

RELC DLL for Windows 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。

■ 著作権

RELC DLL for Windows 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。

■ 商標

Microsoft、Windows、Windows Server、Windows Vista、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。

■ 製品名称

本マニュアルでは、製品名称を次のように略して表記しています。

Windows Server® 2008 operating system を Windows Server 2008 と表記しています。

Windows Server® 2003 operating system を Windows Server 2003 と表記しています。

Windows Vista® operating system を Windows Vista と表記しています。

Windows® XP operating system を Windows XP と表記しています。

Visual Studio 2005 Professional Edition を Visual Studio 2005 と表記しています。

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. 本資料に記載された製品のパスワード機能は簡易的なもので、セキュリティ目的にはご使用になれません。お客様の機密情報保護を目的とした用途にはご使用にならないでください。
7. 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	動作環境	1
3	ライブラリの作成環境	1
4	ライブラ리를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	コンパイルオプションの指定	2
4.3	ライブラリのリンク	2
4.3.1	インポートライブラリを使用する	2
4.3.2	プログラム内からロードする	2
5	API	3
6	高レベル API	5
6.1	relc_HI_FileEncode ()	7
6.2	relc_HI_FileDecode ()	8
6.3	relc_HI_Open ()	9
6.4	relc_HI_Write ()	10
6.5	relc_HI_Read ()	11
6.6	relc_HI_Close ()	12
6.7	relc_HI_GetAddInfoContents ()	13
6.8	relc_HI_GetAddInfo ()	14
6.9	高レベル API の使用方法	15
6.9.1	ファイルからファイルへの圧縮および復元	15
6.9.2	メモリからファイルへの圧縮	15
6.9.3	ファイルからメモリへの復元	16
7	中レベル API	17
7.1	relc_MI_InitEncode	18
7.2	relc_MI_InitDecode	19
7.3	relc_MI_CallBack ()	20
7.4	relc_MI_Encode ()	21
7.5	relc_MI_Decode ()	22
7.6	relc_MI_GetAddInfoContents ()	23
7.7	relc_MI_GetAddInfo ()	24
7.8	中レベル API の使用方法	25
7.8.1	圧縮処理	25
7.8.2	復元処理	26
8	低レベル API	27
8.1	relc_InitEncode ()	28
8.2	relc_InitDecode ()	29
8.3	relc_Encode ()	30
8.4	relc_EncodeFlush ()	31
8.5	relc_Decode ()	32
8.6	低レベル API の使用方法	33
8.6.1	圧縮処理	33

8.6.2	復元処理	34
9	定義	35
9.1	定数	35
9.2	型	35
10	エラーリファレンス	36
11	制限事項	37
12	用語集	38
13	PC 版評価プログラムについて	39

1 概要

RELC DLL for Windows custom01は、可逆圧縮方式(ロスレス圧縮方式)を使用してファイルや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 Embded Lossless Compression の略称です。

製品パッケージには、以下のプログラム、ライブラリ、ヘッダーを格納しています。

RelcEva.exe	PC 版評価用プログラム (32 ビット)
MD¥	MD オプション用ライブラリ
relc_sa_md_x64.dll	64 ビット(x64)用ライブラリ
relc_sa_md_x64.lib	relc_sa_md_x64.dll 用インポートライブラリ
relc_sa_md_x86.dll	32 ビット(x86)用ライブラリ
relc_sa_md_x86.lib	relc_sa_md_x86.dll 用インポートライブラリ
MT¥	MT オプション用ライブラリ
relc_sa_mt_x64.dll	64 ビット(x64)用ライブラリ
relc_sa_mt_x64.lib	relc_sa_mt_x64.dll 用インポートライブラリ
relc_sa_mt_x86.dll	32 ビット(x86)用ライブラリ
relc_sa_mt_x86.lib	relc_sa_mt_x86.dll 用インポートライブラリ
relc.h	低レベル API 用ヘッダーファイル
relcm.h	中レベル API 用ヘッダーファイル
relch.h	高レベル API 用ヘッダーファイル
relctype.h	型定義ヘッダーファイル

※製品パッケージの詳細については、パッケージ内の"README.TXT"に記述しています。

2 動作環境

本ライブラリの動作環境は、以下のとおりです。

Windows Server 2008
Windows Server 2003
Windows Vista
Windows XP

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を用いて作成しています。ライブラリを組み込む際は、Microsoft Visual Studio 2005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 コンパイルオプションの指定

コンパイルオプションにおいて、以下のマクロを定義してください。

"_ANSI_"、"RELC_DYNAMIC"

4.3 ライブラリのリンク

プロセッサの種類(x86/x64)と CRT オプション(MD/MT)を組み合わせた 4 種類のダイナミックリンクライブラリを用意しています。

リンク方法には、インポートライブラリを利用する方法とプログラム内からロードする方法の 2 種類の方法があります。

4.3.1 インポートライブラリを使用する

リンク時に `relc_sa_XX_YYY.lib` を指定します。

4.3.2 プログラム内からロードする

プログラム内から Windows API を使用して、"`relc_sa_XX_YYY.dll`"をロードし API アドレスを取得のうえで使用します。

詳しくは開発環境のマニュアルを参照してください。

XX は CRT オプションの種類、YYY はプロセッサの種類を表します。

5 API

3つのレベル（高、中、低）のAPIを提供しています。レベルが高くなる程ライブラリ側が処理を負担し、呼び出し元のプログラムをシンプルに作成することができます。

各レベルで処理できる内容は、以下のようになっています。

高レベルAPI

- ・ メモリからファイルへの圧縮
- ・ ファイルからメモリへの復元
- ・ ファイルからファイルへの圧縮および復元
- ・ ヘッダー情報および付加情報の付加

中レベルAPI

- ・ 任意サイズでのメモリからメモリへの圧縮および復元
- ・ ヘッダー情報および付加情報の付加

低レベルAPI

- ・ 規定サイズでのメモリからメモリへの圧縮および復元

以降の説明では、区別のために圧縮前の一連の元データを元ストリーム、圧縮後の一連の圧縮データを圧縮ストリーム、復元後の一連の復元データを復元ストリームと記述します。

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（RELC_MODE_MAXIMUM）を指定すると、圧縮能力は最も高く（最高圧縮）になりますが、圧縮速度は低速です。128を指定すると圧縮能力はRELC_MODE_MAXIMUMより若干低下しますが、圧縮速度は若干高速になります。更に小さい値（RELC_MODE_HIGH, RELC_MODE_MID, RELC_MODE_LOW, または 128より小さい値）を指定すると、圧縮能力が低下する代わりに、圧縮速度は向上します。

次の定義を使用することができます。

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)
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、処理する元ストリームの内容に依存します。元ストリームの内容によっては、設定の効果が現れない場合があります。

中レベル以上の API では、圧縮ストリームにヘッダー情報を付加することができます。ヘッダー情報を付加した場合は、さらに付加情報を付加することができますようになります。付加情報を付加することにより、復元の際に元ストリームのサイズを得るといったことができるようになります。ヘッダー情報は、必ず圧縮ストリームの先頭に配置されている必要があります。

付加情報には、次のような種類があります。

RELC_KIND_ZERO	付加情報無し
RELC_KIND_CODESIZE	圧縮ストリームサイズ
RELC_KIND_ORGSIZE	元ストリームサイズ

圧縮ストリームサイズは、圧縮ストリームのサイズをバイト数で格納します。
元ストリームサイズは、元ストリームのサイズをバイト数で格納します。

中レベルの API では、仕様上はリニアなメモリ空間への処理を想定しています。従って、有限サイズのバッファを介してそのバッファの大きさ以上の圧縮ストリームや元ストリームを処理しようとするとはバッファリングが必要になります。また、付加情報を付ける場合に、**RELC_KIND_CODESIZE** といった事前に値が決定しない指定はできません。

高レベルの API では、ファイル操作を全てライブラリが行います。付加情報も全ての種類を指定できます。

6 高レベル API

このレベルでは、メモリからファイルへの圧縮、ファイルからメモリへの復元、ファイルからファイルへの圧縮および復元機能を提供します。

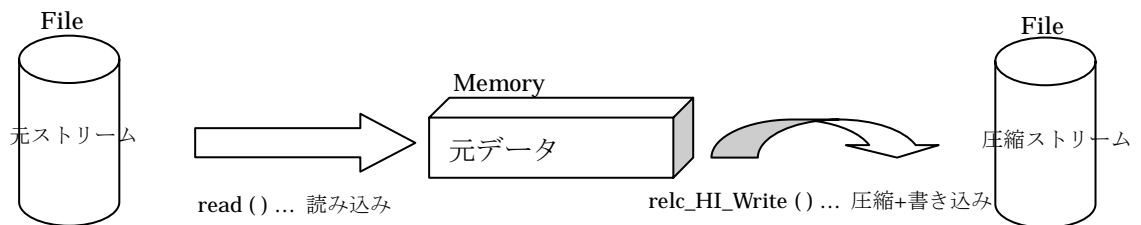
ファイルからファイルへの処理のために、次の 2 つの API を提供しています。

`relc_HI_FileEncode ()`

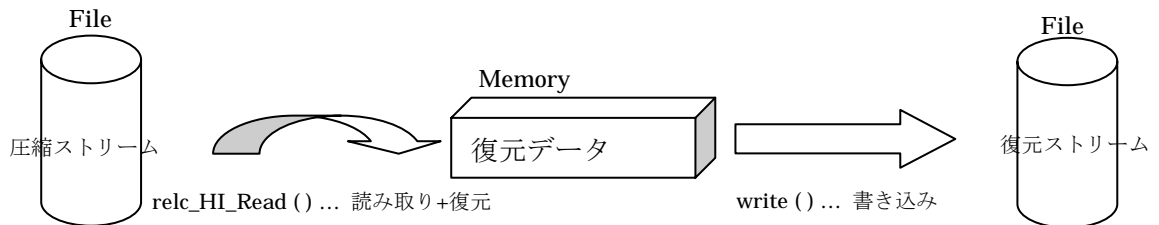
`relc_HI_FileDecode ()`

ファイルからファイルへの圧縮または復元を行う場合は、`relc_HI_FileEncode ()` と `relc_HI_FileDecode ()` を使用します。

`relc_HI_FileEncode ()` の操作イメージ



`relc_HI_FileDecode ()` の操作イメージ

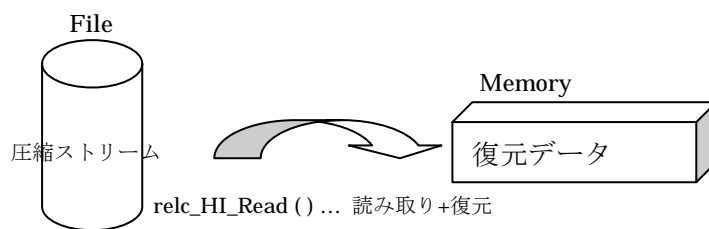


メモリからファイルへの圧縮処理とファイルからメモリへの復元処理のために、以下の API を提供しています。

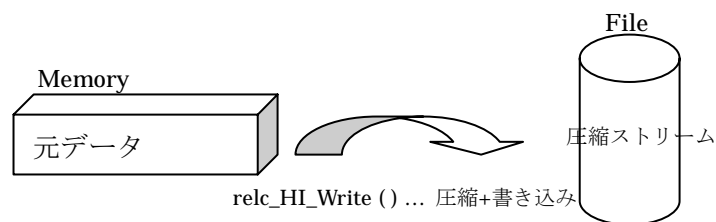
```
relc_HI_Open ()
relc_HI_Read ()
relc_HI_Write ()
relc_HI_Close ()
relc_HI_GetAddInfoContents ()
relc_HI_GetAddInfo ()
```

relc_HI_Read ()は、圧縮ストリームから圧縮データを読み出し、復元した復元データをメモリに出力する処理を行います。また、**relc_HI_Write ()**は、メモリから元データを読み出し、圧縮した圧縮データを圧縮ストリームに出力する処理を行います。

relc_HI_Write () の操作イメージ



relc_HI_Read () の操作イメージ



6.1 relc_HI_FileEncode ()

[機能]

ファイルを圧縮します。

[形式]

```
int relc_HI_FileEncode (
const char RELC__FAR *      pOriginalFileName ,
const char RELC__FAR *      pEncodedFileName ,
int                          nEncodeLevel ,
int                          bAddInfo
)
```

[引数]

pOriginalFileName	元ストリームが格納されているファイルのファイル名が格納された領域へのポインタ
pEncodedFileName	圧縮ストリームを格納するファイルのファイル名が格納された領域へのポインタ
nEncodeLevel	圧縮モード
bAddInfo	付加する情報の種類

[戻り値]

RELc_RV_NORMAL	==	正常終了
RELc_EV_ILLPARAM	==	パラメータが異常
RELc_EV_FILEREAD	==	ファイルの読み取りに失敗しました
RELc_EV_FILEWRITE	==	ファイルの書き込みに失敗しました
RELc_EV_STOPUSER	==	要求により処理を中断しました
RELc_EV_CLOSE	==	ファイルのクローズでエラーがありました。

[説明]

pOriginalFileName が示すファイルの元ストリームを圧縮し、圧縮ストリームを **pEncodedFileName** が示すファイルへ出力します。

nEncodeLevel は圧縮モードを指定します。大きな数値ほど圧縮能力は高くなり、処理速度は低くなります。圧縮モードの指定方法については、「5 API」を参照してください。

bAddInfo は、付加するヘッダー情報や付加情報を指定します。

指定の際は、次の定数の論理和で指定します。

RELc_KIND_NO_INFO	ヘッダー情報無し
RELc_KIND_ZERO	付加情報無し
RELc_KIND_ORGSIze	元ストリームサイズ
RELc_KIND_CODESIze	圧縮ストリームサイズ

RELc_KIND_NO_INFO とそれ以外の定数を同時に指定しても、RELc_KIND_NO_INFO 以外は無視されます。

6.2 relc_HI_FileDecode ()

[機能]

ファイルを復元します。

[形式]

```
int relc_HI_FileDecode (
const char RELC__FAR *      pEncodedFileName ,
const char RELC__FAR *      pOriginalFileName ,
int                          nIsAddInfo ,
int RELC_FAR *              *pbAddInfo
)
```

[引数]

pEncodedFileName	圧縮ストリームが格納されているファイルのファイル名が格納された領域へのポインタ
pOriginalFileName	復元ストリームを格納するファイルのファイル名が格納された領域へのポインタ
nIsAddInfo	ヘッダー情報の有無
pbAddInfo	付加情報の内容を格納する領域へのポインタ

[戻り値]

RELc_RV_NORMAL	==	正常終了
RELc_EV_ILLPARAM	==	パラメータが異常
RELc_EV_FILEREAD	==	ファイルの読み取りに失敗しました
RELc_EV_FILEWRITE	==	ファイルの書き込みに失敗しました
RELc_EV_STOPUSER	==	要求により処理を中断しました
RELc_EV_CLOSE	==	ファイルのクローズでエラーがありました。

[説明]

pEncodedFileName が示すファイルの圧縮ストリームを復元し、復元ストリームを **pOriginalFileName** が示すファイルへ出力します。API の復帰時に **pbAddInfo** へ付加情報の内容を格納します。

nIsAddInfo へは、付加情報の有無を指定します。**pEncodedFileName** が示すファイルに付加情報がある場合は **1** を無い場合は **0** を指定します。

pbAddInfo へ格納される値は、次の定数が論理和された値です

RELc_KIND_ZERO	付加情報無し
RELc_KIND_ORGSIze	元ストリームサイズ
RELc_KIND_CODESIze	圧縮ストリームサイズ

6.3 relc_HI_Open ()

[機能]

圧縮や復元のためにファイルをオープンします。

[形式]

```
int relc_HI_Open (
const char RELC__FAR *      pFilename ,
int                          nOpenMode ,
int                          nEncodeLevel ,
int                          bAddInfo
)
```

[引数]

pInf	高レベル用制御情報領域へのポインタ
pFilename	ファイル名の格納されている領域へのポインタ
nOpenMode	オープンするモード
nEncodeLevel	圧縮モード
bAddInfo	ヘッダー情報、付加情報の種類

[戻り値]

RELC_RV_NORMAL	==	正常終了
RELC_EV_CANTOPEN	==	ファイルが開けません
RELC_EV_ILLPARAM	==	パラメータエラー

[説明]

pFilename のファイルをオープンして圧縮または復元の準備をします。

nOpenMode はオープンするモードを指定します。次の定数を使用してください。

RELC_OPMD_ENCODE	圧縮書き込みでファイルをオープン
RELC_OPMD_DECODE	復元読み出しでファイルをオープン

nEncodeLevel は圧縮モードを指定します。大きな数値ほど圧縮能力は高くなり、処理速度は低くなります。圧縮モードの指定方法については、「5 API」を参照してください。

bAddInfo は、付加するヘッダー情報や付加情報を指定します。

指定の際は、次の定数を使用します。

RELC_KIND_NO_INFO	ヘッダー情報無し
RELC_KIND_ZERO	付加情報無し
RELC_KIND_ORGSIZE	元ストリームサイズ
RELC_KIND_CODESIZE	圧縮ストリームサイズ

[圧縮の場合 (**nOpenMode** == RELC_OPMD_ENCODE)]

定数を論理和した値を指定します。

RELC_KIND_NO_INFO とそれ以外の定数を論理和して同時に指定しても、RELC_KIND_NO_INFO 以外は無視されます。

[復元の場合 (**nOpenMode** == RELC_OPMD_DECODE)]

ヘッダー情報が付加されている圧縮ファイルをオープンする場合は 1、ヘッダー情報が付加されていない圧縮ファイルをオープンする場合は 0 を指定します。

6.4 relc_HI_Write ()

[機能]

指定されたメモリの内容を圧縮しファイルへ書き込みます。

[形式]

```
int relc_HI_Write (
char RELC_FAR *          pInBuf ,
long RELC_FAR *         pnInBuf
)
```

[引数]

pInBuf	元データが格納されているバッファへのポインタ
pnInBuf	[in] バッファ内の元データのサイズ (0x00000001~0x7FFFFFFF) [out] 処理した元データのサイズ (0x00000001~0x7FFFFFFF)

[戻り値]

RELC_RV_CONTINUE	==	バッファ内のデータを使い切りました
RELC_EV_FILEWRITE	==	ファイルの書き込みに失敗しました
RELC_EV_STOPUSER	==	要求により処理を中断しました

[説明]

pInBufから元データを読み取り、圧縮データをファイルへ書き込みます。読み取った元データのサイズは、**pnInBuf**に格納します。

中レベルに定義されている**relc_MI_CallBack ()**を使用してコールバック関数を設定することができます。コールバック関数を設定することにより呼び出し元から中断の指定を行えるようになります。詳しくは、「7.3 relc_MI_CallBack ()」を参照してください。

6.5 relc_HI_Read ()

[機能]

指定されたメモリへファイルから読み込み復元します。

[形式]

```
int relc_HI_Read (
char RELC_FAR *          pOutBuf ,
long RELC_FAR *         pnOutBuf
)
```

[引数]

pOutBuf	復元データを出力するバッファへのポインタ
pnOutBuf	[in]バッファのサイズ (0x00000001~0x7FFFFFFF) [out]出力した復元データのサイズ (0x00000001~0x7FFFFFFF)

[戻り値]

RELC_RV_CONTINUE	==	出力バッファを使い切りました
RELC_RV_ENDEDATA	==	全ての情報を復元しました
RELC_EV_FILEREAD	==	ファイルの読み取りに失敗しました
RELC_EV_ILLFORM	==	ファイル形式が正しくありません
RELC_EV_STOPUSER	==	要求により処理を中断しました

[説明]

ファイルから圧縮データを読み取り *pOutBuf* へ書き込みます。書き込んだ復元データのバイト数は、*pnOutBuf*に格納します。

ファイルの内容を全て復元して *pOutBuf*へ出力できた場合は、RELC_EV_EDTEND を返します。

中レベルに定義されているrelc_MI_CallBack ()を使用してコールバック関数を設定しておくこと、コールバック関数内で呼び出し元から中断の指定をすることができます。詳しくは、「7.3 relc_MI_CallBack ()」を参照してください。

6.6 relc_HI_Close ()

[機能]

ファイルをクローズします。

[形式]

```
int relc_HI_Close (
void
)
```

[引数]

なし

[戻り値]

RELC_RV_NORMAL == 正常終了

RELC_EV_CLOSE == ファイルのクローズでエラーがありました。

[説明]

relc_HI_Open ()でオープンしたファイルをクローズします。

6.7 relc_HI_GetAddInfoContents ()

[機能]

圧縮データから付加情報の内容を得ます。

[形式]

```
int relc_HI_GetAddInfoContents (  
int RELC_FAR *          pbAddInfo  
)
```

[引数]

pbAddInfo 付加情報の種類を格納する領域へのポインタ

[戻り値]

RELc_RV_NORMAL == 正常終了
RELc_EV_ILLFORM == フォーマットが正しくありません

[説明]

relc_HI_Open ()でオープンしたヘッダー情報付きの圧縮ストリームから付加情報を読み取り、**pbAddInfo**へ付加情報の種類を格納します。実際の付加情報の値を得るには、続けて**relc_HI_GetAddInfo ()**を呼び出してください。

pbAddInfoへ格納される値は、次の定数の論理和です。

RELc_KIND_ZERO	付加情報無し
RELc_KIND_ORGSIze	元ストリームサイズ
RELc_KIND_CODESIze	圧縮ストリームサイズ

6.8 relc_HI_GetAddInfo ()

[機能]

圧縮ストリームから付加情報を得ます。

[形式]

```
int relc_HI_GetAddInfo (
int                                nAddInfo ,
RELC_64VAR *                       pnAddInfo
)
```

[引数]

nAddInfo 取得する付加情報の種類
pnAddInfo 値を格納する領域へのポインタ

[戻り値]

RELC_RV_NORMAL == 正常終了
RELC_EV_ILLPARAM == パラメータが不正です

[説明]

nAddInfo が示す圧縮ストリームの付加情報を読み取り、**pnAddInfo** へ格納します。

nAddInfo へは、次の値を指定します。

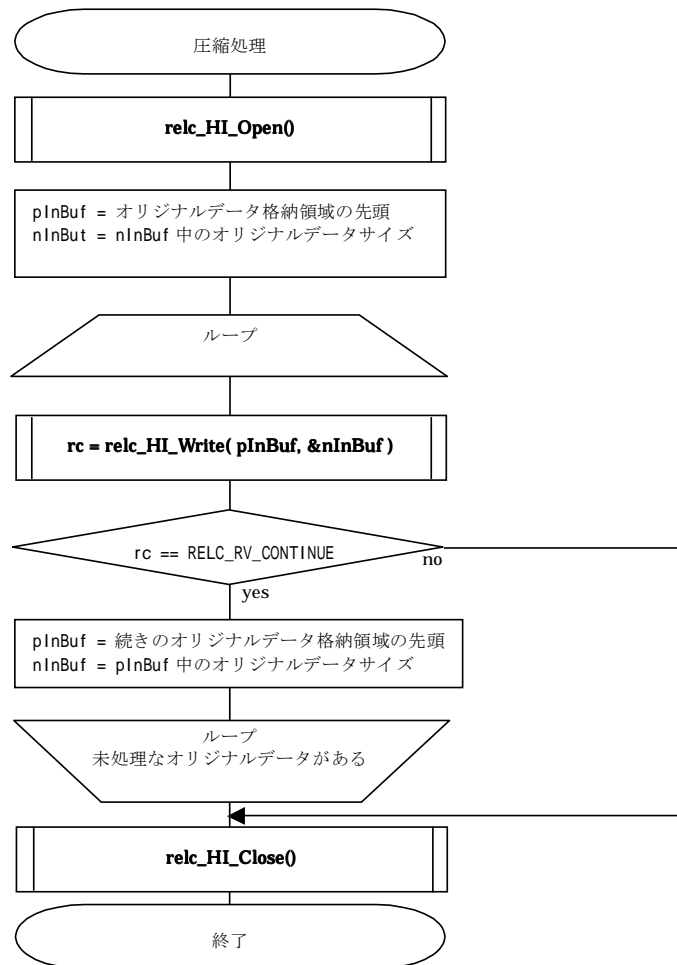
RELC_KIND_ORGSIZE	元ストリームサイズ
RELC_KIND_CODESIZE	圧縮ストリームサイズ

6.9 高レベル API の使用方法

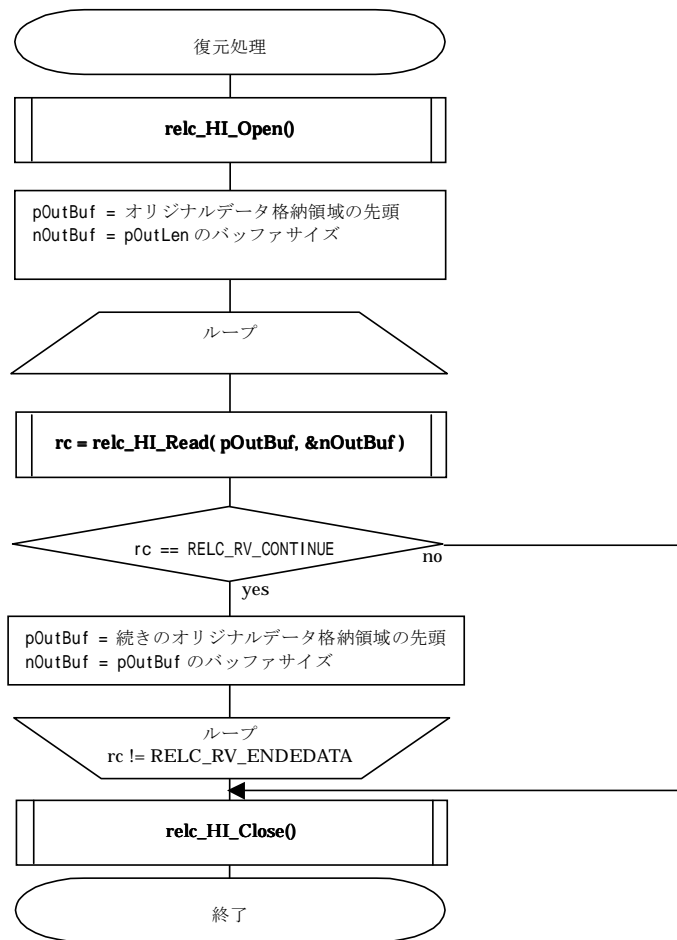
6.9.1 ファイルからファイルへの圧縮および復元

ファイル名指定で `rele_HI_FileEncode ()` または、`rele_HI_FileDecode ()` を呼び出してください。

6.9.2 メモリからファイルへの圧縮



6.9.3 ファイルからメモリへの復元



7 中レベル API

このレベルでは、任意のサイズ指定でのメモリからメモリへの圧縮および復元機能を提供します。また、復元処理の際に利用される付加情報を付ける機能も提供します。

以下の API を提供しています。

```
relc_MI_InitEncode ()
relc_MI_Encode ()
relc_MI_InitDecode ()
relc_MI_Decode ()
relc_MI_CallBack ()
relc_MI_GetAddInfoContents ()
relc_MI_GetAddInfo ()
```

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_MI_InitEncode ()    relc_MI_Encode ()
```

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_MI_InitDecode ()    relc_MI_Decode ()
```

このレベルの API の戻り値は、制御フィールドとエラーフィールドに分かれています。エラーコードを得るには、戻り値と **RELC_MEV_MASK** を論理積してください。

制御フィールドはビットフィールドになっています。そのため、戻り値と次の値を論理積した結果が真の場合は、その状態に応じた処理を行ってください。

RELC_MCV_NODATA	入力データを全て処理しました。次の入力データを与えてください
RELC_MCV_NOBUFF	出力バッファを全て使用しました。新しい出力バッファを用意してください。
RELC_MCV_EDTEND	全ての処理が終了しました。終了処理を行ってください。

7.1 relc_MI_InitEncode

[機能]

圧縮処理のための準備を行います。

[形式]

```
int relc_MI_InitEncode(  
int          nEncodeLevel ,  
int          nAddInfo ,  
RELC_64VAR  nOriginalSize  
)
```

[引数]

nEncodeLevel	圧縮モード
nAddInfo	付加情報の種類
nOriginalSize	元ストリームサイズ

[戻り値]

エラーフィールド (戻り値 & RELC_MEV_MASK)
RELC_RV_NORMAL == 正常終了
RELC_EV_ILLPARAM == 圧縮モード指定エラー

制御フィールド (戻り値 & RELC_MCV_XXXX)
無し

[説明]

元ストリームをrelc_MI_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、**nAddInfo** へ付加情報の種類、**nOriginalSize** へ元ストリームの総バイト数を指定します。

nEncodeLevel は圧縮モードを指定します。大きな数値ほど圧縮能力は高くなり、処理速度は低くなります。圧縮モードの指定方法については、「5 API」を参照してください。

nAddInfo は、次の定数を使用してください。

RELC_KIND_NO_INFO	ヘッダー情報無し
RELC_KIND_ZERO	付加情報無し
RELC_KIND_ORGSIZE	元ストリームサイズ

nOriginalSize へは圧縮しようとする元ストリームの総バイト数を指定してください。

7.2 relc_MI_InitDecode

[機能]

復元処理のための準備を行います。

[形式]

```
int relc_MI_InitDecode(  
int nAddInfo  
)
```

[引数]

nAddInfo ヘッダー情報の有無

[戻り値]

エラーフィールド (戻り値 & RELC_MEV_MASK)
RELC_RV_NORMAL == 正常終了

制御フィールド (戻り値 & RELC_MCV_XXXX)
無し

[説明]

圧縮ストリームをrelc_MI_Decode()で処理する前に呼び出してください。**nlsAddInfo** はヘッダー情報の有無を指定します。復元する圧縮ストリームにヘッダー情報がある場合は **1** を無い場合は **0** を指定します。

7.3 relc_MI_CallBack ()

[機能]

コールバック関数を設定します。

[形式]

```
int relc_MI_CallBack(  
void RELC_FAR *      pCallback  
)
```

[引数]

pCallback コールバック関数へのポインタ

[戻り値]

エラーフィールド (戻り値 & RELC_MEV_MASK)
RELC_RV_NORMAL == 正常終了

制御フィールド (戻り値 & RELC_MCV_XXXX)

無し

[説明]

中レベルの API は、低レベルの API を連続して呼び出します。そのため大きなデータを処理させると、長い時間処理が終わらない場合があります。

この API を使用してコールバック関数を設定しておくと、低レベル API を 1 回呼び出すごとにセットされた関数を呼び出します。

コールバック関数は、次のように定義されている必要があります。

```
void ( RELC_FAR * pCallback)( int RELC_FAR * );
```

ライブラリは、ライブラリ内部の処理中断用の変数を引数としてコールバック関数を呼び出します。この引数に 0 以外の数値を代入することで処理を中断することができます。

7.4 relc_MI_Encode()

[機能]

元データを圧縮します。

[形式]

```
int relc_MI_Encode (
unsigned char RELC_FAR *    pInBuf ,
unsigned char RELC_FAR *    pOutBuf ,
long RELC_FAR *            pnInBuf ,
long RELC_FAR *            pnOutBuf
)
```

[引数]

pInBuf	元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] 入力バッファ中の元データのバイト数 (0x00000000~0x7FFFFFFF)
	[out] 入力バッファ中で使用した元データのバイト数
pnOutBuf	[in] 出力バッファのサイズ (0x00008002~0x7FFFFFFF)
	[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]

エラーフィールド (戻り値 & RELC_MEV_MASK)

RELC_RV_NORMAL	==	正常終了
RELC_EV_STOPUSER	==	処理を中断

制御フィールド (戻り値 & RELC_MCV_XXXX)

RELC_MCV_NODATA	入力データを使い切りました
RELC_MCV_NOBUFF	出力バッファを使い切りました
RELC_MCV_EDTEND	圧縮処理が終了

[説明]

pInBuf から **pnInBuf** バイトの元データを読み取り、圧縮データを **pOutBuf** へ書き込みます。関数から復帰すると、**pnInBuf** へ使用された元データのバイト数、**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。

元データを格納する入力バッファは、RELC_MAX_ORIGINAL バイト以上のサイズを用意してください。

圧縮データを格納する出力バッファは、RELC_MAX_CODE バイト以上のサイズを用意してください。

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。

7.5 relc_MI_Decode ()

[機能]

圧縮データを復元します。

[形式]

```
int relc_MI_Decode(  
  unsigned char RELC_FAR * pInBuf ,  
  unsigned char RELC_FAR * pOutBuf ,  
  long RELC_FAR * pnInLen ,  
  long RELC_FAR * pnOutLen  
)
```

[引数]

pInBuf	圧縮データの入力バッファアドレス
pOutBuf	復元データの出力バッファアドレス
pnInLen	[in] 入力バッファ中の圧縮データのバイト数 (0x00000000 ~ 0x7FFFFFFF)
	[out] 入力バッファで使った圧縮データのバイト数
pnOutLen	[in] 出力バッファサイズ (0x00008000 ~ 0x7FFFFFFF)
	[out] 出力バッファへ出力したバイト数

[戻り値]

エラーフィールド (戻り値 & RELC_MEV_MASK)

RELC_RV_NORMAL	==	正常終了
RELC_EV_STOPUSER	==	処理を中断

制御フィールド (戻り値 & RELC_MCV_XXXX)

RELC_MCV_NODATA	入力データを使い切りました
RELC_MCV_NOBUFF	出力バッファを使い切りました
RELC_MCV_EDTEND	圧縮データが終了

[説明]

pInBufから圧縮データを読み取り、復元した復元データを **pOutBuf**へ書き込みます。関数から復帰すると、**pnInLen**へ読み取った圧縮データのバイト数、**pnOutLen**へ書き込んだ復元データのバイト数が格納されています。

圧縮データを格納する入力バッファは、RELC_MAX_CODE バイト以上のサイズを用意してください。

復元データを格納する出力バッファは、RELC_MAX_ORIGINAL バイト以上のサイズを用意してください。

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の領域への操作は読み書きを行います。

7.6 relc_MI_GetAddInfoContents ()

[機能]

圧縮ストリームから付加情報の内容を得ます。

[形式]

```
int relc_MI_GetAddInfoContents (  
  unsigned char RELC_FAR *    pInBuff ,  
  long RELC_FAR *            pnInLen ,  
  int RELC_FAR *             pbAddInfo ,  
)
```

[引数]

pInBuff	圧縮データの入力バッファアドレス
pnInLen	入力バッファ中の圧縮データのバイト数
pbAddInfo	付加情報の種類

[戻り値]

エラーフィールド (戻り値 & **RELC_MEV_MASK**)

RELC_RV_NORMAL == 正常終了

RELC_EV_ILLFORM == フォーマットが不正です

制御フィールド (戻り値 & **RELC_MCV_XXXX**)

無し

[説明]

pnInBuffの圧縮データから付加情報を読み取り、**pbAddInfo**へ付加情報の種類を格納します。実際の付加情報の値を得るには、続けて**relc_MI_GetAddInfo ()**を呼び出してください。

pbAddInfoへ格納される値は、次の定数の論理和です。

RELC_KIND_NO_INFO	情報無し
RELC_KIND_ORGSIZE	元ストリームサイズ
RELC_KIND_CODESIZE	圧縮ストリームサイズ

7.7 relc_MI_GetAddInfo ()

[機能]

圧縮ストリームの付加情報を読み取ります。

[形式]

```
int relc_MI_GetAddInfo (
int                                nAddInfo ,
RELC_64VAR *                       pnAddInfo
)
```

[引数]

nAddInfo 取得する付加情報
pnAddInfo 値を格納する領域へのポインタ

[戻り値]

エラーフィールド (戻り値 & RELC_MEV_MASK)
RELC_RV_NORMAL == 正常終了
RELC_EV_ILLFORM == フォーマットエラー

制御フィールド (戻り値 & RELC_MCV_XXXX)
無し

[説明]

nAddInfo で指定された付加情報を圧縮ストリームの先頭から読み取り、**pnAddInfo** へ格納します。

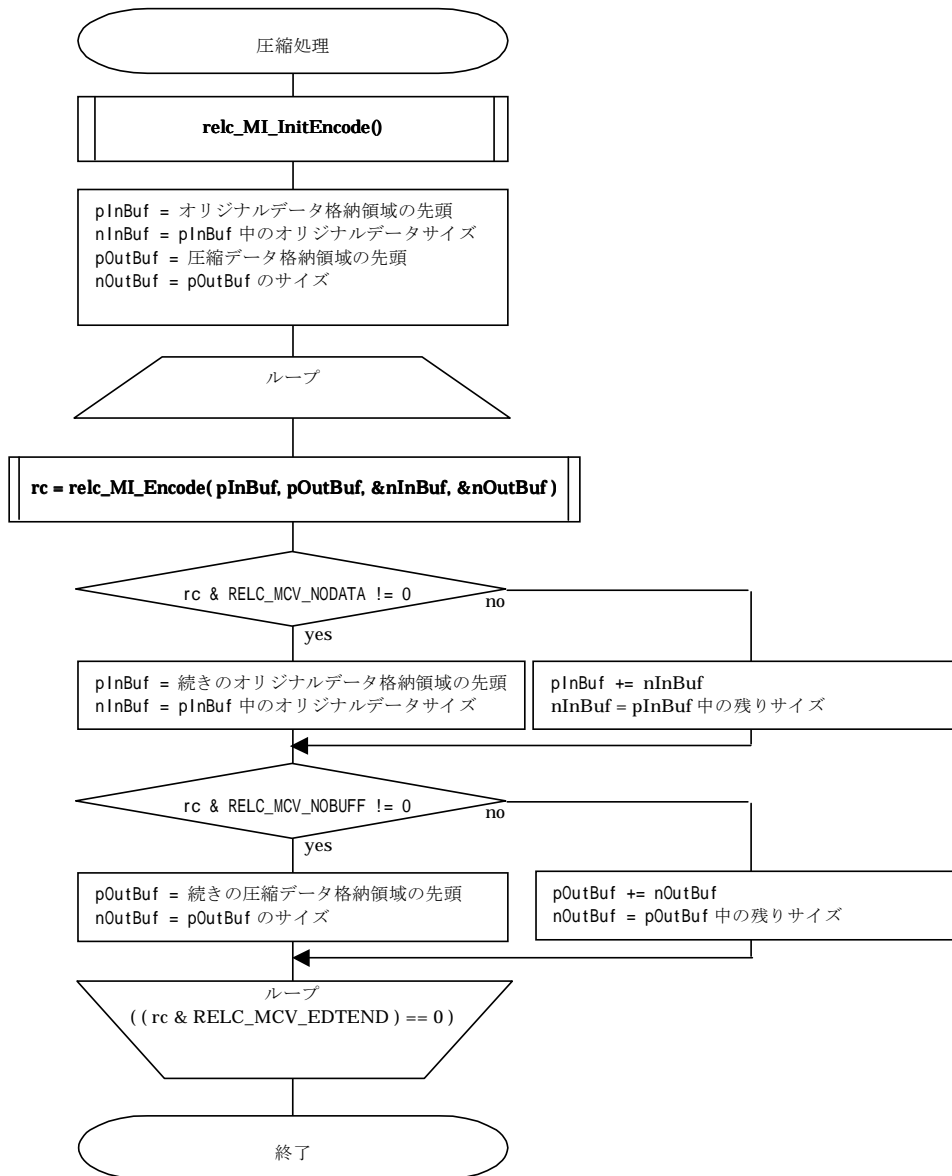
nAddInfo へは、次の値を指定します。

RELC_KIND_ORGSIZE	元ストリームサイズ
RELC_KIND_CODESIZE	圧縮ストリームサイズ

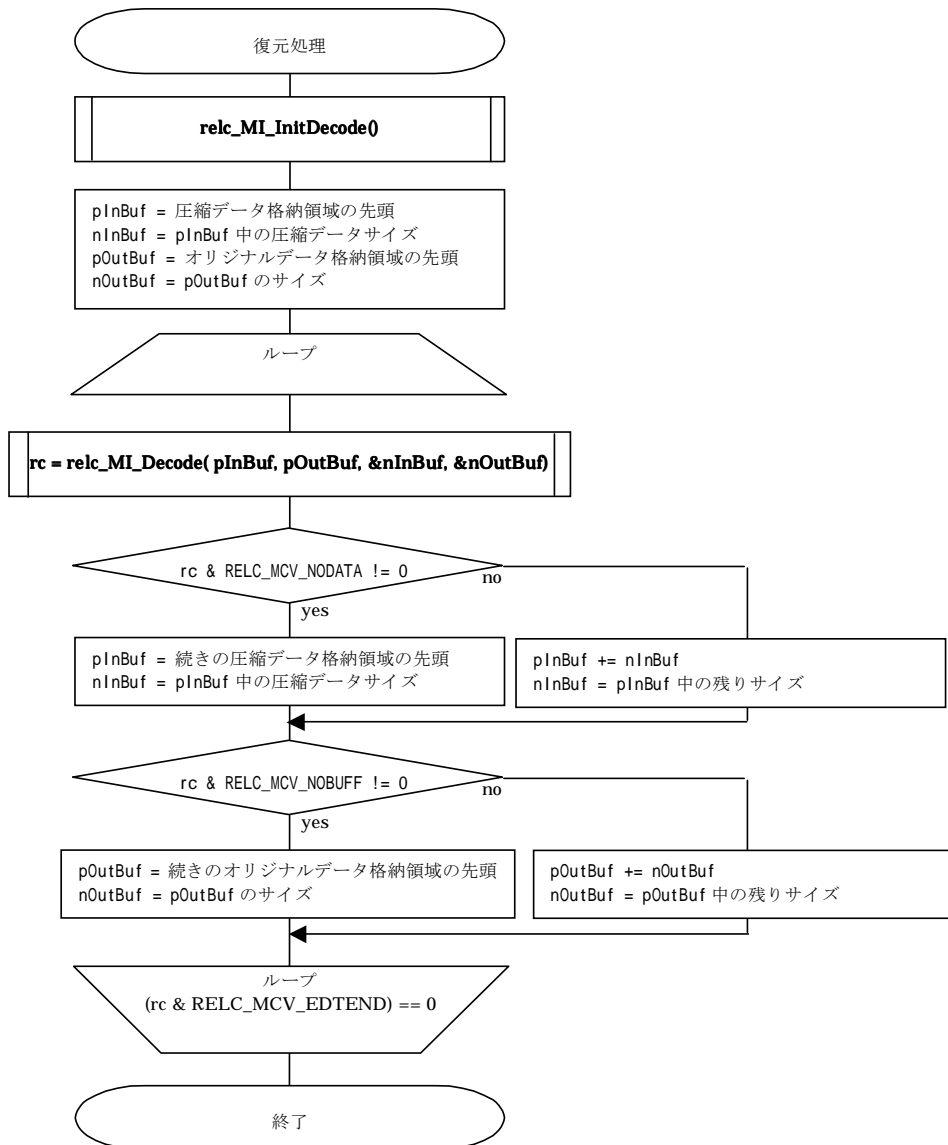
7.8 中レベル API の使用方法

このレベルでは、メモリからメモリへの圧縮および復元を提供しています。

7.8.1 圧縮処理



7.8.2 復元処理



8 低レベル API

このレベルでは、最も基本的な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回の呼出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。

以下の API を提供しています。

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

圧縮処理とは、入力アドレスが示すメモリに格納された復元データを読み出し、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復元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 ()
```

8.1 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]

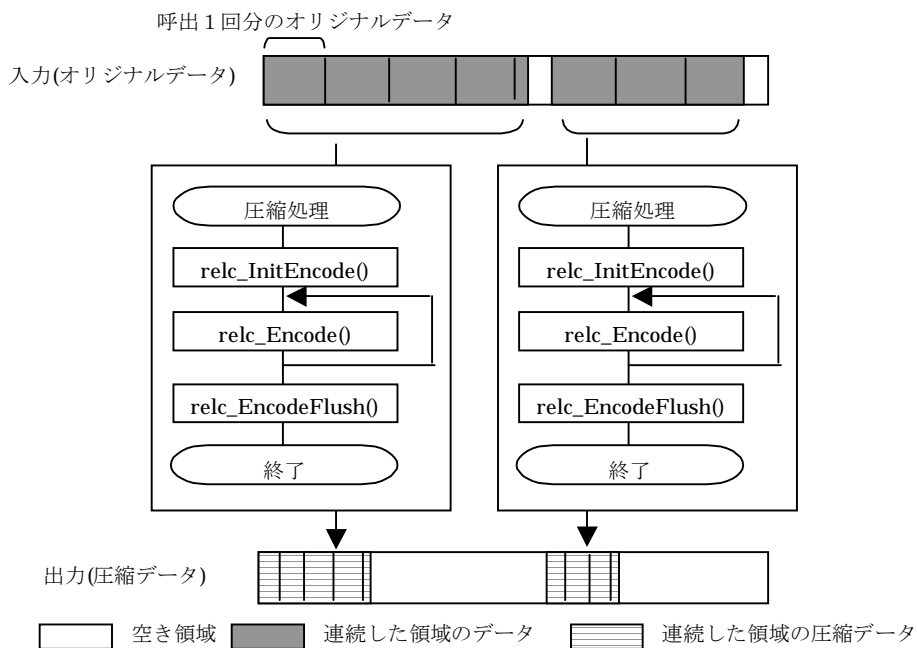
REL_CV_NORMAL == 正常終了
REL_EV_ILLPARAM == 圧縮モード指定エラー

[説明]

圧縮ストリームを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。大きな数値ほど圧縮能力は高くなり、処理速度は低くなります。圧縮モードの指定方法については、「5 API」を参照してください。

圧縮処理は、以下のようなイメージになります。



8.2 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]

なし

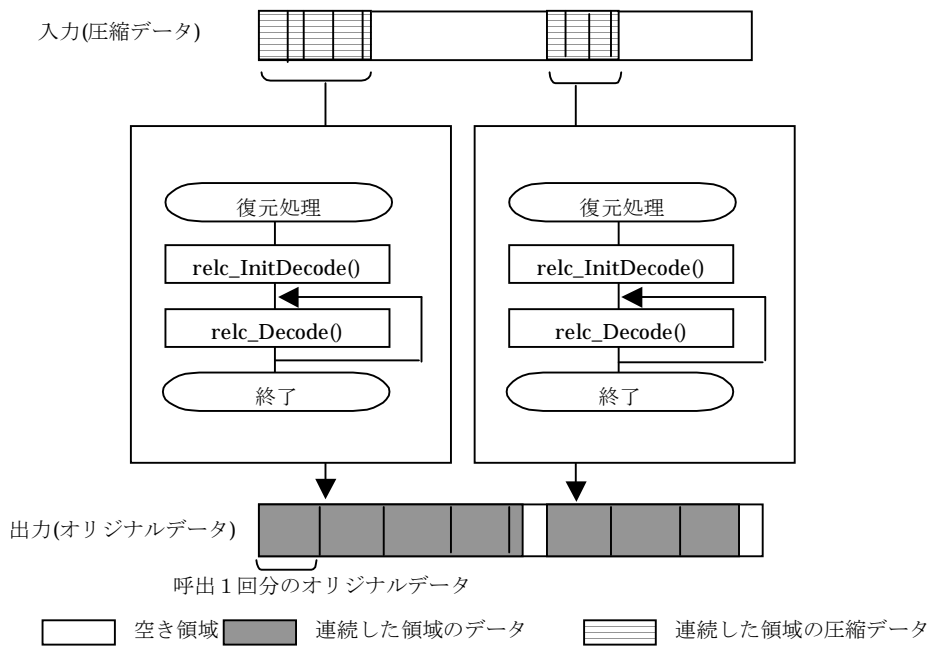
[戻り値]

RELC_RV_NORMAL == 正常終了

[説明]

圧縮ストリームを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。



8.3 relc_Encode ()

[機能]

元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	元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] 入力バッファ中の元データのバイト数
	[out] 入力バッファ中で使用した元データのバイト数
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** バイトの元データを読み取り、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**へ使用された元データのバイト数、**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 バイト以上の元ストリームを圧縮する場合は、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、残りの元ストリームのバイト数が求まります。残りの元ストリームサイズが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。

8.4 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	元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] 入力バッファ中の元データのバイト数
	[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**へ使用された元データのバイト数、**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、圧縮ストリームの末尾となる終端コードが出力されます。

8.5 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	復元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、復元した復元データを **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**へ復元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。

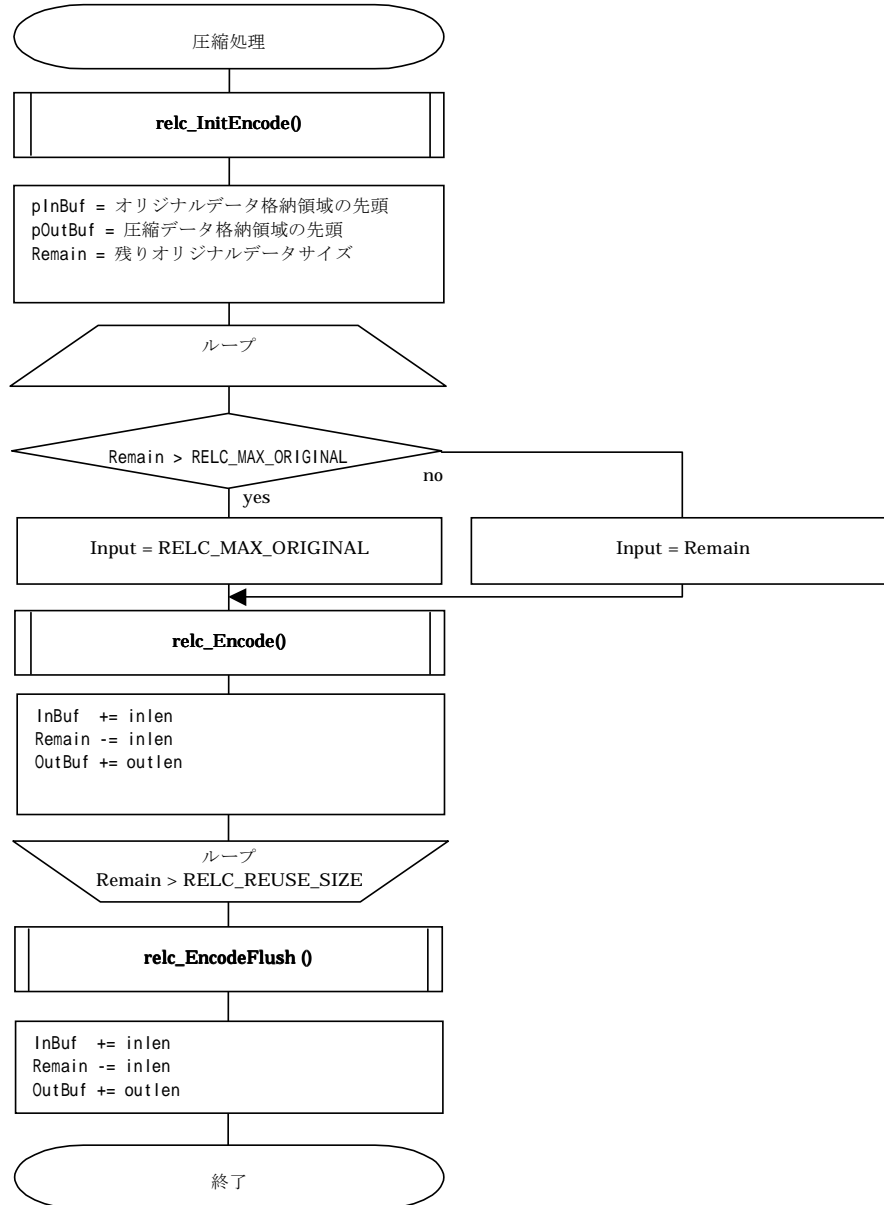
全ての圧縮ストリームの復元を行うと戻り値としてRELC_RV_ENDEDATA を返します。

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。

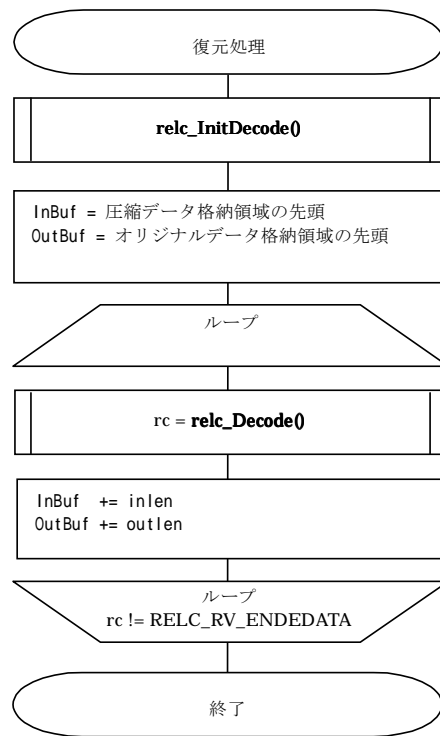
この関数では、処理速度を優先するため厳密なエラーチェックは行いません。従って、圧縮データの正当性はユーザー側で保障してください。

8.6 低レベル API の使用方法

8.6.1 圧縮処理



8.6.2 復元処理



9 定義

本製品の定義について説明します。定義は、`relc.h`、`relcm.h`、`relch.h`、`relctype.h`で記述されています。

9.1 定数

マクロ	値 ⁽¹⁰⁾	説明
<code>RELC_MAX_ORIGINAL</code>	32768	呼び出し1回毎の最大元データサイズ
<code>RELC_MAX_CODE</code>	32770	呼び出し1回毎の最大圧縮データサイズ
<code>RELC_REUSE_SIZE</code>	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
<code>RELC_MEV_MASK</code>	255	中レベルAPI用のエラー取得マスク値
<code>RELC_MODE_MAXIMUM</code>	0	最高圧縮モード
<code>RELC_MODE_HIGH</code>	32	高圧縮モード
<code>RELC_MODE_MIDDLE</code>	16	中圧縮モード
<code>RELC_MODE_LOW</code>	4	低圧縮モード
<code>RELC_MODE_MINIMUM</code>	1	最低圧縮モード
<code>RELC_KIND_NO_INFO</code>	1	ヘッダー情報無し
<code>RELC_KIND_ZERO</code>	2	付加情報を付加しない
<code>RELC_KIND_ORGSIZE</code>	4	元ストリーム付加
<code>RELC_KIND_CODESIZE</code>	8	圧縮ストリーム付加
<code>RELC_RV_NORMAL</code>	0	正常終了時の戻り値
<code>RELC_RV_ENDEDATA</code>	1	復元終了時の戻り値
<code>RELC_RV_CONTINUE</code>	2	関数を呼び出しの継続要求
<code>RELC_EV_ILLPARAM</code>	10	パラメータ異常
<code>RELC_EV_ILLDATA</code>	11	異常データ検出
<code>RELC_EV_ILLFORM</code>	12	フォーマットが異常
<code>RELC_EV_STOPUSER</code>	13	要求による中断
<code>RELC_EV_CANTOPEN</code>	20	ファイルオープンエラー
<code>RELC_EV_FILEREAD</code>	21	ファイル読み取りエラー
<code>RELC_EV_FILEWRITE</code>	22	ファイル書き込みエラー
<code>RELC_EV_CLOSE</code>	23	ファイルのクローズエラー
<code>RELC_EV_NODATA</code>	30	データ不足
<code>RELC_EV_NOBUFF</code>	31	バッファ不足
<code>RELC_MCV_NODATA</code>	256	入力データが無い
<code>RELC_MCV_NOBUFF</code>	512	出力バッファが無い
<code>RELC_MCV_EDTEND</code>	1024	全処理を終了

9.2 型

型名	説明
<code>RELC_64VAR</code>	64ビットの符号付き整数型

10 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_RV_CONTINUE	継続して API を呼び出してください。
RELC_EV_ILLPARAM	パラメータが異常です。
RELC_EV_ILLDATA	圧縮データが異常です。またはフラッシュ動作の呼び出しタイミングが違います。
RELC_EV_ILLFORM	フォーマットが異常です。
RELC_EV_STOPUSER	要求により処理を中断しました。
RELC_EV_CANTOPEN	ファイルオープンに失敗しました。
RELC_EV_FILEREAD	ファイル読み取りに失敗しました。
RELC_EV_CLOSE	ファイルのクローズでエラーがありました。
RELC_EV_FILEWRITE	ファイル書き込みに失敗しました。

11 制限事項

本製品には、以下の制限事項があります。

- ▶ マルチスレッドアプリケーションでの使用について
本ライブラリは、マルチスレッドに対応しておりません。マルチスレッドなアプリケーションから本ライブラリを使用する場合は、アプリケーション側で複数のスレッドが同時に本ライブラリを使用しないように排他制御してください。

- ▶ 異常な圧縮ストリームの復元処理の禁止
本ライブラリでは、復元速度を優先するため、圧縮ストリームについて厳密なチェックを行いません。データ化けや欠落などの異常が生じた圧縮ストリームを復元 API に処理させないでください。

12 用語集

- 呼び出し元
本ライブラリを使用しているプログラム。
- 圧縮モード
圧縮処理で使用される探索レベルを表す値。
- 元ストリーム
圧縮処理の入力となる一連のデータ。
- 圧縮ストリーム
圧縮処理の出力、復元処理の入力となる一連のデータ。
- 復元ストリーム
復元処理の出力となる一連のデータ。内容は元ストリームと同じ。
- 元データ
元ストリームの一部のデータ。
- 圧縮データ
圧縮ストリームの一部のデータ。
- 復元データ
復元ストリームの一部のデータ。

13 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 (RelcEva.exe) が添付されています。このプログラムは、評価用のみ利用できます。

(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> RelcEva.exe option infile outfile [mode]

option 書式 {E [{Z | OC}] | D [H]}¹

圧縮 圧縮処理の場合は、'E' を指定します。
圧縮形式を指定する場合は、'E' に続けてオプション 1 を記述します。

オプション 1	説明
Z	RELC_KIND_ZERO 形式の圧縮ファイルを作成します。
O	RELC_KIND_ORGSIZE 形式の圧縮ファイルを作成します。
C	RELC_KIND_CODESIZE 形式の圧縮ファイルを作成します。

オプション 1 を省略した場合は、RELC_KIND_NO_INFO 形式の圧縮ファイルを作成します。

'O' と 'C' は併記することも可能です。その場合、RELC_KIND_ORGSIZE | RELC_KIND_CODESIZE 形式の圧縮ファイルを作成します。

復元 復元処理の場合は、'D' を指定します。
圧縮形式を指定する場合は、'D' に続けてオプション 2 を記述します。

オプション 2	説明
H	ヘッダー付き圧縮ファイルを復元します。

オプション 2 を省略した場合は、ヘッダー無しの圧縮ファイルを復元します。

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)
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
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 大文字小文字の区別はありません。また、記号の意味は次のとおりです。

[...] は、その中の要素は記述が省略可能であることを示します

{.|.} は、その中の要素内で一つを選択することを示します

FUJITSU