

# Dreamcast

## 組み込みマニュアル

### 簡易アニメーション ライブラリ

1999年07月30日	Ver . 1.00
1999年08月02日	Ver . 1.01
1999年10月07日	Ver . 1.03
1999年11月09日	Ver . 1.06
1999年12月22日	Ver . 1.07
2000年02月29日	Ver . 1.08

## 変 更 履 歴

年月日	バージョン	変 更 内 容
1999.07.30	1.00	・ 新規作成。
1999.08.02	1.01	・ SAN_GetNumFrm を SAN_GetNumPic に修正。 ・ 記述ミスを修正。
1999.10.07	1.03	・ サーフェス制御関数を追加。 ・ 合成への対応に伴い、ヘッダ仕様の変更。 ・ 記述ミスを修正。
1999.11.09	1.06	・ サーフェス情報構造体を追加。 ・ サーフェス情報取得関数の追加。 ・ マルチウィンドウについての説明を追加。 ・ SAN テクスチャについての説明を追加。 ・ 合成についての説明を追加。 ・ 記述ミスを修正。
1999.12.22	1.07	・ 記述ミスを修正。
2000.02.29	1.08	・ SAN_SetVertexBuffer 関数の説明を追加。

# 目 次

1. 概 要 .....	1
1.1 目 的 .....	1
1.2 モジュール構成 .....	1
2. SAN の仕組み .....	2
3. ライブラリの使用方法 .....	3
4. マルチウィンドウ .....	4
5. SAN テクスチャ .....	5
6. 合 成 .....	7
6.1 合成機能 .....	7
6.2 アルファ合成の仕組み .....	7
7. データフォーマット .....	8
8. データの作成方法 .....	9
8.1 不透明表示、半透明合成、加算合成 .....	9
8.2 アルファ合成 .....	10
9. データ仕様 .....	11
9.1 データ型 .....	12
10. 関数仕様 .....	13
10.1 初期化と終了処理 .....	14
10.2 基本動作処理 .....	17
10.3 表示制御 .....	21
10.4 サーフェス制御 .....	25
10.5 テクスチャ .....	31
10.6 エラー処理 .....	32

## 1. 概 要

### 1.1 目 的

本ライブラリは、BMP ファイルを YUV420 フォーマットに変換して、連結したデータを容易に表示するためのライブラリです。

### 1.2 モジュール構成

簡易アニメーションライブラリ (SAN) のモジュール構成図を以下に示します。



図 1 - 1 モジュール構成

## 2. SAN の仕組み

SAN\_LoadTex 関数は、SAN データ内の 1 つのピクチャをテクスチャバッファに転送します。SAN\_Draw 関数は、テクスチャバッファに転送したデータを、フレームバッファに転送し表示します。SAN データ内の 1 ピクチャのデータフォーマットは、YUV420 です。テクスチャバッファに転送されると、YUV422 フォーマットに変換されます。

SAN の仕組みを以下に示します。

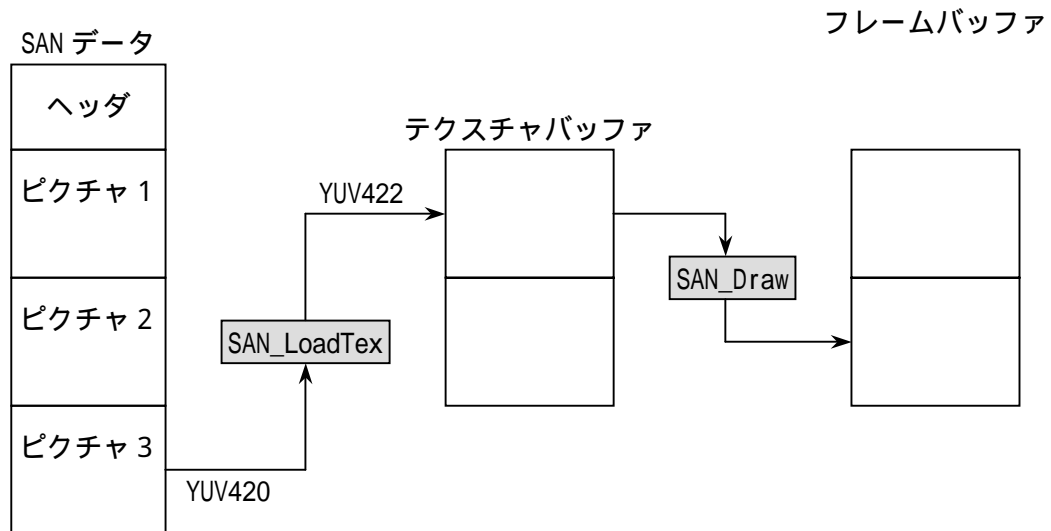


図 2 - 1 SAN の仕組み

### 3. ライブラリの使用方法

以下に、サンプルプログラムを示します。

#### < サンプルプログラム >

```
#define NUM_TEX (2)                                /* テクスチャバッファ数          */

void main(void)
{
    SAN    san;                                     /* SAN ハンドル                  */
    char    *sandat;                               /* SAN データ                    */
    char    *wk;                                    /* 作業領域                     */
    long    wksize;                                 /* 作業領域サイズ               */
    long    pno;                                    /* ピクチャ番号                 */

    SAN_Init();                                     /* ライブラリの初期化          */
    wksize = SAN_CalcWorkSize(NUM_TEX);            /* 作業領域サイズの計算        */
    wk = (char *)syMalloc(wksize);                /* 作業領域の確保              */
    san = SAN_Create(sandat, NUM_TEX, wk);         /* ハンドルの生成              */
    pno = 0;
    for (;;) {
        per = pdGetPeripheral(PDD_PORT_A0);
        if (per->press & PDD_DGT_TA) {
            pno = (pno + 1) % SAN_GetNumPic(san);
        }

        SAN_LoadTex(san, pno, 0);                 /* YUV420 データの V-RAM への転送 */
        SAN_Draw(san, 0);                         /* テクスチャの表示            */

        njWaitVSync();
    }

    SAN_Destroy(san);                             /* ハンドルの消去              */
    SAN_Finish();                                 /* ライブラリの終了処理        */
}
```

## 4. マルチウィンドウ

SAN は、映像を複数のウィンドウに表示することができます。ウィンドウの各 4 頂点に対して、以下のパラメータを独立に設定することができます。

### (1) 表示位置

X 座標、Y 座標、Z 座標を設定できます。X 座標、Y 座標はフレームバッファの座標です。左上が(0.0, 0.0)、右下が(639.0, 479.0)になります。Z 座標を指定しますが、透視変換は行いません。

### (2) 輝度

アルファ、赤、緑、青の強さを指定します。1.0 から 0.0 までの値を設定します。映像データに対して、この設定値が乗じられます。赤、緑、青の値を 0.0 から 1.0 まで変化させると黒からフェードインし、1.0 から 0.0 まで変化させると黒へフェードアウトします。また、アルファは背景との混合比率になります。これを変化させることによりディゾルブ効果が得られます。

### (3) 輝度オフセット

アルファ、赤、緑、青の加算成分を指定します。1.0 から 0.0 までの値を設定します。映像データに対して、この設定値と 255 を乗じた値が加算されます。赤、緑、青を 0.0 から 1.0 まで変化させると白へフェードアウトし、1.0 から 0.0 まで変化させると白からフェードインします。

### (4) 映像切り出し位置

各頂点に対応する映像データの位置を指定します。UV 座標に相当しますが、指定する値は、映像データに対するピクセル単位での位置です。

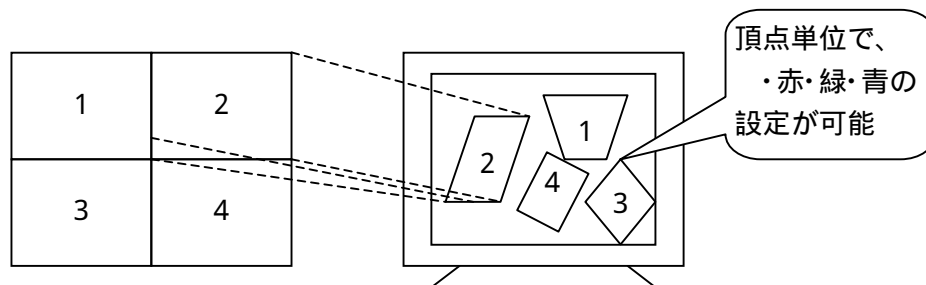


図 4-1 マルチウィンドウ表示

マルチウィンドウ表示を行うためには、サーフェスポイント用バッファの確保と設定が必要です。サーフェスポイントとは、ウィンドウの各頂点を示します。ウィンドウは、4 つのサーフェスポイントから構成されます。例えば、25 個のウィンドウを表示するためには、100 個のサーフェスポイントが必要です。

以下に、サーフェスポイント用バッファの確保と設定の方法を示します。

#### <サーフェスポイント用バッファの確保と設定>

```
size = SAN_CalcSrfBufSize(san, 4*25);  
buf = syMalloc(size);  
SAN_SetSrfPntBuf(san, 4*25, buf, size);
```

## 5. SAN テクスチャ

SAN は、Kamui ドライバを使用して映像用テクスチャサーフェスとマスク用テクスチャサーフェスを確保しています。アプリケーションは、SAN ハンドルからサーフェスを取得し、アプリケーション側のオブジェクトのサーフェスに映像とマスクを貼ることができます。

以下に、SAN テクスチャの仕組みを示します。

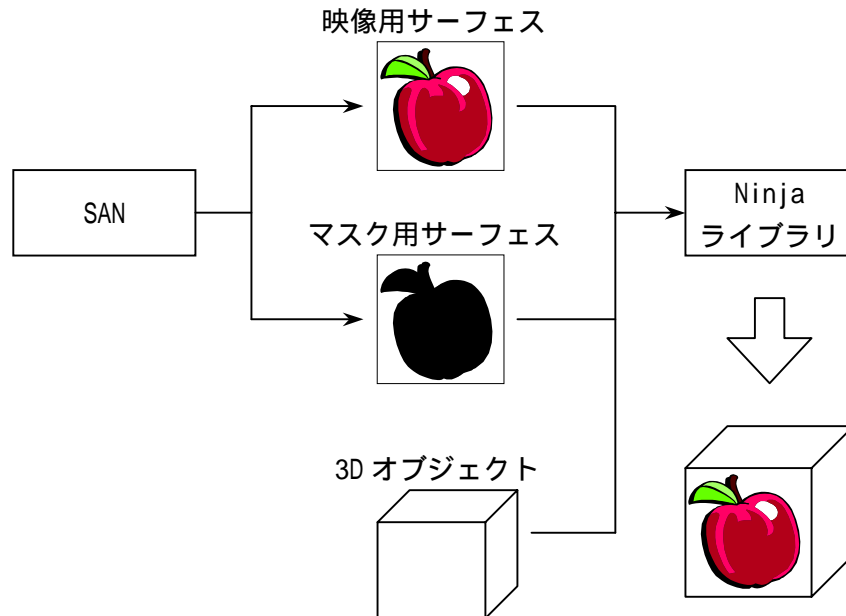


図 5 - 1 SAN テクスチャの仕組み

SAN では、ハンドルから以下の 2 枚のテクスチャサーフェスを取得できます。

### ( 1 ) 映像サーフェス

矩形形式の YUV422 テクスチャ。

### ( 2 ) マスクサーフェス

Twiddle 形式の 8bit/pixel のパレットテクスチャ。

### < サーフェスの取得方法 >

<code>SAN_LoadTex(san, 0);</code>	<code>/* テクスチャ RAM への転送</code>	<code>*/</code>
<code>SAN_GetVideoPic(san, 0, &amp;pic);</code>	<code>/* 映像サーフェスの取得</code>	<code>*/</code>
<code>user_set_video_srf(pic.srf);</code>	<code>/* 映像サーフェスの貼り付け</code>	<code>*/</code>
<code>SAN_GetMskPic(san, 0, &amp;pic);</code>	<code>/* マスクサーフェスの取得</code>	<code>*/</code>
<code>user_set_msk_srf(pic.srf);</code>	<code>/* マスクサーフェスの貼り付け</code>	<code>*/</code>
<code>/* ポリゴンの描画 */</code>		

pic.srf は、Kamui ドライバの割り当てたサーフェスです。



### <Ninja ライブラリによる SAN テクスチャ>

njSetMvSurface 関数によって、SAN のサーフェスをテクスチャリスト内の指定された番号のテクスチャに登録することができます。njSetMvSurface 関数は、ミドルウェアのサンプルプログラム内に定義しています。

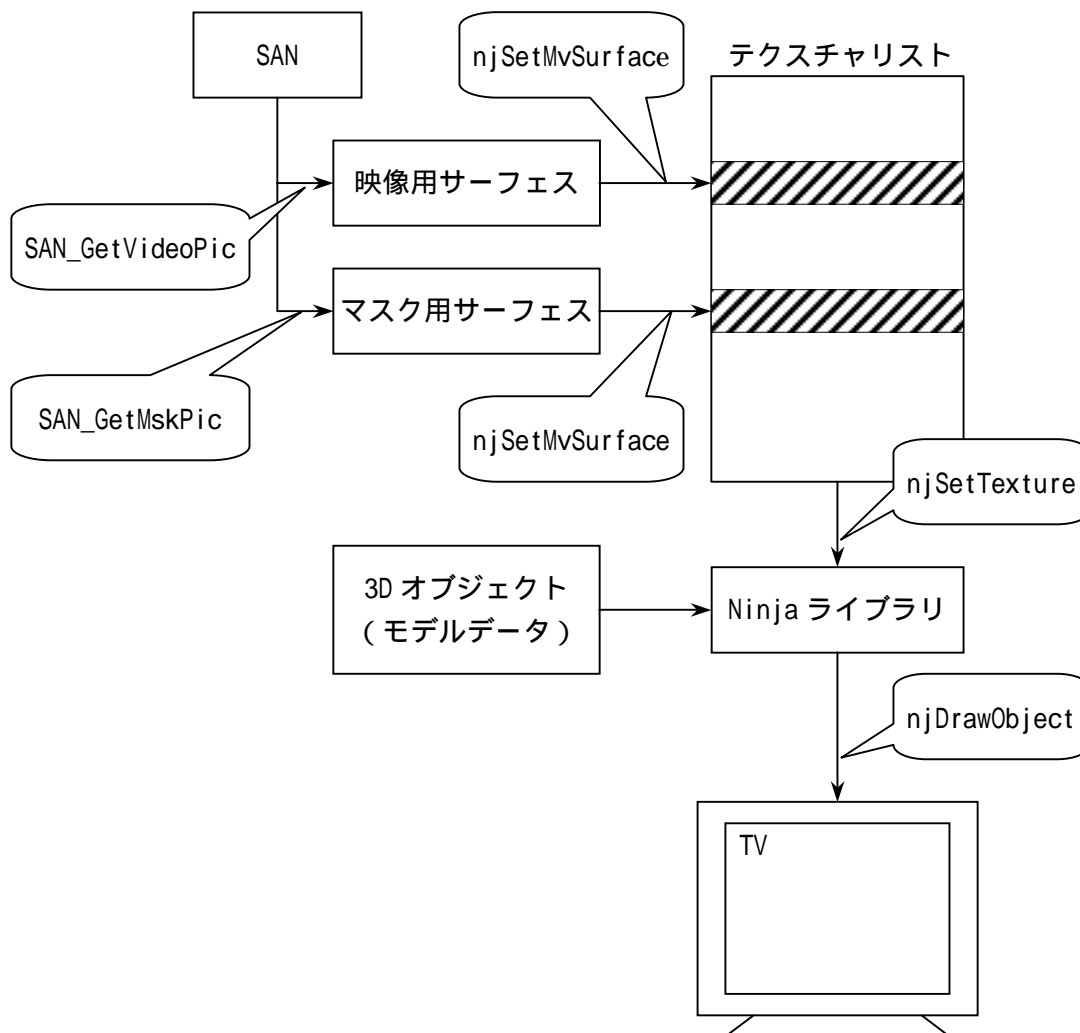


図 5 - 2 Ninja による SAN テクスチャ

## 6. 合 成

### 6.1 合成機能

SAN は、ビジュアルエフェクトのための合成機能を持っています。合成方法として、以下の方法があります。

#### ( 1 ) 不透明表示

映像を合成せずに描画します。

#### ( 2 ) 半透明合成

ウィンドウ単位でアルファブレンディングします。

#### ( 3 ) 加算合成

ポリゴンの RGB 値と SAN の RGB 値を単純に加算します。

#### ( 4 ) アルファ合成

ポリゴンの RGB 値と SAN の RGB 値をピクセル単位でアルファブレンディングします。

現バージョンでは、フルアルファ合成、256 段階の 値を持つ合成のみ対応しています。

### <アルファブレンディング>

アルファブレンディングとは、混合値 により以下のように表現されます。

$$\text{output} = \text{混合値} \times s + (1 - \text{混合値}) \times d$$

output : 合成後の RGB 値

混合値 : 混合比率

s : SAN の RGB 値

d : ポリゴンの RGB 値

### 6.2 アルファ合成の仕組み

アルファ合成は、2 枚の半透明ポリゴンを描画することにより実現しています。マスクサーフェスを描画(乗算処理)後、映像サーフェスを加算しています。

以下に、合成の仕組みを示します。

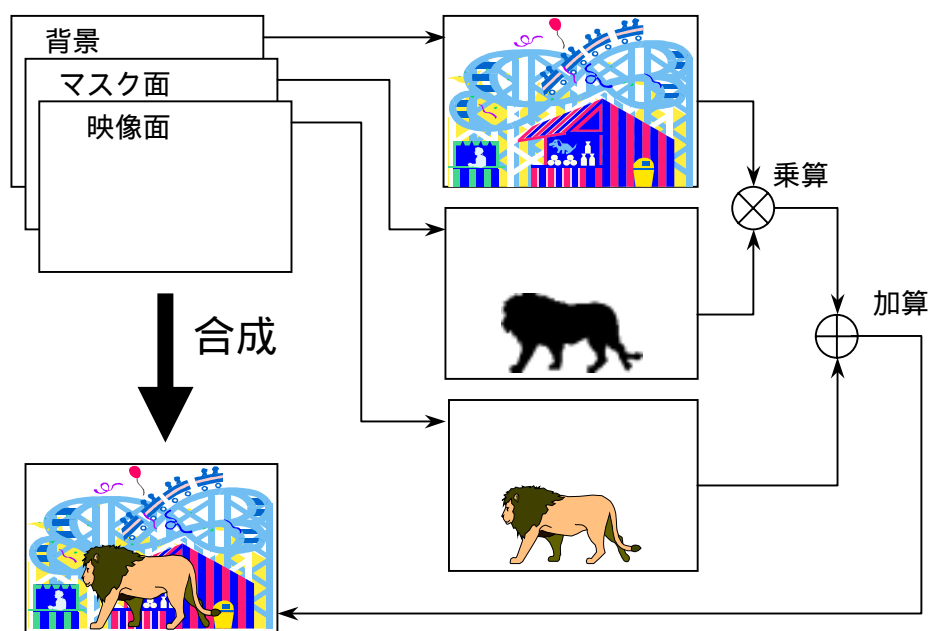


図 6 - 1 合成の仕組み

## 7. データフォーマット

SAN データは、YUV420 データを連結したデータです。bmp2san.exe プログラムにより BMP ファイル (24bit・非圧縮) から作成します。

SAN データのフォーマットを以下に示します。

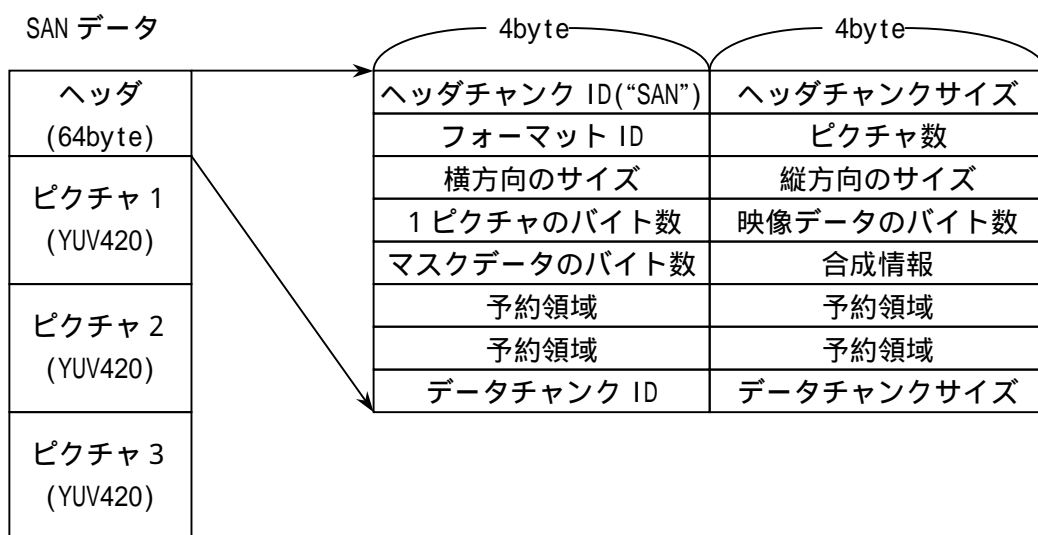


図 7-1 SAN データのフォーマット

## 8. データの作成方法

### 8.1 不透明表示、半透明合成、加算合成

bmp2san.exe プログラムにより、複数の BMP ファイルを YUV420 フォーマットに変換し、連結します。

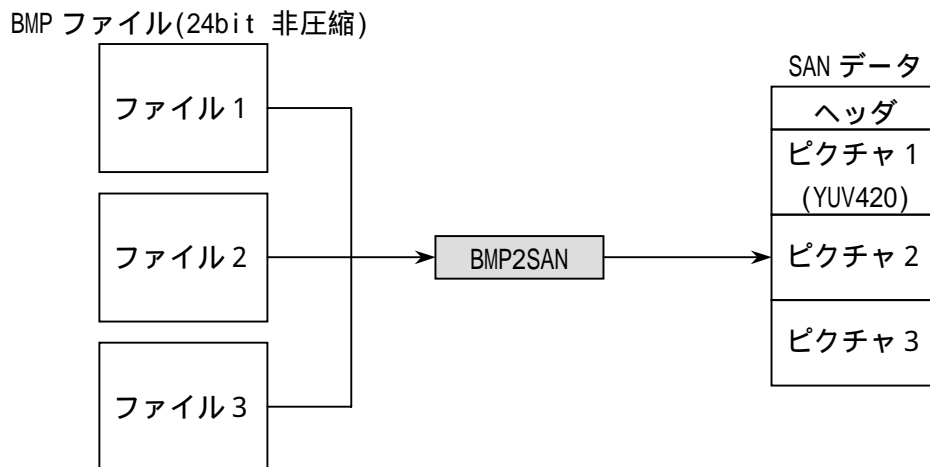


図 8 - 1 SAN データの作成

コマンドプロンプトより、以下のように実行してください。最後の引数が出力ファイルになります。

```
C:¥TEMP>bmp2san file1.bmp file2.bmp file3.bmp sample.san
```

拡張子を省略することもできます。

```
C:¥TEMP>bmp2san file1 file2 file3 sample
```

サブコマンドファイルを使用する場合は以下の通りです。

```
C:¥TEMP>bmp2san -sub=san.sub
```

サブコマンドファイル(san.sub)

```
file1.bmp
file2.bmp
file3.bmp
sample.san
```

以下の手順で、ディレクトリ内の全ての BMP ファイルを SAN データに変換することもできます。

```
C:¥TEMP>dir /b *.bmp > tmp.sub
```

サブコマンドファイル(tmp.sub)

```
C:¥TEMP>bmp2san -sub=tmp.sub sample1.san
```

```
file1.bmp
file2.bmp
file3.bmp
```

#### < 半透明合成の場合 >

以下のように、オプションを追加して下さい。

```
C:¥TEMP>bmp2san -compo=trnsp -sub=tmp.sub sample1.san
```

#### < 加算合成の場合 >

以下のように、オプションを追加して下さい。

```
C:¥TEMP>bmp2san -compo=add -sub=tmp.sub sample1.san
```

## 8.2 アルファ合成

アルファ合成用 SAN データは、映像データとマスクデータをグラフィックツールにより作成してから、bmp2san.exe プログラムにより連結します。マスターデータは、Twiddle 形式の 8bit パレットに変換されます。

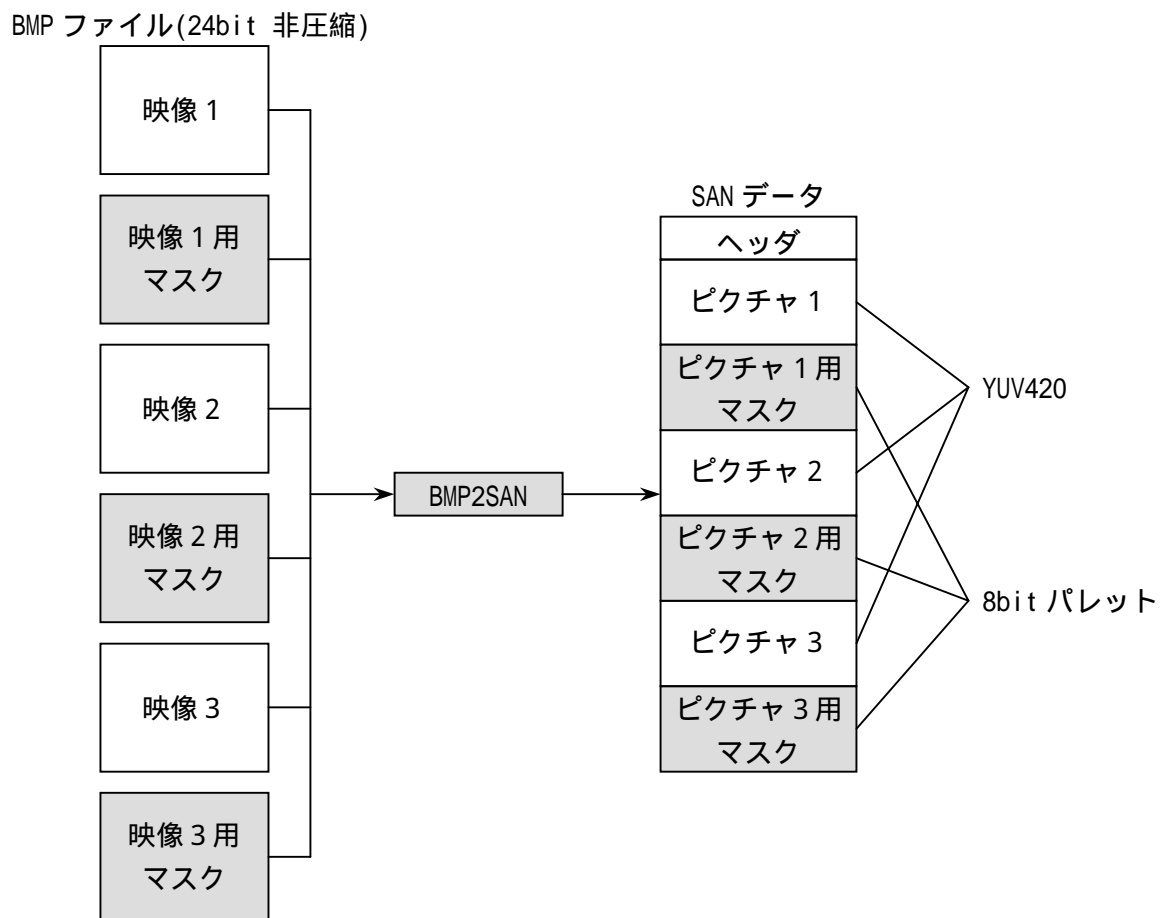


図 8 - 2 アルファ合成用 SAN データの作成

コマンドプロンプトより以下のように実行して下さい。合成モードにフルアルファ(“alph256”)を設定し、BMP ファイルは映像データ、マスクデータの順に入力します。

```
C:¥TEMP>bmp2san -compo=alph256 file1v.bmp file1a.bmp file2v.bmp file2a.bmp sample.san
```

file1v.bmp, file2v.bmp ... 映像データ

file1a.bmp, file2a.bmp ... マスクデータ

不透明表示の場合と同様に、サブコマンドファイルを指定することもできます。

## 9. データ仕様

ライブラリのデータ一覧を以下に示します。

表 9 - 1 データ一覧

データ名		機 能	番号
データ型			
SAN		SAN ハンドル	1.1
SAN_PIC		サーフェス情報構造体	1.2

## 9.1 データ型

Title	Data Name	Data	No
データ	SAN	SAN ハンドル	1.1

SAN を制御するためのハンドルです。SAN\_Create 関数で生成されます。

Title	Data Name	Data	No
データ	SAN_PIC	サーフェス情報構造体	1.2

SAN をテクスチャとして使用する場合に、グラフィックライブラリにサーフェス情報を渡すための構造体です。

メンバー	型 名	説 明
srf	void *	サーフェス
width	long	有効サーフェスサイズ 幅
height	long	有効サーフェスサイズ 高さ

## 10. 関数仕様

ライブラリの関数一覧を以下に示します。

表 10-1 関数一覧

関数名	機 能	番号
<b>初期化と終了処理</b>		
SAN_Init	初期化	1.1
SAN_Finish	終了処理	1.2
SAN_InitKm	初期化(Kamui 用)	1.3
SAN_FinishKm	終了処理(Kamui 用)	1.4
SAN_SetVertexBuffer	頂点バッファの設定(Kamui 用)	1.5
<b>基本動作処理</b>		
SAN_CalcWorkSize	作業領域サイズの計算	2.1
SAN_CalcWorkSizeAlph	作業領域サイズの計算(合成用)	2.2
SAN_Create	ハンドルの生成	2.3
SAN_Destroy	ハンドルの消去	2.4
SAN_LoadTex	YUV420 データの V-RAM への転送	2.5
SAN_Draw	テクスチャの表示	2.6
SAN_GetNumPic	ピクチャ数の取得	2.7
<b>表示制御</b>		
SAN_SetDispPos	表示位置の設定	3.1
SAN_SetDispSize	表示サイズの設定	3.2
SAN_SetDispZ	表示スクリーンの奥行き値の設定	3.3
SAN_GetDispZ	表示スクリーンの奥行き値の取得	3.4
SAN_SetDispBright	輝度の設定	3.5
SAN_GetDispBright	輝度の取得	3.6
SAN_SetDispBrightOfst	輝度オフセットの設定	3.7
SAN_GetDispBrightOfst	輝度オフセットの取得	3.8
<b>サーフェス制御</b>		
SAN_CalcSrfBufSize	サーフェスポイント用バッファサイズの計算	4.1
SAN_SetSrfPntBuf	サーフェスポイント用バッファの設定	4.2
SAN_SetSrfPos	表示位置の設定	4.3
SAN_GetSrfPos	表示位置の取得	4.4
SAN_SetSrfBright	輝度の設定	4.5
SAN_GetSrfBright	輝度の取得	4.6
SAN_SetSrfBrightOfst	輝度オフセットの設定	4.7
SAN_GetSrfBrightOfst	輝度オフセットの取得	4.8
SAN_SetImgPos	イメージ位置の設定	4.9
SAN_GetImgPos	イメージ位置の取得	4.10
SAN_GetImgSize	イメージサイズの取得	4.11
<b>テクスチャ</b>		
SAN_GetVideoPic	ビデオピクチャの取得	5.1
SAN_GetMskPic	マスクピクチャの取得	5.2
<b>エラー処理</b>		
SAN_EntryErrFunc	エラーコールバック関数の登録	6.1



## 1 0.1 初期化と終了処理

Title	Function Name	Function	No
関 数	SAN_Init	初期化	1.1

[書 式] void SAN\_Init(void);

[入 力] なし

[出 力] なし

[関数値] なし

[機 能] ライブラリを初期化します。

Title	Function Name	Function	No
関 数	SAN_Finish	終了処理	1.2

[書 式] void SAN\_Finish(void);

[入 力] なし

[出 力] なし

[関数値] なし

[機 能] ライブラリの終了処理をします。

Title 関 数	Function Name SAN_InitKm	Function 初期化(Kamui 用)	No 1.3
--------------	-----------------------------	--------------------------	-----------

[書 式] void SAN\_InitKm(void);

[入 力] なし

[出 力] なし

[関数値] なし

[機 能] ライブラリを初期化します。

[備 考] グラフィックライブラリとして Ninja を使用せず、Kamui のみを使用する場合の初期化時に使用します。

Title 関 数	Function Name SAN_FinishKm	Function 終了処理(Kamui 用)	No 1.4
--------------	-------------------------------	---------------------------	-----------

[書 式] void SAN\_FinishKm(void);

[入 力] なし

[出 力] なし

[関数値] なし

[機 能] ライブラリの終了処理をします。

[備 考] グラフィックライブラリとして Ninja を使用せず、Kamui のみを使用する場合の終了処理時に使用します。

Title	Function Name	Function	No
関 数	SAN_SetVertexBuffer	頂点バッファの設定 (Kamui 用)	1.5

[ 書 式 ] void SAN\_SetVertexBuffer(void \*vbuf);

[ 入 力 ] vbuf : 頂点バッファ

[ 出 力 ] なし

[ 関数値 ] なし

[ 機 能 ] 頂点バッファを設定します。

[ 備 考 ] SAN\_InitKm 関数で初期化した場合に使用します。SAN\_Init 関数は、内部で頂点バッファの設定を行っているので、SAN\_Init 関数で初期化した場合は使用しないでください。

## 1 0.2 基本動作処理

Title	Function Name	Function	No
関 数	SAN_CalcWorkSize	作業領域サイズの計算	2.1

[ 書 式 ] long SAN\_CalcWorkSize(long ntex);

[ 入 力 ] ntex : テクスチャバッファ数

[ 出 力 ] なし

[ 関数値 ] 作業領域のサイズ(単位 : バイト)

[ 機 能 ] 作業領域のサイズを計算します。

Title	Function Name	Function	No
関 数	SAN_CalcWorkSizeAlph	作業領域サイズの計算(合成用)	2.2

[ 書 式 ] long SAN\_CalcWorkSizeAlph(long ntex, long alph\_flg);

[ 入 力 ] ntex : テクスチャバッファ数

alph\_flg : 合成するか否かのフラグ( 1 : 合成する、 0 : 合成しない)

[ 出 力 ] なし

[ 関数値 ] 作業領域のサイズ(単位 : バイト)

[ 機 能 ] 作業領域のサイズを計算します。

Title 関 数	Function Name SAN_Create	Function ハンドルの生成	No 2.3
--------------	-----------------------------	---------------------	-----------

[ 書 式 ] SAN SAN\_Create(void \*sandat, long ntex, void \*wk);

[ 入 力 ] sandat : SAN データ

ntex : テクスチャバッファ数

wk : 作業領域

[ 出 力 ] なし

[ 関数値 ] SAN ハンドル

[ 機 能 ] ハンドルを生成します。

Title 関 数	Function Name SAN_Destroy	Function ハンドルの消去	No 2.4
--------------	------------------------------	---------------------	-----------

[ 書 式 ] void SAN\_Destroy(SAN san);

[ 入 力 ] san : SAN ハンドル

[ 出 力 ] なし

[ 関数値 ] なし

[ 機 能 ] ハンドルを消去します。

Title 関 数	Function Name SAN_LoadTex	Function YUV420 データの V-RAM への転送	No 2.5
--------------	------------------------------	------------------------------------	-----------

[ 書 式 ] void SAN\_LoadTex(SAN san, long sno, long dno);

[ 入 力 ] san : SAN ハンドル  
sno : ピクチャ番号  
dno : テクスチャバッファ番号

[ 出 力 ] なし

[ 関数値 ] なし

[ 機 能 ] YUV420 データをテクスチャバッファ(V-RAM)に転送します。

[ 備 考 ] ピクチャ番号は、0 ~ SAN データ内のピクチャ数 - 1 までの範囲です。SAN データ内のピクチャ数は SAN\_GetNumPic 関数で取得できます。テクスチャバッファ番号は、0 ~ テクスチャバッファ数 - 1 です。テクスチャバッファ数は、SAN\_CalcWorkSize 関数、SAN\_Create 関数の引数 ntex です。

Title 関 数	Function Name SAN_Draw	Function テクスチャの表示	No 2.6
--------------	---------------------------	----------------------	-----------

[ 書 式 ] void SAN\_Draw(SAN san, long dno);

[ 入 力 ] san : SAN ハンドル  
dno : テクスチャバッファ番号

[ 出 力 ] なし

[ 関数値 ] なし

[ 機 能 ] テクスチャを表示します。

[ 備 考 ] テクスチャバッファ番号は、0 ~ テクスチャバッファ数 - 1 です。テクスチャバッファ数は、SAN\_CalcWorkSize 関数、SAN\_Create 関数の引数 ntex です。

Title 関 数	Function Name SAN_GetNumPic	Function SAN データ内のピクチャ数の取得	No 2.7
--------------	--------------------------------	-------------------------------	-----------

[ 書 式 ] long SAN\_GetNumPic(SAN san);

[ 入 力 ] san : SAN ハンドル

[ 出 力 ] なし

[ 関数値 ] ピクチャ数

[ 機 能 ] SAN データ内のピクチャ数(連結したファイル数)を取得します。

### 1 0.3 表示制御

Title	Function Name	Function	No
関 数	SAN_SetDispPos	表示位置の設定	3.1

[ 書 式 ] void SAN\_SetDispPos(SAN san, float lx, float ly);

[ 入 力 ] san : SAN ハンドル

lx : X 座標

ly : Y 座標

[ 出 力 ] なし

[ 関数値 ] なし

[ 機 能 ] 表示位置を設定します。

Title	Function Name	Function	No
関 数	SAN_SetDispSize	表示サイズの設定	3.2

[ 書 式 ] void SAN\_SetDispSize(SAN san, float sx, float sy);

[ 入 力 ] san : SAN ハンドル

sx : X 座標

sy : Y 座標

[ 出 力 ] なし

[ 関数値 ] なし

[ 機 能 ] 表示サイズを設定します。



Title 関 数	Function Name	Function	No
	SAN_SetDispZ	表示スクリーンの奥行き値の設定	3.3

[書 式] void SAN\_SetDispZ(SAN san, float z);  
 [入 力] san : SAN ハンドル  
           z : 奥行き値 1.0(最手前)65536.0(最奥)  
 [出 力] なし  
 [関数値] なし  
 [機 能] 表示スクリーンの奥行き値を設定します。

Title 関 数	Function Name	Function	No
	SAN_GetDispZ	表示スクリーンの奥行き値の取得	3.4

[書 式] float SAN\_GetDispZ(SAN san);  
 [入 力] san : SAN ハンドル  
 [出 力] なし  
 [関数値] 奥行き値 1.0(最手前)65536.0(最奥)  
 [機 能] 表示スクリーンの奥行き値を取得します。

Title 関 数	Function Name	Function	No
	SAN_SetDispBright	輝度の設定	3.5

[書 式] void SAN\_SetDispBright(SAN san, long val);

[入 力] san : SAN ハンドル  
val : 輝度 (0~255)

[出 力] なし

[関数値] なし

[機 能] 輝度を設定します。

[備 考] デフォルト値は、224 が設定されています。これは、Sofdec のデフォルト値と同じです。  
Dreamcast の映像出力は、255 で 110IRE の輝度となるため、デフォルトで 224 を設定しています。素材によっては、この値を調整する必要があります。

Title 関 数	Function Name	Function	No
	SAN_GetDispBright	輝度の取得	3.6

[書 式] long SAN\_GetDispBright(SAN san);

[入 力] san : SAN ハンドル

[出 力] なし

[関数値] 輝度 (0~255)

[機 能] 輝度を取得します。

Title 関 数	Function Name SAN_SetDispBrightOfst	Function 輝度オフセットの設定	No 3.7
--------------	--	------------------------	-----------

[書 式] void SAN\_SetDispBrightOfst(SAN san, long val);

[入 力] san : SAN ハンドル  
val : 輝度オフセット (0~255)

[出 力] なし

[関数値] なし

[機 能] 輝度オフセットを設定します。

[備 考] デフォルトで、6 が設定されています。これは、Sofdec のデフォルト値と同じです。

CG などでは、黒がつぶれる傾向があるため、デフォルトで 6 が設定されています。

素材によっては、この値を調整する必要があります。この値を 255 から徐々に減少させることにより、フェードインさせることができます。また、反対に 255 まで徐々に増加させることにより、フェードアウトさせることができます。

Title 関 数	Function Name SAN_GetDispBrightOfst	Function 輝度オフセットの取得	No 3.8
--------------	--	------------------------	-----------

[書 式] long SAN\_GetDispBrightOfst(SAN san);

[入 力] san : SAN ハンドル

[出 力] なし

[関数値] 輝度オフセット (0~255)

[機 能] 輝度オフセットを取得します。

#### 1 0.4 サーフェス制御

Title 関 数	Function Name	Function	No
	SAN_CalcSrfBufSize	サーフェスポイント用バッファサイズの 計算	4.1

[ 書 式 ] long SAN\_CalcSrfBufSize(SAN san, long npnt);

[ 入 力 ] san : SAN ハンドル

npnt : サーフェスポイントの数

[ 出 力 ] なし

[ 関数値 ] サーフェスポイント用バッファサイズ(単位 : バイト)

[ 機 能 ] サーフェスポイント用のバッファサイズを取得します。

Title 関 数	Function Name	Function	No
	SAN_SetSrfPntBuf	サーフェスポイント用バッファの設定	4.2

[ 書 式 ] void SAN\_SetSrfPntBuf(SAN san, long npnt, void \*buf, long bsize);

[ 入 力 ] san : SAN ハンドル

npnt : サーフェスポイントの数

buf : バッファ

bsize : バッファサイズ(単位 : バイト)

[ 出 力 ] なし

[ 関数値 ] なし

[ 機 能 ] 指定されたバッファをサーフェスポイント用バッファに設定します。

Title 関 数	Function Name	Function	No
	SAN_SetSrfPos	表示位置の設定	4.3

[書 式] void SAN\_SetSrfPos(SAN san, unsigned long no, float lx, float ly, float lz);

[入 力] san : SAN ハンドル

no : 頂点番号(“Z”順に 0 ~ 3)

lx : X 座標

ly : Y 座標

lz : Z 座標(1.0(最手前) ~ 65536.0(最奥))

[出 力] なし

[関数値] なし

[機 能] 表示するサーフェスの位置を各頂点ごとに設定します。

Title 関 数	Function Name	Function	No
	SAN_GetSrfPos	表示位置の取得	4.4

[書 式] void SAN\_GetSrfPos(SAN san, unsigned long no, float \*lx, float \*ly, float \*lz);

[入 力] san : SAN ハンドル

no : 頂点番号(“Z”順に 0 ~ 3)

[出 力] lx : X 座標

ly : Y 座標

lz : Z 座標(1.0(最手前) ~ 65536.0(最奥))

[関数値] なし

[機 能] サーフェスの位置を各頂点ごとに取得します。

Title 関 数	Function Name SAN_SetSrfBright	Function 輝度の設定	No 4.5
--------------	-----------------------------------	-------------------	-----------

[ 書 式 ] void SAN\_SetSrfBright(SAN san, unsigned long no,  
float a, float r, float g, float b);

[ 入 力 ] san : SAN ハンドル  
no : 頂点番号(“Z”順に 0 ~ 3)  
a : アルファ値(0.0~1.0)  
r : 赤の成分値(0.0~1.0)  
g : 緑の成分値(0.0~1.0)  
b : 青の成分値(0.0~1.0)

[ 出 力 ] なし

[ 関数値 ] なし

[ 機 能 ] 各頂点ごとに輝度を設定します。

Title 関 数	Function Name SAN_GetSrfBright	Function 輝度の取得	No 4.6
--------------	-----------------------------------	-------------------	-----------

[ 書 式 ] void SAN\_GetSrfBright(SAN san, unsigned long no,  
float \*a, float \*r, float \*g, float \*b);

[ 入 力 ] san : SAN ハンドル  
no : 頂点番号(“Z”順に 0 ~ 3)  
[ 出 力 ] a : アルファ値(0.0~1.0)  
r : 赤の成分値(0.0~1.0)  
g : 緑の成分値(0.0~1.0)  
b : 青の成分値(0.0~1.0)

[ 関数値 ] なし

[ 機 能 ] 各頂点ごとに輝度値を取得します。

Title 関 数	Function Name	Function	No
	SAN_SetSrfBrightOfst	輝度オフセットの設定	4.7

[ 書 式 ] void SAN\_SetSrfBrightOfst(SAN san, unsigned long no,  
float a, float r, float g, float b);

[ 入 力 ] san : SAN ハンドル  
no : 頂点番号(“Z”順に 0 ~ 3)  
a : アルファ値(0.0~1.0)  
r : 赤の成分値(0.0~1.0)  
g : 緑の成分値(0.0~1.0)  
b : 青の成分値(0.0~1.0)

[ 出 力 ] なし

[ 関数値 ] なし

[ 機 能 ] 各頂点ごとに輝度オフセットを設定します。

Title 関 数	Function Name	Function	No
	SAN_GetSrfBrightOfst	輝度オフセットの取得	4.8

[ 書 式 ] void SAN\_GetSrfBrightOfst(SAN san, unsigned long no,  
float \*a, float \*r, float \*g, float \*b);

[ 入 力 ] san : SAN ハンドル  
no : 頂点番号(“Z”順に 0 ~ 3)  
[ 出 力 ] a : アルファ値(0.0~1.0)  
r : 赤の成分値(0.0~1.0)  
g : 緑の成分値(0.0~1.0)  
b : 青の成分値(0.0~1.0)

[ 関数値 ] なし

[ 機 能 ] 各頂点ごとの輝度オフセットを取得します。

Title 関 数	Function Name	Function	No
	SAN_SetImgPos	イメージ位置の設定	4.9

[書 式] void SAN\_SetImgPos(SAN san, unsigned long no, float lx, float ly);

[入 力] san : SAN ハンドル  
no : 頂点番号(“Z”順に 0 ~ 3)  
lx : X 座標  
ly : Y 座標

[出 力] なし

[関数値] なし

[機 能] イメージの位置を各頂点ごとに設定します。

Title 関 数	Function Name	Function	No
	SAN_GetImgPos	イメージ位置の取得	4.10

[書 式] void SAN\_GetImgPos(SAN san, unsigned long no, float \*lx, float \*ly);

[入 力] san : SAN ハンドル  
no : 頂点番号(“Z”順に 0 ~ 3)  
[出 力] lx : X 座標  
ly : Y 座標

[関数値] なし

[機 能] 各頂点ごとのイメージ位置を取得します。



Title 関 数	Function Name SAN_GetImgSize	Function イメージサイズの取得	No 4.11
--------------	---------------------------------	------------------------	------------

[ 書 式 ] void SAN\_GetImgSize(SAN san, long \*isx, long \*isy);

[ 入 力 ] san : SAN ハンドル

[ 出 力 ] isx : X 座標

isy : Y 座標

[ 関数値 ] なし

[ 機 能 ] イメージのサイズを取得します。

## 1 0.5 テクスチャ

Title 関 数	Function Name	Function	No
	SAN_GetVideoPic	ビデオピクチャの取得	5.1

[書 式] void SAN\_GetVideoPic(SAN san, long dno, SAN\_PIC \*pic);

[入 力] san : SAN ハンドル

dno : テクスチャバッファ番号

[出 力] pic : サーフェス情報構造体

[関数値] なし

[機 能] テクスチャバッファ番号を指定して、映像サーフェスを取得します。

[備 考] テクスチャバッファ番号は、0 ~ テクスチャバッファ数 - 1 です。テクスチャバッファ数は、SAN\_CalcWorkSize 関数、SAN\_Create 関数の引数 ntex です。

Title 関 数	Function Name	Function	No
	SAN_GetMskPic	マスクピクチャの取得	5.2

[書 式] void SAN\_GetMskPic(SAN san, long dno, SAN\_PIC \*pic);

[入 力] san : SAN ハンドル

dno : テクスチャバッファ番号

[出 力] pic : サーフェス情報構造体

[関数値] なし

[機 能] テクスチャバッファ番号を指定して、マスクサーフェスを取得します。

[備 考] テクスチャバッファ番号は、0 ~ テクスチャバッファ数 - 1 です。テクスチャバッファ数は、SAN\_CalcWorkSize 関数、SAN\_Create 関数の引数 ntex です。

## 1 0.6 エラー処理

Title 関 数	Function Name SAN_EntryErrFunc	Function エラーコールバック関数の登録	No 6.1
--------------	-----------------------------------	----------------------------	-----------

[ 書 式 ] void SAN\_EntryErrFunc(SAN\_ERRFN errfn, void \*obj);

[ 入 力 ] errfn : ユーザのコールバック関数

obj : コールバック関数の第 1 引数

[ 出 力 ] なし

[ 関数値 ] なし

[ 機 能 ] エラーコールバック関数を登録します。エラーが発生すると、登録されたコールバック関数が以下の形式で呼び出されます。

```
(*func)(void *obj, char *msg);
```

この関数の第 1 引数 obj は、SAN\_EntryErrFunc 関数の第 2 引数となります。また、第 2 引数 msg は、エラーメッセージです。この関数は、デバッグ用の関数なので、アプリケーションのマスタアップ時には、何もしない関数に置き換えてください。