



データ圧縮ライブラリ

RELC DLL for Windows Mobile

custom01

ユーザーズマニュアル

01 版

富士通エレクトロニクス株式会社

はじめに

- 対象読者

本マニュアルは、C 言語の知識がある技術者の方を対象に記述しています。

- 商標

RELC DLL for Windows Mobile custom01 の著作権は富士通エレクトロニクス株式会社が保有しています。

Microsoft、Windows、Windows Mobile、Visual C++は、Microsoft Corporation の商標または登録商標です。

その他、会社名、製品名の固有名詞は各社の商号、商標または登録商標です。

本マニュアルに記載されている会社名、システム名、製品名等には必ずしも商標表示 (TM・®) を付記していません。

1. 本資料に記載した製品および製品の仕様につきましては、製品改善のため予告なしに変更することがあります。したがって、ご使用を検討の際には、本資料に記載の情報が最新のものであることを弊社技術担当、あるいは弊社営業担当にご確認ください。
2. 本資料に記載された情報・回路図は、当社製品の応用例として使用されており、実際に使用する機器への搭載を目的としたものではありません。また、これらの情報・回路図の使用に起因する第三者の特許権、その他権利侵害について、当社はその責任を負いません。
3. 本資料に記載された製品は、一般事務用、パーソナル用、家庭用、通常の産業用等の一般的用途を想定して設計・製造されているものであり、原子力施設における核反応制御、航空機自動飛行制御、航空交通管制、大量輸送システムにおける運行制御、生命維持のための医療用機器、兵器システムにおけるミサイル発射制御など、極めて高度な安全性が要求され、仮に当該安全性が確保されない場合、直接生命・身体に対する重大な危険性を伴う用途（以下「ハイセイフティ用途」という）に使用されるよう設計・製造されたものではございません。お客様は、当該ハイセイフティ用途に要する安全性を確保する措置を施すことなく、本製品を使用しないでください。ハイセイフティ用途に使用される場合は、当社担当営業までご相談ください。
4. 本資料に記載した内容を、弊社に無断で転載または複製することはご遠慮ください。
5. 本資料に記載された製品が、「外国為替および外国貿易法」に基づき規制されている貨物または技術に該当する場合には、本製品を輸出するに際して、同法に基づく許可が必要となります。
6. 各社が提供する開発ツール（コンパイラおよび付属のライブラリ）の不具合および仕様に起因する問題については弊社の保証対象外とします。

目次

1	概要	1
2	構成ファイル	1
3	ライブラリの作成環境	1
4	ライブラリを組み込む際の注意事項	2
4.1	ヘッダーファイルのインクルード	2
4.2	コンパイルスイッチの指定	2
4.3	ライブラリのリンク	2
5	ライブラリの情報取得関数	3
5.1	relc_GetVersion()	4
6	データ処理用関数	5
6.1	relc_InitEncode ()	6
6.2	relc_InitDecode ()	7
6.3	relc_Encode ()	8
6.4	relc_EncodeFlush ()	9
6.5	relc_Decode ()	10
6.6	API の使用方法	11
6.6.1	圧縮処理	11
6.6.2	復元処理	12
7	定義	13
7.1	定数	13
8	エラーリファレンス	14
9	PC 版評価プログラムについて	15

1 概要

本製品に含まれている Windows Mobile 用 DLL は、可逆圧縮方式(ロスレス圧縮方式)を使用してメモリに格納されたデータを圧縮するライブラリ製品です。

RELC は Rapid Embedded Lossless Compression の略称です。

2 構成ファイル

本機能を利用するためには、次のライブラリとヘッダーファイルを使用します。

relcl_wce6.dll	圧縮復元ライブラリ
relcl_wce6.lib	圧縮復元ライブラリ用インポートライブラリ
relc.h	API 用ヘッダーファイル
relctype.h	型定義ヘッダーファイル

製品パッケージの全体構成については、“ README.TXT ” を参照してください

3 ライブラリの作成環境

Microsoft社製の開発ツールであるVisual Studio 2005+Windows Mobile 6 Standard SDK を用いてコンパイルしています。ライブラリを組み込む際は、Visual Studio 2005+Windows Mobile 6 Standard SDKを使用してください。

4 ライブラリを組み込む際の注意事項

ライブラリを組み込む際の注意事項について説明します。

4.1 ヘッダーファイルのインクルード

ライブラリの API を使用するソースファイルでインクルードしてください。

4.2 コンパイルスイッチの指定

次のコンパイルスイッチを指定してください。

"_ANSI_"、"RELC_DYNAMIC"

4.3 ライブラリのリンク

スタティックライブラリのリンクと同様の指定方法で"relcl_wce6.lib"を指定します。

詳しくは、開発環境のマニュアルを参照してください。

5 ライブラリの情報取得関数

ライブラリの情報を取得する関数を提供しています。

次の関数を提供しています。

```
relc_GetVersion ( )
```

5.1 relc_GetVersion()

[機能]

ライブラリのバージョンを取得します。

[形式]

```
long relc_GetVersion(  
void  
)
```

[引数]

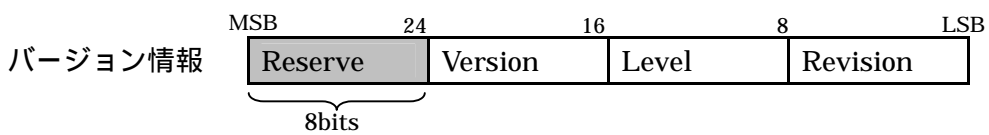
無し

[戻り値]

long == バージョン情報

[説明]

バージョンは、long(32bit)を1バイトずつに区切って上位からリザーブ、バージョン、レベル、リビジョンの順番で格納されています。



6 データ処理用関数

メモリからメモリへの圧縮および復元機能を提供しています。1回の呼び出しで処理できるサイズは規定されているため、それ以上のサイズを処理する場合は、バッファリングが必要となります。

次の5つの関数を提供しています。

```
relc_InitEncode ()
relc_Encode ()
relc_EncodeFlush ()
relc_InitDecode ()
relc_Decode ()
```

以降の説明では、区別のために圧縮前のデータをオリジナルデータ、圧縮後のデータを圧縮データと記述します。

圧縮処理とは、入力アドレスが示すメモリに格納されたオリジナルデータを読み出し、出力アドレスが示すメモリへ圧縮データを書き出す処理です。圧縮処理における関数の呼び出しは、次の順序で行います。

```
relc_InitEncode ()   relc_Encode ()   relc_EncodeFlush ()
```

復元処理とは、入力アドレスが示すメモリに格納された圧縮データを読み出し、出力アドレスが示すメモリへオリジナルデータを書き出す処理です。復元処理における関数の呼び出しは、次の順序で行います。

```
relc_InitDecode ()   relc_Decode ()
```

圧縮を行う場合は、圧縮モードとして圧縮能力を数値(0もしくは1~128)で指定します。0を指定すると、圧縮能力は最も高くなりますが、処理速度は最も遅くなります。1~128の数値を指定すると、大きな数値ほど圧縮能力は高く、処理速度は遅くなり、小さな数値ほど圧縮能力は低く、処理速度は速くなります。128を指定すると、0を指定したときに比べて圧縮能力は若干低下しますが、処理速度は若干速くなります。

圧縮モードの代表的な値は、次のように定義しています。

RELC_MODE_MAXIMUM	圧縮モードを最高圧縮に設定します。(圧縮モード 0) 圧縮能力は最も高くなりますが、処理速度は最も遅くなります。
RELC_MODE_HIGH	圧縮モードを高圧縮に設定します。(圧縮モード 32) [推奨値]
RELC_MODE_MIDDLE	圧縮モードを中圧縮に設定します。(圧縮モード 16)
RELC_MODE_LOW	圧縮モードを低圧縮に設定します。(圧縮モード 4)
RELC_MODE_MINIMUM	圧縮モードを最低圧縮に設定します。(圧縮モード 1) 圧縮能力は最も低くなりますが、処理速度は最も速くなります。

この設定の有効性は、処理するオリジナルデータに依存します。オリジナルデータによっては、設定の効果が現れない場合があります。

6.1 relc_InitEncode ()

[機能]

圧縮処理のための準備を行います。

[形式]

```
int relc_InitEncode(  
int          nEncodeLevel  
)
```

[引数]

nEncodeLevel 圧縮モード

[戻り値]

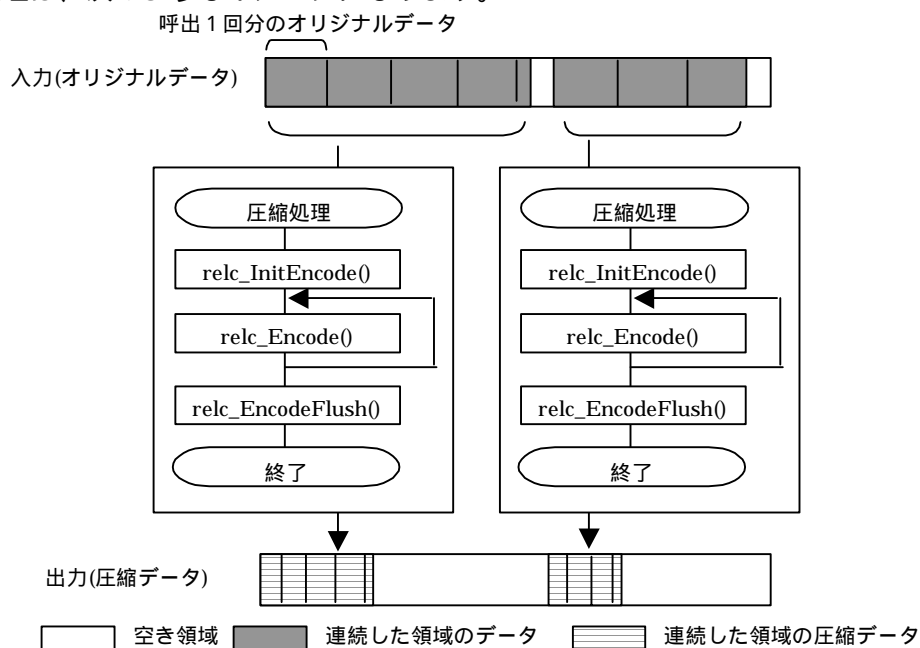
REL_CV_NORMAL == 正常終了
REL_EV_ILLPARAM == 圧縮モード指定エラー

[説明]

一連のオリジナルデータをrelc_Encode()で処理する前に呼び出してください。引数として *nEncodeLevel* へ圧縮モードを指定する必要があります。

nEncodeLevel は圧縮モードを指定します。圧縮モードの指定方法については、「6 データ処理用関数」を参照してください。

圧縮処理は、次のようなイメージになります。



6.2 relc_InitDecode ()

[機能]

復元処理のための準備を行います。

[形式]

```
int relc_InitDecode(  
void  
)
```

[引数]

無し

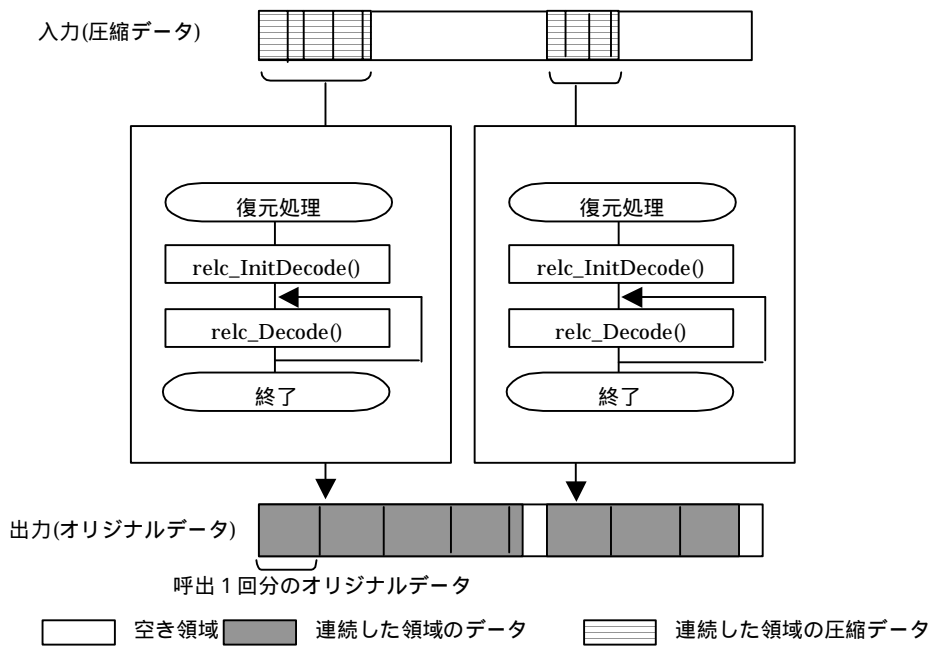
[戻り値]

RELC_RV_NORMAL == 正常終了

[説明]

連続した領域の圧縮データをrelc_Decode()で処理する前に呼び出してください。

復元処理は、次のようなイメージになります。



6.3 relc_Encode ()

[機能]

オリジナルデータを圧縮します。

[形式]

```
int relc_Encode(  
unsigned char RELC_FAR *   pInBuf ,  
unsigned char RELC_FAR *   pOutBuf ,  
unsigned short RELC_FAR *  pnInBuf ,  
unsigned short RELC_FAR *  pnOutBuf  
)
```

[引数]

pInBuf	オリジナルデータの入力バッファアドレス
pOutBuf	圧縮データの出力バッファアドレス
pnInBuf	[in] 入力バッファ中のオリジナルデータのバイト数
	[out] 入力バッファ中で使用したオリジナルデータのバイト数
pnOutBuf	[in] 無視
	[out] 出力バッファへ出力された圧縮データのバイト数

[戻り値]

RELC_RV_NORMAL	==	正常終了
RELC_EV_ILLPARAM	==	入力サイズエラー

[説明]

pInBufから **pnInBuf**バイトのオリジナルデータを読み取り、圧縮データを **pOutBuf**へ書き込みます。 **pnInBuf**へは、最大RELC_MAX_ORIGINAL(32768)が指定できます。関数から復帰すると、**pnInBuf**へ使用されたオリジナルデータのバイト数、**pnOutBuf**へ出力された圧縮データのバイト数が格納されています。

RELC_MAX_ORIGINAL バイト以上のオリジナルデータを圧縮する場合は、ループ処理を行い RELC_MAX_ORIGINAL 毎にこの関数を呼び出してください。関数を呼び出した後に得られる **pnInBuf**を減算することで、残りのオリジナルデータのバイト数が求められます。残りのオリジナルデータサイズがRELC_REUSE_SIZE 以下になったらループを終了し、relc_EncodeFlush()を呼び出してください。

pInBuf の領域への操作は読み取りのみで書き込みは行いません。 **pOutBuf** の領域への操作は書き込みのみで読み取りは行いません。

6.4 relc_EncodeFlush ()

[機能]

圧縮の処理を終了します。

[形式]

```
int relc_EncodeFlush (  
  unsigned char RELC_FAR *   pInBuf ,  
  unsigned char RELC_FAR *   pOutBuf ,  
  unsigned short RELC_FAR *  pnInBuf ,  
  unsigned short RELC_FAR *  pnOutBuf  
)
```

[引数]

pInBuf	オリジナルデータの入力バッファアドレス
pOutBuf	圧縮データの出力バッファアドレス
pnInBuf	[in] 入力バッファ中のオリジナルデータのバイト数 [out] 入力バッファへ使用したバイト数
pnOutBuf	[in] 無視 [out] 出力バッファへ出力したバイト数

[戻り値]

RELC_RV_NORMAL	==	正常終了
RELC_EV_ILLDATA	==	呼び出し順序不正エラー

[説明]

*pInBuf*から圧縮データを読み取り、終端を含めた圧縮データを *pOutBuf*へ書き込みます。関数から復帰すると、*pnInBuf*へ使用されたオリジナルデータのバイト数、*pnOutBuf*へ出力した圧縮データのバイト数が格納されています。

この呼び出しにより、圧縮データの末尾となる終端コードが出力されます。

6.5 relc_Decode ()

[機能]

圧縮データを復元します。

[形式]

```
int relc_Decode(  
unsigned char RELC_FAR *   pInBuf ,  
unsigned char RELC_FAR *   pOutBuf ,  
unsigned short RELC_FAR *  pnInBuf ,  
unsigned short RELC_FAR *  pnOutBuf  
)
```

[引数]

pInBuf	圧縮データの入力バッファアドレス
pOutBuf	オリジナルデータの出力バッファアドレス
pnInBuf	[in] 無視
	[out] 入力バッファへ使用したバイト数
pnOutBuf	[in] 無視
	[out] 出力バッファへ出力したバイト数

[戻り値]

RELC_RV_NORMAL	==	正常終了
RELC_RV_ENDEDATA	==	全ての圧縮データを処理した
RELC_EV_ILLDATA	==	データエラー

[説明]

*pInBuf*から圧縮データを読み取り、復元したオリジナルデータを *pOutBuf*へ書き込みます。関数から復帰すると、*pnInBuf*へ圧縮データへのバッファアドレスの更新バイト数、*pnOutBuf*へオリジナルデータデータへのバッファアドレスの更新バイト数が格納されています。

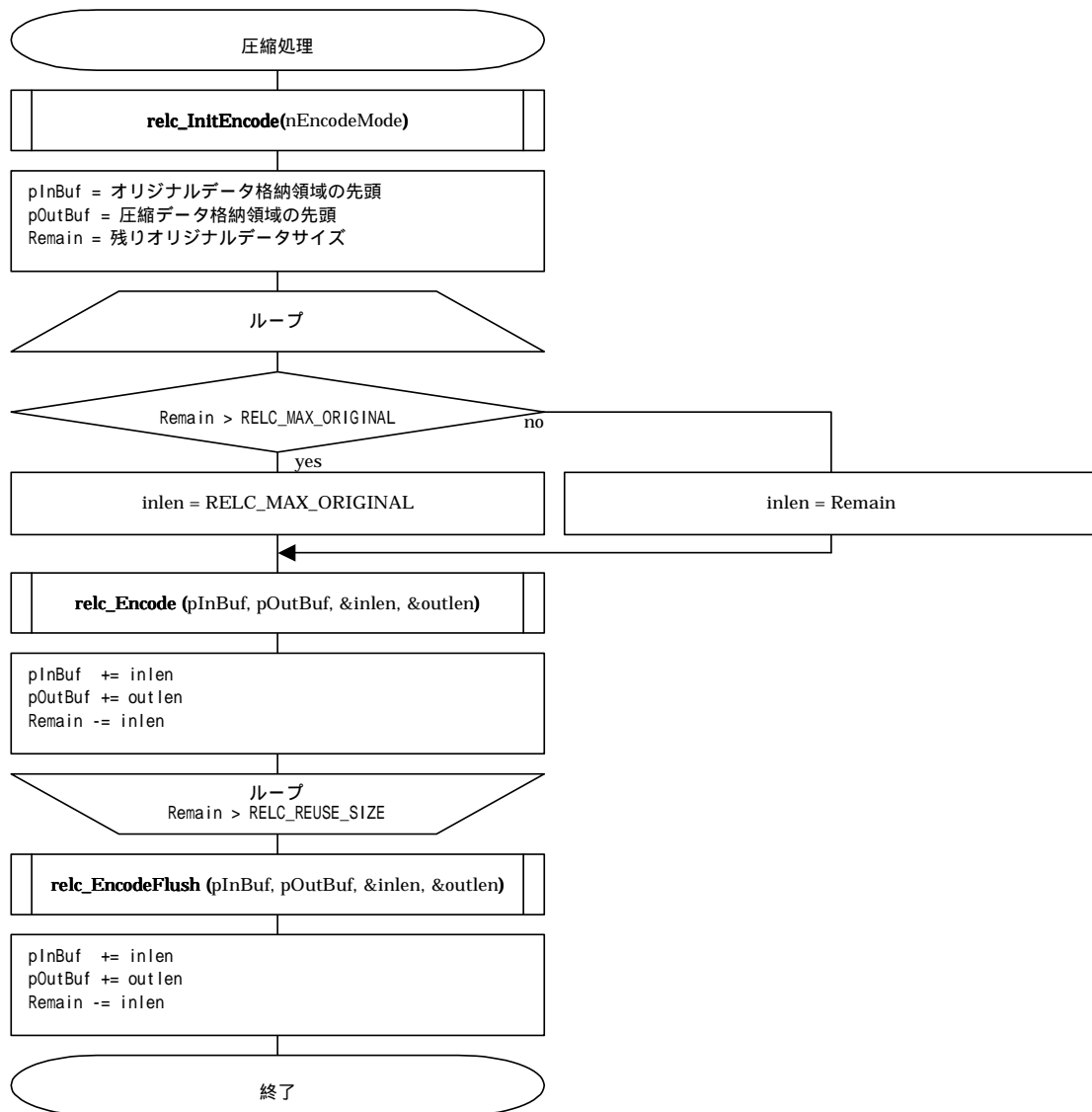
全ての圧縮データの復元を行うと戻り値としてRELC_RV_ENDEDATAを返します。戻り値としてRELC_RV_NORMALが返った場合は、次の単位分の圧縮データを準備したうえで本関数を再び呼び出して下さい。

*pInBuf*の領域への操作は読み取りのみで書き込みは行いません。*pOutBuf*の領域への操作は読み書きを行います。

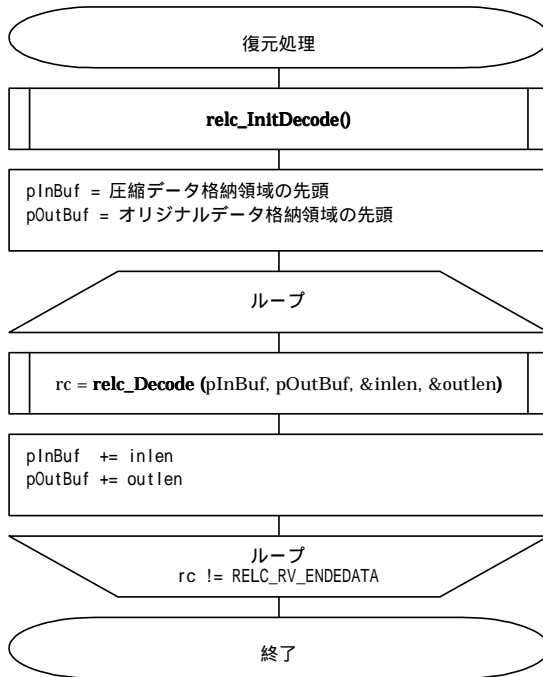
この関数では、処理速度を優先するため厳密なエラーチェックは行いません。従って、圧縮データの正当性はユーザー側で保障して下さい。

6.6 API の使用方法

6.6.1 圧縮処理



6.6.2 復元処理



7 定義

本製品の定義について説明します。定義は、relic.hで記述されています。

7.1 定数

マクロ	値 ⁽¹⁰⁾	説明
RELC_MAX_ORIGINAL	32768	呼び出し1回毎の最大オリジナルデータサイズ
RELC_MAX_CODE	32770	呼び出し1回毎の最大圧縮データサイズ
RELC_REUSE_SIZE	8192	バッファリング時の再利用サイズおよび圧縮時の終了条件
RELC_MODE_MAXIMUM	0	圧縮モード最高
RELC_MODE_HIGH	32	圧縮モード高
RELC_MODE_MIDDLE	16	圧縮モード中
RELC_MODE_LOW	4	圧縮モード低
RELC_MODE_MINIMUM	1	圧縮モード最低
RELC_RV_NORMAL	0	正常終了時の戻り値
RELC_RV_ENDEDATA	1	復元終了時の戻り値
RELC_EV_ILLPARAM	10	パラメーター不正
RELC_EV_ILLDATA	11	不正データ検出

8 エラーリファレンス

エラーコードについて説明します。

定義	説明
RELC_RV_NORMAL	正常終了しました。
RELC_RV_ENDEDATA	全ての圧縮データを処理しました。
RELC_EV_ILLPARAM	パラメーターが不正です。
RELC_EV_ILLDATA	圧縮データが異常です。 または relc_EncodeFlush ()の呼び出しタイミングが違います。

9 PC 版評価プログラムについて

本製品には、PC 上で圧縮ファイルを作成、および、作成された圧縮ファイルを復元するための評価プログラム (RelcL.exe) が添付されています。このプログラムは、評価用にのみ利用できます。

(このプログラムを配布することはできません。システムに利用する場合は、別途ライセンス製品を購入する必要があります。)

・起動方法

prompt> **RelcL.exe option infile outfile [mode]** ¹

option 圧縮処理の場合は、'E' または 'e' を指定します。
復元処理の場合は、'D' または 'd' を指定します。

infile 入力ファイル名を指定します。


outfile 出力ファイル名を指定します。

mode 圧縮モードを指定します。(0 または 1 ~ 128)
この指定を省略した場合、'0' が指定されたときと同じ動作となります。
指定の際の目安として本製品の定数値を示します。

値	定数
0	RELC_MODE_MAXIMUM [デフォルト]
32	RELC_MODE_HIGH
16	RELC_MODE_MIDDLE
4	RELC_MODE_LOW
1	RELC_MODE_MINIMUM

圧縮時のみ有効で、復元時は無視されます。

¹ [...] は、その記述が省略可能であることを示します。


FUJITSU