



# C言語プログラミング Part I:基本、ハードウェア制御編 for 8 Series Mobile Computers

英文・和文で相違がある場合は、英文を優先して解釈をお願いします





## 目次

はじめに .....	VII
<b>1 開発環境 .....</b>	<b>1</b>
1.1 フォルダ構成と変数 .....	2
1.1.1 フォルダ構成 .....	2
1.1.2 環境設定 .....	3
1.2 プログラム開発ツール .....	4
1.2.1 Cソースプログラム .....	4
1.2.2 コンパイル .....	5
1.2.3 リンク .....	6
1.2.4 フォーマット変換 .....	8
1.2.5 プログラムのダウンロード .....	8
1.3 Cコンパイラ .....	9
1.3.1 変数型のサイズ .....	9
1.3.2 整数型の範囲定数 .....	9
1.3.3 浮動小数点型 .....	9
1.3.4 メモリアリメント .....	9
1.3.5 リストと割り込みの扱い .....	10
1.3.6 基本予約語 .....	10
1.3.7 拡張予約語 .....	10
1.3.8 ビットフィールド .....	11
<b>2 CIPHERLAB シリーズ ハンデーターミナル専用タイプリ .....</b>	<b>13</b>
2.1 システム .....	14
2.1.1 システム関数 .....	14
2.1.2 POR (Power On Reset) .....	17
2.1.3 システムグローバル変数 .....	18
2.1.4 システム情報 .....	20
2.1.5 セキュリティ関数 .....	25
2.1.6 プログラムマネージャ .....	26
2.1.7 ダウンロードモード .....	33
2.1.8 メニューデザイン .....	34
2.2 バイコードリーダー .....	37
2.2.1 バイコードの読み取り .....	37
2.2.2 CodeType 変数 .....	39
2.2.3 スキャン設定テーブル変数 .....	41
2.3 RFIDリーダー .....	42
2.3.1 バイチャル COM .....	43
2.3.2 RFIDパラメータ構造体 .....	43
2.3.3 RFIDデータフォーマット .....	43
2.3.4 RFID認証 .....	45
2.4 キーボードウェッジインターフェイス .....	46
2.4.1 WedgeSetting 変数定義 .....	47
2.4.2 キーボードウェッジインターフェイスキャラクタ表 .....	48
2.4.3 ウェッジエミュレータ .....	49
2.5 ブザー .....	50
2.5.1 ブザーシーケンス .....	50
2.5.2 ブザー周波数 .....	50
2.5.3 ブザー間隔 .....	50
2.6 LED .....	53
2.7 バイブレータとヒータ .....	54
2.7.1 バイブレータ .....	54
2.7.2 ヒータ .....	55
2.8 リアルタイムクロック .....	56

2.8.1	加算機 .....	56
2.8.2	アラーム .....	57
<b>2.9</b>	<b>パワーマネジメント .....</b>	<b>58</b>
2.9.1	バッテリー電圧 .....	58
2.9.2	充電 .....	59
<b>2.10</b>	<b>キーボード .....</b>	<b>60</b>
2.10.1	一般 .....	60
2.10.2	ALPHA キー .....	64
2.10.3	SHIFT キー .....	66
2.10.4	ALT キー .....	67
2.10.5	ファンクションキー .....	68
2.10.6	ENTER キー .....	70
<b>2.11</b>	<b>LCD .....</b>	<b>71</b>
2.11.1	プロパティ .....	71
2.11.2	カーソル .....	76
2.11.3	表示 .....	78
2.11.4	画面消去 .....	81
2.11.5	イメージ .....	83
2.11.6	グラフィック .....	85
<b>2.12</b>	<b>タッチスクリーン .....</b>	<b>88</b>
2.12.1	ItemProperty 構造体 .....	88
2.12.2	使用例 .....	90
<b>2.13</b>	<b>フォント .....</b>	<b>91</b>
2.13.1	フォントサイズ .....	91
2.13.2	表示性能 .....	91
2.13.3	マルチ言語フォント .....	91
2.13.4	特殊フォント .....	91
2.13.5	フォントファイル .....	94
<b>2.14</b>	<b>メモリ .....</b>	<b>95</b>
2.14.1	フラッシュメモリ .....	95
2.14.2	SRAM .....	97
2.14.3	SD カード .....	98
<b>2.15</b>	<b>ファイル操作 .....</b>	<b>99</b>
2.15.1	ファイルシステム .....	99
2.15.2	ディレクトリ .....	99
2.15.3	ファイル名 .....	99
2.15.4	ファイルハンドル .....	99
2.15.5	エラーコード .....	99
2.15.6	DAT ファイル .....	102
2.15.7	DBF ファイルと IDX ファイル .....	110
2.15.8	SD 経由のファイル転送 .....	120
<b>2.16</b>	<b>SD カード .....</b>	<b>124</b>
2.16.1	ファイルシステム .....	124
2.16.2	ディレクトリ .....	125
2.16.3	ファイル名 .....	126
2.16.4	FILEINFO 構造体 .....	126
2.16.5	SD カード 操作 .....	127
2.16.6	マストレージ デバイス .....	141
2.16.7	エラーコード .....	142
<b>3</b>	<b>標準ライブラリ関数 .....</b>	<b>144</b>
<b>4</b>	<b>リアルタイムカーネル .....</b>	<b>147</b>
<b>付録 1 SCANNERDESTBL 変数 .....</b>		<b>151</b>
バーコードパラメータ表 1 .....		151

バーコードパラメータ表 2 .....	155
<b>付録 2 バーコードパラメータ .....</b>	<b>160</b>
CCD、レーザ-の読み取り .....	160
CODABAR .....	160
CODE 2 of 5 .....	160
CODE 39 .....	162
CODE 93 .....	162
CODE 128 / EAN 128 / ISBT 128 .....	162
ITALIAN / FRENCH PHARMACODE .....	163
MSI .....	163
補助バーコード .....	163
PLESSEY .....	164
RSS .....	164
TELEPEN .....	164
UPC / EAN .....	165
2次元、(エクストラ)ロングレンジレーザ-の読み取り .....	167
CODABAR .....	167
CODE 2 of 5 .....	167
CODE 39 .....	168
CODE 93 .....	169
CODE 128 .....	169
MSI .....	169
RSS .....	170
UPC / EAN .....	170
UCC COUPON CODE .....	171
共通設定 .....	171
CODE 11 .....	172
2次元読み取り限定 .....	173
1次元バーコード .....	173
マトリックスコード .....	174
2次元バーコード .....	175
<b>付録 3 読み取りパラメータ .....</b>	<b>177</b>
読み取りモード .....	177
比較表 .....	177
読み取り照合 .....	178
タイムアウト .....	178
レーザ-設定 .....	179

Blank page

## はじめに

このプログラミングガイドは、ユーザーがCコンパイラを使用して CipherLab 8 シリーズ・ハンデーターミナル用アプリケーションプログラムを書くためのガイドです。以下の章で構成されています。

### Part1 基本とハードウェア制御

- 第1章：開発環境。Cコンパイラでハンデーターミナル用アプリケーションを開発するための導入ガイドです。
- 第2章：ハンデーターミナル関数。ハンデーターミナル用ライブラリ関数の使用方法です。データ通信用ライブラリについては Part2 を参照してください。
- 第3章：標準関数。ANSI 標準関数に関する簡単な説明です。
- 第4章：リアルタイムカーネル。リアルタイムカーネル、uC/OS に関する記述です。ユーザーは、uC/OS 機能を使用したリアルタイム・マルチタスク・システムを構築することができます。
- 第5章：シミュレータ。シミュレータの機能と、アプリケーションプログラムを開発する際の使用方法に関する記述です。

### Part2 データ通信

- 第1章：通信ポート。
- 第2章：TCP/IP 通信。
- 第3章：ワイヤレスネットワーク。
- 第4章：IEEE 802.11b/g。
- 第5章：Bluetooth。
- 第6章：GSM/GPRS。
- 第7章：音響モデム。
- 第8章：モデム、イーサネット& GPRS 接続。
- 第9章：USB 接続。
- 第10章：GPS 機能。
- 第11章：FTP 機能。

# 1 開発環境

C コンパイラの開発環境は、大きく bin, etc, include, lib, readme, user の 6 つのディレクトリで構成されます。

お使いのパソコンに C コンパイラの開発環境をインストールする場合は、ルートディレクトリにディレクトリ名 C\_Compiler を作成し、そこへ上記の 6 つのディレクトリを CD-ROM から全てコピーします。

C コンパイラが動作する OS は次の通りです。

- Windows 2000
- Windows XP
- Windows Vista
- Windows 7



## 1.1 フォルダ構成と変数

### 1.1.1 フォルダ構成

各フォルダの用途と内容は次の通りです。

#### bin

実行ファイル(exe)のフォルダです。各実行ファイルの使用方法などの詳細は、後述します。

- Windows 2000、Windows XP では“BIN”フォルダの”の実行ファイルを使用します。
- Windows Vista、Windows 7 では“BIN for Vista-7”フォルダの実行ファイルを使用します。
- 実行ファイル一覧

ASM900.EXE	CC900.EXE	EZDRIVER.DLL	MAC900.EXE
THC1.EXE	THC2.EXE	TUARP.EXE	TUCONV.EXE
TUFAL.EXE	TULIB.EXE	TULINK.EXE	TUMPL.EXE

※ ご使用になる OS に応じて、正しいリンクファイルを使用してください。

#### etc

ヘルプやコンパイルバージョン情報などのフォルダです。

#### include

ヘッダファイル(h)のフォルダです。

- 各ハードウェアミドルウェア対応ライブラリ用ヘッダファイル。8200lib.h、8500lib.h ...など。
- リアルタイムカーネルライブラリ用ヘッダファイル。UCOS.H。
- 標準ライブラリ用ヘッダファイル。

CTYPE.H	ERRNO.H	FLOAT.H	LIMITS.H	MATH.H
STDARG.H	STDDEF.H	STDIO.H	STDLIB.H	STRING.H
TCPIP.H				

#### LIB

ライブラリファイル(lib)のフォルダです。

- C 標準ライブラリ。c900ml.lib。
- 各ハードウェアミドルウェア対応ライブラリ

8000lib.lib	8200lib.lib	8300lib.lib	8400lib.lib	8500lib.lib
8700lib.lib				

#### Readme

C コンパイラに関するバージョン履歴情報や補足情報ファイルのフォルダです。

#### Download Utilities

ハードウェアミドルウェアアプリケーションやフォントファイルをダウンロードするユーティリティのフォルダです。

#### Font

フォントファイルのフォルダです。

#### Kernel

カーネルのフォルダです。

#### Link File

リンクファイルのフォルダです。

#### Manual

マニュアルのフォルダです。

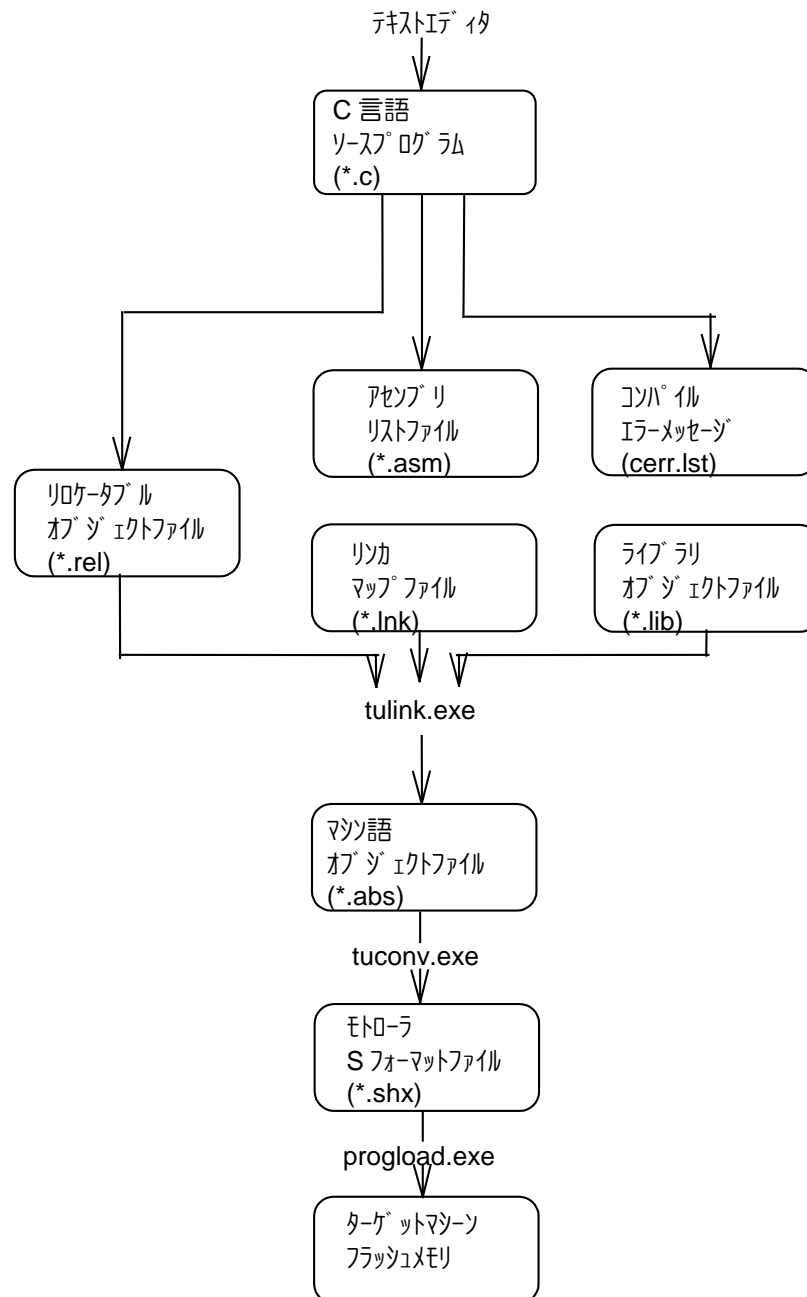
## 1.1.2 環境設定

C 32bit を正しく動作させるためには、幾つかの環境変数の設定が必要になります。下記の例に従って、環境変数の設定を autoexec.bat ファイルに追加してください。

- 1) path = (既に設定されているパス);c:\¥c\_compiler¥bin
  - 2) set thome900=c:\¥c\_compiler
  - 3) set tmp = C: ¥tmp
- \* 既に temp が定義されている場合は、不要です。

## 1.2 プログラム開発フロー

下記にプログラム開発フローを示します。



### 1.2.1 Cソースプログラム

まず、テキストエディタでCソースプログラムをコーディングし、拡張子を「.c」として、userディレクトリに保存します。userディレクトリが作業ディレクトリとなります。また、プログラムを作成する場合は、できる限り関連する関数をモジュール化し、複数ファイルに分割することをお奨めします。これは、コンパイル時間の短縮にもつながり、作業効率がアップします。

## 1.2.2 コンパイル

C プログラムのコンパイルは、cc900.exe プログラムで行います。

### cc900 -[options] ファイル名.c

cc900.exe プログラム及びそのオプションに関する詳細は、etc ディレクトリにある cc900.hlp を参照ください。  
簡単にコンパイルを行うためのバッチファイル y.bat が user ディレクトリに用意されています。下記の書式で、コマンドラインから実行すればコンパイルが起動します。

### y ファイル名.c

このバッチファイルを実行すると、bin ディレクトリ下に置かれた複数の実行プログラムが順に起動され、コンパイルが行われます。複数の実行ファイルは、コンパイルによって順番に起動されるため、ユーザーはその実行方法などを取得する必要はありません。また、コンパイルを起動する場合、引数としてオプション指定が必須となります。弊社が提供するバッチファイル y.bat を使用せずに、独自のバッチファイルを作成する場合は、下記を参照し、正しくオプションを指定するようにしてください。

- -XA1, -XC1, -XD1, -Xp1 : メモリアライメントの設定を全て 1 にします。
- -XF : アセンブリファイルを削除しません。アセンブリファイルの解析が必要ない場合は、このオプションは省略可能です。
- -O3 : 最適化レベルを 0(無し)~3(最大)の範囲で指定します。コードサイズやパフォーマンスが問題で無い場合は、このオプションは省略可能です。省略した場合は、デフォルトの -O0(最適化無し)が適用されます。

最適化を有効にすると、幾つか命令が最適化の過程で削除される場合があるので、注意が必要です。例えば、

```
test()
{
    unsigned int old_msec;
    old_msec=sys_msec;
    while (old_msec == sys_msec) ;
}
```

上記は、sys\_msec 値に変化があるまで待つ関数です。ここで使用される sys\_msec は、5 ミリ秒毎に更新されるシステムグローバル変数ですが、最適化が有効となっている場合、無意味な無限ループと判断され、関数全体が切り詰められてしまいます。これを避けたい場合は、タイプ修飾子 volatile を使うことで最適化を防ぐことが可能です。

- -c : オブジェクトファイルを生成します。(リリンクは無し)
- -e cerr.lst : エラーリストファイル cerr.lst を生成します。

コンパイルが完了すると、実行オブジェクトファイルを生成する際にリリンクによって使用される「プログラム名.rel」というリリンクファイルが生成されます。また、同時にデバッグにも利用可能な「プログラム名.asm」というアセンブリファイルも生成します。コンパイルで何らかのエラーが発生した場合は、エラーリストファイル「cerr.lst」にその詳細が書かれます。

### 1.2.3 リンク

コマンドによって、リンクターゲットファイルが正しく生成されれば、次にリンクを実行してマシン語オブジェクトファイルを生成します。リンクを実行する場合、プログラムに応じたリンクマップファイルを作成する必要があります。

#### tulink ファイル名.lnk

リンクマップファイル「プログラム名.lnk」は、ターゲットマシンの環境に従って、コード、データ、定数などの絶対アドレスの割り当てをリンクへ指示するためのファイルです。通常は、ハードウェア知識などを熟知した上で作成する非常にめんどろな作業ですが、下記の例のように各 CipherLab シリーズ ハードウェア用のリンクマップファイルサンプルが用意されていますので、それを元にファイル名の記述部分(下線付太字部分)を変更するだけでターゲットプログラムに合ったリンクマップファイルを作成することができます。リンクが完了すると、マシン語オブジェクトファイル「file1.abs」が生成され、同時に全コード、変数アドレス、エラーメッセージをリスト化したマップファイル「file1.map」も生成されます。

```
-lm -lg -ll          /* For Windows 2000, XP: parameters for TULINK, don't change */
                    /* For Windows Vista, Windows7: remove "-lg" */

file1.rel           /* your C program name */
file2.rel           /* your C program name */
....
....
filen.rel          /* your C program name */

..\lib\c900ml.lib    /* standard library */
..\lib\8xxxlib.lib   /* 8xxx Function library */

/*****
/*  User could provide suitable values to  */
/*  the following variables                */
*****/
MainStackSize = 0x001000;
HeapSize      = 0x000100;
MaxSysRamSize = 0x020000;

/*****
/*  Do not modify anything beyond this line */
*****/
memory
{
    IRAM: org = 0x001100, len = 0x000e00    /* 0x1000 - 0x10ff IntVec */
                                           /* 0x1f00 - 0x1fff Stack */

    RAM   : org = 0x205000, len = 0x3b000
    ROM   : org = 0xf00000, len = 0x0e0000
}

sections
{
    code org = 0xf00000 : {
        *(f_head)
        *(f_code)
    } > ROM

    area org = 0x205000 : {
        . += MainStackSize;
        . += HeapSize;
        *(f_bcr)
        *(f_area)
    } > RAM

    data org=org(code)+sizeof(code)  addr=org(area)+sizeof(area) : {
        *(f_data)
    } /* global variables with initial values */

    xcode org = org(data) + sizeof(data)  addr = addr(data) + sizeof(data) : {
        *(f_xcode) /* code reside on RAM */
    }

    RAM_OVERFLOW_CHECK org = org(area) + MaxSysRamSize : {
        . += 1;
    } > RAM

    icode org = org(xcode) + sizeof(xcode)  addr = 0x001100 : {
        *(f_icode) /* code reside on IRAM */
    }
}
```

```

    const org = org(xcode) + sizeof(xcode) : {
        *(f_const)
        *(f_tail)
    } > ROM
}

ActualRamSize = (addr(xcode) + sizeof(xcode)+3)/4*4 - 0x205000 ;
                                                    /* long boundary */
SysRamEnd      = org(area) + MaxSysRamSize;      /* long boundary */
DataRam        = addr(data);
XCodeRam       = addr(xcode);
ICodeRam       = addr(icode);
HeapTop        = org(area) + MainStackSize;

/* End */

```

## 1.2.4 フォーマット変換

リカが生成したマシ語オブジェクトファイルは東芝フォーマットになっているため、CipherLabシリーズハンディターミナルにダウンロードする前に `tuconv.exe` ユティリティプログラムでモトローSフォーマット(shx)に変換する必要があります。

**`tuconv -Fs32 -o ファイル名.shx ファイル名.abs`**

変換後のファイルの拡張子は必ず `shx` でなければいけません。

`user` ディレクトリにあるバッチファイル `z.bat` を利用すれば簡単にフォーマット変換を実行することが可能です。単純に、コマンドラインで下記のようにタイプしてください。(z.bat ファイル中のファイル名を編集する必要があります。)

**z**

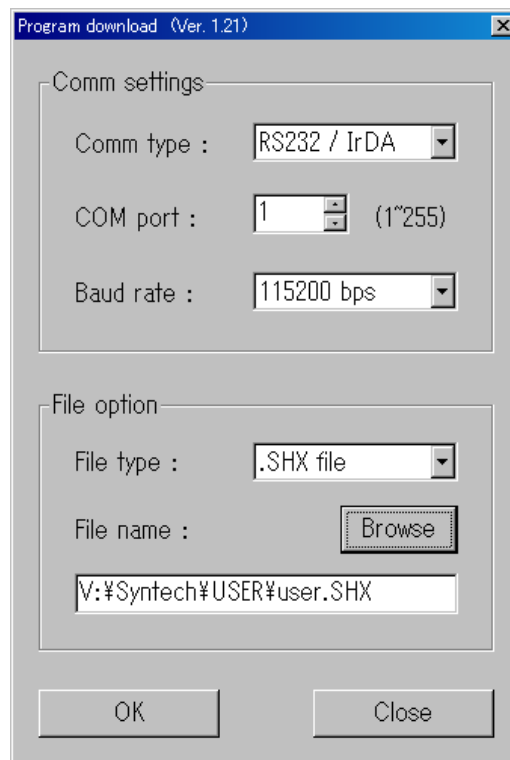
エラーが発生しなければ、shx ファイルが生成されます。

## 1.2.5 プログラムのダウンロード

モトローSフォーマットファイル(SHX)が正しく生成できれば、いよいよターゲットとなる CipherLab シリーズハンディターミナルのフラッシュメモリへプログラムをダウンロードして、プログラムが正しく動作するかを確認します。progload.exe ダウンロード ユティリティプログラムを実行して、下記のパラメータをターゲットマシンの設定に合わせ、プログラム(shx)をダウンロードします。

- File Name : 「Browse」ボタンをクリックして、ダウンロードするプログラムファイルを選択
- COM port : 通信に使用する COM ポートの番号(1~255)
- Baud rate : 9600, 19200, 38400, 57600, 115200bps
- Parity : なし
- Data Bits : 8
- Flow Control : なし

※ ポーレート、パリティ、データビットなどは、ハンディターミナルの COM ポート設定にあわせてください。



## 1.3 C コパイル

本製品は、東芝 TLCS-900 ファミリー 16ビット MCU 専用の ANSI 互換 C コパイルです。但し、一部特有の特徴がありますので、下記を参照ください。

### 1.3.1 変数型のサイズ

変数型	サイズ (バイト)
char, unsigned char	1
short int, unsigned short int, int, unsigned int	2
long int, unsigned long int,	4
pointer	4
structure, union	4

### 1.3.2 整数型の範囲定数

ヘッダファイル LIMITS.H には、整数型の範囲を表す、下記のマクロが定義されています。

マクロ名	説明
CHAR_BIT	1 バイトのビット数
SCHAR_MIN	signed char 型の最小値
SCHAR_MAX	signed char 型の最大値
CHAR_MIN	char 型の最小値
CHAR_MAX	char 型の最大値
UCHAR_MAX	unsigned char 型の最大値
MB_LEN_MAX	wide character 定数のバイト数
SHRT_MIN	short int 型の最小値
SHRT_MAX	short int 型の最大値
USHRT_MAX	unsigned short int 型の最大値
INT_MIN	int 型の最小値
INT_MAX	int 型の最大値
UINT_MAX	unsigned int 型の最大値
LONG_MIN	long int 型の最小値
LONG_MAX	long int 型の最大値
ULONG_MAX	unsigned long int 型の最大値

### 1.3.3 浮動小数点型

IEEE 規格に適合した下記の浮動小数点型をサポートしています。

変数型	サイズ (ビット)
float	32
double	64
long double	80

### 1.3.4 メモリアライメント

メモリ効率と CPU パフォーマンスを最適化するために、各変数型のメモリアライメントをコパイルオプションで調整することができますが、本コパイルのターゲットシステムは、8ビットデータバスを活用しているため、このオプションの効果はありません。従って、メモリアライメントに関するコパイルオプション値は、全て 1 (-XA1, -XD1, -XC1, -Xp1) を指定します。



### 1.3.5 リースと割り込みの扱い

CipherLab リース ハードウェアミクロシステムズへのアクセスは、弊社が提供している専用ライブラリを通して行わなければならないため、C 言語からリースや割り込みを直接扱うことは禁止しています。

### 1.3.6 基本予約語

一般的な C 言語における基本予約語を下記に列挙します。

auto	double	int	struct	break
else	long	switch	case	enum
register	typedef	char	extern	return
union	const	float	short	unsigned
continue	for	signed	void	default
goto	sizeof	volatile	do	if
static	while			

### 1.3.7 拡張予約語

本 C コパイにおける拡張予約語を下記に列挙します。全ての拡張予約語は、2 つのアダプト " \_ " で始まります。

__adcel	__cdcel	__near	__far
__tiny	__asm	__io	
__XWA	__XBC	__XDE	__XHL
__XIX	__XIY	__XIZ	__XSP
__WA	__BC	__DE	__HL
__IX	__IY	__IZ	__W
__A	__B	__C	__D
__E	__H	__L	__SF
__ZF	__VF	__CF	
__DMAS0	__DMAS1	__DMAS2	__DMAS3
__DMAD0	__DMAD1	__DMAD2	__DMAD3
__DMAC0	__DMAC1	__DMAC2	__DMAC3
__DMAM0	__DMAM1	__DMAM2	__DMAM3
__NSP	__XNSP	__INTNEST	

### 1.3.8 ビットフィールド

ビットフィールド型のベースとして下記の変数型を使用することができます。

変数型	ビット
char, unsigned char	8
short int, int, unsigned short int, unsigned int	16
long int, unsigned long int	32

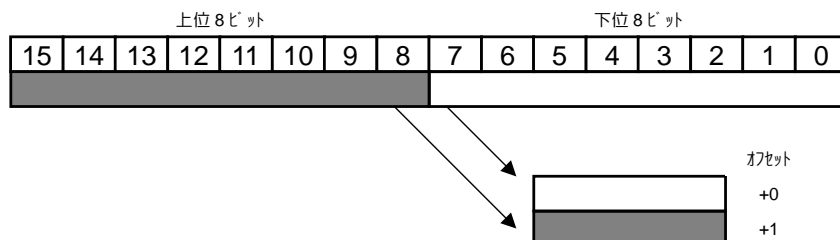
ビットフィールドの割り当ては、下記のように行われます。

1. フィールドは、上位ビット(MSB)から順に格納されます。

```
struct field1 {
    unsigned int a:1;
    unsigned int b:2;
    unsigned int c:3;
    unsigned int d:1;
    unsigned int e:8;
}
```

上位8ビット								下位8ビット							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
a	b		c			d									

2. ビットフィールドのベース変数型が unsigned int 型のように、2 バイト以上を必要とする場合、下記に示すようにトップサイドダウン式にメモリへ格納されます。



3. ビットフィールドを異なるベース変数型で定義した場合、新たな領域が割り当てられます。

例 1 ) char/short の場合

```
struct field {
    unsigned char a:2;
    unsigned short b:3;
}
```

7	6	5	4	3	2	1	0
a							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
b															

例 2 ) signed/unsigned の場合 (変数型は同じ)

```
struct field {
    signed short  a:2;
    unsigned short b:3;
    signed short  c:4;
}
```

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
a		b			c										

例 3 ) short/int の場合

```
struct field {
    signed short  a:5;
    unsigned int  b:4;
}
```

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
a															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
b															

ビットフィールドは、幾つかのケースでは非常に使いやすいものですが、特にメモリ消費に問題が無ければ、コードサイズ及びパフォーマンス低下を招くため、ビットフィールドの使用は避けることをお勧めします。

## 2 CipherLab シリーズ ハンデ ィターミナル専用ライブラリ

CipherLab シリーズ ハンデ ィターミナルのアプリケーション プログラム作成を容易にするため、各マシンで利用できる専用ライブラリを用意しています。このライブラリは、通信、LCD ディスプレイ、ブザー、バーコード スキャナ、ファイルアクセスなど、CipherLab シリーズ ターミナルで利用できる様々なシステムリソースを簡単に制御するための様々な関数を提供しています。本章では、それらの関数をカテゴリ分けし、各節で説明を行っています。各ライブラリの関数プロトタイプ やシステム グローバル変数定義は、各ライブラリに対応するヘッダ ファイル(例 8300LIB.H)を参照ください。尚、本書は、C 言語に関する十分な知識がある方を前提に書かれていますので、予めご了承ください。

## 2.1 システム

### 2.1.1 システム関数

<b>_KeepAlive__</b>	
目的	システムによる自動電源OFF(オートOFF)を無効にします。
書式	void _KeepAlive__(void);
コーディング例	_KeepAlive__();
戻り値	無し
備考	この関数をコールすると、システムグローバル変数 AUTO_OFF が 0 にリセットされます。つまり、手動で電源をOFFにしない限り、プログラムは稼働し続けます。
参照	AUTO_OFF

ChangeSpeed		8000,8300	
目的	CPU スピードを設定します。		
書式	void ChangeSpeed (int speed);		
引数	int speed		
	1	1/16 スピード	4
	2	1/8 スピード	5
	3	1/4 スピード	
コーディング例	ChangeSpeed (4);		
戻り値	無し		
備考	CPU スピードを 1~5(最速)の範囲で設定します。高速処理が必要で無い場合は、CPU スピードを遅く設定し、バッテリー消費を避けるようにします。		

CheckWakeUp		8000,8200,8400,8700
目的	設定されている電源立ち上げ方法を取得します。	
書式	int CheckWakeUp(void);	
コーディング例	if (CheckWakeUp()==TIME_IS_UP) printf("Time is up");	
戻り値	8000 シリーズの場合、下記の何れかの値を返します。	
	0	設定なし
	1	POWER_KEY_PRESSED 電源キーが押された時
	2	CHARGE_OK 充電が完了した時
	3	TIME_IS_UP セットされた時刻になった時(アラーム機能)
	8400 シリーズの場合、下記の何れかの値を返します。	
	0	設定なし
	2	RS232_CABLE_DETECTED RS232 ケーブルを検出した時
	4	CHARGING 充電が開始された時
	8	CHARGE_OK 充電が完了した時
	16	POWER_KEY_PRESSED 電源キーが押された時
	32	TIME_IS_UP セットされた時刻になった時(アラーム機能)
	64	USB_DETECTED USB ケーブルを検出した時
	128	RS232_DATA_RXED RS232 ケーブルでデータを受信した時

8200/8700 シリーズ の場合、下記の何れかの値を返します。

0		設定なし
2	RS232_CABLE_DETECTED	RS232 ケーブルを検出した時
4	CHARGING	充電が開始された時
8	CHARGE_OK	充電が完了した時
16	POWER_KEY_PRESSED	電源キーが押された時
32	TIME_IS_UP	セットされた時刻になった時(アラーム機能)
64	USB_DETECTED	USB ケーブルを検出した時

<b>GetIOPinStatus</b>	<b>8200,8400,8700</b>
-----------------------	-----------------------

**目的** I/O ピンの状態を取得します。

**書式** unsigned int GetIOPinStatus(void);

**コーディング例**

```
iStatus = GetIOPinStatus();
if (iStatus & 0x10)
    printf("RS232 cradle is connected.");
else if (iStatus & 0x20)
    printf("USB cradle is connected.");
if (iStatus & 0x40)
    printf("Adapter is connected.");
```

**戻り値** 各状態を OR した値

**備考** 各状態とビット値は次の通りです。

BIT	値		備考
0	0x00	NO_CRADLE	クレードルに設置されていない
3	0x01	MODEM_CRADLE	モデムクレードルに設置されている
	0x02	ETHERNET_CRADLE	イーサネットクレードルに設置されている
	0x03	GPS_CRADLE	GPS クレードルに設置されている
	0x04	CHARGER_CRADLE	充電アダプタクレードルに設置されている
4	0x00	RS232_CABLE_DISCONNECTED	RS232 ケーブルに接続されていない
	0x10	RS232_CABLE_CONNECTED	RS232 ケーブルに接続されている
5	0x00	USB_CABLE_DISCONNECTED	USB ケーブルに接続されていない
	0x20	USB_CABLE_CONNECTED	USB ケーブルに接続されている
6	0x00	ADAPTER_DISCONNECTED	SV DC アダプタに接続されていない
	0x40	ADAPTER_CONNECTED	SV DC アダプタに接続されている

<b>SetPwrKey</b>
------------------

**目的** 電源キーを有効、または無効にします。

**書式** void SetPwrKey (int mode);

**引数** int mode

0	POWER_KEY_DISABLE	電源キーを無効に設定する
1	POWER_KEY_ENABLE	電源キーを有効に設定する

**コーディング例** SetPwrKey (POWER\_KEY\_DISABLE);

**戻り値** 無し

**備考** 電源キーを有効、または無効にします。但し、電源キーを無効にした場合、電源キーによる電源の切替ができなくなるため、十分注意をして、この関数を使用するようにしてください。

<b>shut_down</b>	
------------------	--

目的	システムをシャットダウンします。
書式	void shut_down(void);
コーディング例	shut_down();
戻り値	無し
備考	この関数をコールすると、システムがシャットダウンされます。シャットダウン後、電源を切にすると、プログラムは常にリスタートします。
参照	system_restart

<b>SysSuspend</b>	
-------------------	--

目的	システムをサスペンド (シャットダウン) します。
書式	void SysSuspend(void);
コーディング例	SysSuspend();
戻り値	無し
備考	この関数をコールすると、システムがサスペンド (シャットダウン) されます。サスペンド (シャットダウン) 後、電源を切にすると、システムの設定に応じて、プログラムはレジューム又はリスタートします。

<b>system_restart</b>	
-----------------------	--

目的	システムをリスタートします。
書式	void system_restart(void);
コーディング例	system_restart();
戻り値	無し
備考	PORルーチンへジャンプし、システムをリスタートします。
参照	shut_down

## 2.1.2 POR (Power On Reset)

リセット後、下記に列挙するハードウェア、メモリマップ、パラメータなどの初期化を行う POR ルーチンが実行されます。C プログラムでは、プログラムの開始ポイントとなる main 関数が必ず 1 つ存在しなければいけません。上記の POR ルーチンによる初期化処理が終了した時点で、main 関数へ制御が移ります。

### COM ポート

全て無効となります。

### リグポート

全て無効となります。

### キーボードスキャンング

有効です。

### LCD ディスプレイ

LCD ディスプレイは初期化/クリアされ、カーソル位置は画面左上(0,0)となります。

➤ コントラスト : Level 4

### バックライト

以下の値で初期化されます。

- 継続時間            20 秒
- 光度                Level 2 (= BKLIT\_LO)
- 陰影効果            有効 (8200/8400 シリーズ では BKLIT\_SHADE\_LO)

### LED

デフォルト値 (8200/8400 シリーズ では LDE\_SYSTEM\_CTRL) にリセットされます。

### 加算器

初期化されます。

### ブザー音量

デフォルト値 (= HIGH\_VOL) にリセットされます。

### USB 充電電流(8200/8400 シリーズ のみ)

500mA にリセットされます。

### その他

スタックエリア及びパラメータが初期化されます。



### 2.1.3 システムグローバル変数

下記に列挙する幾つかのシステムグローバル変数が定義されています。

システムタイマ変数 sys\_msec と sys\_sec は、電源が時に 0 にリセットされ、タイマ割り込みにより自動的に更新されます。  
システムタイマ変数に値を書き込んではいけません。

<b>extern volatile unsigned long sys_msec;</b>	<b>/* 5ミリ秒 単位 */</b>
<b>extern volatile unsigned long sys_sec;</b>	<b>/* 1秒 単位 */</b>

<b>extern unsigned int</b>	<b>AUTO_OFF;</b>	<b>/* 1秒 単位 */</b>
----------------------------	------------------	--------------------

自動的に電源が切れる迄の時間を秒単位で設定します。ここで設定された秒数、何の操作もされなかった場合、自動的に電源が切れます。この機能を無効にしたい場合は、0 を設定します。

<b>extern unsigned int</b>	<b>POWER_ON;</b>
----------------------------	------------------

電源が切れた時の動作を POWERON\_RESUME(レジューム) 又は POWERON\_RESTART(リスタート)の何れかに設定します。デフォルトは、POWERON\_RESUME(レジューム)です。但し、POWERON\_RESUME(レジューム)に設定した場合でも、メインバッテリー交換を行った場合やシステムメニューを起動した場合は、プログラムはリスタートされます。

<b>extern const int</b>	<b>SYSTEM_BEEP[];</b>
-------------------------	-----------------------

システムメニューに入った時の BEEP 音の周波数と鳴動時間をそれぞれ設定します。次のようにコーディングすることで使用できます。

on\_beeper(SYSTEM\_BEEP);

<b>extern unsigned int</b>	<b>BKLIT_TIMEOUT;</b>	<b>/* 1秒 単位 */</b>
----------------------------	-----------------------	--------------------

LCD バックライトの継続時間を秒単位で設定します。デフォルトは 20 秒です。

<b>extern long</b>	<b>AIMING_TIMEOUT;</b>	<b>/* 5ミリ秒 単位 */</b>
--------------------	------------------------	----------------------

スキャナが Aiming モードのときの照準時間を 5 ミリ秒単位で設定します。デフォルトは 200(=1 秒)です。0 は設定できません。

<b>extern int</b>	<b>IrDA_Timeout;</b>	<b>8000,8300,8500</b>
-------------------	----------------------	-----------------------

IrDA 接続のタイムアウト値を 1(3 秒)~8(40 秒)の範囲内で設定します。ここで設定された時間内に IrDA デバイスとの接続を確立できなければ、システムは接続リタイアメントを中止します。下記の範囲内で設定が可能です。

1	3 秒(デフォルト)	5	20 秒
2	8 秒	6	25 秒
3	12 秒	7	30 秒
4	16 秒	8	40 秒

<b>extern int</b>	<b>BC_X, BC_Y;</b>
-------------------	--------------------

バッテリーアイコンの表示座標を設定します。

- 8000 シリーズ:      デフォルトは(96,51)
- 8300 シリーズ:      デフォルトは(120,51)
- 8200/8400 シリーズ:      デフォルトは(144,152)
- 8500/8700 シリーズ:      デフォルトは(144,152)

<b>extern int</b>	<b>KEY_CLICK[4];</b>
-------------------	----------------------

この配列変数には、サウト周波数と鳴動時間をそれぞれ設定します。例えば、下記のように on\_beeper 関数をコールすると、キークリック音と同じサウト音を鳴動することができます。

on\_beeper(KEY\_CLICK);

**extern unsigned char    WakeUp\_Event\_Mask;**

この配列変数には、サウト 周波数と鳴動時間をヘアで設定します。例えば、下記のように on\_beeper 関数をコールすると、キークリック音と同じサウト 音を鳴動することができます。

8000 シーズ	PwrKey_WakeUp	電源キーが押された時
	Alarm_WakeUp	セットされた時刻になった時
8300 シーズ	Wedge_WakeUp	キーボードウェッジケーブルが接続された時
	RS232_WakeUp	RS232C ケーブルが接続された時
	Charging_WakeUp	充電が開始された時
	ChargeDone_WakeUp	充電が完了した時
8400 シーズ	USB_WakeUp	USB ケーブルが接続された時
	RS232RXD_WakeUp	RS232 ケーブルでデータを受信した時
	RS232_WakeUp	RS232C ケーブルが接続された時
	Charging_WakeUp	充電が開始された時
	ChargeDone_WakeUp	充電が完了した時
	PwrKey_WakeUp	電源キーが押された時
	Alarm_WakeUp	セットされた時刻になった時
8500 シーズ	Charging_WakeUp	充電が開始された時
	ChargeDone_WakeUp	充電が完了した時
8200,8700 シーズ	USB_WakeUp	USB ケーブルが接続された時
	RS232_WakeUp	RS232C ケーブルが接続された時
	Charging_WakeUp	充電が開始された時
	ChargeDone_WakeUp	充電が完了した時
	PwrKey_WakeUp	電源キーが押された時
	Alarm_WakeUp	セットされた時刻になった時

例)

```
WakeUp_Event_Mask = RS232_WakeUp | Charging_WakeUp;
// RS232C ケーブルが接続されるか、充電が開始された時にウェイクアップ
```

**extern char            ProgVersion[16];**

この配列変数には、ユーザープログラムのバージョン情報を設定します。ここで設定したバージョン情報は、システムメニューの「Information」で表示されます。C プログラム内で、下記のように宣言して、システムのデフォルト値を上書きしてください。  
char ProgVersion[16] = "Power AP 1.00";

## 2.1.4 システム情報

ハードウェアの構成についての情報を取得するための関数です。

### DeviceType

**目的** ハードウェアのモジュール構成情報を取得します。

**書式** void \*DeviceType(void);

**コーディング例** prints("DEV : %s - %01d", DeviceType(), KeypadLayout());

**戻り値** 該当情報のポインタ

**備考** デバイスタイプの情報は、「xxxx」のフォーマットで表示されます。各要素は 0 から 9 まで数字で表されます。

8000 シリーズ	0xxx	リーダ なし
	1xxx	CCD スキャナ
	2xxx	レーザー スキャナ
	x0xx	ワイヤレス モジュール なし
	x4xx	802.11b/g
	x5xx	Bluetooth
	x6xx	音響 カプラ
	xx0x	単 4 アルカリ 電池
	xx1x	リチウムイオン 電池
8200 シリーズ	0xxx	リーダ なし
	1xxx	CCD スキャナ
	2xxx	レーザー スキャナ
	3xxx	2 次元 スキャナ
	x0xx	ワイヤレス モジュール なし
	x5xx	Bluetooth
	x8xx	802.11b/g + Bluetooth
	0xxx	リーダ なし
8300 シリーズ	1xxx	CCD スキャナ(H/W バージョン 4.0 以外)
	2xxx	レーザー スキャナ CCD スキャナまたはレーザー スキャナ(H/W バージョン 4.0)
	4xxx	ロングレンジ レーザ スキャナ
	x0xx	ワイヤレス モジュール なし
	x1xx	433MHz 無線
	x2xx	2.4GHz 無線
	x4xx	802.11b/g
	x5xx	Bluetooth
	x6xx	音響 カプラ
	xx0x	RFID 非対応
	xx1x	RFID 対応
	xxx0	なし
	xxx1	CCD スキャナ(H/W バージョン 4.0 のみ)
	H/W バージョンが 4.0 の場合、先頭桁が"2"のとき、スキャナ種別は CCD またはレーザー となります。このとき最終桁が"1"なら CCD、"0"ならレーザー となります。	
8400 シリーズ	0xxx	リーダ なし
	1xxx	CCD スキャナ
	2xxx	レーザー スキャナ
	3xxx	2 次元 スキャナ
	x4xx	802.11b/g + Bluetooth
	x5xx	Bluetooth のみ
8500 シリーズ	0xxx	リーダ なし
	1xxx	CCD スキャナ
	2xxx	レーザー スキャナ
	3xxx	2 次元 スキャナ
	4xxx	ロングレンジ レーザ スキャナ
	5xxx	エクストラロングレンジ レーザ スキャナ
	x4xx	802.11b/g + Bluetooth

8700 シリーズ	x5xx	Bluetooth のみ
	xx0x	RFID 非対応
	xx1x	RFID 対応
	0xxx	リダ なし
	1xxx	CCD スキャナ
	2xxx	レーザースキャナ
	3xxx	2 次元スキャナ
	4xxx	ロングレンジレーザースキャナ
	x3xx	3.5G 通信
	x4xx	802.11b/g + Bluetooth
	x5xx	Bluetooth のみ
	x7xx	802.11b/g + 3.5G 通信 + Bluetooth
	xx0x	RFID 非対応
	xx1x	RFID 対応
	xx2x	GPS

参照 KeypadLayout

<b>BootloaderVersion</b>	<b>8200</b>
--------------------------	-------------

目的 フォントデータのバージョン情報を取得します。

書式 void \*BootloaderVersion(void);

コーディング例 prints("BL : %s", BootloaderVersion());

戻り値 該当情報のポインタ

参照 LibraryVersion

<b>FontVersion</b>
--------------------

目的 フォントファイルのバージョン情報を取得します。

書式 void \*FontVersion(void);

コーディング例 prints("FONT : %s", FontVersion());

戻り値 該当情報のポインタ

参照 CheckFont

<b>GetRFmode</b>
------------------

目的 インストールされている RF モジュールを取得します。

書式 int GetRFmode(void);

コーディング例 GetRFmode();

戻り値 インストールされている RF モジュールに応じた値(0~8)を返します。

備考 各状態と戻り値は次の通りです。

0x00	NO_RF_MODEL	(8000,8200,8300)
0x01	MODE_433M	廃止
0x02	MODE_24G	廃止
0x03	MODE_GSMGPRS	(8780)
0x04	MODE_802DOT11	(8071,8370,8470,8570,8770)
0x05	MODE_BLUETOOTH	(8062,8260,8362,8400,8500,8700)
0x06	MODE_ACOUSTIC	(8020,8021)
0x07	MODE_802DOT11_GSM	(8790)
0x08	MODE_802DOT11_BT	(8230,8330)

## HardwareVersion

目的	ハードウェアのバージョン情報を取得します。
書式	void *HardwareVersion(void);
コーディング例	prints("H/W : %s", HardwareVersion());
戻り値	該当情報のポインタ

## KernelVersion

目的	カーネルのバージョン情報を取得します。
書式	void *KernelVersion(void);
コーディング例	prints("KNL : %s", KernelVersion());
戻り値	該当情報のポインタ

## KeypadLayout

目的	キーパッドのレイアウト情報を取得します。
書式	int KeypadLayout(void);
コーディング例	prints("DEV : %s - %01d", DeviceType(), KeypadLayout());
戻り値	キーパッドのレイアウト情報

8000	0 (21-key)
8200	0 (24-key)
8300	0 (24-key) , 1 (39-key)
8400	0 (29-key) , 1 (39-key)
8500	0 (24-key) , 1 (44-key Type I) , 2 (44-key Type II [44-TE-key])
8700	0 (24-key) , 1 (44-key Type II [44-TE-key])

## LibraryVersion

目的	ライブラリのバージョン情報を取得します。
書式	void *LibraryVersion(void);
コーディング例	prints("LIB : %s", LibraryVersion());
戻り値	該当情報のポインタ

8000	標準関数(8000lib.lib)ライブラリのバージョン
8200	標準関数(8200lib.lib)ライブラリのバージョン
8300	標準関数(8300lib.lib)ライブラリのバージョン
8400	標準関数(8400lib.lib)ライブラリのバージョン
8500	標準関数(8500lib.lib)ライブラリのバージョン
8700	標準関数(8700lib.lib)ライブラリのバージョン

参照	BootloaderVersion, NetVersion
----	-------------------------------

## ManufactureDate

目的	製造年月日を取得します。
書式	void * ManufacturingDate(void);
コーディング例	prints("MDATE : %s", ManufacturingDate());
戻り値	該当情報のポインタ

NetVersion		
目的	外部ライブラリのバージョン情報を取得します。	
書式	void *NetVersion(void);	
コーディング例	prints("NetLIB : %s", NetVersion());	
戻り値	該当情報のポインタ	
備考	外部ライブラリがある場合はそのライブラリの、なければ専用ライブラリのバージョン情報を返します。	
	外部ライブラリ	専用ライブラリ
	8000 80PPP.lib, 80NEP.lib, 80WLAN.lib	8000lib.lib
	8200	8200lib.lib
	8300 83PPP.lib, 83NEP.lib, 83WLAN.lib	8300lib.lib
	8400 84PPP.lib, 84WLAN.lib	8400lib.lib
	8500	8500lib.lib
	8700 87PPP.lib, 87WLAN.lib	8700lib.lib
参照	DeviceType, LibraryVersion, PPPVersion	

OriginalSerialNumber	
目的	オリジナルシリアル番号を取得します。
書式	void * OriginalSerialNumber(void);
コーディング例	prints("S/N : %s", OriginalSerialNumber());
戻り値	該当情報のポインタ
備考	オリジナルシリアル番号が「???」の場合、シリアル番号が変更されていないことを意味します。
参照	SerialNumber

PPPVersion		8000,8300,8400
目的	外部 PPP ライブラリのバージョン情報を取得します。	
書式	void *PPPVersion(void);	
コーディング例	prints("PPPLIB : %s", PPPVersion());	
戻り値	該当情報のポインタ	
備考	外部ライブラリがある場合はそのライブラリのバージョン情報を返し、なければ NONE を返します。	
	外部 PPP ライブラリ	専用ライブラリ
	8000 80PPP.lib, 80NEP.lib, 80WLAN.lib	8000lib.lib
	8300 83PPP.lib, 83NEP.lib, 83WLAN.lib	8300lib.lib
	8400 84PPP.lib, 84WLAN.lib	8400lib.lib
参照	DeviceType, LibraryVersion, NetVersion	

RFIDVersion		8300,8500,8700
目的	RFID のバージョン情報を取得します。	
書式	void *RFIDVersion(void);	
コーディング例	prints("RFID : V%s", RFIDVersion());	
戻り値	該当情報のポインタ	
参照	DeviceType	

SerialNumber	
目的	現在の列列番号を取得します。
書式	void *SerialNumber(void);
コーディング例	prints("S/N : %s", SerialNumber());
戻り値	該当情報のポインタ
参照	OriginalSerialNumber

## 2.1.5 セキュリティ関数

ここでは、システムメニューをパスワード保護する際に使用するセキュリティ関数の説明を行います。尚、パスワードの設定は、システムメニューからも行うことができます。

### CheckPasswordActive

目的	システムパスワードが有効であることをチェックします。
書式	int CheckPasswordActive(void);
コーディング例	<pre>if (CheckPasswordActive())     printf ("Please input password:");</pre>
戻り値	0 = 無効, 1 = 有効
備考	デフォルトでは、パスワードは無効になっています。

### CheckSysPassword

目的	引数で与えられた文字列がパスワードと一致するかをチェックします。
書式	int ChechSysPassword(const char *psw);
コーディング例	<pre>if ( !ChechSysPassword(szInput) )     printf("Password incorrect !!!");</pre>
戻り値	0 = 不一致, 1 = 一致
備考	引数で与えられた文字列がパスワードと一致するかをチェックします。この関数は、プログラムを同じパスワードで保護したい場合などに使用することができます。

### InputPassword

目的	ユーザーにパスワード入力を求めます。
書式	int InputPassword(char *psw);
コーディング例	<pre>char szPsw[10]; printf ("Input password:"); if ( InputPassword (szPsw) )     if ( !ChechSysPassword (szPsw) )         printf ("Illegal password!");</pre>
戻り値	1 = ユーザーが ENTER キーで入力を確定, 0 = ユーザーが ESC キーで入力をキャンセル
備考	ユーザーにパスワード入力を求める簡易入力関数です。入力された文字は、アスタリスク(*)でエコー表示されます。

### SaveSysPassword

目的	システムパスワードを保存(変更)します。
書式	int SaveSysPassword(const char *psw);
コーディング例	SaveSysPassword("12345");
戻り値	1 = 正常終了, 0 = システムパスワードが8文字を超えている
備考	システムパスワードを保存(変更)します。但し、引数で指定するシステムパスワードは8文字以内の文字列でなければいけません。 ➤ Null を指定した場合は、システムパスワードは無効に設定されます。



## 2.1.6 プログラムマネージャ

カーネルプログラムの一部であるプログラムマネージャは、複数のプログラム(.shx)を管理することができます。

### フロッピーディスク(プログラムマネージャ)

LoadProgram()をコールすることにより、プログラムを複数ダウンロードすることができます。

- 8000/8300/8500 シリーズは 6 つまでダウンロードすることができます。
- 8200/8400/8700 シリーズは 7 つまでダウンロードすることができます。

ActivateProgram()をコールすることで、それらのうちの 1 つのみをアクティブな状態にすることができます。アクティブなプログラムは電源を入れることで起動します。

### SRAM(ファイルシステム)

同様に DownLoadProgram()をコールすることで、プログラムをファイルシステムにダウンロードすることができます。ダウンロードできるプログラムの数は、SRAM またはメモリのサイズによります。ただし、上限は 253 となっています。

ProgVersion[]に実行ファイル名が設定されていれば、ダウンロード後にデフォルトのファイル名として表示されます。設定されていない場合は、"UNKOWN"と表示されます。ファイル名は必要に応じて変更できます。

- ファイルシステム中のプログラムは UpdateBank()をコールすることで、プログラムマネージャ(フロッピーディスク)にロードすることができます。プログラム・バージョンと同様、ファイル名もプログラムマネージャに従って記述されます。その後、ActivateProgram()をコールすることでアクティブな状態にすることができます。

また、ファイルシステム中のプログラムは、UpdateUser()を呼び出すことで直接アクティブな状態にすることもできます。

### プログラムマネージャメニュー

- Download

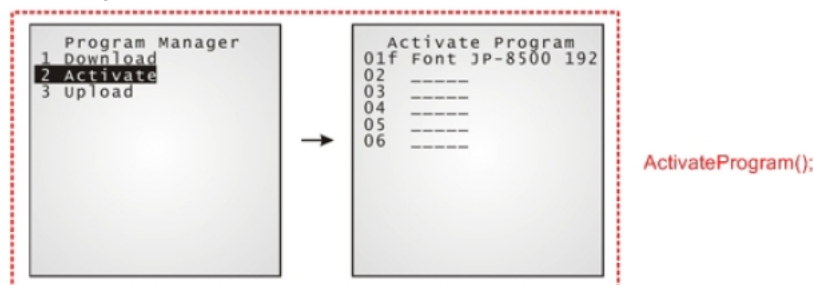
LoadProgram()と同等の処理となります。

ダウンロード方法の選択は、ハードウェアの種類によって異なります。画面は 8500 シリーズのものです。8300 シリーズの場合、Direct RS-232、Cradle-IR、IrDA からの選択となります。8200/8400/8700 シリーズの場合、RS-232、USB Virtual COM、Bluetooth、SD Card からの選択となります。



- Activate

ActivateProgram()と同等の処理となります。



## ➤ Upload

プログラムマネージャメニューでは、ユーザが別のハードウェアまたはホストコンピュータにプログラムをアップロードすることが出来ます。"Upload"を選択した後、1つのプログラムをアップロードするか、すべてのプログラムをアップロードするかを選択します。

ただし、ProgVersion[]に設定されているファイル名の先頭に"#"が付いている場合、プログラムが除外されていることを意味し、アップロードすることはできません。

## ActivateProgram

目的	指定プログラムエリアのプログラムをアクティブプログラムに設定します。	
書式	void ActivateProgram(int Prog, int mode);	
引数	int Prog	
	1～6	最大 6 つまで アクティブに設定するプログラムエリアの番号 (8000/8200/8300/8700 シリーズ)
	1～7	最大 7 つまで アクティブに設定するプログラムエリアの番号(8400 シリーズ)
	int mode	
	0	KEEP_FILE_SYSTEM ファイルシステムの元の状態を保持
	1	CLEAR_FILE_SYSTEM ファイルシステムの元の状態をクリア
コーディング例	<pre>ActivateProgram(3, KEEP_FILE_SYSTEM); // make program #3 become active and keep the file system</pre>	
戻り値	無し	
備考	<p>指定のプログラムをアクティブエリアに設定し、そのプログラムをアクティブプログラムに設定します。</p> <ul style="list-style-type: none"> <li>➤ アクティブエリアに常駐していた前のプログラムは、新しいプログラムで上書きされます。</li> <li>➤ プログラム更新中はシステム保護のため、電源キーが無効となります。</li> <li>➤ 成功すれば直ちに新しいプログラムがアクティブに設定されます。失敗していれば元のプログラムの次の処理が実行されています。</li> </ul>	
参照	DeleteBank, LoadProgram, ProgramManager, ProgramManager	

## DeleteBank

目的	指定プログラムエリアのユーザプログラム(.shx)をプログラムマネージャ(フラッシュメモリ)から削除します。	
書式	int DeleteBank(int slot);	
引数	int slot	
	1～6	最大 6 つまで 削除するプログラムエリアの番号(8000/8200/8300/8700 シリーズ)
	1～7	最大 7 つまで 削除するプログラムエリアの番号(8400 シリーズ)
コーディング例	<pre>if (DeleteBank(1))     Printf("Delete OK"); else     printf("Delete NG");</pre>	
戻り値	0 =失敗, 1 =成功	
参照	ActivateProgram, LoadProgram, UpdateBank	

DownloadProgram					
目的	ファイルシステム(SRAM)にユーザ・プログラム(*.SHX) をダウンロードします。				
書式	int DownloadProgram(char *filename, int comport, int baudrate);				
引数	<div>char *filename</div> <div>プログラムのファイル名が格納された変数のポインタ</div> <div>           &gt; ファイル名は最大 8 文字まで、null 文字を含むことはできません。            &gt; ファイル名が既存のプログラムと同じ場合、処理に失敗します。         </div> <div>int comport</div> <div> <table border="1"> <tr> <td>1 or 2 or 5</td><td>COM ポートの番号 (COM5 は 8200/8400/8700 シリーズのみ対応)</td></tr> </table> </div> <div>int baudrate</div> <div> <table border="1"> <tr> <td>           BAUD_115200            BAUD_76800            BAUD_57600            BAUD_38400            BAUD_19200            BAUD_9600            BAUD_4800            BAUD_2400         </td><td>ポートレート</td></tr> </table> </div>	1 or 2 or 5	COM ポートの番号 (COM5 は 8200/8400/8700 シリーズのみ対応)	BAUD_115200 BAUD_76800 BAUD_57600 BAUD_38400 BAUD_19200 BAUD_9600 BAUD_4800 BAUD_2400	ポートレート
1 or 2 or 5	COM ポートの番号 (COM5 は 8200/8400/8700 シリーズのみ対応)				
BAUD_115200 BAUD_76800 BAUD_57600 BAUD_38400 BAUD_19200 BAUD_9600 BAUD_4800 BAUD_2400	ポートレート				
コーディング例	<pre>val = DownloadProgram(filename_buffer, 1, BAUD_115200); // download user program via COM1 at 115200 bps and return file name to filename_buffer</pre>				
戻り値	0 =失敗, 1 =成功, -1=異常終了				
備考	8300 シリーズでは、この処理をコールする前に Direct RS-232 なら SetCommType(1,0)、Cradle-IR なら SetCommType(1,2)のように、指定するポートの通信タイプを設定する必要があります。 > IrDA によるダウンロードはこの関数ではなく Loadprogram()で行います。				
参照	UpdateBank, UpdateUser				

LoadProgram		
目的	指定プログラムエリアにユーザプログラム (*.SHX) をダウンロードします。	
書式	void LoadProgram(int Prog);	
引数	int Prog	
	1〜6	最大 6 つまで
	7	最大 7 つまで
	プログラムエリアの番号(8000/8200/8300/8700)	
	プログラムエリアの番号(8400)	
コーディング例	LoadProgram(3); /* プログラムエリア 3 にダウンロードします */	
戻り値	無し	
備考	この関数をコールすると、システムは指定のプログラムエリアにユーザプログラムをダウンロードするため、プログラムダウンロードペースにジャンプします。	
参照	ActivateProgram, DeleteBank, ProgramInfo, ProgramManager	

ProgramInfo		
目的	指定のプログラムエリアにあるプログラムの情報をリストアップします。	
書式	int ProgramInfo (int slot, char *programtype, char *programname);	
引数	int slot	
	1～6	最大 6 つまで リストアップ するプログラムエリアの番号(8000/8200/8300/8700)
	1～7	最大 7 つまで リストアップ するプログラムエリアの番号(8400)
	char *programtype	
取得したプログラム種別を格納する変数へのポインタ		
	char *programname	
取得したプログラム名を格納する変数へのポインタ		
コーディング 例	val = ProgramInfo (2, typebuffer, namebuffer);	
戻り値	プログラムが占有しているメモリサイズを K バイト単位で返します。0 は指定プログラムエリアにプログラムがないことを意味します。	
備考	指定のプログラムエリアにあるプログラムのサイズとプログラム名、プログラム種別をチェックします。	
	➤ プログラムが占有しているメモリサイズは K バイト単位で返します。	
	➤ プログラム名はプログラムマネージャで表示されるものと同じものが表示されます。	
	➤ プログラム種別はアルファベット小文字で格納されます。"c"は C プログラム、"b"はベースックプログラム、"f"はフォントファイルを意味します。	
	➤ 1 メモリは、64K バイトのため、戻り値は、64, 128, 192 ... となります。	
参照	ActivateProgram, LoadProgram, ProgramManager	

ProgramManager		
目的	プログラムマネージャメニューを表示します。	
書式	void ProgramManager(void);	
コーディング 例	ProgramManager(); /* プログラムマネージャメニューへジャンプ */	
戻り値	無し	
備考	この関数をコールすると、プログラムの実行は中断され、プログラムマネージャへ処理が移ります。	
参照	ActivateProgram, LoadProgram, ProgramInfo	

UpdateBank		
目的	ファイルシステム(SRAM または SD)にあるユーザプログラム(.shx または .bin)をプログラムマネージャ(フラッシュメモリ)にコピーします。	
書式	int UpdateBank(const char *filename);	
引数	char *filename プログラム名が格納された変数へのポインタ	
コーディング例	<pre>val = UpdateBank("PlayTest");    // update bank via a file in SRAM val = UpdateBank("A:\\PlayTest"); // update bank via a file on SD card</pre>	
戻り値	プログラムの番号(8000/8200/8300/8500/8700 シリーズ は 1~6、8400 シリーズ は 1~7) 失敗した場合、エラー原因の値を返します。	
	-1	ファイルを開くのに失敗した
	-2	ファイルフォーマット不正
	-3	プログラムマネージャのプログラム番号に空きがない
	-4	フラッシュメモリに十分な空き容量がない
	-5	元ファイルからプログラムロードを読めなかった
	-6	フラッシュメモリに消去/書き込みに失敗した
備考	<p>以下の制約があります。</p> <ul style="list-style-type: none"> <li>➢ SRAM にあるファイルのファイル名は 8 バイトまでです。null 文字は使用できません。</li> <li>➢ 指定されたファイル名がフラッシュメモリ中の既存のプログラム名と同じ場合、ファイルは上書きされます。ファイル名が異なる場合は、自動的に割り当てられたプログラムメモリに格納されます。</li> <li>➢ SD カード は 8200/8400/8700 シリーズ のみ対応しています。ファイル名の先頭に"A:\\"のように A ドライブ上を指定すると SD カード 上のファイルとして処理されます。ファイルパスについては『2.16.2 デュアル』を参照してください。"ProgVersion"に設定されているユーザプログラムのバージョン情報がフラッシュメモリ中の既存のプログラムのものと同じ場合、ファイルは上書きされます。SD カード 上のファイル名ではないことに注意してください。</li> </ul>	
参照	DeleteBank, DownloadProgram, UpdateUser	

UpdateBootloader		8200
目的	ファイルシステム(SRAM または SD)にあるブートローダプログラム(.shx または .bin)をブートローダ(フラッシュメモリ)にコピーし、ブートローダを更新します。	
書式	void UpdateBootloader(const char *filename, int mode, int remove);	
引数	char *filename プログラム名が格納された変数へのポインタ	
	int mode	
	0	KEEP_FILE_SYSTEM SRAM ファイルシステムを保持
	1	CLEAR_FILE_SYSTEM SRAM ファイルシステムをクリア
	int remove	
	0	ファイルシステムのプログラムはそのまま
	1	ファイルシステムのプログラムを削除
コーディング例	<pre>val = UpdateBootLoader("8200B100", KEEP_FILE_SYSTEM, 0); // update bootloader via a file in SRAM val = UpdateBootLoader("A:\\8200B100", KEEP_FILE_SYSTEM, 0); // update bootloader via a file on SD card</pre>	
戻り値	更新成功すれば、デバイスがリスタートします。失敗した場合、エラー原因の値を返します。	
	0	ファイルがない
	1	不正なフォーマット、またはファイルの読み込みに失敗
	2	現在のブートローダよりバージョンが古い
備考	<p>ダウングレードはできません。</p> <p>SRAM にあるファイルのファイル名は 8 バイトまでです。null 文字は使用できません。</p> <p>ファイル名の先頭に"A:\\"のように A ドライブ上を指定すると SD カード 上のファイルとして処理されます。</p>	
参照	DownloadProgram, UpdateKernel, UpdateUser	

UpdateKernel		
目的	ファイルシステム(SRAM または SD)にあるカーネルプログラム(.shx または .bin)をカーネル(フラッシュメモリ)にコピーし、カーネルを更新します。	
書式	void UpdateKernel(const char *filename, int mode, int remove);	
引数	char *filename プログラム名が格納された変数へのポインタ	
	int mode	
	0	KEEP_FILE_SYSTEM SRAM ファイルシステムを保持
	1	CLEAR_FILE_SYSTEM SRAM ファイルシステムをクリア
	int remove	
	0	ファイルシステムのプログラムはそのまま
	1	ファイルシステムのプログラムを削除
コーディング例	<pre>val = UpdateBootLoader("8400K100", KEEP_FILE_SYSTEM, 0);     // update kernel via a file in SRAM val = UpdateBootLoader("A:\\8400K100", KEEP_FILE_SYSTEM, 0);     // update kernel via a file on SD card</pre>	
戻り値	更新成功すれば、デバイスはリスタートします。失敗した場合、エラー原因の値を返します。	
	0	ファイルがない
	1	ファイルフォーマット不正
	2	フラッシュメモリに十分な空き容量がない
	3	フラッシュメモリ書き込みに失敗した
	4	フラッシュメモリ読み込みに失敗した
	5	現在のカーネルよりバージョンが古い
備考	以下の制約があります。 ➢ 8200 シリーズを除き、ダウンロードはできません。 ➢ 8200 シリーズを除き、フラッシュメモリに 125 の KB 以上の空きが必要です。 ➢ 8200 シリーズの場合、ファイルが SD カード上にある場合、SRAM ファイルシステムに 1.5MB 以上の空き容量が必要です。 ➢ SRAM にあるファイルのファイル名は 8 バイトまでです。null 文字は使用できません。 ➢ SD カードは 8200/8400/8700 シリーズのみ対応しています。ファイル名の先頭に"A:\\\"のように A ドライブ上を指定すると SD カード上のファイルとして処理されます。	
参照	DownLoadProgram, UpdateBootLoader, UpdateUser	

UpdateUser

目的

ファイルシステム(SRAM または SD)からロードされたユーザープログラム(.shx または .bin)をアクティブにします。

書式

int UpdateUser(const char \*file\_name, int mode...);

引数

char \*filename

プログラム名が格納された変数へのポインタ

int mode

0	KEEP_FILE_SYSTEM	ファイルシステムの元の状態を保持
1	CLEAR_FILE_SYSTEM	ファイルシステムの元の状態をクリア

int remove

0		ファイルシステムのプログラムはそのまま
1		ファイルシステムのプログラムを削除

コーディング例

```
val = UpdateUser("PlayTest", KEEP_FILE_SYSTEM, 0);
// activate then program in SRAM, and keep the file system as well as this program
val = UpdateUser("A:\\PlayTest", KEEP_FILE_SYSTEM, 0);
// activate then program on SD card, and keep the file system as well as this program
```

戻り値

更新成功すれば、デバイスはリスタートします。失敗した場合、エラー原因の値を返します。

0	ファイルがない
1	ファイルフォーマット不正
2	フラッシュメモリに十分な空き容量がない
3	ファイル名の長さが既定以上

備考

UpdateUser(const char \*file\_name, int mode)、UpdateUser(const char \*file\_name, int mode, int remove)のどちらでもコールできます。

ファイルシステムからフラッシュメモリ中のプログラムメモリアreaに指定のプログラムを直接コピーし、プログラムをアクティブにします。元のファイルシステムは引数により元の状態を保持、またはクリアされます。ファイルシステムが保持される場合、プログラムはファイルシステムから削除されます。

➤ SRAMにあるファイルのファイル名は 8 バイトまでです。null 文字は使用できません。

➤ SDにあるファイルのファイル名は 64 バイトまでです。null 文字は使用できません。

➤ アクティブ Area にプログラムが存在する場合、新しいプログラムに上書きされます。

➤ SD カードは 8200/8400/8700 シリーズのみ対応しています。ファイル名の先頭に"A:\\\"のように A ドライブ上を指定すると SD カード上のファイルとして処理されます。

➤ プログラム更新中はシステム保護のため、電源キーが無効となります。

➤ 成功すれば直ちに新しいプログラムがアクティブに設定されます。失敗していれば元のプログラムの次の処理が実行されています。

参照

DownloadProgram, UpdateBank

## 2.1.7 ダウンロードモード

### DownloadPage

**目的** プログラムの実行を中止して、システムメニューのプログラムダウンロードページへジャンプします。

**書式** void DownloadPage(void);  
void DownloadPage(int detect, int comtype, int baudrate);

**コーディング例** open\_com(1, 0x80);  
DownloadPage();

**戻り値** 無し

**備考** ハンディターミナルをダウンロードモードにします。ダウンロード画面が表示され、ユーザーは通信ポートとボーレートを設定します。  
引数を指定してコールすることもできます。

引数 1	detect	NO_MENU 固定
引数 2	comtype	コミュニケーションタイプ、SetCommType 参照
引数 3	baudrate	ボーレート、open_com 参照

(例)

DownloadPage (NO\_MENU, COMM\_DIRECT, BAUD\_115200);

この方法で DownloadPage 関数をコールした場合、システムは「Ready to download」画面へジャンプします。



## 2.1.8 メニューデザイン

SMENU および MENU 構造体は、各モジュールのヘッダファイルで定義されています。ユーザーは MENU 構造体を設定し、prc\_menu をコールするだけで階層メニュー形式のユーザーインターフェイスを使用することができます。

### MENU 構造体

```
struct SMENU {
    int total_entry;
    int selected_entry;
    int ReturnFlag;
    char *title;
    struct SMENU_ENTRY *entry_list[14];
};
typedef struct SMENU MENU;
```

int total_entry	メニュー項目数(1~14)
int selected_entry	選択中メニュー番号(1~total_entry)
int ReturnFlag	次メニューから処理が戻った場合の動作、1 = 続行, 0 = 終了
char *title	メニュータイトル
struct SMENU_ENTRY *entry_list[14]	MENU_ENTRY 構造体参照

### MENU\_ENTRY 構造体

```
struct SMENU_ENTRY {
    int text_x;
    int text_y;
    char *text;
    void (*func)(void);
    struct SMENU *sub_menu;
};
typedef struct SMENU_ENTRY MENU_ENTRY;
```

int text_x	サブメニュー項目の X 座標
int text_y	サブメニュー項目の Y 座標
char *text	サブメニュー項目のタイトル
void (*func)(void)	選択されたときに実行する関数へのポインタ
struct SMENU *sub_menu	選択されたときに実行するサブメニューへのポインタ

prc_menu	
目的	メニュー選択インターフェイスを生成します。
書式	int prc_menu (MENU *menu);
引数	MENU *menu
	SMENU および MENU 構造体は、各モデルのヘッダファイルで定義されています。ユーザーは MENU 構造体を設定し、prc_menu をコールするだけで階層メニュー形式のユーザーインターフェイスを使用することができます。
コーディング例	<pre>// declare the MENU_ENTRY before the Menu reference MENU_ENTRY Collect; MENU_ENTRY Upload; MENU_ENTRY Download; MENU MyMenu = {3, 1, 0, "My menu", {&amp;Collect, &amp;Upload, &amp;Download}};  // declare function before the MENU_ENTRY reference Void FuncCollect (void); Void FuncUpload (void); Void FuncDownload (void); MENU_ENTRY Collect = {0, 1, "1. Collect", FuncCollect, 0}; MENU_ENTRY Upload = {0, 2, "2. Upload", FuncUpload, 0}; MENU_ENTRY Download = {0, 3, "3. Download", FuncDownload, 0};  Void FuncCollect (void) {     /* この関数の処理コードを記述 */ } Void FuncUpload (void) {     /* この関数の処理コードを記述 */ } Void FuncDownload (void) {     /* この関数の処理コードを記述 */ } Void main (void) {     // start_menu     clr_scr();     gotoxy(0,0);     while(1)     {         prc_menu (&amp;MyMenu);    /* MyMenu インターフェイスを生成 */         . . .     } }</pre>
戻り値	MENU 構造体の ReturnFlag が 1 の場合、1 が戻ります。オペレーションを中止で ESC キーが押された場合は、0 が戻ります。
備考	<p>ユーザー定義メニューを生成します。上下キー、ENT キーに加えて、ショートカットキーでの操作が提供されます。各項目タイトルの 1 文字目は、はショートカットキーとして扱われます。上記の例において、1、2、3 は、それぞれの項目のショートカットキーとなります。すなわち、[1]を押すと"Collect"の処理に入ります。メニュー項目の文字列の長さが画面 1 行分を超えている場合、一定間隔で分割して表示されます。</p> <p>➤ 8500/8700 シリーズはタッチスクリーンに対応しています。すなわち、項目にタッチすることで選択することができます。メニューが複数ページにわたる場合、最初のページ以外の毎ページの最終行に「ページ・アップ」アイコンが表示されます。</p>
参照	GetMenuPauseTime, SetMenuPauseTime

GetMenuPauseTime	
目的	メニュー項目の文字列の長さが画面 1 行分を超えている場合に、分割表示する間隔を取得します。
書式	unsigned long GetMenuPauseTime(void);
コーディング例	interval = GetMenuPauseTime();
戻り値	分割表示する間隔(5msec 単位)
参照	prc_menu

SetMenuPauseTime	
目的	メニュー項目の文字列の長さが画面 1 行分を超えている場合に、分割表示する間隔を設定します。
書式	void SetMenuPauseTime(unsigned long time);
引数	unsigned long time 分割表示する間隔(5msec 単位)
コーディング例	SetMenuPauseTime(200);            // set display interval to 1 second
戻り値	なし
備考	画面サイズやフォントサイズによってメニュー項目の文字列が 1 行で表示できる長さが変わってきます。メニュー項目の文字列の長さが画面 1 行分を超えている場合は分割して表示されます。それら分割表示される各文字列の表示間隔を設定します。デフォルトは 2 秒となっています。
参照	prc_menu

## 2.2 バ-コードリーダ

バ-コードリーダモジュールは下記にあるような多くのスキャンが用意されています。

スキャンタイプ: "O"は対応機種あり		8000	8200	8300	8400	8500	8700
1D	CCD	○	○	○	○	○	○
	標準レーザ	○	○	○	○	○	○
	ロングレンジレーザ	—	—	○	—	○	○
	エクストラロングレンジレーザ (ELR)	—	—	—	—	○	—
2D	2次元イメージャ	—	○	—	○	○	○

### 2.2.1 バ-コードの読取り

下記はバ-コードを解釈する処理と関係する4つのグローバル変数です。これらの変数はシステムによって宣言されていますので、ユーザープログラムはそれらを宣言する必要はありません。

<b>extern unsigned char</b>	<b>ScannerDesTbl[23];</b>	<b>// 8000</b>
	<b>ScannerDesTbl[40];</b>	<b>// 8300</b>
	<b>ScannerDesTbl[83];</b>	<b>// 8200,8400,8500,8700</b>

Decode()関数の動作はこの変数によって管理されます。

➤ ScannerDesTblの詳細については付録1、2を参照してください。

➤ 8200/8400/8500/8700シリーズでは、現状、ScannerDesTblの最初の43バイトのみが使用されていて、残りは予備となっています。

※ 2次元、(エクストラ)ロングレンジレーザのスキャンの場合、ScannerDesTblの設定変更を有効にするには、ConfigureReader()をコールする必要があります。

<b>extern char</b>	<b>CodeBuf[];</b>
--------------------	-------------------

バ-コードの読取りに成功すると、読取ったデータはこの変数に格納されます。

<b>extern char</b>	<b>CodeType;</b>
--------------------	------------------

バ-コードの読取りに成功すると、読取ったバ-コードの種別がこの変数に格納されます。

<b>extern char</b>	<b>CodeLen;</b>
--------------------	-----------------

バ-コードの読取りに成功すると、読取ったバ-コードの長さがこの変数に格納されます。

バ-コード読取りを有効にするにはまず InitScanner1()をコールして、リーダーポートを初期化します。リーダーポートを初期化後、プログラム内で Decode()をコールし、デコード処理を行います。

➤ CCD、レーザスキャンの場合、バ-コード読取りを行う際に利用する関数は、InitScanner1()、Decode()、HaltScanner1()の3つとなります。

➤ 2次元、(エクストラ)ロングレンジレーザのスキャンの場合、ScannerDesTblの設定変更を有効にするには、InitScanner1()をコールする前に ConfigureReader()をコールする必要があります。

<b>ConfigureReader</b>	<b>8200,8300,8400,8500,8700</b>
------------------------	---------------------------------

<b>目的</b>	ScannerDesTbl配列の設定変更を有効にします。
<b>書式</b>	int ConfigureReader(void);
<b>コーディング例</b>	<pre>memcpy(ScannerDesTbl, DefaultSetting, sizeof(DefaultSetting)); if (configureReader())     printf("Set OK"); else     printf("Set NG");</pre>
<b>戻り値</b>	0 = 失敗, 1 = 成功
<b>備考</b>	2次元、(エクストラ)ロングレンジレーザスキャンの場合、ScannerDesTblの新しい設定を有効にするには、この処理を実行する必要があります。 また、この処理は、InitScanner1 の前、HaltScanner1 の後に行う必要があります。
<b>参照</b>	ScannerDesTbl

Decode	
目的	バーコードのデコードを実行します。
書式	int Decode (void);
コーディング例	<pre>while (1) {     if (Decode()) break; }</pre>
戻り値	デコードに成功すると、読取ったバーコードデータの桁数(>0)を返し、デコードに成功していなければ、0を返します。
備考	<p>リーダポートを InitScanner1() で初期化後、この Decode 関数をコールしてバーコードのデコードを行います。</p> <p>➤ 上記のコーディング例のように、バーコードのデコードが必要な場合は、プログラムのループ内で Decode 関数をコールします。</p> <p>➤ バーコード読取りを長時間必要としない場合は、HaltScanner1() をコールし、リーダポートをリセット処理することをお奨めします。</p> <p>➤ Decode 関数がデコードに成功すると、システムグローバル変数 CodeBuf[] に読取ったバーコードデータがセットされます。また、システムグローバル変数 CodeLen 及び CodeType にもそれぞれ、適切な値がセットされます。</p>
参照	HaltScanner, InitScanner

HaltScanner1	
目的	リーダポートをリセットします。
書式	void HaltScanner1 (void);
コーディング例	HaltScanner1();
戻り値	無し
備考	<p>リーダポートをリセットします。再度、リーダを動作させたい場合は、InitScanner1() をコールする必要があります。</p> <p>➤ バーコード読取りを長時間必要としない場合は、HaltScanner1() をコールし、リーダポートをリセット処理することをお奨めします。</p>
参照	Decode, InitScanner1

InitScanner1	
目的	リーダポートを初期化します。
書式	void InitScanner1(void);
コーディング例	<pre>InitScanner1(); while (1) {     if (Decode()) break; }</pre>
戻り値	無し
備考	リーダポートは初期化後、使用可能となります。
参照	Decode, InitScanner1

## 2.2.2 CodeType 変数

下記に CodeType 変数にセットされ得るキャラクタとそれに対応するバーコード シンボルタイプを表に示します。

※ CCD,レーザースキャナでは、コード変換処理が行われた時に元のコード種別を格納するために OrgCodeType 変数が用意されています。

### 1 次元スキャナ用 CodeType 表

文字コード	キャラクタ	バーコード シンボルタイプ	対応スキャナ
63	?	Coop 25	8000,8200,8300,8400,8700-CCD,レーザースキャナ
64	@	ISBT 128	CCD,レーザースキャナ
65	A	Code 39	CCD,レーザースキャナ
66	B	Italian Pharmacode	CCD,レーザースキャナ
67	C	CIP 39 (French Pharmacode)	CCD,レーザースキャナ
68	D	Industrial 25	CCD,レーザースキャナ
69	E	Interleaved 25	CCD,レーザースキャナ
70	F	Matrix 25	CCD,レーザースキャナ
71	G	Codabar (NW7)	CCD,レーザースキャナ
72	H	Code 93	CCD,レーザースキャナ
73	I	Code 128	CCD,レーザースキャナ
74	J	UPC-E0 / UPC-E1	CCD,レーザースキャナ
75	K	UPC-E with Addon 2	CCD,レーザースキャナ
76	L	UPC-E with Addon 5	CCD,レーザースキャナ
77	M	EAN-8	CCD,レーザースキャナ
78	N	EAN-8 with Addon 2	CCD,レーザースキャナ
79	O	EAN-8 with Addon 5	CCD,レーザースキャナ
80	P	EAN-13 / UPC-A	CCD,レーザースキャナ
81	Q	EAN-13 with Addon 2	CCD,レーザースキャナ
82	R	EAN-13 with Addon 5	CCD,レーザースキャナ
83	S	MSI	CCD,レーザースキャナ
84	T	Plessey	CCD,レーザースキャナ
85	U	GS1-128 (EAN-128)	CCD,レーザースキャナ
86	V	予備	
87	W	予備	
88	X	予備	
89	Y	予備	
90	Z	Telepen	CCD,レーザースキャナ
91	[	GS1 DataBar (RSS)	CCD,レーザースキャナ
92	\	予備	
93	]	予備	

OrgCodeType 変数は、コード変換処理が行われた時に元のコード種別を格納するために用意されています。

例えば、「EAN-8→EAN-13 変換」が設定されている場合、EAN-8 のバーコードを読み取ると EAN-13 に変換されます。このとき、バーコード種別は EAN-13 で、元のバーコード種別は EAN-8 となります。

### OrgCodeType 表

文字コード	キャラクタ	バーコード シンボルタイプ	対応スキャナ
65	A	UPC-E	CCD,レーザースキャナ
66	B	UPC-E with Addon 2	CCD,レーザースキャナ
67	C	UPC-E with Addon 5	CCD,レーザースキャナ
68	D	EAN-8	CCD,レーザースキャナ
69	E	EAN-8 with Addon 2	CCD,レーザースキャナ
70	F	EAN-8 with Addon 5	CCD,レーザースキャナ
71	G	EAN-13	CCD,レーザースキャナ
72	H	EAN-13 with Addon 2	CCD,レーザースキャナ
73	I	EAN-13 with Addon 5	CCD,レーザースキャナ
74	J	UPC-A	CCD,レーザースキャナ
75	K	UPC-A with Addon 2	CCD,レーザースキャナ
76	L	UPC-A with Addon 5	CCD,レーザースキャナ
0	NUL	変換なし	CCD,レーザースキャナ

2次元スキャナ,(エクストラ)ロングレンジレーザースキャナ用 CodeType 表

文字コード	キャラクタ	バーコード シンボルタイプ	対応スキャナ
64	@	ISBT 128	2次元,(エクストラ)ロングレンジレーザ -
65	A	Code 39	2次元,(エクストラ)ロングレンジレーザ -
66	B	Italian Pharmacode	2次元,(エクストラ)ロングレンジレーザ -
67	C	なし	
68	D	なし	
69	E	Interleaved 25	2次元,(エクストラ)ロングレンジレーザ -
70	F	Matrix 25	8200,8400,8700-2次元
71	G	Codabar (NW7)	2次元,(エクストラ)ロングレンジレーザ -
72	H	Code 93	2次元,(エクストラ)ロングレンジレーザ -
73	I	Code 128	2次元,(エクストラ)ロングレンジレーザ -
74	J	UPC-E0	2次元,(エクストラ)ロングレンジレーザ -
75	K	UPC-E with Addon 2	2次元,(エクストラ)ロングレンジレーザ -
76	L	UPC-E with Addon 5	2次元,(エクストラ)ロングレンジレーザ -
77	M	EAN-8	2次元,(エクストラ)ロングレンジレーザ -
78	N	EAN-8 with Addon 2	2次元,(エクストラ)ロングレンジレーザ -
79	O	EAN-8 with Addon 5	2次元,(エクストラ)ロングレンジレーザ -
80	P	EAN-13	2次元,(エクストラ)ロングレンジレーザ -
81	Q	EAN-13 with Addon 2	2次元,(エクストラ)ロングレンジレーザ -
82	R	EAN-13 with Addon 5	2次元,(エクストラ)ロングレンジレーザ -
83	S	MSI	2次元,(エクストラ)ロングレンジレーザ -
84	T	なし	
85	U	GS1-128 (EAN-128)	2次元,(エクストラ)ロングレンジレーザ -
86	V	予備	
87	W	予備	
88	X	予備	
89	Y	予備	
90	Z	予備	
91	[	GS1 DataBar Omnidirectional (RSS-14)	2次元,(エクストラ)ロングレンジレーザ -
92	\	GS1 DataBar Limited (RSS Limited)	2次元,(エクストラ)ロングレンジレーザ -
93	]	GS1 DataBar Expanded (RSS Expanded)	2次元,(エクストラ)ロングレンジレーザ -
94	^	UPC-A	2次元,(エクストラ)ロングレンジレーザ -
95	_	UPC-A with Addon 2	2次元,(エクストラ)ロングレンジレーザ -
96	`	UPC-A with Addon 5	2次元,(エクストラ)ロングレンジレーザ -
97	a	UPC-E1	2次元,(エクストラ)ロングレンジレーザ -
98	b	UPC-E1 with Addon 2	2次元,(エクストラ)ロングレンジレーザ -
99	c	UPC-E1 with Addon 5	2次元,(エクストラ)ロングレンジレーザ -
100	d	TLC-39 (TCIF Linked Code 39)	2次元
101	e	Trioptic (Code 39)	2次元,(エクストラ)ロングレンジレーザ -
102	f	Bookland (ENA)	2次元,(エクストラ)ロングレンジレーザ -
103	g	Code 11	2次元,8300,8700-ロングレンジレーザ -
104	h	Code 39 Full ASCII	2次元,(エクストラ)ロングレンジレーザ -
105	i	IATA(※) (25)	2次元,(エクストラ)ロングレンジレーザ -
106	j	Industrial 25 (Discrete 25)	2次元,(エクストラ)ロングレンジレーザ -
107	k	PDF417	2次元
108	l	MicroPDF417	2次元
109	m	Data Matrix	2次元
110	n	Maxicode	2次元
111	o	QR Code	2次元
112	p	US Postnet	2次元
113	q	US Planet	2次元
114	r	UK Postal	2次元
115	s	Japan Postal	2次元
116	t	Australian Postal	2次元
117	u	Dutch Postal	2次元
118	v	Composite Code	2次元
119	w	Macro PDF417	2次元

120	x	Macro MicroPDF417	2 次元
121	y	Chinese 25	8200,8400,8700-2 次元
122	z	Aztec	8200,8400,8700-2 次元
123	{	MicroQR	8200,8400,8700-2 次元
124		USPS 4CB / One Code / Intelligent Mail	8200,8400,8700-2 次元
125	}	UPU FICS Postal	8200,8400,8700-2 次元
126	~	Coupon Code	2 次元,(エクストラ)ロング レザ -

※ IATA は国際航空運送協会(International Air Transport Association)の略称です。このバーコード 種別は航空チケットで使用されています。

### 2.2.3 スキャナ設定テーブル変数

符号なし配列の ScannerDesTable(=Scanner Description Table)は Decode 関数の動作を管理しています。

ScannerDesTable 変数の詳細は付録 1 にある 2 つの表を参照してください。

➤ 表 1 は CCD,レザ -スキャナ用、表 2 は 2 次元,(エクストラ)ロング レザ -スキャナ用です。

バーコード 毎のパラメータは付録 2 を参照してください。スキャナのパラメータは付録 3 を参照してください。



## 2.3 RFID リーダ

8300/8500/8700 シリーズには、バーコードリーダーにオプションの RFID リーダを付けることができます。

➤ RFID を使用するには外部ライブラリが必要です。

	ハードウェア構成	必要なライブラリ
8300	8300 – バッチ + RFID	83RFID.lib
	8370 – 802.11b/g + RFID	83WLAN.lib + 83RFID.lib

RFID リーダは、使用するタグにより異なりますが、読取り/書き込み処理をサポートしています。ISO 15693 Icode、ISO 14443A および ISO 14443B をサポートしています。動作確認の結果は次の通りです。

※ プログラミングするにあたり、RFID タグの仕様を理解することをお勧めします。

タグ種別	UID のみ	読取り	書き込み
<b>TAG_MifareISO14443A</b>			
Mifare Standard 1K	○	○	○
Mifare Standard 4K	○	○	○
Mifare Standard Ultralight	○	○	○
Mifare DESFire	○	—	—
Mifare S50	○	○	○
SLE44R35	○	—	—
SLE66R35	○	○	○
<b>TAG_SR176</b>			
SRIX 4K	○	○	○
SRIX176	○	○	○
<b>TAG_ISO15693</b>			
ICODE SLI	○	○	○
SRF55V02P	○	—	—
SRF55V02S	○	—	—
SRF55V10P	○	—	—
TI Tag-it HF-I	○	○	○
<b>TAG_ICODE</b>			
ICODE	○	○	○

※ この結果は、RFID モジュールバージョン 1.0 のものです。○はサポートありを意味します。RFIDVersion()を使用することで、バージョン情報を取得することができます。

### 2.3.1 パーチャル COM

RFID リーダプロトコルミッドのAPI リストは、COM ポートに関する処理をコールすることで実現されます。RFID のパーチャル COM ポートは COM4 に定義されています。

- open\_com(4, int)      RFID COM ポートを初期化、有効にします。(int は任意の値)
- close\_com(4)      RFID COM ポートを切断、無効にします。
- read\_com(4, char\*)      RFID COM ポートからカードデータを読み取ります。
- write\_com(4, char\*)      RFID COM ポート経由でカードデータを書込みます。

各関数の戻り値は次の通りです。

関数	戻り値	
read_com(4, char*)	-1	タグなし
	-2	タグ取得失敗
	-3	タグパスワード取得失敗
	-5	認証失敗
	0 以上	取得したデータの長さ
Com_eot(4)	-1	タグなし
	-2	タグ取得失敗
	-3	タグパスワード取得失敗
	-4	タグパスワード書き込み失敗
	-5	認証失敗
	0	その他エラー
	1	成功

### 2.3.2 RFID パラメータ構造体

特定のタグを読み書きする前に、RFIDReadFormat()および RFIDWriteFormat()をコールして RFID のパラメータを設定する必要があります。

パラメータ	説明														
unsigned char TagType[4]	TagType[0]														
	<table><tr><td>Bit6～7</td><td>Bit5</td><td>Bit4</td><td>Bit3</td><td>Bit2</td><td>Bit1</td><td>Bit0</td></tr><tr><td>予備</td><td>ISO 14443B</td><td>SR176</td><td>ISO 14443A</td><td>Icode</td><td>Tagit</td><td>ISO 15693</td></tr></table>	Bit6～7	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	予備	ISO 14443B	SR176	ISO 14443A	Icode	Tagit	ISO 15693
	Bit6～7	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0								
	予備	ISO 14443B	SR176	ISO 14443A	Icode	Tagit	ISO 15693								
➤ TagType[1～3]：予備															
unsigned int StartByte	読み込み/書き込み処理のデータの開始バイト位置														
unsigned int MaxLen	➤ 読み込み時：最大データ長(1～255)、0 は UID データのみ ➤ 書き込み時：予備(任意の値)														
unsigned char Reserve[20]	予備														

### 2.3.3 RFID データフォーマット

read\_com()で取得したデータのフォーマットは次の通りです。

0 バイト			1~17 バイト	18 バイト以降
タグ種別	'V' 'T' 'I' 'M' 'S' 'Z'	TAG_ISO15693 TAG_Tagit TAG_Icode TAG_MifareISO14443A TAG_SR176 TAG_ISO14443B	Tag UID (SN)	データ

RFIDReadFormat		8300,8500,8700
目的	RFID 読取りのパラメータを設定します。	
書式	void RFIDReadFormat(RFIDParameter *source);	
引数	RFIDParameter *source 読取り動作のためのパラメータ	
コーディング例	<pre>parameter.TagType[0] = 0x3f;    // all supported tag types are enabled parameter.StartByte = 0; parameter.MaxLen = 150; RFIDReadFormat(&amp;parameter);</pre>	
戻り値	なし	
備考	パラメータは読取り処理を行う前に設定する必要があります。	

RFIDWriteFormat		8300,8500,8700
目的	RFID 読取りのパラメータを設定します。	
書式	void RFIDWriteFormat(RFIDParameter *source);	
引数	RFIDParameter *source 書込み動作のためのパラメータ	
コーディング例	<pre>parameter.TagType[0] = 0x01;    // tag type ISO15693 is enabled parameter.StartByte = 0; parameter.MaxLen = 0; RFIDWriteFormat(&amp;parameter);</pre>	
戻り値	なし	
備考	パラメータは書込み処理を行う前に設定する必要があります。	

## 2.3.4 RFID 認証

<b>GetRFIDSecurityKey</b>	<b>8300,8500,8700</b>
---------------------------	-----------------------

目的	タグ固有のセキュリティキーのステータスをチェックします。														
書式	int GetRFIDSecurityKey(unsigned char TagType, unsigned char *KeyString, unsigned char *KeyType);														
引数	<table> <tr> <td>unsigned char TagType</td><td></td></tr> <tr> <td>'V'</td><td>TAG_ISO15693</td></tr> <tr> <td>'T'</td><td>TAG_Tagit</td></tr> <tr> <td>'I'</td><td>TAG_Icode</td></tr> <tr> <td>'M'</td><td>TAG_MifareISO14443A</td></tr> <tr> <td>'S'</td><td>TAG_SR176</td></tr> <tr> <td>'Z'</td><td>TAG_ISO14443B</td></tr> </table> タグ種別の詳細は 2.3 にある表を参照してください。	unsigned char TagType		'V'	TAG_ISO15693	'T'	TAG_Tagit	'I'	TAG_Icode	'M'	TAG_MifareISO14443A	'S'	TAG_SR176	'Z'	TAG_ISO14443B
unsigned char TagType															
'V'	TAG_ISO15693														
'T'	TAG_Tagit														
'I'	TAG_Icode														
'M'	TAG_MifareISO14443A														
'S'	TAG_SR176														
'Z'	TAG_ISO14443B														
	unsigned char *KeyString														
	セキュリティキーの値が格納される変数(文字列)へのポインタ														
	unsigned char *KeyType														
	セキュリティ種別の値が格納される変数へのポインタ														
コーディング例	<pre>if (!GetRFIDSecurityKey(TAG_MifareISO14443A, key_buffer, &amp;keytype)) {     Printf("No Security Key."); }</pre>														
戻り値	セキュリティキーが設定されている場合は 1、それ以外の場合は 0 が戻ります。														
備考	この処理はタグ固有のセキュリティキーが Mifare Standard 1K/4K または SLE66R35 のように設定されているかどうか調べるために使用されます。														

<b>SetRFIDSecurityKey</b>	<b>8300,8500,8700</b>
---------------------------	-----------------------

目的

タグ固有のセキュリティキーを設定します。

書式

void SetRFIDSecurityKey(unsigned char TagType, unsigned char \*KeyString, unsigned char KeyType);

引数

unsigned char TagType

'V'	TAG_ISO15693	タグ種別の詳細は 2.3 にある表を参照してください。
'T'	TAG_Tagit	
'I'	TAG_Icode	
'M'	TAG_MifareISO14443A	
'S'	TAG_SR176	
'Z'	TAG_ISO14443B	

unsigned char \*KeyString

セキュリティキーの値が格納される変数(文字列)へのポインタ

unsigned char KeyType

1	MIFARE_KEY_A	Key A
2	MIFARE_KEY_B	Key B

コーディング例

SetRFIDSecurityKey(TAG\_MifareISO14443A, "FFFFFFFF", MIFARE\_KEYA);  
// Set Key A with a specified value for ISO14443A tags

戻り値

なし。

備考

この処理はタグ固有のセキュリティキーを Mifare Standard 1K/4K または SLE66R35 のように設定します。

## 2.4 キーボードウエッジインターフェイス

モデル 8300 シリーズ は、専用のキーボードウエッジケーブルを本体に接続することで、PC との間でキーボードウエッジによるデータ送信を行うことが可能になります。システムグローバル変数 WedgeSetting で、PC やキーボードウエッジに関する適切な設定を行い、SendData() で実際のデータを送信します。

Bluetooth HID、USB HID、ウエッジエミュレータモジュールはキーボードウエッジケーブルに対応していません。下の表『2.4.3 ウエッジエミュレータ』、Part II の『付録 4 使用例』を参照してください。

ウエッジ	使用する関数	対象ハードウェアミドル
キーボードウエッジケーブル	WedgeSetting 変数 SendData() WedgeReady()	8300 シリーズ
ウエッジエミュレータ(IR, IrDA, RS-232)	SendData() WedgeReady() open_com() SetCommType() close_com()	8000/8300/8500 シリーズ
ウエッジエミュレータ(Bluetooth SPP)	SendData() WedgeReady() open_com() SetCommType() close_com()	8000/8300/8500 シリーズ
Bluetooth HID または USB HID	WedgeSetting 変数 SetCommType() open_com() com_eot() write_com() nwrite_com() close_com()	8000/8200/8300/8400/8500/8700 シリーズ

**extern char WedgeSetting[3];**

SendData() の処理はこの変数によって管理されています。

**SendData** 8000,8300,8500

**目的** キーボードウエッジインターフェイスを通して、データを送信します。

**書式** void SendData (char\* out\_str);

**引数** char\* out\_str

送信するデータの格納された変数へのポインタ

**コーディング例** SendData (CodeBuf);

**戻り値** 無し

**WedgeReady** 8000,8300,8500

**目的** キーボードウエッジインターフェイスケーブルが正しく接続されているかをチェックします。

**書式** int WedgeReady (void);

**コーディング例** if (WedgeReady())

SendData (CodeBuf);

**戻り値** キーボードウエッジインターフェイスケーブルが正しく接続されていれば、1 を返し、接続されていなければ、0 を返します。

**備考** 実際にデータを送信する前に、この関数でキーボードウエッジインターフェイスケーブルの接続状態を確認します。

## 2.4.1 WedgeSetting 変数定義

配列	ビット位置	説明
0	7-0	キーボード (PC) タイプ
1	7	1: CAPS ロック自動検出有り 0: CAPS ロック自動検出無し
1	6	1: CAPS ロック オフ 0: CAPS ロック オン
1	5	1: 大文字/小文字を無視 0: 大文字/小文字を認識
1	4-3	00: 一般キーボード 10: 下段数字キーボード 11: 上段数字キーボード
1	2-1	00: 一般キーボード 10: CAPS ロックキーボード 11: SHIFT ロックキーボード
1	0	1: 数字データをテンキーデータとして送信 0: 数字データをフルキーデータとして送信
2	7-0	送信キャラクタ間ディレイ

※ 上記の設定は、一部の PC では正しく動作しない場合があります。動作検証の上、ご使用ください。

### 配列 0: キーボード (PC) タイプ

下記に WedgeSetting[0] に設定可能な値を示します。

設定値	キーボード (PC) タイプ	設定値	キーボード (PC) タイプ
1	PC AT (USA)	21	PS55 002-81, 003-81
2	PC AT (フランス)	22	PS55 002-2, 003-2
3	PC AT (ドイツ)	23	PS55 002-82, 003-82
4	PC AT (イタリア)	24	PS55 002-3, 003-3
5	PC AT (スイーデン)	25	PS55 002-8A, 003-8A
6	PC AT (ノルウェー)	26	IBM 3477 TYPE 4
7	PCAT (UK)	27	PS2-30
8	PCAT (ベルギー)	28	Memorex Telex 122 Keys
9	PCAT (スペイン)	29	PCXT
10	PCAT (ポルトガル)	30	IBM 5550
11	PS55 A01-1	31	NEC 5200
12	PS55 A01-2	32	NEC 9800
13	PS55 A01-3	33	DEC VT220,320,420
14	PS55 001-1	34	Macintosh (ADB)
15	PS55 001-81	35	Hitachi Elles
16	PS55 001-2	36	Wyse Enhance KBD (US)
17	PS55 001-82	37	NEC Astra
18	PS55 001-3	38	Unisys TO-300
19	PS55 001-8A	39	Televideo 965
20	PS55 002-1, 003-1	40	ADDS 1010

### 配列 1

- **CAPS ロック自動検出**  
この設定が有効(有効)に設定されている場合、SendData 関数は、データ送信前に CAPS ロックステータスを自動検出し、データを正しく入力することができます。但し、この設定は、PC AT, PS2-30, PS55, Memorex Telex でのみ有効で、その他のキーボード (PC) タイプでは、次の CAPS ロックステータスの設定に従って、データは送信されます。
- **CAPS ロックステータス**  
SendData 関数でフルバートタイプ(大文字・小文字)に従って、正しくデータを送信しようとする場合、予め CAPS ロックステータスが正しく設定されていることが前提となります。間違った CAPS ロックステータスが設定されていると、実際の大文字データが小文字に、小文字データが大文字で入力されることになります。
- **大文字/小文字**  
この設定を「大文字/小文字を認識」に設定した場合、SendData 関数は、データの大文字・小文字を区別して正しく送信を行います。但し、「大文字/小文字を無視」に設定した場合は、大文字・小文字に関係無く、SHIFT キーを付加せずにデータを送信します。
- **数字キーボード**

この設定を「上段数字キーボード」に設定した場合、SendData 関数は、数字データを送信する際、SHIFT キーを付加して送信します。通常は、「一般キーボード」に設定してください。尚、この設定は、PC AT, PS2-30, PS55, Memorex Telex でのみ有効です。

- **CAPS/SHIFT ロックキーボード**

キーボードが CAPS ロックタイプであるか、SHIFT ロックタイプであるかを設定します。通常は、「一般キーボード」に設定してください。尚、この設定は、PC AT, PS2-30, PS55, Memorex Telex でのみ有効です。

- **数字データ送信**

この設定を「数字データをテンキーデータとして送信」に設定した場合、SendData 関数は、数字データをテンキーデータとして送信します。フルキーデータとして送信したい場合は、「数字データをフルキーデータとして送信」に設定してください。

## 配列 2：送信キャラクタ間デレイ

送信キャラクタ間デレイには、0 ~ 255 ミリ秒を設定することが可能です。レスポンスの遅い PC など、データ送信に支障が生じる場合、送信キャラクタ間デレイに適切な値を設定し、送信速度を調整します。

### 2.4.2 キーボードウィザインターフェイスキャラクタ表

SendData()は、下記の表に従ってデータの送信を行います。

	00	10	20	30	40	50	60	70	80
0		F2	SP	0	@	P	`	p	⑩
1	INS	F3	!	1	A	Q	a	q	①
2	DLT	F4	"	2	B	R	b	r	②
3	Home	F5	#	3	C	S	c	s	③
4	End	F6	\$	4	D	T	d	t	④
5	Up	F7	%	5	E	U	e	u	⑤
6	Down	F8	&	6	F	V	f	v	⑥
7	Left	F9	'	7	G	W	g	w	⑦
8	BS	F10	(	8	H	X	h	x	⑧
9	HT	F11	)	9	I	Y	i	y	⑨
A	LF	F12	*	:	J	Z	j	z	
B	Right	ESC	+	;	K	[	k	{	
C	PgUp	Exec	,	<	L	\	l		
D	CR	CR*	-	=	M	]	m	}	
E	PgDn		.	>	N	^	n	~	
F	F1		/	?	O	_	o	Dly	ENTER*

(1) Dly：デレイ 100 ミリ秒

(2) ⑩～⑨：テンキーデータ

(3) CR\*/Send/ENTER\*：テンキーの ENTER キー

SendData()では、前頁のキャラクタを単純に送信するだけでなく、特殊マシコードを利用して、ジョブネーションやダクトスキャンコードの送信も行うことができます。下記に特殊マシコードの説明を行います。

- 0xC0：次に続くコードがダクトスキャンコードであることを意味します。
- 0xC0 | 0x01：次に続くキャラクタに SHIFT キーを付加して送信します。
- 0xC0 | 0x02：次に続くキャラクタに左 CTRL キーを付加して送信します。
- 0xC0 | 0x04：次に続くキャラクタに左 ALT キーを付加して送信します。
- 0xC0 | 0x08：次に続くキャラクタに右 CTRL キーを付加して送信します。
- 0xC0 | 0x10：次に続くキャラクタに右 ALT キーを付加して送信します。
- 0xC0 | 0x20：次に続くキャラクタを送信後、全てのジョブネーションステータスをクリアします。

例えば、

[A] [CTRL-INSERT] [5] [スキャンコード 0x29] [TAB] [2] [SHIFT-CTRL-A] [B] [ALT-1] [ALT-2-BREAK] [ALT-1] [ALT-3] と送信したい場合、下記のキャラクタを SendData 関数で送信します。

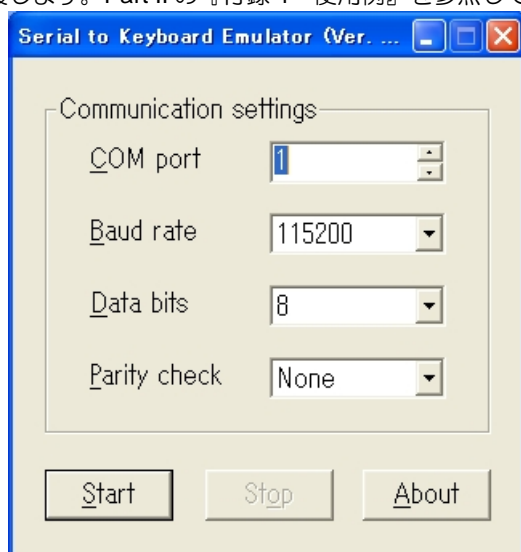
0x41,0xC2,0x01,0x35,0xC0,0x29,0x09,0x32,0xC3,0x41,0x42,0xC4,0x31,0xE4,0x32,0xC4,0x31,0xC4,0x33,0x00

※ スキャンコード 0x29 = スペース (PC AT 機), ALT+12 = フォームフィード, ALT+13 = ENTER に相当します。

※ ALT+12 の後の BREAK コードは必須です。省略した場合、ALT-12, ALT-13 では無く、ALT1213 と送信されます。

### 2.4.3 ウィッツィ イミュレータ

8000/8300/8500 シリーズ用にウィッツィ イミュレータプログラム「シリアル→キーボード変換」(Serial2KB.exe)が用意されています。SendData()や WedgeReady()のように、一般的なウィッツィ 機能である IR/IrDA/RS-232/Bluetooth SPP によって受信したデータをキーボード入力に変換します。このユーティリティは、シリアルポート入力処理のないアプリケーションでのキーボードキー入力を開発するのを支援します。Part II の『付録 4 使用例』を参照してください。





## 2.5 ブザー

ここでは、ブザーを操作する関数について説明します。ブザーを鳴動させる場合、ブザー周波数とブザー間隔をヘッパで定義したブザーシケンスを引数として、on\_beeper 関数または play 関数をコールします。この関数をコールすると、ハードウェアシステムは自動的に定義されたブザーシケンスに従ってブザーをバックグラウンドで鳴動させます。従って、プログラムは、ブザーの鳴動が終わるのを待つ必要はありません。

beeper\_status()はブザーが鳴動中か判断するために、off\_beeper()はブザーを直ちに終了させるために使用します。

※ 8200 シリーズはブザーの代わりにスピーカを搭載しています。

### 2.5.1 ブザーシケンス

ブザー周波数及びブザー間隔を指示するブザーシケンスは、int 型配列変数で定義します。ブザー周波数とブザー間隔をヘッパで定義し、1つのヘッパが1回のブザーを表します。

### 2.5.2 ブザー周波数

ブザー周波数には、ブザーを鳴動した時のブザー周波数(トーン)を int 型の値で指定します。指定する周波数値は、実際の周波数とは異なり、下記の計算式で算出します。

ブザー周波数 = 76000 / 実際に鳴動したい周波数

例えば、4KHz のブザーを鳴動したい場合は、ブザー周波数を 19 にセットします。また、ブザーを一時停止(ホース)したい場合は、ブザー周波数を 0 にセットします。適切なブザー周波数範囲は、1~6KHz で、4KHz がピークとなります。また、最後で、ブザー間隔を 0 にすれば、ブザーの鳴動を終了することができます。

※ 停止ではなく、一時停止にするには、ブザー周波数に 0 にセットします。

### 2.5.3 ブザー間隔

ブザー間隔には、ブザーを鳴動したい時間を int 型の値で指定します。設定は、10 ミリ秒単位で行い、1 秒間ブザーを鳴動したい場合は、100 をセットします。また、0 をセットすると、ブザーの鳴動が終了します。

#### beeper\_status

目的	ブザーが鳴動中であることをチェックします。
書式	int beeper_status (void);
コーディング例	while (beeper_status());                    /* ブザー鳴動が終了するまで待機 */
戻り値	ブザーが鳴動中であれば、1 を返し、終了していれば、0 を返します。

#### get\_beeper\_vol

8200,8400

目的	音量を取得します。
書式	int get_beeper_vol(void);
コーディング例	val = get_beeper_vol();                    // get the volume level
戻り値	音量

#### set\_beeper\_vol

8200,8400

目的	音量を設定します。
書式	void set_beeper_vol(int level);
引数	int level

0	MUTE_VOL	消音 (8200 のみ)
1	LOW_VOL	音量小
2	MEDIUM_VOL	音量中
3	HIGH_VOL	音量大

コーディング例	set_beeper_vol(2);                    // set the volume level to "Medium"
戻り値	なし

on_beeper							
目的	指定のブザーシーケンスに従ってブザーを鳴動します。						
書式	void on_beeper(const int* sequence); // 8000,8300,8400,8500,8700 unsigned char on_beeper(const void *buffer); // 8200 のみ						
引数	const int* sequence ブザーシーケンス配列変数へのポインタ const void *buffer 以下のいずれかの変数へのポインタ (1) ブザーシーケンス配列変数 (2) ウェーブテーブルの格納された変数 (3) SD カードに保存されている WAV ファイルの名前。ファイル名は"A:\\"、"a:\\"、"A:/\"、"a:/"で始まる。						
コーディング例(1)	int two_beeps[] = { 19, 10, 0, 10, 19, 10, 0, 0 }; on_beeper (two_beeps);						
コーディング例(2)	on_beeper ("A:\\Sound.wav"); // play a wave file from SD card on 8200						
コーディング例(3)	on_beeper ("A:\\Sound"); // filename extension is optional						
戻り値	8200 シリーズの場合、以下の戻り値があります。						
	<table> <tr><td>0</td><td>成功</td></tr> <tr><td>1</td><td>フォーマット不正</td></tr> <tr><td>2</td><td>SD カードに該当ファイルなし</td></tr> </table>	0	成功	1	フォーマット不正	2	SD カードに該当ファイルなし
0	成功						
1	フォーマット不正						
2	SD カードに該当ファイルなし						
備考	<p>指定のブザーシーケンスに従ってブザーを鳴動します。但し、ブザーシーケンス終了前に、次のブザーシーケンスが実行されると、先のシーケンスはキャンセルされ、新しいシーケンスだけが実行されます。</p> <p>8200 シリーズでは以下の仕様の WAV 形式のファイルを使用できます。</p> <ul style="list-style-type: none"> <li>➤ チャンネル：モノラル</li> <li>➤ サンプルレート：8000、11025、22050、32000、44100</li> <li>➤ サンプルサイズ：8bit、16bit</li> </ul>						
off_beeper							
目的	ブザーシーケンスを強制終了します。						
書式	void off_beeper(void);						
コーディング例	off_beeper();						
戻り値	無し						

play	
目的	指定されたブザー・シーケンスに従ってメロディを演奏します。
書式	void play(const char* sequence);
引数	const int* sequence
	メロディ・シーケンスが格納されたバッファへのポインタ
コーディング例	<pre>const char song[] = {0x31, 10, 0x32, 10, 0x33, 10, 0x34, 10, 0x35, 10, 0x36, 10,                     0x37, 10, 0x41, 10, 0x31, 4, 0x32, 4, 0x33, 4, 0x34, 4, 0x35, 4, 0x36, 4,                     0x37, 4, 0x41, 4, 0x00, 0 }; play (song);</pre>
戻り値	なし。
備考	この関数は、on_beeper()に似ています。ただし、周波数特性は以下のように指定されています。

ビット	7	6	5	4	3	2	1	0
	予備	A(ラ)音階の周波数			#記号	音階		
		000 : 予備			0:無効	000 : 予備		
		001(1) : 55Hz			1:有効	001(1) : ド		
		010(2) : 110Hz				001(2) : レ		
		011(3) : 220Hz				001(3) : ミ		
		100(4) : 440Hz				001(4) : ファ		
		101(5) : 880Hz				001(5) : ソ		
		110(6) : 1760Hz				001(6) : ラ		
		111(7) : 3520Hz				001(7) : シ		

## 2.6 LED

Cipher シリーズ ハンディターミナルは、ホータヘステ-ルを知らせるための LED を装備しています。

### set\_led

**目的** LED を制御します。

**書式** void set\_led (int led, int mode, int duration);

**引数** int led

0	LED_RED	赤色。
1	LED_GREEN	緑色。
2	LED_BULE	青色。8200/8400/8700 シリーズ で、ワイヤ通信時に使用する方の LED。
3	LED_GREEN2	緑色。8200/8400/8700 シリーズ で、ワイヤ通信時に使用する方の LED。

int mode

0	LED_OFF	duration で指定された時間(x10 ミリ秒単位)消灯してから点灯
1	LED_ON	duration で指定された時間(x10 ミリ秒単位)点灯してから消灯
2	LED_FLASH	duration で指定された間隔(x10 ミリ秒単位)でフラッシュ(点滅)
0xf0	LED_SYSTEM_CTRL	8200/8400/8700 シリーズ で、ワイヤ通信時に使用する方の LED をデフォルトに設定します。 ➤ 青色は、Bluetooth の通信状態です。速い点滅は、接続待ちを意味し、ゆっくりとした点滅は接続済みを意味します。 ➤ 緑色は、Wi-Fi の通信状態です。速い点滅は、接続待ちを意味し、ゆっくりとした点滅は接続済みを意味します。
0xf1	LED_USER_CTRL	8200/8400/8700 シリーズ で、ワイヤ通信時に使用する方の LED を使用します。

int duration

間隔を 10 ミリ秒単位で設定します。  
➤ 引数 mode が LED\_SYSTEM\_CTRL または LED\_USER\_CTRL の場合、このパラメータは無視されます。

**コーディング例**

```
set_led (LED_RED, LED_FLASH, 50);
// set red LED to flash for each 1 second cycle
set_led (LED_BLUE, LED_USER_CTRL, 0);
set_led (LED_BLUE, LED_FLASH, 0); // set blue LED on 8400 for user control
```

**戻り値**

無し

## 2.7 バイブレータとヒータ

この章はバイブレータとヒータの処理に関する記述です。

➤ バイブレータ：ステータス認識のために使用します。

➤ ヒータ：使用環境の温度が-10℃未満に下がるような寒い場所で、LCD を機能させるために使用されます。

### 2.7.1 バイブレータ

バイブレータ機能は 8200/8300/8400/8500/8700 シリーズでサポートされています。

※ ただし、8300 シリーズでは、ハードウェアのバージョン 4 以上が必要です。

GetVibrator	8200,8300,8400,8500,8700
-------------	--------------------------

目的	バイブレータのステータスを取得します。。
書式	int GetVibrator(void);
コーディング例	val = GetVibrator();
戻り値	1 = バイブレータ ON, 0 = バイブレータ OFF

SetVibrator	8200,8300,8400,8500,8700
-------------	--------------------------

目的	バイブレータのステータスを設定します。。	
書式	void SetVibrator(int mode);	
引数	int mode	
	0	バイブレータ OFF
	1	バイブレータ ON
コーディング例	SetVibrator(1); // turn on the vibrator	
戻り値	なし	
備考	一度 SetVibrator(1)でバイブレータを ON にすると、SetVibrator(0)で OFF にするまで、振動し続けます。	

## 2.7.2 ヒータ

<b>GetHeaterMode</b>	<b>8500</b>
----------------------	-------------

**目的** ヒータのステータスを取得します。。

**書式** `int GetHeaterMode(void);`

**コーディング例** `val = GetHeaterMode();`

**戻り値** 1 = ヒータ ON, 0 = ヒータ OFF

<b>SetHeaterMode</b>	<b>8500</b>
----------------------	-------------

**目的** ヒータのステータスを設定します。。

**書式** `void SetHeaterMode(int mode);`

**引数** `int mode`

0	ヒータ OFF
1	ヒータ ON

**コーディング例** `SetHeaterMode(1); // turn on the heater`

**戻り値** なし

**備考** 一度 `SetHeaterMode(1)` でヒータを ON し、使用環境の温度が-10℃未満に下がると `SetHeaterMode(0)` で OFF にするまで、自動的に発熱します。

## 2.8 リアルタイムクロック

ここでは、カウンタやタイマーを操作する関数について説明します。

### 2.8.1 カウンタ

CipherLab シリーズ ハードウェアモジュールの日付及び時刻は、バックアップ付カウンタチップ内に保持されており、`get_time()`及び`set_time()`で取得及び設定を行うことができます。

➤ うるう年の扱いは、カウンタチップが自動的にを行っているため、ユーザーは特に意識をする必要はありません。

※ システムグローバル変数として用意されている `sys_msec` 変数及び `sys_sec` 変数は、カウンタチップとは無関係に CPU タイマーにより操作されているため、精度は CPU クロックに依存しています。また、電源が時に 0 ヘルシットされるため、正確な時間操作には適していません。

#### DayOfWeek

目的	カウンタチップから現在の日付に対応する曜日を取得します。
書式	int DayOfWeek (void);
コーディング例	day = DayOfWeek ( );
戻り値	曜日に対応する 1~7 を返します。
備考	1(月), 2(火), 3(水), 4(木), 5(金), 6(土), 7(日)

#### get\_time

目的	カウンタチップから現在の日付・時刻を取得します。
書式	void get_time (char*cur_time);
引数	char* cur_time 取得した日付・時刻を格納する変数へのポインタ。 Char 型配列変数は、日付・時刻及びアルファベットを含め、最低 15 バイトの大きさが必要です。 "YYYYMMDDhhmmss" YYYY = 年 4 桁, MM = 月 2 桁, DD = 日 2 桁, hh = 時 2 桁, mm = 分 2 桁, ss = 秒 2 桁
コーディング例	get_time (system_time);
戻り値	なし。

#### set\_time

目的	カウンタチップに新しい日付・時刻を設定します。
書式	int set_time (char* new_time);
引数	char* new_time 設定する日付・時刻を格納する変数へのポインタ。 Char 型配列変数は、日付・時刻及びアルファベットを含め、最低 15 バイトの大きさが必要です。 "YYYYMMDDhhmmss" YYYY = 年 4 桁, MM = 月 2 桁, DD = 日 2 桁, hh = 時 2 桁, mm = 分 2 桁, ss = 秒 2 桁
コーディング例	set_time ("20050105125800"); /* JAN 5, 2005 12:58:00 */
戻り値	通常、1 を返しますが、カウンタチップの動作不良などが発生した場合は、0 を返します。
備考	引数がフォーマットエラー(例:時刻を 25 時とした)の場合は、単純に処理が拒否されるのみで、エラー扱いにはなりません。

## 2.8.2 アラーム

8000/8200/8400 シリーズでのみ使用できます。

GetAlarm		8000,8200,8400
目的	電源を自動立ち上げる時刻を読み出します。(アラーム機能)	
書式	void GetAlarm(char *cur_time);	
引数	char* cur_time 取得した日付・時刻を格納する変数へのポインタ。 Char 型配列変数は、日付・時刻及びルータ名を含め、最低 15 バイトの大きさが必要です。 "YYYYMMDDhhmmss" YYYY = 年 4 桁, MM = 月 2 桁, DD = 日 2 桁, hh = 時 2 桁, mm = 分 2 桁, ss = 秒 2 桁	
コーディング例	GetAlarm(time_buf);	
戻り値	無し	

SetAlarm		8000,8200,8400
目的	電源を自動立ち上げる時刻を設定します。(アラーム機能)	
書式	void SetAlarm(const char *time);	
引数	char* cur_time 設定する日付・時刻を格納する変数へのポインタ。 Char 型配列変数は、日付・時刻及びルータ名を含め、最低 15 バイトの大きさが必要です。 "YYYYMMDDhhmmss" YYYY = 年 4 桁, MM = 月 2 桁, DD = 日 2 桁, hh = 時 2 桁, mm = 分 2 桁, ss = 秒 2 桁	
コーディング例	SetAlarm("20050105125800") /* 2005/01/05 12:58:00 に電源を立ち上げる */	
戻り値	無し	
備考	引数がフォーマットエラー(例: 時刻を 25 時とした)の場合は、単純に処理が拒否されるのみで、エラー扱いにはなりません。	



## 2.9 パワーマネージメント

この章では、メインとバックアップのバッテリーの電圧レベルをモニターするためのパワーマネージメント機能について説明します。CipherLab シリーズ ハンデーターミナルは、通常動作を行うための主電池と SRAM メモリ及びリアルタイムクロックをバックアップするための副電池を装備しています。

### 2.9.1 バッテリー電圧

#### get\_vmain

目的	主電池の電圧を取得します。	
書式	int get_vmain(void);	
コーディング例	if (get_vmain( ) < 2200)	// 主電池の電圧をチェック
	puts ("Battery is low");	
戻り値	主電池の電圧(mV)を返します。	

#### get\_vbackup

目的	副電池の電圧を取得します。	
書式	int get_vbackup(void);	
コーディング例	bat1 = get_vbackup();	
戻り値	副電池の電圧(mV)を返します。	

## 2.9.2 充電

### charger\_status

**目的** 主電池の充電ステータスを取得します。

**書式** int charger\_status (void);

**コーディング例** if (charger\_status() == CHARGE\_DONE)  
puts ("Battery is full");

**説明** 主電池の充電ステータスを取得します。

**戻り値** 8000/8300 シリーズ の場合、下記の何れかの値を返します。

0	CHARGE_STANDBY	外部電源に接続されていません。
1	CHARGING	主電池を充電中です。
2	CHARGE_DONE	主電池は満タリです。
3	CHARGE_FAIL	主電池を充電できません。

8200/8400/8700 シリーズ の場合、下記の何れかの値を返します。

0	CHARGE_STANDBY	外部電源に接続されていません。
1	CHARGING_SV	主電池を SV から充電中です。
2	CHARGE_DONE	主電池は満タリです。
3	CHARGE_FAIL	主電池を充電できません。
17	CHARGING_USB	主電池を USB から充電中です。

8500 シリーズ の場合、下記の何れかの値を返します。

0	CHARGING	主電池を充電中です。
1	CHARGE_DONE	主電池は満タリです。
2	CHARGE_FAIL	主電池を充電できません。
3	CHARGE_STANDBY	外部電源に接続されていません。

**参考** GetUSBChargeCurrent, SetUSBChargeCurrent

### GetUSBChargeCurrent

8200,8400

**目的** USB からの充電電流を取得します。

**書式** int GetUSBChargeCurrent(void);

**コーディング例** val = GetUSBChargeCurrent(); // get charging setting

**戻り値** 下記の何れかの値を返します。

0	CURRENT_500mA	500mA で充電。
1	CURRENT_100mA	100mA で充電。
2	CURRENT_0mA	充電無効。

### SetUSBChargeCurrent

8200,8400

**目的** USB への充電電流を設定します。

**書式** void SetUSBChargeCurrent(int current\_type);

**引数** int current\_type

0	CURRENT_500mA	500mA で充電。
1	CURRENT_100mA	100mA で充電。
2	CURRENT_0mA	充電無効。

**コーディング例** SetUSBChargeCurrent(CURRENT\_500mA); // set 500mA for USB setting

**戻り値** なし

## 2.10 キーボード

CipherLab シリーズ ハンディターミナルは、32 バイトのキーボードバッファを備えています。バッファが一杯になると、以降のキーストロークは破棄されてしまいます。C プログラムを開発する際は、キーボードバッファを定期的にチェックするようにコーディングしてください。

### 2.10.1 一般

#### CheckKey

目的	指定のキーが押されたかを検出します。	
書式	int CheckKey(const int scan_code,...);	
引数	検出したいキーを全て引数として列挙します。但し、最後の引数は、検出モードを下記の何れかで指定します。 int current_type	
	-1	CHK_EXC 列挙したキーが同時に押された場合に、戻り値 1 を返します。
	-2	CHK_INK 列挙したキーを含むキーが押された場合に、戻り値 1 を返します。
コーディング例	<pre>while (1) {     if ( CheckKey(SC_1, SC_2, SC_3, CHK_EXC))         printf ("The user presses 1, 2, 3 simultaneously");     OSTimeDly(8);          // delay 8x5 = 40ms }</pre>	
戻り値	正常終了すると、1 を返し、それ以外は、0 を返します。	
備考	引数で列挙されたキーが押されたかを指定モードに従って検出します。通常、この関数は、特殊な目的で、コンビネーションキーを検出したい場合に使用します。システムが全てのキーをスキャンするには約 40msec 要するため、連続して 2 回コールすることは避けてください。もし、プログラム内でコールしている間隔が分からない場合は、上記の例のように OSTimeDly 関数を使って、明示的に 40 ミリ秒の間隔を空けることができます。	
参照	OSTimeDly	

#### clr\_kb

目的	キーボードバッファをクリアします。
書式	void clr_kb(void);
コーディング例	clr_kb();
戻り値	無し
備考	キーボードバッファをクリアします。この関数は、電源を立ち上げた時、システムにより自動的にコールされます。
参照	getchar, kbhit

getchar	
目的	キーボードバッファから 1 文字を取得します。
書式	int getchar(void);
コーディング例	<pre>c = getchar(); if (c &gt; 0 )     printf ("Key %d pressed", c); else     printf ("No key pressed");</pre>
戻り値	正常終了すると、取得した文字を返します。キーボードバッファが空の場合は、\0(0x00)を返します。
備考	キーボードバッファから 1 文字を取得し、そのデータをバッファから削除します。
参照	clr_kb, kbhit, putchar

GetKBDModifierStatus

目的

修飾キー(SHIFT/ALT/FN)の情報を取得します。

書式

unsigned int GetKBDModifierStatus(void);

コーディング例

status = GetKBDModifierStatus();

戻り値

修飾キーの状態を bit 値で返します。

備考

各ビットの意味は次の通りです。

ビット	項目	値
0	電源キー	0=無効, 1=有効
1	FN キー修飾	0=無効, 1=有効
2	FN トグル	0=自動再開モード, 1=トグルモード
3	LCD コントラスト操作 FN + Up/Down (8000/8300/8500/8700) Backlight key + Left/Right (8200/8400)	0=無効, 1=有効
4	SHIFT キー修飾	0=無効, 1=有効
5	通常キーとしての FN キー	0=無効, 1=有効
6	通常キーとしての SHIFT キー	0=無効, 1=有効
7	通常キーとしての ALT キー	0=無効, 1=有効
8	ALT キー修飾	0=無効, 1=有効
9	LCD バックライト操作 FN + Left/Right (8500/8700) Backlight key + Up/Down (8200/8400)	0=無効, 1=有効
10	マルチモード	0=無効, 1=有効
11	バックライトキー	0=無効, 1=有効
12	F9～F20 キーステータス (8400, 29-key のみ)	0=無効, 1=有効

8000/8300 シリーズの場合、初期設定は 9 になっています。

➤ Bit 0 : 電源キー有効

➤ Bit 3 : LCD コントラスト操作有効

8200/8400/8500/8700 シリーズの場合、初期設定は 0x209 になっています。

➤ Bit 0 : 電源キー有効

➤ Bit 3 : LCD コントラスト操作有効

➤ Bit 9 : LCD バックライト操作有効

参照

get\_shift\_lock\_state, GetAltKeyState, GetFuncExtKey, GetFuncToggle, set\_shift\_lock, SetAltKey, SetFuncExtKey, SetFuncToggle, SetPwrKey

GetKeyClick	
目的	キークリック音の設定を取得します。
書式	int GetKeyClick(void);
コーディング例	state = GetKeyClick();
戻り値	0 =無効(キークリック音なし) 1~5 =キークリックトーン 1~5
備考	キークリック音は、デフォルトでは、有効に設定されていますが、システムメニュー又はプログラマシグで、設定を変更することができます。
参照	SetKeyClick

kbhit	
目的	キーボードバッファにデータがあるかをチェックします。
書式	int kbhit(void);
コーディング例	for ( ;!kbhit(); ); // キーが押下されるまで待ちます
戻り値	キーボードバッファが空の場合は、0 を返します。何かキーが押下され、キーボードバッファ内にデータがある場合は、1 を返します。
参照	SetKeyClick

putch		8200,8400,8500,8700
目的	キーボードバッファに 1 文字出力します。	
書式	void putch(unsigned char c);	
引数	unsigned char c キーボードバッファに出力する文字。	
コーディング例	putch(KEY_ESC); // put ESC key value to keyboard buffer	
戻り値	成功すると、キーボードバッファから読まれた文字を返します。そうでなければ、バッファが空であることを示す null 文字(0x00)を返します。	
参照	clr_kb, getchar	

SetKeyClick					
目的	キークリック音を設定します。				
書式	void SetKeyClick(int status);				
引数	int status				
	<table border="1"> <tr> <td>0</td><td>無効(キークリック音なし)</td></tr> <tr> <td>1~5</td><td>キークリックトーン 1~5</td></tr> </table>	0	無効(キークリック音なし)	1~5	キークリックトーン 1~5
0	無効(キークリック音なし)				
1~5	キークリックトーン 1~5				
コーディング例	SetKeyClick(1); // キークリック音あり				
戻り値	無し				
備考	キークリック音は、デフォルトでは、有効に設定されていますが、システムメニュー又はプログラマシグで、設定を変更することができます。また、システムグローバル変数 KEY_CLICK にキークリック音の周波数及び間隔がセットされているため、on_beeper 関数を使って、キークリック音と同じ音を鳴らすことが可能です。 on_beeper (KEY_CLICK);				
参照	GetKeyClick, KEY_CLICK				

## TriggerStatus

目的	読取りトリガボタン(スキャンキー)が押下されているかをチェックします。
書式	int TriggerStatus (void);
コーディング例	<pre>if ( TriggerStatus() )     printf ("Scan key is pressed");</pre>
戻り値	1 = 押下されている 0 = 押下されていない

## 2.10.2 ALPHA キー

### dis\_alpha

目的	ALPHA キーを無効にします。
書式	void dis_alpha(void);
コーディング例	dis_alpha();
戻り値	無し
備考	無効に設定した場合、ALPHA キーを押してもALPHA 入力モードへの切り替えはできなくなります。 ➤ LockAlphaState(0)と同じ結果になります。

### en\_alpha

目的	ALPHA キーを有効またはロック解除します。 (1) 8000/8200/8500/8700 シリーズ：引数は ALPHA_ROLLING のみ (2) 8300 シリーズ, 24-key：引数は ALPHA_ROLLING のみ (3) 8300 シリーズ, 39-key：引数は ALPHA_FIXED または ALPHA_ROLLING (4) 8400 シリーズ, 24-key：引数は ALPHA_ROLLING のみ (5) 8400 シリーズ, 39-key：引数は ALPHA_FIXED のみ		
書式	void en_alpha(int type);		
引数	int type		
	1	ALPHA_FIXED	キー入力 1 に対して 1 文字表示されます。表示される文字は入力モードによります。
	2	ALPHA_ROLLING	同じキーを 1 秒以内に押すことにより、ALPHA 入力モードと数字を順番に表示します。例えば、"2ABC"キーを 1 秒以内に続けて押すと、A→B→C→2 と順番に表示されます。  8300 シリーズ, 39-key の場合 同じキーを 1 秒以内に押すことにより、ALPHA 入力モードと数字を順番に表示します。例えば、"2B"キーを 1 秒以内に続けて押すと、B→2 と順番に表示されます。
コーディング例	en_alpha();		
戻り値	無し		
備考	デフォルトで、入力モードは数値で、ALPHA キーによって変更することができます。 ➤ ALPHA キーは dis_alpha() で無効に設定し、この処理で有効に設定します。 ➤ ALPHA キーは LockAlphaState() でロックし、この処理でロック解除します。		

### get\_alpha\_enable\_state

目的

現在のALPHAキー状態を取得します。

書式

int get\_alpha\_enable\_state(void);

コーディング例

state = get\_alpha\_enable\_state();

戻り値

下記の何れかの値を返します。

-1	8500,44-key(Type I)ですべてのALPHAキーが無効。
0	ALPHAキー-無効。
1	ALPHAキー-有効(ALPHA_FIXED)。
2	ALPHAキー-有効(ALPHA_ROLLING)。

備考

デフォルトで、ALPHAキーは有効に設定されています。

## get\_alpha\_lock\_state

**目的** アルファキーロック状態を取得します。

**書式** int get\_alpha\_lock\_state(void);

**コーディング例** state = get\_alpha\_lock\_state();

**戻り値** 下記の何れかの値を返します。

-1	8500,44-key(Type I)ですべてのアルファキーが無効。
0	数値のみ。
1	アルファベット大文字。
2	アルファベット小文字。
3	ファンクションモード。

**備考** デフォルトで、アルファキーはアロックに設定されています。

## LockAlphaState

**目的** 文字入力を指定のモードに変更し、アルファキーをロック状態にします。

**書式** void LockAlphaState(int state);

**引数** int state

1	NUMERIC_KEYPAD	数値入力固定。
2	UPPER_CASE	アルファベット大文字入力固定。
3	LOWER_CASE	アルファベット小文字入力固定。
4	FUNCTION_KEY	ファンクションモード固定。(8000,8200のみ)

**コーディング例** LockAlphaState(2); // lower case alpha mode, ALPHA key disabled

**戻り値** 無し

**備考** 入力を指定のモードに固定します。アルファキーを押してもモードは変更されません。

## set\_alpha\_lock

**目的** 文字入力を指定のモードに変更し、アルファキーをロック解除にします。

**書式** void set\_alpha\_lock (int state);

**引数** int state

1	数値入力。
2	アルファベット大文字入力。
3	アルファベット小文字入力。
4	ファンクションモード。(8000,8200のみ)

**コーディング例** set\_alpha\_lock(1); // upper case alpha mode, ALPHA key enabled

**戻り値** 無し

**備考** 入力を指定のモードに設定します。アルファキーを押すとモードは変更されます。  
➤ ALPHAキーがdis\_alpha()で無効、またはLockAlphaState()でロックされている場合、en\_alpha()で、有効(ロック解除)してください。



## 2.10.3 SHIFT キー

SHIFTキーはアルファベットを大文字から小文字に変換する修飾キーです。

※ SHIFTキーは 8500,44-key(Type I)でのみ使用できます。

<b>get_shift_lock_state</b>	<b>8500</b>
-----------------------------	-------------

目的	SHIFTキー状態を取得します。
書式	int get_shift_lock_state(void);
コーディング例	state = get_shift_lock_state();
戻り値	0〜3 が返ります。ただし、8500 シリーズの 24-key や 44-key(Type II)にはSHIFTキーがないため、-1 が返ります。

<b>set_shift_lock</b>	<b>8500</b>
-----------------------	-------------

目的	SHIFTキー状態を変更します。								
書式	void set_shift_lock(int state);								
引数	int state								
	<table><tr><td>0</td><td>SHIFTキー変更不可。(デフォルト)</td></tr><tr><td>1</td><td>SHIFTキー変更可。</td></tr><tr><td>2</td><td>SHIFTキー変更不可 +通常キー扱い。</td></tr><tr><td>3</td><td>SHIFTキー変更可 +通常キー扱い。</td></tr></table>	0	SHIFTキー変更不可。(デフォルト)	1	SHIFTキー変更可。	2	SHIFTキー変更不可 +通常キー扱い。	3	SHIFTキー変更可 +通常キー扱い。
0	SHIFTキー変更不可。(デフォルト)								
1	SHIFTキー変更可。								
2	SHIFTキー変更不可 +通常キー扱い。								
3	SHIFTキー変更可 +通常キー扱い。								
コーディング例	set_shift_lock(0); // no SHIFT modification								
戻り値	無し								
備考	この関数は、SHIFTキーで変更できるSHIFTキー状態を設定します。								

## 2.10.4 ALT キー

Altキーは修飾キーとして使用します。

Altキーは 8500,44-key(Type I)または 8500/8700,44-TE(Type II)でのみ使用できます。

<b>GetAltKeyState</b>	<b>8500,8700</b>
-----------------------	------------------

目的	Altキー状態を取得します。
書式	int GetAltKeyState(void);
コーディング例	state = GetAltKeyState();
戻り値	0〜3 が返ります。ただし、8500/8700 シリーズの 24-key にはAltキーがないため、-1 が返ります。

<b>SetAltKey</b>	<b>8500,8700</b>
------------------	------------------

目的	Altキー状態を変更します。								
書式	void SetAltKey(int state);								
引数	int state								
	<table><tr><td>0</td><td>Altキー変更不可。(デフォルト)</td></tr><tr><td>1</td><td>Altキー変更可。</td></tr><tr><td>2</td><td>Altキー変更不可 + 通常キー扱い。</td></tr><tr><td>3</td><td>Altキー変更可 + 通常キー扱い。</td></tr></table>	0	Altキー変更不可。(デフォルト)	1	Altキー変更可。	2	Altキー変更不可 + 通常キー扱い。	3	Altキー変更可 + 通常キー扱い。
0	Altキー変更不可。(デフォルト)								
1	Altキー変更可。								
2	Altキー変更不可 + 通常キー扱い。								
3	Altキー変更可 + 通常キー扱い。								
コーディング例	SetAltKey(0); // no ALT modification								
戻り値	無し								
備考	この関数は、ALT キーで変更できる ALT キー状態を設定します。								

## 2.10.5 ファンクションキー

ファンクション(FN)キーは組み合わせて使うことで修飾キーとしてします。

(1) この修飾キーを使用する場合、キーパッド上でファンクション(FN)キーを押してください。すると、ステータス・アイコン「F」が画面上に表示されます。

(2) ファンクションキーが SetFuncToggle()でオートレジュームモードに設定されている場合、ファンクションキーと組み合わせとなるキーが押されると、ステータス・アイコンは画面から消去されます。つまり、この修飾キーは1操作分のみです。

(3) 他のファンクションキー動作を行う場合は、上記操作を繰り返してください。

ただし、ファンクション(FN)キーが SetFuncToggle()によってトグルモードに設定されている場合、この修飾キーは再度ファンクション(FN)キーが押されるまで有効となります。

<b>GetFuncToggle</b>	<b>8300,8400,8500,8700</b>
----------------------	----------------------------

**目的** ファンクション(FN)キーのトグル状態を取得します。

**書式** int GetFuncToggle(void);

**コーディング例** state = GetFuncToggle();

**戻り値** (1) 8300 シリーズでは 0～1 が返ります。  
 (2) 8400 シリーズでは 0～4、6 が返ります。  
 (3) 8500/8700 シリーズでは、以下の値が返ります。  
 ➤ 24-key または 44-key, Type I の場合、0～3  
 ➤ 44-key, Type II の場合、0～4、6

<b>SetFuncToggle</b>	<b>8300,8400,8500,8700</b>
----------------------	----------------------------

**目的** ファンクション(FN)キーのトグル状態を設定します。

**書式** void SetFuncToggle(int status);

**引数** int state

8300 シリーズ, 24-key、39-key

0	オートレジュームモード + マルチモード (デフォルト)
1	トグルモード + マルチモード

(1)8400 シリーズ, 24-key、39-key (2)8500/8700 シリーズ, key Type II

0	オートレジュームモード + マルチモード (デフォルト)
1	トグルモード + マルチモード
2	オートレジュームモード + マルチモード + 通常キー扱い
3	トグルモード + マルチモード + 通常キー扱い
4	マルチモード
6	マルチモード + 通常キー扱い

8500 シリーズ, 24-key、44-key Type I

0	オートレジュームモード + マルチモード (デフォルト)
1	トグルモード + マルチモード
2	オートレジュームモード + マルチモード + 通常キー扱い
3	トグルモード + マルチモード + 通常キー扱い
4	効果なし

➤ オートレジュームモード：ファンクションキーを押すことでファンクションモードとなり、組み合わせとなるキーを押すことでモードが終了します。ステータス・アイコンが画面上に表示されます。8300/8400/8700 シリーズでは、ファンクションモード中に再度ファンクションキーを押すことで、モードを終了させることができます。

➤ トグルモード：ファンクションキーを押すことでファンクションモードとなり、再度ファンクションキーを押すまでこのモードが継続します。ステータス・アイコンが画面上に表示されます。

➤ マルチモード：ファンクションキーと組み合わせとなるキーを同時に押す、またはファンクションキーを押し、そのまま組み合わせとなるキーを押します。

**コーディング例** state = SetFuncToggle(0); // set the FN state to Auto Resume and Multi-key mode

**戻り値** なし

## 拡張ファンクションキー

8400シリーズ, 24-key では、デフォルトで F1～F8 が利用可能ですが、SetFunctionKey(1)をコールすることで、F9～F20 を使用することができるようになります。

<b>GetFuncExKey</b>	<b>8400</b>
---------------------	-------------

**目的** 拡張ファンクション(FN)キー F9～F20 が有効かどうかチェックします。

**書式** int GetFuncExKey(void);

**コーディング例** state = GetFuncExKey();

**戻り値** 有効なら 1、無効なら 0 が返ります。

<b>SetFuncExKey</b>	<b>8400</b>
---------------------	-------------

**目的** 拡張ファンクション(FN)キー F9～F20 の状態を設定します。

**書式** void SetFuncExKey(int status);

**引数** int state

0	拡張ファンクション(FN)キー F9～F20 無効
1	拡張ファンクション(FN)キー F9～F20 有効

**コーディング例** state = SetFuncExKey(1); // enable key combination F9-F20

**戻り値** なし

**備考** F9～F20 の組み合わせは次の通りです。

キーの組み合わせ	結果
FN + [-]	F9
FN + [.]	F10
FN + [1]	F11
FN + [2]	F12
FN + [3]	F13
FN + [4]	F14
FN + [5]	F15
FN + [6]	F16
FN + [7]	F17
FN + [8]	F18
FN + [9]	F19
FN + [0]	F20

**参照** SetFuncToggle

## 2.10.6 ENTER キー

8200 シリーズにある黄色のキーはデフォルトでEnter(改行)キーとしても機能します。使用すると中Enter(改行)キーとして認識されます。

- InitScanner1()をコールした後は、黄色のキーはスキャンキーとして機能します。
- HaltScanner1()をコールした後は、黄色のキーはEnter(改行)キーとして機能します。

<b>CheckKeyEnter</b>	<b>8200</b>
----------------------	-------------

**目的** ハンデタイミルのEnter(改行)キーが押されているかどうかチェックします。

**書式** unsigned char CheckKeyEnter(void);

**コーディング例**

```
c = getchar();
if (c == KEY_CR) {
    type = CheckKeyEnter();
    if (type == 1) {
        printf("right enter");
    } else if (type == 2) {
        printf("left enter");
    } else if (type == 3) {
        printf("middle enter");
    }
}
```

**戻り値** 下記の何れかの値を返します。

0	Enter(改行)キーは押されていない。
1	右Enter(改行)キーが押されている。
2	左Enter(改行)キーが押されている。
3	中Enter(改行)キーが押されている。

**備考** この関数は getchar()の後にコールすることをお勧めします。

<b>SetMiddleEnter</b>	<b>8200</b>
-----------------------	-------------

**目的** 黄色のスキャンキーをEnter(改行)キーとして使用できるかどうかを設定します。

**書式** void SetMiddleEnter(int status);

**引数** int state

0	中Enter(改行)キー無効
1	中Enter(改行)キー有効

**コーディング例** state = SetMiddleEnter(1);

**戻り値** なし

## 2.11 LCD

CipherLab シリーズ ハンディターミナルは、本体に FSTN グラフィックディスプレイを装備しています。ディスプレイの性能は LCD パネルのサイズによります。カーソルの行列(x, y)を指定するために、座標系はカーソル移動処理に使用されます。左上の座標は(0,0)です。一方、右下の座標は、LCD とフォントのサイズによって異なります。グラフィック表示では、ピクセル単位で座標の指定を行います。

ハンディターミナル	画面サイズ	左上座標	右下座標
8000	100 × 64 ドット	(0, 0)	(99, 63)
8300	128 × 64 ドット	(0, 0)	(127, 63)
8200,8400	160 × 160 ドット	(0, 0)	(159, 159)
8500,8700	160 × 160 ドット	(0, 0)	(159, 159)

### 2.11.1 フォント

➤ コントラスト：0～7。デフォルトは 4。

➤ バックライト：デフォルトは消灯です。8200/8400 シリーズでは、バックライトボタンの使用します。それ以外のハンディターミナルでは FN+Enter キーを使用します。

※ 8200/8400/8500/8700 シリーズでは、バックライト点灯時の明るさは 2 に設定されています。

#### DecContrast

目的	LCD コントラストレベルを 1 段階下げます。
書式	void DecContrast(void);
コーディング例	DecContrast();
戻り値	なし
備考	この関数を 1 回コールすると LCD コントラストは 1 段階下がります。コントラストの最小値は 0 です。

#### IncContrast

目的	LCD コントラストレベルを 1 段階上げます。
書式	void IncContrast(void);
コーディング例	IncContrast();
戻り値	なし
備考	この関数を 1 回コールすると LCD コントラストは 1 段階上がります。コントラストの最大値は 7 です。
参照	GetContrast, SetContrast, SetContrastControl

#### GetContrast

目的	LCD コントラストレベルを取得します。
書式	int GetContrast(void);
コーディング例	int nContrastLevel = GetContrast();
戻り値	LCD コントラストレベルを 0～7 の範囲で返します。
備考	現在の LCD コントラストレベルを取得します。デフォルトは 4 です。

## SetContrast

目的	LCD コントラストレベルを設定します。
書式	void SetContrast(int level);
コーディング例	SetContrast(4);
戻り値	無し
備考	引数で指定されたレベルに LCD コントラストをセットします。設定可能な範囲は、0(暗い)~7(明るい)です。
参照	DecContrast, IncContrast, SetContrastControl

## GetVideoMode

目的

LCD 表示モードを取得します。

書式

int GetVideoMode(void);

コーディング例

if (GetVideoMode( ) == VIDEO\_NORMAL)  
puts("Normal Mode");

戻り値

下記の何れかの値を返します。

0	VIDEO_NORMAL	通常表示
1	VIDEO_REVERSE	反転表示

備考

現在の LCD 表示モードを取得します。

## SetVideoMode

目的

LCD 表示モードを設定します。

書式

void SetVideoMode(int mode);

引数

int mode

0	VIDEO_NORMAL	通常表示
1	VIDEO_REVERSE	反転表示

コーディング例

SetVideoMode(VIDEO\_REVERSE);

// set reverse video mode

戻り値

無し

備考

LCD 表示モードを設定します。

lcd_backlit	
-------------	--

**目的** LCDバックライトを設定します。

書式      void lcd\_backlit(int state);

引数	int state
----	-----------

8000/8300 シー-ス で使用する引数は次の通りです

0	BKLT_OFF	バックライト OFF
1	BKLT_ON	バックライト ON

8200/8400 シー・エス・で使用する引数は次の通りです

0x0000	BKLT_OFF	バックライト OFF
0x0001	BKLT_VERY_LO	バックライト明るさ最低
0x0002	BKLT_LO	バックライト明るさ低
0x0003	BKLT_MED	バックライト明るさ中
0x0004	BKLT_HI	バックライト明るさ高
0x0010	BKLT_SHADE_OFF	バックライト陰影効果なし
0x0011	BKLT_SHADE_VL	バックライト陰影効果最低
0x0012	BKLT_SHADE_LO	バックライト陰影効果低
0x0013	BKLT_SHADE_MED	バックライト陰影効果中
0x0014	BKLT_SHADE_HI	バックライト陰影効果高

8500/8700 シリ-ズ で使用する引数は次の通りです

0	BKLT_OFF	バックライト OFF
1	BKLT_VERY_LO	バックライト明るさ最低
2	BKLT_LO	バックライト明るさ低
3	BKLT_MED	バックライト明るさ中
4	BKLT_HI	バックライト明るさ高

```
コード例    lcd_backlit(1);           // turn on LCD backlight, low density
```

戻り値 無し

**備考** LCD バックライトを OFF 又は指定の明るさに設定します。

➤ LCD バックライトを ON に設定した場合、システムグローバル変数 BKLIT\_TIMEOUT で指定された秒数、何も操作が行われないとバックライトは自動的に OFF に戻ります。但し、この値を 0 にセットしている場合は、ユーザーが手動で ON にするか、引数に 0 を指定して、lcd\_backlit 関数をコールするまで、バックライトは OFF に戻りません。

**参照** [BKLT\\_TIMEOUT](#), [SetBklitControl](#)



<b>SetBklitControl</b>	<b>8200,8400,8500,8700</b>
------------------------	----------------------------

**目的** キー組み合わせ操作で LCD バックライトを設定できるようにするかどうかを設定します。

**書式** void SetBklitControl(int mode);

**引数** int mode

8200/8400 シリーズで使用する引数は次の通りです

0	[バックライト] + [↑]/[↓]無効
1	[バックライト] + [↑]/[↓]有効
2	[バックライト] + [↑]/[↓]無効、[バックライト]キーを通常キー扱い
3	[バックライト] + [↑]/[↓]有効、[バックライト]キーを通常キー扱い

8500/8700 シリーズで使用する引数は次の通りです

0	FN + [←]/[→]無効
1	FN + [←]/[→]有効

**コーディング例** SetBklitControl(0); // disable the key combination for Backlight Control

**戻り値** 無し

**備考** キー組み合わせで LCD バックライトを設定できるようにするかどうかを設定します。

- 8200/8400 シリーズでキー組み合わせ操作が有効の場合、バックライトは[バックライト] + [↑]で明るくなり、[バックライト] + [↓]で暗くなります。
- 8200/8400 シリーズでキー組み合わせ操作が無効の場合、キーボードバックファに KEY\_BUP、KEY\_BDOWN が格納されます。
- 8200/8400 シリーズでのバックライトキーを通常キー扱い - バックライトキーは通常キーとして扱われます。
- 8500/8700 シリーズでキー組み合わせ操作が有効の場合、バックライトは FN + [→]で明るくなり、FN + [←]で暗くなります。
- 8500/8700 シリーズでキー組み合わせ操作が無効の場合、キーボードバックファに KEY\_FLEFT、KEY\_FRIGHT が格納されます。

**参照** lcd\_backlit

## SetContrastControl

**目的** キー組み合わせ操作で LCD コントラストを設定できるようにするかどうかを設定します。

**書式** void SetContrastControl(int mode);

**引数** int mode

8000/8300/8500/8700 シリーズで使用する引数は次の通りです

0	FN + [↑]/[↓]無効 (8500/8700 44-TE key は FN + [3]/[6]無効)
1	FN + [↑]/[↓]有効 (8500/8700 44-TE key は FN + [3]/[6]有効)

8200/8400 シリーズで使用する引数は次の通りです

0	[バックライト] + [↑]/[↓]無効 (39-key は FN + [0]/[.]も無効)
1	[バックライト] + [↑]/[↓]有効 (39-key は FN + [0]/[.]も有効)

**コーディング例** SetContrastControl(0); // disable the key combination for Contrast Control

**戻り値** 無し

**備考** キー組み合わせで LCD コントラストを設定できるようにするかどうかを設定します。

- 800/8300/8500/8700 シリーズでキー組み合わせ操作が有効の場合、コントラストは FN + [↑]で高くなり、FN + [↓]で低くなります。
- 800/8300/8500/8700 シリーズでキー組み合わせ操作が無効の場合、キーボードバッファに KEY\_FUP、KEY\_FDOWN が格納されます。
- 8200/8400 シリーズでキー組み合わせ操作が有効の場合、コントラストは[バックライト] + [→]で高くなり、[バックライト] + [←]で低くなります。
- 8200/8400 シリーズでキー組み合わせ操作が無効の場合、キーボードバッファに KEY\_BLEFT、KEY\_BRIGHT が格納されます。

**参照** DecContrast, GetContrast, IncContrast, SetContrast,

## 2.11.2 カーソル

### GetCursor

目的	カーソル表示/非表示を取得します。
書式	int GetCursor(void);
コーディング例	if (GetCursor() == 0) puts ("Cursor Off");
戻り値	カーソル表示がONの場合は 1 を返し、OFFの場合は 0 を返します。

### SetCursor

目的	カーソル表示を切り切します。	
書式	void SetCursor(int status);	
引数	int status	
	0	CURSOR_OFF      カーソル非表示
	1	CURSOR_ON        カーソル表示
コーディング例	SetCursor(0);	// trueに off the cursor indication
戻り値	無し	

### gotoxy

目的	指定位置へカーソルを移動します。
書式	void gotoxy(int x_position, int y_position);
引数	int x_position カーソルを移動したい X 座標
	int y_position カーソルを移動したい Y 座標
コーディング例	gotoxy(10, 0); // 1 行目の 11 桁目へカーソルを移動
戻り値	なし
備考	<p>引数で指定された位置へカーソルを移動します。移動できる範囲には以下の設定の制約があります。</p> <ul style="list-style-type: none"> <li>➢ ICON_ZONE() で指定されたアイコン表示エリア。</li> <li>➢ LCD のサイズ。</li> <li>➢ 使用しているフォントファイル。</li> </ul> <p>8500/8700 シリーズで、フォントサイズが 6×8 で ICON_ZONE(0) の場合、Y 座標が 18 以上はありません。</p>
参照	wherexy

### wherex

目的	現在のカーソル位置(X 座標)を取得します。
書式	int wherex(void);
コーディング例	x_position = wherex();
戻り値	現在のカーソル位置(X 座標)を返します。

wherexy	
目的	現在のカーソル位置を取得します。
書式	int wherexy (int* column, int* row);
引数	int* column
	X 座標値を格納する変数へのポインタ
	int* row
	Y 座標値を格納する変数へのポインタ
コーディング例	wherexy (&x_position, &y_position);
戻り値	無し
備考	現在のカーソル位置座標を引数で与えられた変数に格納します。

wherey	
目的	現在のカーソル位置(Y 座標)を取得します。
書式	int wherey(void);
コーディング例	y_position = wherey( );
戻り値	現在のカーソル位置(Y 座標)を返します。

## 2.11.3 表示

fill_rect	
目的	指定の長方形エリアを塗りつぶします。
書式	void fill_rect (int left, int top, int width, int height);
引数	int left
	長方形エリアの左上 X 座標
	int top
	長方形エリアの左上 Y 座標
	int width
	長方形エリアの幅をドット単位で指定
	int height
	長方形エリアの高さをドット単位で指定
コーディング例	fill_rect (12, 8, 40, 8);
戻り値	無し
備考	引数で指定された長方形エリアを塗りつぶします。 ➤ この処理を実行しても、カーソル位置には、影響しません。
参照	clr_rect

ICON_ZONE		
目的	アイコン表示エリアを設定します。	
書式	void ICON_ZONE(int mode);	
引数	int mode	
	0	ICON_ZONE_DISABLE    アイコン表示エリアとして使用(デフォルト)
	1	ICON_ZONE_ENABLE    1-ザ-表示エリアとして使用
コーディング例	ICON_ZONE(1);	
戻り値	無し	
備考	アイコン表示エリアは、通常、バッテリーステータスアイコン等を表示するための予約領域です。	
	8000	100 × 64 ドット    アイコン表示エリアは右端の 4×64 ドットを占めています。4 ドットでは、1 文字を表現できないため、ICON_ZONE(1)を設定しても表示可能桁数は増えません。
	8200,8400	160 × 160 ドット    アイコン表示エリアは下端の 160×160 ドットを占めています。ICON_ZONE(1)を設定すると、6×8 フォントでは 20 行×26 桁、8×16 フォントでは 10 行×20 桁まで表示できるようになります。
	8300	128 × 64 ドット    アイコン表示エリアは右端の 8×64 ドットを占めています。ICON_ZONE(1)を設定すると、6×8 フォントでは 8 行×21 桁、8×16 フォントでは 4 行×16 桁まで表示できるようになります。
	8500,8700	160 × 160 ドット    アイコン表示エリアは 6×8 フォントでは下端の 160×8 ドットを、8×16 フォントでは下端の 160×16 ドットを占めています。ICON_ZONE(1)を設定すると、6×8 フォントでは 20 行×26 桁、8×16 フォントでは 10 行×20 桁まで表示できるようになります。

上記ディスプレイのいずれでも、ICON\_ZONE(1)を設定した場合、clr\_scr()をコールすると、LCDディスプレイ全体がクリアされます。ステータス・アイコンは ICON\_ZONE(1)を設定した場合でも表示されます。

## printf

目的	文字列や変数値を指定の書式に従って、表示します。
書式	int printf (char* format, var);
引数	char* format 表示書式 var... LCD に表示する変数リスト
コーディング例	printf ("ID : %s", id_buffer);
戻り値	実際に表示した文字数を返します。
説明	引数で指定された文字列や変数値を指定の書式に従って、表示します。使用できる書式は、下記の通りです。 %[flags][width].[precision][size][type]

フィールド	説明																
% (必須)	書式定義の開始を意味します。																
flags (オプション)	<div> <p>プラスマイナス符号や文字揃えを指示する1つ又は複数のフラグキャラクター(' ', '+', '#', '-')を指定します。</p> <table> <tr> <td>-</td><td>左詰めフィールド (デフォルトは、右詰め)を意味します。</td></tr> <tr> <td>+</td><td>符号付の値にプラスマイナス符号を付けます。マイナス値の場合は、このオプションに関係なく、マイナス符号が表示されます。</td></tr> <tr> <td>blank</td><td>プラス値はスペース、マイナス値はマイナス符号を値の前に表示します。但し、先の '+' オプションが指定されている場合は、無視されます。</td></tr> <tr> <td>#</td><td>表示する値のタイプが、0 以外の o, x, X の場合に、それぞれ前に 0, 0x, 0X を表示します。</td></tr> </table> </div>	-	左詰めフィールド (デフォルトは、右詰め)を意味します。	+	符号付の値にプラスマイナス符号を付けます。マイナス値の場合は、このオプションに関係なく、マイナス符号が表示されます。	blank	プラス値はスペース、マイナス値はマイナス符号を値の前に表示します。但し、先の '+' オプションが指定されている場合は、無視されます。	#	表示する値のタイプが、0 以外の o, x, X の場合に、それぞれ前に 0, 0x, 0X を表示します。								
-	左詰めフィールド (デフォルトは、右詰め)を意味します。																
+	符号付の値にプラスマイナス符号を付けます。マイナス値の場合は、このオプションに関係なく、マイナス符号が表示されます。																
blank	プラス値はスペース、マイナス値はマイナス符号を値の前に表示します。但し、先の '+' オプションが指定されている場合は、無視されます。																
#	表示する値のタイプが、0 以外の o, x, X の場合に、それぞれ前に 0, 0x, 0X を表示します。																
width (オプション)	表示文字数(フィールド幅)を指定します。																
precision (オプション)	表示する数字の桁数を指定します。																
size (オプション)	short 型及び long 型を区別する長さ修飾子 'h', 'l', 'L' を指定します。'h' は、short 整数を示し、'l' と 'L' は long 型を示します。																
type (必須)	<div> <p>表示しようとする変数タイプを指定します。</p> <table> <tr> <td>c</td><td>シングルキャラクター</td></tr> <tr> <td>d</td><td>符号有り 10 進数</td></tr> <tr> <td>i</td><td>符号有り 10 進数</td></tr> <tr> <td>o</td><td>符号無し 8 進数</td></tr> <tr> <td>u</td><td>符号無し 10 進数</td></tr> <tr> <td>x</td><td>16 進数(小文字 0~9, a~f)</td></tr> <tr> <td>X</td><td>16 進数(大文字 0~9, A~F)</td></tr> <tr> <td>s</td><td>文字列</td></tr> </table> </div>	c	シングルキャラクター	d	符号有り 10 進数	i	符号有り 10 進数	o	符号無し 8 進数	u	符号無し 10 進数	x	16 進数(小文字 0~9, a~f)	X	16 進数(大文字 0~9, A~F)	s	文字列
c	シングルキャラクター																
d	符号有り 10 進数																
i	符号有り 10 進数																
o	符号無し 8 進数																
u	符号無し 10 進数																
x	16 進数(小文字 0~9, a~f)																
X	16 進数(大文字 0~9, A~F)																
s	文字列																

## putchar

目的	1 文字表示します。
書式	int putchar(char c);
引数	char c LCD に表示する文字
コーディング例	putchar('A');
戻り値	常に 1 が戻ります。
備考	引数で指定された文字(キャラクター)を現在のカーソル位置に表示します。処理後、カーソル位置は 1 文字分移動します。

puts	
目的	文字列を表示します。
書式	char puts (char* string);
引数	char* string LCD に表示する文字列
コーディング例	puts ("Password : ");
戻り値	実際に表示した文字数を返します。
説明	引数で指定された文字列を現在のカーソル位置から表示します。処理後、カーソル位置は表示文字数分移動します。

WaitHourglass		
目的	回転する砂時計を表示します。	
書式	int putchar(int UppLeftX, int UppLeftY, int type);	
引数	int UppLeftX, int UppLeftY	
	砂時計を表示する位置の左上の X,Y 座標	
	Int type	
	1	HOURGLASS_24x23 24×23ピクセルサイズ
	2	HOURGLASS_8x8 8×8ピクセルサイズ
コーディング例	<pre>while(IsRunning) {     . . .     WaitHourglass(68, 68, HOURGLASS_24x23);                                 // show the 24 x 23 hourglass during the loop     . . . }</pre>	
戻り値	なし	
備考	砂時計を回転させるためには、この関数を繰り返し続けてください。 ➤ 一定間隔で、異なる 5 種類の砂時計を表示して時間の経過を表します。	

## 2.11.4 画面消去

### clr\_eol

目的	カーソル位置から行の最後までをクリアし、カーソルを元の位置に戻します。
書式	void clr_eol(void);
コーディング例	clr_eol( );
戻り値	無し
参照	clr_scr

### clr\_icon

目的	アイコン表示エリアをクリアします。
書式	void clr_icon (void);
コーディング例	clr_icon();
戻り値	無し
備考	<p>アイコン表示エリアをクリアします。アイコン表示エリアは、バッテリーステータスアイコン等を表示するための予約領域です。</p> <p>➤ show_image 関数を利用することで、ユーザーは独自のアイコンをここへ表示することができます。</p> <p>➤ clr_scr 関数は、アイコン表示エリアをクリアしないため、アイコン表示エリアをクリアしたい場合は、clr_icon 関数をコールする必要があります。</p>
参照	clr_scr

### clr\_rect

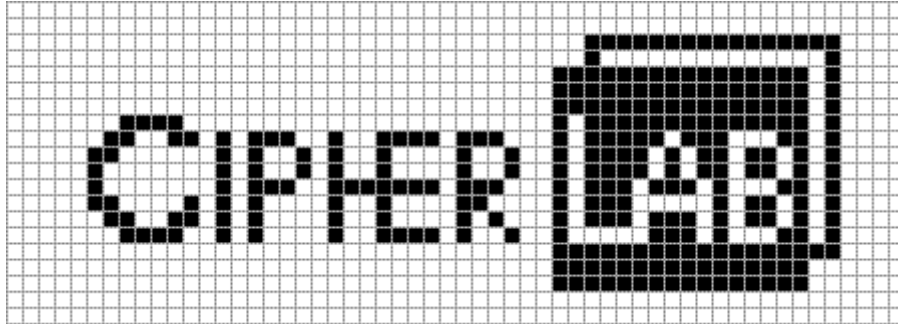
目的	指定の長方形エリアをクリアします。								
書式	void clr_rect (int left, int top, int width, int height);								
引数	<table><tr><td>int left</td></tr><tr><td>長方形エリアの左上 X 座標</td></tr><tr><td>int top</td></tr><tr><td>長方形エリアの左上 Y 座標</td></tr><tr><td>int width</td></tr><tr><td>長方形エリアの幅をドット単位で指定</td></tr><tr><td>int height</td></tr><tr><td>長方形エリアの高さをドット単位で指定</td></tr></table>	int left	長方形エリアの左上 X 座標	int top	長方形エリアの左上 Y 座標	int width	長方形エリアの幅をドット単位で指定	int height	長方形エリアの高さをドット単位で指定
int left									
長方形エリアの左上 X 座標									
int top									
長方形エリアの左上 Y 座標									
int width									
長方形エリアの幅をドット単位で指定									
int height									
長方形エリアの高さをドット単位で指定									
コーディング例	clr_rect (12, 8, 40, 8 );								
戻り値	無し								
備考	引数で指定された長方形エリアをクリアします。 ➤ この処理を実行しても、カーソル位置には、影響しません。								
参照	fill_rect								



clr_scr	
目的	LCDディスプレイをクリアします。
書式	void clr_scr(void);
コーディング例	clr_scr();
戻り値	無し
説明	LCDディスプレイをクリアします。処理実行後、カーソルは、(0,0) 座標へリセットされます。
参照	clr_eof, clr_icon, clr_rect

### 2.11.5 イメージ

show\_image 関数は、グラフィックディスプレイ上に町などのビットマップイメージを表示する場合に利用します。1ビットに1ピクセルを対応させ、ビットマップイメージデータを上段から順に unsigned char 型配列変数にセットします。各ビットは、LSB から順に各ピクセルに対応し、ビットを 1 にセットした場合は、その座標がマークされ、0 にセットした場合は、マークされません。例えば、CipherLab の町をビットマップイメージとして定義する場合は、下記のように定義します。



```
static unsigned char CipherLab_logo[] = { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xf0, 0xff, 0x0f, 0x00, 0x00, 0x00, 0x00, 0x10, 0x00, 0x08, 0x00, 0x00, 0x00, 0x00, 0xfc, 0xff, 0x0b, 0x00, 0x00, 0x00, 0x00, 0xfc, 0xff, 0x0b, 0x00, 0x00, 0x00, 0x00, 0xfc, 0xff, 0x0b, 0x80, 0x07, 0x00, 0x00, 0xf4, 0xff, 0x0b, 0xc0, 0xac, 0x93, 0x77, 0xf4, 0x1d, 0x0b, 0x60, 0xa0, 0x94, 0x90, 0xf4, 0xda, 0x0a, 0x20, 0xa0, 0x94, 0x90, 0xf4, 0xda, 0x0a, 0x20, 0xa0, 0xf3, 0x77, 0x74, 0x17, 0x0b, 0x60, 0xa8, 0x90, 0x30, 0x74, 0xd0, 0x0a, 0xc0, 0xac, 0x90, 0x50, 0x74, 0xd7, 0x0a, 0x80, 0xa7, 0x90, 0x97, 0x04, 0x17, 0x0b, 0x00, 0x00, 0x00, 0x00, 0xfc, 0xff, 0x0f, 0x00, 0x00, 0x00, 0x00, 0xfc, 0xff, 0x03, 0x00, 0x00, 0x00, 0x00, 0xfc, 0xff, 0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00};
```

get_image	
目的	指定の長方形エリアのビットマップパターンを取得します。
書式	void get_image(int left, int top, int width, int height, unsigned char *pat);
引数	int left
	長方形エリアの左上 X 座標
	int top
	長方形エリアの左上 Y 座標
	int width
	長方形エリアの幅をドット単位で指定
	int height
	長方形エリアの高さをドット単位で指定
	unsigned char *pat
	ビットマップデータを格納する変数へのポインタ
コーディング例	get_image (12, 32, 60, 16, buf);
戻り値	無し
備考	引数で指定された長方形エリアのビットマップパターンを指定の変数へ格納します。 ➤ この処理を実行しても、呼び出し位置には、影響しません。







show_image	
目的	長方形エリアにビットマップデータを描画します。
書式	void show_image(int left, int top, int width, int height, unsigned char *pat);
引数	int left
	長方形エリアの左上 X 座標
	int top
	長方形エリアの左上 Y 座標
	int width
	長方形エリアの幅をドット単位で指定
	int height
	長方形エリアの高さをドット単位で指定
	unsigned char *pat
	ビットマップデータを格納した変数へのポインタ
コーディング例	show_image(35, 5, 52, 24, CipherLab_logo);
戻り値	無し
備考	引数で指定された長方形エリアに指定のビットマップパターンを描画します。 ➤ この処理を実行しても、呼び出し位置には、影響しません。

## 2.11.6 グラフィック

白黒グラフィックには下の表のような3つの要素があります。

要素	パラメータ	関数
表示モード	VIDEO_REVERSE VIDEO_NORMAL	1 0 SetVideoMode()
ピクセル	DOT_MARK DOT_CLEAR DOT_RESERVE	1 0 -1 circle(), line(), putpixel(), rectangle()
シェイプ	SHAPE_FILL SHAPE_NORMAL	1 0 circle(), rectangle()

上記の組み合わせ例は次の通りです。

シェイプ	ピクセル		
	DOT_MARK	DOT_CLEAR	DOT_RESERVE
SHAPE_FILL			
SHAPE_NORMAL			

circle		
目的	LCD に円を描画します。	
書式	void circle(int x, int y, int r, int type, int mode);	
引数	int x, y	
	円の中心の X,Y 座標	
	int r	
	円の半径	
	int type	
	0	SHAPE_NORMAL 塗りつぶしなし
	1	SHAPE_FILL 塗りつぶしあり
	int mode	
	0	DOT_RESERVE 反転ドット
	1	DOT_CLEAR ドットなし
	1	DOT_MARK ドットあり
コーディング例	<pre>circle(80, 120, 8, SHAPE_FILL, DOT_MARK); // show a solid black circle centered at the position of (80, 120) with radius of 8 pixels</pre>	
戻り値	無し	
参照	line, rectangle	

line		
目的	LCD に直線を描画します。	
書式	void line(int x1, int y1, int x2, int y2, int mode);	
引数	int x1, y1	
	直線の始点 X,Y 座標	
	int x2, y2	
	直線の終点 X,Y 座標	
	int mode	
	0	DOT_RESERVE 反転ドット
	1	DOT_CLEAR ドットなし
	1	DOT_MARK ドットあり
コーディング例	<pre>line(10, 10, 120, 10, DOT_MARK); // draw a horizontal line line(80, 120, 10, 10, DOT_MARK); // draw a oblique line</pre>	
戻り値	無し	
参照	circle, rectangle	

pupixel		
目的	LCD に点を描画します。	
書式	void pupixel(int pos_x, int pos_y, int mode);	
引数	int x1, y1	
	点の X,Y 座標	
	int mode	
	0	DOT_RESERVE 反転ドット
	1	DOT_CLEAR ドットなし
	1	DOT_MARK ドットあり
コーディング例	<pre>pupixel 80, 120, DOT_REVERSE); // mark or clear the dot at (80,120) depending on the pixel status</pre>	
戻り値	無し	

rectangle		
目的	LCD に長方形を描画します。	
書式	void rectangle(int x1, int y1, int x2, int y2, int type, int mode);	
引数	int x1, y1	
	長方形の開始 X,Y 座標	
	int x2, y2	
	長方形の終了 X,Y 座標	
	int type	
	0	SHAPE_NORMAL 塗りつぶしなし
	1	SHAPE_FILL 塗りつぶしあり
	int mode	
	0	DOT_RESERVE 反転ドット
	1	DOT_CLEAR ドットなし
	1	DOT_MARK ドットあり
コーディング例	rectangle(10, 20, 80, 100, SHAPE_FILL, DOT_MARK); // show a solid black rectangle	
戻り値	無し	
参照	circle, line	

## 2.12 タッチスクリーン

8500/8700 シリーズは、InitTouchScreen()で初期化することにより、液晶ディスプレイ(LCD)をタッチスクリーンで使用できます。

### ➤ 文字認識

SignatureCapture()で定義された LCD のエリアに直接何か文字を書くのにスタイラスを使用します。その文字入力は GetScreenItem()でキャプチャすることができます。

### ➤ 操作できるアイテム

グラフィックアイテムは、例えば電卓のようなタッチ操作時の動作を策定することができます。ItemProperty 構造体に位置やサイズを含む「グラフィックアイテム」(ポインタ)の情報を前もって定義する必要があります。

グラフィックアイテムのパターンを設計し、show\_image()で LCD に表示することができます。それらのアイテムは GetScreenItem()によってタッチしたことを検知することができます。

選択されたアイテムの表示モードが ITEM\_REVERSE に設定されている場合、そのアイテムにタッチすると、アイテムは反転表示されます。

### 2.12.1 ItemProperty 構造体

```
typedef struct {  
    int UpLeftX;  
    int UpLeftY;  
    int SizeX;  
    int SizeY;  
};ItemProperty;
```

int UpLeftX	アイテムの左上 X 座標
int UpLeftY	アイテムの左上 Y 座標
int SizeX	アイテムの幅をドット単位で指定
int SizeY	アイテムの高さをドット単位で指定

GetPoint	8500,8700				
目的	タッチスクリーン上の動作の開始位置と終了位置を取得します。				
書式	int GetPoint(int* DownX, int* DownY, int* UpX, int* UpY);				
引数	<table><tr><td>int DownX, DownY</td></tr><tr><td>開始 X,Y 座標を格納する変数へのポインタ</td></tr><tr><td>int UpX, UpY</td></tr><tr><td>終了 X,Y 座標を格納する変数へのポインタ</td></tr></table>	int DownX, DownY	開始 X,Y 座標を格納する変数へのポインタ	int UpX, UpY	終了 X,Y 座標を格納する変数へのポインタ
int DownX, DownY					
開始 X,Y 座標を格納する変数へのポインタ					
int UpX, UpY					
終了 X,Y 座標を格納する変数へのポインタ					
コーディング例	val = GetPoint(&dX, &dY, &uX, &uY);				
戻り値	成功すれば 1 を返します。タッチ操作がない場合、0 を返します。				
参照	circle, rectangle				

GetScreenItem		8500,8700
目的	アイテムを選択したアイテム番号、または文字認識エリアに書かれた入力を検知します。	
書式	int GetScreenItem(ItemProperty* Item, int TotalItems, int mode);	
引数	ItemProperty* Item	
	アイテムサイズ情報のリスト	
	int TotalItems	
	アイテム総数	
	int mode	
	0	ITEM_NORMAL      選択したアイテムをそのまま表示
	1	ITEM_REVERSE      選択したアイテムを反転表示
コーディング例	<pre>const ItemProperty Buttonlist[3] = {{8, 8, 24, 16}, {38, 8, 24, 16}, {68, 8, 24, 16}}; while(event) {     . . .     Val = GetScreenItem((void *)Buttonlist, 3, ITEM_REVERSE); }</pre>	
戻り値	成功すれば選択されたアイテムの番号を返します (文字認識の戻り値はありません)。選択されたアイテムがない、または文字認識されていない場合、0 が返ります。	
備考	<p>コーディング例のようなループ文の前に、InitTouchScreen() をコールする必要があります。関数がコールされた時に検知するので、機能を継続するには続けてコールする必要があります。</p> <p>➤ ItemProperty は、1 つのアイテムの左上の XY 座標と幅、高さからなるデータ構造体です。</p>	
参照	InitTouchScreen, show_image, SignatureCapture	

GetTouchScreenState		8500,8700
目的	現在のタッチスクリーン状態を取得します。	
書式	int GetTouchScreenState(void);	
コーディング例	val = GetTouchScreenState();	
戻り値	タッチスクリーンが有効(初期化済み)なら 1 を返します。そうでない場合は 0 を返します。	
参照	HaltTouchScreen, InitTouchScreen	

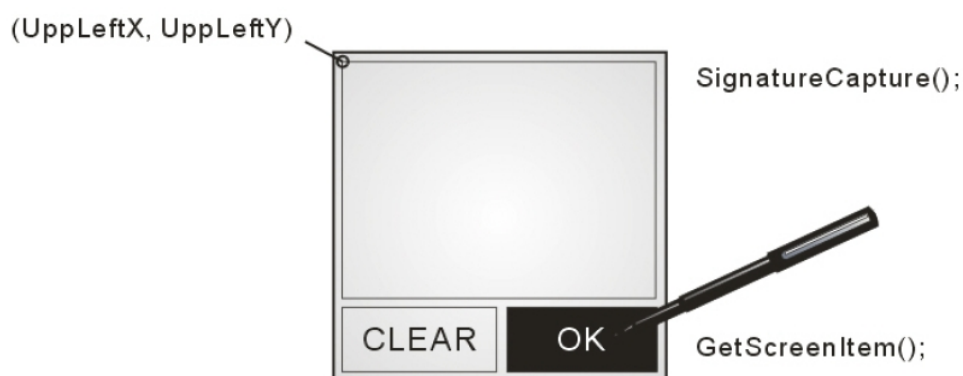
HaltTouchScreen		8500,8700
目的	タッチスクリーン操作を終了します。	
書式	void HaltTouchScreen(void);	
コーディング例	HaltTouchScreen();	
戻り値	なし	
備考	タッチスクリーンを再開するには、InitTouchScreen をコールする必要があります。タッチスクリーンは初期化されるまで機能しません。	
参照	InitTouchScreen	

InitTouchScreen		8500,8700
目的	タッチスクリーン操作を初期化します。	
書式	void InitTouchScreen(void);	
コーディング例	InitTouchScreen();	
戻り値	なし	
参照	HaltTouchScreen	



SignatureCapture		8500,8700
目的	タッチスクリーン上の文字認識エリアを定義します。1-サ-はそのエリアにスタイラスを使って自由に書くことができます。	
書式	int SignatureCapture(int UppLeftX, int UppLeftY, int LowRightX, int LowRightY);	
引数	int UppLeftX, int UppLeftY	
	文字認識エリアの左上 XY 座標	
	int LowRightX, int LowRightY	
	文字認識エリアの右下 XY 座標	
コーディング例	SignatureCapture(8, 8, 150, 100);	
戻り値	なし	
参照	GetScreenItem	

## 2.12.2 使用例



```

main()
{
    :
    OSTaskCreate(TouchScreenTask...);
    :
    while(1) {
        getchar();
        :
    }
}

TouchScreenTask()
{
    :
    InitTouchScreen();
    SignatureCapture(...);
    while(1) {
        c = GetScreenItem(...);
        :
        putch(c);
    }
}

```

## 2.13 フォント

### 2.13.1 フォントサイズ

基本的に、ハンディターミナルはシステムフォントとして 6×8 と 8×16 の 2 種類のフォントを選択できます。これらの選択は、マルチ言語フォントファイルやアプリ言語/北欧語/ポランド語/ロシア語フォントファイルのような 1 バイト文字のフォントファイルでも適用されます。

➤ LCD のデフォルトは 6×8 です。

標準のシステムフォント(英語)に加え、別途対応するファイルを CipherLab シリーズハンディターミナルにダウンロードすることで、各国のフォントをディスプレイに表示することができます。

フォントファイル		カスタムフォントサイズ	SetFont 関数で選択可能なパラメータ
1 バイト文字	システムフォント(デフォルト)	該当なし	FONT_6X8, FONT_8X16
	マルチ言語フォントファイル	該当なし	FONT_6X8, FONT_8X16
	その他: He, Nd, Po, Ru	該当なし	FONT_6X8, FONT_8X16
2 バイト文字	Tc, Sc, Jp, Kr	16×16	FONT_6X8, FONT_8X16
	Tc12, Sc12, Jp12, Kr12	12×12	FONT_6X12, FONT_12X12

### 2.13.2 表示性能

画面サイズとフォントサイズによる英数文字の表示可能行数と文字数は次の通りです。

画面サイズ(ドット)		英数字フォント	表示可能文字数	アイコン表示エリア
8000	100 × 64	フォントサイズ 6 × 8ドット	16 文字 × 8 行	右端 4 × 64
		フォントサイズ 8 × 16ドット	12 文字 × 4 行	右端 4 × 64
8300	128 × 64	フォントサイズ 6 × 8ドット	20 文字 × 8 行	右端 8 × 64
		フォントサイズ 8 × 16ドット	15 文字 × 4 行	右端 8 × 64
8200, 8400	160 × 160	フォントサイズ 6 × 8ドット	26 文字 × 18 行	下端 160 × 16
		フォントサイズ 8 × 16ドット	20 文字 × 9 行	下端 160 × 16
8500, 8700	160 × 160	フォントサイズ 6 × 8ドット	26 文字 × 19 行	下端 160 × 8
		フォントサイズ 8 × 16ドット	20 文字 × 9 行	下端 160 × 16

※ 8200/8400/8500/8700 シリーズでは、ICON\_ZONE(1)に設定し、ハッチアイコン等を表示しないようにすれば、20 行 (または 10 行)まで表示することができます。

### 2.13.3 マルチ言語フォント

マルチ言語フォントファイルには英語(デフォルト)、フランス語、アプリ言語、デンマーク語、北欧語、ポルトガル語、トルコ語、ロシア語、ポランド語、スラブ語、スロバキア語などが含まれています。これらの言語のうち英語以外の言語を表示するには、地域による言語を指定するために SetLanguage()をコールする必要があります。

### 2.13.4 特殊フォント

ファイル名に Tc12(繁体字中国語)、Sc12(簡体字中国語)、Jp12(日本語)あるいは Kr12(朝鮮語)があるフォントは、特殊フォントファイルと呼ばれます。これは、これらのフォントサイズは SetFont()で 6×12 または 12×12 のいずれかに指定する必要があります。設定しなければ、文字が正しく表示されません。

## CheckFont

**目的** インストールされているフォントファイルをチェックします。

**書式** int CheckFont(void);

**コーディング例** n = CheckFont();

**戻り値** 下記の何れかの値を返します。

0x00	フォントファイル無し(システムフォントのみ)	
0x01	TC(繁体字中国語)	16×16, Big-5 コード
0x02	予備	
0x03	SC(簡体字中国語)	16×16, GB コード
0x04	KR(韓国語)	
0x05	JP(日本語)	16×16
0x06	HE(ヘブライ語)	
0x07	PO(ポーランド語)	
0x08	RU(ロシア語)	
0x09	TC12(繁体字中国語)	12×12, Big-5 コード
0x0a	予備	
0x0b	SC12(簡体字中国語)	12×12, GB コード
0x0c	JP12(日本語)	12×12
0x0d	KR12(韓国語)	12×12
0x10	MULTI(マルチ言語)	

**参照** FontVersion, SetLanguage

## GetFont

**目的** フォント情報を取得します。

**書式** int GetFont (void);

**コーディング例** if (GetFont( ) == FONT8X16)  
puts ("Font : 8X16");

**戻り値** 下記の何れかの値を返します。

FONT6X8	6X8ドットフォント
FONT8X16	8X16ドットフォント
FONT6X12	6X12ドットフォント
FONT12X12	12X12ドットフォント

SetFont									
目的	使用するフォントサイズ タイプ を指定します。								
書式	void SetFont(int font);								
引数	int font								
	<table> <tr> <td>FONT6X8</td><td>6X8 ドットフォント</td></tr> <tr> <td>FONT8X16</td><td>8X16 ドットフォント</td></tr> <tr> <td>FONT6X12</td><td>6X12 ドットフォント</td></tr> <tr> <td>FONT12X12</td><td>12X12 ドットフォント</td></tr> </table>	FONT6X8	6X8 ドットフォント	FONT8X16	8X16 ドットフォント	FONT6X12	6X12 ドットフォント	FONT12X12	12X12 ドットフォント
FONT6X8	6X8 ドットフォント								
FONT8X16	8X16 ドットフォント								
FONT6X12	6X12 ドットフォント								
FONT12X12	12X12 ドットフォント								
コーディング例	SetFont (FONT_8X16);								
戻り値	無し								
備考	<p>使用するフォントサイズ をこの関数をコールすることにより明示します。</p> <ul style="list-style-type: none"> <li>➤ 1 バイト文字 1 バイト文字(システム、マルチ言語、など)では、FONT_6X8 または FONT_8X16 を設定することができます。</li> <li>➤ 16×16 の 2 バイト文字 英数文字表示のために FONT_6X8 または FONT_8X16 を設定することができます。</li> <li>➤ 12×12 の 2 バイト文字 FONT_6X12 を割り当てれば、1 バイト文字のフォントサイズ は 6x12 になります。その一方で 2 バイト文字(Tc12、Sc12、Jp12、Kr12) のフォントサイズ は 12x12 になります。</li> </ul>								
参照	SetLanguage								

SetLanguage			
目的	マルチ言語フォントファイルから使用するフォントを指定します。		
書式	void SetLanguage (int setting);		
引数	int setting		
	0x10	English_437	英語(デフォルト)
	0x11	French_863	カナダ・フランス語
	0x12	Hebrew_862	ヘブライ語
	0x13	Latin_850	multilingual Latin
	0x14	Nordic_865	北欧語
	0x15	Portugal_860	ポルトガル語
	0x16	CP_1251	キリル文字(ロシア語)
	0x17	CP_852	ラテン語 II (スラブ 語)
	0x18	CP_1250	中央ヨーロッパ・ラテン語(ホーランド 語)
	0x19	Turkish_857	トルコ語
	0x1a	Latin_II	ラテン語 II (スロバキア語)
	0x1b	WIN1250	Windows1250
	0x1c	ISO_28592	ISO-28592 / ISO-8859-2
	0x1d	IBM_LATIN_II	IBM-ラテン語 II
	0x1e	Greek_737	ギリシア語
	0x1f	CP_1252	ラテン語 I
	0x20	CP_1253	ギリシア語
コーディング例	SetLanguage (Nordic_865); /* 北欧語を指定*/		
戻り値	無し		
説明	マルチ言語フォントファイルから使用するフォントを指定します。但し、CipherLab シリーズ ハードウェアミカルにマルチ言語フォントがダウンロードされていることが前提となります。		
参照	CheckFont, SetFont		

## 2.13.5 フォントファイル

8000,8300 シリーズのフォントファイル	フォントサイズ
Font-Hebrew.shx	6×8 または 8×16
Font-Japanese.shx	16×16(4 行)
Font-Japanese12.shx	6×12 または 12×12(5 行)
Font-Korean.shx	16×16(4 行)
Font-Korean12.shx	6×12 または 12×12(5 行)
Font-Nordic.shx	6×8 または 8×16
Font-Polish.shx	6×8 または 8×16
Font-Russoan.shx	6×8 または 8×16
Font-SimplifiedChinese.shx	16×16(4 行)
Font-SimplifiedChinese12.shx	6×12 または 12×12(5 行)
Font-TraditionalChinese.shx	16×16(4 行)
Font-TraditionalChinese12.shx	6×12 または 12×12(5 行)
Font-MultiLanguage.shx	6×8 または 8×16

※ 上記のフォントファイルは 2MB のフラッシュメモリを確保するために再コンパイルされ、それに応じて変更されています。

8200,8400,8700 シリーズのフォントファイル	フォントサイズ
Font8x00-Hebrew.shx	6×8 または 8×16
Font8x00-Japanese.shx	16×16(9 行)
Font8x00-Japanese12.shx	6×12 または 12×12(12 行)
Font8x00-Korean.shx	16×16(9 行)
Font8x00-Nordic.shx	6×8 または 8×16
Font8x00-Polish.shx	6×8 または 8×16
Font8x00-Russoan.shx	6×8 または 8×16
Font8x00-SimplifiedChinese.shx	16×16(9 行)
Font8x00-SimplifiedChinese12.shx	6×12 または 12×12(12 行)
Font8x00-TraditionalChinese.shx	16×16(9 行)
Font8x00-TraditionalChinese12.shx	6×12 または 12×12(12 行)
Font8x00-MultiLanguage.shx	6×8 または 8×16

8500 シリーズのフォントファイル	フォントサイズ
Font8500-Japanese.shx	16×16(9 行)
Font8500-Korean.shx	16×16(9 行)
Font8500-SimplifiedChinese.shx	16×16(9 行)
Font8500-SimplifiedChinese12.shx	6×12 または 12×12(12 行)
Font8500-TraditionalChinese.shx	16×16(9 行)
Font8500-TraditionalChinese12.shx	6×12 または 12×12(12 行)
Font8500-MultiLanguage.shx	6×8 または 8×16

## 2.14 メリ

この章では、フラッシュメモリと、プログラムメモリーおよびファイルシステムがある SRAM に関連する処理について説明します。

➤ 8200/8400/8700 シリーズは SD カード も使用することができます。

	フラッシュメモリ	SRAM	SD カード
8000 シリーズ	2MB	2MB, 4MB	対応なし
8200 シリーズ	8MB	4MB, 8MB	対応
8300 シリーズ	2MB	2MB, 6MB, 10MB	対応なし
8400 シリーズ	4MB	4MB, 16MB	対応
8500 シリーズ	2MB	2MB, 6MB, 10MB	対応なし
8700 シリーズ	8MB	4MB, 16MB	対応

### 2.14.1 フラッシュメモリ

フラッシュメモリは、いくつかのメモリバンクに分割されています。1 バンクあたり 64 バイトです。

➤ 2MB の場合、32 バンクに分割されています。(8000/8300/8500)

➤ 4MB の場合、64 バンクに分割されています。(8400)

➤ 8MB の場合、128 バンクに分割されています。(8200/8700)

#### 8000, 8300, 8400, 8500

カーネル自体は 2 つのバンクを使用し、システムがアプリケーションの設定として、データストレージ用に 1 つのバンク(0xF60000～0xF6FFFF)を使用します。残りのバンクは、フォントファイル同様、保存するユーザープログラムのために用意されています。フラッシュメモリは不揮発性であるため、同じバンク 0xF60000-0xF6FFFF に書き込む前に消去する必要があります。このメモリバンクは、さらに 1～256 に番号を振られた最大 255 バイトの、256 レコードに分割されています。

(1) 最大 256 個のレコードに保存することができます。フラッシュメモリは、バンクごとに消去することができます。つまり、0xF60000-0xF6FFFF に格納されているすべてのレコードが消えてしまいます。

(2) 8400 シリーズは、将来用に 6 バンク(0xF00000-0xF5FFFF)を予備としています。

#### 8200, 8700

カーネル自体は 22 バンクを使用し、システムがアプリケーションの設定として、データストレージ用に 2 つのバンク (0xF60000～0xF6FFFF, 0x800000～0xBF5FFFF) を使用します。残りのバンクは、フォントファイル同様、保存するユーザープログラムのために用意されています。

➤ フラッシュメモリ上のユーザープログラム領域：0xC00000～0xDFFFFFFF

➤ フラッシュメモリ上のカーネル領域：0xE00000～0xF5FFFF

➤ フラッシュメモリ上のユーザープログラム領域：0xFF0000～0xFFFFFFFF

EraseSector	
目的	フラッシュメモリのセクタ全体を消去します。
書式	int EraseSector (void* sector_start_addr);
コーディング例	EraseSector (0xf60000);
戻り値	消去したバイト数を返します。
備考	フラッシュメモリのセクタ全体を消去します。WriteFlash()をコールする前に必ず実行しなければいけません。

FlashSize	
目的	フラッシュメモリサイズを取得します。
書式	int FlashSize (void);
コーディング例	FlashSize ( );
戻り値	フラッシュメモリサイズをバイト(KB)単位で返します。

WriteFlash	
目的	フラッシュメモリにデータを書込みます。
書式	int WriteFlash (void *target_addr, void *source_addr, unsigned long size);
引数	void *target_addr
	書込みを行う開始アドレス
	void *source_addr
	書込みデータを格納した変数へのポインタ
	unsigned long size
	書込みデータサイズ
コーディング例	char szData[100]; EraseSector (0xf60000); WriteFlash (0xf60000, szData, 100);
戻り値	書込みを行ったバイト数を返します。
説明	フラッシュメモリにデータを書込みます。 ➤ 書込みを行える開始アドレスは、0xF60000 で、サイズは 64K バイトです。

## 2.14.2 SRAM

ファイルシステムは、バックアップバッテリーが供給される間ユーザーデータを SRAM に保存しておくことができます。ただし、ローバッテリー時、またはバッテリーが消耗しているとデータを損失する場合があります。使用後、必要に応じてホストコンピュータにデータをアップロードしてください。

### free\_memory

目的	SRAM メモリの空き容量を取得します。
書式	long free_memory (void);
コーディング例	available_memory = free_memory ( );
戻り値	SRAM メモリの空き容量をバイト単位で返します。
備考	SRAM メモリの空き容量を取得します。

### init\_free\_memory

目的	SRAM メモリ(ファームウェア)を初期化します。
書式	void init_free_memory (void);
コーディング例	init_free_memory ( );
戻り値	無し
説明	<p>この関数は、最初に搭載されている SRAM の総容量をチェックし、全領域を初期化します(総メモリからシステム領域とユーザー領域を除く)。</p> <ul style="list-style-type: none"><li>➤ 初期化を実行すると、ファームウェアの内容は全て消失しますので、注意してください。</li><li>➤ この処理は、SRAM メモリを増設した場合など、メモリ容量に変更を生じた場合に必ず実行する必要があります。</li></ul>

### RamSize

目的	SRAM メモリサイズを取得します。
書式	int RamSize (void);
コーディング例	RamSize( );
戻り値	SRAM メモリサイズをバイト(KB)で返します。



### 2.14.3 SD カード

#### ffreebyte

目的	SD カード の空き容量をバイト単位で取得します。
書式	long ffreebyte(void);
コーディング 例	<pre>long freekb; if ((freekb = ffreebyte()) == -1L)     printf("Get free bytes failed");</pre>
戻り値	SD カード の空き容量をバイト(KB)で返します。空き容量取得に失敗した場合、-1 が返ります。発生したエラー状態はグローバル変数 <code>errno</code> にセットされます。

#### fsize

目的	SD カード の使用可能領域のサイズ を取得します。
書式	long fsize(void);
コーディング 例	<pre>long size; if ((size = fsize()) == -1L)     printf("Get card size failed");</pre>
戻り値	SD カード のサイズ をバイト(KB)で返します。サイズ 取得に失敗した場合、-1 が返ります。発生したエラー状態はグローバル変数 <code>errno</code> にセットされます。

## 2.15 ファイル操作

プログラムが高速且つ容易にトランザクションやデータ操作を行えるよう、多数のファイル操作関数が用意されています。これらの関数は、データ変更処理に有効で、データシステムの実装を容易にします。

次の2種類のファイルタイプをサポートしています。

➤ シーケンシャル構造の DAT ファイルは主にトランザクションデータの保存に使用されます。

➤ DBF ファイルは、データレコードの保存やインデックスを利用した検索ファイルとして利用されます。

これら2種類のファイルタイプについては、別途詳細を後述します。

8200/8400/8700 の場合、DAT ファイルや DBF ファイルを保存している SD カードをサポートしています。『2.16 SD カード』を参照してください。

ファイル構造	SRAM 上のファイル	SD カード上のファイル
DAT ファイル	『2.15.6 DAT ファイル』参照	『2.16.5 SD カード 操作』参照
DBF および IDX ファイル	『2.15.7 DBF ファイルと IDX ファイル』参照	

### 2.15.1 ファイルシステム

CipherLab シリーズ ハードウェアは、モデルにより容量は異なりますが、オプション SRAM メモリを搭載しています。この SRAM メモリをシステムパラメータ、プログラム変数、プログラムスタック、ファイルシステムで共用することになります。

### 2.15.2 ディレクトリ

ファイルシステムは、サブディレクトリなどの階層構造をサポートしていません。また、作成できるファイル数は、DAT、DBF 及び関連する IDX ファイルを全て含めて、最大 254 ファイル迄となります。ファイルディレクトリ内の情報は、filelist 関数で取得することができます。

### 2.15.3 ファイル名

1~8 文字(リターンシーケンスを含まない)のファイル名が使用可能で、MS-DOS などで使用されている拡張子はサポートしていません。ファイル作成後、rename() でファイル名を変更することも可能です。

➤ 8 文字を超えるファイル名が指定された場合、8 文字に切り詰められます。

➤ ファイル名は、大文字・小文字が区別されます。

### 2.15.4 ファイルハンドル

ファイルオープン(又は作成)時にシステムが割り当てる 0 以上の値で、以降のファイル識別番号となります。ファイルオープン後のファイル操作では、ファイル名の変わりに、このファイルハンドルを指定して関数をコールします。

### 2.15.5 エラーコード

システムグローバル変数 fErrorCode に最後に行ったファイル操作のエラーコードがセットされます。

➤ 値が 0 以外の場合は、エラーであることを意味しています。このエラーコードは、read\_error\_code() をコールして取得することも可能です。

## access

**目的** ファイルが存在するかをチェックします。

**書式** int access(char\* filename);

**引数** char\* filename

存在するかをチェックしたいファイル名が格納されたポインタ。  
8文字を超えるファイル名が指定された場合、8文字に切り詰められます。

**コーディング例** if (access("data1")) puts("data1 exist!\n");

**戻り値** ファイルが存在する場合、1を返し、存在しない場合は、0を返します。  
エラーが発生すると、戻り値 -1 を返し、エラーコードがシステムグローバル変数 fErrorCode にセットされます。

エラーコード	説明
1	ファイル名が空文字列です。

## filelist

**目的** ファイルディレクトリ情報を取得します。

**書式** int filelist(char\* dir);

**引数** char\* dir

ファイルディレクトリ情報を格納する変数へのポインタ。  
バッファのサイズは少なくとも 25×ファイル数+1 用意する必要があります。25 バイトはファイル情報が格納され、最後の 1 バイトは終端文字が格納されます。ファイル情報のフォーマットは次の通りです。

ファイル名 最大 8 バイト	スペース 1 バイト	ファイル種別 最大 4 バイト	スペース 1 バイト	ファイル長 最大 10 バイト	スペース 1 バイト
次ファイル 最大 25 バイト	...	NULL 1 バイト			

**コーディング例** total\_file = filelist(dir);

**戻り値** 存在するファイル数を返します。

**備考** 存在する全ファイルの情報(ファイル名, ファイルタイプ, ファイルサイズ)をスペースキャラクタで区切った書式で、引数で与えられた変数へ格納します。

## get\_file\_number

**目的** 引数で指定された種別のファイル数を取得します。

**書式** int get\_file\_number(int type);

**引数** int type

0	全ファイル数
1	DAT ファイル数
2	DBF ファイル数
3	インデックスファイル数

**コーディング例** total\_DAT\_file = get\_file\_number(1);

**戻り値** 引数で指定された種別のファイル数

**備考** filelist()と get\_file\_number(0)は同じ戻り値になります。

<b>read_error_code</b>
------------------------

目的	システムグローバル変数 fErrorCode の値を取得します。
書式	int read_error_code( );
コーディング例	if (read_error_code() == 2) puts("File not exist!\n");
戻り値	システムグローバル変数 fErrorCode の値を返します。
備考	システムグローバル変数 fErrorCode の値を戻り値として返します。システムグローバル変数 fErrorCode は、プログラム直接参照可能ですが、この関数を利用することも可能です。

<b>remove</b>
---------------

目的	ファイルを削除します。								
書式	int remove(char* filename);								
引数	char* filename 削除するファイル名が格納されたポインタ。 8 文字を超えるファイル名が指定された場合、8 文字に切り詰められます。 DBF ファイルを指定した場合は、関連する全ての IDX ファイルも同時に削除されます。								
コーディング例	if (remove("data1")) puts("data1 deleted!\n");								
戻り値	正常終了の場合 1 を返し、エラーが発生した場合は 0 を返します。 エラーが発生すると、エラーコードがシステムグローバル変数 fErrorCode にセットされます。								
	<table border="1"><tr><th>エラーコード</th><th>説明</th></tr><tr><td>1</td><td>ファイル名が空文字列です。</td></tr><tr><td>2</td><td>指定されたファイルが存在しません。</td></tr><tr><td>10</td><td>空き容量が足りません。</td></tr></table>	エラーコード	説明	1	ファイル名が空文字列です。	2	指定されたファイルが存在しません。	10	空き容量が足りません。
エラーコード	説明								
1	ファイル名が空文字列です。								
2	指定されたファイルが存在しません。								
10	空き容量が足りません。								

<b>rename</b>
---------------

目的	ファイル名を変更します。								
書式	int rename(char* old_filename, char* new_filename);								
引数	char* old_filename 変更前のファイル名が格納されたポインタ。 char* new_filename 変更後のファイル名が格納されたポインタ。 ➤ 8 文字を超えるファイル名が指定された場合、8 文字に切り詰められます。 ➤ DBF ファイルを指定した場合は、関連する全ての IDX ファイルも同時にリネームされます。								
コーディング例	if (rename("data1", "text1")) puts("data1 renamed!\n");								
戻り値	正常終了の場合 1 を返し、エラーが発生した場合は 0 を返します。 エラーが発生すると、エラーコードがシステムグローバル変数 fErrorCode にセットされます。								
	<table border="1"><tr><th>エラーコード</th><th>説明</th></tr><tr><td>1</td><td>old_filename 又は new_filename で指定されたファイル名が空文字列です。</td></tr><tr><td>2</td><td>old_filename で指定されたファイルが存在しません。</td></tr><tr><td>3</td><td>new_filename で指定されたファイル名が既に存在します。</td></tr></table>	エラーコード	説明	1	old_filename 又は new_filename で指定されたファイル名が空文字列です。	2	old_filename で指定されたファイルが存在しません。	3	new_filename で指定されたファイル名が既に存在します。
エラーコード	説明								
1	old_filename 又は new_filename で指定されたファイル名が空文字列です。								
2	old_filename で指定されたファイルが存在しません。								
3	new_filename で指定されたファイル名が既に存在します。								

## 2.15.6 DAT ファイル

DAT ファイルは、シーケンシャルファイル構造になっています。

- 先頭データは、`delete_top()` 又は `delete_topln()` で削除することができます。それに相応して、先頭データ削除後、ファイル先頭位置、ファイルポインタ、DAT ファイルサイズ の調整が行われます。
- データの追加は、`append()` 及び `appendln()` で行います。この 2 つの関数は、ファイルポインタ位置に関係なく EOF (ファイルの終端) 位置にデータの追加書き込みを行うことができます。つまり、これらの関数がコールされても、ファイルポインタ位置は影響を受けず、そのままです。通常、DAT ファイルは、このような特徴を活かして、ファイルの先頭からデータを読み取り・削除し、ファイルの終端へ新しいデータを追加する様なトランザクションデータを扱うために利用されます。

append															
目的	DAT ファイルの終端へ指定バイト数のデータを書込みます。														
書式	<code>int append (int fd, char* buffer, int count);</code>														
引数	<code>int fd</code>														
	DAT ファイルのファイルハンドル。														
	<code>char* buffer</code>														
	書込むデータが格納されたポインタ。														
	<code>int count</code>														
	書込むバイト数。書き込みを行える最大バイト数は、32,767 です。														
コーディング例	<code>append(fd, "1234567890", 10);</code>														
戻り値	実際に書込んだデータのバイト数を返します。														
	エラーが発生した場合は -1 を返し、エラーコードがシステムグローバル変数 <code>fErrorCode</code> にセットされます。														
	<table><tr><th>エラーコード</th><th>説明</th></tr><tr><td>2</td><td>fd で指定されたファイルが存在しません。</td></tr><tr><td>4</td><td>fd で指定されたファイルは DAT ファイルではありません。</td></tr><tr><td>7</td><td>ファイルハンドルが無効です。</td></tr><tr><td>8</td><td>ファイルがオープンされていません。</td></tr><tr><td>9</td><td>count に負の値が指定されています。</td></tr><tr><td>10</td><td>データを書込む空き容量が足りません。</td></tr></table>	エラーコード	説明	2	fd で指定されたファイルが存在しません。	4	fd で指定されたファイルは DAT ファイルではありません。	7	ファイルハンドルが無効です。	8	ファイルがオープンされていません。	9	count に負の値が指定されています。	10	データを書込む空き容量が足りません。
エラーコード	説明														
2	fd で指定されたファイルが存在しません。														
4	fd で指定されたファイルは DAT ファイルではありません。														
7	ファイルハンドルが無効です。														
8	ファイルがオープンされていません。														
9	count に負の値が指定されています。														
10	データを書込む空き容量が足りません。														
備考	引数で指定されたバイト数のデータを DAT ファイルの終端に書込みます。 ➤ この操作を実行してもファイルポインタには、影響を与えません。また、この操作により、DAT ファイルサイズ は自動的に拡張されます。														
参照	<code>appendln</code> , <code>read</code> , <code>readln</code> , <code>write</code> , <code>writeln</code>														

appendln	
目的	DAT ファイルの終端ヘッダコード (双終端文字列) を書込みます。
書式	int appendln (int fd, char* buffer);
引数	int fd
	DAT ファイルのファイルハンドル。
引数	char* buffer
	書込むデータが格納されたポインタ。
コーディング例	appendln (fd, data_buffer);
戻り値	実際に書込んだデータのバイト数を返します。
	エラーが発生した場合は -1 を返し、エラーコードがシステムグローバル変数 fErrorCode にセットされます。
備考	エラーコード
	説明
	2 fd で指定されたファイルが存在しません。
	4 fd で指定されたファイルは DAT ファイルではありません。
	7 ファイルハンドルが無効です。
	8 ファイルがオープンされていません。
	10 データを書込む空き容量が足りません。
	11 buffer に双終端文字がありません。
備考	引数で指定された Null 文字で終わるデータコードを DAT ファイルの終端に書込みます。
	<ul style="list-style-type: none"> <li>➤ Null 文字が検出されるまでの文字がファイルに書き込まれます。Null 文字も、ファイルに書き込まれます。</li> <li>➤ この操作を実行してもファイルポインタには、影響を与えません。また、この操作により、DAT ファイルサイズは自動的に拡張されます。</li> </ul>
参照	append, read, readln, write, writeln

chsize	
目的	DAT ファイルサイズを拡張又は縮小します。
書式	int chsize (int fd, long new_size);
引数	int fd
	DAT ファイルのファイルハンドル。
引数	long new_size
	新しいファイルサイズ (バイト単位)。
コーディング例	if (chsize(fd,0L)) puts("file truncated!\n");
戻り値	正常終了の場合 1 を返します。
	エラーが発生した場合は 0 を返し、エラーコードがシステムグローバル変数 fErrorCode にセットされます。
備考	エラーコード
	説明
	2 fd で指定されたファイルが存在しません。
	4 fd で指定されたファイルは DAT ファイルではありません。
	7 ファイルハンドルが無効です。
	8 ファイルがオープンされていません。
	10 データを書込む空き容量が足りません。
備考	引数で指定されたサイズ (バイト) に DAT ファイルを拡張又は縮小します。
	<ul style="list-style-type: none"> <li>➤ 縮小した場合、新しいファイルサイズを超えるデータは全て失われます。</li> <li>➤ 拡張した場合は、初期値無しで新しいエリアが拡張されます。</li> </ul>

close											
目的	DAT ファイルをクローズ します。										
書式	int close(int fd);										
引数	int fd DAT ファイルのファイルハンドル。										
コーディング 例	if (close(fd)) puts("file closed!\n");										
戻り値	正常終了の場合 1 を返します。 エラーが発生した場合は 0 を返し、エラーコード がシステム グローバル変数 fErrorCode にセットされます。										
	<table> <tr> <th>エラーコード</th><th>説明</th></tr> <tr> <td>2</td><td>fd で指定されたファイルが存在しません。</td></tr> <tr> <td>4</td><td>fd で指定されたファイルは DAT ファイルではありません。</td></tr> <tr> <td>7</td><td>ファイルハンドルが無効です。</td></tr> <tr> <td>8</td><td>ファイルがオープンされていません。</td></tr> </table>	エラーコード	説明	2	fd で指定されたファイルが存在しません。	4	fd で指定されたファイルは DAT ファイルではありません。	7	ファイルハンドルが無効です。	8	ファイルがオープンされていません。
エラーコード	説明										
2	fd で指定されたファイルが存在しません。										
4	fd で指定されたファイルは DAT ファイルではありません。										
7	ファイルハンドルが無効です。										
8	ファイルがオープンされていません。										
参照	open										

delete_top													
目的	DAT ファイルの先頭位置から指定バイト数のデータを削除します。												
書式	int delete_top (int fd, int count);												
引数	int fd DAT ファイルのファイルハンドル。 int count 削除するバイト数。												
コーディング 例	delete_top (fd, 80);												
戻り値	実際に削除したデータのバイト数を返します。 エラーが発生した場合は-1 を返し、エラーコード がシステム グローバル変数 fErrorCode にセットされます。												
	<table> <tr> <th>エラーコード</th><th>説明</th></tr> <tr> <td>2</td><td>fd で指定されたファイルが存在しません。</td></tr> <tr> <td>4</td><td>fd で指定されたファイルは DAT ファイルではありません。</td></tr> <tr> <td>7</td><td>ファイルハンドルが無効です。</td></tr> <tr> <td>8</td><td>ファイルがオープンされていません。</td></tr> <tr> <td>9</td><td>count に負の値が指定されています。</td></tr> </table>	エラーコード	説明	2	fd で指定されたファイルが存在しません。	4	fd で指定されたファイルは DAT ファイルではありません。	7	ファイルハンドルが無効です。	8	ファイルがオープンされていません。	9	count に負の値が指定されています。
エラーコード	説明												
2	fd で指定されたファイルが存在しません。												
4	fd で指定されたファイルは DAT ファイルではありません。												
7	ファイルハンドルが無効です。												
8	ファイルがオープンされていません。												
9	count に負の値が指定されています。												
備考	<p>引数で指定された DAT ファイルの先頭位置から指定バイト数のデータを削除します。</p> <ul style="list-style-type: none"> <li>➤ この操作が行われると、ファイル サイズは自動的に調整されます。</li> <li>➤ 例えば、最初にファイル サイズが 10 バイト目を指していた場合、先頭から 8 バイトを削除すると、ファイル サイズは 2 バイト目を指すようになります。また、この操作により、DAT ファイルサイズは自動的に縮小されます。</li> </ul>												
参照	delete_topln												

delete_topln											
目的	DAT ファイルの先頭からデータコード (双終端文字列) を削除します。										
書式	int delete_topln (int fd);										
引数	int fd DAT ファイルのファイルハンドル。										
コーディング例	delete_topln (fd);										
戻り値	正常終了の場合、実際に削除したデータのバイト数を返します。 エラーが発生した場合は-1 を返し、エラーコードがシステムグローバル変数 fErrorCode にセットされます。										
	<table> <tr> <th>エラーコード</th><th>説明</th></tr> <tr> <td>2</td><td>fd で指定されたファイルが存在しません。</td></tr> <tr> <td>4</td><td>fd で指定されたファイルは DAT ファイルではありません。</td></tr> <tr> <td>7</td><td>ファイルハンドルが無効です。</td></tr> <tr> <td>8</td><td>ファイルがオープンされていません。</td></tr> </table>	エラーコード	説明	2	fd で指定されたファイルが存在しません。	4	fd で指定されたファイルは DAT ファイルではありません。	7	ファイルハンドルが無効です。	8	ファイルがオープンされていません。
エラーコード	説明										
2	fd で指定されたファイルが存在しません。										
4	fd で指定されたファイルは DAT ファイルではありません。										
7	ファイルハンドルが無効です。										
8	ファイルがオープンされていません。										
説明	<p>引数で指定された DAT ファイルの先頭からデータコード (双終端文字列) を削除します。</p> <ul style="list-style-type: none"> <li>➤ ファイルの先頭位置からキャラクタ位置又はファイルの終端迄を削除します。</li> <li>➤ この操作が行われると、ファイルポインタは自動的に調整されます。また、この操作により、DAT ファイルサイズは自動的に縮小されます。</li> </ul>										
参照	delete_top										

eof											
目的	DAT ファイルの終端であるかをチェックします。										
書式	int eof (int fd);										
引数	int fd DAT ファイルのファイルハンドル。										
コーディング例	if (eof(fd)) puts("end of file reached!\n");										
戻り値	ファイルポインタが終端を指している場合 1 を返し、終端でない場合は 0 と返します。 エラーが発生した場合は-1 を返し、エラーコードがシステムグローバル変数 fErrorCode にセットされます。										
	<table> <tr> <th>エラーコード</th><th>説明</th></tr> <tr> <td>2</td><td>fd で指定されたファイルが存在しません。</td></tr> <tr> <td>4</td><td>fd で指定されたファイルは DAT ファイルではありません。</td></tr> <tr> <td>7</td><td>ファイルハンドルが無効です。</td></tr> <tr> <td>8</td><td>ファイルがオープンされていません。</td></tr> </table>	エラーコード	説明	2	fd で指定されたファイルが存在しません。	4	fd で指定されたファイルは DAT ファイルではありません。	7	ファイルハンドルが無効です。	8	ファイルがオープンされていません。
エラーコード	説明										
2	fd で指定されたファイルが存在しません。										
4	fd で指定されたファイルは DAT ファイルではありません。										
7	ファイルハンドルが無効です。										
8	ファイルがオープンされていません。										

filelength											
目的	DAT ファイルのサイズをバイト単位で取得します。										
書式	long filelength (int fd);										
引数	int fd DAT ファイルのファイルハンドル。										
コーディング例	data_size = filelength (fd);										
説明	引数で指定された DAT ファイルのサイズをバイト単位で取得します。										
戻り値	正常終了の場合、DAT ファイルのサイズを long integer 値 (バイト単位) で返します。 エラーが発生した場合は-1 を返し、エラーコードがシステムグローバル変数 fErrorCode にセットされます。										
	<table> <tr> <th>エラーコード</th><th>説明</th></tr> <tr> <td>2</td><td>fd で指定されたファイルが存在しません。</td></tr> <tr> <td>4</td><td>fd で指定されたファイルは DAT ファイルではありません。</td></tr> <tr> <td>7</td><td>ファイルハンドルが無効です。</td></tr> <tr> <td>8</td><td>ファイルがオープンされていません。</td></tr> </table>	エラーコード	説明	2	fd で指定されたファイルが存在しません。	4	fd で指定されたファイルは DAT ファイルではありません。	7	ファイルハンドルが無効です。	8	ファイルがオープンされていません。
エラーコード	説明										
2	fd で指定されたファイルが存在しません。										
4	fd で指定されたファイルは DAT ファイルではありません。										
7	ファイルハンドルが無効です。										
8	ファイルがオープンされていません。										



lseek															
目的	DAT ファイルのファイルポインタを新しい位置へ移動します。														
書式	long lseek (int fd, long offset, int origin);														
引数	int fd														
	DAT ファイルのファイルハンドル。														
	long offset														
	移動させたい原点からのオフセット値(バイト)。														
	int origin														
	移動の基準となる原点位置														
	1      ファイルの先頭位置。														
	0      現ファイルポインタ位置														
	-1     ファイルの終端位置。														
コーディング例	lseek(fd, 512L, 0);                      /* 原点位置 0(ファイルの先頭)から 512 バイト分スキップ */														
戻り値	正常終了の場合、新しいファイルポインタの位置をファイルの先頭からのオフセットアドレスで返します。														
	エラーが発生した場合は-1L を返し、エラーコードがシステムグローバル変数 fErrorCode にセットされます。														
	<table> <tr> <th>エラーコード</th><th>説明</th></tr> <tr> <td>2</td><td>fd で指定されたファイルが存在しません。</td></tr> <tr> <td>4</td><td>fd で指定されたファイルは DAT ファイルではありません。</td></tr> <tr> <td>7</td><td>ファイルハンドルが無効です。</td></tr> <tr> <td>8</td><td>ファイルがオープンされていません。</td></tr> <tr> <td>9</td><td>引数 origin の値が無効です。</td></tr> <tr> <td>15</td><td>新しいファイルポインタ位置がファイルの終端位置を越えています。</td></tr> </table>	エラーコード	説明	2	fd で指定されたファイルが存在しません。	4	fd で指定されたファイルは DAT ファイルではありません。	7	ファイルハンドルが無効です。	8	ファイルがオープンされていません。	9	引数 origin の値が無効です。	15	新しいファイルポインタ位置がファイルの終端位置を越えています。
エラーコード	説明														
2	fd で指定されたファイルが存在しません。														
4	fd で指定されたファイルは DAT ファイルではありません。														
7	ファイルハンドルが無効です。														
8	ファイルがオープンされていません。														
9	引数 origin の値が無効です。														
15	新しいファイルポインタ位置がファイルの終端位置を越えています。														
備考	引数で指定された原点位置及びオフセット値を元に DAT ファイルのファイルポインタを移動します。														
参照	tell														

open											
目的	DAT ファイルをオープンします。										
書式	int open (char* filename);										
引数	char* filename										
	<p>オープンするファイル名が格納されたポインタ。</p> <p>指定されたファイル名が存在しない場合は、そのファイルが新規に作成されます。</p> <p>8 文字を超えるファイル名が指定された場合、8 文字に切り詰められます。</p>										
コーディング例	if ((fd = open("data1")) > 0) puts("data1 opened!\n");										
戻り値	正常終了の場合、ファイルハンドルとして正の値(>0)を返します。										
	エラーが発生した場合は-1 を返し、エラーコードがシステムグローバル変数 fErrorCode にセットされます。										
	<table> <tr> <th>エラーコード</th><th>説明</th></tr> <tr> <td>1</td><td>ファイル名が空文字列です。</td></tr> <tr> <td>4</td><td>指定されたファイル名は DAT ファイルではありません。</td></tr> <tr> <td>5</td><td>指定されたファイルは既にオープンされています。</td></tr> <tr> <td>6</td><td>システムで許可されている最大ファイル数を越えているため、ファイルを作成できません。</td></tr> </table>	エラーコード	説明	1	ファイル名が空文字列です。	4	指定されたファイル名は DAT ファイルではありません。	5	指定されたファイルは既にオープンされています。	6	システムで許可されている最大ファイル数を越えているため、ファイルを作成できません。
エラーコード	説明										
1	ファイル名が空文字列です。										
4	指定されたファイル名は DAT ファイルではありません。										
5	指定されたファイルは既にオープンされています。										
6	システムで許可されている最大ファイル数を越えているため、ファイルを作成できません。										
備考	<p>ファイルハンドルはファイル操作時のファイルを識別するために使用される正の整数です。</p> <p>ファイルがオープンされると、ファイルポインタはファイルの先頭にあります。</p>										
参照	close										

read	
目的	DAT ファイルから指定バイト数のデータを読み出します。
書式	int read (int fd, char* buffer, int count);
引数	int fd
	DAT ファイルのファイルハンドル。
	char* buffer
	読み出したデータを格納する変数へのポインタ。
引数	int count
	読み出すバイト数。読み込みを行える最大バイト数は、32,767 です。
コーディング例	if ((bytes_read = read(fd, buffer,80)) == -1) puts("read error!\n");
戻り値	正常終了の場合、実際に読み出したデータのバイト数を返します。
	エラーが発生した場合は-1 を返し、エラーコードがシステムグローバル変数 fErrorCode にセットされます。
エラーコード	説明
	2 fd で指定されたファイルが存在しません。
	4 fd で指定されたファイルは DAT ファイルではありません。
	7 ファイルハンドルが無効です。
	8 ファイルがオープンされていません。
	9 引数 count の値が不正です。
備考	引数で指定されたバイト数のデータを DAT ファイルの現在のファイルポインタ位置から読み出し、指定の変数に格納します。 ➤ この操作が正常終了すると、ファイルポインタは読み出したバイト数分移動します。
参照	readln, write, writeln

readln	
目的	DAT ファイルからデータ行コード (改行終端文字列)を読み出します。
書式	int readln(int fd, char* buffer, int max_count);
引数	int fd
	DAT ファイルのファイルハンドル。
	char* buffer
	読み出したデータを格納する変数へのポインタ。
引数	int count
	最大読取りバイト数。読み込みを行える最大バイト数は、32,767 です。
コーディング例	readln(fd, buffer,80);
戻り値	正常終了の場合、実際に読み出したデータのバイト数(改行を含む)を返します。
	エラーが発生した場合は-1 を返し、エラーコードがシステムグローバル変数 fErrorCode にセットされます。
エラーコード	説明
	2 fd で指定されたファイルが存在しません。
	4 fd で指定されたファイルは DAT ファイルではありません。
	7 ファイルハンドルが無効です。
	8 ファイルがオープンされていません。
	9 引数 count の値が不正です。
説明	DAT ファイルの Null で終わる文字列を読取り、指定された文字列バッファに格納します。データ行コードの読込みは、EOF に到達した場合、改行文字に到達した場合、指定された最大バイト数に到達した場合の何れかの条件が成立するまで行なわれます。 ➤ Null 文字が検出されるまで文字が読まれた場合、Null 文字もバッファに格納されます。戻り値にも加算されます。Null 文字に到達しなかった場合は、バッファに格納されるデータに Null 文字がない可能性があります。 ➤ この操作が正常終了すると、ファイルポインタは読み出したバイト数分移動します。
参照	read, write, writeln

tell											
目的	DAT ファイルのファイルポインタ位置を取得します。										
書式	long tell (int fd);										
引数	int fd DAT ファイルのファイルハンドル。										
コーディング例	current_position = tell (fd);										
戻り値	<p>正常終了の場合、現在のファイルポインタ位置を long 型の値で返します。  エラーが発生した場合は-1L を返し、エラーコードがシステムグローバル変数 fErrorCode にセットされます。</p> <table> <tr> <th>エラーコード</th><th>説明</th></tr> <tr> <td>2</td><td>fd で指定されたファイルが存在しません。</td></tr> <tr> <td>4</td><td>fd で指定されたファイルは DAT ファイルではありません。</td></tr> <tr> <td>7</td><td>ファイルハンドルが無効です。</td></tr> <tr> <td>8</td><td>ファイルがオープンされていません。</td></tr> </table>	エラーコード	説明	2	fd で指定されたファイルが存在しません。	4	fd で指定されたファイルは DAT ファイルではありません。	7	ファイルハンドルが無効です。	8	ファイルがオープンされていません。
エラーコード	説明										
2	fd で指定されたファイルが存在しません。										
4	fd で指定されたファイルは DAT ファイルではありません。										
7	ファイルハンドルが無効です。										
8	ファイルがオープンされていません。										
説明	<p>DAT ファイルのファイルポインタ位置は、ファイル先頭からのバイト数で表現されます。  ➤ 例えば、ファイルの先頭にファイルポインタがある場合は、戻り値 0L が返されます。</p>										
参照	lseek										

write															
目的	DAT ファイルに指定バイト数のデータを書込みます。														
書式	int write (int fd, char* buffer, int count);														
引数	int fd DAT ファイルのファイルハンドル。 char* buffer 書込むデータが格納されたポインタ。 int count 書込むバイト数。書込みを行える最大バイト数は、32,767 です。														
コーディング例	write (fd, data_buffer, 1024);														
戻り値	<p>正常終了の場合、実際に書込みを行ったバイト数を返します。  エラーが発生した場合は-1 を返し、エラーコードがシステムグローバル変数 fErrorCode にセットされます。</p> <table> <tr> <th>エラーコード</th><th>説明</th></tr> <tr> <td>2</td><td>fd で指定されたファイルが存在しません。</td></tr> <tr> <td>4</td><td>fd で指定されたファイルは DAT ファイルではありません。</td></tr> <tr> <td>7</td><td>ファイルハンドルが無効です。</td></tr> <tr> <td>8</td><td>ファイルがオープンされていません。</td></tr> <tr> <td>9</td><td>引数 count の値が不正です。</td></tr> <tr> <td>10</td><td>データを書込む空き容量が足りません。</td></tr> </table>	エラーコード	説明	2	fd で指定されたファイルが存在しません。	4	fd で指定されたファイルは DAT ファイルではありません。	7	ファイルハンドルが無効です。	8	ファイルがオープンされていません。	9	引数 count の値が不正です。	10	データを書込む空き容量が足りません。
エラーコード	説明														
2	fd で指定されたファイルが存在しません。														
4	fd で指定されたファイルは DAT ファイルではありません。														
7	ファイルハンドルが無効です。														
8	ファイルがオープンされていません。														
9	引数 count の値が不正です。														
10	データを書込む空き容量が足りません。														
備考	<p>引数で指定されたバイト数のデータを DAT ファイルに書込みます。  ➤ データの書込みは、現在のファイルポインタ位置から行われ、処理終了後、ファイルポインタは自動的に更新されます。  ➤ 処理中に EOF に到達した場合は、ファイルを自動的に拡張し、処理を完了させます。</p>														
参照	append, appendln, read, readln, writeln														

writeln															
目的	DAT ファイルにデータ(改行終端文字列)を書込みます。														
書式	int writeln (int fd, char* buffer);														
引数	int fd														
	DAT ファイルのファイルハンドル。														
	char* buffer														
	書込むデータが格納されたメモリ。														
コーディング例	writeln (fd, data_buffer);														
戻り値	正常終了の場合、実際に書き込みを行ったバイト数(改行文字を含む)を返します。														
	エラーが発生した場合は-1 を返し、エラーコードがシステムグローバル変数 fErrorCode にセットされます。														
	<table> <tr> <th>エラーコード</th><th>説明</th></tr> <tr> <td>2</td><td>fd で指定されたファイルが存在しません。</td></tr> <tr> <td>4</td><td>fd で指定されたファイルは DAT ファイルではありません。</td></tr> <tr> <td>7</td><td>ファイルハンドルが無効です。</td></tr> <tr> <td>8</td><td>ファイルがオープンされていません。</td></tr> <tr> <td>10</td><td>データを書込む空き容量が足りません。</td></tr> <tr> <td>11</td><td>引数 buffer に改行終端文字列がありません。</td></tr> </table>	エラーコード	説明	2	fd で指定されたファイルが存在しません。	4	fd で指定されたファイルは DAT ファイルではありません。	7	ファイルハンドルが無効です。	8	ファイルがオープンされていません。	10	データを書込む空き容量が足りません。	11	引数 buffer に改行終端文字列がありません。
エラーコード	説明														
2	fd で指定されたファイルが存在しません。														
4	fd で指定されたファイルは DAT ファイルではありません。														
7	ファイルハンドルが無効です。														
8	ファイルがオープンされていません。														
10	データを書込む空き容量が足りません。														
11	引数 buffer に改行終端文字列がありません。														
備考	<p>引数で指定されたバッファの Null で終わる文字列を DAT ファイルに書込みます。</p> <ul style="list-style-type: none"> <li>➤ Null 文字が検出されるまでの文字列がファイルに書込まれます。Null 文字もファイルに書込まれます。</li> <li>➤ データの書き込みは、現在のファイルポインタ位置から行われ、処理終了後、ファイルポインタは自動的に更新されます。</li> <li>➤ 処理中に EOF に到達した場合は、ファイルを自動的に拡張し、処理を完了させます。</li> </ul>														
参照	append, appendln, read, readln, write														

## 2.15.7 DBF ファイルと IDX ファイル

DBF ファイルと IDX ファイルは、データベースのプラットフォームを形成しています。

➤ DBF ファイルは、固定レコード長の構造を持っています。これは、レコード (メンバー) を格納するファイルです。一方、IDX ファイルは DBF ファイルに格納されている各レコードの位置の情報を保持するファイルで、いくつかの特定のキー値に従ってを再配置 (ソート) されています。

図書館を例に DBF と IDX ファイルを説明します。図書館内の特定の書籍を検索するとき、インデックスで検索します。本のタイトル、作家、出版社、ISBN 番号などのインデックスカテゴリで探して本を見つけ出すことができます。これらのインデックスエントリは、書籍の情報 (本のタイトル、作家、出版社、ISBN 番号など) に応じて昇順にソートされています。インデックス中に本を見つけた時、どこにその本が置いてあるかを知ることができます。

図書館の保管書籍は DBF ファイルに格納されているレコードに類似しており、様々なインデックスエントリは、その関連 IDX ファイルにあたります。レコードの一部の情報 (本のタイトル、作家、出版社、ISBN 番号など) IDX ファイルを作成するために使用されています。

### キー番号 (KEY NUMBER)

各 DBF ファイルは、最大 8 個の IDX ファイルを持つことができ、それら各々は、そのキー (インデックス) 番号によって識別されます。キー番号 (値は 1~8 の範囲内) は、IDX ファイルを作成したユーザー プログラムによって割り当てられます。

### キー値 (KEY VALUE)

レコードは、DBF ファイルから直接取得されるものでなく、関連する IDX ファイルを介して取得されます。

IDX ファイル (インデックスポイント) のファイルポイントは、DBF ファイルに格納されているレコードのポイントを表すものではなく、IDX ファイル内の特定のレコードのシーク番号を示しています。

add_member													
目的	DBF ファイルにレコード (メンバー) を追加します。												
書式	int add_member (int DBF_fd, char* member);												
引数	int DBF_fd												
	DBF ファイルのファイルハンドル。												
	char* member												
	追加するレコードが格納されたポイント。												
コーディング例	add_member(DBF_fd, member);												
戻り値	正常終了の場合 1 を返します。												
	エラーが発生した場合は 0 を返し、エラーコードがシステムグローバル変数 fErrorCode にセットされます。												
	<table><thead><tr><th>エラーコード</th><th>説明</th></tr></thead><tbody><tr><td>2</td><td>DBF_fd で指定されたファイルが存在しません。</td></tr><tr><td>4</td><td>DBF_fd で指定されたファイルは DBF ファイルではありません。</td></tr><tr><td>7</td><td>ファイルハンドルが無効です。</td></tr><tr><td>8</td><td>ファイルがオープンされていません。</td></tr><tr><td>10</td><td>レコードを追加する空き容量が足りません。</td></tr></tbody></table>	エラーコード	説明	2	DBF_fd で指定されたファイルが存在しません。	4	DBF_fd で指定されたファイルは DBF ファイルではありません。	7	ファイルハンドルが無効です。	8	ファイルがオープンされていません。	10	レコードを追加する空き容量が足りません。
エラーコード	説明												
2	DBF_fd で指定されたファイルが存在しません。												
4	DBF_fd で指定されたファイルは DBF ファイルではありません。												
7	ファイルハンドルが無効です。												
8	ファイルがオープンされていません。												
10	レコードを追加する空き容量が足りません。												
備考	引数で指定されたレコード (メンバー) を DBF ファイルに追加します。また、関係する IDX ファイルが存在する場合は、IDX ファイルのインデックスも更新します。 ➤ 引数で指定したレコードが、定義された DBF ファイル長 (create_DBF 関数を参照) より長い場合、レコードは、DBF ファイル長に切り詰められます。												
参照	create_DBF, delete_member												

close_DBF											
目的	DBF ファイル及びそれに関連する IDX ファイルを閉じます。										
書式	int close_DBF (int DBF_fd);										
引数	int DBF_fd DBF ファイルのファイルハンドル。										
コーディング例	if (close_DBF(DBF_fd)) send_lcds("DBF file closed!\n");										
戻り値	正常終了の場合 1 を返します。 エラーが発生した場合は 0 を返し、エラーコードがシステムグローバル変数 fErrorCode にセットされます。										
	<table> <tr> <th>エラーコード</th><th>説明</th></tr> <tr> <td>2</td><td>DBF_fd で指定されたファイルが存在しません。</td></tr> <tr> <td>4</td><td>DBF_fd で指定されたファイルは DBF ファイルではありません。</td></tr> <tr> <td>7</td><td>ファイルハンドルが無効です。</td></tr> <tr> <td>8</td><td>ファイルがオープンされていません。</td></tr> </table>	エラーコード	説明	2	DBF_fd で指定されたファイルが存在しません。	4	DBF_fd で指定されたファイルは DBF ファイルではありません。	7	ファイルハンドルが無効です。	8	ファイルがオープンされていません。
エラーコード	説明										
2	DBF_fd で指定されたファイルが存在しません。										
4	DBF_fd で指定されたファイルは DBF ファイルではありません。										
7	ファイルハンドルが無効です。										
8	ファイルがオープンされていません。										
備考	引数で指定されたファイルハンドルを持つ DBF ファイル及びそれに関連する全ての IDX ファイルを閉じます。 ➤ 引数で指定したデータコードが、定義された DBF ファイル長(create_DBF()を参照)より長い場合、データコードは、DBF ファイル長に切り詰められます。										
参照	open_DBF										

create_DBF											
目的	DBF ファイルを作成します。										
書式	int create_DBF (char* filename, int member_len);										
引数	char* filename DBF ファイル名(8 文字以内)が格納されたポインタ。 8 文字を超えるファイル名が指定された場合、8 文字に切り詰められます。 8200/8400/8700 シリーズで、SD カードに DBF ファイルを作成する場合、ファイル名を 250 バイト以下のフルパスで指定する必要があります。『2.16.2 デイクトリ』を参照してください。										
	int member_len DBF ファイルのレコード長(メンバー長)。 このレコード長を超えるデータは、切り詰められて DBF ファイルに書込まれます。										
コーディング例	if ((fd = create_DBF("data1",64))> 0) puts("data1 created!\n");										
戻り値	正常終了の場合、ファイルハンドルとして正の値(>0)を返します。 エラーが発生した場合は 0 を返し、エラーコードがシステムグローバル変数 fErrorCode にセットされます。										
	<table> <tr> <th>エラーコード</th><th>説明</th></tr> <tr> <td>1</td><td>ファイル名が空文字列です。</td></tr> <tr> <td>6</td><td>システムで許容されているファイルの最大数を超えているため、ファイルを作成できません。</td></tr> <tr> <td>9</td><td>引数 member_len の値が不正です。</td></tr> <tr> <td>12</td><td>引数 filename で指定されたファイルはすでに存在します。</td></tr> </table>	エラーコード	説明	1	ファイル名が空文字列です。	6	システムで許容されているファイルの最大数を超えているため、ファイルを作成できません。	9	引数 member_len の値が不正です。	12	引数 filename で指定されたファイルはすでに存在します。
エラーコード	説明										
1	ファイル名が空文字列です。										
6	システムで許容されているファイルの最大数を超えているため、ファイルを作成できません。										
9	引数 member_len の値が不正です。										
12	引数 filename で指定されたファイルはすでに存在します。										
備考	引数で指定されたファイル名、レコード長で DBF ファイルを作成し、以降の処理で使用するファイルハンドルを戻り値として返します。 ➤ ファイルハンドルはファイル操作するファイルを識別するための正の整数です。 ➤ DBF ファイル作成後、ユーザー定義のインデックスが作成されることがあります。										
参照	close_DBF, create_index, open_DBF										

## create\_index

**目的** 指定の DBF ファイルに対応する IDX ファイルを作成します。

**書式** int create\_index (int DBF\_fd, int key\_number, int key\_offset, int key\_len);

**引数** int DBF\_fd

DBF ファイルのファイルハンドル。

int key\_number

作成するインデックスファイルのキーナンバー(1~8)。

int key\_offset

キーデータの開始位置, オフセットアドレス(バイト)。

int key\_len

キーデータ長(サイズ)。

**コーディング例** create\_index (DBF\_fd,1,0,10);

**戻り値** 正常終了の場合 1 を返します。

エラーが発生した場合は 0 を返し、エラーコードがシステムグローバル変数 fErrorCode にセットされます。

エラーコード	説明
2	DBF_fd で指定されたファイルが存在しません。
4	DBF_fd で指定されたファイルは DBF ファイルではありません。
6	システムで許容されているファイルの最大数を超えているため、ファイルを作成できません。
7	ファイルハンドルが無効です。
8	ファイルがオープンされていません。
13	引数 key_number の値が不正です。
17	引数 key_offset の値が不正です。
18	DBF_fd で指定された DBF ファイルは空ではありません。
19	引数 key_number で指定された IDX ファイルは既に存在します。

**備考** 引数で指定された DBF ファイルに対応する IDX ファイルを与えられた条件を元に作成します。

- 引数 key\_offset には、キーデータの開始位置(オフセットアドレス)、key\_len には、キーデータ長(サイズ)をそれぞれバイト単位で指定します。(key\_offset と key\_len は、指定した DBF ファイルの member\_len 以下を指定します。)
- この create\_index 関数は、DBF ファイルに何のデータも書込まれていない状態、つまり空の状態の場合のみ呼び出すことができます。既に DBF ファイルにデータが書込まれている場合は、rebuild\_index 関数を代わりに使用します。

**参照** close\_DBF, rebuild\_index, remove\_index

delete_member																	
目的	DBF ファイルのデータレコード (メンバー) を削除します。																
書式	int delete_member (int DBF_fd, int key_number);																
引数	int DBF_fd																
	DBF ファイルのファイルハンドル。																
	int key_number																
	削除するデータレコード 位置を示す IDX ファイルのキーナンバー。																
コーディング 例	delete_member (DBF_fd, 1);																
戻り値	正常終了の場合 1 を返します。																
	エラーが発生した場合は 0 を返し、エラーコード がシステム グローバル変数 fErrorCode にセットされます。																
	<table> <tr> <th>エラーコード</th><th>説明</th></tr> <tr> <td>2</td><td>DBF_fd で指定されたファイルが存在しません。</td></tr> <tr> <td>4</td><td>DBF_fd で指定されたファイルは DBF ファイルではありません。</td></tr> <tr> <td>7</td><td>ファイルハンドルが無効です。</td></tr> <tr> <td>8</td><td>ファイルがオープンされていません。</td></tr> <tr> <td>13</td><td>引数 key_number の値が不正です。</td></tr> <tr> <td>14</td><td>キーナンバーで指定された IDX ファイルは存在しません。</td></tr> <tr> <td>16</td><td>DBF ファイルにデータレコード がありません。</td></tr> </table>	エラーコード	説明	2	DBF_fd で指定されたファイルが存在しません。	4	DBF_fd で指定されたファイルは DBF ファイルではありません。	7	ファイルハンドルが無効です。	8	ファイルがオープンされていません。	13	引数 key_number の値が不正です。	14	キーナンバーで指定された IDX ファイルは存在しません。	16	DBF ファイルにデータレコード がありません。
エラーコード	説明																
2	DBF_fd で指定されたファイルが存在しません。																
4	DBF_fd で指定されたファイルは DBF ファイルではありません。																
7	ファイルハンドルが無効です。																
8	ファイルがオープンされていません。																
13	引数 key_number の値が不正です。																
14	キーナンバーで指定された IDX ファイルは存在しません。																
16	DBF ファイルにデータレコード がありません。																
備考	引数で指定された IDX ファイルが指しているデータレコード を指定の DBF ファイルから削除します。																
参照	add_member, has_member																

get_member																	
目的	DBF ファイルのデータレコード (メンバー) を読み出します。																
書式	int get_member (int DBF_fd, int key_number, char* buffer);																
引数	int DBF_fd																
	DBF ファイルのファイルハンドル。																
	int key_number																
	読み出すデータレコード 位置を示す IDX ファイルのキーナンバー。																
	char* buffer																
	読み出したデータレコード を格納する変数へのポインタ。																
コーディング 例	if (get_member(DBF_fd,1,buffer) == 0) puts(buffer);																
戻り値	正常終了の場合 1 を返します。																
	エラーが発生した場合は 0 を返し、エラーコード がシステム グローバル変数 fErrorCode にセットされます。																
	<table> <tr> <th>エラーコード</th><th>説明</th></tr> <tr> <td>2</td><td>DBF_fd で指定されたファイルが存在しません。</td></tr> <tr> <td>4</td><td>DBF_fd で指定されたファイルは DBF ファイルではありません。</td></tr> <tr> <td>7</td><td>ファイルハンドルが無効です。</td></tr> <tr> <td>8</td><td>ファイルがオープンされていません。</td></tr> <tr> <td>13</td><td>引数 key_number の値が不正です。</td></tr> <tr> <td>14</td><td>キーナンバーで指定された IDX ファイルは存在しません。</td></tr> <tr> <td>16</td><td>DBF ファイルにデータレコード がありません。</td></tr> </table>	エラーコード	説明	2	DBF_fd で指定されたファイルが存在しません。	4	DBF_fd で指定されたファイルは DBF ファイルではありません。	7	ファイルハンドルが無効です。	8	ファイルがオープンされていません。	13	引数 key_number の値が不正です。	14	キーナンバーで指定された IDX ファイルは存在しません。	16	DBF ファイルにデータレコード がありません。
エラーコード	説明																
2	DBF_fd で指定されたファイルが存在しません。																
4	DBF_fd で指定されたファイルは DBF ファイルではありません。																
7	ファイルハンドルが無効です。																
8	ファイルがオープンされていません。																
13	引数 key_number の値が不正です。																
14	キーナンバーで指定された IDX ファイルは存在しません。																
16	DBF ファイルにデータレコード がありません。																
備考	引数で指定された IDX ファイルが指しているデータレコード を指定の DBF ファイルから読み出し、指定の変数へ格納します。																
参照	has_member																



has_member															
目的	IDX ファイルに特定キーデータが存在するかをチェックします。														
書式	int has_member (int DBF_fd, int key_number, char* key_value);														
引数	int DBF_fd														
	DBF ファイルのファイルハンドル。														
	int key_number														
	チェックする IDX ファイルのキーナンバー。														
	char* key_value														
	チェックするキーデータが格納された変数へのポインタ。														
コーディング例	if (has_member (DBF_fd, 1, "JOHN")) puts ("JOHN is on the name list!\n");														
戻り値	合致するデータが存在した場合は 1 を返し、存在しなかった場合は 0 を返します。														
	エラーが発生した場合は-1 を返し、エラーコードがシステムグローバル変数 fErrorCode にセットされます。														
<table border="1"> <thead> <tr> <th>エラーコード</th><th>説明</th></tr> </thead> <tbody> <tr> <td>2</td><td>DBF_fd で指定されたファイルが存在しません。</td></tr> <tr> <td>4</td><td>DBF_fd で指定されたファイルは DBF ファイルではありません。</td></tr> <tr> <td>7</td><td>ファイルハンドルが無効です。</td></tr> <tr> <td>8</td><td>ファイルがオープンされていません。</td></tr> <tr> <td>13</td><td>引数 key_number の値が不正です。</td></tr> <tr> <td>14</td><td>キーナンバーで指定された IDX ファイルは存在しません。</td></tr> </tbody> </table>		エラーコード	説明	2	DBF_fd で指定されたファイルが存在しません。	4	DBF_fd で指定されたファイルは DBF ファイルではありません。	7	ファイルハンドルが無効です。	8	ファイルがオープンされていません。	13	引数 key_number の値が不正です。	14	キーナンバーで指定された IDX ファイルは存在しません。
エラーコード	説明														
2	DBF_fd で指定されたファイルが存在しません。														
4	DBF_fd で指定されたファイルは DBF ファイルではありません。														
7	ファイルハンドルが無効です。														
8	ファイルがオープンされていません。														
13	引数 key_number の値が不正です。														
14	キーナンバーで指定された IDX ファイルは存在しません。														
備考	引数で指定されたキーデータが指定の IDX ファイルに存在するかをチェックします。														
	<ul style="list-style-type: none"> <li>➤ 合致するキーデータが存在した場合、合致した最初のキーデータにインデックスポインタがセットされます。</li> <li>➤ 複数のデータレコードが該当した場合、順番にインデックスポインタを移動させて、目的のデータレコードを探してください。合致するキーデータが無かった場合は、指定されたキーデータより大きい(下位の)キーデータにインデックスポインタがセットされます。</li> </ul>														
参照	get_member														

lseek_DBF	
目的	IDX ファイルのインデックスポインタを新しい位置へ移動します。
書式	long lseek_DBF (int DBF_fd, int key_number, long offset, int origin);
引数	int DBF_fd
	DBF ファイルのファイルハンドル。
	int key_number
	ターゲットとなる IDX ファイルのキーナンバー。
	long offset
	移動させる原点からのオフセット値(デ-ルコード 単位)。
	int origin
	移動の基準となる原点位置
	1
	IDX ファイルの先頭インデックス。
	0
	現インデックスポインタ位置
	-1
	IDX ファイルの終端インデックス。
コーディング例	lseek_DBF(DBF_fd, 1, 1L, 0); /* 次のデ-ルコードへ移動 */
戻り値	正常終了の場合、新しい位置を IDX ファイルの先頭インデックス位置からのオフセットアドレスで返します。
	エラーが発生した場合は-1 を返し、エラーコードがシステムグローバル変数 fErrorCode にセットされます。
	エラーコード
	説明
	2
	DBF_fd で指定されたファイルが存在しません。
	4
	DBF_fd で指定されたファイルは DBF ファイルではありません。
	7
	ファイルハンドルが無効です。
	8
	ファイルがオープンされていません。
	9
	引数 origin の値が不正です。
	13
	引数 key_number の値が不正です。
	14
	キーナンバーで指定された IDX ファイルは存在しません。
	15
	新しいインデックスポインタ位置がファイルの終端位置を越えています。
備考	引数で指定された原点位置及びオフセット値を元に IDX ファイルのインデックスポインタをデ-ルコード 単位で移動します。
参照	tell_DBF

member_in_DBF	
目的	DBF ファイルのデ-ルコード 数を取得します。
書式	long member_in_DBF (int DBF_fd);
引数	int DBF_fd
	DBF ファイルのファイルハンドル。
コーディング例	total_member = member_in_DBF(DBF_fd);
戻り値	正常終了の場合、デ-ルコード 数を long 型の値で返します。
	エラーが発生した場合は-1L を返し、エラーコードがシステムグローバル変数 fErrorCode にセットされます。
	エラーコード
	説明
	2
	DBF_fd で指定されたファイルが存在しません。
	4
	DBF_fd で指定されたファイルは DBF ファイルではありません。
	7
	ファイルハンドルが無効です。
	8
	ファイルがオープンされていません。

open_DBF											
目的	DBF ファイルを開きます。										
書式 引数	<pre>int open_DBF (char* filename);</pre> <p>char* filename</p> <p>DBF ファイル名(8 文字以内)が格納されたポインタ。        8 文字を超えるファイル名が指定された場合、8 文字に切り詰められます。        8200/8400/8700 シリーズで、SD カードに DBF ファイルを作成する場合、ファイル名を 250 バイト以下のフルパスで指定する必要があります。『2.16.2 ディレクトリ』を参照してください。</p>										
コーディング例	<pre>if ((fd = open_DBF("data1")) &gt; 0) puts("data1 opened!\n");</pre>										
戻り値	<p>正常終了の場合、ファイルハンドルとして正の値(&gt;0)を返します。        エラーが発生した場合は-1 を返し、エラーコードがシステムグローバル変数 fErrorCode にセットされます。</p> <table> <tr> <th>エラーコード</th><th>説明</th></tr> <tr> <td>1</td><td>ファイル名が空文字列です。</td></tr> <tr> <td>2</td><td>指定されたファイルが存在しません。</td></tr> <tr> <td>4</td><td>指定されたファイルは DBF ファイルではありません。</td></tr> <tr> <td>5</td><td>指定されたファイルは既にオープンされています。</td></tr> </table>	エラーコード	説明	1	ファイル名が空文字列です。	2	指定されたファイルが存在しません。	4	指定されたファイルは DBF ファイルではありません。	5	指定されたファイルは既にオープンされています。
エラーコード	説明										
1	ファイル名が空文字列です。										
2	指定されたファイルが存在しません。										
4	指定されたファイルは DBF ファイルではありません。										
5	指定されたファイルは既にオープンされています。										
備考	<p>引数で指定されたファイル名の DBF ファイル及びそれに関連する全ての IDX ファイルを開き、以降の処理で使用するファイルハンドルを戻り値として返します。ファイルが開かれると、全てのインデックスポインタは IDX ファイルの先頭位置にセットされます。        ➤ ファイルハンドルはファイル操作するファイルを識別するための正の整数です。</p>										
参照	close_DBF, create_DBF, create_index										

rebuild_index																									
目的	DBF ファイルの IDX ファイルを再ビルド (作成) します。																								
書式	int rebuild_index (int DBF_fd, int key_number, int base_index, int key_offset, int key_len);																								
引数	int DBF_fd																								
	DBF ファイルのファイルハンドル。																								
	int key_number																								
	再ビルド (作成) 後の IDX ファイルのキーナンバー (1~8)。 IDX ファイルがすでに存在する場合は、上書きされます。存在しない場合は、新たに作成されます。																								
	int base_index																								
	ベースとなるインデックスファイルのキーナンバー (1~8)。 ベースとなるインデックスファイルを指定せずに、インデックスファイルを作成したい場合は、引数に 0 を指定します。この場合、DBF ファイルの元のデータ順をベースにインデックスファイルが作成されます。																								
	int key_offset																								
	データの開始位置、オフセットアドレス (バイト)。																								
	int key_len																								
	データ長 (サイズ)。最大値は SRAM で 32767、SD カード で 1024																								
コーディング例	rebuild_index(DBF_fd,1,0,10);																								
戻り値	正常終了の場合 0 を返します。																								
	エラーが発生した場合は -1 を返し、エラーコードがシステムグローバル変数 fErrorCode にセットされます。																								
<table border="1"> <thead> <tr> <th>エラーコード</th><th>説明</th></tr> </thead> <tbody> <tr> <td>2</td><td>DBF_fd で指定されたファイルが存在しません。</td></tr> <tr> <td>4</td><td>DBF_fd で指定されたファイルは DBF ファイルではありません。</td></tr> <tr> <td>6</td><td>システムで許容されているファイルの最大数を超えているため、ファイルを作成できません。</td></tr> <tr> <td>7</td><td>ファイルハンドルが無効です。</td></tr> <tr> <td>8</td><td>ファイルがオープンされていません。</td></tr> <tr> <td>10</td><td>インデックスを再ビルドする空き容量が足りません。</td></tr> <tr> <td>13</td><td>引数 key_number の値が不正です。</td></tr> <tr> <td>14</td><td>キーナンバーで指定された IDX ファイルは存在しません。</td></tr> <tr> <td>17</td><td>引数 key_offset または key_len の値が不正です。</td></tr> <tr> <td>20</td><td>引数 base_index の値が不正です。</td></tr> <tr> <td>21</td><td>base_index は存在しません。</td></tr> </tbody> </table>		エラーコード	説明	2	DBF_fd で指定されたファイルが存在しません。	4	DBF_fd で指定されたファイルは DBF ファイルではありません。	6	システムで許容されているファイルの最大数を超えているため、ファイルを作成できません。	7	ファイルハンドルが無効です。	8	ファイルがオープンされていません。	10	インデックスを再ビルドする空き容量が足りません。	13	引数 key_number の値が不正です。	14	キーナンバーで指定された IDX ファイルは存在しません。	17	引数 key_offset または key_len の値が不正です。	20	引数 base_index の値が不正です。	21	base_index は存在しません。
エラーコード	説明																								
2	DBF_fd で指定されたファイルが存在しません。																								
4	DBF_fd で指定されたファイルは DBF ファイルではありません。																								
6	システムで許容されているファイルの最大数を超えているため、ファイルを作成できません。																								
7	ファイルハンドルが無効です。																								
8	ファイルがオープンされていません。																								
10	インデックスを再ビルドする空き容量が足りません。																								
13	引数 key_number の値が不正です。																								
14	キーナンバーで指定された IDX ファイルは存在しません。																								
17	引数 key_offset または key_len の値が不正です。																								
20	引数 base_index の値が不正です。																								
21	base_index は存在しません。																								
備考	この関数は、DBF ファイル(DBF_fd)に関連付けられている IDX ファイルを再構築または作成します。IDX ファイルのキーフィールドは key_offset と key_len で指定します。																								
	<ul style="list-style-type: none"> <li>➤ base_index はこの関数が、新しい IDX ファイルを構築するのに元となる IDX ファイルの番号を指定します。例えば、レポートが日付、部門、ID 番号のシーケンスで生成されていて、日付と部門のデータが繰り返される場合。これは、最初に ID 番号のインデックスを再構築することで行われます。その後、ベースのインデックスとして ID 番号のインデックスを持つ部門のインデックスを作成します。最後に、ベースのインデックスとして部門のインデックスを持つ日付のインデックスを作成します。日付インデックスのメジャーの順序の結果は、日付、部門、ID 番号となります。</li> <li>➤ create_DBF() で定義されているキーフィールドは member_len の値以内でなければなりません。つまり、key_offset+key_len の値が member_len を超えることはできません。</li> </ul>																								
参照	create_index, remove_index																								

remove_index	
目的	IDX ファイルを削除します。
書式	int remove_index(int DBF_fd, int key_number);
引数	int DBF_fd
	DBF ファイルのファイルハンドル。
引数	int key_number
	削除する IDX ファイルのキーナンバー。
コーディング例	if (remove_index(DBF_fd, 1)) puts ("index removed!\n");
戻り値	正常終了の場合 1 を返します。
	エラーが発生した場合は 0 を返し、エラーコードがシステムグローバル変数 fErrorCode にセットされます。
	エラーコード
	説明
	2 DBF_fd で指定されたファイルが存在しません。
	4 DBF_fd で指定されたファイルは DBF ファイルではありません。
	7 ファイルハンドルが無効です。
	8 ファイルがオープンされていません。
	10 空き容量が足りません。
	13 引数 key_number の値が不正です。
	14 キーナンバーで指定された IDX ファイルは存在しません。
参照	create_index, rebuild_index

tell_DBF	
目的	IDX ファイルのインデックスファイル位置を取得します。
書式	long tell_DBF (int DBF_fd, int key_number);
引数	int DBF_fd
	DBF ファイルのファイルハンドル。
引数	int key_number
	チェックするインデックスファイルのキーナンバー(1~8)
コーディング例	rank_number = tell_DBF(DBF_fd, 1);
戻り値	正常終了の場合、現在のインデックスファイル位置を long 型の値で返します。
	エラーが発生した場合は-1 を返し、エラーコードがシステムグローバル変数 fErrorCode にセットされます。
	エラーコード
	説明
	2 DBF_fd で指定されたファイルが存在しません。
	4 DBF_fd で指定されたファイルは DBF ファイルではありません。
	7 ファイルハンドルが無効です。
	8 ファイルがオープンされていません。
	13 引数 key_number の値が不正です。
	14 キーナンバーで指定された IDX ファイルは存在しません。
備考	引数で指定された IDX ファイルの現在のインデックスファイル位置を戻り値として返します。 ➤ IDX ファイルのインデックスファイル位置は、整列順の番号で表現され、最初のインデックスファイルがある場合は、戻り値 1L が返されます。
参照	lseek_DBF

<b>UnpackDBF</b>	<b>8000,8200,8300,8400,8700</b>
------------------	---------------------------------

目的	PC の DataConverter.exe で作成した DBF ファイルを解凍します。										
書式	int UnpackDBF (const char *filenameSource);										
引数	const char *filenameSource 元ファイル名が格納された変数へのポインタ。										
コーディング例 1	unpack_file_count = UnpackDBF("packdata"); // File stored in SRAM										
コーディング例 2	unpack_file_count = UnpackDBF("A:\\packdata"); // File stored on SD(8200/8400/8700)										
戻り値	正常終了の場合、解凍した DBF ファイル数を返します。 エラーが発生した場合は 0 を返し、エラーコードがシステムグローバル変数 fErrorCode にセットされます。										
	<table border="1"> <thead> <tr> <th>エラーコード</th><th>説明</th></tr> </thead> <tbody> <tr> <td>2</td><td>元ファイルが SRAM に存在しません。</td></tr> <tr> <td>4</td><td>元ファイルのフォーマットが不正です。</td></tr> <tr> <td>10</td><td>SRAM に空き容量がありません。</td></tr> <tr> <td>31</td><td>SD カードにある元ファイルを開けませんでした。さらなる詳細はシステムグローバル変数 ferrno を参照してください。</td></tr> </tbody> </table>	エラーコード	説明	2	元ファイルが SRAM に存在しません。	4	元ファイルのフォーマットが不正です。	10	SRAM に空き容量がありません。	31	SD カードにある元ファイルを開けませんでした。さらなる詳細はシステムグローバル変数 ferrno を参照してください。
エラーコード	説明										
2	元ファイルが SRAM に存在しません。										
4	元ファイルのフォーマットが不正です。										
10	SRAM に空き容量がありません。										
31	SD カードにある元ファイルを開けませんでした。さらなる詳細はシステムグローバル変数 ferrno を参照してください。										
備考	<p>ハンディターミナルに RS-232 または FTP 経由でダウンロードし、SRAM または SD カードに保存する前に、PC のユーティリティ "DataConverter.exe" で DBF ファイルを圧縮したファイルを作成します。ハンディターミナルで UnpackDBF をコールすると、ファイルを解凍します。</p> <p>➤ SRAM に保存されているファイルを解凍すると、元の圧縮ファイルは処理完了後、自動的に削除されます。</p>										

<b>update_member</b>
----------------------

目的	DBF ファイルのデータレコード (メンバー) を更新(上書き)します。																
書式	int update_member(int DBF_fd, int key_number, char* member);																
引数	int DBF_fd DBF ファイルのファイルハンドル。 int key_number 更新するデータレコード位置を指しているインデックスファイルのキーナンバー(1~8) char* member 上書きするデータが格納された変数へのポインタ																
コーディング例	update_member (DBF_fd, 1,10);																
戻り値	正常終了の場合 1 を返します。 エラーが発生した場合は 0 を返し、エラーコードがシステムグローバル変数 fErrorCode にセットされます。																
	<table border="1"> <thead> <tr> <th>エラーコード</th><th>説明</th></tr> </thead> <tbody> <tr> <td>2</td><td>DBF_fd で指定されたファイルが存在しません。</td></tr> <tr> <td>4</td><td>DBF_fd で指定されたファイルは DBF ファイルではありません。</td></tr> <tr> <td>7</td><td>ファイルハンドルが無効です。</td></tr> <tr> <td>8</td><td>ファイルがオープンされていません。</td></tr> <tr> <td>13</td><td>引数 key_number の値が不正です。</td></tr> <tr> <td>14</td><td>キーナンバーで指定された IDX ファイルは存在しません。</td></tr> <tr> <td>16</td><td>DBF ファイルにデータレコードがありません。</td></tr> </tbody> </table>	エラーコード	説明	2	DBF_fd で指定されたファイルが存在しません。	4	DBF_fd で指定されたファイルは DBF ファイルではありません。	7	ファイルハンドルが無効です。	8	ファイルがオープンされていません。	13	引数 key_number の値が不正です。	14	キーナンバーで指定された IDX ファイルは存在しません。	16	DBF ファイルにデータレコードがありません。
エラーコード	説明																
2	DBF_fd で指定されたファイルが存在しません。																
4	DBF_fd で指定されたファイルは DBF ファイルではありません。																
7	ファイルハンドルが無効です。																
8	ファイルがオープンされていません。																
13	引数 key_number の値が不正です。																
14	キーナンバーで指定された IDX ファイルは存在しません。																
16	DBF ファイルにデータレコードがありません。																
備考	引数で指定された IDX ファイルが指しているデータレコードを指定データで更新(上書き)します。																
参照	has_member																

## 2.15.8 SD 経由のファイル転送

8200/8400/8700 シリーズでの SD カードの詳細は『2.16 SD カード』を参照してください。

RAMtoSD_DAT	8200,8400,8700
-------------	----------------

**目的** SRAM から SD カードへ DAT ファイルをコピーします。

**書式** int RAMtoSD\_DAT(const char \*filenameRAM, const char \*filenameSD int mode);

**引数**

const char \*filenameRAM  
 コピー元ファイル名が格納された変数へのポインタ。  
 8 文字を超えるファイル名が指定された場合、8 文字に切り詰められます。

const char \*filenameSD  
 コピー先ファイル名が格納された変数へのポインタ。  
 ファイル名を 250 バイト以下の範囲で指定する必要があります。『2.16.2 デイクトリ』を参照してください。

int mode

0	コピー元ファイルを削除。
1	コピー元ファイルを残す。

**コーディング例**

```
const static char SrcDat[] = "data1";
const static char TarDAT[] = "A:\\XACT\\data1.dat";
printf("Copy the file to SD card...");
fremove(TarDAT); // remove target if it exists
if (!i = RAMtoSD_DAT((void *)SrcDAT, (void *)TarDAT, 0)) {
    printf("\n Fail ErrorCode=%d\n", read_error_code());
    while(1);
}
printf("Done! File %s on SD card is created\n", TarDAT);
```

**戻り値** 正常終了の場合 1 を返します。  
 エラーが発生した場合は 0 を返し、エラーコードがシステムグローバル変数 fErrorCode にセットされます。

エラーコード	説明
1	コピー元/コピー先ファイル名が不正です。
2	コピー元ファイルが存在しません。
4	コピー元ファイルは DAT ファイルではありません。
5	コピー元ファイルが開かれています。
10	SD カードに空き容量がありません。
32	コピー先ファイルを作成できません。詳細はシステムグローバル変数 ferrno を参照してください。
33	コピー先ファイルに書込めません。詳細はシステムグローバル変数 ferrno を参照してください。

**備考** コピー元 DAT ファイルはこの処理をコールする前に閉じておいてください。コピー先ファイルが既にある場合、ファイルは上書きされます。ない場合は、新規に作成されます。

**参照** SDtoRAM\_DAT, SDtoRAM\_DBF, RAMtoSD\_DBF

SDtoRAM_DAT		8200,8400,8700
目的	SD から SRAM カード へ DAT ファイルをコピーします。	
書式	int SDtoRAM_DAT(const char *filenameSD, const char *filenameRAM int mode);	
引数	const char *filenameSD	コピー元ファイル名が格納された変数へのポインタ。 ファイル名を 250 バイト以下の範囲で指定する必要があります。『2.16.2 デバイスリ』を参照してください。
	const char *filenameRAM	コピー先ファイル名が格納された変数へのポインタ。 8 文字を超えるファイル名が指定された場合、8 文字に切り詰められます。
	int mode	
	0	コピー元ファイルを削除。
コーディング例	1	コピー元ファイルを残す。
	<pre>const static char SrcDat[] = "A:\\XACT\\data2.dat"; const static char TarDAT[] = "data2"; printf("Copy the file to RAM..."); remove(TarDAT); // remove target if it exists if (!i = SDtoRAM_DAT((void *)SrcDAT, (void *)TarDAT, 1))) {     printf("\n Fail ErrorCode=%d\r", read_error_code());     while(1); } printf("Done! File %s in RAM is created\r\n", TarDAT);</pre>	
	<p>戻り値</p> <p>正常終了の場合 1 を返します。 エラーが発生した場合は 0 を返し、エラーコードがシステムグローバル変数 fErrorCode にセットされます。</p>	
	エラーコード	説明
備考	1	コピー元/コピー先ファイル名が不正です。
	6	システムで許容されているファイルの最大数を超過しているため、ファイルを作成できません。
	10	SD カードに空き容量がありません。
	31	SD カードにある元ファイルを開けませんでした。さらなる詳細はシステムグローバル変数 ferrno を参照してください。
参照	RAMtoSD_DAT, SDtoRAM_DBF, RAMtoSD_DBF	



RAMtoSD_DBF	8200,8400,8700
-------------	----------------

**目的** SRAM から SD カード へ DBF ファイルと関連する IDX ファイルをコピーします。

**書式** int RAMtoSD\_DBF(const char \*filenameRAM, const char \*filenameSD int mode);

**引数**

const char *filenameRAM	
コピー元ファイル名が格納された変数へのポインタ。 8 文字を超えるファイル名が指定された場合、8 文字に切り詰められます。	
const char *filenameSD	
コピー先ファイル名が格納された変数へのポインタ。 ファイル名を 250 バイト以下の範囲で指定する必要があります。『2.16.2 デバイクトリ』を参照してください。	
int mode	
0	コピー元ファイルを削除。
1	コピー元ファイルを残す。

**コーディング例**

```
const static char dbfname2[] = "RAMdbf1";
const static char dbfname3[] = "A:\\XACT\\SDdbf2";
printf("Copy the file to SD card...");
fremove(dbfname3); // remove target if it exists
if (!i = RAMtoSD_DBF((void *)dbfname2, (void *)dbfname3, 0)) {
    printf("\n Fail ErrorCode=%d\r", read_error_code());
    while(1);
}
printf("Done! File %s on SD card is created\r\n", dbfname3);
```

**戻り値** 正常終了の場合 1 を返します。  
エラーが発生した場合は 0 を返し、エラーコードがシステムグローバル変数 fErrorCode にセットされます。

エラーコード	説明
1	コピー元/コピー先ファイル名が不正です。
4	コピー元ファイルは DBF ファイルではありません。
5	コピー元ファイルが開かれています。
6	システムで許容されているファイルの最大数を超過しているため、ファイルを作成できません。
10	空き容量がありません。

**備考** コピー元 DBF ファイルはこの処理をコールする前に閉じておいてください。コピー先ファイルが既にある場合、ファイルは上書きされます。ない場合は、新規に作成されます。

**参照** RAMtoSD\_DAT, SDtoRAM\_DAT, SDtoRAM\_DBF

SDtoRAM_DBF		8200,8400,8700
目的	SD から SRAM カード へ DBF ファイルと関連する IDX ファイルをコピーします。	
書式	int SDtoRAM_DBF(const char *filenameSD, const char *filenameRAM int mode);	
引数	const char *filenameSD	コピー元ファイル名が格納された変数へのポインタ。 ファイル名を 250 バイト以下の範囲で指定する必要があります。『2.16.2 デバイス』を参照してください。
	const char *filenameRAM	コピー先ファイル名が格納された変数へのポインタ。 8 文字を超えるファイル名が指定された場合、8 文字に切り詰められます。
	int mode	
	0	コピー元ファイルを削除。
	1	コピー元ファイルを残す。
コーディング例	<pre> const static char dbfname1[] = "A:\\SDdbf1"; const static char dbfname2[] = "RAMdbf1"; printf("Copy the file to RAM..."); remove(dbfname2); // remove target if it exists if (!i = SDtoRAM_DBF((void *)dbfname1, (void *)dbfname2, 1))) {     printf("\n Fail ErrorCode=%d\r", read_error_code());     while(1); } printf("Done! File %s in RAM is created\r\n", dbfname2); </pre>	
戻り値	正常終了の場合 1 を返します。 エラーが発生した場合は 0 を返し、エラーコードがシステムグローバル変数 fErrorCode にセットされます。	
	エラーコード	説明
	1	コピー元/コピー先ファイル名が不正です。
	4	コピー元ファイルは DBF ファイルではありません。
	5	コピー元ファイルが開かれています。
	6	システムで許容されているファイルの最大数を超過しているため、ファイルを作成できません。
	10	空き容量がありません。
備考	コピー元 DBF ファイルはこの処理をコールする前に閉じておいてください。コピー先ファイルが既にある場合、ファイルは上書きされます。ない場合は、新規に作成されます。	
参照	RAMtoSD_DAT, RAMtoSD_DBF, SDtoRAM_DAT	

## 2.16 SD カード

SD カード は、提供されている関数を用いたアプリケーションで使用することで、直接アクセスすることができます。また、8200/8400/8700 シリーズでは、SD カード をハンディターミナルに搭載し、USB ケーブル経由でコンピュータに接続した場合、プログラミング または System Menu で、SD Card Menu → Run as USB Disk を選択すると、リムーバブルディスク(USB マスストレージデバイス)として扱うことができます。Part II の『9. USB 接続』と『2.16.6 マスストレージデバイス』を参照ください。詳細情報は『2.14.3 SD カード』を参照してください。

※ COM5 をマスストレージ (SetcommType に COMM\_USBDISK の引数)で使用中は、ハンディターミナルから SD カード に直接アクセスできません。

### SD での DAT ファイルの直接アクセス

➤ 任意のディレクトリ下に置くことができ、SD カード 上の DAT ファイルにアクセスするには『2.16.5 SD カード 操作』で提供される関数を使用します。ファイル名はフルパスで指定する必要があります。

※ 最大 32 のファイルと 3 つのフォルダを同時に開くことができますが、ファイルハンドルの枯渇を防ぐためにも不要となったファイルやフォルダを閉じることをお勧めします。

### SD での DBF ファイルの直接アクセス

➤ 任意のディレクトリ下に置くことができ、SD カード 上の DBF ファイルにアクセスするには『2.15.7 DBF ファイルと IDX ファイル』で提供される関数を使用します。ファイル名は拡張子を除いてフルパスで指定する必要があります。ファイル名は拡張子を除いてフルパスで指定する必要があります。DBF ファイルが作成されるとき、DBF ファイルは".DB0"の拡張子となり、IDX ファイルは".DB1"～".DB8"の拡張子となります。

➤ SRAM から SD カード またはその反対の処理には『2.15.8 SD 経由のファイル転送』で提供されている関数を使用します。元 DBF ファイルは予め閉じておく必要があります。

### USB マスストレージデバイス

マスストレージ使用中は

- (1) 開いているすべてのファイルは自動的に閉じられます。
- (2) 『2.16.5 SD カード 操作』にあるどの処理も close\_com(5)の前にコールされた場合、SD カード はマスストレージ にデバイスとして使用中であることを意味する"E\_SD\_OCCUPIED"のエラーコードが戻ります。

## 2.16.1 ファイルシステム

FAT12/FAT16/FAT32 をサポートしています。プログラミング、または System Menu の SD Card Menu → Access SD Card でカードをフォーマットすることができます。format()はコールされると、カード容量によって自動的に FAT フォーマットを決定します。

カード 容量	FAT フォーマット	クラスあたりのセクタ数
≤ 32MB	FAT12	32
≤ 1GB	FAT16	32
≤ 2GB	FAT16	64
≤ 8GB	FAT32	8

## 2.16.2 デレクトリ

SRAM 上のファイルシステムとは異なり、SD カード 上のファイルシステムは、階層ツリーのデレクトリ構造を特徴とし、サブデレクトリを作成することができます。

いくつかのデレクトリは、特定の用途のために予約されています。

予約ディレクトリ	関連する機能	備考																																																																			
\\Program	<div>➢ System Menu → Load Program</div> <div>➢ Program Manager → Download</div> <div>➢ Program Manager → Activate</div> <div>➢ Kernel Menu → Load Program</div> <div>➢ Kernel Menu → Kernel Update</div> <div>➢ UPDATE_BASIC()</div>	<div>このフォルダに保存しているプログラムをハンディターミナルにダウンロードすることができます。</div> <div>➢ C プログラム・・・*.shx</div> <div>➢ BASIC プログラム・・・*.ini と+.syn</div>																																																																			
\\BasicRun	BASIC ランタイム	<div>BASIC ランタイムで作成およびアクセスされる DAT、DBF ファイルが保存されています。</div> <div>それらのファイル名は次の通りです。</div> <div>DAT ファイル名</div> <table><tr><td>DAT file #1</td><td>TXACT1.DAT</td></tr><tr><td>DAT file #2</td><td>TXACT2.DAT</td></tr><tr><td>DAT file #3</td><td>TXACT3.DAT</td></tr><tr><td>DAT file #4</td><td>TXACT4.DAT</td></tr><tr><td>DAT file #5</td><td>TXACT5.DAT</td></tr><tr><td>DAT file #6</td><td>TXACT6.DAT</td></tr></table> <div>DBF ファイル名</div> <table><tr><td rowspan="5">DBF file #1</td><td>レコード ファイル</td><td>F1.DB0</td></tr><tr><td>デフォルトインデックスファイル</td><td>F1.DB1</td></tr><tr><td>インデックスファイル#1</td><td>F1.DB2</td></tr><tr><td>インデックスファイル#2</td><td>F1.DB3</td></tr><tr><td>インデックスファイル#3</td><td>F1.DB4</td></tr><tr><td rowspan="5">DBF file #2</td><td>レコード ファイル</td><td>F2.DB0</td></tr><tr><td>デフォルトインデックスファイル</td><td>F2.DB1</td></tr><tr><td>インデックスファイル#1</td><td>F2.DB2</td></tr><tr><td>インデックスファイル#2</td><td>F2.DB3</td></tr><tr><td>インデックスファイル#3</td><td>F2.DB4</td></tr><tr><td rowspan="5">DBF file #3</td><td>レコード ファイル</td><td>F3.DB0</td></tr><tr><td>デフォルトインデックスファイル</td><td>F3.DB1</td></tr><tr><td>インデックスファイル#1</td><td>F3.DB2</td></tr><tr><td>インデックスファイル#2</td><td>F3.DB3</td></tr><tr><td>インデックスファイル#3</td><td>F3.DB4</td></tr><tr><td rowspan="5">DBF file #4</td><td>レコード ファイル</td><td>F4.DB0</td></tr><tr><td>デフォルトインデックスファイル</td><td>F4.DB1</td></tr><tr><td>インデックスファイル#1</td><td>F4.DB2</td></tr><tr><td>インデックスファイル#2</td><td>F4.DB3</td></tr><tr><td>インデックスファイル#3</td><td>F4.DB4</td></tr><tr><td rowspan="5">DBF file #5</td><td>レコード ファイル</td><td>F5.DB0</td></tr><tr><td>デフォルトインデックスファイル</td><td>F5.DB1</td></tr><tr><td>インデックスファイル#1</td><td>F5.DB2</td></tr><tr><td>インデックスファイル#2</td><td>F5.DB3</td></tr><tr><td>インデックスファイル#3</td><td>F5.DB4</td></tr></table>	DAT file #1	TXACT1.DAT	DAT file #2	TXACT2.DAT	DAT file #3	TXACT3.DAT	DAT file #4	TXACT4.DAT	DAT file #5	TXACT5.DAT	DAT file #6	TXACT6.DAT	DBF file #1	レコード ファイル	F1.DB0	デフォルトインデックスファイル	F1.DB1	インデックスファイル#1	F1.DB2	インデックスファイル#2	F1.DB3	インデックスファイル#3	F1.DB4	DBF file #2	レコード ファイル	F2.DB0	デフォルトインデックスファイル	F2.DB1	インデックスファイル#1	F2.DB2	インデックスファイル#2	F2.DB3	インデックスファイル#3	F2.DB4	DBF file #3	レコード ファイル	F3.DB0	デフォルトインデックスファイル	F3.DB1	インデックスファイル#1	F3.DB2	インデックスファイル#2	F3.DB3	インデックスファイル#3	F3.DB4	DBF file #4	レコード ファイル	F4.DB0	デフォルトインデックスファイル	F4.DB1	インデックスファイル#1	F4.DB2	インデックスファイル#2	F4.DB3	インデックスファイル#3	F4.DB4	DBF file #5	レコード ファイル	F5.DB0	デフォルトインデックスファイル	F5.DB1	インデックスファイル#1	F5.DB2	インデックスファイル#2	F5.DB3	インデックスファイル#3	F5.DB4
DAT file #1	TXACT1.DAT																																																																				
DAT file #2	TXACT2.DAT																																																																				
DAT file #3	TXACT3.DAT																																																																				
DAT file #4	TXACT4.DAT																																																																				
DAT file #5	TXACT5.DAT																																																																				
DAT file #6	TXACT6.DAT																																																																				
DBF file #1	レコード ファイル	F1.DB0																																																																			
	デフォルトインデックスファイル	F1.DB1																																																																			
	インデックスファイル#1	F1.DB2																																																																			
	インデックスファイル#2	F1.DB3																																																																			
	インデックスファイル#3	F1.DB4																																																																			
DBF file #2	レコード ファイル	F2.DB0																																																																			
	デフォルトインデックスファイル	F2.DB1																																																																			
	インデックスファイル#1	F2.DB2																																																																			
	インデックスファイル#2	F2.DB3																																																																			
	インデックスファイル#3	F2.DB4																																																																			
DBF file #3	レコード ファイル	F3.DB0																																																																			
	デフォルトインデックスファイル	F3.DB1																																																																			
	インデックスファイル#1	F3.DB2																																																																			
	インデックスファイル#2	F3.DB3																																																																			
	インデックスファイル#3	F3.DB4																																																																			
DBF file #4	レコード ファイル	F4.DB0																																																																			
	デフォルトインデックスファイル	F4.DB1																																																																			
	インデックスファイル#1	F4.DB2																																																																			
	インデックスファイル#2	F4.DB3																																																																			
	インデックスファイル#3	F4.DB4																																																																			
DBF file #5	レコード ファイル	F5.DB0																																																																			
	デフォルトインデックスファイル	F5.DB1																																																																			
	インデックスファイル#1	F5.DB2																																																																			
	インデックスファイル#2	F5.DB3																																																																			
	インデックスファイル#3	F5.DB4																																																																			
\\AG\\DBF \\AG\\DAT \\AG\\EXPORT \\AG\\IMPORT	アプリケーションジェネレータ(別名、AG)	アプリケーションジェネレータで作成および/またはアクセスされる DAT、DBF、参照ファイルが保存されています。																																																																			

以下に示すように、ファイル名は関数呼び出しに渡される引数として必要な場合は、フルパスで指定する必要があります。

ファイルパス	ルートディレクトリにあるファイル	サブディレクトリにあるファイル
A:\...	A:\UserFile	A:\SubDir\UserFile
a:\...	a:\UserFile	a:\SubDir\UserFile
A:/...	A:/UserFile	A:/SubDir/UserFile
a:/...	a:/UserFile	a:/SubDir/UserFile

(1) DAT ファイルは、ファイル名に拡張子があってもなくても有効です。

(2) DBF ファイルはファイル名に拡張子は不要です。

## 2.16.3 ファイル名

ファイル名は最大 8 文字まで、ファイルの拡張子は最大 3 文字まで - ファイル名は 8.3 フォーマット(=短いファイル名)となります。

「"」「\*」「+」「,」「:」「;」「<」「=」「>」「?」「|」「[」「]」はファイル名に使用できません。

➤ 1～8 文字のファイル名(null 文字は不可)が表示され、拡張子があればそれ也表示されます。ファイル名が 8 文字以上の場合、8 文字に切り詰められます。

➤ マストロージデバイスとして SD カードを搭載したハンディターミナルの場合、長いファイル名では、最大で 255 文字まで可能です。例えば、パソコンで作成したファイルのファイル名は、"123456789.txt"である場合もあります。ただし、同じファイルを直接ハンディターミナルでアクセスすると、ファイル名は"123456~1.TXT"に切り詰められます。

➤ ファイル名に ASCII 文字以外がある場合、ハンディターミナルがファイル名を正しく表示するためには、ハンディターミナルに該当のフォントファイルをダウンロードしておく必要があります。

➤ ファイル名はアルファベットの大文字・小文字を区別しません。

## 2.16.4 FILEINFO 構造体

ファイルまたはディレクトリ情報にアクセスする fgetinfo()と freaddir()で使います。

```
typedef struct {
    char fname[13];
    unsigned char fattrib;
    unsigned int ftime;
    unsigned int fdate;
    unsigned long fsize;
}FILEINFO
```

char fname[13]	ファイル名は 8.3 フォーマット(=短いファイル名)となります。(ファイル名は 8 文字まで、拡張子は 3 文字まで)	
unsigned char fattrib	ファイル属性	
	0x01	読取り専用
	0x02	隠しファイル
	0x04	システムファイル
	0x08	ボリューム ID
	0x10	ディレクトリ
	0x20	アーカイブ
unsigned int ftime	更新時間(16ビットフィールド)	
	Bit 0 ～ 4	秒(2 秒で 1 増加) ➤ 0～29 で 0～58 を表現
	Bit 5 ～ 10	分 ➤ 0～59
	Bit 11 ～ 15	時 ➤ 0～23
unsigned int fdate	更新日時(16ビットフィールド)	
	Bit 0 ～ 4	日 ➤ 0～31
	Bit 5 ～ 8	月 ➤ 1～12
	Bit 9 ～ 15	1980 年からの経過年 ➤ 0～127 で 1980～2107 を表現
unsigned long fsize	ファイルサイズ (バイト単位)	

## 2.16.5 SD カード 操作

<b>chmod</b>	<b>8200,8400,8700</b>
--------------	-----------------------

**目的** ファイルの属性を変更します。

**書式** int chmod (const char \*filename, int attribute);

**引数** const char \*filename

属性を変更するファイル名が格納された変数へのポインタ。

int attribute

新しいファイル属性。以下のリストから 1 つまたは複数の属性を指定。

0x00	FA_NOR	通常ファイル
0x01	FA_RDO	読取り専用
0x02	FA_HID	隠しファイル(アクセスは可能です)
0x04	FA_SYS	システムファイル
0x20	FA_ARC	アーカイブビット(前回、フラグがクリアされた後にファイルが変更されていることを示す)

**コーディング例**

```
int att;
att = chmod ("a:¥¥myfile.bin", FA_SYS | FA_RDO)
if (att == EOF)
    printf ("Chmod error, a:¥¥myfile.bin¥n");
```

**戻り値** 正常終了すると新しい属性値を返します。エラーが発生した場合は-1 を返し、システムグローバル変数 errno に発生したエラー状態がセットされます。

**備考** 引数"filename"で指定されたファイルの属性を変更します。ファイル名はルパスの 8.3 フォーマットで指定します。

**参照** chmodfp

chmodfp		8200,8400,8700
目的	ファイルハンドルで指定されたファイルの属性を変更します。	
書式	int chmodfp (int fd, int function, int attribute);	
引数	int fd	対象ファイルのファイルハンドル。
	int function	
	0	ファイル属性を戻り値として返す
	1	新しいファイル属性をセット
	int attribute	新しいファイル属性。以下のリストから 1 つまたは複数の属性を指定。
	0x00	FA_NOR 通常ファイル
	0x01	FA_RDO 読取り専用
0x02	FA_HID 隠しファイル(アクセスは可能です)	
0x04	FA_SYS システムファイル	
0x20	FA_ARC アーカイブビット(前回、フラグがクリアされた後にファイルが変更されていることを示す)	
コーディング例	<pre>FILE *fp; Int att; fp = fopen ("A:¥¥MYFILE.BIN", "r+b") att = chmodfp (fp, 1, FA_SYS   FA_RDO); if (att == EOF)     printf ("Chmod error, a:¥¥myfile.bin¥n");</pre>	
戻り値	正常終了すると新しい属性値を返します。エラーが発生した場合は-1 を返し、システムグローバル変数 ferrno に発生したエラー状態がセットされます。	
備考	ファイルハンドルで指定されたファイルの属性を変更します。変更したファイル属性は、ファイルを一度クローズし、再オープンした時点で有効となります。例えば、書込みモード ファイルをオープンし、属性を読取り専用に変更しても、ファイルを一度クローズし、再オープンするまで書込み可能です。	
参照	chmod	

fclose	8200,8400,8700
目的	ファイルをクローズします。
書式	int fclose (int fd);
引数	<div>int fd</div> <div>対象ファイルのファイルハンドル。</div>
コーディング例	<pre>int fd; fd = fopen ("A:¥¥file1", "r+");    /* オープン: リード &amp; ライトモード */ /* processing */ if ( fclose (fd) != NULL )     printf ("file close error);</pre>
戻り値	正常終了すると 0 を返します。エラーが発生した場合は-1 を返し、システムグローバル変数 ferrno に発生したエラー状態がセットされます。
備考	引数で指定されたファイルをクローズします。ファイルが書込みモード でオープンされている場合は、クローズする前に、書込みバッファがフラッシュされます。
参照	fflush, fopen

fclosedir		8200,8400,8700
目的	ディレクトリをクローズします。	
書式	int fclose (int dir_handle);	
引数	int dir_handle	対象ディレクトリのハンドル。
コーディング例	<pre>int dir_handle; dir_handle = opendir ("A:¥¥SubDir"); if ( fclose (dir_handle) != NULL )     printf ("Fail close a directory");</pre>	
戻り値	正常終了すると 0 を返します。エラーが発生した場合は-1 を返し、システムグローバル変数 errno に発生したエラー状態がセットされます。	
参照	fopendir	

fcopy		8200,8400,8700
目的	ファイルをコピーします。	
書式	int fcopy(const char *srcfile, const char *dstfile);	
引数	const char *srcfile	コピー元ファイル名が格納された変数へのポインタ。
	const char *dstfile	コピー先ファイル名が格納された変数へのポインタ。
コーディング例	fcopy("a:\\SrcFile.txt","a:\\DstFile.txt");	
戻り値	正常終了すると 0 を返します。エラーが発生した場合は-1 を返し、システムグローバル変数 errno に発生したエラー状態がセットされます。	
備考	ファイルをコピーします。コピー先ファイルがすでに存在する場合、この処理は失敗します。ファイル名はフルパスの 8.3 フォーマットで指定します。	

feof		8200,8400,8700
目的	EOF(ファイルの終わり)フラグをチェックします。	
書式	int fclose (int fd);	
引数	int fd	対象ファイルのファイルハンドル。
コーディング例	<pre>int fd; int c; fd = fopen ("A:¥¥file1", "r+");    /* オープン: リード &amp; ライトモード */ while ( !feof (fd) ) {     c = fgetc (fd); }</pre>	
戻り値	EOF に到達した場合は、0 以外を返します。EOF に到達していない場合は、0 を返します。	
参照	clearerr	



fflush		8200,8400,8700
目的	ファイルに関連付けられた出力バッファをフラッシュします。出力バッファに残っているデータをファイルに書込みます。	
書式	int fflush (int fd);	
引数	int fd	対象ファイルのファイルハンドル。
コーディング例	<pre>int fd; if (fflush (fd)) {     /* ファイル フラッシュエラー */ }</pre>	
戻り値	正常終了すると 0 を返します。エラーが発生した場合は-1 を返し、システムグローバル変数 ferrno に発生したエラー状態がセットされます。	
参照	fclose	

fformat		8200,8400,8700
目的	SD カード をフォーマット(初期化)します。	
書式	int fformat (void);	
コーディング例	<pre>if (fformat() != NULL)     printf("Format failed");</pre>	
戻り値	正常終了すると 0 を返します。エラーが発生した場合は 0 以外を返し、システムグローバル変数 ferrno に発生したエラー状態がセットされます。	
備考	SD カード をデータ容量に対応した FAT フォーマットで初期化します。データ容量が 2GB 以下だと FAT16 で、それ以上の容量だと FAT32 で初期化します。	
参照	fopendir, freaddir	

fgetc		8200,8400,8700
目的	ファイルからデータを 1 バイト読み出します。	
書式	int fgetc (int fd);	
引数	int fd	対象ファイルのファイルハンドル。
コーディング例	<pre>int fd; char buffer[81]; int i, c; if ((fd = fopen ("A:¥¥file1", "r")) == NULL) {     printf ("fopen failed.¥n");     exit (0); } c = fgetc (fd); for (i=0; (i &lt; 80) &amp;&amp; (feof (fd) == 0) &amp;&amp; (c != '¥n'); i++) {     buffer [i] = c;     c = fgetc (fd); } buffer [i] = '¥0'; printf ("First line of A:file1 : %s¥n", buffer);</pre>	
戻り値	正常終了すると、読み出したキャラクタを int 型で返します。エラーが発生した場合は 1 を返します。ferror、feof をコールして、エラーか、EOF かをチェックすることが可能です。	
備考	引数で指定されたファイルからデータを 1 バイト読み出します。データの読み出しは、現在のファイルポインタ位置から行われ、読み出し後は、ファイルポインタを 1 バイト分進めます。	
参考	fgets, fputc, fputs	

fgetinfo		8200,8400,8700
目的	ファイルまたはディレクトリの情報を取得します。	
書式	int fgetinfo(const char* file_name, FILEINFO *fileinfo);	
引数	const char *srcfile	
	情報を取得するファイル名またはディレクトリ名が格納された変数へのポインタ。 ファイル名はファイルの 8.3 フォーマットで指定します。	
	FILEINFO *fileinfo	8200lib.h、8400lib.h、8700lib.h で定義されている FILEINFO 構造体の変数へのポインタ。
コーディング例	<pre>FILEINFO fileinfo; if ((fileinfo("A:\\userfile.txt", &amp;fileinfo)) == 0) {     printf("file size:%ld", fileinfo.fsize); }</pre>	
戻り値	正常終了すると 0 を返します。エラーが発生した場合は-1 を返し、システムグローバル変数 errno に発生したエラー状態がセットされます。	
参照	fopendir, freaddir	

fgetpos		8200,8400,8700
目的	現在のファイルポインタ位置を取得します。	
書式	int fgetpos (int fd, unsigned long *position);	
引数	int fd	
	対象ファイルのファイルハンドル。	
	unsigned long *position	ファイルポインタ位置を格納する変数へのポインタ。
コーディング例	<pre>int fd; int c; unsigned long position; if ((fd = fopen ("a:\\SubDir\\UserFile", "r")) == NULL) {     printf ("fopen failed. \n");     exit (0); } c = fgetc (fd); if (fgetpos (fd, &amp;position) != 0)     printf ("fgetpos failed");</pre>	
戻り値	正常終了すると 0 を返します。エラーが発生した場合は-1 を返し、システムグローバル変数 errno に発生したエラー状態がセットされます。	
備考	引数で指定されたファイルの現在のファイルポインタ位置を取得し、指定の変数に格納します。	
参照	fsetpos	

fgets	8200,8400,8700
目的	ファイルからデータを 1 行読み出します。データの読み出しは、現在のファイルポインタ位置から行われ、改行文字(¥n)を検出する、又は指定された最大バイト数までです。
書式	char *fgets (char *string, int max_char, int fd);
引数	char *string
	読込んだデータを格納する変数へのポインタ。
	unsigned long *position
	読み込みを行う最大バイト数。
	int fd
	対象ファイルのファイルハンドル。
コーディング例	<pre> int fd; char string[81]; if ((fd = fopen ("a:\\SubDir\\UserFile", "r")) == NULL) {     printf ("fopen failed. ¥n");     exit (0); } while (fgets (string, 80, fd) != NULL)     printf ("%s¥n", string); </pre>
戻り値	正常終了すると、読み出した文字列 string を返します。エラーが発生した場合は 0 を返します。ferror、feof をコールして、エラーか、EOF かをチェックすることが可能です。
備考	引数で指定されたファイルからデータを 1 行読み出します。データの読み出しは、現在のファイルポインタ位置から行われ、改行文字(¥n)を検出する又は指定された最大バイト数までを 1 行とし、ヌルを付加して、指定の変数に格納します。データ読み出し後、ファイルポインタも適切に更新されます。
参考	fgetc, fputc, fputs

fopen		8200,8400,8700
目的	ファイルをオープンします。	
書式	int fopen (const char *filename, const char *mode);	
引数	const char *filename	
	オープンするファイル名が格納された変数へのポインタ。ファイル名はフルパスの 8.3 フォーマットで指定します。	
	const char *mode	
	アクセスモード。	
	"r"	テキストファイルを読み取りモードでオープン。
	"w"	テキストファイルを書込みモードで作成(ファイルがあった場合、元のファイルのデータはすべて消去)。
	"a"	テキストファイルを最終位置からの追加書き込みモードでオープン。
	"rb"	バイナリファイルを読み取りモードでオープン。
	"wb"	バイナリファイルを書込みモードで作成(ファイルがあった場合、元のファイルのデータはすべて消去)。
	"ab"	バイナリファイルを最終位置からの追加書き込みモードでオープン。
	"r+"	テキストファイルを更新モードでオープン。
	"w+"	テキストファイルを更新モードで作成(ファイルがあった場合、元のファイルのデータはすべて消去)。
	"a+"	テキストファイルを最終位置からの追加更新モードでオープン。
	"r+b"	バイナリファイルを更新モードでオープン。
	"w+b"	バイナリファイルを更新モードで作成(ファイルがあった場合、元のファイルのデータはすべて消去)。
	"a+b"	バイナリファイルを最終位置からの追加更新モードでオープン。
コーディング例	<pre>int fd; if ((fd = fopen ("a:\\SubDir\\UserFile", "r")) == NULL) {     printf ("fopen failed. \n");     exit (0); }</pre>	
戻り値	正常終了すると、ファイルハンドルを返します。エラーが発生した場合は 0 を返し、システムグローバル変数 <code>errno</code> に発生したエラー状態がセットされます。	
備考	<p>引数 <code>filename</code> で指定されたファイルを開きます。<code>mode</code> 文字列は、ファイルアクセスのタイプを指定します。ファイルオープン成功した場合、ファイルのファイルハンドルが戻ります。</p> <p>➤ 同時に 32 ファイルまで開くことができますが、ファイルハンドルが枯渇しないためにも使用しないファイルは閉じることをお勧めします。</p>	
参考	fclose	

fopendir		8200,8400,8700
目的	ディレクトリをオープンします。	
書式	int fopendir (const char *dirname);	
引数	const char *dirname	
	オープンするディレクトリ名が格納された変数へのポインタ。	
コーディング例	<pre>if (fopendir ("a:\\SubDir", "r") == 0)     printf ("fail to open a directory. ");</pre>	
戻り値	正常終了すると 0 を返します。エラーが発生した場合は 0 以外を返し、システムグローバル変数 <code>errno</code> に発生したエラー状態がセットされます。	
備考	<p>引数 <code>dirname</code> で指定されたディレクトリを開きます。ディレクトリ名はフルパスで指定します。</p> <p>➤ 同時に 3 ディレクトリまで開くことができますが、ディレクトリハンドルが枯渇しないためにも使用しないディレクトリは閉じることをお勧めします。</p>	
参考	fclosedir, fformat, freaddir	

fputc		8200,8400,8700
目的	ファイルヘデータを1バイト書込みます。	
書式	int fputc (int c, int fd);	
引数	int c	
	ファイルに書込むデータ。	
	int fd	
	対象ファイルのファイルハンドル。	
コーディング例	<pre>int fd; char buffer[81] = "Testing the function fputc"; int i; if ((fd = fopen ("a:\\SubDir\\UserFile", "w")) == NULL) {     printf ("fopen failed.\n");     exit (0); } for (i=0; (i &lt; 80) &amp;&amp; buffer[i] &amp;&amp; (fputc (buffer[i], fd) != EOF); i++);</pre>	
戻り値	正常終了すると、書込んだデータ(キャラクタ)を返します。エラーが発生した場合は-1 を返します。ferror をコールして、エラー状態をチェックしてください。	
備考	引数で指定されたファイルヘデータを1バイト書込みます。データの書込みは、現在のファイルポインタ位置から行われ、書込み後は、ファイルポインタを1バイト分進めます。	
参照	fgetc, fgets, fputs	

fputs		8200,8400,8700
目的	ファイルに文字列データを書込みます。	
書式	int fputs (const char *string, int fd);	
引数	const char *string	
	ファイルに書込むデータが格納された変数へのポインタ。	
	int fd	
	対象ファイルのファイルハンドル。	
コーディング例	<pre>int fd; char string[81] = "Testing the function fputs"; if ((fd = fopen ("a:\\SubDir\\UserFile", "w")) == NULL) {     printf ("fopen failed.\n");     exit (0); } fputs (string, fd);</pre>	
戻り値	正常終了すると、書込んだバイト数を返します。エラーが発生した場合は-1 を返します。ferror をコールして、エラー状態をチェックしてください。	
備考	引数で指定されたファイルへ文字列データを書込みます。データの書込みは、現在のファイルポインタ位置から行われ、データ書込み後、ファイルポインタも適切に更新されます。	
参照	fgetc, fgets, fputc	

fread		8200,8400,8700
目的	ファイルから指定サイズのデータアイテムを指定数分読み出します。	
書式	int fread (void *buffer, int size, int count, int fd);	
引数	void *string	
	読込んだデータを格納する変数へのポインタ。	
	int size	
	各データアイテムのサイズ。	
	int count	
	読み込みを行う最大データアイテム数。	
	int fd	
	対象ファイルのファイルハンドル。	
コーディング例	<pre>int fd; char buffer[81]; int count; if ((fd = fopen ("a:\\SubDir\\UserFile", "r")) == NULL) {     printf ("fopen failed.\n");     exit (0); } count = fread (buffer, 1, 80, fp); printf ("Read these %d characters:\n %s\n", count, buffer);</pre>	
戻り値	正常終了すると、読込んだデータアイテム数を返します。但し、戻り値が引数 count と等しくない場合は、エラー又は EOF が発生した可能性があります。ferror、feof をコールして、エラーか、EOF かをチェックすることが可能です。	
備考	引数で指定されたファイルから指定サイズのデータアイテムを指定数分読み込みます。データの読み込みは、現在のファイルポインタ位置から行われ、データ読み込み後、ファイルポインタも適切に更新されます。	
参照	fwrite	

freaddir		8200,8400,8700
目的	ディレクトリをオープンします。	
書式	int freaddir (int dir_handle, FILEINFO *fileinfo);	
引数	int dir_handle	
	対象ディレクトリのハンドル。	
	FILEINFO *fileinfo	8200lib.h、8400lib.h、8700lib.h で定義されている FILEINFO 構造体の変数へのポインタ。
コーディング例	<pre>FILEINFO fileinfo; int dir_handle if ((dir_handle = fopendir ("a:\\SubDir", "r")) == 0)     printf ("fail to open a directory. "); if ((freaddir (dir_handle, &amp;fileinfo) == NULL) &amp;&amp; fileinfo.fname[0])     printf ("file name is %s", fileinfo.fname);</pre>	
戻り値	正常終了すると、ディレクトリハンドルを返します。エラーが発生した場合は 0 を返し、システムグローバル変数 ferrno に発生したエラー状態がセットされます。	
備考	ディレクトリハンドルで指定されたディレクトリのエントリを取得します。繰り返し freaddir をコールすることによってそのディレクトリ内のすべての項目を取得することができます。すべての項目を取得し終わり、項目が残ってない場合、エラーは発生せず、fileinfo.fname に null 文字がセットされます。	
参照	fformat, fopendir	

fremove		8200,8400,8700
目的	ファイルを削除します。	
書式	int fremove (const char *filename);	
引数	const char *filename 削除するファイル名を格納した変数へのポインタ。 ファイル名はファイル名の 8.3 フォーマットで指定します。	
コーディング例	if ( fremove ("a:\\subdir\\thisfile.txt") ) printf ("ferrno = %d\n", ferrno);	
戻り値	正常終了すると 0 を返します。エラーが発生した場合は 0 以外を返し、システムグローバル変数 ferrno に発生したエラー状態がセットされます。	
備考	引数 filename で指定されたファイルを削除します。ファイル名は、ドライブ情報も含めて指定します。	
参照	frename, rmdir	

frename		8200,8400,8700
目的	ファイル名を変更します。	
書式	int rename (const char *oldname, const char *newname);	
引数	const char *oldname 変更前ファイル名を格納した変数へのポインタ。 const char *newname 変更後ファイル名を格納した変数へのポインタ。	
コーディング例	if ( rename("a:\\file1.txt", "a:\\file2.txt") ) printf ("ferrno = %d\n", ferrno);	
戻り値	正常終了すると 0 を返します。エラーが発生した場合は 0 以外を返し、システムグローバル変数 ferrno に発生したエラー状態がセットされます。	
備考	ファイル名を変更します。変更前と変更後のサブディレクトリ名に異なる名前を指定すれば、新しいサブディレクトリへファイルを移動したように処理することも可能です。ファイル名はファイル名の 8.3 フォーマットで指定します。	
参照	fremove, mkdir, rmdir	

fscan		8200,8400,8700
目的	SD カードの空き容量情報を更新します。	
書式	int fscan(void);	
コーディング例	if ( fscan() != 0 ) printf ("ferrno = %d\n", ferrno);	
戻り値	正常終了すると 0 を返します。エラーが発生した場合は 0 以外を返し、システムグローバル変数 ferrno に発生したエラー状態がセットされます。	
備考	SD カードによっては ffreebyte() で現在の値を取得に失敗し、空き容量の情報が不正確な場合があります。この処理は、空き容量情報を更新するために、SD カードをスキャンします。処理完了までに時間がかかる場合があります。	

fseek		8200,8400,8700
目的	ファイルポインタ位置を移動します。	
書式	int fseek (int fd, long offset, int origin);	
引数	int fd	
	対象ファイルのファイルハンドル。	
	long offset	
	移動させたい原点からのオフセット値(バイト)。	
	int origin	
	移動の基準となる原点位置	
	1	SEEK_SET      ファイルの先頭位置。
	0	SEEK_CUR      現ファイルポインタ位置
	-1	SEEK_END      ファイルの終端位置。
コーディング例	<pre>int fd; if (fseek(fd, 30L, SEEK_SET) != 0)     printf ("fseek failed!");</pre>	
戻り値	正常終了すると 0 を返します。エラーが発生した場合は 0 以外を返し、システムグローバル変数 errno に発生したエラー状態がセットされます。	
備考	引数で指定されたファイルのファイルポインタ位置を指定のオフセット位置へ移動します。ファイルがテキストモードでオープンされている場合は、基準位置 origin を SEEK_CUR (0) 又は ftell 関数で取得した現在のファイルポインタ位置に指定してください。	
参照	ftell	

fsetpos		8200,8400,8700
目的	ファイルポインタ位置をセットします。	
書式	int fsetpos (int fd, const unsigned long *newposition);	
引数	int fd	
	対象ファイルのファイルハンドル。	
	const unsigned long *newposition	
	新しいファイルポインタ位置。	
コーディング例	<pre>int fd; unsigned long curpos; char buffer [80]; if (fgetpos (fd, &amp;curpos) != 0)      /* 現在のファイルポインタ位置を取得 */     printf ("fgetpos failed!"); if (fgets(buffer, 20, fd) == NULL) /* 20 バイトを読み出し */     printf ("fgets failed!"); if (fsetpos (fd, &amp;curpos) != 0)      /* ファイルポインタを読み出し前の位置にセット */     printf ("fsetpos failed!");</pre>	
戻り値	正常終了すると 0 を返します。エラーが発生した場合は 0 以外を返し、システムグローバル変数 errno に発生したエラー状態がセットされます。	
備考	引数で指定されたファイルのファイルポインタ位置をセットします。	
参照	fgetpos	



ftell		8200,8400,8700	
目的	現在のファイルポインタ位置を取得します。		
書式	long ftell (int fd);		
引数	int fd	対象ファイルのファイルハンドル。	
コーディング例	<pre>int fd; long curpos; if ((curpos = ftell (fd)) == -1L)     printf ("ftell failed!");</pre>		
戻り値	正常終了すると、現在のファイルポインタ位置(ファイルの先頭からのオフセット値)を long 型の値(バイト)で返します。エラーが発生した場合は-1L を返し、システムグローバル変数 <code>errno</code> に発生したエラー状態がセットされます。		
備考	引数で指定されたファイルの現在のファイルポインタ位置を取得します。		
参照	fseek		

ftruncate		8200,8400,8700	
目的	ファイルポインタ位置以降のファイルの内容を削除します。		
書式	int ftruncate (int fd);		
引数	int fd	対象ファイルのファイルポインタ。	
コーディング例	int fd; fd = fopen("A:\\UserFile.txt", "wb"); fseek(fd, 10, SEEK_SET); ftruncate (fd); fclose(fd);		
戻り値	正常終了すると 0 を返します。エラーが発生した場合は-1 を返し、システムグローバル変数 errno に発生したエラー状態がセットされます。		
備考	fseek() で以降を削除したい位置にファイルポインタを移動します。		
参照	fseek		

fwrite		8200,8400,8700
目的	ファイルに指定のデータを指定サイズで指定数分書込みます。	
書式	int fwrite (const void *buffer, int size, int count, int fd);	
引数	const void *buffer	書込むデータが格納された変数へのポインタ。
	int size	各データアイテムのサイズ。
	int count	書込みを行う最大データアイテム数。
	int fd	対象ファイルのファイルハンドル。
コーディング例	<pre>int fd; char buffer[81] = "Testing the fwrite function"; int count; if ((fd = fopen ("A:\\UserFile", "w")) == NULL) {     printf ("fopen failed. \n");     exit (0); } count = fwrite (buffer, 1, 20, fd); printf ("%d characters written to a file", count);</pre>	
戻り値	正常終了すると、書込んだデータアイテム数を返します。但し、戻り値が引数 count と等しくない場合は、エラー又は EOF が発生した可能性があります。ferror、feof をコールして、エラーか、EOF かをチェックすることが可能です。	
備考	引数で指定されたファイルに指定サイズのデータアイテムを指定数分書込みます。データの書込みは、現在のファイルポインタ位置から行われ、データ書込み後、ファイルポインタも適切に更新されます。	
参照	fread	

mkdir		8200,8400,8700
目的	新しいディレクトリを作成します。	
書式	int mkdir (const char *newdir);	
引数	const char *newdir	作成するディレクトリ名が格納された変数へのポインタ。
コーディング例	<pre>if (mkdir ("A:\\SubDir\\SubDir2\\new_dir") != 0)     printf ("Fail to create a directory");</pre>	
戻り値	正常終了すると 0 を返します。エラーが発生した場合は 0 以外を返し、システムグローバル変数 errno に発生したエラー状態がセットされます。	
備考	引数で指定された新しいディレクトリを作成します。ディレクトリ名はフルパスの 8.3 フォーマットで指定します。	
参照	rmdir	

rmkdir	8200,8400,8700
目的	ディレクトリを削除します。
書式	int rmkdir (const char *newdir);
引数	const char *newdir 作成するディレクトリ名が格納された変数へのポインタ。
コーディング例	if (rmkdir ("a:\\SubDir1\\SubDir2") != 0) printf ("Fail to delete the directory");
戻り値	正常終了すると 0 を返します。エラーが発生した場合は 0 以外を返し、システムグローバル変数 errno に発生したエラー状態がセットされます。
備考	引数で指定されたディレクトリを削除します。但し、削除しようとするディレクトリは空でなければなりません。ディレクトリ下にファイルが存在する場合やルートディレクトリを指定した場合は、エラーが発生します。
参照	fremove, mkdir

2.16.6 マスストレージ デバイス

マスストレージ 使用中は

(1) 開いているすべてのファイルは自動的に閉じられます。

(2) 『2.16.5 SD カード 操作』にあるどの処理も close\_com(5)の前にコールされた場合、SD カード はマスストレージ にデバイスとして使用中であることを意味する"E\_SD\_OCCUPIED"のエラーコード が戻ります。

GetMassStorageStatus		8200,8400,8700
目的	マスストレージ 使用中のステータスを取得します。	
書式	int GetMassStorageStatus();	
コーディング 例	<pre>int status; atatus = GetMassStorageStatus(); if (status &amp; 0x1) {     printf("USB is connected"); } else {     printf("USB is disconnected"); } }</pre>	
戻り値	各項目の現在のステータスの値を OR した整数値が戻ります。	
備考	各ビットが示すステータスは次の通りです。	
	ビット	
	0	0 : USB 未接続 1 : USB 接続済
	1	0 : デバイス未使用 1 : デバイス使用中
参照	SetCommType	

2.16.7 エラーコード

ほとんどの SD 関連の関数は、グローバル変数 `errno` に発生したエラー状態をセットしています。例えば、

```
fd = fopen("a:\\file1", "rb");
if (!fd) {
    printf("%d", errno);
}
```

発生したエラー状態の詳細は `error()`にあるエラーコード リストを参照してください。また、ファイルへの読み取り/書き込み操作実行時のエラーコード にアクセスするには、`error()`をコールします。

ferrno 使用例

```
fwrite(x, x, x, fd1);
error1 = ferrno;
fwrite(x, x, x, fd2);
error2 = ferrno;
```

SD 関連の機能を実行した後、グローバル変数 `errno` はそれに応じて更新されます。したがって、上記例では `error1` と `error2` は値が異なる場合があります。

Error()使用例

```
fwrite(x, x, x, fd1);
error1 = error(fd1);
fwrite(x, x, x, fd2);
error2 = error(fd2);
error1 = error(fd1);
```

ファイルへの読み込み/書き込み動作に関連する機能を実行した後に、`error()`をコールして取得した値は、`errno` が保持している値と同じです。唯一の違いは、`error()`によって返される値は、同じファイルが読み込み/書き込み動作に関連する関数を実行するまで更新されないことです。したがって、上記例では最初の `error1` の処理と 2 回目の `error1` の処理では同じ値が戻ります。

clearerr		8200,8400,8700
目的	ファイルのエラーコード をリセットします。	
書式	void clearerr (int fd);	
引数	int fd 対象ファイルのファイルハンドル。	
コーディング 例	<pre>int fd; char string [81]; if ((fp = fopen ("A:\\UserFile", "r")) == NULL)  {     printf ("fopen failed. \n");     exit (0); } fgets (string, 80, fd); if (ferror (fd) != 0)  {     printf ("Error detected. \n");     clearerr (fd);     printf ("Error cleared. \n"); }</pre>	
戻り値	無し	
備考	引数で指定されたファイルのエラーコード を 0 にクリアします。	

<b>error</b>	<b>8200,8400,8700</b>
--------------	-----------------------

**目的** ファイルの読み取り/書き込み操作時にエラーが発生したかどうかをチェックします。

**書式** int ferror (FILE \*file\_pointer);

**引数** int fd

対象ファイルのファイルハンドル。

**コーディング例**

```
int fd;
int c;
fp = fopen ("A:\\UserFile, "r+");    /* オープン: リード & ライトモード */
while ( !feof (fd) ) {
    c = fgetc (fd);
    if (ferror (fd)) {
        printf ("Error detected. \n");
        clearerr (fd);
        printf ("Error cleared. \n");
    }
}
```

**戻り値** エラーフラグを検出した場合は、0 以外を返します。

1	E_SD_NOT_READY	SD カードが準備されていません。
2	E_NO_FILESYSTEM	マウントされていないファイルシステムです。
3	E_NO_OBJECT	オブジェクトが見つかりません。
4	E_NO_PATH	パスが見つかりません。
5	E_NOT_DIR	ディレクトリではありません。
6	E_NOT_FILE	ファイルではありません。
7	E_DIR_NOT_EMPTY	ディレクトリは空ではありません。
8	E_INVALID_NAME	無効な名前です。
9	E_INVALID_OBJECT	オブジェクトは正しく開かれていません。
10	E_READ_ONLY	オブジェクトは読み取り専用です。。
11	E_ACCESS_DENIED	アクセスがオープンモードと一致しません。
12	E_OBJECT_EXIST	オブジェクトは既に存在します。
13	E_DISK_FULL	ディスクがいっぱいです。
14	E_RW_ERROR	読み込み/書き込みエラー不良。
15	E_INVALID_HANDLE	無効なハンドルです。
16	E_NO_AVAILABLE_HANDLE	利用できないハンドルです。
17	E_INVALID_MODE	無効なモードです。
18	E_SD_OCCUPIED	SD はマストレーズで使用中です。

**説明** この関数をコールすることで fgetc()、fgets()、fputc()、fputs()、fread()、fwrite()のエラー状態を取得できます。

### 3 標準ライブラリ関数

#### 標準入出力関数 : <stdio.h>

- ファイル操作 対応していません。CipherLab 専用ライブラリ関数を使用してください。
- 書式付出力関数 sprintf 関数のみ対応しています。
- 書式付入力関数 sscanf 関数のみ対応しています。
- 文字入出力関数 対応していません。CipherLab 専用ライブラリ関数を使用してください。
- 直接入出力関数 対応していません。

#### 文字クラス関数 : <ctype.h>

各関数の引数は、char 型で、EOF 又は unsigned char 型で表現できる値でなければならず、戻り値は、int 型になります。引数 c が各条件を満たす場合は、0 以外の値を返し、そうでない場合は、0 を返します。

- isalnum(c) isalpha(c) 又は isdigit(c) が真である
- isalpha(c) isupper(c) 又は islower(c) が真である
- iscntrl(c) 制御文字
- isdigit(c) 10 進数字
- isgraph(c) 空白以外の印字可能文字
- islower(c) 小文字
- isprint(c) 空白を含む印字可能文字
- ispunct(c) 空白、英字、数字以外の印字可能文字
- isspace(c) 空白、改行、改行、復帰、タブ、垂直タブ
- isupper(c) 大文字
- isxdigit(c) 16 進数字

また、英字の大小文字を変換するための下記の 2 種類の関数が用意されています。

- int tolower(c) c を小文字に変換
- int toupper(c) c を大文字に変換

#### 文字列関数 : <string.h> str で始まる関数

引数として、下記の変数タイプが使用されています。

- ```
char *s;  
const char *cs, *ct;  
size_t n;  
int c;
```
- char \*strcpy(s, ct) '¥0'を含めて文字列 ct を s にコピーし、s を返す
  - char \*strncpy(s, ct, n) 文字列 ct のうち最大 n 文字を s にコピーし、s を返す  
ct が、n 文字より少ないときは、'¥0' を詰める
  - char \*strcat(s, ct) 文字列 ct を文字列 s の終わりに連結し、s を返す
  - char \*strncat(s, ct, n) 文字列 ct の最大 n 文字を文字列 s に連結し、終わりに '¥0' を付け、s を返す
  - int strcmp(cs, ct) 文字列 cs と文字列 ct を比較し、cs<ct なら <0、cs==ct なら 0、cs>ct なら >0 を返す
  - int strncmp(cs, ct, n) 文字列 cs と文字列 ct の最大 n 文字を比較し、cs<ct なら <0、cs==ct なら 0、cs>ct なら >0 を返す
  - char \*strchr(cs, c) cs 中にある最初の c へのポインタ又はそれが無い場合は、NULL を返す
  - char \*strrchr(cs, c) cs 中にある最後の c へのポインタ又はそれが無い場合は、NULL を返す
  - size\_t strspn(cs, ct) ct に入っている文字よりなる cs の接頭子(prefix)の長さを返す
  - size\_t strcspn(cs, ct) ct に無い文字よりなる cs の接頭子(prefix)の長さを返す
  - char \*strpbrk(cs, ct) 文字列 ct の任意の文字が文字列 cs の中で最初に出てくる位置へのポインタ又はそれが無い場合は、NULL を返す
  - char \*strstr(cs, ct) cs の中で文字列 ct が最初に現れる位置へのポインタ又はそれが無い場合は、NULL を返す
  - size\_t strlen(cs) cs の長さを返す
  - char \*strtok(s, ct) ct 中の文字によって区切られる文字列(トークン)が s の中から探され、文字列(トークン)へのポインタを返す
  - strcoll 対応していません
  - strerror 対応していません

## 文字列関数 : <string.h> mem で始まる関数

引数として、下記の変数タイプが使用されています。

- ```
void *s;
const void *cs, *ct;
size_t n;
int c;
```
- void \*memcpy(s, ct, n) ct の n 文字を s にコピーし、s を返す
  - void \*memmove(s, ct, n) コピー元とコピー先のメモリが重複しても動く点を除けば、memcpy 関数と同じ
  - int memcmp(cs, ct, n) cs の最初の n 文字を ct と比較  
戻り値は、strcmp 関数と同じ
  - void \*memchr(cs, c, n) cs の中で c が最初に出てくる位置へのポインタ、又はそれが最初の n 文字内に無いときは、NULL を返す
  - void \*memset(s, c, n) s の最初の n 文字の中に文字 c を入れ、s を返す

## 数学関数 : <math.h>

引数として、下記の変数タイプが使用されており、全ての関数は double 型の値を返します。

- ```
double x, y;
int n;
```
- sin(x) x の正弦(sin)
  - cos(x) x の余弦(cos)
  - tan(x) x の正接(tan)
  - asin(x) 範囲  $[-\pi/2, \pi/2]$ ,  $x \in [-1, 1]$  の  $\sin^{-1}(x)$
  - acos(x) 範囲  $[0, \pi]$ ,  $x \in [-1, 1]$  の  $\cos^{-1}(x)$
  - atan(x) 範囲  $[-\pi/2, \pi/2]$  の  $\tan^{-1}(x)$
  - atan2(y, x) 範囲  $[-\pi, \pi]$  の  $\tan^{-1}(y/x)$
  - sinh(x) x の双曲線正弦(sin)
  - cosh(x) x の双曲線余弦(cos)
  - tanh(x) x の双曲線正接(tan)
  - exp(x) 指数関数  $e^x$
  - log(x) 自然対数  $\ln(x)$ ,  $x > 0$
  - log10(x) 基底 10 対数  $\log_{10}(x)$ ,  $x > 0$
  - pow(x, y)  $x^y$  ( $x=0$  で  $y \leq 0$ 、又は  $x < 0$  で  $y$  が整数で無いときには領域エラーが起きる)
  - sqrt(x)  $\sqrt{x}$ ,  $x \geq 0$
  - ceil(x) 切上げ : x より小さくない最小の整数(double 型として)
  - floor(x) 切捨て : x より大きくない最大の整数(double 型として)
  - fabs(x) 絶対値  $|x|$
  - ldexp(x, n)  $x * 2^n$
  - frexp(x, int \*exp) x を仮整数部(0.5~1)を返し、指数部を exp に格納する  
(x が 0 なら、結果は両部分とも 0 である)
  - modf(x, double \*ip) x をそれぞれ x と同じ符号を持つ整数部と小数部に分割する  
整数部は、\*ip に格納され、小数部が返される
  - fmod(x, y) x/y の余り  
符号は x と同じで、y が 0 のとき、結果は処理系依存となる



## ユーティリティ関数 : <stdlib.h> 数値変換

- `double atof( const char *s)`  
s を double 型に変換します。これは、`strtod(s, (char **)NULL)` と同じです。
- `int atoi(const char *s)`  
s を int 型に変換します。これは、`strtol(s, (char **)NULL, 10)` 関数と同じです。
- `int atol(const char *s)`  
s を long 型に変換します。これは、`strtol(s, (char **)NULL, 10)` 関数と同じです。
- `double strtod(const char *s, char **endp)`  
s の文字列を double 型に変換します。s の先頭には、空白が 1 文字必要です。変換できない文字があれば、変換を中止し、その文字のポインタを endp に格納します。
- `long strtol(const char *s, char **endp, int base)`  
s の文字列を base で指定した数を基数として、long 型に変換します。s の先頭には、空白が 1 文字必要です。変換できない文字があれば、変換を中止し、その文字のポインタを endp に格納します。
- `unsigned long strtoul(const char *s, char **endp, int base)`  
unsigned long 型である以外は、strtol 関数と同じです。
- `int rand(void)`  
0 ~ 32767 の範囲で擬似乱数整数を返します。
- `void srand(unsigned int seed)`  
擬似乱数の新しい系列の種として seed が使われます。最初の seed は 1 です。
- `void *bsearch()`  
バイナリ検索。
- `void qsort()`  
昇順ソート。
- `int abs(int n)`  
int 型 n の絶対値を返します。
- `long labs(long n)`  
long 型 n の絶対値を返します。
- `div_t div(int num, int denom)`  
num/denom の商と余りを求めるためのものです。結果は、div\_t 構造体の int 型メンバ `quot` 及び `rem` に格納されます。
- `ldiv_t ldiv(long num, long denom)`  
num/denom の商と余りを求めるためのものです。結果は、ldiv\_t 構造体の long 型メンバ `quot` 及び `rem` に格納されます。

## ユーティリティ関数 : <stdlib.h> 領域割当

サポートしていません。CipherLab 専用ライブラリ関数を使用してください。

## 診断 : <assert.h>

サポートしていません。

## 可変引数リスト : <stdarg.h>

個数及び型の分からない関数引数のリストを次々に使ってゆく機能を与えるものです。

```
va_start(va_list ap, lastarg)
type va_arg(va_list ap, type)
void va_end(va_list ap)
```

## 非局所的ジャンプ : <setjmp.h>

サポートしていません。

## シグナル : <signal.h>

サポートしていません。

## 日付 & 時刻関数 : <time.h>

サポートしていません。

## 処理系で定義される制限 : <limits.h> and <float.h>

ヘッダファイル `limit.h` 及び `float.h` を参照下さい。

## 4 リアルタイムカーネル

CipherLab リソース ハンデ ィターミナルは、プ リエティブ マルチタスクをサポ ートしたリアルタイムカーネル (μC/OS) を搭載しています。これにより、ユーザ ーは、プ ライオリティバ ルをベ ースに各ア プ リケーションタスクでシステムリソースを共有することが可能になります。

μC/OS では、システムリソースへのアクセスを制御するために、タスクの効 率を向上させるために、通常、タスクは、CREATE, PEND, POST の 3 つの 状態のみのみを実行します。タスクは、タスクが実行を継続するために必要となるキ ーとなります。タスクが既に使用中の場合、現在使用中のタスクによってタスクが リリースされるまで、タスクをリクエストしたタスクはサス ペンド されます。

ここでいうタスクとは、無限ル ープ 型の関数又は実行完了後に自身を削除する関数で、各タスクには、適切なプ ライオリティバ ル(重要なタスクには、より高いプ ライオリティバ ルを割り当てる)を割り当てます。μC/OS では、32 個までのタスクを管理することができ、それぞれにプ ライオリティバ ルを 0~31(低い数字ほどプ ライオリティバ ルが高い)の範囲で割り当てます。メインタスクである main 関数は、プ ライオリティバ ル 16 となります。

タスクをリクエストするタスクは、PEND 状態のみのみを実行し、POST 状態のみのみにより、タスクをリリースします。タスクがリリースされたときに、複数のタスクがペ ンディング リストに存在する場合は、プ ライオリティバ ルが一番高いタスクが次にタスクを受け取ります。ペ ンディング リストは、初期状態では常に空となります。

タスクは、しばしば過剰に使用されます。割り込みの無効/有効でより効率的に運用をすることができます。すべてのリアルタイムカーネルは、コード のクリティカルセクション中の割り込みを無効にします。カーネルが割り込み待ち時間に影響を与えないように、基本的には多くの割り込みを無効にすることができます。

下記に、μC/OS に関連する関数について説明します。

### OS\_ENTER\_CRITICAL

|          |                                                                                                                                                                                                                                         |
|----------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 目的       | プ ロセッサの割り込みを無効にします。                                                                                                                                                                                                                     |
| 書式       | void OS_ENTER_CRITICAL(void);                                                                                                                                                                                                           |
| コーディング 例 | OS_ENTER_CRITICAL();<br>/* ユーザ ーコード */<br>OS_ENTER_CRITICAL();                                                                                                                                                                          |
| 戻り値      | 無し                                                                                                                                                                                                                                      |
| 備考       | プ ログラムのクリティカルなコード は独立して処理を行う必要があるため、そのコード を実行する前に、プ ロセッサによる割り込みをディ ィズエーブル(無効)にしなければいけません。<br>OS_ENTER_CRITICAL 関数をコールして、明示的に割り込みを禁止し、クリティカルなコード を実行します。この関数の実行には、5 CPU クロックサイクルが必要となります。<br>➤ この関数は、次の OS_EXIT_CRITICAL 関数とペアで使用してください。 |

### OS\_EXIT\_CRITICAL

|               |                                                                                                                |
|---------------|----------------------------------------------------------------------------------------------------------------|
| 目的            | プ ロセッサの割り込みを有効にします。                                                                                            |
| 書式            | void OS_EXIT_CRITICAL(void);                                                                                   |
| コーディング 例 call | OS_ENTER_CRITICAL();<br>/* ユーザ ーコード */<br>OS_EXIT_CRITICAL();                                                  |
| 戻り値           | 無し                                                                                                             |
| 備考            | プ ロセッサの割り込みをエ ンア ーブル(有効)にします。この関数の実行には、5 CPU クロックサイクルが必要となります。<br>➤ この関数は、次の OS_ENTER_CRITICAL 関数とペアで使用してください。 |

| OSSemCreate |                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 目的          | セマフォを作成し、初期化します。                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 書式          | OS_EVENT *OSSemCreate(unsigned value);                                                                                                                                                                                                                                                                                                                                                                                                     |
| 引数          | <p>OS_EVENT は、下記のデータ構造を持つ構造体で、イベントコントロールブロック(ECB)と呼ばれるイベントリストを保持します。</p> <pre>typedef struct os_event {     unsigned char  OSEventTbl[8];    /* イベント発生待ちタスクに対応したグループ */     unsigned char  OSEventGrp;       /* イベント発生待ちタスクリスト */     long  OSEventCnt;                /* イベントがセマフォの場合に使われるカウンタ */     void *OSEventPtr;                /* メッセージ 又はキュー構造体へのポインタ */ } OS_EVENT;</pre> <p>unsigned value<br/>セマフォの初期値。0~32767 の範囲で指定します。</p> |
| コーディング例     | sem_time = OSSemCreate(1); /* セマフォ set_time 作成し、初期値 1 で初期化 */                                                                                                                                                                                                                                                                                                                                                                              |
| 戻り値         | セマフォに割り当てられたイベントコントロールブロック(ECB)へのポインタを返します。イベントコントロールブロック(ECB)が利用できない場合は、NUL を返します。                                                                                                                                                                                                                                                                                                                                                        |
| 備考          | <p>セマフォを作成し、初期化します。セマフォは</p> <ul style="list-style-type: none"> <li>➢ タスクは、割り込みサブルーチン(ISR)、タスクのどちらかと同期することができます。</li> <li>➢ リソースへの排他アクセスを得ることができます。</li> <li>➢ イベントの発生を通知します。</li> </ul> <p>セマフォを使用する前に必ず、作成・初期化を実行しなければいけません。この関数を割り込みサブルーチン(ISR)からコールすることはできません。</p>                                                                                                                                                                     |
| OSSemPend   |                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| 目的          | タスクをイベントリストに登録します。                                                                                                                                                                                                                                                                                                                                                                                                                         |
| 書式          | unsigned char OSSemPend (OS_EVENT *pevent, unsigned long timeout, unsigned char *err);                                                                                                                                                                                                                                                                                                                                                     |
| 引数          | <p>OS_EVENT *pevent<br/>セマフォを作成した際に返されたポインタ。</p> <p>unsigned long timeout<br/>セマフォを受け取るまでの待ち時間(クロックチック数)。ここで指定したクロックチック数の間にセマフォを受け取ることができなかった場合、タスクに処理を戻します。0 を指定すると、タスクはセマフォを永遠に待ちます。指定可能な最大値は、65535 です。</p> <p>unsigned char *err<br/>エラーコードを保持する変数へのポインタ。下記の何れかのエラーコードがセットされます。</p> <ul style="list-style-type: none"> <li>➢ OS_NO_ERR       セマフォが利用可能です</li> <li>➢ OS_TIMEOUT      タイムアウトが発生しました</li> </ul>                            |
| コーディング例     | OSSemPend (sem_time, 0, &err);                                                                                                                                                                                                                                                                                                                                                                                                             |
| 戻り値         | 無し                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| 説明          | <p>タスクが OSSemPend 関数をコールした時にセマフォの値が 1 以上であれば、セマフォを -1 した後、OSSemPend 関数は、コール元に戻ります。セマフォの値が 0 以下であれば、セマフォを -1 し、セマフォのイベントリストにコールしたタスクに登録します。そのため、このタスクは、タスク又は割り込みハンドラがセマフォをリリースするか、イベント発生サブルーチンを受け取るまで待ち状態となります。この場合、再スケジュールが発生し、実行準備ができていた次の最優先タスクが CPU を使用します。セマフォを待つ場合は、タイムアウト値を設定することをお奨めします。</p> <p>セマフォを使用する前に必ず、作成・初期化を実行しなければいけません。この関数を割り込みサブルーチン(ISR)からコールすることはできません。</p>                                                          |

| OSSemPost    |                                                                                                                                                                                                                                                                         |
|--------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 目的           | セマフォのシグナルを発生させます。                                                                                                                                                                                                                                                       |
| 書式           | unsigned char OSSemPost (OS_EVENT *pevent);                                                                                                                                                                                                                             |
| 引数           | OS_EVENT *pevent<br>セマフォを作成した際に返されたポインタ。                                                                                                                                                                                                                                |
| コーディング例      | OSSemPost (sem_time);                                                                                                                                                                                                                                                   |
| 戻り値          | OS_NO_ERR                      セマフォが利用可能です<br>OS_TIMEOUT                    タイムアウトが発生しました                                                                                                                                                                               |
| 説明           | この関数は、セマフォのシグナルを発生させたい場合に使用します。セマフォの値が0以上の場合、セマフォを+1した後、OSSemPost関数は、コール元に戻ります。セマフォがマイナス値の場合、タスクはセマフォがシグナルを発生するのを待ちます。この場合、OSSemPost関数は、セマフォを待っている最優先タスクを待ち行列から削除し、このタスクを実行可能にします。そして、再開したタスクが実行可能な最優先タスクであるかを判断するために、スケジューラがコールされます。セマフォを使用する前に必ず、作成・初期化を実行しなければいけません。 |
| OSTaskCreate |                                                                                                                                                                                                                                                                         |
| 目的           | タスクを生成します。                                                                                                                                                                                                                                                              |
| 書式           | unsigned char OSTaskCreate (void (*task)(void *pd), void *pdata, unsigned char *pstk, unsigned long stk_size, unsigned char piro);                                                                                                                                      |
| 引数           | void (*task)<br>タスクコードへのポインタ。                                                                                                                                                                                                                                           |
|              | void *pdata<br>タスク作成時にパラメータを渡すためのデータエリアへのポインタ。                                                                                                                                                                                                                          |
|              | unsigned char *pstk<br>タスクスタックへの先頭ポインタ。スタックは、ローカル変数・関数引数・戻りアドレス・割り込み処理中のCPUレジスタを保存するために使用されます。                                                                                                                                                                          |
|              | unsigned long stk_size<br>スタックサイズ。スタックサイズは、タスクの要件と割り込み時の罫を予想して定義します。タスク自身と全ての罫される関数、それらと同様に罫を考慮した割り込み要求で使用するローカル変数の保存に必要とするバイト数を考慮して、スタックサイズを決定します。                                                                                                                     |
|              | unsigned char piro<br>タスクのプライオリティレベルです。各タスクに重複しないユニークなプライオリティレベルを割り当てる必要があります。値が低いほどプライオリティレベルが高くなります。                                                                                                                                                                  |
| コーディング例      | OSTaskCreate (beep_task, (void *)0, beep_stk, 256, 10);<br>/* beep_task タスクをプライオリティレベル10で生成します */                                                                                                                                                                       |
| 戻り値          | OS_NO_ERR                      正常終了<br>OS_PRIO_EXIST                  同じプライオリティレベルが既に存在します                                                                                                                                                                              |
| 備考           | この関数は、アプリケーションプログラムでタスクを作成したい場合に使用します。タスクは、μC/OSにて管理されており、マルチタスク又はタスクの実行に先立って作成することができます。タスクを割り込みサブルーチン(ISR)から生成することはできません。                                                                                                                                             |

| OSTaskDel |                                                                                                                                                                                                                                   |
|-----------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 目的        | タスクを削除します。                                                                                                                                                                                                                        |
| 書式        | unsigned char OSTaskDel (unsigned char piro);                                                                                                                                                                                     |
| 引数        | unsigned char piro<br>タスクのプライオリティレベル。                                                                                                                                                                                             |
| コーディング例   | OSTaskDel (10);                      /* プライオリティレベル 10 のタスクを削除します */                                                                                                                                                               |
| 戻り値       | OS_TASK_DEL_IDLE      削除しようとするタスクは、アイドル中です<br>OS_TASK_DEL_ERR      削除しようとするタスクは、存在しません<br>OS_NO_ERR              タスクは、削除されました                                                                                                     |
| 備考        | ユーザアプリケーションから引数で指定されたプライオリティレベルに該当するタスクを削除します。自身のプライオリティレベルを指定することによって、コール元のタスクを削除することができます。削除されたタスクは、停止状態に戻り、再度アクティブにする場合は、OSTaskCreate 関数をコールします。割り込みサブルーチン(ISR)からタスクを削除することはできません。この関数は、μC/OS のアイドルタスクを削除しようとしていないことが確認されています。 |
| OSTimeDly |                                                                                                                                                                                                                                   |
| 目的        | 指定の CPU クロック数の間、タスクを遅延します。                                                                                                                                                                                                        |
| 書式        | void OSTimeDly (unsigned long ticks);                                                                                                                                                                                             |
| 引数        | unsigned long ticks<br>タスクの遅延時間(1~65535)。0 は無限遅延を意味します。<br>8000/8200/8300/8400/8700 シリーズでは、1/200 秒(5 ミリ秒)単位の設定となります。<br>8500 シリーズでは、1/256 秒単位の設定となります。                                                                            |
| コーディング例   | OSTimeDly(10);                      /* タスクデレイ 10 x 5 ミリ秒(8000/8300 シリーズ) */                                                                                                                                                       |
| 戻り値       | 無し                                                                                                                                                                                                                                |
| 説明        | 引数で指定された CPU クロック数の間、タスクを遅延します。クロック数を 1 以上に指定した場合、常に再スケジュールが発生します。指定可能な範囲は、1~65535 です。この関数を割り込みサブルーチン(ISR)からコールすることはできません。                                                                                                        |

## 付録1 ScannerDesTbl 変数

バーコードパラメータ表 1

| バイト | ビット | 説明                                                                          | デフォルト | 対象スキャナ    |
|-----|-----|-----------------------------------------------------------------------------|-------|-----------|
| 0   | 7   | 1 : Code 39 読取り有り<br>0 : Code 39 読取り無し                                      | 1     | CCD,Laser |
|     | 6   | 1 : Italian Pharmacode 読取り有り<br>0 : Italian Pharmacode 読取り無し                | 0     | CCD,Laser |
|     | 5   | 1 : CIP 39 読取り有り<br>0 : CIP 39 読取り無し                                        | 0     | CCD,Laser |
|     | 4   | 1 : Industrial 25 読取り有り<br>0 : Industrial 25 読取り無し                          | 1     | CCD,Laser |
|     | 3   | 1 : Interleaved 25 読取り有り<br>0 : Interleaved 25 読取り無し                        | 1     | CCD,Laser |
|     | 2   | 1 : Matrix 25 読取り有り<br>0 : Matrix 25 読取り無し                                  | 0     | CCD,Laser |
|     | 1   | 1 : Codabar(NW7) 読取り有り<br>0 : Codabar(NW7) 読取り無し                            | 1     | CCD,Laser |
|     | 0   | 1 : Code 93 読取り有り<br>0 : Code 93 読取り無し                                      | 1     | CCD,Laser |
| 1   | 7   | 1 : Code 128 & EAN-128 読取り有り<br>0 : Code 128 & EAN-128 読取り無し                | 1     | CCD,Laser |
|     | 6   | 1 : UPC-E 読取り有り<br>0 : UPC-E 読取り無し                                          | 1     | CCD,Laser |
|     | 5   | 1 : UPC-E Addon 2 読取り有り<br>0 : UPC-E Addon 2 読取り無し                          | 0     | CCD,Laser |
|     | 4   | 1 : UPC-E Addon 5 読取り有り<br>0 : UPC-E Addon 5 読取り無し                          | 0     | CCD,Laser |
|     | 3   | 1 : EAN-8 読取り有り<br>0 : EAN-8 読取り無し                                          | 1     | CCD,Laser |
|     | 2   | 1 : EAN-8 Addon 2 読取り有り<br>0 : EAN-8 Addon 2 読取り無し                          | 0     | CCD,Laser |
|     | 1   | 1 : EAN-8 Addon 5 読取り有り<br>0 : EAN-8 Addon 5 読取り無し                          | 0     | CCD,Laser |
|     | 0   | 1 : EAN-13 読取り有り<br>0 : EAN-13 読取り無し                                        | 1     | CCD,Laser |
| 2   | 7   | 1 : EAN-13 Addon 2 読取り有り<br>0 : EAN-13 Addon 2 読取り無し                        | 0     | CCD,Laser |
|     | 6   | 1 : EAN-13 Addon 5 読取り有り<br>0 : EAN-13 Addon 5 読取り無し                        | 0     | CCD,Laser |
|     | 5   | 1 : MSI 読取り有り<br>0 : MSI 読取り無し                                              | 0     | CCD,Laser |
|     | 4   | 1 : Plessey 読取り有り<br>0 : Plessey 読取り無し                                      | 0     | CCD,Laser |
|     | 3   | 1 : Coop 25 読取り有り<br>0 : Coop 25 読取り無し<br>(8500 シリーズは Coop 25 をサポートしていません。) | 0     | CCD,Laser |
|     | 2   | 1 : Telepen 読取り有り<br>0 : Telepen 読取り無し                                      | 0     | CCD,Laser |
|     | 1   | 1 : 利ザル Telepen (数字)<br>0 : AIM Telepen (フルアスキー)                            | 0     | CCD,Laser |
|     | 0   | 1 : RSS Limited 読取り有り<br>0 : RSS Limited 読取り無し                              | 0     | CCD,Laser |
| 3   | 7   | 予約                                                                          |       |           |
|     | 6   | 1 : RSS-14 & RSS Expanded 読取り有り<br>0 : RSS-14 & RSS Expanded 読取り無し          | 0     | CCD,Laser |
|     | 5   | 1 : RSS-14 コード ID 送信有り<br>0 : RSS-14 コード ID 送信無し                            | 1     | CCD,Laser |

|   |       |                                                                                                                                                       |    |           |
|---|-------|-------------------------------------------------------------------------------------------------------------------------------------------------------|----|-----------|
|   | 4     | 1 : RSS-14 アプリケーション ID 送信有り<br>0 : RSS-14 アプリケーション ID 送信無し                                                                                            | 1  | CCD,Laser |
|   | 3     | 1 : RSS-14 チェックサム 送信有り<br>0 : RSS-14 チェックサム 送信無し                                                                                                      | 1  | CCD,Laser |
|   | 2     | 1 : RSS Limited コード ID 送信有り<br>0 : RSS Limited コード ID 送信無し                                                                                            | 1  | CCD,Laser |
|   | 1     | 1 : RSS Limited アプリケーション ID 送信有り<br>0 : RSS Limited アプリケーション ID 送信無し                                                                                  | 1  | CCD,Laser |
|   | 0     | 1 : RSS Limited チェックサム 送信有り<br>0 : RSS Limited チェックサム 送信無し                                                                                            | 1  | CCD,Laser |
|   |       |                                                                                                                                                       |    |           |
| 4 | 7     | 1 : RSS14 Expanded コード ID 送信有り<br>0 : RSS14 Expanded コード ID 送信無し                                                                                      | 1  | CCD,Laser |
|   | 6     | 1 : UPC-E1 & UPC-E0 有効<br>0 : UPC-E0 のみ                                                                                                               | 0  | CCD,Laser |
|   | 5 - 2 | 予約                                                                                                                                                    |    |           |
|   | 1     | 1 : Coop 25 チェックサム 検査有り<br>0 : Coop 25 チェックサム 検査無し<br>(8500 シリーズ は Coop 25 をサポートしていません。)                                                              | 0  | CCD,Laser |
|   | 0     | 1 : Coop 25 チェックサム 送信有り<br>0 : Coop 25 チェックサム 送信無し<br>(8500 シリーズ は Coop 25 をサポートしていません。)                                                              | 1  | CCD,Laser |
| 5 | 7     | 1 : Code 39 スタート/ストップ キャラクタ送信有り<br>0 : Code 39 スタート/ストップ キャラクタ送信無し                                                                                    | 0  | CCD,Laser |
|   | 6     | 1 : Code 39 チェックサム 検査有り<br>0 : Code 39 チェックサム 検査無し                                                                                                    | 0  | CCD,Laser |
|   | 5     | 1 : Code 39 チェックサム 送信有り<br>0 : Code 39 チェックサム 送信無し                                                                                                    | 1  | CCD,Laser |
|   | 4     | 1 : Code 39 フルアスキー<br>0 : Code 39 標準                                                                                                                  | 0  | CCD,Laser |
|   | 3     | 1 : Italian Pharmacode チェックサム 送信有り<br>0 : Italian Pharmacode チェックサム 送信無し                                                                              | 0  | CCD,Laser |
|   | 2     | 1 : CIP 39 チェックサム 送信有り<br>0 : CIP 39 チェックサム 送信無し                                                                                                      | 0  | CCD,Laser |
|   | 1     | 1 : Interleaved 25 チェックサム 検査有り<br>0 : Interleaved 25 チェックサム 検査無し                                                                                      | 0  | CCD,Laser |
|   | 0     | 1 : Interleaved 25 チェックサム 送信有り<br>0 : Interleaved 25 チェックサム 送信無し                                                                                      | 1  | CCD,Laser |
|   |       |                                                                                                                                                       |    |           |
|   |       |                                                                                                                                                       |    |           |
| 6 | 7     | 1 : Industrial 25 チェックサム 検査有り<br>0 : Industrial 25 チェックサム 検査無し                                                                                        | 0  | CCD,Laser |
|   | 6     | 1 : Industrial 25 チェックサム 送信有り<br>0 : Industrial 25 チェックサム 送信無し                                                                                        | 1  | CCD,Laser |
|   | 5     | 1 : Matrix 25 チェックサム 検査有り<br>0 : Matrix 25 チェックサム 検査無し                                                                                                | 0  | CCD,Laser |
|   | 4     | 1 : Matrix 25 チェックサム 送信有り<br>0 : Matrix 25 チェックサム 送信無し                                                                                                | 1  | CCD,Laser |
|   | 3 - 2 | Interleaved 25 スタート/ストップ パターン<br>00 : Industrial 25 スタート/ストップ パターン<br>01 : Interleaved 25 スタート/ストップ パターン<br>10 : Matrix 25 スタート/ストップ パターン<br>11 : 未定義 | 01 | CCD,Laser |
|   | 1 - 0 | Industrial 25 スタート/ストップ パターン<br>00 : Industrial 25 スタート/ストップ パターン<br>01 : Interleaved 25 スタート/ストップ パターン<br>10 : Matrix 25 スタート/ストップ パターン<br>11 : 未定義  | 00 | CCD,Laser |
|   |       |                                                                                                                                                       |    |           |
| 7 | 7 - 6 | Matrix 25 スタート/ストップ パターン<br>00 : Industrial 25 スタート/ストップ パターン<br>01 : Interleaved 25 スタート/ストップ パターン<br>10 : Matrix 25 スタート/ストップ パターン<br>11 : 未定義      | 10 | CCD,Laser |

|    |       |                                                                                                        |        |           |
|----|-------|--------------------------------------------------------------------------------------------------------|--------|-----------|
|    | 5 - 4 | Codabar(NW7) スタート/ストップ パターン<br>00 : abcd/abcd<br>01 : abcd/tn*e<br>10 : ABCD/ABCD<br>11 : ABCD/TN*E    | 00     | CCD,Laser |
|    | 3     | 1 : Codabar(NW7) スタート/ストップ キャラクタ送信有り<br>0 : Codabar(NW7) スタート/ストップ キャラクタ送信無し                           | 0      | CCD,Laser |
|    | 2 - 0 | 予約                                                                                                     |        |           |
| 8  | 7 - 0 | 予約                                                                                                     |        |           |
| 9  | 7 - 6 | MSI チェックデジットタイプ<br>00 : シングルモジュラス 10<br>01 : ダブルモジュラス 10<br>10 : モジュラス 11 & 10<br>11 : 未定義             | 10     | CCD,Laser |
|    | 5 - 4 | MSI チェックデジット送信<br>00 : ラストチェックデジット送信無し<br>01 : チェックデジット両方送信有り<br>10 : チェックデジット両方送信無し                   | 01     | CCD,Laser |
|    | 3     | 1 : Plessey チェックデジット送信有り<br>0 : Plessey チェックデジット送信無し                                                   | 1      | CCD,Laser |
|    | 2     | 1 : Plessey 標準 → UK Plessey 変換有り<br>0 : Plessey 標準 → UK Plessey 変換無し                                   | 1      | CCD,Laser |
|    | 1     | 1 : UPC-E → UPC-A 変換有り<br>0 : UPC-E → UPC-A 変換無し                                                       | 0      | CCD,Laser |
|    | 0     | 1 : UPC-A → EAN-13 変換有り<br>0 : UPC-A → EAN-13 変換無し                                                     | 1      | CCD,Laser |
|    |       |                                                                                                        |        |           |
| 10 | 7     | 1 : ISBN 変換有り<br>0 : ISBN 変換無し                                                                         | 0      | CCD,Laser |
|    | 6     | 1 : ISSN 変換有り<br>0 : ISSN 変換無し                                                                         | 0      | CCD,Laser |
|    | 5     | 1 : UPC-E チェックデジット送信有り<br>0 : UPC-E チェックデジット送信無し                                                       | 1      | CCD,Laser |
|    | 4     | 1 : UPC-A チェックデジット送信有り<br>0 : UPC-A チェックデジット送信無し                                                       | 1      | CCD,Laser |
|    | 3     | 1 : EAN-8 チェックデジット送信有り<br>0 : EAN-8 チェックデジット送信無し                                                       | 1      | CCD,Laser |
|    | 2     | 1 : EAN-13 チェックデジット送信有り<br>0 : EAN-13 チェックデジット送信無し                                                     | 1      | CCD,Laser |
|    | 1     | 1 : UPC-E システムパッド送信有り<br>0 : UPC-E システムパッド送信無し                                                         | 0      | CCD,Laser |
|    | 0     | 1 : UPC-A システムパッド送信有り<br>0 : UPC-A システムパッド送信無し                                                         | 1      | CCD,Laser |
| 11 | 7     | 1 : EAN-8 → EAN-13 変換有り<br>0 : EAN-8 → EAN-13 変換無し                                                     | 0      | CCD,Laser |
|    | 6     | 予約                                                                                                     |        |           |
|    | 5     | 1 : GTIN 読取り有り<br>0 : GTIN 読取り無し                                                                       | 0      | CCD,Laser |
|    | 4     | 1 : 初パコード 読取り有り<br>0 : 初パコード 読取り無し                                                                     | 1      | CCD,Laser |
|    | 3 - 2 | 00 : 読取り照合無し(リダボート 1)<br>01 : 読取り照合 1 回(リダボート 1)<br>10 : 読取り照合 2 回(リダボート 1)<br>11 : 読取り照合 3 回(リダボート 1) | 00     | CCD,Laser |
|    | 1     | 1 : UPC-E1 トリアルチェック有り<br>0 : UPC-E1 トリアルチェック無し                                                         | 0      | CCD,Laser |
|    | 0     | 予約                                                                                                     |        |           |
| 12 | 7     | 1 : Industrial 25 桁数チェック 最大桁数/最小桁数<br>0 : Industrial 25 桁数チェック 固定桁数                                    | 1      | CCD,Laser |
|    | 6 - 0 | Industrial 25 最大桁数/固定桁数 1                                                                              | Max.64 | CCD,Laser |
| 13 | 7 - 0 | Industrial 25 最小桁数/固定桁数 2                                                                              | Min.1  | CCD,Laser |



|    |       |                                                                                                                                                                                    |        |           |
|----|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|-----------|
| 14 | 7     | 1 : Interlieved 25 桁数チェック 最大桁数/最小桁数<br>0 : Interlieved 25 桁数チェック 固定桁数                                                                                                              | 1      | CCD,Laser |
|    | 6 - 0 | Interlieved 25 最大桁数/固定桁数 1                                                                                                                                                         | Max.64 | CCD,Laser |
| 15 | 7 - 0 | Interlieved 25 最小桁数/固定桁数 2                                                                                                                                                         | Min.1  | CCD,Laser |
| 16 | 7     | 1 : Matrix 25 桁数チェック 最大桁数/最小桁数<br>0 : Matrix 25 桁数チェック 固定桁数                                                                                                                        | 1      | CCD,Laser |
|    | 6 - 0 | Matrix 25 最大桁数/固定桁数 1                                                                                                                                                              | Max.64 | CCD,Laser |
| 17 | 7 - 0 | Matrix 25 最小桁数/固定桁数 2                                                                                                                                                              | Min.1  | CCD,Laser |
| 18 | 7     | 1 : MSI 桁数チェック 最大桁数/最小桁数<br>0 : MSI 桁数チェック 固定桁数                                                                                                                                    | 1      | CCD,Laser |
|    | 6 - 0 | MSI 最大桁数/固定桁数 1                                                                                                                                                                    | Max.64 | CCD,Laser |
| 19 | 7 - 0 | MSI 最小桁数/固定桁数 2                                                                                                                                                                    | Min.1  | CCD,Laser |
| 20 | 7 - 4 | 読取りモード (リダボート1)<br>0000 : オートモード<br>0001 : コンティナスモード<br>0010 : オートパワーオフモード<br>0011 : ネットモード<br>0100 : モニタリモード<br>0101 : リポートモード<br>0110 : レザモード<br>0111 : テストモード<br>1000 : イミシグモード | 0110   | CCD,Laser |
|    | 3 - 0 | 予約                                                                                                                                                                                 |        |           |
| 21 | 7 - 0 | 読取りタイムアウト時間(秒単位, オートオフ, オートパワーオフモード用)<br>1~255、0 はタイムアウトなし                                                                                                                         | 3 Sec  | CCD,Laser |
| 22 | 7 - 6 | バイト1-ビット7 が ON の場合<br>00 : Code 128 & EAN-128 読取り<br>(古いフォーマットと互換)<br>01 : EAN-128 のみ読取り<br>10 : Code 128 のみ読取り<br>11 : Code 128 & EAN-128 読取り                                     | 00     | CCD,Laser |
|    | 5     | バイト1-ビット7 が ON の場合<br>1 : EAN-128 コード ID 除去<br>0 : EAN-128 コード ID 除去なし<br>(古いフォーマットと互換)                                                                                            | 0      | CCD,Laser |
|    | 4     | 1 : ISBT 128 読取り有り<br>0 : ISBT 128 読取り無し                                                                                                                                           | 1      | CCD,Laser |
|    | 3 - 0 | 予約                                                                                                                                                                                 |        |           |

バーコードパラメータ表 2

| バイト | ビット                       | 説明                                                                                                                                                    | デフォルト | 対象スキャナ                    |
|-----|---------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------|-------|---------------------------|
| 0   | 7                         | 1 : Code 39 読取り有り<br>0 : Code 39 読取り無し                                                                                                                | 1     | 2D,<br>(Extra) Long range |
|     | 6                         | 1 : Code 32(Italian Pharmacode)読取り有り<br>0 : Code 32 読取り無し                                                                                             | 0     | 2D,<br>(Extra) Long range |
|     | 5                         | 無し                                                                                                                                                    |       |                           |
|     | 4                         | 無し                                                                                                                                                    |       |                           |
|     | 3                         | 1 : Interleaved 25 読取り有り<br>0 : Interleaved 25 読取り無し                                                                                                  | 1     | 2D,<br>(Extra) Long range |
|     | 2                         | 1 : Matrix 25 読取り有り<br>0 : Matrix 25 読取り無し                                                                                                            | 0     | 8200,8400,<br>8700-2D     |
|     | 1                         | 1 : Codabar(NW7) 読取り有り<br>0 : Codabar(NW7) 読取り無し                                                                                                      | 1     | 2D,<br>(Extra) Long range |
|     | 0                         | 1 : Code 93 読取り有り<br>0 : Code 93 読取り無し                                                                                                                | 1     | 2D,<br>(Extra) Long range |
| 1   | 7                         | 1 : Code 128 読取り有り<br>0 : Code 128 読取り無し                                                                                                              | 1     | 2D,<br>(Extra) Long range |
|     | 6                         | 1 : UPC-E0 読取り有り<br>0 : UPC-E0 読取り無し                                                                                                                  | 1     | 2D,<br>(Extra) Long range |
|     | 3                         | 1 : EAN-8 読取り有り<br>0 : EAN-8 読取り無し                                                                                                                    | 1     | 2D,<br>(Extra) Long range |
|     | 0                         | 1 : EAN-13 読取り有り<br>0 : EAN-13 読取り無し                                                                                                                  | 1     | 2D,<br>(Extra) Long range |
|     | 5 or<br>4 or<br>3 or<br>2 | 1 : UPC と EAN で Addon 2 と 5 のみ読取り可<br>(いずれかのビットが ON)<br>0 : UPC と EAN で Addon 2 と 5 のみ読取り不可<br>(すべてのビットが OFF)<br>バイト 2-ビット 7~6、バイト 27-ビット 6 または 4 も対象 | 0     | 2D,<br>(Extra) Long range |
| 2   | 7 - 6                     | 上記参照                                                                                                                                                  | 0     | 2D,<br>(Extra) Long range |
|     | 5                         | 1 : MSI 読取り有り<br>0 : MSI 読取り無し                                                                                                                        | 1     | 2D,<br>(Extra) Long range |
|     | 4                         | 無し                                                                                                                                                    |       |                           |
|     | 3                         | 予約                                                                                                                                                    |       |                           |
|     | 2                         | 無し                                                                                                                                                    |       |                           |
|     | 1                         | 無し                                                                                                                                                    |       |                           |
|     | 0                         | 無し                                                                                                                                                    |       |                           |
| 3   | 7 - 0                     | 無し                                                                                                                                                    |       |                           |
| 4   | 7 - 6                     | 無し                                                                                                                                                    |       |                           |
|     | 5 - 0                     | 予約                                                                                                                                                    |       |                           |
| 5   | 7                         | 無し                                                                                                                                                    |       |                           |
|     | 6                         | 1 : Code 39 チェックディジット検査有り<br>0 : Code 39 チェックディジット検査無し                                                                                                | 0     | 2D,<br>(Extra) Long range |
|     | 5                         | 1 : Code 39 チェックディジット送信有り<br>0 : Code 39 チェックディジット送信無し                                                                                                | 0     | 2D,<br>(Extra) Long range |
|     | 4                         | 1 : Code 39 フルASCII<br>0 : Code 39 標準                                                                                                                 | 0     | 2D,<br>(Extra) Long range |
|     | 3 - 1                     | 無し                                                                                                                                                    |       |                           |
|     | 0                         | 1 : Interleaved 25 チェックディジット送信有り<br>0 : Interleaved 25 チェックディジット送信無し                                                                                  | 0     | 2D,<br>(Extra) Long range |
| 6   | 7 - 6                     | 予約                                                                                                                                                    |       |                           |
|     | 5                         | 1 : Matrix 25 チェックディジット検査有り<br>0 : Matrix 25 チェックディジット検査無し                                                                                            | 0     | 8200, 8400,<br>8700-2D    |
|     | 4                         | 1 : Matrix 25 チェックディジット送信有り<br>0 : Matrix 25 チェックディジット送信無し                                                                                            | 0     | 8200, 8400,<br>8700-2D    |
|     | 3 - 0                     | 予約                                                                                                                                                    |       |                           |
| 7   | 7 - 4                     | 無し                                                                                                                                                    |       |                           |

|    |       |                                                                                                                                              |        |                           |
|----|-------|----------------------------------------------------------------------------------------------------------------------------------------------|--------|---------------------------|
|    | 3     | 1 : Codabar(NW7) スタート/ストップ キャラクタ送信有り<br>0 : Codabar(NW7) スタート/ストップ キャラクタ送信無し                                                                 | 0      | 2D,<br>(Extra) Long range |
|    | 2 - 0 | 予約                                                                                                                                           |        |                           |
| 8  | 7 - 0 | 予約                                                                                                                                           |        |                           |
| 9  | 7 - 6 | MSI チェックデジットタイプ<br>00 : シングル モジュラス 10<br>01 : ダブル モジュラス 10<br>10 : モジュラス 11 & 10<br>11 : 未定義                                                 | 01     | 2D,<br>(Extra) Long range |
|    | 5 - 4 | MSI チェックデジット送信<br>00 : ラストチェックデジット送信無し<br>01 : チェックデジット両方送信有り<br>10 : チェックデジット両方送信無し                                                         | 00     | 2D,<br>(Extra) Long range |
|    | 3 - 2 | 無し                                                                                                                                           |        |                           |
|    | 1     | 1 : UPC-E0 → UPC-A 変換有り<br>0 : UPC-E0 → UPC-A 変換無し                                                                                           | 0      | 2D,<br>(Extra) Long range |
|    | 0     | 無し                                                                                                                                           |        |                           |
|    |       |                                                                                                                                              |        |                           |
| 10 | 7 - 6 | 無し                                                                                                                                           |        |                           |
|    | 5     | 1 : UPC-E0 チェックデジット送信有り<br>0 : UPC-E0 チェックデジット送信無し                                                                                           | 1      | 2D,<br>(Extra) Long range |
|    | 4     | 1 : UPC-A チェックデジット送信有り<br>0 : UPC-A チェックデジット送信無し                                                                                             | 1      | 2D,<br>(Extra) Long range |
|    | 3 - 2 | 無し                                                                                                                                           |        |                           |
|    | 1     | 1 : UPC-E0 システム番号 - 送信有り<br>0 : UPC-E0 システム番号 - 送信無し                                                                                         | 1      | 2D,<br>(Extra) Long range |
|    | 0     | 1 : UPC-A システム番号 - 送信有り<br>0 : UPC-A システム番号 - 送信無し                                                                                           | 1      | 2D,<br>(Extra) Long range |
| 11 | 7     | 1 : EAN-8 → EAN-13 変換有り<br>0 : EAN-8 → EAN-13 変換無し                                                                                           | 1      | 2D,<br>(Extra) Long range |
|    | 6     | 予約                                                                                                                                           |        |                           |
|    | 5 - 0 | 無し                                                                                                                                           |        |                           |
|    | 0     | 予約                                                                                                                                           |        |                           |
| 12 | 7 - 0 | 無し                                                                                                                                           |        |                           |
| 13 | 7 - 0 | 無し                                                                                                                                           |        |                           |
| 14 | 7     | 1 : Interleaved 25 桁数チェック 最大桁数/最小桁数<br>0 : Interleaved 25 桁数チェック 固定桁数                                                                        | 0      | 2D,<br>(Extra) Long range |
|    | 6 - 0 | Interleaved 25 最大桁数/固定桁数 1                                                                                                                   | 0      | 2D,<br>(Extra) Long range |
| 15 | 7 - 0 | Interleaved 25 最小桁数/固定桁数 2<br>桁数 1 > 桁数 2 となるよう設定してください                                                                                      | 0      | 2D,<br>(Extra) Long range |
| 16 | 7     | 1 : Matrix 25 桁数チェック 最大桁数/最小桁数<br>0 : Matrix 25 桁数チェック 固定桁数                                                                                  | 1      | 8200, 8400<br>8700-2D     |
|    | 6 - 0 | Matrix 25 最大桁数/固定桁数 1                                                                                                                        | 0      | 8200, 8400<br>8700-2D     |
| 17 | 7 - 0 | Matrix 25 最小桁数/固定桁数 2<br>桁数 1 > 桁数 2 となるよう設定してください                                                                                           | 0      | 8200, 8400<br>8700-2D     |
| 18 | 7     | 1 : MSI 桁数チェック 最大桁数/最小桁数<br>0 : MSI 桁数チェック 固定桁数                                                                                              | 1      | 2D,<br>(Extra) Long range |
|    | 6 - 0 | MSI 最大桁数/固定桁数 1                                                                                                                              | Max.31 | 2D,<br>(Extra) Long range |
| 19 | 7 - 0 | MSI 最小桁数/固定桁数 2<br>桁数 1 > 桁数 2 となるよう設定してください                                                                                                 | Min.3  | 2D,<br>(Extra) Long range |
| 20 | 7 - 4 | 読取りモード (リダポート 1)<br>1000 : イミグモード<br>0111 : テストモード<br>0110 : レザモード<br>0011 : カタネットモード<br>0001 : コンティニアモード<br>0000 : オートモード<br>上記以外の値 : レザモード | 0110   | 2D,<br>(Extra) Long range |
|    | 3 - 0 | 予約                                                                                                                                           |        |                           |

|    |        |                                                                                                                                                         |    |                           |
|----|--------|---------------------------------------------------------------------------------------------------------------------------------------------------------|----|---------------------------|
| 21 | 7 - 0  | 無し                                                                                                                                                      |    |                           |
| 22 | 7 - 0  | 予約                                                                                                                                                      |    |                           |
| 23 | 7      | 1 : Code 39 桁数チェック 最大桁数/最小桁数<br>0 : Code 39 桁数チェック 固定桁数                                                                                                 | 0  | 2D,<br>(Extra) Long range |
|    | 6 - 0  | Code 39 最大桁数/固定桁数 1                                                                                                                                     | 0  | 2D,<br>(Extra) Long range |
| 24 | 7 - 0  | Code 39 最小桁数/固定桁数 2<br>桁数 1 > 桁数 2 となるよう設定してください                                                                                                        | 0  | 2D,<br>(Extra) Long range |
| 25 | 7      | 1 : UPC-E1 システムバ - 送信有り<br>0 : UPC-E1 システムバ - 送信無し                                                                                                      | 0  | 2D,<br>(Extra) Long range |
|    | 6      | 1 : UPC-E1 チェックデジット送信有り<br>0 : UPC-E1 チェックデジット送信無し                                                                                                      | 0  | 2D,<br>(Extra) Long range |
|    | 5      | 1 : UPC/EAN コドビットコードの GS1-128 イミューションポート 有効<br>0 : UPC/EAN コドビットコードの GS1-128 イミューションポート 有効                                                              | 0  | 2D                        |
|    | 4      | 1 : TCIF Linked Code 39 読取り有り<br>0 : TCIF Linked Code 39 読取り無し                                                                                          | 1  | 2D                        |
|    | 3      | 1 : UPC-E1 → UPC-A 変換有り<br>0 : UPC-E1 → UPC-A 変換無し                                                                                                      | 0  | 2D,<br>(Extra) Long range |
|    | 2      | 1 : Code 11 読取り有り<br>0 : Code 11 読取り無し                                                                                                                  | 1  | 2D,<br>8300, 8700-LR のみ   |
|    | 1      | 1 : Bookland EAN 読取り有り<br>(ビット 1-ビット 0 が ON である必要があります)<br>0 : Bookland EAN 読取り無し                                                                       | 0  | 2D,<br>(Extra) Long range |
|    | 0      | 1 : 1 : UPC/EAN で Addon 無しと有りの共通設定有効<br>0 : 共通設定無効                                                                                                      | 0  | 2D,<br>(Extra) Long range |
| 26 | 7      | 1 : Industrial 25 読取り有り<br>0 : Industrial 25 読取り無し                                                                                                      | 1  | 2D,<br>(Extra) Long range |
|    | 6      | 1 : ISBT 128 読取り有り<br>0 : ISBT 128 読取り無し                                                                                                                | 1  | 2D,<br>(Extra) Long range |
|    | 5      | 1 : Trioptic Code 39 読取り有り<br>0 : Trioptic Code 39 読取り無し                                                                                                | 0  | 2D,<br>(Extra) Long range |
|    | 4      | 1 : UCC/EAN-128 読取り有り<br>0 : UCC/EAN-128 読取り無し                                                                                                          | 1  | 2D,<br>(Extra) Long range |
|    | 3      | 1 : RSS → UPC/EAN 変換有り<br>0 : RSS → UPC/EAN 変換無し                                                                                                        | 0  | 2D,<br>(Extra) Long range |
|    | 2      | 1 : RSS Expanded 読取り有り<br>0 : RSS Expanded 読取り無し                                                                                                        | 1  | 2D,<br>(Extra) Long range |
|    | 1      | 1 : RSS Limited 読取り有り<br>0 : RSS Limited 読取り無し                                                                                                          | 1  | 2D,<br>(Extra) Long range |
|    | 0      | 1 : RSS-14 読取り有り<br>0 : RSS-14 読取り無し                                                                                                                    | 1  | 2D,<br>(Extra) Long range |
| 27 | 7      | 1 : UPC-A 読取り有り<br>0 : UPC-A 読取り無し                                                                                                                      | 1  | 2D,<br>(Extra) Long range |
|    | 5      | 1 : UPC-E1 読取り有り<br>0 : UPC-E1 読取り無し                                                                                                                    | 0  | 2D,<br>(Extra) Long range |
|    | 6 or 4 | 1 : UPC と EAN で Addon 2 と 5 のみ読取り可<br>(いずれかのビットが ON)<br>0 : UPC と EAN で Addon 2 と 5 のみ読取り不可<br>(すべてのビットが OFF)<br>ビット 1-ビット 5, 4, 2, 1、ビット 2-ビット 7~6 も対象 | 0  | 2D,<br>(Extra) Long range |
|    | 3 - 2  | 00 : UPC リンクしない<br>01 : UPC 常にリンク<br>10 : UPC コドビット自動識別<br>11 : 未定義                                                                                     | 01 | 2D                        |
|    | 1      | 1 : コドビット CC-A/B 読取り有り<br>0 : コドビット CC-A/B 読取り無し                                                                                                        | 0  | 2D                        |
|    | 0      | 1 : コドビット CC-C 読取り有り<br>0 : コドビット CC-C 読取り無し                                                                                                            | 0  | 2D                        |
| 28 | 7      | 1 : Code 93 桁数チェック 最大桁数/最小桁数<br>0 : Code 93 桁数チェック 固定桁数                                                                                                 | 0  | 2D,<br>(Extra) Long range |

|    |       |                                                                                                                               |       |                           |
|----|-------|-------------------------------------------------------------------------------------------------------------------------------|-------|---------------------------|
|    | 6 - 0 | Code 93 最大桁数/固定桁数 1                                                                                                           | 0     | 2D,<br>(Extra) Long range |
| 29 | 7 - 0 | Code 93 最小桁数/固定桁数 2<br>桁数 1 > 桁数 2 となるよう設定してください                                                                              | 0     | 2D,<br>(Extra) Long range |
| 30 | 7     | 1 : Code 11 桁数チェック 最大桁数/最小桁数<br>0 : Code 11 桁数チェック 固定桁数                                                                       | 0     | 2D,<br>8300, 8700-LR のみ   |
|    | 6 - 0 | Code 11 最大桁数/固定桁数 1                                                                                                           | 0     | 2D,<br>8300, 8700-LR のみ   |
| 31 | 7 - 0 | Code 11 最小桁数/固定桁数 2<br>桁数 1 > 桁数 2 となるよう設定してください                                                                              | 0     | 2D,<br>8300, 8700-LR のみ   |
| 32 | 7     | 1 : Industrial 25 桁数チェック 最大桁数/最小桁数<br>0 : Industrial 25 桁数チェック 固定桁数                                                           | 0     | 2D,<br>(Extra) Long range |
|    | 6 - 0 | Industrial 25 最大桁数/固定桁数 1                                                                                                     | 0     | 2D,<br>(Extra) Long range |
| 33 | 7 - 0 | Industrial 25 最小桁数/固定桁数 2<br>桁数 1 > 桁数 2 となるよう設定してください                                                                        | 0     | 2D,<br>(Extra) Long range |
| 34 | 7     | 1 : Codabar(NW7)桁数チェック 最大桁数/最小桁数<br>0 : Codabar(NW7)桁数チェック 固定桁数                                                               | 0     | 2D,<br>(Extra) Long range |
|    | 6 - 0 | Codabar(NW7)最大桁数/固定桁数 1                                                                                                       | 0     | 2D,<br>(Extra) Long range |
| 35 | 7 - 0 | Codabar(NW7)最小桁数/固定桁数 2<br>桁数 1 > 桁数 2 となるよう設定してください                                                                          | 0     | 2D,<br>(Extra) Long range |
| 36 | 7     | 1 : US Postal チェックデジット送信有り<br>0 : US Postal チェックデジット送信無し                                                                      | 1     | 2D                        |
|    | 6     | 1 : Maxicode 読取り有り<br>0 : Maxicode 読取り無し                                                                                      | 1     | 2D                        |
|    | 5     | 1 : Data Matrix 読取り有り<br>0 : Data Matrix 読取り無し                                                                                | 1     | 2D                        |
|    | 4     | 1 : QR Code 読取り有り<br>0 : QR Code 読取り無し                                                                                        | 1     | 2D                        |
|    | 3     | 1 : US Planet 読取り有り<br>0 : US Planet 読取り無し                                                                                    | 1     | 2D                        |
|    | 2     | 1 : US Postnet 読取り有り<br>0 : US Postnet 読取り無し                                                                                  | 1     | 2D                        |
|    | 1     | 1 : MicroPDF417 読取り有り<br>0 : MicroPDF417 読取り無し                                                                                | 1     | 2D                        |
|    | 0     | 1 : PDF417 読取り有り<br>0 : PDF417 読取り無し                                                                                          | 1     | 2D                        |
|    |       |                                                                                                                               |       |                           |
| 37 | 7 - 6 | 00 : Interleaved 25 チェックデジット検査無し<br>01 : Interleaved 25 USS チェックデジット検査有り<br>10 : Interleaved 25 OPCC チェックデジット検査有り<br>11 : 未定義 | 00    | 2D,<br>(Extra) Long range |
|    | 5     | 予約                                                                                                                            |       |                           |
|    | 4     | 1 : JapanPostal 読取り有り<br>0 : JapanPostal 読取り無し                                                                                | 1     | 2D                        |
|    | 3     | 1 : Australian Postal 読取り有り<br>0 : Australian Postal 読取り無し                                                                    | 1     | 2D                        |
|    | 2     | 1 : Dutch Postal 読取り有り<br>0 : Dutch Postal 読取り無し                                                                              | 1     | 2D                        |
|    | 1     | 1 : UK Postal チェックデジット有効<br>0 : UK Postal チェックデジット無効                                                                          | 1     | 2D                        |
|    | 0     | 1 : UK Postal 読取り有り<br>0 : UK Postal 読取り無し                                                                                    | 1     | 2D                        |
|    |       |                                                                                                                               |       |                           |
| 38 | 7 - 0 | 読取りタイムアウト時間(秒単位, オートオフ, オートパワーオフモード 用)<br>1~255、0 はタイムアウトなし                                                                   | 3 Sec | 2D,<br>(Extra) Long range |
| 39 | 7     | 1 : UPC-A システムナバー-&国コード 有効<br>0 : UPC-A システムナバー-&国コード 無効                                                                      | 1     | 2D,<br>(Extra) Long range |
|    | 6     | 1 : UPC-E システムナバー-&国コード 有効<br>0 : UPC-E システムナバー-&国コード 無効                                                                      | 1     | 2D,<br>(Extra) Long range |
|    | 5     | 1 : UPC-E1 システムナバー-&国コード 有効<br>0 : UPC-E1 システムナバー-&国コード 無効                                                                    | 1     | 2D,<br>(Extra) Long range |
|    |       |                                                                                                                               |       |                           |

|    |       |                                                                                                          |    |                           |
|----|-------|----------------------------------------------------------------------------------------------------------|----|---------------------------|
|    | 4     | 1 : Interleaved 25 → EAN13 変換有り<br>0 : Interleaved 25 → EAN13 変換無し                                       | 0  | 2D,<br>(Extra) Long range |
|    | 3 - 2 | MacroPDF 送信/変換モード<br>00 : パスル<br>01 : すべてのシンボルをバッファリング / 完了時に送信<br>02 : 特定の指示なし                          | 00 | 2D                        |
|    | 1     | 1 : MacroPDF 1スケール文字有効<br>0 : MacroPDF 1スケール文字無効                                                         | 0  | 2D                        |
|    | 0     | 1 : USPS 4CB / One Code / Intelligent Mail 読取り有り<br>0 : USPS 4CB / One Code / Intelligent Mail 読取り無し     | 0  | 8200, 8400,<br>8700-2D    |
| 40 | 7 - 6 | 00 : Far フォー加<br>01 : Near フォー加<br>10 : Smart フォー加                                                       | 00 | 8500-2D                   |
|    | 5     | 1 : 照準ターゲットコード 有効<br>0 : 照準ターゲットコード 無効                                                                   | 1  | 2D                        |
|    | 4     | 1 : 照準ターゲットコード 有効<br>0 : 照準ターゲットコード 無効                                                                   | 1  | 2D                        |
|    | 3     | 1 : ビックリストモード 有効<br>0 : ビックリストモード 無効                                                                     | 0  | 8200, 8400,<br>8700-2D    |
|    | 2 - 1 | 1 次元白黒反転バースコード<br>00 : 通常 1 次元バースコード のみ読取り<br>01 : 反転 1 次元バースコード のみ読取り<br>10 : 通常/反転バースコード 読取り           | 00 | 8200, 8400,<br>8700-2D    |
|    | 0     | 1 : システムサスペンド 中、リーダ はスリープ 状態<br>0 : システムサスペンド 中、リーダ は電源 OFF 状態                                           | 0  | 8200, 8400,<br>8700-2D    |
|    |       |                                                                                                          |    |                           |
| 41 | 7     | 1 : UPU FICS 読取り有り<br>0 : UPU FICS 読取り無し                                                                 | 0  | 8200, 8400,<br>8700-2D    |
|    | 6     | UPC/EAN Bookland ISBN フォーマット<br>1 : UPC/EAN – Bookland ISBN 13<br>0 : UPC/EAN – Bookland ISBN 10         | 0  | 8200, 8400,<br>8700-2D    |
|    | 5 - 4 | Data Matrix 白黒反転<br>00 : 通常 Data Matrix のみ読取り<br>01 : 反転 Data Matrix のみ読取り<br>10 : 通常/反転 Data Matrix 読取り | 00 | 8200, 8400,<br>8700-2D    |
|    | 3 - 2 | Data Matrix 左右反転<br>00 : 通常 Data Matrix のみ読取り<br>01 : 反転 Data Matrix のみ読取り<br>10 : 通常/反転 Data Matrix 読取り | 00 | 8200, 8400,<br>8700-2D    |
|    | 1 - 0 | QR Code 白黒反転<br>00 : 通常 QR Code のみ読取り<br>01 : 反転 QR Code のみ読取り<br>10 : 通常/反転 QR Code 読取り                 | 00 | 8200, 8400,<br>8700-2D    |
| 42 | 7     | 1 : MicroQR 読取り有り<br>0 : MicroQR 読取り無し                                                                   | 1  | 8200, 8400,<br>8700-2D    |
|    | 6     | 1 : Aztec 読取り有り<br>0 : Aztec 読取り無し                                                                       | 1  | 8200, 8400,<br>8700-2D    |
|    | 5 - 4 | Aztec 白黒反転<br>00 : 通常 Aztec のみ読取り<br>01 : 反転 Aztec のみ読取り<br>10 : 通常/反転 Aztec 読取り                         | 00 | 8200, 8400,<br>8700-2D    |
|    | 3     | 1 : UCC Coupon 読取り有り<br>0 : UCC Coupon 読取り無し                                                             | 0  | 2D,<br>(Extra) Long range |
|    | 2     | 1 : Chinese 25 読取り有り<br>0 : Chinese 25 読取り無し                                                             | 0  | 8200, 8400,<br>8700-2D    |
|    | 1 - 0 | Code 11 チェックディジット検査<br>00 : 無効<br>01 : 1 チェックディジット<br>10 : 2 チェックディジット                                   | 00 | 2D,<br>8300, 8700-LR のみ   |
|    |       |                                                                                                          |    |                           |

## 付録2 バーコードパラメータ

スキャナは、さまざまな種類のバーコードを読み取ることができます。ここでは、バーコードの種類ごとに関連するパラメータを記述します。

### CCD、レーザーのスキャナ

#### CODABAR

| バイト | ビット   | 説明                                                                                                 | デフォルト | 対象スキャナ    |
|-----|-------|----------------------------------------------------------------------------------------------------|-------|-----------|
| 0   | 7     | 1 : Codabar(NW7) 読み取り有り<br>0 : Codabar(NW7) 読み取り無し                                                 | 1     | CCD,Laser |
| 7   | 5 - 4 | Codabar(NW7) スタート/ストップパターン<br>00 : abcd/abcd<br>01 : abcd/tn*e<br>10 : ABCD/ABCD<br>11 : ABCD/TN*E | 00    | CCD,Laser |
| 7   | 3     | 1 : Codabar(NW7) スタート/ストップキャラクタ送信有り<br>0 : Codabar(NW7) スタート/ストップキャラクタ送信無し                         | 0     | CCD,Laser |

#### スタート/ストップパターン選択

スタート/ストップ文字なし、または送信されるデータに含める文字のパターンを4パターンから選択します。

- abcd/abcd
- abcd/tn\*e
- ABCD/ABCD
- ABCD/TN\*E

#### スタート/ストップキャラクタ送信

送信するデータにスタート/ストップ文字を含めるかどうかを設定します。

### CODE 2 of 5

#### INDUSTRIAL 25

| バイト | ビット   | 説明                                                                                                                                               | デフォルト  | 対象スキャナ    |
|-----|-------|--------------------------------------------------------------------------------------------------------------------------------------------------|--------|-----------|
| 0   | 4     | 1 : Industrial 25 読み取り有り<br>0 : Industrial 25 読み取り無し                                                                                             | 1      | CCD,Laser |
| 6   | 7     | 1 : Industrial 25 チェックディジット検査有り<br>0 : Industrial 25 チェックディジット検査無し                                                                               | 0      | CCD,Laser |
| 6   | 6     | 1 : Industrial 25 チェックディジット送信有り<br>0 : Industrial 25 チェックディジット送信無し                                                                               | 1      | CCD,Laser |
| 6   | 1 - 0 | Industrial 25 スタート/ストップパターン<br>00 : Industrial 25 スタート/ストップパターン<br>01 : Interleaved 25 スタート/ストップパターン<br>10 : Matrix 25 スタート/ストップパターン<br>11 : 未定義 | 00     | CCD,Laser |
| 12  | 7     | 1 : Industrial 25 桁数チェック 最大桁数/最小桁数<br>0 : Industrial 25 桁数チェック 固定桁数                                                                              | 1      | CCD,Laser |
| 12  | 6 - 0 | Industrial 25 最大桁数/固定桁数 1                                                                                                                        | Max.64 | CCD,Laser |
| 12  | 7 - 0 | Industrial 25 最小桁数/固定桁数 2                                                                                                                        | Min.1  | CCD,Laser |

#### チェックディジット検査

読み取り時にチェックディジット検査を行うかどうかを設定します。

- チェックディジットに誤りがあるバーコードは受け付けません。

#### チェックディジット送信

読み取りデータにチェックディジットを含めるかどうかを設定します。

#### スタート/ストップパターン選択

さまざまな種類の2 of 5バーコードを読み取るのに適したスタート/ストップパターンを選択します。

- 例えば、航空券は Industrial 2 of 5 を使用していますが、Interleaved 2 of 5 のスタート/ストップパターンになっています。このバーコードを読み取るには Industrial 2 of 5 のスタート/ストップパターンを"Interleaved 2 of 5"に設定する必要があります。

## 桁数設定

2 of 5 バーコードの脆弱な構造のため、ショートキャンセルを起こす可能性があります。ショートキャンセルを防ぐために、正しいバーコードの桁数の範囲を設定します。

- "固定桁数"を選択した場合、最大2つの固定桁数を設定できます。  
 ➤ "最大桁数/最小桁数"を選択した場合、最大桁数と最小桁数を設定する必要があります。指定された範囲の長さのバーコードのみを受け付けます。

| INTERLEAVED 25 |       |                                                                                                                                                   |        |           |
|----------------|-------|---------------------------------------------------------------------------------------------------------------------------------------------------|--------|-----------|
| バイト            | ビット   | 説明                                                                                                                                                | デフォルト  | 対象スキャナ    |
| 0              | 3     | 1 : Interleaved 25 読取り有り<br>0 : Interleaved 25 読取り無し                                                                                              | 1      | CCD,Laser |
| 5              | 1     | 1 : Interleaved 25 チェックディジット検査有り<br>0 : Interleaved 25 チェックディジット検査無し                                                                              | 0      | CCD,Laser |
| 5              | 0     | 1 : Interleaved 25 チェックディジット送信有り<br>0 : Interleaved 25 チェックディジット送信無し                                                                              | 1      | CCD,Laser |
| 6              | 3 - 2 | Interleaved 25 スタート/ストップパターン<br>00 : Industrial 25 スタート/ストップパターン<br>01 : Interleaved 25 スタート/ストップパターン<br>10 : Matrix 25 スタート/ストップパターン<br>11 : 未定義 | 01     | CCD,Laser |
| 14             | 7     | 1 : Interleaved 25 桁数チェック 最大桁数/最小桁数<br>0 : Interleaved 25 桁数チェック 固定桁数                                                                             | 1      | CCD,Laser |
| 14             | 6 - 0 | Interleaved 25 最大桁数/固定桁数 1                                                                                                                        | Max.64 | CCD,Laser |
| 15             | 7 - 0 | Interleaved 25 最小桁数/固定桁数 2                                                                                                                        | Min.1  | CCD,Laser |

| MATRIX 25 |       |                                                                                                                                              |        |           |
|-----------|-------|----------------------------------------------------------------------------------------------------------------------------------------------|--------|-----------|
| バイト       | ビット   | 説明                                                                                                                                           | デフォルト  | 対象スキャナ    |
| 0         | 2     | 1 : Matrix 25 読取り有り<br>0 : Matrix 25 読取り無し                                                                                                   | 0      | CCD,Laser |
| 6         | 5     | 1 : Matrix 25 チェックディジット検査有り<br>0 : Matrix 25 チェックディジット検査無し                                                                                   | 0      | CCD,Laser |
| 6         | 4     | 1 : Matrix 25 チェックディジット送信有り<br>0 : Matrix 25 チェックディジット送信無し                                                                                   | 1      | CCD,Laser |
| 7         | 7 - 6 | Matrix 25 スタート/ストップパターン<br>00 : Industrial 25 スタート/ストップパターン<br>01 : Interleaved 25 スタート/ストップパターン<br>10 : Matrix 25 スタート/ストップパターン<br>11 : 未定義 | 10     | CCD,Laser |
| 16        | 7     | 1 : Matrix 25 桁数チェック 最大桁数/最小桁数<br>0 : Matrix 25 桁数チェック 固定桁数                                                                                  | 1      | CCD,Laser |
|           | 6 - 0 | Matrix 25 最大桁数/固定桁数 1                                                                                                                        | Max.64 | CCD,Laser |
| 17        | 7 - 0 | Matrix 25 最小桁数/固定桁数 2                                                                                                                        | Min.1  | CCD,Laser |

| COOP 25 |  |  |  |  |
|---------|--|--|--|--|
|---------|--|--|--|--|

Coop 25 は、8000/8200/8300/8400/8700 シリーズでサポートされています。

| バイト | ビット | 説明                                                     | デフォルト | 対象スキャナ    |
|-----|-----|--------------------------------------------------------|-------|-----------|
| 2   | 3   | 1 : Coop 25 読取り有り<br>0 : Coop 25 読取り無し                 | 0     | CCD,Laser |
| 4   | 1   | 1 : Coop 25 チェックディジット検査有り<br>0 : Coop 25 チェックディジット検査無し | 0     | CCD,Laser |
| 4   | 0   | 1 : Coop 25 チェックディジット送信有り<br>0 : Coop 25 チェックディジット送信無し | 1     | CCD,Laser |



## チェックビット検査

読取り時にチェックビット検査を行うかどうかを設定します。

➤ チェックビットに誤りがあるバーコードは受け付けません。

※ チェックビット検査は、チェックビットを送信したくないときに、チェックビットを省くことができるように有効にする必要があります。

## チェックビット送信

読取りデータにチェックビットを含めるかどうかを設定します。

### CODE 39

| バイト | ビット | 説明                                                               | デフォルト | 対象スキャナ    |
|-----|-----|------------------------------------------------------------------|-------|-----------|
| 0   | 7   | 1 : Code 39 読取り有り<br>0 : Code 39 読取り無し                           | 1     | CCD,Laser |
| 5   | 7   | 1 : Code 39 スタートストップ キャラクタ送信有り<br>0 : Code 39 スタートストップ キャラクタ送信無し | 0     | CCD,Laser |
| 5   | 6   | 1 : Code 39 チェックビット検査有り<br>0 : Code 39 チェックビット検査無し               | 0     | CCD,Laser |
| 5   | 5   | 1 : Code 39 チェックビット送信有り<br>0 : Code 39 チェックビット送信無し               | 1     | CCD,Laser |
| 5   | 4   | 1 : Code 39 フルアスキー<br>0 : Code 39 標準                             | 0     | CCD,Laser |

## スタートストップ キャラクタ送信

送信するデータにスタートストップ文字を含めるかどうかを設定します。

## チェックビット検査

読取り時にチェックビット検査を行うかどうかを設定します。

➤ チェックビットに誤りがあるバーコードは受け付けません。

## チェックビット送信

読取りデータにチェックビットを含めるかどうかを設定します。

## Code 39 フルアスキー

すべての英数字、および特殊文字を含めた Code 39 フルアスキーを標準にするかどうかを設定します。

### CODE 93

| バイト | ビット | 説明                                     | デフォルト | 対象スキャナ    |
|-----|-----|----------------------------------------|-------|-----------|
| 0   | 0   | 1 : Code 93 読取り有り<br>0 : Code 93 読取り無し | 1     | CCD,Laser |

### CODE 128 / EAN 128 / ISBT 128

| バイト | ビット   | 説明                                                                                                                                           | デフォルト | 対象スキャナ    |
|-----|-------|----------------------------------------------------------------------------------------------------------------------------------------------|-------|-----------|
| 1   | 7     | 1 : Code 128 & EAN-128 読取り有り<br>0 : Code 128 & EAN-128 読取り無し                                                                                 | 1     | CCD,Laser |
| 22  | 7 - 6 | バイト1-ビット7がONの場合<br>00 : Code 128 & EAN-128 読取り<br>(古いフォーマットとの互換)<br>01 : EAN-128 のみ読取り<br>10 : Code 128 のみ読取り<br>11 : Code 128 & EAN-128 読取り | 00    | CCD,Laser |
| 22  | 5     | バイト1-ビット7がONの場合<br>1 : EAN-128 コード ID 除去<br>0 : EAN-128 コード ID 除去なし<br>(古いフォーマットとの互換)                                                        | 0     | CCD,Laser |
| 22  | 4     | 1 : ISBT 128 読取り有り<br>0 : ISBT 128 読取り無し                                                                                                     | 1     | CCD,Laser |

## ITALIAN / FRENCH PHARMACODE

| バイト | ビット | 説明                                                                           | デフォルト | 対象スキャナ    |
|-----|-----|------------------------------------------------------------------------------|-------|-----------|
| 0   | 6   | 1 : Italian Pharmacode 読取り有り<br>0 : Italian Pharmacode 読取り無し                 | 0     | CCD,Laser |
| 0   | 5   | 1 : CIP 39 読取り有り<br>0 : CIP 39 読取り無し                                         | 0     | CCD,Laser |
| 5   | 3   | 1 : Italian Pharmacode チェックディジット送信有り<br>0 : Italian Pharmacode チェックディジット送信無し | 0     | CCD,Laser |
| 5   | 2   | 1 : CIP 39 チェックディジット送信有り<br>0 : CIP 39 チェックディジット送信無し                         | 0     | CCD,Laser |

### チェックディジット送信

読取りデータにチェックディジットを含めるかどうかを設定します。

※ スタート/ストップ 文字送信設定は Code 39 と共通です。

## MSI

| バイト | ビット   | 説明                                                                                            | デフォルト  | 対象スキャナ    |
|-----|-------|-----------------------------------------------------------------------------------------------|--------|-----------|
| 2   | 5     | 1 : MSI 読取り有り<br>0 : MSI 読取り無し                                                                | 0      | CCD,Laser |
| 9   | 7 - 6 | MSI チェックディジットタイプ<br>00 : シングル モジュラス 10<br>01 : ダブル モジュラス 10<br>10 : モジュラス 11 & 10<br>11 : 未定義 | 10     | CCD,Laser |
| 9   | 5 - 4 | MSI チェックディジット送信<br>00 : ラストチェックディジット送信無し<br>01 : チェックディジット両方送信有り<br>10 : チェックディジット両方送信無し      | 01     | CCD,Laser |
| 18  | 7     | 1 : MSI 桁数チェック 最大桁数/最小桁数<br>0 : MSI 桁数チェック 固定桁数                                               | 1      | CCD,Laser |
| 18  | 6 - 0 | MSI 最大桁数/固定桁数 1                                                                               | Max.64 | CCD,Laser |
| 19  | 7 - 0 | MSI 最小桁数/固定桁数 2                                                                               | Min.1  | CCD,Laser |

### チェックディジット検査

読取り時にチェックディジット検査を行うかどうかを設定します。

➤ チェックディジットに誤りがあるバーコードは受け付けません。

### チェックディジット送信

読取りデータにチェックディジットを含めるかどうかを設定します。

### 桁数設定

バーコードの脆弱な構造のため、ショートスキャンを起こす可能性があります。ショートスキャンを防ぐために、正しいバーコードの桁数の範囲を設定します。

➤ “固定桁数”を選択した場合、最大 2 つの固定桁数を設定できます。

➤ “最大桁数/最小桁数”を選択した場合、最大桁数と最小桁数を設定する必要があります。指定された範囲の長さのバーコードのみを受け付けます。

## 初バーコード

| バイト | ビット | 説明                                   | デフォルト | 対象スキャナ    |
|-----|-----|--------------------------------------|-------|-----------|
| 11  | 4   | 1 : 初バーコード 読取り有り<br>0 : 初バーコード 読取り無し | 1     | CCD,Laser |

## PLESSEY

| バイト | ビット | 説明                                                                   | デフォルト | 対象スキャナ    |
|-----|-----|----------------------------------------------------------------------|-------|-----------|
| 2   | 4   | 1 : Plessey 読取り有り<br>0 : Plessey 読取り無し                               | 0     | CCD,Laser |
| 9   | 3   | 1 : Plessey チェックデジット送信有り<br>0 : Plessey チェックデジット送信無し                 | 1     | CCD,Laser |
| 9   | 2   | 1 : Plessey 標準 → UK Plessey 変換有り<br>0 : Plessey 標準 → UK Plessey 変換無し | 1     | CCD,Laser |

### チェックデジット送信

読取りデータにチェックデジットを含めるかどうかを設定します。

### UK Plessey 変換

読取りデータの'A'から'X'の変換を行うかどうかを設定します。

## RSS

| バイト | ビット | 説明                                                                   | デフォルト | 対象スキャナ    |
|-----|-----|----------------------------------------------------------------------|-------|-----------|
| 2   | 0   | 1 : RSS Limited 読取り有り<br>0 : RSS Limited 読取り無し                       | 0     | CCD,Laser |
| 3   | 6   | 1 : RSS-14 & RSS Expanded 読取り有り<br>0 : RSS-14 & RSS Expanded 読取り無し   | 0     | CCD,Laser |
| 3   | 5   | 1 : RSS-14 コード ID 送信有り<br>0 : RSS-14 コード ID 送信無し                     | 1     | CCD,Laser |
| 3   | 4   | 1 : RSS-14 アプリケーション ID 送信有り<br>0 : RSS-14 アプリケーション ID 送信無し           | 1     | CCD,Laser |
| 3   | 3   | 1 : RSS-14 チェックデジット送信有り<br>0 : RSS-14 チェックデジット送信無し                   | 1     | CCD,Laser |
| 3   | 2   | 1 : RSS Limited コード ID 送信有り<br>0 : RSS Limited コード ID 送信無し           | 1     | CCD,Laser |
| 3   | 1   | 1 : RSS Limited アプリケーション ID 送信有り<br>0 : RSS Limited アプリケーション ID 送信無し | 1     | CCD,Laser |
| 3   | 0   | 1 : RSS Limited チェックデジット送信有り<br>0 : RSS Limited チェックデジット送信無し         | 1     | CCD,Laser |
| 4   | 7   | 1 : RSS14 Expanded コード ID 送信有り<br>0 : RSS14 Expanded コード ID 送信無し     | 1     | CCD,Laser |

### コード ID 送信

読取りデータにコード ID("00")を含めるかどうかを設定します。

### アプリケーション ID 送信

読取りデータにアプリケーション ID("01")を含めるかどうかを設定します。

### チェックデジット送信

読取りデータにチェックデジットを含めるかどうかを設定します。

## TELEPEN

| バイト | ビット | 説明                                                | デフォルト | 対象スキャナ    |
|-----|-----|---------------------------------------------------|-------|-----------|
| 2   | 2   | 1 : Telepen 読取り有り<br>0 : Telepen 読取り無し            | 0     | CCD,Laser |
| 2   | 1   | 1 : リジアル Telepen (数字)<br>0 : AIM Telepen (フルアスキー) | 0     | CCD,Laser |

### リジアル Telepen(数字)

Telepen のフルアスキーを無効にするかどうかを設定します。デフォルトではフルアスキーを無効にします。

➤ AIM Telepen(フルアスキー)はすべての英数字、および特殊文字を無効にします。

## UPC / EAN

### EAN-8

| バイト | ビット | 説明                                                 | デフォルト | 対象スキャナ    |
|-----|-----|----------------------------------------------------|-------|-----------|
| 1   | 3   | 1 : EAN-8 読取り有り<br>0 : EAN-8 読取り無し                 | 1     | CCD,Laser |
| 1   | 2   | 1 : EAN-8 Addon 2 読取り有り<br>0 : EAN-8 Addon 2 読取り無し | 0     | CCD,Laser |
| 1   | 1   | 1 : EAN-8 Addon 5 読取り有り<br>0 : EAN-8 Addon 5 読取り無し | 0     | CCD,Laser |
| 10  | 3   | 1 : EAN-8 チェックディジット送信有り<br>0 : EAN-8 チェックディジット送信無し | 1     | CCD,Laser |
| 11  | 7   | 1 : EAN-8 → EAN-13 変換有り<br>0 : EAN-8 → EAN-13 変換無し | 0     | CCD,Laser |

#### チェックディジット送信

読取りデータにチェックディジットを含めるかどうかを設定します。

#### EAN-8 → EAN-13 変換

読取った EAN-8 を EAN-13 に拡張するかどうかを設定します。有りの場合、以降の処理は EAN-13 の設定に従います。

### EAN-13

| バイト | ビット | 説明                                                   | デフォルト | 対象スキャナ    |
|-----|-----|------------------------------------------------------|-------|-----------|
| 1   | 0   | 1 : EAN-13 読取り有り<br>0 : EAN-13 読取り無し                 | 1     | CCD,Laser |
| 2   | 7   | 1 : EAN-13 Addon 2 読取り有り<br>0 : EAN-13 Addon 2 読取り無し | 0     | CCD,Laser |
| 2   | 6   | 1 : EAN-13 Addon 5 読取り有り<br>0 : EAN-13 Addon 5 読取り無し | 0     | CCD,Laser |
| 10  | 7   | 1 : ISBN 変換有り<br>0 : ISBN 変換無し                       | 0     | CCD,Laser |
| 10  | 6   | 1 : ISSN 変換有り<br>0 : ISSN 変換無し                       | 0     | CCD,Laser |
| 10  | 2   | 1 : EAN-13 チェックディジット送信有り<br>0 : EAN-13 チェックディジット送信無し | 1     | CCD,Laser |

#### EAN-13 → ISBN 変換

978、979 で始まる EAN-13 を ISBN に変換するかどうかを設定します。

#### EAN-13 → ISSN 変換

977 で始まる EAN-13 を ISSN に変換するかどうかを設定します。

#### チェックディジット送信

読取りデータにチェックディジットを含めるかどうかを設定します。

### GTIN

| バイト | ビット | 説明                               | デフォルト | 対象スキャナ    |
|-----|-----|----------------------------------|-------|-----------|
| 11  | 5   | 1 : GTIN 読取り有り<br>0 : GTIN 読取り無し | 0     | CCD,Laser |

### UPC-A

| バイト | ビット | 説明                                                 | デフォルト | 対象スキャナ    |
|-----|-----|----------------------------------------------------|-------|-----------|
| 9   | 0   | 1 : UPC-A → EAN-13 変換有り<br>0 : UPC-A → EAN-13 変換無し | 1     | CCD,Laser |
| 10  | 4   | 1 : UPC-A チェックディジット送信有り<br>0 : UPC-A チェックディジット送信無し | 1     | CCD,Laser |

|    |   |                                                |   |           |
|----|---|------------------------------------------------|---|-----------|
| 10 | 0 | 1 : UPC-A システムバース送信有り<br>0 : UPC-A システムバース送信無し | 1 | CCD,Laser |
|----|---|------------------------------------------------|---|-----------|

#### UPC-A → EAN-13 変換

読取った UPC-A を EAN-13 に拡張するかどうかを設定します。有りの場合、以降の処理は EAN-13 の設定に従います。

#### チェックビット送信

読取りデータにチェックビットを含めるかどうかを設定します。

| UPC-E |     |                                                    |       |           |
|-------|-----|----------------------------------------------------|-------|-----------|
| バイト   | ビット | 説明                                                 | デフォルト | 対象スキャナ    |
| 1     | 6   | 1 : UPC-E 読取り有り<br>0 : UPC-E 読取り無し                 | 1     | CCD,Laser |
| 1     | 5   | 1 : UPC-E Addon 2 読取り有り<br>0 : UPC-E Addon 2 読取り無し | 0     | CCD,Laser |
| 1     | 4   | 1 : UPC-E Addon 5 読取り有り<br>0 : UPC-E Addon 5 読取り無し | 0     | CCD,Laser |
| 4     | 6   | 1 : UPC-E1 & UPC-E0 有効<br>0 : UPC-E0 のみ            | 0     | CCD,Laser |
| 9     | 1   | 1 : UPC-E → UPC-A 変換有り<br>0 : UPC-E → UPC-A 変換無し   | 0     | CCD,Laser |
| 10    | 5   | 1 : UPC-E チェックビット送信有り<br>0 : UPC-E チェックビット送信無し     | 1     | CCD,Laser |
| 10    | 1   | 1 : UPC-E システムバース送信有り<br>0 : UPC-E システムバース送信無し     | 0     | CCD,Laser |
| 11    | 1   | 1 : UPC-E1 トリプルチェック有り<br>0 : UPC-E1 トリプルチェック無し     | 0     | CCD,Laser |

#### UPC-E → UPC-A 変換

読取った UPC-E を UPC-A に拡張するかどうかを設定します。有りの場合、以降の処理は EAN-13 の設定に従います。

#### チェックビット送信

読取りデータにチェックビットを含めるかどうかを設定します。

#### システムバース送信

読取りデータにシステムバースを含めるかどうかを設定します。

#### UPC-E1 トリプルチェック

UPC-E1 の読取り照合を適用するかどうかを設定します。有りの場合、有効な読取りとするのに、同じ UPC-E1 を3回読取ります。

➤ バースコードが汚れているなど読取りにくい場合に便利です。

## 2 次元、(エクストラ)ロング レンジ レーザ ーのキャパ

### CODABAR

| バ イ ト | ビ ッ ト | 説明                                                                           | デ フォルト | 対象キャパ                     |
|-------|-------|------------------------------------------------------------------------------|--------|---------------------------|
| 0     | 1     | 1 : Codabar(NW7) 読取り有り<br>0 : Codabar(NW7) 読取り無し                             | 1      | 2D,<br>(Extra) Long range |
| 7     | 3     | 1 : Codabar(NW7) スタート/ストップ キャラクタ送信有り<br>0 : Codabar(NW7) スタート/ストップ キャラクタ送信無し | 0      | 2D,<br>(Extra) Long range |
| 34    | 7     | 1 : Codabar(NW7)桁数チェック 最大桁数/最小桁数<br>0 : Codabar(NW7)桁数チェック 固定桁数              | 0      | 2D,<br>(Extra) Long range |
| 34    | 6 - 0 | Codabar(NW7)最大桁数/固定桁数 1                                                      | 0      | 2D,<br>(Extra) Long range |
| 35    | 7 - 0 | Codabar(NW7)最小桁数/固定桁数 2<br>桁数 1 > 桁数 2 となるよう設定してください                         | 0      | 2D,<br>(Extra) Long range |

#### スタート/ストップ キャラクタ送信

送信するデータにスタート/ストップ 文字を含めるかどうかを設定します。

#### 桁数設定

バーコード の読取りを固定桁数または最大桁数/最少桁数で制限することができます。バーコード の長はチェック デイ ッ トを含む文字数(人が読める文字)に関係します。

➤ “固定桁数”を選択した場合、最大 2 つの固定桁数を設定できます。

➤ “最大桁数/最小桁数”を選択した場合、最大桁数と最小桁数を設定する必要があります。指定された範囲の長さのバーコード のみを受け付けます。

※ 固定桁数の場合、桁数 1 は桁数 2 より大きい値に設定しなければなりません。でなければ、最大桁数/最少桁数として動作します。桁数 1 が最小値、桁数 2 が最大値となります。どちらかの桁数指定でも、両方の値が 0 に設定されている場合、長さ制限がないことを意味します。

### CODE 2 of 5

#### INDUSTRIAL 25

| バ イ ト | ビ ッ ト | 説明                                                                  | デ フォルト | 対象キャパ                     |
|-------|-------|---------------------------------------------------------------------|--------|---------------------------|
| 26    | 7     | 1 : Industrial 25 読取り有り<br>0 : Industrial 25 読取り無し                  | 1      | 2D,<br>(Extra) Long range |
| 32    | 7     | 1 : Industrial 25 桁数チェック 最大桁数/最小桁数<br>0 : Industrial 25 桁数チェック 固定桁数 | 0      | 2D,<br>(Extra) Long range |
| 32    | 6 - 0 | Industrial 25 最大桁数/固定桁数 1                                           | 0      | 2D,<br>(Extra) Long range |
| 33    | 7 - 0 | Industrial 25 最小桁数/固定桁数 2<br>桁数 1 > 桁数 2 となるよう設定してください              | 0      | 2D,<br>(Extra) Long range |

#### 桁数設定

2 of 5 バーコード の脆弱な構造のため、ショートスキャンエラーを起こす可能性があります。ショートスキャンエラーを防ぐために、正しいバーコード の桁数の範囲を設定します。CodaBar を参照してください。

#### INTERLEAVED 25

| バ イ ト | ビ ッ ト | 説明                                                                       | デ フォルト | 対象キャパ                     |
|-------|-------|--------------------------------------------------------------------------|--------|---------------------------|
| 0     | 3     | 1 : Interlieved 25 読取り有り<br>0 : Interlieved 25 読取り無し                     | 1      | 2D,<br>(Extra) Long range |
| 5     | 0     | 1 : Interlieved 25 チェック デイ ッ ト送信有り<br>0 : Interlieved 25 チェック デイ ッ ト送信無し | 0      | 2D,<br>(Extra) Long range |
| 14    | 7     | 1 : Interlieved 25 桁数チェック 最大桁数/最小桁数<br>0 : Interlieved 25 桁数チェック 固定桁数    | 0      | 2D,<br>(Extra) Long range |
| 14    | 6 - 0 | Interlieved 25 最大桁数/固定桁数 1                                               | 0      | 2D,<br>(Extra) Long range |
| 15    | 7 - 0 | Interlieved 25 最小桁数/固定桁数 2<br>桁数 1 > 桁数 2 となるよう設定してください                  | 0      | 2D,<br>(Extra) Long range |

|    |       |                                                                                                                                     |    |                           |
|----|-------|-------------------------------------------------------------------------------------------------------------------------------------|----|---------------------------|
| 37 | 7 - 6 | 00 : Interleaved 25 チェック イグジット検査無し<br>01 : Interleaved 25 USS チェック イグジット検査有り<br>10 : Interleaved 25 OPCC チェック イグジット検査有り<br>11 : 未定義 | 00 | 2D,<br>(Extra) Long range |
| 39 | 4     | 1 : Interleaved 25 → EAN13 変換有り<br>0 : Interleaved 25 → EAN13 変換無し                                                                  | 0  | 2D,<br>(Extra) Long range |

### 桁数設定

2 of 5 バージョンの脆弱な構造のため、ショートスキャンを起こす可能性があります。ショートスキャンを防ぐために、正しいバージョンの桁数の範囲を設定します。Codabar を参照してください。

### チェック イグジット検査

読取り時にチェック イグジット検査を行うかどうかを設定します。

➤ チェック イグジットに誤りがあるバージョンは受け付けません。

### EAN-13 変換

読取った 14 文字の Interleaved 25 を EAN-13 に拡張するかどうかを設定します。有りの場合、以降の処理は EAN-13 の設定に従います。

➤ Interleaved 25 が先頭で、有効な EAN-13 チェック イグジットを持っている必要があります。

※ EAN-13 変換はチェック イグジット検査無しでない限り有効にすることはできません。

## CODE39

| ビット | ビット   | 説明                                                        | デフォルト | 対象スケール                    |
|-----|-------|-----------------------------------------------------------|-------|---------------------------|
| 0   | 7     | 1 : Code 39 読取り有り<br>0 : Code 39 読取り無し                    | 1     | 2D,<br>(Extra) Long range |
| 0   | 6     | 1 : Code 32(Italian Pharmacode)読取り有り<br>0 : Code 32 読取り無し | 0     | 2D,<br>(Extra) Long range |
| 5   | 6     | 1 : Code 39 チェック イグジット検査有り<br>0 : Code 39 チェック イグジット検査無し  | 0     | 2D,<br>(Extra) Long range |
| 5   | 5     | 1 : Code 39 チェック イグジット送信有り<br>0 : Code 39 チェック イグジット送信無し  | 0     | 2D,<br>(Extra) Long range |
| 5   | 4     | 1 : Code 39 フルアスキー<br>0 : Code 39 標準                      | 0     | 2D,<br>(Extra) Long range |
| 23  | 7     | 1 : Code 39 桁数チェック 最大桁数/最小桁数<br>0 : Code 39 桁数チェック 固定桁数   | 0     | 2D,<br>(Extra) Long range |
| 23  | 6 - 0 | Code 39 最大桁数/固定桁数 1                                       | 0     | 2D,<br>(Extra) Long range |
| 24  | 7 - 0 | Code 39 最小桁数/固定桁数 2<br>桁数 1 > 桁数 2 となるよう設定してください          | 0     | 2D,<br>(Extra) Long range |
| 26  | 5     | 1 : Trioptic Code 39 読取り有り<br>0 : Trioptic Code 39 読取り無し  | 0     | 2D,<br>(Extra) Long range |

### チェック イグジット検査

読取り時にチェック イグジット検査を行うかどうかを設定します。

➤ チェック イグジットに誤りがあるバージョンは受け付けません。

※チェック イグジット検査は、チェック イグジットを送信したくないときに、チェック イグジットを省くことができるように有効にする必要があります。

### チェック イグジット送信

読取りデータにチェック イグジットを含めるかどうかを設定します。

### Code 39 フルアスキー

すべての英数字、および特殊文字を含めた Code 39 フルアスキーを出力するかどうかを設定します。

### 桁数設定

Codabar を参照してください。

## CODE 93

| バイト | ビット   | 説明                                                      | デフォルト | 対象スキャナ                    |
|-----|-------|---------------------------------------------------------|-------|---------------------------|
| 0   | 0     | 1 : Code 93 読取り有り<br>0 : Code 93 読取り無し                  | 1     | 2D,<br>(Extra) Long range |
| 28  | 7     | 1 : Code 93 桁数チェック 最大桁数/最小桁数<br>0 : Code 93 桁数チェック 固定桁数 | 0     | 2D,<br>(Extra) Long range |
| 28  | 6 - 0 | Code 93 最大桁数/固定桁数 1                                     | 0     | 2D,<br>(Extra) Long range |
| 29  | 7 - 0 | Code 93 最小桁数/固定桁数 2<br>桁数 1 > 桁数 2 となるよう設定してください        | 0     | 2D,<br>(Extra) Long range |

### 桁数設定

Codabar を参照してください。

## CODE 128

| CODE 128 |     |                                          |       |                           |
|----------|-----|------------------------------------------|-------|---------------------------|
| バイト      | ビット | 説明                                       | デフォルト | 対象スキャナ                    |
| 1        | 7   | 1 : Code 128 読取り有り<br>0 : Code 128 読取り無し | 1     | 2D,<br>(Extra) Long range |

## ISBT 128

| バイト | ビット | 説明                                       | デフォルト | 対象スキャナ                    |
|-----|-----|------------------------------------------|-------|---------------------------|
| 26  | 6   | 1 : ISBT 128 読取り有り<br>0 : ISBT 128 読取り無し | 1     | 2D,<br>(Extra) Long range |

## UCC/EAN-128

| バイト | ビット | 説明                                             | デフォルト | 対象スキャナ                    |
|-----|-----|------------------------------------------------|-------|---------------------------|
| 26  | 4   | 1 : UCC/EAN-128 読取り有り<br>0 : UCC/EAN-128 読取り無し | 1     | 2D,<br>(Extra) Long range |

## MSI

| バイト | ビット   | 説明                                                                                           | デフォルト  | 対象スキャナ                    |
|-----|-------|----------------------------------------------------------------------------------------------|--------|---------------------------|
| 2   | 5     | 1 : MSI 読取り有り<br>0 : MSI 読取り無し                                                               | 1      | 2D,<br>(Extra) Long range |
| 9   | 7 - 6 | MSI チェック イグジットタイプ<br>00 : シングル エッジ 10<br>01 : ダブル エッジ 10<br>10 : エッジ 11 & 10<br>11 : 未定義     | 01     | 2D,<br>(Extra) Long range |
| 9   | 5 - 4 | MSI チェック イグジット送信<br>00 : ラストチェック イグジット送信無し<br>01 : チェック イグジット両方送信有り<br>10 : チェック イグジット両方送信無し | 00     | 2D,<br>(Extra) Long range |
| 18  | 7     | 1 : MSI 桁数チェック 最大桁数/最小桁数<br>0 : MSI 桁数チェック 固定桁数                                              | 1      | 2D,<br>(Extra) Long range |
| 18  | 6 - 0 | MSI 最大桁数/固定桁数 1                                                                              | Max.31 | 2D,<br>(Extra) Long range |
| 19  | 7 - 0 | MSI 最小桁数/固定桁数 2                                                                              | Min.3  | 2D,<br>(Extra) Long range |

### チェック イグジット検査

読取り時にチェック イグジット検査を行うかどうか設定します。

➤ チェック イグジットに誤りがあるバーコード は受け付けません。

### チェック イグジット送信

読取りデータにチェック イグジットを含めるかどうかを設定します。



## 桁数設定

バーコードの脆弱な構造のため、ショートスキャンエラーを起こす可能性があります。ショートスキャンエラーを防ぐために、正しいバーコードの桁数の範囲を設定します。Codabar を参照してください。

### RSS

| バイト | ビット | 説明                                               | デフォルト | 対象スキャン                    |
|-----|-----|--------------------------------------------------|-------|---------------------------|
| 26  | 3   | 1 : RSS → UPC/EAN 変換有り<br>0 : RSS → UPC/EAN 変換無し | 0     | 2D,<br>(Extra) Long range |
| 26  | 2   | 1 : RSS Expanded 読取り有り<br>0 : RSS Expanded 読取り無し | 1     | 2D,<br>(Extra) Long range |
| 26  | 1   | 1 : RSS Limited 読取り有り<br>0 : RSS Limited 読取り無し   | 1     | 2D,<br>(Extra) Long range |
| 26  | 0   | 1 : RSS-14 読取り有り<br>0 : RSS-14 読取り無し             | 1     | 2D,<br>(Extra) Long range |

### RSS → UPC/EAN 変換

RSS を UPC/EAN に変換するかどうかを設定します。有りの場合、

- (1) 先頭の "010" はバーコードから除去され、"0" が最初の数字としてインコードされます。RSS を EAN-13 に変換します。
- (2) ゼロ 6 つを除く 2 つ以上の 0 で始まるバーコードは、この設定では先頭の "0010" を除去し、UPC-A と認識します。

システムキャラクタと国コードを送信する UPC-A プリアンブル設定は変換されたバーコードに適用されます。

システムキャラクタもチェックデジットも除去されることに注意してください。

➤ コンボジットコードの一部としてとしてデコードされない RSS-14 と RSS Limited にのみ適用されます。

### UPC / EAN

UPC/EAN は以下のバーコードに適用なし、適用 2、適用 5 を含みます。

- UPC-E0
- UPC-E1
- UPC-A
- EAN-8
- EAN-13
- Bookland EAN (ISBN)

UPC / EAN に属するすべてのメタデータについて、25 バイトのビット 0 は、適用なし、適用 2、適用 5 の共通設定となっています。それ以外のパラメータの設定は次の通りです。

| バイト | ビット | 説明                                                             | デフォルト | 対象スキャン                    |
|-----|-----|----------------------------------------------------------------|-------|---------------------------|
| 9   | 1   | 1 : UPC-E0 → UPC-A 変換有り<br>0 : UPC-E0 → UPC-A 変換無し             | 0     | 2D,<br>(Extra) Long range |
| 10  | 5   | 1 : UPC-E0 チェックデジット送信有り<br>0 : UPC-E0 チェックデジット送信無し             | 1     | 2D,<br>(Extra) Long range |
| 10  | 4   | 1 : UPC-A チェックデジット送信有り<br>0 : UPC-A チェックデジット送信無し               | 1     | 2D,<br>(Extra) Long range |
| 10  | 1   | 1 : UPC-E0 システムナビゲーション送信有り<br>0 : UPC-E0 システムナビゲーション送信無し       | 1     | 2D,<br>(Extra) Long range |
| 10  | 0   | 1 : UPC-A システムナビゲーション送信有り<br>0 : UPC-A システムナビゲーション送信無し         | 1     | 2D,<br>(Extra) Long range |
| 11  | 7   | 1 : EAN-8 → EAN-13 変換有り<br>0 : EAN-8 → EAN-13 変換無し             | 1     | 2D,<br>(Extra) Long range |
| 25  | 7   | 1 : UPC-E1 システムナビゲーション送信有り<br>0 : UPC-E1 システムナビゲーション送信無し       | 0     | 2D,<br>(Extra) Long range |
| 25  | 6   | 1 : UPC-E1 チェックデジット送信有り<br>0 : UPC-E1 チェックデジット送信無し             | 0     | 2D,<br>(Extra) Long range |
| 25  | 3   | 1 : UPC-E1 → UPC-A 変換有り<br>0 : UPC-E1 → UPC-A 変換無し             | 0     | 2D,<br>(Extra) Long range |
| 39  | 7   | 1 : UPC-A システムナビゲーション&国コード有効<br>0 : UPC-A システムナビゲーション&国コード無効   | 1     | 2D,<br>(Extra) Long range |
| 39  | 6   | 1 : UPC-E システムナビゲーション&国コード有効<br>0 : UPC-E システムナビゲーション&国コード無効   | 1     | 2D,<br>(Extra) Long range |
| 39  | 5   | 1 : UPC-E1 システムナビゲーション&国コード有効<br>0 : UPC-E1 システムナビゲーション&国コード無効 | 1     | 2D,<br>(Extra) Long range |

## UPC-E0/ UPC-E1 → UPC-A 変換

読取った UPC-E0/ UPC-E1 を UPC-A に拡張するかどうかを設定します。有りの場合、以降の処理は EAN-13 の設定に従います。

## EAN-8 → EAN-13 変換

読取った EAN-8 を EAN-13 に拡張するかどうかを設定します。有りの場合、以降の処理は EAN-13 の設定に従います。

## チェックビット送信

読取りデータにチェックビットを含めるかどうかを設定します。

## システムバース送信

読取りデータにシステムバースを含めるかどうかを設定します。

## UCC COUPON CODE

| バイト | ビット | 説明                                           | デフォルト | 対象スキャナ                    |
|-----|-----|----------------------------------------------|-------|---------------------------|
| 42  | 3   | 1 : UCC Coupon 読取り有り<br>0 : UCC Coupon 読取り無し | 0     | 2D,<br>(Extra) Long range |

## 共通設定

- 共通設定の 25 バイト-ビット 0 に 1 がセットされると、表 1 のパラメータを個別に設定できます。
- 共通設定の 25 バイト-ビット 0 に 0 がセットされると、表 2 のいずれかのビットに 1 が設定されると、UPC/EAN のアドカ 2 と 5 のみが読取り可となります。(アドカなしは読取り不可)
- 表 2 のすべてのビットに 0 が設定されると、表 1 で読取り可となっているバーコードのアドカ無しのみが読取り可となります。

| 設定           | 表 1 のバイト-ビット | 表 2 のバイト-ビット | 結果       |          |
|--------------|--------------|--------------|----------|----------|
| 25 バイト-ビット 0 | 表 1 のバイト-ビット | 表 2 のバイト-ビット | アドカ無し    | アドカ有り    |
| =1           | =1           | N/A          | 読取り可     | 読取り可     |
| =1           | =0           | N/A          | 読取り不可    | 読取り不可    |
| =0           | N/A          | いずれか 1       | 読取り不可(*) | 読取り可(*)  |
| =0           | =1           | すべて 0        | 読取り可     | 読取り不可(*) |
| =0           | =0           | すべて 0        | 読取り不可    | 読取り不可(*) |

※ (\*)は UPC/EAN のすべてのバーコードで発生します。

| 表1  |     |                                                                                   |       |                           |
|-----|-----|-----------------------------------------------------------------------------------|-------|---------------------------|
| バイト | ビット | 説明                                                                                | デフォルト | 対象スキャナ                    |
| 1   | 6   | 1 : UPC-E0 読取り有り<br>0 : UPC-E0 読取り無し                                              | 1     | 2D,<br>(Extra) Long range |
| 1   | 3   | 1 : EAN-8 読取り有り<br>0 : EAN-8 読取り無し                                                | 1     | 2D,<br>(Extra) Long range |
| 1   | 0   | 1 : EAN-13 読取り有り<br>0 : EAN-13 読取り無し                                              | 1     | 2D,<br>(Extra) Long range |
| 25  | 1   | 1 : Bookland EAN 読取り有り<br>(バイト 1-ビット 0 が ON である必要があります)<br>0 : Bookland EAN 読取り無し | 0     | 2D,<br>(Extra) Long range |
| 27  | 7   | 1 : UPC-A 読取り有り<br>0 : UPC-A 読取り無し                                                | 1     | 2D,<br>(Extra) Long range |
| 27  | 5   | 1 : UPC-E1 読取り有り<br>0 : UPC-E1 読取り無し                                              | 0     | 2D,<br>(Extra) Long range |

- バイト 25-ビット 0 に 1 がセットされている場合、アドカ無し、アドカ 2、アドカ 5 が読取り可となります。
- バイト 25-ビット 0 に 0 がセットされている(表 2 にあるすべてのバイト-ビットも 0)場合、アドカ無しのみが読取り可となります。

表2

| ビット | ビット              | 説明                                                                                                          | デフォルト | 対象スケジュー                   |
|-----|------------------|-------------------------------------------------------------------------------------------------------------|-------|---------------------------|
| 1   | 5 or 4 or 3 or 2 | 1: UPC と EAN で Addon 2 と 5 のみ読取り可<br>(いずれかのビットが ON)<br>0: UPC と EAN で Addon 2 と 5 のみ読取り不可<br>(すべてのビットが OFF) | 0     | 2D,<br>(Extra) Long range |
| 2   | 7 - 6            |                                                                                                             | 0     | 2D,<br>(Extra) Long range |
| 27  | 6 or 4           |                                                                                                             | 0     | 2D,<br>(Extra) Long range |

## CODE 11

| ビット | ビット   | 説明                                                                  | デフォルト | 対象スケジュー                 |
|-----|-------|---------------------------------------------------------------------|-------|-------------------------|
| 25  | 2     | 1: Code 11 読取り有り<br>0: Code 11 読取り無し                                | 1     | 2D,<br>8300, 8700-LR のみ |
| 30  | 7     | 1: Code 11 桁数チェック 最大桁数/最小桁数<br>0: Code 11 桁数チェック 固定桁数               | 0     | 2D,<br>8300, 8700-LR のみ |
| 30  | 6 - 0 | Code 11 最大桁数/固定桁数 1                                                 | 0     | 2D,<br>8300, 8700-LR のみ |
| 31  | 7 - 0 | Code 11 最小桁数/固定桁数 2<br>桁数 1 > 桁数 2 となるよう設定してください                    | 0     | 2D,<br>8300, 8700-LR のみ |
| 42  | 1 - 0 | Code 11 チェックディジット検査<br>00: 無効<br>01: 1 チェックディジット<br>10: 2 チェックディジット | 00    | 2D,<br>8300, 8700-LR のみ |

## 桁数設定

バーコードの読取りを固定桁数または最大桁数/最少桁数で制限することができます。バーコードの長はチェックディジットを含む文字数(人が読める文字)に関係します。

➤ “固定桁数”を選択した場合、最大 2 つの固定桁数を設定できます。

➤ “最大桁数/最少桁数”を選択した場合、最大桁数と最小桁数を設定する必要があります。指定された範囲の長さのバーコードのみを受け付けます。

※ 固定桁数の場合、桁数 1 は桁数 2 より大きい値に設定しなければなりません。でなければ、最大桁数/最少桁数として動作します。桁数 1 が最小値、桁数 2 が最大値となります。どちらかの桁数指定でも、両方の値が 0 に設定されている場合、長さ制限がないことを意味します。

## 2 次元スキャン限定

前述のバーコードに加えて、2次元スキャンでは以下のバーコードもサポートしています。

### 1 次元バーコード

#### CHINESE 25

| ビット | ビット | 説明                                           | デフォルト | 対象スキャン                 |
|-----|-----|----------------------------------------------|-------|------------------------|
| 42  | 2   | 1 : Chinese 25 読取り有り<br>0 : Chinese 25 読取り無し | 0     | 8200, 8400,<br>8700-2D |

#### MATRIX 25

| ビット | ビット   | 説明                                                          | デフォルト | 対象スキャン                 |
|-----|-------|-------------------------------------------------------------|-------|------------------------|
| 0   | 2     | 1 : Matrix 25 読取り有り<br>0 : Matrix 25 読取り無し                  | 0     | 8200,8400,<br>8700-2D  |
| 6   | 5     | 1 : Matrix 25 チェックディジット検査有り<br>0 : Matrix 25 チェックディジット検査無し  | 0     | 8200, 8400,<br>8700-2D |
| 6   | 4     | 1 : Matrix 25 チェックディジット送信有り<br>0 : Matrix 25 チェックディジット送信無し  | 0     | 8200, 8400,<br>8700-2D |
| 16  | 7     | 1 : Matrix 25 桁数チェック 最大桁数/最小桁数<br>0 : Matrix 25 桁数チェック 固定桁数 | 1     | 8200, 8400<br>8700-2D  |
| 16  | 6 - 0 | Matrix 25 最大桁数/固定桁数 1                                       | 0     | 8200, 8400<br>8700-2D  |
| 17  | 7 - 0 | Matrix 25 最小桁数/固定桁数 2<br>桁数 1 > 桁数 2 となるよう設定してください          | 0     | 8200, 8400<br>8700-2D  |

#### UPC/EAN – BOOKLAND ISBN フォーマット

| ビット | ビット | 説明                                                                                               | デフォルト | 対象スキャン                 |
|-----|-----|--------------------------------------------------------------------------------------------------|-------|------------------------|
| 41  | 6   | UPC/EAN Bookland ISBN フォーマット<br>1 : UPC/EAN – Bookland ISBN 13<br>0 : UPC/EAN – Bookland ISBN 10 | 0     | 8200, 8400,<br>8700-2D |

#### 1次元白黒反転バーコード

| ビット | ビット   | 説明                                                                                         | デフォルト | 対象スキャン                 |
|-----|-------|--------------------------------------------------------------------------------------------|-------|------------------------|
| 40  | 2 - 1 | 1 次元白黒反転バーコード<br>00 : 通常 1 次元バーコード のみ読取り<br>01 : 反転 1 次元バーコード のみ読取り<br>10 : 通常/反転バーコード 読取り | 00    | 8200, 8400,<br>8700-2D |

#### POSTAL CODE

| ビット | ビット | 説明                                                         | デフォルト | 対象スキャン |
|-----|-----|------------------------------------------------------------|-------|--------|
| 36  | 7   | 1 : US Postal チェックディジット送信有り<br>0 : US Postal チェックディジット送信無し | 1     | 2D     |
| 36  | 3   | 1 : US Planet 読取り有り<br>0 : US Planet 読取り無し                 | 1     | 2D     |
| 36  | 2   | 1 : US Postnet 読取り有り<br>0 : US Postnet 読取り無し               | 1     | 2D     |
| 37  | 4   | 1 : JapanPostal 読取り有り<br>0 : JapanPostal 読取り無し             | 1     | 2D     |
| 37  | 3   | 1 : Australian Postal 読取り有り<br>0 : Australian Postal 読取り無し | 1     | 2D     |
| 37  | 2   | 1 : Dutch Postal 読取り有り<br>0 : Dutch Postal 読取り無し           | 1     | 2D     |
| 37  | 1   | 1 : UK Postal チェックディジット有効<br>0 : UK Postal チェックディジット無効     | 1     | 2D     |
| 37  | 0   | 1 : UK Postal 読取り有り<br>0 : UK Postal 読取り無し                 | 1     | 2D     |

|    |   |                                                                                                      |   |                        |
|----|---|------------------------------------------------------------------------------------------------------|---|------------------------|
| 39 | 0 | 1 : USPS 4CB / One Code / Intelligent Mail 読取り有り<br>0 : USPS 4CB / One Code / Intelligent Mail 読取り無し | 0 | 8200, 8400,<br>8700-2D |
| 41 | 7 | 1 : UPU FICS 読取り有り<br>0 : UPU FICS 読取り無し                                                             | 0 | 8200, 8400,<br>8700-2D |

## チェックデジット送信

読取りデータにチェックデジットを含めるかどうかを設定します。

## コンボジットコード

| CC-A/B/C |     |                                                    |       |        |
|----------|-----|----------------------------------------------------|-------|--------|
| バイト      | ビット | 説明                                                 | デフォルト | 対象スキャン |
| 27       | 1   | 1 : コンボジット CC-A/B 読取り有り<br>0 : コンボジット CC-A/B 読取り無し | 0     | 2D     |
| 27       | 0   | 1 : コンボジット CC-C 読取り有り<br>0 : コンボジット CC-C 読取り無し     | 0     | 2D     |

| TLC-39 |     |                                                                |       |        |
|--------|-----|----------------------------------------------------------------|-------|--------|
| バイト    | ビット | 説明                                                             | デフォルト | 対象スキャン |
| 25     | 4   | 1 : TCIF Linked Code 39 読取り有り<br>0 : TCIF Linked Code 39 読取り無し | 1     | 2D     |

| UPC コンボジット |       |                                                                      |       |        |
|------------|-------|----------------------------------------------------------------------|-------|--------|
| バイト        | ビット   | 説明                                                                   | デフォルト | 対象スキャン |
| 27         | 3 - 2 | 00 : UPC リンクしない<br>01 : UPC 常にリンク<br>10 : UPC コンボジット自動識別<br>11 : 未定義 | 01    | 2D     |

## UPC コンボジットモード選択

UPC は送信時に 2 次元バーコードと 1 つのバーコードとしてリンクすることができます。次の 3 つのオプションがあります。

(1) UPC リンクしない

2 次元バーコードの有無に関係なく UPC を読取ります。

(2) UPC 常にリンク

UPC と 2 次元バーコード部の両方を読取ります。2 次元バーコード部がない場合、UPC は読取られません。

(3) UPC コンボジット自動識別

2 次元バーコード部があれば、UPC と両方を読取ります。

※ “UPC 常にリンク”が設定されている場合は、CC-A/B または CC-C のどちらかを有効にする必要があります。でなければ、UPC があっても読取られません。

| UPC/EAN コンボジットコードの GS1-128 インターゾネート |     |                                                                                        |       |        |
|-------------------------------------|-----|----------------------------------------------------------------------------------------|-------|--------|
| バイト                                 | ビット | 説明                                                                                     | デフォルト | 対象スキャン |
| 25                                  | 5   | 1 : UPC/EAN コンボジットコードの GS1-128 インターゾネート有効<br>0 : UPC/EAN コンボジットコードの GS1-128 インターゾネート無効 | 0     | 2D     |

## 2次元バーコード

### MAXICODE, DATA MATRIX, QR CODE

| バイト | ビット | 説明                                             | デフォルト | 対象スケッチ                 |
|-----|-----|------------------------------------------------|-------|------------------------|
| 36  | 6   | 1 : Maxicode 読取り有り<br>0 : Maxicode 読取り無し       | 1     | 2D                     |
| 36  | 5   | 1 : Data Matrix 読取り有り<br>0 : Data Matrix 読取り無し | 1     | 2D                     |
| 36  | 4   | 1 : QR Code 読取り有り<br>0 : QR Code 読取り無し         | 1     | 2D                     |
| 42  | 7   | 1 : MicroQR 読取り有り<br>0 : MicroQR 読取り無し         | 1     | 8200, 8400,<br>8700-2D |
| 42  | 6   | 1 : Aztec 読取り有り<br>0 : Aztec 読取り無し             | 1     | 8200, 8400,<br>8700-2D |

### 2次元白黒反転バーコード, 左右反転バーコード

| バイト | ビット   | 説明                                                                                                       | デフォルト | 対象スケッチ                 |
|-----|-------|----------------------------------------------------------------------------------------------------------|-------|------------------------|
| 41  | 5 - 4 | Data Matrix 白黒反転<br>00 : 通常 Data Matrix のみ読取り<br>01 : 反転 Data Matrix のみ読取り<br>10 : 通常/反転 Data Matrix 読取り | 00    | 8200, 8400,<br>8700-2D |
| 41  | 3 - 2 | Data Matrix 左右反転<br>00 : 通常 Data Matrix のみ読取り<br>01 : 反転 Data Matrix のみ読取り<br>10 : 通常/反転 Data Matrix 読取り | 00    | 8200, 8400,<br>8700-2D |
| 41  | 1 - 0 | QR Code 白黒反転<br>00 : 通常 QR Code のみ読取り<br>01 : 反転 QR Code のみ読取り<br>10 : 通常/反転 QR Code 読取り                 | 00    | 8200, 8400,<br>8700-2D |
| 42  | 5 - 4 | Aztec 白黒反転<br>00 : 通常 Aztec のみ読取り<br>01 : 反転 Aztec のみ読取り<br>10 : 通常/反転 Aztec 読取り                         | 00    | 8200, 8400,<br>8700-2D |

### PDF417

| バイト | ビット   | 説明                                                                                | デフォルト | 対象スケッチ |
|-----|-------|-----------------------------------------------------------------------------------|-------|--------|
| 36  | 1     | 1 : MicroPDF417 読取り有り<br>0 : MicroPDF417 読取り無し                                    | 1     | 2D     |
| 36  | 0     | 1 : PDF417 読取り有り<br>0 : PDF417 読取り無し                                              | 1     | 2D     |
| 39  | 3 - 2 | MacroPDF 送信/変換モード<br>00 : パススルー<br>01 : すべてのシンボルをバッファリング / 完了時に送信<br>02 : 特定の指示なし | 00    | 2D     |
| 39  | 1     | 1 : MacroPDF イスケープ文字有効<br>0 : MacroPDF イスケープ文字無効                                  | 0     | 2D     |

### Macro PDF 送信/読取りモード

MacroPDF は、MacroPDF417 または MicroPDF417 と呼ばれる 1 つのファイルに複数の PDF バーコードを連結するための特別な機能です。

MacroPDF デコードの処理方法を設定します。

(1) すべてのシンボルをバッファリング / 完了時に送信

シークス全体がスキップされ、デコードされている場合にのみ MacroPDF シークス全体からすべてデコードされたデータを送信します。デコードされたデータが上限の 50 シンボルを超えた場合、シークス全体がスキップされませんされていないため送信されません。

➤ コントロールタグの送信を無効にする必要があります。

(2) 特定の指示なし

シークスにかかわらず、デコードされた MacroPDF からデータを送信します。

➤ コントロールハッグの送信を有効にする必要があります。

(3) パスル

すべての MacroPDF を送信し、デコードすると、何も処理しません。このモードではホスト側で MacroPDF シーケンスを測定、解析する必要があります。

### Macro PDF 이스케이프 문자

이스케이프 문자を送信するかどうかを設定します。送信ありの場合、特殊なデータ配列を含む伝送を処理できるシステムの 이스케이프 문자として 백슬래시 "\" を使用します。

➤ MacroPDF 伝送のデータ部分に影響を与えるグローバルな識別子(GLI)プロトコルに従って特殊なデータをフォーマットします。コントロールハッグは、常に GLI のフォーマットで送信されます。

## 付録3 スキャナパラメータ

ここでは、スキャナのパラメータについて説明します。

### スキャンモード

ScannerDesTbl 配列の 20 バイト目は処理に最も適したスキャンモードを設定するために使用されます。タイムアウトも参照してください。

| バイト | ビット   | 説明                                                                                                                                                                                     | デフォルト | 対象スキャナ                    |
|-----|-------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------|---------------------------|
| 20  | 7 - 4 | 読取りモード (リーダーポート 1)<br>0000: オートオフモード<br>0001: コンティニアスモード<br>0010: オートパワーオフモード<br>0011: アルターネイトモード<br>0100: モーメンタリモード<br>0101: リピートモード<br>0110: レーザーモード<br>0111: テストモード<br>1000: イミグモード | 0110  | CCD, Laser                |
| 20  | 7 - 4 | 読取りモード (リーダーポート 1)<br>1000: イミグモード<br>0111: テストモード<br>0110: レーザーモード<br>0011: アルターネイトモード<br>0001: コンティニアスモード<br>0000: オートオフモード<br>上記以外の値: レーザーモード                                       | 0110  | 2D,<br>(Extra) Long range |

➤ CCD、レーザーは 9 種類のモードをサポートしています。下記の比較表を参照してください。21 バイトは必要に応じてタイムアウトを設定します。

➤ (エクストラ)ロングレンジレーザーはレーザーモードとイミグモードのみをサポートしています。イミグモードの場合、1 度トリガキーを押すと読み取りビームを照射します。読み取りビームはタイムアウトが発生するかバースコード読取りのためにもう一度トリガキーを押すまで消えません。38 バイトは必要に応じてタイムアウトを設定します。

### 比較表

| スキャンモード     | スキャン開始 |                 |                |                 | スキャン終了       |                 |               |        |
|-------------|--------|-----------------|----------------|-----------------|--------------|-----------------|---------------|--------|
|             | 常時     | トリガキーを<br>1 回押す | トリガキーを<br>押し続け | トリガキーを<br>2 回押す | トリガキーを<br>放す | トリガキーを<br>1 回押す | バースコード<br>読取り | タイムアウト |
| コンティニアスモード  | ○      |                 |                |                 |              |                 |               |        |
| テストモード      | ○      |                 |                |                 |              |                 |               |        |
| リピートモード     | ○      |                 |                |                 |              |                 |               |        |
| モーメンタリモード   |        |                 | ○              |                 | ○            |                 |               |        |
| アルターネイトモード  |        | ○               |                |                 |              | ○               |               |        |
| イミグモード      |        |                 |                | ○               |              |                 | ○             | ○      |
| レーザーモード     |        |                 | ○              |                 | ○            |                 | ○             | ○      |
| オートオフモード    |        | ○               |                |                 |              |                 | ○             | ○      |
| オートパワーオフモード |        | ○               |                |                 |              |                 |               | ○      |

#### コンティニアスモード

常にバースコード読み取り状態となります。

#### テストモード

常にバースコード読み取り状態となります。同一のバースコードラベルでも連続して読み取るテスト用のモードです。通常のアプリケーションでは、使用しないでください。

#### リピートモード

常にバースコード読み取り状態となります。バースコードを読み取った後、1 秒以内にトリガボタンを押すと、直前に読み取ったバースコードデータを再入力します。この機能は、同一バースコードデータを続けて入力するようなアプリケーションで非常に有効です。



## メモタリモード

トリガボタンを押している間、バーコードの読み取りを行い、トリガボタンを離すと、読み取りを終了します。

## カルーネイトモード

トリガボタンを押すと、バーコードの読み取り状態となり、再度トリガボタンを押すと、読み取りを終了します。

## エイミングモード

1度目のトリガで読み取りビームを照射し、2度目のトリガでバーコードを読み取ります。的を絞ってから読み取りを行いたい場合に有効です。但し、2度目のトリガを一定時間内(デフォルト 1 秒)に押されなければ、読み取りビームはリセットされます。1度目のトリガと2度目のトリガ間隔は、システムグローバル変数 AIMING\_TIMEOUT に 5 ミリ秒単位で指定します。

## レーザーモード

トリガボタンを押すと、バーコードの読み取りを行い、バーコードの読み取りに成功するか、トリガボタンを離すと、読み取りを終了します。

## オートオフモード

トリガボタンを押すと、バーコードの読み取りを開始します。バーコードの読み取りに成功するか、読み取りタイムアウト時間で設定された時間が経過すると、読み取りを終了します。

## オートパワーオフモード

トリガボタンを押すと、バーコードの読み取りを開始します。バーコードの読み取りに成功すると、読み取りタイムアウトを再加算し、読み取りを続けます。読み取りタイムアウト時間で設定された時間が経過すると、読み取りを終了します。

## 読取り照合

読取りの精度を設定するのに使用するパラメータです。読取りの精度と速度を調節します。

| バイト | ビット   | 説明                                                                                                         | デフォルト | 対象スキャナ     |
|-----|-------|------------------------------------------------------------------------------------------------------------|-------|------------|
| 11  | 3 - 2 | 00: 読取り照合無し(リダンドポート 1)<br>01: 読取り照合 1 回(リダンドポート 1)<br>10: 読取り照合 2 回(リダンドポート 1)<br>11: 読取り照合 3 回(リダンドポート 1) | 00    | CCD, Laser |

- 照合無し  
“読取り照合無し”に設定している場合、1 度デコードに成功すると、読取りイベントが発生します。
- 照合 1 回、2 回 3 回  
“読取り照合 3 回”に設定している場合、4 回続けて同じバーコードの読取りが成功すると読取りイベントが発生します。読取り精度が向上する一方、読取り速度が遅くなります。

## タイムアウト

特定のスキャンモードの最大スキャン時間を制限するためのパラメータです。

| バイト | ビット   | 説明                                                         | デフォルト | 対象スキャナ                    |
|-----|-------|------------------------------------------------------------|-------|---------------------------|
| 21  | 7 - 0 | 読取りタイムアウト時間(秒単位, オートオフ, オートパワーオフモード用)<br>1~255、0 はタイムアウトなし | 3 Sec | CCD, Laser                |
| 38  | 7 - 0 | 読取りタイムアウト時間(秒単位, オートオフ, オートパワーオフモード用)<br>1~255、0 はタイムアウトなし | 3 Sec | 2D,<br>(Extra) Long range |

※ イーミングモードのイーミングタイムアウトの値は、グローバル変数 AIMING\_TIMEOUT で設定します。『2.1.3 システムグローバル変数』を参照してください。

## ユーザ設定

| バイト | ビット   | 説明                                                    | デフォルト | 対象スキャナ                 |
|-----|-------|-------------------------------------------------------|-------|------------------------|
| 40  | 7 - 6 | 00 : Far フォーカス<br>01 : Near フォーカス<br>10 : Smart フォーカス | 00    | 8500-2D                |
| 40  | 5     | 1 : 照準ハターソデコード 有効<br>0 : 照準ハターソデコード 無効                | 1     | 2D                     |
| 40  | 4     | 1 : イルミネーションデコード 有効<br>0 : イルミネーションデコード 無効            | 1     | 2D                     |
| 40  | 3     | 1 : ビックリストモード 有効<br>0 : ビックリストモード 無効                  | 0     | 8200, 8400,<br>8700-2D |

※ ビックリストモードでは、読取りビームの中央下に並んでいるバーコードのみをデコードするします。

|    |   |                                                         |   |                        |
|----|---|---------------------------------------------------------|---|------------------------|
| 40 | 0 | 1 : システムスlep中、リーダはスリープ状態<br>0 : システムスlep中、リーダは電源 OFF 状態 | 0 | 8200, 8400,<br>8700-2D |
|----|---|---------------------------------------------------------|---|------------------------|

※ システムスlep中、リーダは電源 OFF 状態にしておくと、バッテリーを節約できます。ただし、ビームから再開後、使用開始まで約 3 秒かかります。

Blank page