For Microsoft® Windows®

NetCOBOL

CBL Subroutines User's Guide

FUJITSU

Fourth Edition: July 2007

The contents of this manual may be revised without prior notice. No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of Fujitsu Limited.

© 1998-2007 Fujitsu Limited. All Rights Reserved.

Preface

About CBL Routines

The CBL routines explained in this manual are functionally compatible with the COBOL CBL subroutines of Micro Focus Co., Ltd.

Purpose

This manual explains the functions and specifications of NetCOBOL CBL routines.

Audience

This manual is intended for programmers who develop COBOL programs using NetCOBOL.

The CBL routines explained in this manual are functionally compatible with the COBOL CBL subroutines of Micro Focus Co., Ltd. Many CBL routines, however, may also be helpful to general users.

Prerequisite

Readers of this manual are assumed to be familiar with the following:

- Basic knowledge of COBOL syntax
- Basic knowledge of Windows

Organization

This manual classifies the CBL routines into the following categories which are covered by the corresponding chapters:

- Introduction
- Byte-stream File Routines
- Filename Routines
- File Routines
- Memory Allocation Routines
- Virtual Heap Routines
- Operating System Information Routines
- Run-unit Handling Routines
- Text Routines
- Logic Operator Routines
- Screen Routines
- Mouse Routines
- KeyBoard Routines

Related Manuals

Other NetCOBOL manuals are listed below.

Manual name	Description
NetCOBOL Language Reference	Detailed explanation of COBOL syntax
NetCOBOL User's Guide	How to use COBOL functions
NetCOBOL Debugging Guide	Explanation of debugger and debug functions

Trademarks

The trademarks appearing in this manual are as follows:

NetCOBOL is a trademark or registered trademark of Fujitsu Limited or its subsidiaries in the United States or other countries or in both.

Microsoft, Windows, Visual C++, and Visual Basic are the registered trademarks of Microsoft Corporation in the United States and other countries.

UNIX is a registered trademark in the United States and other countries, licensed exclusively through X/Open Company Limited.

Micro Focus and COBOL/2 are registered trademarks of Micro Focus.

Sun is a trademark of U.S. Sun Microsystems, Inc.

HP and HP-UX are the trademarks of the U.S. Hewlett-Packard Company.

Other company names and product names are the trademarks or registered trademarks of the respective companies.

Contents

Chapter 1. Introduction 9

Link Requirements 9

Notes on the “RETURNING” Parameter 9

Chapter 2. Byte-stream File Routines 11

CBL_CLOSE_FILE 11

CBL_CREATE_FILE 11

CBL_CREATE_FILE2 13

CBL_FLUSH_FILE 14

CBL_OPEN_FILE 15

CBL_OPEN_FILE2 16

CBL_READ_FILE 17

CBL_WRITE_FILE 18

CBL_CLOSE_64BIT_FILE 19

CBL_CREATE_64BIT_FILE 20

CBL_FLUSH_64BIT_FILE 21

CBL_OPEN_64BIT_FILE 22

CBL_READ_64BIT_FILE 23

CBL_WRITE_64BIT_FILE 24

Chapter 3. Filename Routines 27

CBL_JOIN_FILNAME 27

CBL_SPLIT_FILNAME 29

Chapter 4. File Routines 33

CBL_CHANGE_DIR 33

CBL_CHANGE_DIR2 33

CBL_CHECK_FILE_EXIST 34

CBL_CHECK_FILE_EXIST2 35

CBL_COPY_FILE 36

CBL_COPY_FILE2 37

CBL_CREATE_DIR 38

CBL_CREATE_DIR2 38

CBL_DELETE_DIR 39

CBL_DELETE_DIR2 40
CBL_DELETE_FILE 40
CBL_DELETE_FILE2 41
CBL_LOCATE_ FILE 42
CBL_LOCATE_FILE2 44
CBL_READ_DIR 46
CBL_RENAME_FILE 46
CBL_RENAME_FILE2 47
PC_FIND_DRIVES 48
PC_READ_DRIVE 48
PC_SET_DRIVE 49

Chapter 5. Memory Allocation Routines 51
CBL_ALLOC_MEM 51
CBL_FREE_MEM2 52

Chapter 6. Virtual Heap Routines 53
CBL_OPEN_VFILE 53
CBL_CLOSE_VFILE 54
CBL_WRITE_VFILE 55
CBL_READ_VFILE 56

Chapter 7. Operating System Information Routines 57
CBL_GET_OS_INFO 57

Chapter 8. Run-unit Handling Routines 59
CBL_YIELD_RUN_UNIT 59

Chapter 9. Text Routines 61
CBL_TOUPPER 61
CBL_TOLOWER 62

Chapter 10. Logical Operator Routines 63
CBL_AND 63
CBL_EQ 64
CBL_IMP 65
CBL_NOT 66
CBL_OR 67

CBL_XOR 68

Chapter 11. Screen Routines 69

CBL_GET_CSR_POS 69
CBL_SET_CSR_POS 70
CBL_SET_CSR_SHAPE 71
CBL_CREATE_SCR 72
CBL_WRITE_SCR_TTY_CHAR 72
CBL_WRITE_SCR_TTY 73
CBL_CLEAR_SCR 74
CBL_GET_SCR_SIZE 75
CBL_GET_SCR_GRAPHICS 75
CBL_GET_SCR_LINE_DRAW 77
CBL_ALARM_SOUND 78
CBL_BELL_SOUND 79
CBL_GET_VGA_MODE 79
CBL_WRITE_SCR_ATTRS 80
CBL_WRITE_SCR_CHARS 81
CBL_WRITE_SCR_CHARS_ATTR 82
CBL_WRITE_SCR_CHATTRS 83
CBL_WRITE_SCR_N_ATTR 84
CBL_WRITE_SCR_N_CHAR 85
CBL_WRITE_SCR_N_CHATTR 86
CBL_READ_SCR_ATTRS 87
CBL_READ_SCR_CHARS 88
CBL_READ_SCR_CHATTRS 89
CBL_SWAP_SCR_CHATTRS 90
CBL_SET_SCR_TERMKEY 91
CBL_SET_SCR_KEYFILE 92
CBL_READ_SCR_KEY 93
CBL_INIT_SCR_ACCEPT_ATTR 94

Chapter 12. Mouse Routines 97

CBL_GET_MOUSE_MASK 97
CBL_GET_MOUSE_POSITION 98
CBL_GET_MOUSE_STATUS 99

CBL_HIDE_MOUSE 99
CBL_INIT_MOUSE 100
CBL_READ_MOUSE_EVENT 101
CBL_SET_MOUSE_MASK 102
CBL_SET_MOUSE_POSITION 102
CBL_SHOW_MOUSE 103
CBL_TERM_MOUSE 104
WIN_GET_MOUSE_SHAPE 104
WIN_SET_MOUSE_SHAPE 105

Chapter 13. Keyboard Routines 107
CBL_GET_KBD_STATUS 107
CBL_READ_KBD_CHAR 108

Appendix A. List of CBL Routines 109

Chapter 1. Introduction

Link Requirements

When calling CBL_routines with dynamic-link structure, link (using WINLINK) F3BICBLR.LIB when linking the calling program.

When calling CBL_routines with dynamic-program structure, the following entry information is required.

```
:
[ENTRY]
CBL_routine_name=F3BICBLR.DLL
```

Refer to "Entry Information" in Chapter 5 in the "NetCOBOL User's Guide" for details on specifying entry information.

Notes on the “RETURNING” Parameter

The status-code is described in this manual as being acquired by use of the RETURNING clause. In fact, RETURNING can be omitted. When RETURNING is omitted, status-code can be acquired from the PROGRAM-STATUS special register.

CBL routines under UNIX do not support the RETURNING clause.

Chapter 2. Byte-stream File Routines

CBL_CLOSE_FILE

This routine closes a file.

Specification

Parameter data definition

```
01  file-handle:  PIC X(4).
01  status-code:  PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_CLOSE_FILE"
    USING file-handle
    RETURNING status-code.
```

Interface

file-handle

Specifies the file handle returned by CBL_OPEN_FILE, CBL_OPEN_FILE2, CBL-CREATE_FILE, or CBL_CREATE_FILE2.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code

0: Successful
-1: Parameter error
Other than above: Value indicating ANSI74 file status

CBL_CREATE_FILE

This routine generates a new file.

Specification

Parameter data definition

```
01  file-name:       PIC X(n).
01  access-mode:     PIC 9(4) BINARY.
01  exclusion-mode:  PIC 9(4) BINARY.
01  device:          PIC 9(4) BINARY.
01  file-handle:     PIC X(4).
01  status-code:     PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_CREATE_FILE"
    USING file-name
          access-mode
          exclusion-mode
          device
          file-handle
    RETURNING status-code.
```

Interface

file-name
Specifies the name of the file to be generated. The end of the file name is indicated by a blank or null character. The maximum size of the file name is 255.

access-mode
Specifies one of the following access modes:

1: Read-only mode
2: Write-only mode
3: Read-write mode

exclusion-mode
Specifies one of the following exclusion modes for other processes attempting to open the file:

0: No reading or writing
1: No writing
2: No reading
3: No restrictions

device
Indicates an area reserved for future expansion. 0 must be set in this area.

file-handle
Specifies the area to store the file handle.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
-1: Parameter error
Other than above: Value indicating ANSI74 file status

NOTES
When the file exists, it is overwritten.

CBL_CREATE_FILE2

CBL_CREATE_FILE2 generates a new file.

The difference between the CBL_CREATE_FILE subroutine and this subroutine is that this subroutine can support a file name containing a space.

Specification

Parameter data definition

```
01 file-name PIC X(n).
01 access-mode PIC 9(4) BINARY.
01 exclusion-mode PIC 9(4) BINARY.
01 device PIC 9(4) BINARY.
01 file-handle PIC X(4).
01 status-code PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_CREATE_FILE2"
  USING file-name
     access-mode
     exclusion-mode
     device
     file-handle
  RETURNING status-code.
```

Interface

file-name

Specifies the name of the file to be generated. It must be terminated by a null character. The maximum size of the file name is 255.

access-mode

Specifies one of the following access modes:

1: Read-only mode
2: Write-only mode
3: Read-write mode

exclusion-mode

Specifies one of the following exclusion modes:

0: No reading or writing
1: No writing
2: No reading
3: No restrictions

device

Indicates an area reserved for future expansion. 0 must be set in this area.

file-handle

Specifies the area to store the file handle.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code

0: Successful
-1: Parameter error

Other values indicate an ANSI74 file status.

NOTES
If the file specified in CBL_CREATE_FILE2 already exists, it is overwritten.

CBL_FLUSH_FILE

This routine outputs buffer data to a file.

Specification

Parameter data definition

```
01  file-handle:  PIC X(4).
01  status-code:  PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_FLUSH_FILE"
    USING file-handle
    RETURNING status-code.
```

Interface

file-handle
Specifies the file handle returned by CBL_OPEN_FILE or CBL-CREATE_FILE.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
-1: Buffer flushing failure or parameter error
Other than above: Value indicating ANSI74 file status

CBL_OPEN_FILE

This routine opens a file.

Specification

Parameter data definition

```
01  file-name:       PIC X(n).
01  access-mode:     PIC 9(4) BINARY.
01  exclusion-mode:  PIC 9(4) BINARY.
01  device:          PIC 9(4) BINARY.
01  file-handle:     PIC X(4).
01  status-code:     PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_OPEN_FILE"
    USING file-name
          access-mode
          exclusion-mode
          device
          file-handle
    RETURNING status-code.
```

Interface

file-name
Specifies the name of the file to be generated. It must be terminated by a blank or null character. The maximum size of the file name is 255.

access-mode
Specifies one of the following access modes:

1: Read-only mode
2: Write-only mode
3: Read-write mode

exclusion-mode
Specifies one of the following exclusion modes:

0: No reading or writing
1: No writing
2: No reading
3: No restrictions

device
Indicates an area reserved for future expansion. 0 must be set in this area.

file-handle
Specifies the area to store the file handle.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
 0: Successful
-1: Parameter error
Other than above: Value indicating ANSI74 file status

CBL_OPEN_FILE2

CBL_OPEN_FILE2 routine opens a file. The difference between the CBL_OPEN_FILE subroutine and this subroutine is that this subroutine can support a file name containing a space.

Specification

Parameter data definition

```
01 file-name PIC X(n).
01 access-mode PIC 9(4) BINARY.
01 exclusion-mode PIC 9(4) BINARY.
01 device PIC 9(4) BINARY.
01 file-handle PIC X(4).
01 status-code PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_OPEN_FILE2"
USING file-name
access-mode
exclusion-mode
device
file-handle
RETURNING status-code.
```

Interface

file-name

Specifies the name of the file to be generated. It must be terminated by a null character. The maximum size of the file name is 255.

access-mode

Specifies one of the following access modes:

1: Read-only mode
2: Write-only mode
3: Read-write mode

exclusion-mode

Specifies one of the following exclusion modes:

0: No reading or writing
1: No writing
2: No reading
3: No restrictions

device

Indicates an area reserved for future expansion. 0 must be set in this area.

file-handle

Specifies the area to store the file handle.

Return code

The return code is set in the status-code field specified in the

RETURNING clause.

status-code

0: Successful

-1: Parameter error

Other values indicate an ANSI74 file status.

CBL_READ_FILE

This routine reads data from a file or gets the size of a file.

Specification

Parameter data definition

```
01  file-handle:               PIC X(4).
01  relative-address-in-file:  PIC 9(18) BINARY.
01  byte-count:                PIC 9(9) BINARY.
01  flag:                      PIC 9(4) BINARY.
01  buffer:                    PIC X(n).
01  status-code:               PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_READ_FILE"
    USING file-handle
          relative-address-in-file
          byte-count
          flag
          buffer
    RETURNING status-code.
```

Interface

file-handle
Specifies the file handle returned by CBL_OPEN_FILE.

relative-address-in-file
Specifies the relative address of data in the file.

byte-count
Specifies the length of data to be read. The maximum value is 0xFFFF.

flag
Specifies flag information as follows:

0: Standard reading
128: Setting of file size at relative address in file

buffer
Specifies the area to store read data.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
-1: Parameter error
Other than above: Value indicating ANSI74 file status

CBL_WRITE_FILE

This routine writes data in a file.

Specification

Parameter data definition

```
01  file-handle:               PIC X(4).
01  relative-address-in-file:  PIC 9(18) BINARY.
01  byte-count:                PIC 9(9) BINARY.
01  flag:                      PIC 9(4) BINARY.
01  buffer:                    PIC X(n).
01  status-code:               PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_WRITE_FILE"
    USING file-handle
          relative-address-in-file
          byte-count
          flag
          buffer
    RETURNING status-code.
```

Interface

file-handle
Specifies the file handle returned by CBL_OPEN_FILE or CBL_CREATE_FILE.

relative-address-in-file
Specifies the relative address of data in the file.

byte-count
Specifies the length of data to be written. The maximum value is 0xFFFF.

flag
Specifies flag information as follows:

0: Standard writing

buffer
Specifies the area storing the data to be written.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
-1: Parameter error

Other than above: Value indicating ANSI74 file status

CBL_CLOSE_64BIT_FILE

The CBL_CLOSE_64BIT_FILE routine closes a file.

This subroutine supports the Solaris large file system.

Specification

Parameter data definition

```
01 file-handle PIC X(4).
01 status-code PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_CLOSE_64BIT_FILE"
  USING file-handle
  RETURNING status-code.
```

Interface

file-handle

Specifies the file handle returned by CBL_OPEN_64BIT_FILE or CBL_CREATE_64BIT_FILE.

Return code

The return code is set in the status-code field specified in the RETURNING clause.

status-code

0: Successful
-1: Parameter error

Other values indicate an ANSI74 file status.

CBL_CREATE_64BIT_FILE

CBL_CREATE_64BIT_FILE generates a new file. This subroutine supports the Solaris large file system.

Specification

Parameter data definition

```
01 file-name PIC X(n).
01 access-mode PIC 9(4) BINARY.
01 exclusion-mode PIC 9(4) BINARY.
01 device PIC 9(4) BINARY.
01 file-handle PIC X(4).
01 status-code PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_CREATE_64BIT_FILE"
  USING file-name
     access-mode
     exclusion-mode
     device
     file-handle
  RETURNING status-code.
```

Interface

file-name

Specifies the name of the file to be generated. File name must terminate in a blank or null character.

access-mode

Specifies one of the following access modes:

1: Read-only mode
2: Write-only mode
3: Read-write mode

exclusion-mode

Specifies one of the following exclusion modes:

0: No reading or writing
1: No writing
2: No reading
3: No restrictions

device

Indicates an area reserved for future expansion. 0 must be set in this area.

file-handle

Specifies the area to store the file handle.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code

0: Successful
-1: Parameter error

Other values indicate an ANSI74 file status.

NOTES
If the file specified in CBL_CREATE_64BIT_FILE already exists, it is overwritten.

CBL_FLUSH_64BIT_FILE

CBL_FLUSH_64BIT_FILE writes buffer data to a file. This subroutine supports the Solaris large file system.

Specification

Parameter data definition

```
01 file-handle PIC X(4).
01 status-code PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_FLUSH_64BIT_FILE"
USING file-handle
RETURNING status-code.
```

Interface

file-handle
Specifies the file handle returned by CBL_OPEN_64BIT_FILE or CBL_CREATE_64BIT_FILE.

Return code

The return code is set in the status-code field of the RETURNING clause.

status-code

0: Successful
-1: Buffer flushing failure or parameter error

Other values indicate an ANSI74 file status.

CBL_OPEN_64BIT_FILE

CBL_OPEN_64BIT_FILE routine opens a file. This subroutine supports the Solaris large file system.

Specification

Parameter data definition

```
01 file-name PIC X(n).
01 access-mode PIC 9(4) BINARY.
01 exclusion-mode PIC 9(4) BINARY.
01 device PIC 9(4) BINARY.
01 file-handle PIC X(4).
01 status-code PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_OPEN_64BIT_FILE"
  USING file-name
     access-mode
     exclusion-mode
     device
     file-handle
  RETURNING status-code.
```

Interface

file-name
Specifies the name of the file to be generated. The end of the file name is indicated a blank or null character.

access-mode
Specifies one of the following access modes:

1: Read-only mode
2: Write-only mode
3: Read-write mode

exclusion-mode
Specifies one of the following exclusion modes:

0: No reading or writing
1: No writing
2: No reading
3: No restrictions

device
Indicates an area reserved for future expansion. 0 must be set in this area.

file-handle
Specifies the area to store the file handle.

Return code

The return code is set in the status-code field specified in the RETURNING clause.

status-code

0: Successful
-1: Parameter error

Other values indicate an ANSI74 file status.

CBL_READ_64BIT_FILE

CBL_READ_64BIT_FILE reads data from a file, or gets the size of a file. This subroutine supports the Solaris large file system.

Specification

Parameter Data Definition

```
01 file-handle PIC X(4).
01 relative-address-in-file PIC 9(18) BINARY.
01 byte-count PIC 9(9) BINARY.
01 flag PIC 9(4) BINARY.
01 buffer PIC X(n).
01 status-code PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_READ_64BIT_FILE"
  USING file-handle
     relative-address-in-file
     byte-count
     flag
     buffer
  RETURNING status-code.
```

Interface

file-handle
Specifies the file handle returned by CBL_OPEN_64BIT_FILE.

relative-address-in-file
Specifies the relative address of data in the file.

byte-count
Specifies the length of data to be read.

Flag
Specifies flag information as follows.

0: Standard reading
128: Setting of file size at relative-address-in-file

buffer
Specifies the area to store read data.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code

0: Successful
-1: Parameter error

Other values indicate an ANSI74 file status.

CBL_WRITE_64BIT_FILE

CBL_WRITE_64BIT_FILE writes data to a file. This subroutine supports the Solaris large file system.

Specification

Parameter data definition

```
01 file-handle PIC X(4).
01 relative-address-in-file PIC 9(18) BINARY.
01 byte-count PIC 9(9) BINARY.
01 flag PIC 9(4) BINARY.
01 buffer PIC X(n).
01 status-code PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_WRITE_64BIT_FILE"
  USING file-handle
     relative-address-in-file
     byte-count
     flag
     buffer
  RETURNING status-code.
```

Interface

file-handle

Specifies the file handle returned by CBL_OPEN_64BIT_FILE or CBL_CREATE_64BIT_FILE.

relative-address-in-file
Specifies the relative address of data in the file.

byte-count
Specifies the length of data to be written.

Flag
Specifies flag information as follows.

0: Standard writing

buffer
Specifies the area storing the data to be written.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code

0: Successful
-1: Parameter error

Other values indicate an ANSI74 file status.

Chapter 3. Filename Routines

CBL_JOIN_FILNAME

This routine joins the file name components (i.e., device name, basic name, extension) to each other to create a file name.

Specification

Parameter data definition

```
01  split/join-parameters.
    02  parameter-length:                    PIC 9(4) BINARY.
    02  split/join-flag-1:                   PIC 9(4) BINARY.
    02  split/join-flag-2:                   PIC 9(4) BINARY.
    02  relative-address-of-device-name:     PIC 9(4) BINARY.
    02  device-name-length:                  PIC 9(4) BINARY.
    02  relative-address-of-basic-name:      PIC 9(4) BINARY.
    02  basic-name-length:                   PIC 9(4) BINARY.
    02  relative-address-of-extension:       PIC 9(4) BINARY.
    02  extension-length:                    PIC 9(4) BINARY.
    02  total-length:                        PIC 9(4) BINARY.
    02  split-name-buffer-size:              PIC 9(4) BINARY.
    02  joined-name-buffer-size:             PIC 9(4) BINARY.
    02  length-of-first-path-name-component: PIC 9(4) BINARY.
01  joined-name-buffer:  PIC X(n).
01  device-name-buffer:  PIC X(n).
01  basic-name-buffer:   PIC X(n).
01  extension-buffer:    PIC X(n).
01  status-code:         PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_JOIN_FILENAME"
    USING split/join-parameters
          joined-name-buffer
          device-name-buffer
          basic-name-buffer
          extension-buffer
    RETURNING status-code.
```

Interface

split/join-parameters

parameter-length
Specifies the length of split/join parameters. The standard value is 24.

split/join-flag-1
Specifies the information of split/join flag 1 in units of bits as follows:

Bit 0

ON: The file name using uppercase characters is returned.
OFF: The file name using specified characters is returned.

Bit 1

ON: The file name string ends with a null character.
OFF: The file name string ends with a blank character.

split/join-flag-2
Specifies the information of split/join-flag-2 in units of bits.

relative-address-of-device-name
Specifies the relative address of device name from the head of the device-name-buffer.

device-name-length
Specifies the length of device name, not including any ending blank or null character. The maximum value is 255.

relative-address-of-basic-name
Specifies the relative address of basic name from the head of the basic-name-buffer.

basic-name-length
Specifies the length of basic name, not including any ending blank or null character. The maximum value is 255.

relative-address-of-extension
Specifies the relative address of extension from the head of the extension-buffer.

extension-length
Specifies the length of the extension, not including any ending blank or null character. The maximum value is 255.

total-length
Specifies the area to store the total number of file name characters.

split-name-buffer-size
Specifies the size of the split-name-buffer.

joined-name-buffer-size
Specifies the size of the joined-name-buffer.

length-of-first-path-name-component
Specifies the length of the first path name component.

joined-name-buffer
Specifies the area to store the joined file name.

device-name-buffer
Specifies the device name.

The device names that need not ":" are as follows.

CON, AUX, COM1, PUN, COM2, LPT1, LPT, LST, PRN, LPT2, LPT3, ERR, NULL

basic-name-buffer
Specifies the basic name.

extension-buffer
Specifies the extension.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
1: Overflow of joined-name-buffer
4: File name invalid

Example

On the following case, this subroutine builds ”A:\Example\Master.Doc” into join-name-buffer.

device-name-buffer is ”A:\Example”.

Basic-name-buffer is ”Master”.

Extension-buffer is ”Doc”.

CBL_SPLIT_FILNAME

This routine splits a file name into components (i.e., device name, basic name, extension).

Specification

Parameter data definition

```
01  split/join-parameters.
    02  parameter-length:                    PIC 9(4) BINARY.
    02  split/join-flag-1:                   PIC 9(4) BINARY.
    02  split/join-flag-2:                   PIC 9(4) BINARY.
    02  relative-address-of-device-name:     PIC 9(4) BINARY.
    02  device-name-length:                  PIC 9(4) BINARY.
    02  relative-address-of-basic-name:      PIC 9(4) BINARY.
    02  basic-name-length:                   PIC 9(4) BINARY.
    02  relative-address-of-extension:       PIC 9(4) BINARY.
    02  extension-length:                    PIC 9(4) BINARY.
    02  total-length:                        PIC 9(4) BINARY.
    02  split-name-buffer-size:              PIC 9(4) BINARY.
    02  joined-name-buffer-size:             PIC 9(4) BINARY.
    02  length-of-first-path-name-component: PIC 9(4) BINARY.
01  split-name-buffer:  PIC X(n).
01  status-code:        PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_SPLIT_FILENAME"
    USING split/join-parameters
          split-name-buffer
    RETURNING status-code.
```

Interface

split/join-parameters

parameter-length
Specifies the length of split/join parameters. The standard value is 24.

split/join-flag-1

Specifies the information of split/join-flag-1 in units of bits as follows:

Bit 0

ON: The file name using uppercase characters is returned.
OFF: The file name using specified characters is returned.

Bit 1

ON: The file name string ends with a null character.
OFF: The file name string ends with a blank character.

split/join-flag-2
Specifies the area to store the information of split/join-flag-2. The information to be stored is as follows:

Bit 1

ON: A wild card is found in device name.
OFF: No wild card is found in device name.

Bit 2

ON: A wild card is found in basic name or extension.
OFF: No wild card is found in basic name and extension.

relative-address-of-device-name
Specifies the area to store the relative address of device name from the head of the split-name-buffer.

device-name-length
Specifies the area to store the length of device name.

relative-address-of-basic-name
Specifies the area to store the relative address of basic name from the head of the split-name-buffer.

basic-name-length
Specifies the area to store the length of basic name.

relative-address-of-extension
Specifies the area to store the relative address of extension from the head of the split-name-buffer.

extension-length
Specifies the area to store the length of extension.

total-length
Specifies the area to store the total number of file name characters.

split-name-buffer-size
Specifies the size of split-name-buffer. The maximum value is 255.

joined-name-buffer-size
Specifies the size of joined-name-buffer.

length-of-first-path-name-component
Specifies the area to store the number of characters from the head of the file name until the first "\", "/", or ":".

split-name-buffer
Specifies the split-name-buffer.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
4: File name invalid

Example

If ”A:\Example\Master.Doc” is stored in split-name-buffer, this subroutine is processed as follows.

relative-address-of device-name will be set 1.

Device-name-length will be set 10.

Relative-address-of-basic-name will be set 12.

Basic-name-length will be set 6.

Relative-address-of-extension will be set 19.

Extention-lemgth will be set 3.

Chapter 4. File Routines

CBL_CHANGE_DIR

This routine changes the current directory to another directory.

Specification

Parameter data definition

```
01  path-name:    PIC X(n).
01  status-code:  PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_CHANGE_DIR"
    USING path-name
    RETURNING status-code.
```

Interface

path-name
Specifies a full or partial path name, ending with a blank or null character. The maximum size of the path name is 255.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_CHANGE_DIR2

CBL_CHANGE_DIR2 changes the current directory to another directory.

The difference between the CBL_CHANGE_DIR subroutine and this subroutine is that this subroutine can support a directory name containing a space.

Specification

Parameter data definition

```
01 path-name PIC X(n).
01 status-code PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_CHANGE_DIR2"
USING path-name
RETURNING status-code.
```

Interface

path-name

Specifies a full or partial path name, ending with a null character. The maximum size of the path name is 255.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code

0: Successful

Other than 0: Failure

CBL_CHECK_FILE_EXIST

This routine checks whether a file exists or not.

Specification

Parameter data definition

```
01  file-name:             PIC X(n).
01  file-details.
    02  file-size:         PIC 9(18) BINARY.
    02  file-date.
        03  day:           PIC 9(4) BINARY.
        03  month:         PIC 9(4) BINARY.
        03  year:          PIC 9(4) BINARY.
    02  file-time.
        03  hour:          PIC 9(4) BINARY.
        03  minute:        PIC 9(4) BINARY.
        03  second:        PIC 9(4) BINARY.
        03  1/100-second:  PIC 9(4) BINARY.
01  status-code:           PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_CHECK_FILE_EXIST"
    USING file-name
          file-details
    RETURNING status-code.
```

Interface

file-name
Specifies the name of the file to be checked. If no path name is specified, the

current directory is assumed. It must be terminated by a blank or null character. The maximum size of the file name is 255.

file-size
Specifies the area to store the size of the file.

file-date
Specifies the area to store the creation date of the file.

file-time
Specifies the area to store the creation time of the file.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_CHECK_FILE_EXIST2

CBL_CHECK_FILE_EXIST2 checks whether a file exists or not. The difference between the CBL_CHECK_FILE_EXIST subroutine and this subroutine is that this subroutine can support a file name containing a space.

Specification

Parameter data definition

```
01 file-name PIC X(n).
01 file-details.
02 file-size PIC 9(18) BINARY.
02 file-date.
03 day PIC 9(4) BINARY.
03 month PIC 9(4) BINARY.
03 year PIC 9(4) BINARY.
02 file-time.
03 hour PIC 9(4) BINARY.
03 minute PIC 9(4) BINARY.
03 second PIC 9(4) BINARY.
03 1-100-second PIC 9(4) BINARY.
01 status-code PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_CHECK_FILE_EXIST2"
USING file-name
file-details
RETURNING status-code.
```

Interface

file-name

Specifies the name of the file to be checked. If no path name is specified, the current directory is assumed. The end of the file name is indicated by a null character. The maximum size of the file name is 255.

file-size

Specifies the area to store the size of the file.

file-date

Specifies the area to store the creation date of the file.

file-time

Specifies the area to store the creation time of the file.

Return code

The return code is set in the status-code specified in the RETURNING clause.

RETURNING clause.

status-code

0: Successful

Other than 0: Failure

CBL_COPY_FILE

This routine copies a file.

Specification

Parameter data definition

```
01  file-name-1:  PIC X(n).
01  file-name-2:  PIC X(n).
01  status-code:  PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_COPY_FILE"
    USING file-name-1
          file-name-2
    RETURNING status-code.
```

Interface

file-name-1
Specifies the name of the file to be copied. If no path name is specified, the current directory is assumed. The end of the file the name is indicated by a blank or null character. The maximum size of the file name is 255.

file-name-2
Specifies the name of the new file. If no path name is specified, the current directory is assumed. It must be terminated by a blank or null character. The maximum size of the file name is 255.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_COPY_FILE2

CBL_COPY_FILE2 copies a file. The difference between the CBL_COPY_FILE subroutine and this subroutine is that this subroutine can support a file name containing a space.

Specification

Parameter data definition

```
01 file-name-1 PIC X(n).
01 file-name-2 PIC X(n).
01 status-code PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_COPY_FILE2"
USING file-name-1
file-name-2
RETURNING status-code.
```

Interface

file-name-1

Specifies the name of the file to be copied. It must be terminated by a null character.If no path name is specified, the current directory is assumed. The maximum size of the file name is 255.

file-name-2

Specifies the name of the new file. If no path name is specified, the current directory is assumed. The end of the file name is indicated by a null character. The maximum size of the file name is 255.

Return code

The return code is set in the status-code specified in the RETURNING clause.

CBL_CREATE_DIR

This routine creates a directory.

Specification

Parameter data definition

```
01  path-name:    PIC X(n).
01  status-code:  PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_CREATE_DIR"
    USING path-name
    RETURNING status-code.
```

Interface

path-name
Specifies a full or partial path name, ending with a blank or null character. The maximum size of the path name is 248.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_CREATE_DIR2

CBL_CREATE_DIR2 creates a directory. The difference between the CBL_CREATE_DIR subroutine and this subroutine is that this subroutine can support a directory name containing a space.

Specification

Parameter data definition

```
01 path-name PIC X(n).
01 status-code PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_CREATE_DIR2"
USING path-name
RETURNING status-code.
```

Interface

path-name

Specifies a full or partial path name, ending with a null character. The maximum size of the path name is 248.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code

0: Successful

Other than 0: Failure

CBL_DELETE_DIR

This routine deletes a directory.

Specification

Parameter data definition

```
01  path-name:    PIC X(n).
01  status-code:  PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_DELETE_DIR"
    USING path-name
    RETURNING status-code.
```

Interface

path-name
Specifies a full or partial path name, ending with a blank or null character. The maximum size of the path name is 255.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_DELETE_DIR2

CBL_DELETE_DIR2 deletes a directory. The difference between the CBL_DELETE_DIR subroutine and this subroutine is that this subroutine can support a directory name containing a space.

Specification

Parameter data definition

```
01 path-name PIC X(n).
01 status-code PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_DELETE_DIR2"
USING path-name
RETURNING status-code.
```

Interface

path-name

Specifies a full or partial path name, ending with a null character. The maximum size of the path name is 255.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code

0: Successful
Other than 0: Failure

CBL_DELETE_FILE

This routine deletes a file.

Specification

Parameter data definition

```
01  file-name:    PIC X(n).
01  status-code:  PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_DELETE_FILE"
    USING file-name
    RETURNING status-code.
```

Interface

file-name
Specifies a file name, ending with a blank or null character. If no path name is specified, the current directory is assumed. The maximum size of the file name is 255.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_DELETE_FILE2

CBL_DELETE_FILE2 deletes a file. The difference between the CBL_DELETE_FILE subroutine and this subroutine is that this subroutine can support a file name containing a space.

Specification

Parameter data definition

```
01 file-name PIC X(n).
01 status-code PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_DELETE_FILE2"
USING file-name
RETURNING status-code.
```

Interface

file-name

Specifies a file name, ending with a null character. If no path name is specified, the current directory is assumed. The maximum size of the file name is 255.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code

0: Successful

Other than 0: Failure

CBL_LOCATE_ FILE

When the file specification includes an environment variable, this routine returns the name of the file into which the environment variable is expanded. This routine also determines whether a file on another disk is to be used.

Specification

Parameter data definition

```
01  user-file-specification:  PIC X(n).
01  user-mode:                PIC 9(4) BINARY.
01  real-file-specification.
    02  buffer-size:          PIC 9(4) BINARY.
    02  buffer:               PIC X(n).
01  existence-flag:           PIC 9(4) BINARY.
01  path-flag:                PIC 9(4) BINARY.
01  status-code:              PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_LOCATE_FILE"
    USING user-file-specification
          user-mode
          real-file-specification
          existence-flag
          path-flag
    RETURNING status-code.
```

Interface

user-file-specification
Specifies a user file, ending with a blank or null character. Specification contents are as follows:

When specifying a standard file name:

device-name\file-name.extension

When specifying an embedded environment variable:

$environment\file-name.extension

The maximum size of the file name is 255.

user-mode
Specifies one of the following modes to handle the user-file specification:

0: The file is checked to determine whether it exists on another disk. When the user-file specification includes an environment variable, files are searched sequentially along the path specified by the environment variable. If the file is found, the file path name is expanded and set in real-file-specification (to be output by this routine). If the file is not found, the result of expansion using the first path name specified by the environment variable is set in the real-file-specification.

1: When the user-file specification includes an environment variable, the file is not searched, but the result of expansion using the first path name is set in the real-file-specification.

2: When the user-file specification does not include an environment variable, the file is not searched, but the result of expansion using the next path name is set in the real-file-specification.

real-file-specification
Specifies the size of the next buffer in the buffer-size parameter, and the area to store the expanded file path name in the buffer parameter.

existence-flag
Specifies the area to store the existence flag. The information to be set is as follows:

When user-mode is 0:

0: The file is not found.
3: The file is found on another disk.

When user-mode is other than 0:

The routine returns 0.

path-flag
Specifies the area to store the path flag. The information to be set is as follows:

0: The real-file-specification does not include an expanded environment variable.
1: The real-file-specification includes an expanded environment variable.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
1: Environment variable not found
2: Next path not found
3: Expanded path name longer than buffer size
255: Other error

NOTES
This subroutine supports the environment variable, which has 1 to 255 bytes of right parts.

CBL_LOCATE_FILE2

When the file specification includes an environment variable, CBL_LOCATE_ FILE2 returns the name of the file into which the environment variable is expanded. CBL_LOCATE_ FILE2 also determines whether a file on another disk is to be used. The difference between the CBL_LOCATE_FILE subroutine and this subroutine is that this subroutine can support a file name containing a space.

Specification

Parameter data definition

```
01 user-file-specification PIC X(n).
01 user-mode PIC 9(4) BINARY.
01 real-file-specification.
02 buffer-size PIC 9(4) BINARY.
02 buffer PIC X(n).
01 existence-flag PIC 9(4) BINARY.
01 path-flag PIC 9(4) BINARY.
01 status-code PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_LOCATE_FILE2"
USING user-file-specification
user-mode
real-file-specification
existence-flag
path-flag
RETURNING status-code.
```

Interface

user-file-specification

Specifies a user file, ending with a null character. Specification contents are as follows. When specifying a standard file name:

device-name\file-name.extension

When specifying an embedded environment variable:

$environment\file-name.extension

The maximum size of the file name is 255.

user-mode

Specifies one of the following modes to handle the user-file specification.

0: The file is checked to determine whether it exists on another disk. When the user-file specification includes an environment variable, files are searched sequentially along the path specified by the environment variable.

If the file is found, the file path name is expanded and set in real-file-specification (to be output by this routine). If the file is not found, the result of the expansion using the first path name specified by the environment variable is set in the real-file-specification.

1: When the user-file specification includes an environment variable, the file is not searched, but the expansion result using the first path name is set in the real-file-specification.

2: When the user-file specification includes an environment variable, the file is not searched, but the expansion result using the next path name is set in the real-file-specification.

real-file-specification

Specifies the size of the next buffer in the buffer-size parameter, and the area to store the expanded file path name in the buffer parameter.

existence-flag

Specifies the area to store the existence flag. The information to be set is as follows.

When user-mode is 0:

0: The file is not found.

3: The file is found on another disk.

When user-mode is other than 0:

The routine returns 0.

path-flag

Specifies the area to store the path flag. The information to be set is as follows.

0: The real-file-specification does not include an expanded environment variable.

1: The real-file-specification includes an expanded environment variable.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code

0: Successful

1: Environment variable not found

2: Next path not found

3: Expanded path name longer than buffer size

255: Other error

NOTES

This subroutine supports an environment variable, which has between 1 and 255 bytes. When multiple directory paths are specified in a environment variable, specify the environment variable as follows:

Set $environment-variable = dir-path1;dir-path2;...

CBL_READ_DIR

This routine returns the full path name of the current directory.

Specification

Parameter data definition

```
01  path-name:         PIC X(n).
01  path-name-length:  PIC 9(4) BINARY.
01  status-code:       PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_READ_DIR"
    USING path-name
          path-name-length
    RETURNING status-code.
```

Interface

path-name
Specifies the area to store the path name.

path-name-length
Specifies the size of the area to store the path name. The maximum value is 255.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_RENAME_FILE

This routine changes the name of a file.

Specification

Parameter data definition

```
01  old-file-name:  PIC X(n).
01  new-file-name:  PIC X(n).
01  status-code:    PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_RENAME_FILE"
    USING old-file-name
          new-file-name
    RETURNING status-code.
```

Interface

old-file-name
Specifies the old name of the file, ending with a blank or null character. The maximum size of the file name is 255.

new-file-name
Specifies the new name of the file, ending with a blank or null character. The maximum size of the file name is 255.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_RENAME_FILE2

CBL_RENAME_FILE2 changes the name of a file. The difference between the CBL_RENAME_FILE subroutine and this subroutine is that this subroutine can support a file name containing a space.

Specification

Parameter data definition

```
01 old-file-name PIC X(n).
01 new-file-name PIC X(n).
01 status-code PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_RENAME_FILE2"
USING old-file-name
new-file-name
RETURNING status-code.
```

Interface

old-file-name

Specifies the old name of the file, ending with a null character. The maximum size of the file name is 255.

new-file-name

Specifies the new name of the file, ending with a null character. The maximum size of the file name is 255.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code

0: Successful

Other than 0: Failure

PC_FIND_DRIVES

This routine returns information about usable disk drives.

Specification

Parameter data definition

```
01  drive-information:  PIC 9(9) BINARY.
```

Calling format

```
CALL "PC_FIND_DRIVES"
    USING drive-information.
```

Interface

drive-information
Specifies the area to store the disk drive information. Bits 0 to 25 correspond to disk drives A: to Z:.

Return code

None

PC_READ_DRIVE

This routine returns the name of the current drive.

Specification

Parameter data definition

```
01  drive:        PIC X.
01  status-code:  PIC S9(4) COMP-5.
```

Calling format

```
CALL "PC_READ_DRIVE"
    USING drive
    RETURNING status-code.
```

Interface

drive
Specifies the area to store the drive name. The drive name is returned with capital letters.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

PC_SET_DRIVE

This routine changes the current drive to another drive.

Specification

Parameter data definition

```
01  drive:        PIC X.
01  status-code:  PIC S9(4) COMP-5.
```

Calling format

```
CALL "PC_SET_DRIVE"
    USING drive
    RETURNING status-code.
```

Interface

drive
Specifies a uppercase or lowercase drive character.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

Chapter 5. Memory Allocation Routines

CBL_ALLOC_MEM

This routine allocates memory.

Specification

Parameter data definition

```
01  memory-pointer:  USAGE POINTER.
01  memory-size:     PIC S9(9) COMP-5.
01  flag:            PIC S9(9) COMP-5.
01  status-code:     PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_ALLOC_MEM"
    USING memory-pointer
    BY VALUE memory-size
             flag
    RETURNING status-code.
```

Interface

memory-pointer
Specifies the area to store the pointer for allocated memory.

memory-size
Specifies the size of memory to be allocated. The maximum value is 65531.

flag
Specifies the attributes of memory in units of bits. Bit information is as follows:

Bit 0

ON: Memory is shared.
OFF: Memory is not shared.

NOTE:
Non-shared memory is not supported.

Bit 1

ON: Allocated memory cannot be reallocated.
OFF: Allocated memory can be reallocated.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_FREE_MEM2

This routine releases memory.

Specification

Parameter data definition

```
01  memory-pointer:  Usage pointer.
```

Calling format

```
CALL "CBL_FREE_MEM2"
    USING memory-pointer
    RETURNING status-code.
```

Interface

memory-pointer
Specifies the pointer returned by CBL_ALLOC_MEM.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

Chapter 6. Virtual Heap Routines

CBL_OPEN_VFILE

This routine opens a map object.

Specification

Parameter data definition

```
01  heap-ID:      PIC 9(4) COMP-5.
01  heap-status:  PIC X(2).
01  status-code:  PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_OPEN_VFILE"
    USING heap-ID
          heap-status
    RETURNING status-code.
```

Interface

heap-ID
Specifies the area to store the heap-ID.

heap-status
Specifies the area to store heap-status.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_CLOSE_VFILE

This routine closes a map object.

Specification

Parameter data definition

```
01  heap-ID:      PIC 9(4) COMP-5.
01  status-code:  PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_CLOSE_VFILE"
    USING BY VALUE heap-ID
    RETURNING status-code.
```

Interface

heap-ID
Specifies the heap-ID.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_WRITE_VFILE

This routine writes data in a map object.

Specification

Parameter data definition

```
01  heap-ID:                   PIC 9(4) COMP-5.
01  relative-address-in-heap:  PIC 9(4) COMP-5.
01  data-length:               PIC 9(4) COMP-5.
01  string:                    PIC X(n).
01  status-code:               PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_WRITE_VFILE"
    USING BY VALUE heap-ID
          relative-address-in-heap
          data-length
          BY REFERENCE string
    RETURNING status-code.
```

Interface

heap-ID
Specifies the heap-ID.

relative-address-in-heap
Specifies the relative address in a heap.

data-length
Specifies the length of data to be written.

string
Specifies the string to be written.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_READ_VFILE

This routine reads data from a map object.

Specification

Parameter data definition

```
01  heap-ID:                   PIC 9(4) COMP-5.
01  relative-address-in-heap:  PIC 9(4) COMP-5.
01  data-length:               PIC 9(4) COMP-5.
01  string-storage-area:       PIC X(n).
01  status-code:               PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_READ_VFILE"
    USING BY VALUE heap-ID
          relative-address-in-heap
          data-length
          BY REFERENCE string-storage-area
    RETURNING status-code.
```

Interface

heap-ID
Specifies the heap-ID.

relative-address-in-heap
Specifies the relative address in a heap.

data-length
Specifies the length of data to be read.

string-storage-area
Specifies the area to store the read string.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

Chapter 7. Operating System Information Routines

CBL_GET_OS_INFO

This routine acquires information about the operating system.

Specification

Parameter data definition

```
01  parameter-block.
    02  parameter-size:            PIC 9(4) BINARY value 14.
    02  operating-system-type:     PIC 9(4) BINARY.
    02  operating-system-version:  PIC 9(9) BINARY.
    02  DBCS-support:              PIC 9(4) BINARY.
    02  character-code:            PIC 9(4) BINARY.
    02  country-ID:                PIC 9(4) BINARY.
    02  code-page:                 PIC 9(4) BINARY.
    02  processing-type:           PIC 9(9) BINARY.
01  status-code:                   PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_GET_OS_INFO"
    USING parameter-block
    RETURNING status-code.
```

Interface

operating-system-type
Specifies the area to store the type of operating system. The information to be stored is as follows:

128=UNIX
131=Windows

operating-system-version
Specifies the area to store the version of the operating system.

DBCS-support
Specifies the area to store DBCS support information. The information to be stored is as follows:

Bit 0

ON: DBCS validity check is supported.
OFF: DBCS validity check is not supported.

Bit 1

ON: Micro Focus PIC N is supported.
OFF: Micro Focus PIC N is not supported.

This subroutine always returns OFF to Bit1.

character-code
Specifies the area to store character code information. The information to be stored is as follows:

0: ASCII
1: Shift JIS
2: EUC

country-ID
Reserved area

code-page
Reserved area

processing-type
Specifies the area to store processing type information. The information to be stored is as follows:

0: Processing is executed as a full-screen session.
1: Processing is executed in a compatible box.
2: Processing is executed in a graphic character screen emulation window.
3: Processing is executed as a true graphical application.
4: Processing is executed independently.
5: Processing is executed independently in non-interactive mode.

This subroutine always returns 0.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

Chapter 8. Run-unit Handling Routines

CBL_YIELD_RUN_UNIT

This routine cancels the remaining time slice of the run unit.

Specification

Parameter data definition

None

Calling format

```
CALL "CBL_YIELD_RUN_UNIT".
```

Interface

None

Return code

None

Chapter 9. Text Routines

CBL_TOUPPER

This routine converts the characters of a string into uppercase characters.

Specification

Parameter data definition

```
01 string:       PIC X(n).
01 length:       PIC 9(4) COMP-5.
01 status-code:  PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_TOUPPER"
    USING string
    BY VALUE length
    RETURNING status-code.
```

Interface

string
Specifies the string to be converted. After conversion, the converted string is stored in this area.

length
Specifies the length of the string to be converted.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_TOLOWER

This routine converts the characters of a string into lowercase characters.

Specification

Parameter data definition

```
01 string:       PIC X(n).
01 length:       PIC 9(4) COMP-5.
01 status-code:  PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_TOLOWER"
    USING string
    BY VALUE length
    RETURNING status-code.
```

Interface

string
Specifies the string to be converted. After conversion, the converted string is stored in the same area.

length
Specifies the length of the string to be converted.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

Chapter 10. Logical Operator Routines

CBL_AND

This routine ANDs two items of data.

Specification

Parameter data definition

```
01 data-name-1:  Arbitrary item of data.
01 data-name-2:  Arbitrary item of data.
01 length:       PIC S9(9) COMP-5.
01 status-code:  PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_AND"
    USING data-name-1
          data-name-2
    BY VALUE length
    RETURNING status-code.
```

Interface

data-name-1
Specifies an item of data to be ANDed.

data-name-2
Specifies the other item of data to be ANDed. The result is stored in this area.

length
Specifies the length of data-name-1 and data-name-2.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_EQ

This routine obtains logical equivalence of two items of data.

Specification

Parameter data definition

```
01 data-name-1:  Arbitrary item of data.
01 data-name-2:  Arbitrary item of data.
01 length:       PIC 9(9) COMP-5.
01 status-code:  PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_EQ"
    USING data-name-1
          data-name-2
    BY VALUE length
    RETURNING status-code.
```

Interface

data-name-1
Specifies an item of data to obtain logical equivalence.

data-name-2
Specifies the other item of data to obtain logical equivalence. The result is stored in this area.

length
Specifies the length of data-name-1 and data-name-2.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_IMP

This routine obtains logical implication of two items of data.

Specification

Parameter data definition

```
01 data-name-1:  Arbitrary item of data.
01 data-name-2:  Arbitrary item of data.
01 length:       PIC 9(9) COMP-5.
01 status-code:  PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_IMP"
    USING data-name-1
          data-name-2
    BY VALUE length
    RETURNING status-code.
```

Interface

data-name-1
Specifies an item of data to obtain logical implication.

data-name-2
Specifies the other item of data to obtain logical implication. The result is stored in this area.

length
Specifies the length of data-name-1 and data-name-2.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_NOT

This routine obtains the logical negation of an item of data.

Specification

Parameter data definition

```
01 data-name-1:  Arbitrary item of data.
01 length:       PIC 9(9) COMP-5.
01 status-code:  PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_NOT"
    USING data-name-1
    BY VALUE length
    RETURNING status-code.
```

Interface

data-name-1
Specifies an item of data to obtain logical negation. The result is stored in this area.

length
Specifies the length of data-name-1.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_OR

This routine ORs two items of data.

Specification

Parameter data definition

```
01 data-name-1:  Arbitrary item of data.
01 data-name-2:  Arbitrary item of data.
01 length:       PIC 9(9) COMP-5.
01 status-code:  PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_OR"
    USING data-name-1
          data-name-2
    BY VALUE length
    RETURNING status-code.
```

Interface

data-name-1
Specifies an item of data to be ORed.

data-name-2
Specifies the other item of data to be ORed. The result is stored in this area.

length
Specifies the length of data-name-1 and data-name-2.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_XOR

This routine XORs two items of data.

Specification

Parameter data definition

```
01 data-name-1:  Arbitrary item of data.
01 data-name-2:  Arbitrary item of data.
01 length:       PIC 9(9) COMP-5.
01 status-code:  PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_XOR"
    USING data-name-1
          data-name-2
    BY VALUE length
    RETURNING status-code.
```

Interface

data-name-1
Specifies an item of data to be XORed.

data-name-2
Specifies the other item of data to be XORed. The result is stored in this area.

length
Specifies the length of data-name-1 and data-name-2.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

Chapter 11. Screen Routines

CBL_GET_CSR_POS

This routine returns the numeric values indicating the current line and column positions of the cursor. The top-left corner of the screen is assumed to be the origin (where the line and column numbers are 0).

Specification

Parameter data definition

```
01 text-cursor-position.
   02 line:    PIC 9(4) BINARY.
   02 column:  PIC 9(4) BINARY.
01 status-code:  PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_GET_CSR_POS"
    USING text-cursor-position
    RETURNING status-code.
```

Interface

line
Specifies the area to store the line position of the text cursor.

column
Specifies the area to store the column position of the text cursor.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

NOTE
Note the following points when using CBL_GET_CSR_POS:

If the text cursor is not found on the screen, 255 is set in the line and column parameters.

If the specified cursor position exceeds the size of the screen, this routine will fail.

CBL_SET_CSR_POS

This routine moves the cursor to the specified position. The cursor position is specified by line and column numbers. The top-left corner of the screen is assumed to be the origin (where line and column numbers are 0).

Specification

Parameter data definition

```
01 text-cursor-position.
   02 line:    PIC 9(4) BINARY.
   02 column:  PIC 9(4) BINARY.
01 status-code:  PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_SET_CSR_POS"
    USING text-cursor-position
    RETURNING status-code.
```

Interface

line
Specifies a new line position of the text cursor to be moved.

column
Specifies a new column position of the text cursor to be moved.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

NOTE
Note the following point when using CBL_SET_CSR_POS:

To delete the text cursor from the screen, specify 255 in the line and column parameters.

CBL_SET_CSR_SHAPE

This routine changes the size of the text cursor.

Specification

Parameter data definition

```
01 start-position:  PIC 9(4) BINARY.
01 end-position:    PIC 9(4) BINARY.
01 status-code:     PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_SET_CSR_POS"
    USING start-position
          end-position
    RETURNING status-code.
```

Interface

start-position
Specifies the start position using a numeric value from 0 to 11.

end-position
Specifies the end position using a numeric value from 0 to 11.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_CREATE_SCR

This routine creates a screen.

Specification

Parameter data definition

```
01 status-code: PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_CREATE_SCR"
    RETURNING status-code.
```

Interface

None

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_WRITE_SCR_TTY_CHAR

This routine writes a character at the current cursor position on the screen.

Specification

Parameter data definition

```
01 character:    PIC X(1).
01 status-code:  PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_WRITE_SCR_TTY_CHAR"
    USING character
    RETURNING status-code.
```

Interface

character
Specifies the character to be written on the screen.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_WRITE_SCR_TTY

This routine writes a string at the current cursor position on the screen.

Specification

Parameter data definition

```
01 string-storage-area:  PIC X(n).
01 string-length:        PIC 9(4) BINARY.
01 status-code:          PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_WRITE_SCR_TTY"
    USING string-storage-area
          string-length
    RETURNING status-code.
```

Interface

string-storage-area
Specifies the string to be written on the screen.

string-length
Specifies the length of the string to be written on the screen.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_CLEAR_SCR

This routine clears the screen by using a specified character with the specified attribute.

Specification

Parameter data definition

```
01 character:    PIC X(1).
01 attribute:    PIC 1(8) BIT.
01 status-code:  PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_CLEAR_SCR"
    USING character
          attribute
    RETURNING status-code.
```

Interface

character
Specifies the character to clear the screen.

attribute
Specifies the attribute of the character used to clear the screen.

Refer to “CBL_WRITE_SCR_ATTRS” for information about attributes.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_GET_SCR_SIZE

This routine obtains the logical screen size of the currently displayed window to be used for a screen operation function.

Specification

Parameter data definition

```
01 number-of-window-lines:    PIC 9(4) BINARY.
01 number-of-window-columns:  PIC 9(4) BINARY.
01 status-code:               PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_GET_SCR_SIZE"
    USING number-of-window-lines
          number-of-window-columns
    RETURNING status-code.
```

Interface

number-of-window-lines
Specifies the area used to set the number of window lines.

number-of-window-columns
Specifies the area to set the number of window columns.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_GET_SCR_GRAPHICS

This routine returns a one-byte or two-byte code to display a graphic character.

Specification

Parameter data definition

```
01 graphic-parameters.
   02 graphic-flag:         PIC 9(4) BINARY.
   02 graphic-buffer-size:  PIC 9(4) BINARY.
   02 graphic-buffer OCCURS 15.
      03 high-order-byte:  PIC X.
      03 low-order-byte:   PIC X.
01 status-code:            PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_GET_SCR_GRAPHICS"
    USING graphic-parameters
    RETURNING status-code.
```

Interface

graphic-flag
Specifies the graphic character code as follows:

0: The graphic character code suited to the screen is returned. (When one-byte code is valid, one-byte code is set in the graphic buffer. If one-byte code is invalid, two-byte code is set in the graphic buffer.)

1: One-byte code is returned.
2: Two-byte code is returned.

graphic-buffer-size
Specifies the size of the graphic buffer to be used for output. To obtain all graphic characters, 30 must be specified.

graphic-buffer
Specifies the area used to set the graphic code. When a one-byte graphic character code is obtained, a low value is set in the high-order byte and the graphic character code is set in the low-order byte. When a two-byte graphic character code is obtained, the graphic character code is set in the high-order and low-order bytes.

If no valid graphic character code is found, a low value is set in both the high-order and low-order bytes.

Examples of graphic characters are include the up arrow, down arrow, right arrow, left arrow, black up pointing triangle, black down pointing triangle, check, restoration, down zigzag arrow, scroll, black square, black diamond, closing angle bracket, opening angle bracket, and BTAB graphics.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
1: Requested graphic character code partially invalid
2: Reserved
3: Failure

NOTES
The graphic character gotten by this subroutine displays using Terminal Fonts.

There are graphic characters, which are not able to get, causing different operating system or locale of operating system.

On the operating system, which supports DBCS, one byte graphic character may not be gotten without Japan.

CBL_GET_SCR_LINE_DRAW

This routine returns a table drawn by using ruled-line characters consisting of a combination of no lines, single thick lines, double lines, and extended lines (thick lines), or returns a specified ruled-line character.

Specification

Parameter data definition

```
01 function-code:  PIC 9(4) BINARY.
01 line-drawing-table.
   For function-code 0:
   02 line-drawing-code:            PIC 1(8) BIT.
   02 DBCS-line-drawing-character:  PIC X occurs 256 times.
   For function-code 1:
   02 line-drawing-code:            PIC 1(8) BIT.
   02 DBCS-line-drawing-character:  PIC X(2) occurs 256 times.
   For function-code 2:
   02 line-drawing-code:            PIC 1(8) BIT.
   02 DBCS-line-drawing-character:  PIC X.
   For function-code 3:
   02 line-drawing-code:            PIC 1(8) BIT.
   02 DBCS-line-drawing-character:  PIC X(2).

01 status-code:                     PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_GET_SCR_LINE_DRAW"
     USING function-code
           line-drawing-table
     RETURNING status-code.
```

Interface

function-code
Specifies the function code as follows:

0: One-byte line-drawing table is returned.
1: Two-byte line-drawing table is returned.
2: One-byte line-drawing code is returned.
3: Two-byte line-drawing code is returned.

line-drawing-code
Specifies the line drawing code. (This parameter is valid only when function-code is 2 or 3.)

This parameter also indicates the line type that cannot be returned as follows:

Bit 0: Single thick line
Bit 1: Double line
Bit 2: Extended line
Bits 3 to 7: Reserved

DBCS-line-drawing-character
Specifies the area to set the line-drawing character to be set as follows:

For function-code 0: One-byte line-drawing table
For function-code 1: Two-byte line-drawing table

For function-code 2: One-byte line-drawing code
For function-code 3: Two-byte line-drawing code

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

NOTES
The graphic character gotten by this subroutine displays using Terminal Fonts.

There are graphic characters, which are not able to get, causing different operating system or locale of operating system.

On the operating system, which supports DBCS, one byte graphic character may not be gotten without Japan.

CBL_ALARM_SOUND

This routine generates an alarm sound.

Specification

Parameter data definition

```
None
```

Calling format

```
CALL "CBL_ALARM_SOUND".
```

Interface

None

Return code

None

CBL_BELL_SOUND

This routine generates an inquiry sound.

Specification

Parameter data definition

```
None
```

Calling format

```
CALL "CBL_BELL_SOUND".
```

Interface

None

Return code

None

CBL_GET_VGA_MODE

This routine returns a screen type.

Specification

Parameter data definition

```
01 screen-type  PIC 1(8) BIT.
```

Calling format

```
CALL "CBL_GET_VGA_MODE"
    USING screen-type.
```

Interface

screen-type
Specifies the area used to store the screen type as follows:

Bit 0: Monochrome screen
Bit 1: Reserved
Bit 2: Reserved
Bit 3: EGA-type screen
Bit 4: VGA-type screen
Bit 5: Reserved
Bit 6: Reserved
Bit 7: Reserved

Return code

None

NOTES
This subroutine returns only "VGA-type screen".

CBL_WRITE_SCR_ATTRS

This routine writes a string of attributes at the specified position on the screen.

Specification

Parameter data definition

```
01 position-on-screen.
   02 line-position:    PIC 9(4) BINARY.
   02 column-position:  PIC 9(4) BINARY.
01 attribute-buffer.
   02 attribute:        PIC 1(8) BIT occurs n times.
01 string-length:       PIC 9(4) BINARY.
01 status-code:         PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_WRITE_SCR_ATTRS"
USING position-on-screen
      attribute-buffer
      string-length
RETURNING status-code.
```

Interface

line-position
Specifies a line by using a numeric value with the upper-most line assumed to be the origin (0).

column-position
Specifies a column by using a numeric value with the left-most column assumed to be the origin (0).

attribute-buffer
Specifies the attributes to be displayed.

Attributes are as follows.

B"10000000" OVERLINE
B"01000000" REVERSE-VIDEO
B"00100000" GRID
B"00010000" UNDERLINE
B"00001000" HIGHLIGHT
B"00000100" FOREGROUND-COLOR(RED)
B"00000010" FOREGROUND-COLOR(GREEN)
B"00000001" FOREGROUND-COLOR(BLUE)

For UNIX systems:

B”10000000” Reserved
B”01000000” Reserved
B”00100000” Reserved
B”00010000” Reserved
B”00001000” BLANK
B”00000100” REVERSE-VIDEO
B”00000010” UNDERLINE
B”00000001” HIGHLIGHT

string-length
Specifies the length of the string to be displayed.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_WRITE_SCR_CHARS

This routine writes a string at the specified position on the screen.

Specification

Parameter data definition

```
01 position-on-screen.
   02 line-position:    PIC 9(4) BINARY.
   02 column-position:  PIC 9(4) BINARY.
01 string-buffer:       PIC X(n).
01 string-length:       PIC 9(4) BINARY.
01 status-code:         PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_WRITE_SCR_CHARS"
    USING position-on-screen
          string-buffer
          string-length
    RETURNING status-code.
```

Interface

line-position
Specifies a line by using a numeric value with the upper-most line assumed to be the origin (0).

column-position
Specifies a column by using a numeric value with the left-most column assumed to be the origin (0).

string-buffer
Specifies the string to be written on the screen.

string-length
Specifies the length of the string to be written.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_WRITE_SCR_CHARS_ATTR

This routine writes a string at the specified position on the screen and sets the same attribute for all characters of the string.

Specification

Parameter data definition

```
01 position-on-screen.
   02 line-position:    PIC 9(4) BINARY.
   02 column-position:  PIC 9(4) BINARY.
01 string-buffer:       PIC X(n).
01 string-length:       PIC 9(4) BINARY.
01 attribute:           PIC 1(8) BIT.
01 status-code:         PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_WRITE_SCR_CHARS_ATTR"
    USING position-on-screen
          string-buffer
          string-length
          attribute
    RETURNING status-code.
```

Interface

line-position
Specifies a line by using a numeric value with the upper-most line assumed to be the origin (0).

column-position
Specifies a column by using a numeric value with the left-most column assumed to be the origin (0).

string-buffer
Specifies the string to be written on the screen.

string-length
Specifies the length of the string to be written.

attribute
Specifies the attribute to be set for the string to be written.

Refer to “CBL_WRITE_SCR_ATTRS” about attributes.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_WRITE_SCR_CHATTRS

This routine writes a string and the attribute of each character of the string at the specified position on the screen.

Specification

Parameter data definition

```
01 position-on-screen.
   02 line-position:    PIC 9(4) BINARY.
   02 column-position:  PIC 9(4) BINARY.
01 character-buffer:    PIC X(n).
01 attribute-buffer.
   02 attribute:        PIC 1(8) BIT occurs n times.
01 string-length:       PIC 9(4) BINARY.
01 status-code:         PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_WRITE_SCR_CHATTRS"
    USING position-on-screen
          string-buffer
          attribute-buffer
          string-length
    RETURNING status-code.
```

Interface

line-position
Specifies a line by using a numeric value with the upper-most line assumed to be the origin (0).

column-position
Specifies a column by using a numeric value with the left-most column assumed to be the origin (0).

character-buffer
Specifies the string to be written on the screen.

attribute-buffer
Specifies the attribute to be set for the string to be written on the screen. Refer to “CBL_WRITE_SCR_ATTRS” about attributes.

string-length
Specifies the length of the string to be written.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_WRITE_SCR_N_ATTR

This routine sets a specified number of specified attributes from the specified position on the screen.

Specification

Parameter data definition

```
01 position-on-screen.
   02 line-position:      PIC 9(4) BINARY.
   02 column-position:    PIC 9(4) BINARY.
01 attribute:             PIC 1(8) BIT.
01 number-of-attributes:  PIC 9(4) BINARY.
01 status-code:           PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_WRITE_SCR_N_ATTR"
    USING position-on-screen
          attribute
          number-of-attributes
    RETURNING status-code.
```

Interface

line-position
Specifies a line by using a numeric value with the upper-most line assumed to be the origin (0).

column-position
Specifies a column by using a numeric value with the left-most column assumed to be the origin (0).

attribute
Specifies the attribute to be written on the screen. Refer to "CBL_WRITE_SCR_ATTRS" about attributes.

number-of-attributes
Specifies the number of attributes to be set on the screen.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_WRITE_SCR_N_CHAR

This routine sets a specified number of specified characters from the specified position on the screen.

Specification

Parameter data definition

```
01 position-on-screen.
   02 line-position:      PIC 9(4) BINARY.
   02 column-position:    PIC 9(4) BINARY.
01 character:             PIC X.
01 number-of-characters:  PIC 9(4) BINARY.
01 status-code:           PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_WRITE_SCR_N_CHAR"
    USING position-on-screen
          character
          number-of-characters
    RETURNING status-code.
```

Interface

line-position
Specifies a line by using a numeric value with the upper-most line assumed to be the origin (0).

column-position
Specifies a column by using a numeric value with the left-most column assumed to be the origin (0).

character
Specifies the character to be written on the screen.

number-of-characters
Specifies the number of characters to be set on the screen.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_WRITE_SCR_N_CHATTR

This routine sets a specified number of specified characters having specified attributes from the specified position on the screen.

Specification

Parameter data definition

```
01 position-on-screen.
   02 line-position:      PIC 9(4) BINARY.
   02 column-position:    PIC 9(4) BINARY.
01 character:             PIC X.
01 attribute:             PIC 1(8) BIT.
01 number-of-characters:  PIC 9(4) BINARY.
01 status-code:           PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_WRITE_SCR_N_CHATTR"
    USING position-on-screen
          character
          attribute
          number-of-characters
    RETURNING status-code.
```

Interface

line-position
Specifies a line by using a numeric value with the upper-most line assumed to be the origin (0).

column-position
Specifies a column by using a numeric value with the left-most column assumed to be the origin (0).

character
Specifies the character to be written on the screen.

attribute
Specifies the attribute to be set for the character to be written on the screen. Refer to “CBL_WRITE_SCR_ATTRS” about attributes.

number-of-characters
Specifies the number of characters and attributes to be written on the screen.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_READ_SCR_ATTRS

This routine acquires the attributes of the specified length of a string beginning at the specified position on the screen.

Specification

Parameter data definition

```
01 position-on-screen.
   02 line-position:      PIC 9(4) BINARY.
   02 column-position:    PIC 9(4) BINARY.
01 attribute-buffer.
   02 attribute:          PIC 1(8) BIT occurs n times.
01 string-length:         PIC 9(4) BINARY.
01 status-code:           PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_READ_SCR_ATTRS"
    USING position-on-screen
          attribute-buffer
          string-length
    RETURNING status-code.
```

Interface

line-position
Specifies a line by using a numeric value with the upper-most line assumed to be the origin (0).

column-position
Specifies a column by using a numeric value with the left-most column assumed to be the origin (0).

attribute-buffer
Specifies the area used to store the attributes to be acquired. Refer to "CBL_WRITE_SCR_ATTRS" about attributes.

string-length
Specifies the length of the string of which attributes are to be acquired.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_READ_SCR_CHARS

This routine acquires the specified length of a string beginning at the specified position on the screen.

Specification

Parameter data definition

```
01 position-on-screen.
   02 line-position:      PIC 9(4) BINARY.
   02 column-position:    PIC 9(4) BINARY.
01  character-buffer:     PIC X(n).
01  string-length:        PIC 9(4) BINARY.
01  status-code:          PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_READ_SCR_CHARS"
    USING position-on-screen
          character-buffer
          string-length
    RETURNING status-code.
```

Interface

line-position
Specifies a line by using a numeric value with the upper-most line assumed to be the origin (0).

column-position
Specifies a column by using a numeric value with the left-most column assumed to be the origin (0).

character-buffer
Specifies the area used to store the string to be acquired.

string-length
Specifies the length of the string to be acquired.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_READ_SCR_CHATTRS

This routine acquires the specified length of a string (beginning at the specified position on the screen) and the attributes of the string characters.

Specification

Parameter data definition

```
01 position-on-screen.
   02 line-position:      PIC 9(4) BINARY.
   02 column-position:    PIC 9(4) BINARY.
01 character-buffer:      PIC X(n).
01 attribute-buffer.
   02 attribute:          PIC 1(8) BIT occurs n times.
01 string-length:         PIC 9(4) BINARY.
01 status-code:           PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_READ_SCR_CHATTRS"
    USING position-on-screen
          character-buffer
          attribute-buffer
          string-length
    RETURNING status-code.
```

Interface

line-position
Specifies a line by using a numeric value with the upper-most line assumed to be the origin (0).

column-position
Specifies a column by using a numeric value with the left-most column assumed to be the origin (0).

character-buffer
Specifies the area used to store the string to be acquired.

attribute-buffer
Specifies the area used to store the attributes to be acquired. Refer to "CBL_WRITE_SCR_ATTRS" for information about attributes.

string-length
Specifies the length of the string to be acquired.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_SWAP_SCR_CHATTRS

This routine replaces the specified length of a string (beginning at the specified position on the screen) and its character attributes with a new string and attributes.

Specification

Parameter data definition

```
01 position-on-screen.
   02 line-position:    PIC 9(4) BINARY.
   02 column-position:  PIC 9(4) BINARY.
01 character-buffer:    PIC X(n).
01 attribute-buffer.
   02 attribute:        PIC 1(8) BIT occurs n times.
01 string-length:       PIC 9(4) BINARY.
01 status-code:         PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_SWAP_SCR_CHATTRS"
    USING position-on-screen
          character-buffer
          attribute-buffer
          string-length
    RETURNING status-code.
```

Interface

line-position
Specifies a line by using a numeric value with the upper-most line assumed to be the origin (0).

column-position
Specifies a column by using a numeric value with the left-most column assumed to be the origin (0).

character-buffer
Specifies the string to be replaced. This parameter sets the replaced string.

attribute-buffer
Specifies the attributes to be replaced. This parameter sets the replaced attributes. Refer to “CBL_WRITE_SCR_ATTRS” about attributes.

string-length
Specifies the length of the string to be replaced.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_SET_SCR_TERMKEY

This routine changes active/inactive information of Termination Key that can be acquired in the CRT STATUS clause.

Specification

Parameter data definition

```
01 crt-status-key-1:        PIC X.
01 start-crt-status-key-2:  PIC X(3).
01 end-crt-status-key-2:    PIC X(3).
01 flag:                    PIC X.
01 status-code:         PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_SET_SCR_TERMKEY"
    USING crt-status-key-1
          start-crt-status-key-2
          end-crt-status-key-2
          flag
    RETURNING status-code.
```

Interface

crt-status-key-1
crt-status-key-1 is the first status of a termination key set to activated or deactivated.

"1": crt-status-key-1 is "1".
"2": crt-status-key-1 is "2".

start-crt-status-key-2
The beginning value of crt-status-key-2 of the termination key that sets activated/deactivated is specified by a number from "000" to "999".

end-crt-status-key-2
The end value of crt-status-key-2 of termination key which sets activated/deactivated is specified by a number from "000" to "999". If the value of specified crt-status-key-2 is one, the same value as the beginning value and the end value is set.

flag
Whether specified termination key is activated or it is deactivated is specified.

"0": Deactivated is assumed.

"1": Activated is assumed.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code

0: Successful
Other than 0: Failure

NOTES
When this routine executes, it is necessary to specify execution environment information @CBR_SCR_KEYDEFFILE beforehand.

CBL_SET_SCR_KEYFILE

The environment of termination Key is set again according to information of the specified key definition file.

Specification

Parameter data definition

```
01  key-definition-file-name:   PIC X(n).
01  file-name-length:           PIC 9(9) BINARY.
01  status-code:                PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_SET_SCR_KEYFILE"
    USING key-definition-file-name
          file-name-length
    RETURNING status-code.
```

Interface

key-definition-file-name
The key definition file name is set.

file-name-length
Specifies the length of the key definition file name.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

NOTES
When this routine executes, it is necessary to specify execution environment information @CBR_SCR_KEYDEFFILE beforehand.

CBL_READ_SCR_KEY

The key input is waited, and if the acquired key is a character key, the character is returned. If the key is a key which can be acquired in the CRT STATUS clause, crt-status-key-1 and crt-status-key-2 of the key are returned.

Specification

Parameter data definition

```
01  key-type:     PIC X.
01  key-code:     PIC X(2).
01  status-code:  PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_READ_SCR_KEY"
    USING key-type
          key-code
    RETURNING status-code.
```

Interface

key-type
The type of the input key is set.

"1": The input key is activated key, and crt-status-key-1 is "1".

"2": The input key is activated key, and crt-status-key-1 is "2".

"3": The input key is character key(1 byte).

"4": The input key is character key(2 bytes).

"9": The input key is deactivated key.

key-code

Detailed information of the input key is set. As for the content of X(4), the reference method is different according to the key-type as follows.

key-type is "1" or "2"

crt-status-key-2 is set on STSCODE.

```
01  STSKEY2.
    02  STSCODE       PIC  9(4)  BINARY.
```

key-type is "3"

The input character is set on SBCSCODE.

```
01  SBCSKEY.
    02  FILLER    PIC X.
    02  SBCSCODE  PIC X.
```

key-type is "4"

The input 2-byte-character is set on DBCSCODE.

```
01  DBCSKEY.
    02  DBCSCODE  PIC X(2).
```

key-type is "9"

The deactivated key type is set on ERCODE.

```
01  INVALIDKEY.
    02  FILLER  PIC X.
    02  ERCODE  PIC 9(2) BINARY.
```

The deactivated key type is as follows.

9: The deactivated function key is entered.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

NOTES
When this routine executes, it is necessary to specify execution environment information @CBR_SCR_KEYDEFFILE beforehand.

If you use this CBL routine with key-type=9, you have to compile your cobol program with compile option ”BINARY(BYTE)”.

CBL_INIT_SCR_ACCEPT_ATTR

This subroutine changes the mode whether the attributes of input field are initialized or no when ACCEPT statement is executed.

Specification

Parameter data definition

```
01 init-mode PIC 9(4) BINARY.
```

Calling format

```
CALL "CBL_INIT_SCR_ACCEPT_ATTR "
USING init-mode
RETURNING status-code.
```

Interface

Init-mode

Specify the following value.

0: The attributes of input field are initialized when ACCEPT statement is executed. (Default mode)

1: The attributes of input field are not initialized when ACCEPT statement is executed.

This mode is not changed until next this subroutine is called.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code

0: Successful
Other than 0: Failure

Chapter 12. Mouse Routines

CBL_GET_MOUSE_MASK

This routine acquires a mouse event mask.

Specification

Parameter data definition

```
01 mouse-handle:  PIC X(4).
01 event-mask:    PIC 1(16) BIT.
01 status-code:   PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_GET_MOUSE_MASK"
    USING mouse-handle
          event-mask
    RETURNING status-code.
```

Interface

mouse-handle
Specifies the mouse-handle returned by CBL_INIT_MOUSE.

event-mask
Specifies the area used to store event mask information as follows:

Bit 0: The mouse was moved.
Bit 1: The left mouse button was pressed.
Bit 2: The middle mouse button was pressed.
Bit 3: The right mouse button was pressed.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_GET_MOUSE_POSITION

This routine returns the line and column positions of the current mouse cursor on the current screen.

Specification

Parameter data definition

```
01 mouse-handle:        PIC X(4).
01 mouse-cursor-position.
   02 line-position:    PIC S9(4) BINARY.
   02 column-position:  PIC S9(4) BINARY.
01 status-code:         PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_GET_MOUSE_POSITION"
    USING mouse-handle
          mouse-cursor-position
    RETURNING status-code.
```

Interface

mouse-handle
Specifies the mouse-handle returned by CBL_INIT_MOUSE.

line-position
Specifies the area used to store the numeric value indicating the line position of the mouse cursor. The top-left corner of the screen is assumed to be the origin (where the line and column numbers are 0).

column-position
Specifies the area used to store the numeric value indicating the column position of the mouse cursor. The top-left corner of the screen is assumed to be the origin (where the line and column numbers are 0).

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_GET_MOUSE_STATUS

This routine returns the number of current event masks.

Specification

parameter data definition

```
01 mouse-handle:           PIC X(4).
01 number-of-mask-events:  PIC 9(4) BINARY.
01 status-code:            PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_GET_MOUSE_STATUS"
    USING mouse-handle
          number-of-event-masks
    RETURNING status-code.
```

Interface

mouse-handle
Specifies the mouse-handle returned by CBL_INIT_MOUSE.

number-of-event-masks
Specifies the area used to store the number of event masks.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_HIDE_MOUSE

This routine hides the mouse cursor.

Specification

Parameter data definition

```
01 mouse-handle:           PIC X(4).
01 status-code:            PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_HIDE_MOUSE"
    USING mouse-handle
    RETURNING status-code.
```

Interface

mouse-handle
Specifies the mouse-handle returned by CBL_INIT_MOUSE.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_INIT_MOUSE

This routine enables mouse processing of the CBL routines.

Specification

Parameter data definition

```
01 mouse-handle:          PIC X(4).
01 status-code:           PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_INIT_MOUSE"
    USING mouse-handle
    RETURNING status-code.
```

Interface

mouse-handle
Specifies the area used to store the mouse handle.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_READ_MOUSE_EVENT

This routine acquires the mouse event that meets an event mask condition.

Specification

Parameter data definition

```
01 mouse-handle:    PIC X(4).
01 mouse-event.
   02 event-mask:   PIC 1(16) BIT.
   02 time:         PIC 9(9) BINARY.
   02 mouse-cursor-position.
      03 line-position:  PIC 9(4) BINARY.
      03 column-position:  PIC 9(4) BINARY.
01 read-wait flag:  PIC 9(4) BINARY.
01 status-code:     PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_READ_MOUSE_EVENT"
    USING mouse-handle
          mouse-event
          read-wait-flag
    RETURNING status-code.
```

Interface

mouse-handle
Specifies the mouse-handle returned by CBL_INIT_MOUSE.

event-mask
Specifies the area used to store the event mask.

time
Specifies the area used to store the time.

read-wait-flag
Specifies read-wait flag information as follows:

0: The system waits until the mouse event occurs.
1: The system acquires information immediately even if the mouse event does not occur.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_SET_MOUSE_MASK

This routine sets a specific condition (event mask) for the mouse event acquired by CBL_READ_MOUSE_EVENT.

Specification

Parameter data definition

```
01 mouse-handle:  PIC X(4).
01 event-mask:    PIC 1(16) BIT.
01 status-code:   PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_SET_MOUSE_MASK"
    USING mouse-handle
          event-mask
    RETURNING status-code.
```

Interface

mouse-handle
Specifies the mouse-handle returned by CBL_INIT_MOUSE.

event-mask
Specifies the area used to store event mask information as follows:

Bit 0: The mouse was moved.
Bit 1: The left mouse button was pressed.
Bit 2: The middle mouse button was pressed.
Bit 3: The right mouse button was pressed.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_SET_MOUSE_POSITION

This routine moves the mouse cursor to the specified position on the screen.

Specification

Parameter data definition

```
01 mouse-handle:        PIC X(4).
01 mouse-cursor-position.
   02 line-position:    PIC 9(4) BINARY.
   02 column-position:  PIC 9(4) BINARY.
01 status-code:         PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_SET_MOUSE_POSITION"
    USING mouse-handle
          mouse-cursor-position
    RETURNING status-code.
```

Interface

mouse-handle
Specifies the mouse-handle returned by CBL_INIT_MOUSE.

line-position
Specifies the numeric value indicating the line position to which the mouse cursor is to be moved. The top-left corner of the screen is assumed to be the origin (where the line and column numbers are 0).

column-position
Specifies the numeric value indicating the column position to which the mouse cursor is to be moved. The top-left corner of the screen is assumed to be the origin (where the line and column numbers are 0).

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_SHOW_MOUSE

This routine displays the mouse cursor.

Specification

Parameter data definition

```
01 mouse-handle:  PIC X(4).
01 status-code:   PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_SHOW_MOUSE"
    USING mouse-handle
    RETURNING status-code.
```

Interface

mouse-handle
Specifies the mouse-handle returned by CBL_INIT_MOUSE.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_TERM_MOUSE

This routine disables mouse processing by the CBL routines.

Specification

Parameter data definition

```
01 mouse-handle:  PIC X(4).
01 status-code:   PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_TERM_MOUSE"
    USING mouse-handle
    RETURNING status-code.
```

Interface

mouse-handle
Specifies the mouse-handle returned by CBL_INIT_MOUSE.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

WIN_GET_MOUSE_SHAPE

This routine acquires the shape of the mouse cursor.

Specification

Parameter data definition

```
01 mouse-handle:                              PIC X(4).
01 path-name-of-mouse-cursor-shape:           PIC X(n).
01 length-of-path-name-of-mouse-cursor-shape: PIC 9(4) BINARY.
01 status-code:                               PIC S9(4) COMP-5.
```

Calling format

```
CALL "WIN_GET_MOUSE_SHAPE"
    USING mouse-handle
          path-name-for-mouse-cursor-shape
          length-of-path-name-for-mouse-cursor-shape
    RETURNING status-code.
```

Interface

mouse-handle
Specifies the mouse-handle returned by CBL_INIT_MOUSE.

path-name-for-mouse-cursor-shape
Specifies the area used to store the path name for the mouse cursor shape.

length-of-path-name-for-mouse-cursor-shape
Specifies the area used to store the length of the path name for the mouse cursor shape.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

WIN_SET_MOUSE_SHAPE

This routine changes the shape of the mouse cursor.

Ex. “C:\WINDOWS\CURSORS\ARROW.CUR”, “C:\WINDOWS\CURSORS\ARROW.ANI”

Specification

Parameter data definition

```
01 mouse-handle:                              PIC X(4).
01 path-name-of-mouse-cursor-shape:           PIC X(n).
01 length-of-path-name-of-mouse-cursor-shape: PIC 9(4) BINARY.
01 status-code:                               PIC S9(4) COMP-5.
```

Calling format

```
CALL "WIN_SET_MOUSE_SHAPE"
    USING mouse-handle
          path-name-for-mouse-cursor-shape
          length-of-path-name-for-mouse-cursor-shape
    RETURNING status-code.
```

Interface

mouse-handle
Specifies the mouse-handle returned by CBL_INIT_MOUSE.

path-name-for-mouse-cursor-shape
Specifies the path name for the mouse cursor shape.

length-of-path-name-for-mouse-cursor-shape
Specifies the length of the path name for the mouse cursor shape.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

Chapter 13. Keyboard Routines

CBL_GET_KBD_STATUS

This routine checks whether a character was input from the keyboard.

Specification

Parameter data definition

```
01 key-status:   PIC 9(9) BINARY.
01 status-code:  PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_GET_KBD_STATUS"
    USING key-status
    RETURNING status-code.
```

Interface

key-status
Specifies the area used to store key status information as follows:

0: No character was input.
1: A character was input.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

CBL_READ_KBD_CHAR

This routine waits until a character is input from the keyboard, then reads the input character without displaying it.

Specification

Parameter data definition

```
01 character:    PIC X.
01 status-code:  PIC S9(4) COMP-5.
```

Calling format

```
CALL "CBL_READ_KBD_CHAR"
    USING character
    RETURNING status-code.
```

Interface

character
Specifies the area used to store the input character.

Return code

The return code is set in the status-code specified in the RETURNING clause.

status-code
0: Successful
Other than 0: Failure

Appendix A. List of CBL Routines

Table A.1 List of CBL_ Routines

No	Fujitsu				Micro Focus
	Category / Name	**Win**	**Sol**	**HP**	**Name**
-	**Byte-Stream File Routine**				
1	CBL_CLOSE_FILE	Y	Y	Y	CBL_CLOSE_FILE
2	CBL_CREATE_FILE	Y	Y	Y	CBL_CREATE_FILE
3	CBL_CREATE_FILE2	Y	-	-	-
4	CBL_FLUSH_FILE	Y	Y	Y	CBL_FLUSH_FILE
5	CBL_OPEN_FILE	Y	Y	Y	CBL_OPEN_FILE
6	CBL_OPEN_FILE2	Y	-	-	-
7	CBL_READ_FILE	Y	Y	Y	CBL_READ_FILE
8	CBL_WRITE_FILE	Y	Y	Y	CBL_WRITE_FILE
9	CBL_CLOSE_64BIT_FILE	-	Y	-	-
10	CBL_CREATE_64BIT_FILE	-	Y	-	-
11	CBL_FLUSH_64BIT_FILE	-	Y	-	-
12	CBL_OPEN_64BIT_FILE	-	Y	-	-
13	CBL_READ_64BIT_FILE	-	Y	-	-
14	CBL_WRITE_64BIT_FILE	-	Y	-	-
Filename Routines					
15	CBL_JOIN_FILENAME	Y	Y	Y	CBL_JOIN_FILENAME
16	CBL_SPLIT_FILENAME	Y	Y	Y	CBL_SPLIT_FILENAME
File Routines					
17	CBL_CHANGE_DIR	Y	Y	Y	CBL_CHANGE_DIR
18	CBL_CHANGE_DIR2	Y	-	-	-
19	CBL_CHECK_FILE_EXIST	Y	Y	Y	CBL_CHECK_FILE_EXIST
20	CBL_CHECK_FILE_EXIST2	Y	-	-	-
21	CBL_COPY_FILE	Y	Y	Y	CBL_COPY_FILE
22	CBL_COPY_FILE2	Y	-	-	-
23	CBL_CREATE_DIR	Y	Y	Y	CBL_REATE_DIR
24	CBL_CREATE_DIR2	Y	-	-	-
25	CBL_DELETE_DIR	Y	Y	Y	CBL_DELETE_DIR
26	CBL_DELETE_DIR2	Y	-	-	-
27	CBL_DELETE_FILE	Y	Y	Y	CBL_DELETE_FILE
28	CBL_DELETE_FILE2	Y	-	-	-
29	CBL_LOCATE_FILE	Y	Y	Y	CBL_LOCATE_FILE
30	CBL_LOCATE_FILE2	Y	-	-	-
31	CBL_READ_DIR	Y	Y	Y	CBL_READ_DIR
32	CBL_RENAME_FILE	Y	Y	Y	CBL_RENAME_FILE

No	Fujitsu				Micro Focus
	Category/Name	**Win**	**Sol**	**HP**	**Name**
33	CBL_RENAME_FILE2	Y	-	-	-
34	PC_FIND_DRIVES	Y	-	-	PC_FIND_DRIVES
35	PC_READ_DRIVE	Y	-	-	PC_READ_DRIVE
36	PC_SET_DRIVE	Y	-	-	PC_SET_DRIVE
Memory Allocation Routines					
37	CBL_ALLOC_MEM	Y	Y	Y	CBL_ALLOC_MEM
38	CBL_FREE_MEM2	Y	Y	Y	CBL_FREE_MEM
Virtual Heap Routines					
39	CBL_OPEN_VFILE	Y	-	-	CBL_OPEN_VFILE
40	CBL_CLOSE_VFILE	Y	-	-	CBL_CLOSE_VFILE
41	CBL_WRITE_VFILE	Y	-	-	CBL_WRITE_VFILE
42	CBL_READ_VFILE	Y	-	-	CBL_READVFILE
Operating System Information Routines					
43	CBL_GET_OS_INFO	Y	Y	Y	CBL_GET_OS_INFO
Run-Unit Handling Routines					
44	CBL_YIELD_RUN_UNIT	Y	-	-	CBL_YIELD_RUN_UNIT
-	**Text Routines**				
45	CBL_TOUPPER	Y	Y	Y	CBL_TOUPPER
46	CBL_TOLOWER	Y	Y	Y	CBL_TOLOWER
Logical Operator Routines					
47	CBL_AND	Y	Y	Y	CBL_AND
48	CBL_EQ	Y	Y	Y	CBL_EQ
49	CBL_IMP	Y	Y	Y	CBL_IMP
50	CBL_NOT	Y	Y	Y	CBL_NOT
51	CBL_OR	Y	Y	Y	CBL_OR
52	CBL_XOR	Y	Y	Y	CBL_XOR
Screen Routines					
53	CBL_GET_CSR_POS	Y	-	Y	CBL_GET_CSR_POS
54	CBL_SET_CSR_POS	Y	-	Y	CBL_SET_CSR_POS
55	CBL_SET_CSR_SHAPE	Y	-	-	X”A7” Function 17
56	CBL_CREATE_SCR	Y	-	-	-
57	CBL_WRITE_SCR_TTY_CHAR	Y	-	Y	X”AF” Function 18
58	CBL_WRITE_SCR_TTY	Y	-	Y	CBL_WRITE_SCR_TTY
59	CBL_CLEAR_SCR	Y	-	Y	CBL_CLEAR_SCR
60	CBL_GET_SCR_SIZE	Y	-	Y	CBL_GET_SCR_SIZE
61	CBL_GET_SCR_GRAPHICS	Y	-	-	CBL_GET_SCR_GRAPHICS
62	CBL_GET_SCR_LINE_DRAW	Y	-	-	CBL_GET_SCR_LINE_DRAW
63	CBL_ALARM_SOUND	Y	-	Y	X”AF” Function 22
64	CBL_BELL_SOUND	Y	-	Y	X”E5”
65	CBL_GET_VGA_MODE	Y	-	-	X”A7” Function 25

No	Fujitsu				Micro Focus
	Category / Name	**Win**	**Sol**	**HP**	**Name**
66	CBL_WRITE_SCR_ATTRS	Y	-	Y	CBL_WRITE_SCR_ATTRS
67	CBL_WRITE_SCR_CHARS	Y	-	Y	CBL_WRITE_SCR_CHARS
68	CBL_WRITE_SCR_CHARS_ATTR	Y	-	Y	CBL_WRITE_SCR_CHARS_ATTR
69	CBL_WRITE_SCR_CHATTRS	Y	-	Y	CBL_WRITE_SCR_CHATTRS
70	CBL_WRITE_SCR_N_ATTR	Y	-	Y	CBL_WRITE_SCR_N_ATTR
71	CBL_WRITE_SCR_N_CHAR	Y	-	Y	CBL_WRITE_SCR_N_CHAR
72	CBL_WRITE_SCR_N_CHATTR	Y	-	Y	CBL_WRITE_SCR_N_CHATTR
73	CBL_READ_SCR_ATTRS	Y	-	Y	CBL_READ_SCR_ATTRS
74	CBL_READ_SCR_CHARS	Y	-	Y	CBL_READ_SCR_CHARS
75	CBL_READ_SCR_CHATTRS	Y	-	Y	CBL_READ_SCR_CHATTRS
76	CBL_SWAP_SCR_CHATTRS	Y	-	Y	CBL_SWAP_SCR_CHATTRS
77	CBL_SET_SCR_TERMKEY	Y	-	Y	X”AF” Function 1
78	CBL_SET_SCR_KEYFILE	Y	-	Y	-
79	CBL_READ_SCR_KEY	Y	-	Y	X”AF” Function 26
80	CBL_INIT_SCR_ACCEPT_ATTR	Y	-	-	-
Mouse Routines					
81	CBL_GET_MOUSE_MASK	Y	-	-	CBL_GET_MOUSE_MASK
82	CBL_GET_MOUSE_POSITION	Y	-	-	CBL_GET_MOUSE_POSITION
83	CBL_GET_MOUSE_STATUS	Y	-	-	CBL_GET_MOUSE_STATUS
84	CBL_HIDE_MOUSE	Y	-	-	CBL_HIDE_MOUSE
85	CBL_INIT_MOUSE	Y	-	-	CBL_INIT_MOUSE
86	CBL_READ_MOUSE_EVENT	Y	-	-	CBL_READ_MOUSE_EVENT
87	CBL_SET_MOUSE_MASK	Y	-	-	CBL_SET_MOUSE_MASK
88	CBL_SET_MOUSE_POSITION	Y	-	-	CBL_SET_MOUSE_POSITION
89	CBL_SHOW_MOUSE	Y	-	-	CBL_SHOW_MOUSE
90	CBL_TERM_MOUSE	Y	-	-	CBL_TERM_MOUSE
91	WIN_GET_MOUSE_SHAPE	Y	-	-	PC_GET_MOUSE_SHAPE
92	WIN_SET_MOUSE_SHAPE	Y	-	-	PC_SET_MOUSE_SHAPE
Keyboard Routines					
93	CBL_GET_KBD_STATUS	Y	-	Y	CBL_GET_KBD_STATUS
94	CBL_READ_KBD_CHAR	Y	-	Y	CBL_READ_KBD_CHAR