence capability for cross-language translation.

**Note:** One word of caution concerning porting DBCS windowing applications across operating systems: when going from systems that do not use SO/SI (minis, UNIX systems, and PCs) to those that do (mainframes), the right side of the window may be truncated. The reason is that each DBCS string may be expanded by at least 2 bytes: 1 byte for the SO and 1 for the SI. This addition increases the width of the application and the right side of the display is truncated.

By default, if blank spaces are available, the spaces will be replaced by SO/SI, and no truncation occurs. Thus, when designing DBCS applications on non-SO/SI systems that will be ported to a mainframe, do not use up the entire display width.

For example, Figure 6a shows what happens when porting an FSEDIT window with DBCS from a mainframe to UNIX. The mainframe SO/SI is stripped off, reducing the length of the string. The SAS System fills in the space occupied by the SO/SI with blanks.

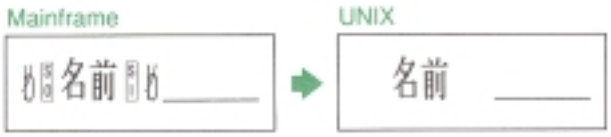


**Figure 6a: Porting from a Mainframe to UNIX: No problems Occur**

However, Figure 6b shows the problem that can occur when going from a non-SO/SI system to the mainframe. An attempt is made to port a 5-byte string, consisting of two characters (4 bytes) and one blank, from UNIX to the mainframe. Because the 2 bytes of SO/SI must be added, there is not enough space for the two characters and the added SO/SI. The SAS System automatically truncates a character and data are lost from the window.


**Figure 6b: Porting from UNIX to a Mainframe: Data Truncation Caused by a Lack of Space**

Figure 6c shows successful porting from UNIX to a mainframe with proper planning. In this case, the original space is 8 bytes long, with two characters (4 bytes) and four blanks. The four blanks permit expansion without truncation, even with the addition of the 2 bytes for SO/SI. The porting of the window is successful.


**Figure 6c: Porting from UNIX to a Mainframe: No Truncation with Proper Planning**

## SAS/CONNECT Software

SAS/CONNECT software can automatically convert data sets and catalogs containing DBCS when uploading and downloading data. It also correctly handles DBCS characters that are remote-submitted to a host and returned to the local level for display.

SAS/CONNECT software uses translate tables that are based on the DBCSTYPE option, which is set on each platform.

For example, if you are taking your SAS data set or flat file from the PC to the mainframe, the SAS System converts the data from the PC DBCS encoding to the mainframe DBCS encoding. It also wraps the DBCS string with the SO/SI control sequence required for mainframe systems.

As stated earlier, data can be truncated if the character string length is not defined long enough to contain the 2 bytes for SO/SI plus the data. The SAS System attempts to use blanks, if available, but data truncation can occur if the string does not have enough blanks. However, when downloading data from the mainframe to the PC, the SAS System automatically converts the data and strips off the SO/SI, which is not used on PC systems. There is, thus, no concern about possible data truncation.

You can also upload and download your SAS/FSP data entry systems or SAS/AF applications using SAS/CONNECTsoftware.

The following example automatically converts the PC SAS/AF catalog to IBM mainframe encoding:

```
options dbcs dbcstype=ibm dbcslang=japanese;

proc upload c=mypc.afcat
            out=mainfr.afcat;
run;
```

Applications developed on a Windows, OS/2, or UNIX system can be uploaded to a mainframe or minicomputer. SAS/CONNECT software converts the encoding system automatically.

## SAS/ACCESS Support for DBCS

SAS/ACCESS software provides support for DBCS data. With the SAS/ACCESS interfaces to DB2 and SQL/DS, DBCS is supported via the GRAPHIC, VARGRAPHIC, and LONG VARGRAPHIC data types. On most other databases, DBCS is supported as standard character data.

Where column names are in DBCS on some database systems, the name must be changed to a valid SAS name.

The following two examples use the SQL procedure:

```
proc sql;
   select * from customer
   where address like '東京%';
run;
```

The SQL procedure supports DBCS queries directly and through SQL pass-through. The following example uses SQL pass-through:

```
proc sql;
   select * from
      connection to DB2(select * from db2.customer
                                where name=G'鈴木');
      where address contains '品川区';
run;
```

The WHERE expression in this example uses DB2 processing capability on the variable NAME, which is a GRAPHIC variable type. The DB2 system stores DBCS as a GRAPHIC variable type, as stated earlier.

### Modifying DBCS Translate Tables

If you are using a DBCS encoding system that is not supported by the SAS System or uses different standard translate tables, you can use the DBCSTAB procedure to modify existing DBCS translate tables.

The following example creates a translate table for a new Korean DBCS system called CUSTTAB:

```
proc dbcstab
           /* name of the new translate table             */
       name=custtab
           /* based on HP encoding                        */
       basetype=hp15
           /* data to create the new table                */
       data=trantab
           /* Korean language                             */
       dbcslang=korean
           /* catalog descriptor                          */
       desc='Modified Korean Trantab'
          /* where the table is stored                    */
      catalog=sasuser.dbcs
          /* checks for invalid DBCS in the new data      */
      verify;
```

The TRANTAB data set is created as follows:

```
data trantab;
   hp15='6189'x; dec='b9b3'x; pcibm='a4d2'x;
run;
```

To specify the translate table, use the TRANTAB option:

```
options trantab=(,,,,,,,,,custtab);
```

The translate table is used for DBCS conversion with SAS/CONNECT software, PROC CPORT and PROC CIMPORT, and the DATA step function, KCVT.

The TRANTAB= option may be used to specify DBCS translate tables. The ninth argument specifies the DBCS system table:

```
options trantab=(,,,,,,,,systab);      /* ninth argument */
```

The tenth argument specifies the DBCS user table:

```
options trantab=(,,,,,,,,,usrtab);     /* tenth argument */
```

## Graphics Support

In graphics, the SAS System provides support at several different levels. The first level is that of hardware fonts. Where DBCS fonts are available at the hardware level and F=NONE is specified, the SAS System uses the hardware font, as shown in Figure 7:

**code**

```
title f=none h=1 '漢字フォント';
```

**output**

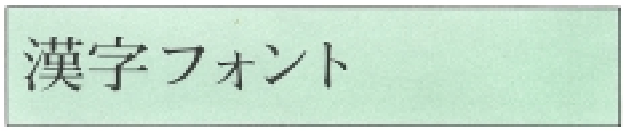


**Figure 7**

In addition, under some mainframe operating systems such as MVS or MSP (by Fujitsu, Inc.), the SAS System supports vendor-supplied software fonts in vendor products such as GDDM and GSP. In this case, the appropriate device driver must be specified to access these fonts. Specify chartype=2 in the GOPTIONS statement to access the graphics library's software fonts.

With the IBM product GDDM:

```
goptions dev=gddmpcg chartype=2;
title f=none 'ハードウェアフォントの使用';
```

With the Fujitsu product GSP:

```
goptions dev=gsp6683 chartype=2;
title f=none 'ハードウェアフォントの使用';
```

SAS Institute provides a number of DBCS software fonts in various Asian languages: for Japanese, KANJI and MINCHO; for Korean, HANGUL; and for traditional Chinese, KAI and MING. Users specify the appropriate font, as shown in Figure 8:

**code**

```
title f=kanji h=1 '漢字フォント';
title f=mincho h=1 '明朝体フォント';
```

**output**

漢字フォント
明朝体フォント

**Figure 8**

In general, these Institute-supplied software fonts are the best fonts for developing applications with newer SAS software products such as SAS/EIS software and the SAS/AF FRAME entries. These fonts are scalable so that they can shrink and grow with the size of the window. They are not device-dependent and, thus, are ideal for portable applications.

For example, to create a SAS/AF FRAME entry that includes DBCS graphics, simply create a graph with a Japanese, Chinese, or Korean font. Then, include it into the FRAME entry as a GRAPH type. Figure 9 shows a sample EIS built using traditional Chinese fonts:

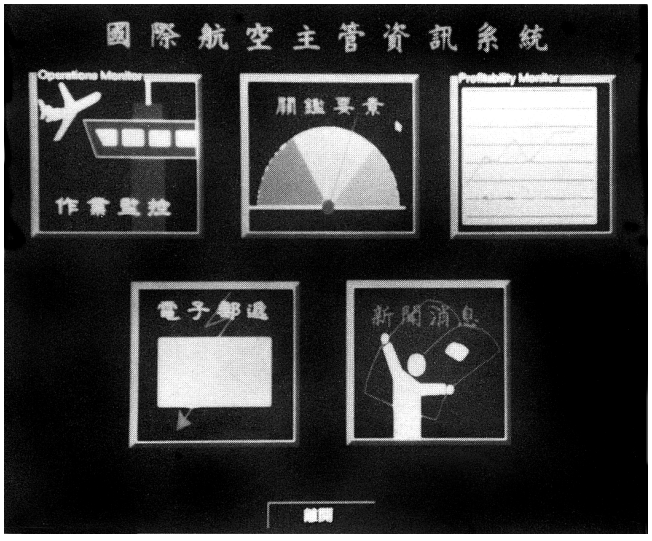


**Figure 9**

## DBCS Documentation

Most of the detailed documentation for DBCS support is in Japanese. These documents are available from SAS Institute's Asia/Pacific Operations. However, some portions of these documents have been translated into English; contact Asia/Pacific Operations for more information.

## Summary

Release 6.07 and later of the SAS System provides extensive support for Asian languages via DBCS support. With the correct hardware and system software in place, users may develop programs, data entry systems, and windowing applications using Japanese, Chinese, or Korean. Graphs may also be produced in these languages using that language's hardware or software fonts.

The philosophy and application portability of the SAS System's MVA is supported via cross-system conversion facilities. DBCS applications may be developed on one platform and ported to another platform using various conversion facilities available.

In conjunction with the SAS System's ability to access virtually any type of file or external database, the SAS System can be used as a utility to simply move DBCS data in external files from one application or platform to another, if such DBCS conversion is not otherwise available.