TK-850/JH3U-SP

グラフィック・ライブラリ

ユーザーズ・マニュアル

（第 1.0 版）

テセラ･テクノロジー株式会社

目次

図表の目次

第1章　グラフィック・ライブラリの概要

1.1. 概説

本グラフィック･ライブラリは､LCD コントローラを接続してグラフィックの表示が可能な V850 マイコン搭載のボードに対して図形の描画や GUI 等のグラフィック機能を提供する汎用ライブラリです。以下のグラフィック機能を実装しています。

表 1-1　実装機能一覧

種類	機能
図形描画	直線の描画
	長方形の描画
	長方形の描画と塗りつぶし
	円の描画
	円の描画と塗りつぶし
	楕円の描画
	楕円の描画と塗りつぶし
	円弧の描画
	円弧の描画と塗りつぶし
	領域の塗りつぶし
	文字の描画
	文字列の描画
	画像の描画
GUI	コマンド・ボタン
	リスト・ボックス
	チェック・ボックス
	オプション・ボタン
	ラベル
	フレーム

1.2. ファイル構成

グラフィック・ライブラリのファイル構成を示します。

(1) TK850¥JH3U_graphic¥bin

Windows で実行可能なユーティリティ・ツールを格納するフォルダです。
以下のファイルが格納されています。

表 1-2 ユーティリティ・ツール一覧

ファイル名	説明
Gbmp2c.exe	ビットマップ変換ツールです。

(2) TK850¥JH3U_graphic¥include

グラフィック・ライブラリのヘッダ・ファイルを格納するフォルダです。
以下のファイルが格納されています。

表 1-3 ヘッダ・ファイル一覧

ファイル名	説明
graphic.h	グラフィック・ライブラリのヘッダ・ファイルです。

(3) TK850¥JH3U_graphic¥lib

グラフィック・ライブラリを格納するフォルダです。
以下のファイルが格納されています。

表 1-4 ライブラリ一覧

ファイル名	説明
libgraphic.a	グラフィック・ライブラリです。

1.3. LCD の座標

LCD の座標は、以下のように扱います。

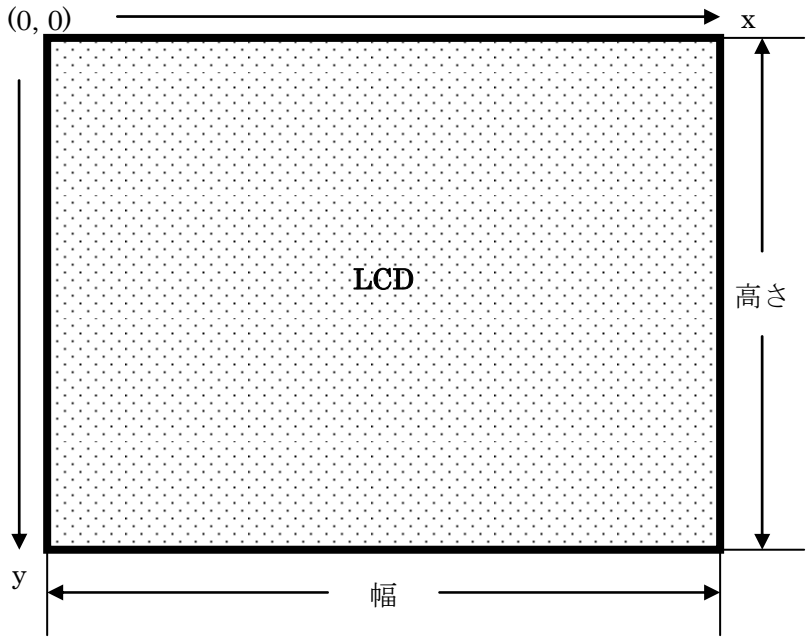


図 1-1 LCD の座標

第2章　データ解説

グラフィック・ライブライリの各 API で使用するデータ・マクロ，構造体について解説します。

2.1. データ・マクロ

データ・マクロは、graphic.h で定義しています。

2.1.1. 返却値マクロ

各 API の返却値として使用するマクロです。

表 2-1　API の返却値一覧

マクロ	値	意味
GRAPHIC_ERROR_NORMAL	0	正常終了
GRAPHIC_ERROR_PARAMETER	-1	パラメータエラー
GRAPHIC_ERROR_FORM	-2	フォーム関連エラー
GRAPHIC_ERROR_OBJECT	-3	GUI オブジェクト関連エラー
GRAPHIC_ERROR_PROPERTY	-4	GUI プロパティ関連エラー

2.1.2. フォントサイズ・マクロ

文字描画 API ”g_character”，文字列描画 API ”g_sring”，GUI のプロパティでフォントサイズを指定するのに使用するマクロです。

表 2-2　フォントサイズ一覧

マクロ	値	意味
GRAPHIC_FONTSIZE_12x6	&g_font12x6	12x6 ドットの等幅フォント
GRAPHIC_FONTSIZE_16x8	&g_font16x8	16x8 ドットの等幅フォント
GRAPHIC_FONTSIZE_14x14	&g_font14x14	14x14 ドットの等幅フォント
GRAPHIC_FONTSIZE_14x14P	&g_font14x14p	14x14 ドットの可変幅フォント

2.1.3. GUI タイプ・マクロ

GUI オブジェクトのタイプを表すマクロです。

表 2-3　GUI のタイプ一覧

マクロ	値	意味
GUI_TYPE_FORM	1	フォーム
GUI_TYPE_COMMANDB	2	コマンド・ボタン
GUI_TYPE_LISTB	3	リスト・ボックス
GUI_TYPE_CHECKB	4	チェック・ボックス
GUI_TYPE_OPTIONB	5	オプション・ボタン
GUI_TYPE_LABEL	6	ラベル
GUI_TYPE_FRAME	7	フレーム

2.2. データ構造体

データ構造体は、graphic.h で宣言しています。

2.2.1. graphic_config 構造体

LCD やドライバ関数の情報です。

```
typedef struct graphic_config {
    unsigned int    (*get_pixel)(unsigned short x, unsigned short y);
    void            (*set_pixel)(unsigned short x, unsigned short y, unsigned int color);
    unsigned char   (*get_touch)(unsigned short *x, unsigned short *y);
    void            (*clear_touch)(void);
    unsigned short  lcd_width;
    unsigned short  lcd_height;
} graphic_config;
```

(1) get_pixel メンバ
get_pixel ドライバ関数のポインタです。

(2) set_pixel メンバ
set_pixel ドライバ関数のポインタです。

(3) get_touch メンバ
get_touch ドライバ関数のポインタです。

(4) clear_touch メンバ
clear_touch ドライバ関数のポインタです。

(5) lcd_width メンバ
LCD の幅（ドット）です。

(6) lcd_height メンバ
LCD の高さ（ドット）です。

2.2.2. graphic_context 構造体

各 API で使用する領域です。

```
typedef struct graphic_context {
    graphic_config    *config;
    int               buffer[480];
} graphic_context;
```

(1) config メンバ
config 情報です。

(2) buffer メンバ
各 API で使用する一時領域です。

2.2.3. pixmap_data 構造体

画像データです。

```
typedef struct pixmap_data {
    unsigned short    width;
    unsigned short    height;
    char              pixel_bytes;
    const void        *data;
} pixmap_data;
```

(1) width メンバ
画像の幅（ドット）です。

(2) height メンバ
画像の高さ（ドット）です。

(3) pixel_bytes メンバ
1 ピクセルあたりのバイト数です。

(4) data メンバ
画像データです。

2.2.4. form 構造体

フォームで使用する領域です。

```
typedef struct form {
    unsigned char           type;
    unsigned char           state;
    const form_property     *property;
    struct form             *next;
    gui_object              *child[2];
    graphic_context         *context;
    unsigned int            control;
} form;
```

(1) type メンバ
GUI オブジェクトのタイプです。

(2) state メンバ
フォームの状態です。

(3) property メンバ
フォームのプロパティです。

(4) next メンバ
他のフォームを表します。

(5) child メンバ
フォームに作成された GUI オブジェクトです。

(6) context メンバ
フォームで使用するコンテキストです。

(7) control メンバ
フォームの制御情報です。

2.2.5. gui_object 構造体

フォーム以外の GUI オブジェクトで使用する領域です。

```
typedef struct gui_object {
    unsigned char                   type;
    union {
        const commandb_property     *commandb;
        const listb_property        *listb;
        const checkb_property       *checkb;
        const optionb_property      *optionb;
        const label_property        *label;
        const frame_property        *frame;
    } property;
    struct gui_object               *next;
    struct form                     *parent;
    unsigned int                    control;
} gui_object;
```

(1) type メンバ
GUI オブジェクトのタイプです。

(2) property メンバ
GUI オブジェクトのプロパティです。

(3) next メンバ
他の GUI オブジェクトを表します。

(4) parent メンバ
GUI オブジェクトが作成されているフォームです。

(5) control メンバ
GUI オブジェクトの制御情報です。

2.2.6. form_property 構造体

フォームのプロパティです。

```
typedef struct form_property {
    unsigned short      x;
    unsigned short      y;
    unsigned short      width;
    unsigned short      height;
    unsigned int        background;
    unsigned int        foreground;
    unsigned int        border_color;
    unsigned char       border_width;
} form_property;
```

(1) x メンバ
LCD 上の x 座標です。

(2) y メンバ
LCD 上の y 座標です。

(3) width メンバ
フォームの幅です。

(4) height メンバ
フォームの高さです。

(5) background メンバ
背景の色です。

(6) foreground メンバ
前景の色です。

(7) border_color メンバ
境界の色です。

(8) border_width メンバ
境界の幅です。

2.2.7. commandb_property 構造体

コマンド・ボタンのプロパティです。

```
typedef struct commandb_property {
    unsigned short          x;
    unsigned short          y;
    unsigned short          width;
    unsigned short          height;
    unsigned int            background;
    unsigned int            foreground;
    unsigned int            border_color;
    unsigned char           border_width;
    const g_font            *font;
    const unsigned char     *name;
    void                    (*function)(void *arg);
    void                    *arg;
} commandb_property;
```

(1) x メンバ
LCD 上の x 座標です。

(2) y メンバ
LCD 上の y 座標です。

(3) width メンバ
コマンド・ボタンの幅です。

(4) height メンバ
コマンド・ボタンの高さです。

(5) background メンバ
背景の色です。

(6) foreground メンバ
前景の色です。

(7) border_color メンバ
境界の色です。

(8) border_width メンバ
境界の幅です。

(9) font メンバ
コマンド・ボタンに表示する文字列のフォントです。

(10) name メンバ
コマンド・ボタンに表示する文字列です。

(11) function メンバ
コマンド・ボタンが選択されたときに呼ばれる関数のポインタです。

(12) arg メンバ
function メンバで指定した関数が呼ばれるときに渡される引数です。

2.2.8. listb_property 構造体

リスト・ボックスのプロパティです。

```
typedef struct listb_property {
    unsigned short          x;
    unsigned short          y;
    unsigned short          item_width;
    unsigned short          item_height;
    unsigned int            background;
    unsigned int            foreground;
    unsigned int            border_color;
    unsigned char           border_width;
    unsigned char           value;
    unsigned char           visible_count;
    const g_font            *font;
    const unsigned char     **items;
    unsigned char           item_count;
    unsigned char           top_index;
} listb_property;
```

(1) x メンバ
LCD 上の x 座標です。

(2) y メンバ
LCD 上の y 座標です。

(3) item_width メンバ
リストの 1 項目の幅です。

(4) item_height メンバ
リストの１項目の高さです。

(5) background メンバ
背景の色です。

(6) foreground メンバ
前景の色です。

(7) border_color メンバ
境界の色です。

(8) border_width メンバ
境界の幅です。

(9) value メンバ
最初に選択されている項目のインデックスです。

(10) visible_count メンバ
一度に表示する項目の数です。

(11) font メンバ
項目名のフォントです。

(12) items メンバ
項目名のリストです。

(13) item_count メンバ
項目数です。

(14) top_index メンバ
最初に先頭に表示される項目のインデックスです。

2.2.9. checkb_property 構造体

チェック・ボックスのプロパティです。

```
typedef struct checkb_property {
    unsigned short          x;
    unsigned short          y;
    unsigned short          width;
    unsigned short          height;
    unsigned int            background;
    unsigned int            foreground;
    unsigned int            border_color;
    unsigned char           border_width;
    unsigned char           value;
    const g_font            *font;
    const unsigned char     *name;
} checkb_property;
```

(1) x メンバ
LCD 上の x 座標です。

(2) y メンバ
LCD 上の y 座標です。

(3) width メンバ
チェック・ボックスの幅です。

(4) height メンバ
チェック・ボックスの高さです。

(5) background メンバ
背景の色です。

(6) foreground メンバ
前景の色です。

(7) border_color メンバ
境界の色です。

(8) border_width メンバ
境界の幅です。

(9) value メンバ
チェック・ボックスの初期値です。
この値が 1 ならチェック状態，0 なら非チェック状態です。

(10) font メンバ
チェック・ボックスに表示する文字列のフォントです。

(11) name メンバ
チェック・ボックスに表示する文字列です。

2.2.10. optionb_property 構造体

オプション・ボタンのプロパティです。

```
typedef struct optionb_property {
    unsigned short          x;
    unsigned short          y;
    unsigned short          width;
    unsigned short          height;
    unsigned int            background;
    unsigned int            foreground;
    unsigned int            border_color;
    unsigned char           border_width;
    unsigned char           value;
    unsigned char           group_id;
    const g_font            *font;
    const unsigned char     *name;
} optionb_property;
```

(1) x メンバ
LCD 上の x 座標です。

(2) y メンバ
LCD 上の y 座標です。

(3) width メンバ
オプション・ボタンの幅です。

(4) height メンバ
オプション・ボタンの高さです。

(5) background メンバ
背景の色です。

(6) foreground メンバ
前景の色です。

(7) border_color メンバ
境界の色です。

(8) border_width メンバ
境界の幅です。

(9) value メンバ
オプション・ボタンの初期値です。
この値が 1 なら選択状態，0 なら非選択状態です。

(10) group_id メンバ
グループ ID です。
同じフォーム内でこの ID が同じ値のオプション・ボタンが同一グループになります。

(11) font メンバ
オプション・ボタンに表示する文字列のフォントです。

(12) name メンバ
オプション・ボタンに表示する文字列です。

2.2.11. label_property 構造体

ラベルのプロパティです。

```
typedef struct label_property {
    unsigned short          x;
    unsigned short          y;
    unsigned short          width;
    unsigned short          height;
    unsigned int            background;
    unsigned int            foreground;
    unsigned int            border_color;
    unsigned char           border_width;
    const g_font            *font;
    const unsigned char     *name;
} label_property;
```

(1) x メンバ
LCD 上の x 座標です。

(2) y メンバ
LCD 上の y 座標です。

(3) width メンバ
ラベルの幅です。

(4) height メンバ
ラベルの高さです。

(5) background メンバ
背景の色です。

(6) foreground メンバ
前景の色です。

(7) border_color メンバ
境界の色です。

(8) border_width メンバ
境界の幅です。

(9) font メンバ
ラベルに表示する文字列のフォントです。

(10) name メンバ
ラベルに表示する文字列です。

2.2.12. frame_property 構造体

フレームのプロパティです。

```
typedef struct frame_property {
    unsigned short      x;
    unsigned short      y;
    unsigned short      width;
    unsigned short      height;
    unsigned int        background;
    unsigned int        foreground;
    unsigned int        border_color;
    unsigned char       border_width;
} frame_property;
```

(1) x メンバ
LCD 上の x 座標です。

(2) y メンバ
LCD 上の y 座標です。

(3) width メンバ
フレームの幅です。

(4) height メンバ
フレームの高さです。

(5) background メンバ
背景の色です。

(6) foreground メンバ
前景の色です。

(7) border_color メンバ
境界の色です。

(8) border_width メンバ
境界の幅です。

2.2.13. g_font_data 構造体

フォントのデータです。

```
typedef struct g_font_data {
    const unsigned char     *data;
    unsigned char           position;
    unsigned char           width;
} g_font_data;
```

(1) data メンバ
フォント・データです。

(2) position メンバ
文字の開始位置です。

(3) width メンバ
文字の幅です。

2.2.14. g_font 構造体

文字のデータです。

```
typedef struct g_font {
    int                 (*get_data)(const unsigned char **code,g_font_data *data);
    unsigned char       height;
    unsigned char       width;
} g_font;
```

(1) get_data メンバ
指定された文字のデータを取得する関数のポインタです。

(2) height メンバ
フォントの高さです。

(3) width メンバ
フォントの幅です。

2.3. フォント用定数

2.3.1. フォント用情報

各フォントの情報を const g_font 型の定数で宣言する。この定数のポインタが API やプロパティで直接指定される。

表 2-4 フォント情報

ファイル名	定数名	説明
font_12x6.c	g_font12x6	12x6 ドットの等幅フォント
font_16x8.c	g_font16x8	16x8 ドットの等幅フォント
font_14x14.c	g_font14x14	14x14 ドットの等幅フォント
font_14x14.c	g_font14x14p	14x14 ドットの可変幅フォント

2.3.2. フォント・データ

各フォントのデータを const unsigned char 型の定数で宣言する。
フォント・データは、左上から横方向に 1 ドットを 1 ビットで、描画する場合は 1，描画しない場合は 0 で表す。

例)
12x6 ドットのフォントのビット順

1	2	3	4	5	6
7	8	9	10	11	12
13	14	15	16	17	18
19	20	21	22	23	24
25	26	27	28	29	30
31	32	33	34	35	36
37	38	39	40	41	42
43	44	45	46	47	48
49	50	51	52	53	54
55	56	57	58	59	60
61	62	63	64	65	66
67	68	69	70	71	72

表 2-5　フォント・データ

定数名	説明
g_font12x6_01[]	12x6 ドットのフォント・データ
g_font16x8_01[]	16x8 ドットのフォント・データ
g_font14x14_01[]	14x14 ドット 1 区のフォント・データ
g_font14x14_02[]	14x14 ドット 2 区のフォント・データ
g_font14x14_03[]	14x14 ドット 3 区のフォント・データ
g_font14x14_04[]	14x14 ドット 4 区のフォント・データ
g_font14x14_05[]	14x14 ドット 5 区のフォント・データ
g_font14x14_06[]	14x14 ドット 6 区のフォント・データ
g_font14x14_07[]	14x14 ドット 7 区のフォント・データ
g_font14x14_08[]	14x14 ドット 8 区のフォント・データ
g_font14x14_09[]	14x14 ドット 9 区のフォント・データ
g_font14x14_10[]	14x14 ドット 10 区のフォント・データ
g_font14x14_11[]	14x14 ドット 11 区のフォント・データ
g_font14x14_12[]	14x14 ドット 12 区のフォント・データ
g_font14x14_13[]	14x14 ドット 13 区のフォント・データ
g_font14x14_14[]	14x14 ドット 14 区のフォント・データ
g_font14x14_15[]	14x14 ドット 15 区のフォント・データ
g_font14x14_16[]	14x14 ドット 16 区のフォント・データ
g_font14x14_17[]	14x14 ドット 17 区のフォント・データ
g_font14x14_18[]	14x14 ドット 18 区のフォント・データ
g_font14x14_19[]	14x14 ドット 19 区のフォント・データ
g_font14x14_20[]	14x14 ドット 20 区のフォント・データ
g_font14x14_21[]	14x14 ドット 21 区のフォント・データ
g_font14x14_22[]	14x14 ドット 22 区のフォント・データ
g_font14x14_23[]	14x14 ドット 23 区のフォント・データ

g_font14x14_24[]	14x14 ドット 24 区のフォント・データ
g_font14x14_25[]	14x14 ドット 25 区のフォント・データ
g_font14x14_26[]	14x14 ドット 26 区のフォント・データ
g_font14x14_27[]	14x14 ドット 27 区のフォント・データ
g_font14x14_28[]	14x14 ドット 28 区のフォント・データ
g_font14x14_29[]	14x14 ドット 29 区のフォント・データ
g_font14x14_30[]	14x14 ドット 30 区のフォント・データ
g_font14x14_31[]	14x14 ドット 31 区のフォント・データ
g_font14x14_32[]	14x14 ドット 32 区のフォント・データ
g_font14x14_33[]	14x14 ドット 33 区のフォント・データ
g_font14x14_34[]	14x14 ドット 34 区のフォント・データ
g_font14x14_35[]	14x14 ドット 35 区のフォント・データ
g_font14x14_36[]	14x14 ドット 36 区のフォント・データ
g_font14x14_37[]	14x14 ドット 37 区のフォント・データ
g_font14x14_38[]	14x14 ドット 38 区のフォント・データ
g_font14x14_39[]	14x14 ドット 39 区のフォント・データ
g_font14x14_40[]	14x14 ドット 40 区のフォント・データ
g_font14x14_41[]	14x14 ドット 41 区のフォント・データ
g_font14x14_42[]	14x14 ドット 42 区のフォント・データ
g_font14x14_43[]	14x14 ドット 43 区のフォント・データ
g_font14x14_44[]	14x14 ドット 44 区のフォント・データ
g_font14x14_45[]	14x14 ドット 45 区のフォント・データ
g_font14x14_46[]	14x14 ドット 46 区のフォント・データ
g_font14x14_47[]	14x14 ドット 47 区のフォント・データ
g_font14x14_48[]	14x14 ドット 48 区のフォント・データ
g_font14x14_49[]	14x14 ドット 49 区のフォント・データ
g_font14x14_50[]	14x14 ドット 50 区のフォント・データ
g_font14x14_51[]	14x14 ドット 51 区のフォント・データ
g_font14x14_52[]	14x14 ドット 52 区のフォント・データ
g_font14x14_53[]	14x14 ドット 53 区のフォント・データ
g_font14x14_54[]	14x14 ドット 54 区のフォント・データ
g_font14x14_55[]	14x14 ドット 55 区のフォント・データ
g_font14x14_56[]	14x14 ドット 56 区のフォント・データ
g_font14x14_57[]	14x14 ドット 57 区のフォント・データ
g_font14x14_58[]	14x14 ドット 58 区のフォント・データ
g_font14x14_59[]	14x14 ドット 59 区のフォント・データ
g_font14x14_60[]	14x14 ドット 60 区のフォント・データ
g_font14x14_61[]	14x14 ドット 61 区のフォント・データ
g_font14x14_62[]	14x14 ドット 62 区のフォント・データ
g_font14x14_63[]	14x14 ドット 63 区のフォント・データ
g_font14x14_64[]	14x14 ドット 64 区のフォント・データ
g_font14x14_65[]	14x14 ドット 65 区のフォント・データ

g_font14x14_66[]	14x14 ドット 66 区のフォント・データ
g_font14x14_67[]	14x14 ドット 67 区のフォント・データ
g_font14x14_68[]	14x14 ドット 68 区のフォント・データ
g_font14x14_69[]	14x14 ドット 69 区のフォント・データ
g_font14x14_70[]	14x14 ドット 70 区のフォント・データ
g_font14x14_71[]	14x14 ドット 71 区のフォント・データ
g_font14x14_72[]	14x14 ドット 72 区のフォント・データ
g_font14x14_73[]	14x14 ドット 73 区のフォント・データ
g_font14x14_74[]	14x14 ドット 74 区のフォント・データ
g_font14x14_75[]	14x14 ドット 75 区のフォント・データ
g_font14x14_76[]	14x14 ドット 76 区のフォント・データ
g_font14x14_77[]	14x14 ドット 77 区のフォント・データ
g_font14x14_78[]	14x14 ドット 78 区のフォント・データ
g_font14x14_79[]	14x14 ドット 79 区のフォント・データ
g_font14x14_80[]	14x14 ドット 80 区のフォント・データ
g_font14x14_81[]	14x14 ドット 81 区のフォント・データ
g_font14x14_82[]	14x14 ドット 82 区のフォント・データ
g_font14x14_83[]	14x14 ドット 83 区のフォント・データ
g_font14x14_84[]	14x14 ドット 84 区のフォント・データ
g_font14x14_85[]	14x14 ドット 85 区のフォント・データ
g_font14x14_86[]	14x14 ドット 86 区のフォント・データ
g_font14x14_87[]	14x14 ドット 87 区のフォント・データ
g_font14x14_88[]	14x14 ドット 88 区のフォント・データ
g_font14x14_89[]	14x14 ドット 89 区のフォント・データ
g_font14x14_90[]	14x14 ドット 90 区のフォント・データ
g_font14x14_91[]	14x14 ドット 91 区のフォント・データ
g_font14x14_92[]	14x14 ドット 92 区のフォント・データ
g_font14x14_93[]	14x14 ドット 93 区のフォント・データ
g_font14x14_94[]	14x14 ドット 94 区のフォント・データ

2.3.3. ユーザー定義定数

使用するフォント・データを指定する定数。const unsigned char *const 型でユーザーが宣言し、グラフィック・ライブラリで参照する。

フォント･データを使用する場合は､該当するフォント･データ定数のポインタを指定する。フォント・データを使用しない場合は、0 を指定する。

表 2-6　ユーザー定義フォント定数

定数名	説明
g_font12x6_list	12x6 ドットのフォント・データの指定
g_font16x8_list	16x8 ドットのフォント・データの指定
g_font14x14_list[94]	14x14 ドットのフォント・データの指定

第3章　ドライバ関数

3.1. 概説

ドライバ関数は、グラフィック・ライブラリが使用する関数で、ハードウェアに依存した処理をユーザ・オウン・コーディング部として切り出したものです。ドライバ関数は、graphic_config 構造体に設定してグラフィック機能の初期化 API “graphic_init”へ渡します。

表 3-1　ドライバ関数一覧

関数名	機能
get_pixel	指定された座標の色を取得します。
set_pixel	指定された座標に指定された色を描画します。
get_touch	LCD のタッチ座標を取得します。
clear_touch	LCD のタッチ情報をクリアします。

3.2. ドライバ関数の仕様

3.2.1. get_pixel

【機能】

指定された座標の色を取得します。

【C 言語形式】

unsigned int get_pixel(unsigned short x, unsigned short y);

【パラメータ】

I/O	パラメータ	説明
I	unsigned short x	色を取得する x 座標です。
I	unsigned short y	色を取得する y 座標です。

【返却値】

マクロ	値	内容
―	ALL	取得した色です。

【処理内容】

パラメータで指定された座標の色を取得し、取得した色を返却します。

3.2.2. set_pixel

【機能】

指定された座標に指定された色を描画します。

【C 言語形式】

void set_pixel(unsigned short x, unsigned short y, unsigned int color);

【パラメータ】

I/O	パラメータ	説明
I	unsigned short x	描画する x 座標です。
I	unsigned short y	描画する y 座標です。
I	unsigned int color	描画する色です。

【返却値】

なし

【処理内容】

パラメータで指定された座標に指定された色の点を描画します。

3.2.3. get_touch

【機能】

LCD のタッチ座標を取得します。

【C 言語形式】

unsigned char get_touch(unsigned short *x, unsigned short *y);

【パラメータ】

I/O	パラメータ	説明
O	unsigned short *x	タッチした x 座標を格納する領域です。
O	unsigned short *y	タッチした y 座標を格納する領域です。

【返却値】

マクロ	値	内容
－	0	LCD にタッチしていません。
－	1	LCD にタッチしていました。

【処理内容】

LCD にタッチしていたかどうかをチェックし、タッチしていない場合は 0 を返却します。タッチしていた場合は、パラメータで指定された領域にタッチした座標を格納して 1

を返却します。ただし、パラメータで指定された領域が 0 の場合は、タッチした座標を格納しません。

3.2.4. clear_touch

【機能】
LCD のタッチ情報をクリアします。

【C 言語形式】
void clear_touch(void);

【パラメータ】
なし

【返却値】
なし

【処理内容】
LCD にタッチしていたかどうかを示す情報をクリアします。すなわち、LCD にタッチしていない状態にします。

第4章　API

4.1. 概説

グラフィック・ライブラリでは、API を実行することによりグラフィック機能を実現します。

4.2. 初期化 API

グラフィック・ライブラリの各機能を使用する前にグラフィック機能の初期化を行う必要があります。graphic_context 構造体は、API で指定する前に初期化する必要があります。

表 4-1　初期化 API 一覧

API 名	機能
graphic_init	グラフィック機能を初期化します。
g_context_init	コンテキストを初期化します。

4.2.1. graphic_init

【機能】

グラフィック機能を初期化します。

【C 言語形式】

int graphic_init(const graphic_config *config);

【パラメータ】

I/O	パラメータ	説明
I	const graphic_config *config	ドライバ関数，LCD のサイズを設定した graphic_config 構造体を指定します。

【返却値】

マクロ	値	内容
GRAPHIC_ERROR_NORMAL	0	正常終了
GRAPHIC_ERROR_PARAMETER	-1	パラメータエラー

4.2.2. g_context_init

【機能】

コンテキストを初期化します。

【C 言語形式】

int g_context_init(graphic_context *ctx);

【パラメータ】

I/O	パラメータ	説明
I/O	graphic_context *ctx	graphic_context 構造体を指定します。

【返却値】

マクロ	値	内容
GRAPHIC_ERROR_NORMAL	0	正常終了
GRAPHIC_ERROR_PARAMETER	-1	パラメータエラー

4.3. 図形描画機能の API

図形を描画する API です。以下の API があります。

表 4-2 図形描画機能の API 一覧

API 名	機能
g_line	直線を描画します。
g_rectangle	長方形を描画します。
g_rectanglef	長方形を描画して塗りつぶします。
g_circle	円を描画します。
g_circlef	円を描画して塗りつぶします。
g_ellipse	楕円を描画します。
g_ellipsef	楕円を描画して塗りつぶします。
g_arc	円弧を描画します。
g_arcf	円弧を描画して塗りつぶします。
g_paint	指定された領域を塗りつぶします。
g_character	1 文字を描画します。
g_string	文字列を描画します。
g_get_character_width	1 文字の幅を取得する。
g_get_string_width	文字列の幅を取得する。
g_pixmap	画像を描画します。

4.3.1. g_line

【機能】

直線を描画します。

【C 言語形式】

```
int g_line(graphic_context *ctx, unsigned short x1, unsigned short y1,
        unsigned short x2, unsigned short y2, unsigned int color);
```

【パラメータ】

I/O	パラメータ	説明
I/O	graphic_context *ctx	初期化済みの graphic_context 構造体を指定します。
I	unsigned short x1	描画する直線の始点の x 座標を指定します。
I	unsigned short y1	描画する直線の始点の y 座標を指定します。
I	unsigned short x2	描画する直線の終点の x 座標を指定します。
I	unsigned short y2	描画する直線の終点の y 座標を指定します。
I	unsigned int color	描画する直線の色を指定します。

【返却値】

マクロ	値	内容
GRAPHIC_ERROR_NORMAL	0	正常終了
GRAPHIC_ERROR_PARAMETER	-1	パラメータエラー

4.3.2. g_rectangle

【機能】

長方形を描画します。

【C 言語形式】

```
int g_rectangle(graphic_context *ctx, unsigned short x, unsigned short y,
                unsigned short w, unsigned short h, unsigned int color);
```

【パラメータ】

I/O	パラメータ	説明
I/O	graphic_context *ctx	初期化済みの graphic_context 構造体を指定します。
I	unsigned short x	描画する長方形の x 座標を指定します。
I	unsigned short y	描画する長方形の y 座標を指定します。
I	unsigned short w	描画する長方形の幅を指定します。
I	unsigned short h	描画する長方形の高さを指定します。
I	unsigned int color	描画する長方形の色を指定します。

【返却値】

マクロ	値	内容
GRAPHIC_ERROR_NORMAL	0	正常終了
GRAPHIC_ERROR_PARAMETER	-1	パラメータエラー

4.3.3. g_rectanglef

【機能】

長方形を描画して塗りつぶします。

【C 言語形式】

```
int g_rectanglef(graphic_context *ctx, unsigned short x, unsigned short y,
                 unsigned short w, unsigned short h, unsigned int color);
```

【パラメータ】

I/O	パラメータ	説明
I/O	graphic_context *ctx	初期化済みの graphic_context 構造体を指定します。
I	unsigned short x	描画する長方形の x 座標を指定します。
I	unsigned short y	描画する長方形の y 座標を指定します。
I	unsigned short w	描画する長方形の幅を指定します。
I	unsigned short h	描画する長方形の高さを指定します。
I	unsigned int color	描画する長方形の色を指定します。

【返却値】

マクロ	値	内容
GRAPHIC_ERROR_NORMAL	0	正常終了
GRAPHIC_ERROR_PARAMETER	-1	パラメータエラー

4.3.4. g_circle

【機能】

円を描画します。

【C 言語形式】

```
int g_circle(graphic_context *ctx, unsigned short x, unsigned short y,
        unsigned short r, unsigned int color);
```

【パラメータ】

I/O	パラメータ	説明
I/O	graphic_context *ctx	初期化済みの graphic_context 構造体を指定します。
I	unsigned short x	描画する円の中心点の x 座標を指定します。
I	unsigned short y	描画する円の中心点の y 座標を指定します。
I	unsigned short r	描画する円の半径を指定します。
I	unsigned int color	描画する円の色を指定します。

【返却値】

マクロ	値	内容
GRAPHIC_ERROR_NORMAL	0	正常終了
GRAPHIC_ERROR_PARAMETER	-1	パラメータエラー

4.3.5. g_circlef

【機能】

円を描画して塗りつぶします。

【C 言語形式】

```
int g_circlef(graphic_context *ctx, unsigned short x, unsigned short y,
        unsigned short r, unsigned int color);
```

【パラメータ】

I/O	パラメータ	説明
I/O	graphic_context *ctx	初期化済みの graphic_context 構造体を指定します。
I	unsigned short x	描画する円の中心点の x 座標を指定します。
I	unsigned short y	描画する円の中心点の y 座標を指定します。
I	unsigned short r	描画する円の半径を指定します。
I	unsigned int color	描画する円の色を指定します。

【返却値】

マクロ	値	内容
GRAPHIC_ERROR_NORMAL	0	正常終了
GRAPHIC_ERROR_PARAMETER	-1	パラメータエラー

4.3.6. g_ellipse

【機能】

楕円を描画します。

【C 言語形式】

```
int g_ellipse(graphic_context *ctx, unsigned short x, unsigned short y,
                unsigned short h, unsigned short v, unsigned int color);
```

【パラメータ】

I/O	パラメータ	説明
I/O	graphic_context *ctx	初期化済みの graphic_context 構造体を指定します。
I	unsigned short x	描画する楕円の中心点の x 座標を指定します。
I	unsigned short y	描画する楕円の中心点の y 座標を指定します。
I	unsigned short h	描画する楕円の水平方向の半径を指定します。
I	unsigned short v	描画する楕円の垂直方向の半径を指定します。
I	unsigned int color	描画する楕円の色を指定します。

【返却値】

マクロ	値	内容
GRAPHIC_ERROR_NORMAL	0	正常終了
GRAPHIC_ERROR_PARAMETER	-1	パラメータエラー

4.3.7. g_ellipsef

【機能】

楕円を描画して塗りつぶします。

【C 言語形式】

```
int g_ellipsef(graphic_context *ctx, unsigned short x, unsigned short y,
                unsigned short h, unsigned short v, unsigned int color);
```

【パラメータ】

I/O	パラメータ	説明
I/O	graphic_context *ctx	初期化済みの graphic_context 構造体を指定します。
I	unsigned short x	描画する楕円の中心点の x 座標を指定します。
I	unsigned short y	描画する楕円の中心点の y 座標を指定します。
I	unsigned short h	描画する楕円の水平方向の半径を指定します。
I	unsigned short v	描画する楕円の垂直方向の半径を指定します。
I	unsigned int color	描画する楕円の色を指定します。

【返却値】

マクロ	値	内容
GRAPHIC_ERROR_NORMAL	0	正常終了
GRAPHIC_ERROR_PARAMETER	-1	パラメータエラー

4.3.8. g_arc

【機能】

円弧を描画します。

【C 言語形式】

int g_arc(graphic_context *ctx, short x, short y, unsigned short h, unsigned short v, unsigned short s, unsigned short a, unsigned int color);

【パラメータ】

I/O	パラメータ	説明
I/O	graphic_context *ctx	初期化済みの graphic_context 構造体を指定します。
I	short x	描画する円弧を含む楕円の中心点の x 座標を指定します。
I	short y	描画する円弧を含む楕円の中心点の y 座標を指定します。
I	unsigned short h	描画する円弧を含む楕円の水平方向の半径を指定します。
I	unsigned short v	描画する円弧を含む楕円の垂直方向の半径を指定します。
I	unsigned short s	描画する円弧の始点の角度（°）を指定します。
I	unsigned short a	描画する円弧の角度（°）を指定します。
I	unsigned int color	描画する円弧の色を指定します。

【返却値】

マクロ	値	内容
GRAPHIC_ERROR_NORMAL	0	正常終了
GRAPHIC_ERROR_PARAMETER	-1	パラメータエラー

4.3.9. g_arcf

【機能】

円弧を描画して扇形に塗りつぶします。

【C 言語形式】

int g_arcf(graphic_context *ctx, short x, short y, unsigned short h, unsigned short v, unsigned short s, unsigned short a, unsigned int color);

【パラメータ】

I/O	パラメータ	説明
I/O	graphic_context *ctx	初期化済みの graphic_context 構造体を指定します。
I	short x	描画する円弧を含む楕円の中心点の x 座標を指定します。
I	short y	描画する円弧を含む楕円の中心点の y 座標を指定します。
I	unsigned short h	描画する円弧を含む楕円の水平方向の半径を指定します。
I	unsigned short v	描画する円弧を含む楕円の垂直方向の半径を指定します。
I	unsigned short s	描画する円弧の始点の角度（°）を指定します。
I	unsigned short a	描画する円弧の角度（°）を指定します。
I	unsigned int color	描画する円弧の色を指定します。

【返却値】

マクロ	値	内容
GRAPHIC_ERROR_NORMAL	0	正常終了
GRAPHIC_ERROR_PARAMETER	-1	パラメータエラー

4.3.10. g_paint

【機能】

指定された領域を塗りつぶします。

【C 言語形式】

int g_paint(graphic_context *ctx, unsigned short x, unsigned short y,
unsigned int color);

【パラメータ】

I/O	パラメータ	説明
I/O	graphic_context *ctx	初期化済みの graphic_context 構造体を指定します。
I	unsigned short x	塗りつぶす領域内の x 座標を指定します。
I	unsigned short y	塗りつぶす領域内の y 座標を指定します。
I	unsigned int color	塗りつぶす色を指定します。

【返却値】

マクロ	値	内容
GRAPHIC_ERROR_NORMAL	0	正常終了
GRAPHIC_ERROR_PARAMETER	-1	パラメータエラー

4.3.11. g_character

【機能】

1 文字を描画します。

【C 言語形式】

int g_character(graphic_context *ctx, unsigned short x, unsigned short y,
const unsigned char *c, const g_font *font, unsigned int color);

【パラメータ】

I/O	パラメータ	説明
I/O	graphic_context *ctx	初期化済みの graphic_context 構造体を指定します。
I	unsigned short x	描画する文字の x 座標を指定します。
I	unsigned short y	描画する文字の y 座標を指定します。
I	const unsigned char *c	描画する文字を指定します。
I	const g_font *font	描画する文字のフォントを指定します。
I	unsigned int color	描画する文字の色を指定します。

【返却値】

マクロ	値	内容
GRAPHIC_ERROR_NORMAL	0	正常終了
GRAPHIC_ERROR_PARAMETER	-1	パラメータエラー

4.3.12. g_string

【機能】

文字列を描画します。

【C 言語形式】

int g_string(graphic_context *ctx, unsigned short x, unsigned short y,
const unsigned char *str, const g_font *font, unsigned int color);

【パラメータ】

I/O	パラメータ	説明
I/O	graphic_context *ctx	初期化済みの graphic_context 構造体を指定します。
I	unsigned short x	描画する文字列の x 座標を指定します。
I	unsigned short y	描画する文字列の y 座標を指定します。
I	const unsigned char *str	描画する文字列を指定します。
I	const g_font *font	描画する文字列のフォントを指定します。
I	unsigned int color	描画する文字列の色を指定します。

【返却値】

マクロ	値	内容
GRAPHIC_ERROR_NORMAL	0	正常終了
GRAPHIC_ERROR_PARAMETER	-1	パラメータエラー

4.3.13. g_get_character_width

【機能】

1 文字の幅を取得する。

【C 言語形式】

int g_get_character_width(graphic_context *ctx, const unsigned char *c, const g_font *font);

【パラメータ】

I/O	パラメータ	説明
I/O	graphic_context *ctx	初期化済みの graphic_context 構造体を指定します。
I	const unsigned char *c	幅を取得する文字を指定します。
I	const g_font *font	文字のフォントを指定します。

【返却値】

マクロ	値	内容
―	0 以上	文字の幅
GRAPHIC_ERROR_PARAMETER	-1	パラメータエラー

4.3.14. g_get_string_width

【機能】

文字列の幅を取得する。

【C 言語形式】

int g_get_string_width(graphic_context *ctx, const unsigned char *str, const g_font *font, unsigned short max_width);

【パラメータ】

I/O	パラメータ	説明
I/O	graphic_context *ctx	初期化済みの graphic_context 構造体を指定します。
I	const unsigned char *str	幅を取得する文字列を指定します。('¥0'で終了)
I	const g_font *font	文字列のフォントを指定します。
I	unsigned short max_width	取得する幅の上限を指定します。

【返却値】

マクロ	値	内容
－	0 以上	文字列の幅
GRAPHIC_ERROR_PARAMETER	-1	パラメータエラー

4.3.15. g_pixmap

【機能】

画像を描画します。

【C 言語形式】

```
int g_pixmap(graphic_context *ctx, unsigned short x, unsigned short y,
                unsigned short w, unsigned short h, const pixmap_data *data);
```

【パラメータ】

I/O	パラメータ	説明
I/O	graphic_context *ctx	初期化済みの graphic_context 構造体を指定します。
I	unsigned short x	描画する画像の x 座標を指定します。
I	unsigned short y	描画する画像の y 座標を指定します。
I	unsigned short w	描画する画像の幅を指定します。
I	unsigned short h	描画する画像の高さを指定します。
I	const pixmap_data *data	描画する画像を指定します。

【返却値】

マクロ	値	内容
GRAPHIC_ERROR_NORMAL	0	正常終了
GRAPHIC_ERROR_PARAMETER	-1	パラメータエラー

4.4. GUI 機能の API

GUI による操作を行うための API です。以下の API があります。

表 4-3 GUI 機能の API 一覧

API 名	機能
gui_create_form	フォームを作成します。
gui_delete_form	フォームを削除します。
gui_draw_form	フォームを表示します。
gui_commandb	コマンド・ボタンを作成します。
gui_listb	リスト・ボックスを作成します。
gui_checkb	チェック・ボックスを作成します。
gui_optionb	オプション・ボタンを作成します。
gui_label	ラベルを作成します。
gui_frame	フレームを作成します。
gui_get_value	GUI オブジェクトの値を取得します。
gui_touch_check	タッチパネルの操作に対応した GUI の動作を行います。

4.4.1. gui_create_form

【機能】

フォームを作成します。

【C 言語形式】

int gui_create_form(graphic_context *ctx, form *fm, const form_property *p);

【パラメータ】

I/O	パラメータ	説明
I	graphic_context *ctx	初期化済みの graphic_context 構造体を指定します。
I/O	form *fm	form 構造体を指定します。
I	const form_property *p	作成するフォームのプロパティを指定します。

【返却値】

マクロ	値	内容
GRAPHIC_ERROR_NORMAL	0	正常終了
GRAPHIC_ERROR_PARAMETER	-1	パラメータエラー
GRAPHIC_ERROR_FORM	-2	指定されたフォームは作成済み

4.4.2. gui_delete_form

【機能】

フォームを削除します。

【C 言語形式】

int gui_delete_form(form *fm);

【パラメータ】

I/O	パラメータ	説明
I	form *fm	作成済みの form 構造体を指定します。

【返却値】

マクロ	値	内容
GRAPHIC_ERROR_NORMAL	0	正常終了
GRAPHIC_ERROR_PARAMETER	-1	パラメータエラー

4.4.3. gui_draw_form

【機能】

フォームを表示します。

【C 言語形式】

int gui_draw_form(form *fm);

【パラメータ】

I/O	パラメータ	説明
I	form *fm	form 構造体の領域を指定します。

【返却値】

マクロ	値	内容
GRAPHIC_ERROR_NORMAL	0	正常終了
GRAPHIC_ERROR_PARAMETER	-1	パラメータエラー
GRAPHIC_ERROR_FORM	-2	指定されたフォームは未作成
GRAPHIC_ERROR_PROPERTY	-4	プロパティのエラー

4.4.4. gui_commandb

【機能】

コマンド・ボタンを作成します。

【C 言語形式】

int gui_commandb(form *fm, gui_object *obj, const commandb_property *p);

【パラメータ】

I/O	パラメータ	説明
I/O	form *fm	作成済みの form 構造体を指定します。
I/O	gui_object *obj	gui_object 構造体を指定します。
I	const commandb_property *p	作成するコマンド・ボタンのプロパティを指定します。

【返却値】

マクロ	値	内容
GRAPHIC_ERROR_NORMAL	0	正常終了
GRAPHIC_ERROR_PARAMETER	-1	パラメータエラー
GRAPHIC_ERROR_FORM	-2	指定されたフォームは未作成
GRAPHIC_ERROR_PROPERTY	-4	プロパティのエラー

4.4.5. gui_listb

【機能】

リスト・ボックスを作成します。

【C 言語形式】

int gui_listb(form *fm, gui_object *obj, const listb_property *p);

【パラメータ】

I/O	パラメータ	説明
I/O	form *fm	親の form 構造体の領域を指定します。
I/O	gui_object *obj	gui_object 構造体の領域を指定します。
I	const listb_property *p	作成するリスト・ボックスのプロパティを指定します。

【返却値】

マクロ	値	内容
GRAPHIC_ERROR_NORMAL	0	正常終了
GRAPHIC_ERROR_PARAMETER	-1	パラメータエラー
GRAPHIC_ERROR_FORM	-2	指定されたフォームは未作成
GRAPHIC_ERROR_PROPERTY	-4	プロパティのエラー

4.4.6. gui_checkb

【機能】

チェック・ボックスを作成します。

【C 言語形式】

int gui_checkb(form *fm, gui_object *obj, const checkb_property *p);

【パラメータ】

I/O	パラメータ	説明
I/O	form *fm	作成済みの form 構造体を指定します。
I/O	gui_object *obj	gui_object 構造体を指定します。
I	const checkb_property *p	作成するチェック・ボックスのプロパティを指定します。

【返却値】

マクロ	値	内容
GRAPHIC_ERROR_NORMAL	0	正常終了
GRAPHIC_ERROR_PARAMETER	-1	パラメータエラー
GRAPHIC_ERROR_FORM	-2	指定されたフォームは未作成
GRAPHIC_ERROR_PROPERTY	-4	プロパティのエラー

4.4.7. gui_optionb

【機能】

オプション・ボタンを作成します。

【C 言語形式】

int gui_optionb(form *fm, gui_object *obj, const optionb_property *p);

【パラメータ】

I/O	パラメータ	説明
I/O	form *fm	作成済みの form 構造体を指定します。
I/O	gui_object *obj	gui_object 構造体を指定します。
I	const optionb_property *p	作成するオプション・ボタンのプロパティを指定します。

【返却値】

マクロ	値	内容
GRAPHIC_ERROR_NORMAL	0	正常終了
GRAPHIC_ERROR_PARAMETER	-1	パラメータエラー
GRAPHIC_ERROR_FORM	-2	指定されたフォームは未作成
GRAPHIC_ERROR_PROPERTY	-4	プロパティのエラー

4.4.8. gui_label

【機能】

ラベルを作成します。

【C 言語形式】

int gui_label(form *fm, gui_object *obj, const label_property *p);

【パラメータ】

I/O	パラメータ	説明
I/O	form *fm	作成済みの form 構造体を指定します。
I/O	gui_object *obj	gui_object 構造体を指定します。
I	const label_property *p	作成するラベルのプロパティを指定します。

【返却値】

マクロ	値	内容
GRAPHIC_ERROR_NORMAL	0	正常終了
GRAPHIC_ERROR_PARAMETER	-1	パラメータエラー
GRAPHIC_ERROR_FORM	-2	指定されたフォームは未作成
GRAPHIC_ERROR_PROPERTY	-4	プロパティのエラー

4.4.9. gui_frame

【機能】

フレームを作成します。

【C 言語形式】

int gui_frame(form *fm, gui_object *obj, const frame_property *p);

【パラメータ】

I/O	パラメータ	説明
I/O	form *fm	作成済みの form 構造体を指定します。
I/O	gui_object *obj	gui_object 構造体を指定します。
I	const frame_property *p	作成するフレームのプロパティを指定します。

【返却値】

マクロ	値	内容
GRAPHIC_ERROR_NORMAL	0	正常終了
GRAPHIC_ERROR_PARAMETER	-1	パラメータエラー
GRAPHIC_ERROR_FORM	-2	指定されたフォームは未作成
GRAPHIC_ERROR_PROPERTY	-4	プロパティのエラー

4.4.10. gui_get_value

【機能】

GUI オブジェクトの値を取得します。

【C 言語形式】

int gui_get_value(gui_object *obj);

【パラメータ】

I/O	パラメータ	説明
I	gui_object *obj	作成済みの gui_object 構造体を指定します。

【返却値】

マクロ	値	内容
－	0 以上	GUI オブジェクトの値
GRAPHIC_ERROR_PARAMETER	-1	パラメータエラー
GRAPHIC_ERROR_OBJECT	-3	指定されたオブジェクトは値を持たない

4.4.11. gui_touch_check

【機能】

タッチパネルの操作に対応した GUI の動作を行います。

【C 言語形式】

gui_object *gui_touch_check(void);

【パラメータ】

なし

【返却値】

マクロ	値	内容
－	0	有効なタッチパネルの操作が行われていない
－	0 以外	タッチされた GUI オブジェクト

第5章　ビットマップ変換ツール

5.1.　概説

任意の画像を表示させるには、画像描画 API “g_pixmap”を使用します。g_pixmap では、画像データを pixmap_data 構造体の形式で指定する必要があります。そのため、グラフィック・ライブラリでは、BMP 形式の画像ファイルを pixmap_data 形式の C 言語ソース・ファイルに変換する Windows 用ツール”Gbmp2c”を提供しています。”Gbmp2c”を使用して pixmap_data 形式に変換した画像は、直接 g_pixmap に指定することができます。

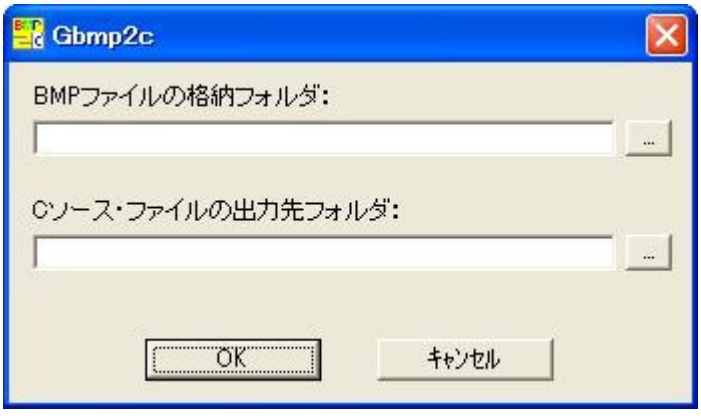
5.2.　画像変換の準備

(1) BMP ファイルを用意する
表示させたい画像の BMP ファイルを用意して任意のフォルダに格納します。

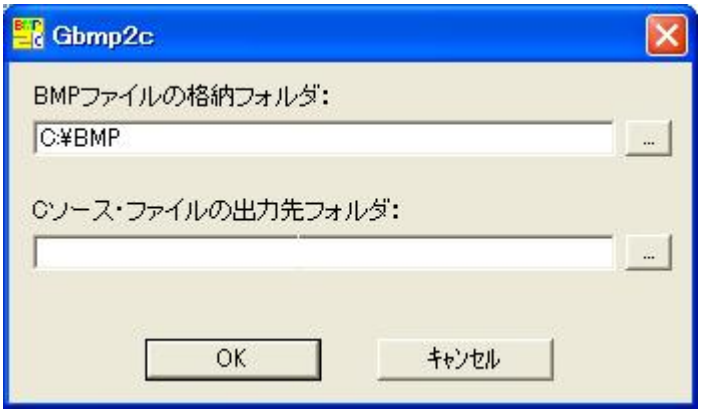
(2) 出力先フォルダを準備する
Gbmp2c を実行した結果出力される C ソース・ファイルを格納するフォルダを準備します。

5.3.　Gbmp2c の使用方法

(1) Gbmp2c を起動する
“TK850¥JH3U_graphic¥bin¥Gbmp2c.exe”を実行します。
以下のダイアログが表示されます。

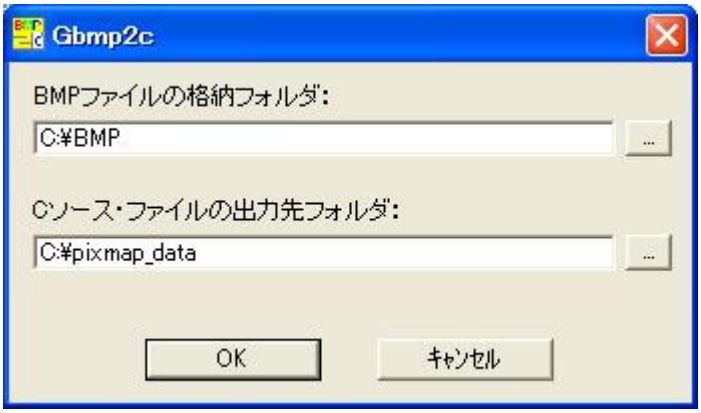


(2) BMP ファイルを指定する
BMP ファイルを格納しているフォルダを指定します。「BMP ファイルの格納フォルダ」に直接パスを入力するか、右側の「…」ボタンを押下して表示される「フォルダの参照」からフォルダを選択します。



(3) C 言語ソース・ファイルの出力先を指定する

変換結果の C 言語ソース・ファイルを出力するフォルダを指定します。「C ソース・ファイルの出力先フォルダ」に直接パスを入力するか、右側の「...」ボタンを押下して表示される「フォルダの参照」からフォルダを選択します。



(4) 変換を実行する

変換を実行するには、「OK」ボタンを押下します。

実行中にエラーが発生した場合は、エラー・メッセージを表示します。

実行が正常に終わると「実行完了しました。」と表示され、「OK」ボタンを押下するとダイアログを閉じます。