イメージ・アルファ画像処理ライブラリ
ＱＲコード処理マニュアル

第 1.01 版

株式会社　イメージ・アルファ

はじめに

本ドキュメントは、イメージ・アルファ画像処理ライブラリ IALIB のＱＲコード復号/符号機能について解説しています。

ＱＲコードの位置計測機能(QRGuide)については、QRGuide マニュアルを参照してください。

表 0-1　本ドキュメントの適用 IALIB バージョン

ソフト名	バージョン番号
IALIB	Version 1.2.2

表 0-2　ＱＲコード復号/符号・修正履歴

バージョン番号	分類	修正内容
version 1.2.0	【機能拡張】 復号	復号処理を新規追加。(3.2 節)
Version 1.2.1	【機能拡張】 符号	符号処理を新規追加。(3.3 節)

目次

1. 機能

IALIB では、ＱＲコードを復号/符号する関数を用意しています。

項目	対応範囲
モデル	[モデル１]、モデル２、マイクロＱＲ
型番	全て
モード	数字、英数字、漢字、8 ビットバイトデータ、混在、[構造的連接]

表 1-1　対応ＱＲコード（ []内は復号には対応、符号には非対応 ）

かざし読み　　一部に影　　円柱の側面

図 1-1　復号できるＱＲコード例

2. 使用方法

下のサンプルコードは、BMP ファイルから画像を読み込み、その画像にあるＱＲコードを復号し、
メッセージボックスで結果を表示するアプリケーションです。

```
int APIENTRY WinMain( HINSTANCE hInstance, HINSTANCE hPrevInstance,  LPSTR lpCmdLine, int nCmdShow )
{
        int i;
        int imgID;
        IA_Size imgSize;
        int qrNum;
        IA_QR_Decode_CtlExt ctl;
        IA_QR_Decode_Rslt rslt;
        char qrStr[IA_QR_MAXSTRBUFF];
        char caption[100];

        // IALIB・有効化
        IA_Open();                                                          → 2.1.1節

        // IALIB・ＱＲコード復号機能・有効化
        IA_QR_Open();                                                       → 2.1.1節

        // 処理対象画面確保
        imgSize.xSize = 640;
        imgSize.ySize = 480;
        imgID = IA_AllocImg( IA_IMG_CHAR, &imgSize );

        // BMPファイルを読み込み処理対象画面へ格納
        IA_BMPFile_Load( imgID, "QRImg.bmp" );

        // 制御パラメータ設定
        IA_QR_GetInitCtl( &ctl );               // ctlを初期化
        ctl.fMicroQR = 1;                       // マイクロＱＲコードも復号
        ctl.maxNum   = IA_QR_MAXQRNUM;          // 複数のＱＲコードを復号
        IA_QR_SetCtl( &ctl );                   // 設定                     → 2.1.2節

        // 復号
        qrNum = IA_QR_Decode( imgID, IA_QR_MAXSTRBUFF, qrStr );             → 2.1.3節

                                ⋮
```

⋮

```
	// 復号文字列のﾒｯｾｰｼﾞ ﾎﾞｯｸｽによる表示
	if( qrNum > 0 ){
		for( i=0; i<qrNum; i++ ){
			IA_QR_GetRslt( i, &rslt );                    → 2.1.4節
			sprintf( caption, "%d番目復号文字列", i );
			MessageBox( NULL, rslt.str, caption, MB_OK );
		}
	}
	else if( qrNum == 0 ){
		MessageBox( NULL, "復号失敗(検知数)", "復号失敗(検知数)", MB_OK );
	}
	else{
		MessageBox( NULL, "エラー", "エラー", MB_OK );
	}

	// IALIB・ＱＲコード復号機能・無効化
	IA_QR_Close();                                        → 2.1.1節

	// IALIB・無効化
	IA_Close();                                           → 2.1.1節

	return 0;
}
```

2.1.1. 各種有効化

IALIB全体の有効化、ＱＲコード復号/符号機能の有効化を最初に行います。無効化されるまで機能が使用できます。

有効時には何回でも復号/符号が行えます。復号/符号のたびに有効化をする必要はありません。

2.1.2. 制御パラメータの設定

構造体 IA_QR_Decode_CtlExt(4.6 節)の変数を初期化して、環境に合わせてカスタマイズし、
関数 IA_QR_SetCtl(3.2.1 節)の引数とします。

2.1.3. 復号

関数 IA_QR_Decode (3.2.3 節)により、制御パラメータ設定値に基づいた復号を行います。

2.1.4. 復号結果の取得

関数 IA_QR_GetRslt(3.2.4 節)によりＱＲコード１つ分の復号結果を取得できます。
ＱＲコードの指定にはＱＲコード ID を使用します。
ＱＲコード ID は検知したＱＲコードに割り振られた番号で、0 以上検知数未満の整数値です。

3. 関数

本章では関数インターフェースを記しています。
関連する列挙体、構造体は４章を参照してください。

区分	節番号	関数名	機能
共通	3.1.1節	IA_QR_Open	有効化
	3.1.2節	IA_QR_Close	無効化
復号	3.2.1節	IA_QR_SetCtl	制御パラメータの設定
	3.2.2節	IA_QR_GetInitCtl	制御パラメータのデフォルト値を取得
	3.2.3節	IA_QR_Decode	復号
	3.2.4節	IA_QR_GetRslt	結果取得（復号結果・詳細）
	3.2.4節	IA_QR_GetRslt	結果取得（復号結果・モジュール毎情報）
	3.2.6節	IA_QR_SetPointsOnView	モジュール毎情報の IA_View 表示
符号	3.3.1節	IA_QR_Encode	符号
	3.3.2節	IA_QR_DrawEncodeQR	符号したＱＲコードの描画
	3.3.3節	IA_QR_GetRslt	結果取得（符号結果）

表 3-1　ＱＲコード復号/符号一覧

3.1. 有効/無効

3.1.1. 有効化 [IA_QR_Open]

インタフェース

```
int IA_QR_Open ();
```

リターン値

基本エラーコード

機能

ＱＲコード復号/符号機能を有効にします。

本関数の呼び出しは、IALIB を有効化した後(つまり関数 IA_Open を呼んだ後)に行ってください。

3.1.2. 無効化 [IA_QR_Close]

インタフェース

```
int IA_QR_Close ();
```

リターン値

基本エラーコード

機能

ＱＲコード復号/符号機能を無効にします。

IALIB 内部にて確保したメモリの開放などの後処理を行いますので、
有効にした場合には本関数を必ず呼び出してください。

本関数の呼び出しは、IALIB を無効化する前(つまり関数 IA_Close を呼ぶ前)に行ってください。

3.2. 復号

3.2.1. 制御パラメータの設定 [IA_QR_SetCtl]

インタフェース

```
int IA_QR_SetCtl (
        IA_QR_Decode_Ctl    *pCtl
    );
int IA_QR_SetCtl (
        IA_QR_Decode_CtlExt *pCtl
    );
```

パラメータ

名称	タイプ	入/出	許容値	意味
pCtl	IA_QR_Decode_Ctl* (4.5節)	入力		制御パラメータ(基本)
pCtl	IA_QR_Decode_CtlExt* (4.6節)	入力		制御パラメータ(詳細)

リターン値

基本エラーコード

機能

制御パラメータを設定します。この設定値で復号(3.2.3節)が行われます。

本関数を呼び出さずに復号する場合はデフォルト値(表 3-2)で動作します。

インターフェースを２つ用意しています。
引数である構造体 IA_QR_Decode_Ctl は構造体 IA_QR_Decode_CtlExt の基本的なメンバを抽出したものとなっています。
IA_QR_Decode_Ctl にないメンバはデフォルト値となります。

3.2.2. 制御パラメータデフォルト値の取得 [IA_QR_GetInitCtl]

インタフェース

```
int IA_QR_GetInitCtl (
        IA_QR_Decode_Ctl    *pCtl
    );
int IA_QR_GetInitCtl (
        IA_QR_Decode_CtlExt *pCtl
    );
```

パラメータ

名称	タイプ	入/出	許容値	意味
pCtl	IA_QR_Decode_Ctl* (4.5節)	出力		制御パラメータデフォルト値(基本)
pCtl	IA_QR_Decode_CtlExt* (4.6節)	出力		制御パラメータデフォルト値(詳細)

リターン値

基本エラーコード

機能

制御パラメータのデフォルト値(表 3-2)を引数の構造体に設定します。
主に構造体の初期化に使用します。

変数名	値	説明
colorOrder	IA_QR_COLORORDER_BLACK	-
moduleRange.min	2	-
moduleRange.max	14	-
fModel1	0	モデル２のみ復号
fModel2	1	
fMicroQR	0	
maxNum	1	ＱＲコードを１つ復号
fStructedAppend	0	構造的連接に関係なく復号
parity	-1	
total	1	
fOrder	すべて0	

表 3-2 制御パラメータデフォルト値(初期値)

3.2.3. 復号 [IA_QR_Decode]

インタフェース

```
int IA_QR_Decode (
        int imgID, int strBuffSize, char str[]
    );
```

パラメータ

名称	タイプ	入/出	許容値	意味
imgID	int	入力		処理対象画像
strBuffSize	int	入力		復号文字列格納領域のバイト数
str	char[]	出力		復号文字列

リターン値

0 以上 : 検知ＱＲコード数
0 未満 : 基本エラーコード

機能

画像からＱＲコードを検知、復号します。

格納領域 str には１つのＱＲコードの復号文字列を格納します。
制御パラメータにて複数のＱＲコードを復号するよう設定した場合は、最初に検知したほうが出力されます。
他のＱＲコードについては関数 IA_QR_GetRslt(3.2.4 節)により文字列を取得できます。

復号文字列が格納領域 str に収まらない場合(文字列バイト数≧strBuffSize)には、エラーとなります。

3.2.4. 結果取得（復号結果・詳細）[IA_QR_GetRslt]

インタフェース

```
int IA_QR_GetRslt (
        int qrID, IA_QR_Decode_Rslt *pRslt
    );
```

パラメータ

名称	タイプ	入/出	許容値	意味
qrID	int	入力	0～検知数-1	ＱＲコード ID
pRslt	IA_QR_Decode_Rslt* (4.7節)	出力		復号結果・詳細

リターン値

基本エラーコード

機能

検知したＱＲコードの各種情報を取得します。
モジュール毎の情報に関しては、3.2.5 節の関数にて取得します。

ＱＲコード ID とは検知したＱＲコード１つずつに割り振られた番号です。
最初に検知したＱＲコードは 0、最後に検知したＱＲコードは検知数-1 となります。

本関数は直前の関数 IA_QR_Decode(3.2.3 節)に対する処理結果を取得するものであり、
２つ前の IA_QR_Decode に対する処理結果を取得することはできません。
ＱＲコード ID は IA_QR_Decode を呼び出すたびにリセットされます。

3.2.5. 結果取得（復号結果・モジュール毎情報）[IA_QR_GetRslt]

インタフェース

```
int IA_QR_GetRslt (
        int qrID, IA_QR_Points *pPoints
    );
```

パラメータ

名称	タイプ	入/出	許容値	意味
qrID	int	入力	0～検知数-1	ＱＲコード ID
pPoints	IA_QR_Points* (4.8 節)	出力		復号結果・モジュール毎情報

リターン値

基本エラーコード

機能

検知したＱＲコードのモジュール毎の情報を取得します。
ＱＲコード全体に関する情報については、3.2.4 節の関数にて取得します。

ＱＲコード ID とは検知したＱＲコード１つずつに割り振られた番号です。
最初に検知したＱＲコードは 0、最後に検知したＱＲコードは検知数-1 となります。

本関数は直前の関数 IA_QR_Decode(3.2.3 節)に対する処理結果を取得するものであり、
２つ前の IA_QR_Decode に対する処理結果を取得することはできません。
ＱＲコード ID は IA_QR_Decode を呼び出すたびにリセットされます。

3.2.6. モジュール毎情報の IA_View 表示 [IA_QR_SetPointsOnView]

インタフェース

```
int IA_QR_SetPointsOnView (
        int imgID, int qrID
    );
```

パラメータ

名称	タイプ	入/出	許容値	意味
imgID	int	入力		対象画像の画面 ID
qrID	int	入力	0～検知数-1	ＱＲコード ID

リターン値

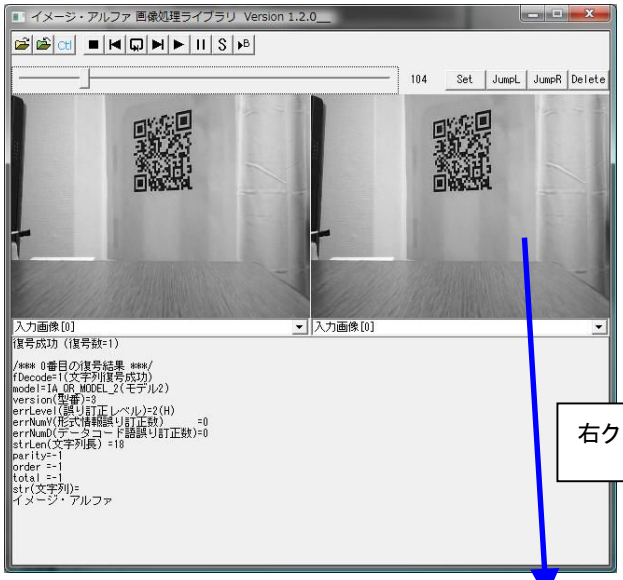
基本エラーコード

機能

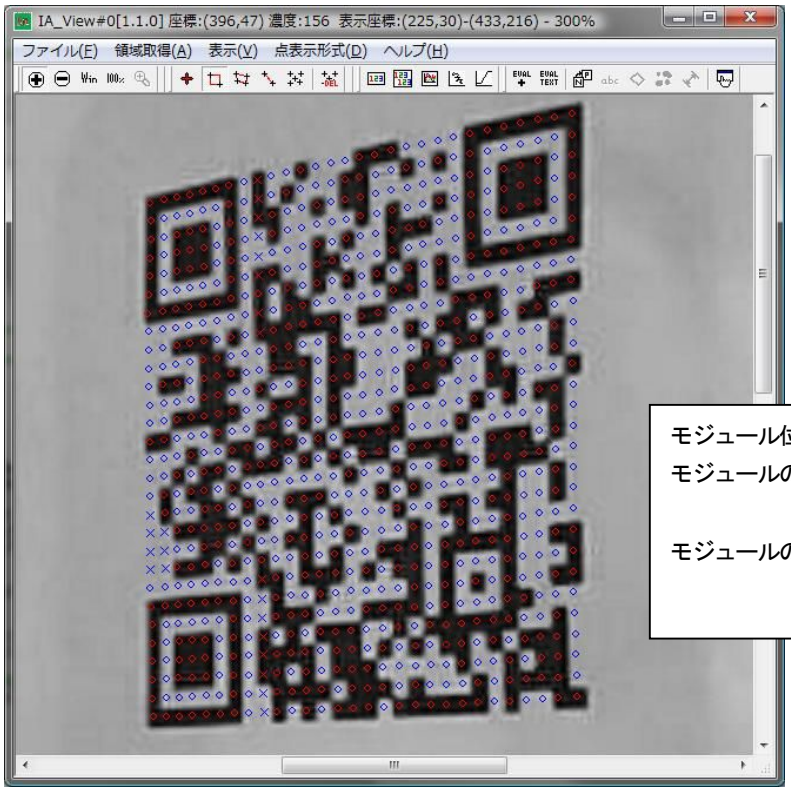
指定した画像が IA_View に表示されている場合に、
モジュール毎情報(モジュールの位置、白黒、エラー)を上書き表示します。

ＱＲコード ID とは検知したＱＲコード１つずつに割り振られた番号です。
最初に検知したＱＲコードは 0、最後に検知したＱＲコードは検知数-1 となります。

Eval アプリケーションでのみ使用可能です。



右クリックで IA_View 表示



モジュール位置にプロットされます。
モジュールの色　：黒と認識なら赤
　　白と認識なら青
モジュールのエラー　：エラー無しなら○
　　エラー有り or 未確認なら×

3.3. 符号

3.3.1. 符号 [IA_QR_Encode]

インタフェース

```
int IA_QR_Encode (
        char str[], IA_QR_Model model, int version, IA_QR_ErrorLevel errLevel
    );
```

パラメータ

名称	タイプ	入/出	許容値	意味
str	char[]	入力		文字列
model	IA_QR_Model (4.1節)	入力	機能欄参照	モデル
version	int	入力	機能欄参照	型番
errLevel	IA_QR_ErrorLevel (4.2節)	入力	機能欄参照	誤り訂正レベル

リターン値

1以上 : 符号化された文字列のバイト数
0未満 : 基本エラーコード

機能

文字列をＱＲコードに符号します。
このＱＲコードを描画するには関数IA_QR_DrawEncodeQR(3.3.2節)を呼び出してください。

文字列が長すぎる場合でも、文字列を切り捨てて符号します。
リターン値により切り捨てが行われたか否かを確認できます。

モデルは、モデル２またはマイクロＱＲを指定してください。モデル１には対応していません。

モデル２の型番は1～40、マイクロＱＲの型番は1～4を指定できます。

引数の組み合わせ方によってはエラーとなることがあります。
例えば、数字以外の文字を含む文字列を、マイクロＱＲ、型番１、誤り訂正レベルＨで符号することはできません。

3.3.2. 符号したＱＲコードの描画 [IA_QR_DrawEncodeQR]

インタフェース

```
int IA_QR_DrawEncodeQR (
        int imgID, IA_Point *pPosLT, int moduleSize, int quietSize
    );
```

パラメータ

名称	タイプ	入/出	許容値	意味
imgID	int	出力		画像
pPosLT	IA_Point*	入力		ＱＲコード左上の画像座標 (クワイエットゾーン含む)
moduleSize	int	入力	1 以上	１モジュールの幅［画素］
quietSize	int	入力	0 以上	クワイエットゾーン(周辺の余白)の幅［画素］

リターン値

基本エラーコード

機能

符号したＱＲコードを指定画面に描画します。

ＱＲコード全体が描画できない場合はエラーとなります。
エラーの場合は、画面サイズを大きくするか、左上の座標や各種幅を調節してください。

3.3.3. 結果取得（符号結果）[IA_QR_GetRslt]

インタフェース

```
int IA_QR_GetRslt (
        IA_QR_Encode_Rslt *pRslt
    );
```

パラメータ

名称	タイプ	入/出	許容値	意味
pRslt	IA_QR_Encode_Rslt (4.9節)	出力		符号結果

リターン値

基本エラーコード

機能

符号したＱＲコードの各種情報を取得します。

4. 列挙体、構造体

4.1. ＱＲコードモデル

```
typedef enum {
    IA_QR_MODEL_1,          // モデル
    IA_QR_MODEL_2,          // モデル
    IA_QR_MODEL_MICRO,      // マイクロQR
    IA_QR_MODEL_END
} IA_QR_Model;
```

ＱＲコードのモデルを表します。

4.2. ＱＲコード誤り訂正レベル

```
typedef enum {
    IA_QR_ERRORLEVEL_L,             // L ( 7%)
    IA_QR_ERRORLEVEL_M,             // M (15%)
    IA_QR_ERRORLEVEL_Q,             // Q (25%)
    IA_QR_ERRORLEVEL_H,             // H (30%)
    IA_QR_ERRORLEVEL_END
} IA_QR_ErrorLevel;
```

ＱＲコードの誤り訂正レベルを表します。

4.3. 色に関する復号順

```
typedef enum {
    IA_QR_COLORORDER_BLACK,          // 黒(通常の QR コード)
    IA_QR_COLORORDER_WHITE,          // 白
    IA_QR_COLORORDER_BLACK_WHITE,    // 黒→白(黒で復号失敗したら、白で再度試みる)
    IA_QR_COLORORDER_WHITE_BLACK,    // 白→黒(白で復号失敗したら、黒で再度試みる)
    IA_QR_COLORORDER_END
} IA_QR_ColorOrder;
```

色に関する復号順を表します。
構造体 IA_QR_Decode_Ctl(4.5 節)、IA_QR_Decode_CtlExt(4.6 節)のメンバとなっています。
IA_QR_COLORORDER_WHITE は、黒い背景にある白いＱＲコードを復号する際に使用します。

4.4. ＱＲコードのモジュールサイズ

```
typedef struct {
    int min;
    int max;
} IA_QR_ModuleRange;
```

ＱＲコードのモジュールサイズの範囲を格納します。単位は画素。
構造体 IA_QR_Decode_Ctl(4.5 節)、IA_QR_Decode_CtlExt(4.6 節)のメンバとなっています。
この範囲を目安に復号が行われます。

4.5. 復号制御パラメータ・基本

```
typedef struct {
    IA_QR_ColorOrder  colorOrder;       // 色に関する復号順
    IA_QR_ModuleRange moduleRange;      // モジュールサイズの目安
    int               fModel1;          // モデル１有効フラグ
    int               fModel2;          // モデル２有効フラグ
    int               fMicroQR;         // マイクロＱＲ有効フラグ
} IA_QR_Decode_Ctl;
```

復号時の基本的な制御パラメータを格納します。
構造体 IA_QR_Decode_CtlExt(4.6 節)から基本的なメンバが抽出されています。

4.6. 復号制御パラメータ・詳細

```
typedef struct {
    IA_QR_ColorOrder  colorOrder;       // 色に関する復号順
    IA_QR_ModuleRange moduleRange;      // モジュールサイズの目安
    int               fModel1;          // モデル１有効フラグ
    int               fModel2;          // モデル２有効フラグ
    int               fMicroQR;         // マイクロＱＲ有効フラグ
    int               maxNum;           // 最大復号ＱＲコード数
    // 構造的連接
    int               fStructedAppend;  // 1：構造的連接ＱＲコードのみ読み取る
    int               parity;           // 0～：パリティ -1：指定なし
    int               total;            // 1～：連結総数-1：指定なし
    char              fOrder[16];       // fOrder[i]==1 なら i 番目に連結されるＱＲコード
                                        // の復号をおこなう
} IA_QR_Decode_CtlExt;
```

復号時の制御パラメータを格納します。

4.7. 復号結果・詳細

```
typedef struct {
    // 復号成否
    int             fDecode;            // 1：文字列への復号成功
                                        // 0：文字列への復号失敗(未対応モード)
    // 各種情報
    IA_QR_Model     model;              // モデル
    int             version;            // 型番
    int             moduleNum;          // １辺あたりのモジュール数
    IA_QR_ErrorLevel errLevel;          // 誤り訂正レベル
    int             errNumV;            // 形式情報復号における、誤り訂正数
    int             errNumD;            // データコード語復号における、誤り訂正数
    // 文字列(fDecode==0 の場合は空文字列)
    int             strLng;             // 文字列長(バイト数)。終端の NULL は含めない
    char            str[IA_QR_MAXSTRBUFF];  // 文字列
    // 構造的連接
    int             parity;             // 0～：パリティ,  -1：非連結
    int             order;              // 0～：連結順番,  -1：非連結
    int             total;              // 1～：連結総数, -1：非連結
} IA_QR_Decode_Rslt;
```

復号における詳細な処理結果を格納します。
関数 IA_QR_GetRslt(3.2.4 節)より出力されます。

IALIB バージョン 1.2.1 から誤り訂正レベルのメンバ errLevel(誤り訂正レベル)は
char 型から IA_QR_ErrorLevel 型(4.2 節)へ変更されました。

4.8. 復号結果・モジュール毎情報

```
typedef struct {
	IA_PointF	pos [IA_QR_MAXMODULE][IA_QR_MAXMODULE];	// 復号・位置座標
	char		data[IA_QR_MAXMODULE][IA_QR_MAXMODULE];	// 復号・白黒
								//	0 : 白
								//	1 : 黒
	char		err [IA_QR_MAXMODULE][IA_QR_MAXMODULE];	// 復号・エラー
								//	0 : 無し
								//	1 : 有り or 未確認
	char		type[IA_QR_MAXMODULE][IA_QR_MAXMODULE];
					// 位置計測・使用状況（使用しません）
					//	0 : 位置計測に使用しなかった
					//	1 : 位置計測に使用した
					//	2 : 位置計測に使用したかつ孤立モジュール
} IA_QR_Points;
```

復号におけるモジュール毎の情報を格納します。
関数 IA_QR_GetRslt(3.2.5 節)より出力されます。

この位置座標の精度は雑になっています。
これは復号に成功した時点で処理を終了しており、位置座標の高精度化は行われないことによります。
同様に処理に無関係だったモジュールについては、エラーの有無を特定していません。
エラーでなくても err[i][j]==1 となることはあります。

4.9. 符号結果

```
typedef struct {

    char            str[IA_QR_MAXSTRBUFF];                  // 文字列
    unsigned char   dcode[IA_QR_MAXDCODE];                  // データコード語
    unsigned char   socode[IA_QR_MAXSOCODE];                // 総コード語
    char            matrix[IA_QR_MAXMODULE][IA_QR_MAXMODULE];// ＱＲコード

    int             strN;                   // 文字列のバイト数
    int             dcN;                    // データコード語数
    int             scN;                    // 総コード数

    IA_QR_Model     model;                  // モデル
    int             size;                   // １行辺りのモジュール数
    IA_QR_ErrorLevel errLevel;              // 誤り訂正レベル

    int             rsN;                    // RS ブロック数
    int             rsC[IA_QR_MAXRSBLOCK];  // RS ブロック毎の総コード数
    int             rsK[IA_QR_MAXRSBLOCK];  // RS ブロック毎のデータコード数
    int             rsR[IA_QR_MAXRSBLOCK];  // RS ブロック毎の誤り訂正可能数

    int             mask;                   // マスクパターン

    int             sizeB;
      // 最適な１行辺りのセル数
      //  sizeB>size  ：文字列がsizeに対して大きすぎる。文字列の一部を符号化した。
      //              さらに、sizeB>MAXVERSIONなら、ＱＲコードの上限として全文字列の符号化はできない。
      //  sizeB=size  ：文字列はsizeに対して適当な大きさ。
      //  sizeB<size  ：文字列はsizeに対して小さすぎる。
    int             strNB;
      // ＱＲコードに変換した文字列のバイト数。(≦strN)
      //  strNB=strN  ：文字列のすべてを符号化した
      //  strNB<strN  ：文字列の一部をを符号化した
} IA_QR_Encode_Rslt;
```

符号結果を格納します。
関数 IA_QR_GetRslt(3.3.3 節)より出力されます。