システムマクロトレース API ライブラリ
リファレンスマニュアル
RTOS 編

株式会社ＤＴＳインサイト

システムマクロトレース API ライブラリ リファレンスマニュアル

発行：株式会社ＤＴＳインサイト

改訂履歴

第 1 版	2012 年 10 月	新規発行

目次

1 はじめに

システムマクロトレース API ライブラリ（以下、API ライブラリ）は、システムマクロトレース出力ポートにデバッグ情報を出力するためのライブラリです。

1.1 API ライブラリリファレンスマニュアルとは

本 API ライブラリをターゲットプログラムに組み込み、任意の関数内にデバッグ関数を挿入し、トレース情報を収集、解析することができます。

本マニュアルでは、

- API ライブラリをご使用いただくために
- API ライブラリ関数リファレンス
- API ライブラリ使用上の注意事項
- API ライブラリの便利な使用方法

以上について記載します。

システムマクロトレース API ライブラリの組み込み方法は、
「システムマクロトレース ポーティングガイド」をご覧ください。

1.2 API ライブラリ関数のヘッダファイル

デバッグ情報を出力したいソースにライブラリ関数の呼び出しを追加してください。
また、使用するライブラリ関数に対応したヘッダファイルをインクルードしてください。

ライブラリ関数仕様については、「2 API ライブラリ関数リファレンス」をご覧ください。

下の API ライブラリヘッダファイルをソースファイル中にインクルードしてください。
（本ファイルは各種 API ライブラリ用ヘッダファイルを統合したものですので、このファイル以外のインクルード記述は不要です）

ファイル名	内容
SMTAPI.h	システムマクロトレース API ライブラリ用ヘッダファイル

ソースファイル記述例

```
#include "SMTAPI.h"
```

2 API ライブラリ関数リファレンス

API ライブラリで提供する関数には、以下のグループに分類されます。

- デバッグプリント文制御関数
- デバッグプリント関数
- ポートアウト関数

関数一覧

グループ	関数名	ヘッダファイル	機能説明
デバッグプリント文制御	_SMT_GetDebugLevel	SMTAPI.h	デバッグプリント出力レベルマスク値参照
	_SMT_SetDebugLevel	SMTAPI.h	デバッグプリント出力レベルマスク値設定
デバッグプリント	_SMT_Puts	SMTAPI.h	puts 相当。 文字列を出力します。
	_SMT_Printf	SMTAPI.h	printf 相当。 書式指定付き文字列を出力します。
	_SMT_UsrMsgTag0 _SMT_UsrMsgTag1 _SMT_UsrMsgTag2 _SMT_UsrMsgTag3 _SMT_UsrMsgTag4	SMTAPI.h	ユーザー定義メッセージタグ出力
ポートアウト	_SMT_PortOut	SMTAPI.h	ポート情報出力

グループ	関数名	ヘッダファイル	機能説明
コンテキストスイッチ	_SMT_OsSwitch_Process	SMTAPI.h	プロセスのスイッチ情報出力
	_SMT_OsSwitch_ThreadProcess	SMTAPI.h	スレッドおよびプロセスのスイッチ情報出力
	_SMT_OsSwitch_Process_Name	SMTAPI.h	プロセスのスイッチ情報、名称の出力
	_SMT_OsSwitch_ThreadProcess_Name	SMTAPI.h	スレッドおよびプロセスのスイッチ情報、名称の出力
	_SMT_OsSwitch_Irq_in	SMTAPI.h	割込み処理開始の情報出力
	_SMT_OsSwitch_Irq_out	SMTAPI.h	割込み処理終了の情報出力
	_SMT_OsSwitch_Idle	SMTAPI.h	アイドルへのスイッチ情報出力
システムコール	_SMT_OsCall0	SMTAPI.h	システムコール処理開始、終了の情報出力（引数なし）
	_SMT_OsCall1	SMTAPI.h	同上（引数 1 個）
	_SMT_OsCall2	SMTAPI.h	同上（引数 2 個）
	_SMT_OsCall3	SMTAPI.h	同上（引数 3 個）
	_SMT_OsCall4	SMTAPI.h	同上（引数 4 個）
	_SMT_OsCall5	SMTAPI.h	同上（引数 5 個）
	_SMT_OsCall6	SMTAPI.h	同上（引数 6 個）
	_SMT_OsCall7	SMTAPI.h	同上（引数 7 個）
	_SMT_OsCall8	SMTAPI.h	同上（引数 8 個）
	_SMT_OsCall9	SMTAPI.h	同上（引数 9 個）
	_SMT_OsCall10	SMTAPI.h	同上（引数 10 個）
	_SMT_OsCall11	SMTAPI.h	同上（引数 11 個）
	_SMT_OsCall12	SMTAPI.h	同上（引数 12 個）
	_SMT_OsCall13	SMTAPI.h	同上（引数 13 個）
	_SMT_OsCall14	SMTAPI.h	同上（引数 14 個）
	_SMT_OsCall15	SMTAPI.h	同上（引数 15 個）
	_SMT_OsCall16	SMTAPI.h	同上（引数 16 個）

2.1 デバッグプリント文制御関数

デバッグプリント文制御関数は、デバッグプリント文のマスク設定値の設定・取得を行います。

2.1.1 _SMT_GetDebugLevel 関数

■機能

現在のデバッグプリント出力レベルマスク設定値を返します。

■呼び出し手順

```
#include "SMTAPI.h"

int level;
level=_SMT_GetDebugLevel();
```

■引数

No.	名前	型	意味
-	-	-	-

■戻り値

型 : int
現在のデバッグプリント出力レベルマスク設定値

_SMT_GetDebugLevel 関数は、現在のデバッグプリント出力レベルマスク設定値を返します。

2.1.2 _SMT_SetDebugLevel 関数

■機能

デバッグプリント出力レベルマスク値を設定します。

■呼び出し手順

```
#include "SMTAPI.h"

int level = 0;

_SMT_SetDebugLevel(level);
```

■引数

No.	名前	型	意味
1	level	int	デバッグプリント出力レベルマスク設定値

■戻り値

型　：　void

_SMT_SetDebugLevel 関数は、デバッグプリント出力レベルマスク設定値を設定します。

2.2 デバッグプリント関数

デバッグプリント関数は、デバッグ用の文字列を出力するための関数です。

2.2.1 _SMT_Puts 関数

■機能

システムマクロトレース上に文字列を出力します。

■呼び出し手順

```
#include "SMTAPI.h"

const char *s = "Debug Message";
int level = 0;

_SMT_Puts(level, s);
```

■引数

No.	名前	型	意味
1	level	int 型	デバッグプリント出力レベル値
2	s	const char 型を指すポインタ	出力する文字列へのポインタ

■戻り値

型 ： int
実行結果を返します。
_SMT_OK（正常終了）又は_SMT_NG（異常終了）を返します。

_SMT_Puts 関数はシステムマクロトレース上に文字列を出力します。複雑な書式変換を行わない分、コードサイズ、実行速度の面で、_SMT_Printf 関数より有利です。

出力可能な文字列長は、終端文字を含め 124byte 以内としてください。
これより長い文字列を指定した場合の動作は保証しません。

2.2.2 _SMT_Printf 関数

■機能

データを書式に従って変換し、システムマクロトレース上に文字列を出力します。

■呼び出し手順

```
#include "SMTAPI.h"

const char *format;
int level = 0;

_SMT_Printf(level, format, [arg]…);
```

■引数

No.	名前	型	意味
1	level	int 型	デバッグ文出力レベル値
2	format	const char 型を指すポインタ	書式を示す文字列へのポインタ
3	arg	書式に従った型	書式に従って出力されるデータ

■戻り値

型 ： int
実行結果を返します。
_SMT_OK（正常終了）又は_SMT_NG（異常終了）を返します。

_SMT_Printf 関数は、format が示す書式を文字列に従って、引数 arg を変換し、トレース情報として出力します。
_SMT_Printf 関数では、書式変換に標準ライブラリ関数の vsprintf 関数を使用しています。
書式の仕様は標準ライブラリ関数の仕様をご覧ください。

出力可能な文字列長は、書式変換後の長さで、終端文字を含め 124byte 以内としてください。
これより長い文字列を指定した場合の動作は保証しません。

2.2.3 _SMT_UsrMsgTag0 関数

■機能

ユーザー定義メッセージタグ番号をシステムマクロトレース情報として出力します。

■呼び出し手順

```
#include "SMTAPI.h"

int level = 0;
unsigned long TagNum = 0x123;

_SMT_UsrMsgTag0(level, TagNum);
```

■引数

No.	名前	型	意味
1	level	int 型	デバッグ文出力レベル値
2	TagNum	unsigned long 型	メッセージタグ番号 0～1023

■戻り値

型 ： int
実行結果を返します。
_SMT_OK（正常終了）又は_SMT_NG（異常終了）を返します。

_SMT_UserTagMsg0 関数は、ユーザー任意のタグ番号を出力します。
タグ番号に対応したユーザー定義メッセージを表示できます。

2.2.4 _SMT_UsrMsgTag1 関数

■機能

ユーザー定義メッセージタグ番号と 1 つのデータをシステムマクロトレース情報として出力します。

■呼び出し手順

```
#include "SMTAPI.h"

int level = 0;
unsigned long TagNum = 0x123;

_SMT_UsrMsgTag1(level,TagNum,arg1);
```

■引数

No.	名前	型	意味
1	level	int 型	デバッグ文出力レベル値
2	TagNum	unsigned long 型	メッセージタグ番号 0～1023
3	arg1	unsigned long 型	書式に従って出力されるデータ

■戻り値

型 ： int
実行結果を返します。
_SMT_OK（正常終了）又は_SMT_NG（異常終了）を返します。

_SMT_UserTagMsg1 関数は、ユーザー任意のタグ番号とデータを出力します。
タグ番号に対応したユーザー定義メッセージを表示できます。

2.2.5 _SMT_UsrMsgTag2 関数

■機能

ユーザー定義メッセージタグ番号と 2 つのデータをシステムマクロトレース情報として出力します。

■呼び出し手順

```
#include "SMTAPI.h"

int level = 0;
unsigned long TagNum = 0x123;

_SMT_UsrMsgTag2(level, TagNum,arg1,arg2);
```

■引数

No.	名前	型	意味
1	level	int 型	デバッグ文出力レベル値
2	TagNum	unsigned long 型	メッセージタグ番号 0～1023
3	arg1	unsigned long 型	書式に従って出力されるデータ
4	arg2	unsigned long 型	書式に従って出力されるデータ

■戻り値

型 ： int
実行結果を返します。
_SMT_OK（正常終了）又は_SMT_NG（異常終了）を返します。

_SMT_UserTagMsg2 関数は、ユーザー任意のタグ番号とデータを出力します。
タグ番号に対応したユーザー定義メッセージを表示できます。

2.2.6 _SMT_UsrMsgTag3 関数

■機能

ユーザー定義メッセージタグ番号と 3 つのデータをシステムマクロトレース情報として出力します。

■呼び出し手順

```
#include "SMTAPI.h"

int level = 0;
unsigned long TagNum = 0x123;

_SMT_UsrMsgTag3(level,TagNum,arg1,arg2,arg3);
```

■引数

No.	名前	型	意味
1	level	int 型	デバッグ文出力レベル値
2	TagNum	unsigned long 型	メッセージタグ番号 0～1023
3	arg1	unsigned long 型	書式に従って出力されるデータ
4	arg2	unsigned long 型	書式に従って出力されるデータ
5	arg3	unsigned long 型	書式に従って出力されるデータ

■戻り値

型 ： int
実行結果を返します。
_SMT_OK（正常終了）又は_SMT_NG（異常終了）を返します。

_SMT_UserTagMsg3 関数は、ユーザー任意のタグ番号とデータを出力します。
タグ番号に対応したユーザー定義メッセージを表示できます。

2.2.7 _SMT_UsrMsgTag4 関数

■機能

ユーザー定義メッセージタグ番号と 4 つのデータをシステムマクロトレース情報として出力します。

■呼び出し手順

```
#include "SMTAPI.h"

int level = 0;
unsigned long TagNum = 0x123;

_SMT_UsrMsgTag4(level,TagNum,arg1,arg2,arg3,arg4);
```

■引数

No.	名前	型	意味
1	level	int 型	デバッグ文出力レベル値
2	TagNum	unsigned long 型	メッセージタグ番号 0～1023
3	arg1	unsigned long 型	書式に従って出力されるデータ
4	arg2	unsigned long 型	書式に従って出力されるデータ
5	arg3	unsigned long 型	書式に従って出力されるデータ
6	arg4	unsigned long 型	書式に従って出力されるデータ

■戻り値

型 ： int
実行結果を返します。
_SMT_OK（正常終了）又は_SMT_NG（異常終了）を返します。

_SMT_UserTagMsg4 関数は、ユーザー任意のタグ番号とデータを出力します。
タグ番号に対応したユーザー定義メッセージを表示できます。

2.2.8 デバッグプリント出力レベル

デバッグプリント関数群は、引数に出力レベル値を持ち、0～15 のレベルを設定することができます。この出力レベル値と、_SMT_SetDebugLevel 関数で設定されるデバッグプリント出力レベルマスクパターンを比較することで、デバッグプリント出力のフィルタリングを実現しています。

デバッグプリント出力レベルマスクパターン値は、ビットパターンとして定義され、各ビット位置がレベル値に対応します。
ビット値が 0 のレベルは出力され、1 のレベルは出力されません。

例えば、以下のようにデバッグプリント出力レベルマスク設定値が 0xFFFC の場合、レベル 0 とレベル 1 として記述されたデバッグプリントのみ出力されます。

ソースコード記述

```
…
_SMT_SetDebugLevel( 0xFFFC );

_SMT_Puts( 0, "Level 0 Message" );
_SMT_Puts( 1, "Level 1 Message" );
_SMT_Puts( 2, "Level 2 Message" );
_SMT_Puts( 3, "Level 3 Message" );
…
```

出力結果

```
Level 0 Message
Level 1 Message
```

デバッグプリント出力レベルマスク設定値は、初期状態では 0x0000 となっており、すべてのレベルが出力有効となっています。この初期値は、SMTUser.h に定義されており変更が可能です(グローバル変数初期化のタイミングでこの定義値で初期化されます)。

```
#define        SMT_LV_MSK     0x0000
```

2.3 ポートアウト関数

ポートアウト関数は、ポートアクセス情報をシステムマクロトレース情報として出力します。

2.3.1 _SMT_PortOut 関数

■機能

ポートアクセス情報を出力します。

■呼び出し手順

```
#include "SMTAPI.h"

unsigned long addr
unsigned char data8;
unsigned short data16;
unsigned long data32;

_SMT_PortOut( addr, data8, _SMT_PSZ8, _SMT_PW);   // 8bit write
_SMT_PortOut( addr, data8, _SMT_PSZ8, _SMT_PR);   // 8bit read
_SMT_PortOut( addr, data16, _SMT_PSZ16, _SMT_PW); // 16bit write
_SMT_PortOut( addr, data16, _SMT_PSZ16, _SMT_PR); // 16bit read
_SMT_PortOut( addr, data32, _SMT_PSZ32, _SMT_PW); // 32bit write
_SMT_PortOut( addr, data32, _SMT_PSZ32, _SMT_PR); // 32bit read
```

■引数

No.	名前	型	意味
1	addr	unsigned long	アドレス値
2	data	unsigned long	データ値
3	size	_SMT_PSZ	ポートサイズ 定義値 _SMT_PSZ8 : 8bit _SMT_PSZ16 : 16bit _SMT_PSZ32 : 32bit
4	rw	_SMT_PRW	アクセスタイプ 定義値 _SMT_PW : ライト _SMT_PR : リード

■戻り値

型 : int

実行結果を返します。

_SMT_OK（正常終了）又は_SMT_NG（異常終了）を返します。

_SMT_PortOut 関数は、ポートアクセス情報を出力します。

ポートサイズ、アクセスタイプの指定には、引数表に記載の定義値を使用してください。

2.4　コンテキストスイッチ関数

プロセスやスレッドの ID または名称、割込みの情報を出力する関数です。
本関数の組み込み方法については、ご購入の『システムマクロトレーススタートアップガイド』を参照願います。
対応オペレーティングシステムにより、一部関数はご利用頂けない場合があります。

2.4.1　_SMT_OsSwitch_Process 関数

■機能

プロセス ID を出力します。

■呼び出し手順

```
#include "SMTAPI.h"

unsigned long processid;

_SMT_OsSwitch_Process( processid );
```

■引数

No.	名前	型	意味
1	processid	unsigned long	プロセス ID 値

■戻り値

型　：　int
実行結果を返します。
_SMT_OK（正常終了）又は_SMT_NG（異常終了）を返します。

2.4.2 _SMT_OsSwitch_ThreadProcess 関数

■機能

スレッド ID およびプロセス ID を出力します。

■呼び出し手順

```
#include "SMTAPI.h"

unsigned long threadid;
unsigned long processid;

_SMT_OsSwitch_ThreadProcess(hreadid, processid);
```

■引数

No.	名前	型	意味
1	threadid	unsigned long	スレッド ID 値
2	processid	unsigned long	プロセス ID 値

■戻り値

型 ： int

実行結果を返します。

_SMT_OK（正常終了）又は_SMT_NG（異常終了）を返します。

2.4.3 _SMT_OsSwitch_Process_Name 関数

■機能

プロセス ID とプロセス名称を出力します。

■呼び出し手順

```
#include "SMTAPI.h"

unsigned long processid;
const char *pname;

_SMT_OsSwitch_Process_Name(processid, pname);
```

■引数

No.	名前	型	意味
1	processid	unsigned long	プロセス ID 値
2	pname	const char*	プロセス名称文字列 （NULL 終端を含め最大 32 文字）

■戻り値

型 ： int

実行結果を返します。

_SMT_OK（正常終了）又は_SMT_NG（異常終了）を返します。

2.4.4 _SMT_OsSwitch_ThreadProcess_Name 関数

■機能

スレッド ID、プロセス ID、スレッド名称およびプロセス名称を出力します。

■呼び出し手順

```
#include "SMTAPI.h"

unsigned long threadid;
unsigned long processid;
const char *tname;
const char *pname;

_SMT_OsSwitch_ThreadProcess_Name(threadid, processid, tname, pname);
```

■引数

No.	名前	型	意味
1	threadid	unsigned long	スレッド ID 値
2	processid	unsigned long	プロセス ID 値
3	tname	const char*	スレッド名称文字列 （NULL 終端を含め最大 32 文字）
4	pname	const char*	プロセス名称文字列 （NULL 終端を含め最大 32 文字）

■戻り値

型 ： int

実行結果を返します。

_SMT_OK（正常終了）又は_SMT_NG（異常終了）を返します。

2.4.5 _SMT_OsSwitch_Irq_in 関数

■機能

割込み処理開始と割込み番号を出力します。

■呼び出し手順

```
#include "SMTAPI.h"

unsigned long irqid;

_SMT_OsSwitch_Irq_in(irqid);
```

■引数

No.	名前	型	意味
1	irqid	unsigned long	割込み番号

■戻り値

型 : int

実行結果を返します。

_SMT_OK（正常終了）又は_SMT_NG（異常終了）を返します。

2.4.6 _SMT_OsSwitch_Irq_out 関数

■機能

割込み処理終了と割込み番号を出力します。

■呼び出し手順

```
#include "SMTAPI.h"

unsigned long irqid;

_SMT_ OsSwitch_Irq_out (irqid);
```

■引数

No.	名前	型	意味
1	irqid	unsigned long	割込み番号

■戻り値

型 : int

実行結果を返します。

_SMT_OK（正常終了）又は_SMT_NG（異常終了）を返します。

2.4.7 _SMT_OsSwitch_Idle 関数

■機能

アイドルプロセスへのスイッチを出力します。

■呼び出し手順

```
#include "SMTAPI.h"

_SMT_ OsSwitch_Idle ();
```

■引数

No.	名前	型	意味
-	-	-	-

■戻り値

型 ： int
実行結果を返します。
_SMT_OK（正常終了）又は_SMT_NG（異常終了）を返します。

2.5　システムコール関数

システムコールの番号、開始終了属性、および引数を出力する関数です。引数は 32bit サイズで、最大 16 個まで出力可能です。
本関数の組み込み方法については、ご購入の『システムマクロトレーススタートアップガイド』を参照願います。
対応オペレーティングシステムにより、一部関数はご利用頂けない場合があります。

2.5.1　_SMT_OsCall0 関数

■機能

システムコール番号、呼び出し属性を出力します。

■呼び出し手順

```
#include "SMTAPI.h"

int osc;

_SMT_OsCall0( osc, _SMT_OS_ATTR_CALL );
_SMT_OsCall0( osc, _SMT_OS_ATTR_RET );
```

■引数

No.	名前	型	意味
1	osc	int	システムコール番号
2	attr	_SMT_OS_CALL	属性 定義値 _SMT_OS_ATTR_CALL : システムコール開始 _SMT_OS_ATTR_RET : システムコール終了

■戻り値

型 : int
実行結果を返します。
_SMT_OK（正常終了）又は_SMT_NG（異常終了）を返します。

2.5.2 _SMT_OsCall1 関数

■機能

システムコール番号、呼び出し属性、引数 1 個を出力します。

■呼び出し手順

```
#include "SMTAPI.h"

int osc;
unsigned long arg1;

_SMT_OsCall1( osc, _SMT_OS_ATTR_CALL,arg1);
_SMT_OsCall1( osc, _SMT_OS_ATTR_RET,arg1);
```

■引数

No.	名前	型	意味
1	osc	int	システムコール番号
2	attr	_SMT_OS_CALL	属性 定義値 _SMT_OS_ATTR_CALL : システムコール開始 _SMT_OS_ATTR_RET : システムコール終了
3	arg1	unsigned long	引数 1

■戻り値

型 : int

実行結果を返します。

_SMT_OK（正常終了）又は_SMT_NG（異常終了）を返します。

2.5.3 _SMT_OsCall2 関数

■機能

システムコール番号、呼び出し属性、引数 2 個を出力します。

■呼び出し手順

```
#include "SMTAPI.h"

int osc;
unsigned long arg1,arg2;

_SMT_OsCall2( osc, _SMT_OS_ATTR_CALL,arg1,arg2);
_SMT_OsCall2( osc, _SMT_OS_ATTR_RET,arg1,arg2);
```

■引数

No.	名前	型	意味
1	osc	int	システムコール番号
2	attr	_SMT_OS_CALL	属性 定義値 _SMT_OS_ATTR_CALL : システムコール開始 _SMT_OS_ATTR_RET : システムコール終了
3	arg1 ～ arg2	unsigned long	引数 1～引数 2

■戻り値

型 : int

実行結果を返します。

_SMT_OK（正常終了）又は_SMT_NG（異常終了）を返します。

2.5.4 _SMT_OsCall3 関数

■機能

システムコール番号、呼び出し属性、引数 3 個を出力します。

■呼び出し手順

```
#include "SMTAPI.h"

int osc;
unsigned long arg1,arg2,arg3;

_SMT_OsCall3( osc, _SMT_OS_ATTR_CALL,arg1,arg2,arg3);
_SMT_OsCall3( osc, _SMT_OS_ATTR_RET,arg1,arg2,arg3);
```

■引数

No.	名前	型	意味
1	osc	int	システムコール番号
2	attr	_SMT_OS_CALL	属性 定義値 _SMT_OS_ATTR_CALL : システムコール開始 _SMT_OS_ATTR_RET : システムコール終了
3	arg1 ～ arg3	unsigned long	引数 1～引数 3

■戻り値

型 : int

実行結果を返します。

_SMT_OK（正常終了）又は_SMT_NG（異常終了）を返します。

2.5.5 _SMT_OsCall4 関数

■機能

システムコール番号、呼び出し属性、引数 4 個を出力します。

■呼び出し手順

```
#include "SMTAPI.h"

int osc;
unsigned long arg1,arg2,arg3,arg4;

_SMT_OsCall4( osc, _SMT_OS_ATTR_CALL,arg1,arg2,arg3,arg4);
_SMT_OsCall4( osc, _SMT_OS_ATTR_RET,arg1,arg2,arg3,arg4);
```

■引数

No.	名前	型	意味
1	osc	int	システムコール番号
2	attr	_SMT_OS_CALL	属性 定義値 _SMT_OS_ATTR_CALL : システムコール開始 _SMT_OS_ATTR_RET : システムコール終了
3	arg1 ～ arg4	unsigned long	引数 1～引数 4

■戻り値

型 : int

実行結果を返します。

_SMT_OK（正常終了）又は_SMT_NG（異常終了）を返します。

2.5.6 _SMT_OsCall5 関数

■機能

システムコール番号、呼び出し属性、引数 5 個を出力します。

■呼び出し手順

```
#include "SMTAPI.h"

int osc;
unsigned long arg1,arg2,arg3,arg4,arg5;

_SMT_OsCall5( osc, _SMT_OS_ATTR_CALL,arg1,arg2,arg3,arg4,arg5);
_SMT_OsCall5( osc, _SMT_OS_ATTR_RET,arg1,arg2,arg3,arg4,arg5);
```

■引数

No.	名前	型	意味
1	osc	int	システムコール番号
2	attr	_SMT_OS_CALL	属性 定義値 _SMT_OS_ATTR_CALL : システムコール開始 _SMT_OS_ATTR_RET : システムコール終了
3	arg1 ～ arg5	unsigned long	引数 1～引数 5

■戻り値

型 : int

実行結果を返します。

_SMT_OK（正常終了）又は_SMT_NG（異常終了）を返します。

2.5.7 _SMT_OsCall6 関数

■機能

システムコール番号、呼び出し属性、引数 6 個を出力します。

■呼び出し手順

```
#include "SMTAPI.h"

int osc;
unsigned long arg1,arg2,arg3,arg4,arg5,arg6;

_SMT_OsCall6( osc, _SMT_OS_ATTR_CALL,arg1,arg2,arg3,arg4,arg5,arg6);
_SMT_OsCall6( osc, _SMT_OS_ATTR_RET,arg1,arg2,arg3,arg4,arg5,arg6);
```

■引数

No.	名前	型	意味
1	osc	int	システムコール番号
2	attr	_SMT_OS_CALL	属性 定義値 _SMT_OS_ATTR_CALL : システムコール開始 _SMT_OS_ATTR_RET : システムコール終了
3	arg1 ～ arg6	unsigned long	引数 1～引数 6

■戻り値

型 : int

実行結果を返します。

_SMT_OK（正常終了）又は_SMT_NG（異常終了）を返します。

2.5.8 _SMT_OsCall7 関数

■機能

システムコール番号、呼び出し属性、引数 7 個を出力します。

■呼び出し手順

```
#include "SMTAPI.h"

int osc;
unsigned long arg1,arg2,arg3,arg4,arg5,arg6,arg7;

_SMT_OsCall7( osc, _SMT_OS_ATTR_CALL,arg1,arg2,arg3,arg4,arg5,arg6,
arg7);

_SMT_OsCall7( osc, _SMT_OS_ATTR_RET,arg1,arg2,arg3,arg4,arg5,arg6,
arg7);
```

■引数

No.	名前	型	意味
1	osc	int	システムコール番号
2	attr	_SMT_OS_CALL	属性 定義値 _SMT_OS_ATTR_CALL : システムコール開始 _SMT_OS_ATTR_RET : システムコール終了
3	arg1 ～ arg7	unsigned long	引数 1～引数 7

■戻り値

型 : int

実行結果を返します。

_SMT_OK（正常終了）又は_SMT_NG（異常終了）を返します。

2.5.9 _SMT_OsCall8 関数

■機能

システムコール番号、呼び出し属性、引数 8 個を出力します。

■呼び出し手順

```
#include "SMTAPI.h"

int osc;
unsigned long arg1,arg2,arg3,arg4,arg5,arg6,arg7,arg8;

_SMT_OsCall8( osc, _SMT_OS_ATTR_CALL,arg1,arg2,arg3,arg4,arg5,arg6
,arg7,arg8);
_SMT_OsCall8( osc, _SMT_OS_ATTR_RET,arg1,arg2,arg3,arg4,arg5,arg6
,arg7,arg8);
```

■引数

No.	名前	型	意味
1	osc	int	システムコール番号
2	attr	_SMT_OS_CALL	属性 定義値 _SMT_OS_ATTR_CALL : システムコール開始 _SMT_OS_ATTR_RET : システムコール終了
3	arg1 ～ arg8	unsigned long	引数 1～引数 8

■戻り値

型 : int

実行結果を返します。

_SMT_OK（正常終了）又は_SMT_NG（異常終了）を返します。

2.5.10 _SMT_OsCall9 関数

■機能

システムコール番号、呼び出し属性、引数 9 個を出力します。

■呼び出し手順

```
#include "SMTAPI.h"

int osc;
unsigned long arg1,arg2,arg3,arg4,arg5,arg6,arg7,arg8,arg9;

_SMT_OsCall9( osc, _SMT_OS_ATTR_CALL,arg1,arg2,arg3,arg4,arg5,arg6
,arg7,arg8,arg9);
_SMT_OsCall9( osc, _SMT_OS_ATTR_RET,arg1,arg2,arg3,arg4,arg5,arg6
,arg7,arg8,arg9);
```

■引数

No.	名前	型	意味
1	osc	int	システムコール番号
2	attr	_SMT_OS_CALL	属性 定義値 _SMT_OS_ATTR_CALL : システムコール開始 _SMT_OS_ATTR_RET : システムコール終了
3	arg1 ～ arg9	unsigned long	引数 1～引数 9

■戻り値

型 : int

実行結果を返します。

_SMT_OK（正常終了）又は_SMT_NG（異常終了）を返します。

2.5.11 _SMT_OsCall10 関数

■機能

システムコール番号、呼び出し属性、引数 10 個を出力します。

■呼び出し手順

```
#include "SMTAPI.h"

int osc;
unsigned long arg1,arg2,arg3,arg4,arg5,arg6,arg7,arg8,arg9,arg10;

_SMT_OsCall10( osc, _SMT_OS_ATTR_CALL,arg1,arg2,arg3,arg4,arg5,arg6
,arg7,arg8,arg9,arg10);
_SMT_OsCall10( osc, _SMT_OS_ATTR_RET,arg1,arg2,arg3,arg4,arg5,arg6
,arg7,arg8,arg9,arg10);
```

■引数

No.	名前	型	意味
1	osc	int	システムコール番号
2	attr	_SMT_OS_CALL	属性 定義値 _SMT_OS_ATTR_CALL : システムコール開始 _SMT_OS_ATTR_RET : システムコール終了
3	arg1 ～ arg10	unsigned long	引数 1～引数 10

■戻り値

型 : int

実行結果を返します。

_SMT_OK（正常終了）又は_SMT_NG（異常終了）を返します。

2.5.12 _SMT_OsCall11 関数

■機能

システムコール番号、呼び出し属性、引数 11 個を出力します。

■呼び出し手順

```
#include "SMTAPI.h"

int osc;
unsigned long arg1,arg2,arg3,arg4,arg5,arg6,arg7,arg8,arg9,arg10
,arg11;

_SMT_OsCall11( osc, _SMT_OS_ATTR_CALL,arg1,arg2,arg3,arg4,arg5,arg6
,arg7,arg8,arg9,arg10,arg11);
_SMT_OsCall11( osc, _SMT_OS_ATTR_RET,arg1,arg2,arg3,arg4,arg5,arg6
,arg7,arg8,arg9,arg10,arg11);
```

■引数

No.	名前	型	意味
1	osc	int	システムコール番号
2	attr	_SMT_OS_CALL	属性 定義値 _SMT_OS_ATTR_CALL : システムコール開始 _SMT_OS_ATTR_RET : システムコール終了
3	arg1 ～ arg11	unsigned long	引数 1～引数 11

■戻り値

型 : int

実行結果を返します。

_SMT_OK（正常終了）又は_SMT_NG（異常終了）を返します。

2.5.13 _SMT_OsCall12 関数

■機能

システムコール番号、呼び出し属性、引数 12 個を出力します。

■呼び出し手順

```
#include "SMTAPI.h"

int osc;
unsigned long arg1,arg2,arg3,arg4,arg5,arg6,arg7,arg8,arg9,arg10
,arg11,arg12;

_SMT_OsCall12( osc, _SMT_OS_ATTR_CALL,arg1,arg2,arg3,arg4,arg5,arg6
,arg7,arg8,arg9,arg10,arg11,arg12);
_SMT_OsCall12( osc, _SMT_OS_ATTR_RET,arg1,arg2,arg3,arg4,arg5,arg6
,arg7,arg8,arg9,arg10,arg11,arg12);
```

■引数

No.	名前	型	意味
1	osc	int	システムコール番号
2	attr	_SMT_OS_CALL	属性 定義値 _SMT_OS_ATTR_CALL : システムコール開始 _SMT_OS_ATTR_RET : システムコール終了
3	arg1 ～ arg12	unsigned long	引数 1～引数 12

■戻り値

型 : int

実行結果を返します。

_SMT_OK（正常終了）又は_SMT_NG（異常終了）を返します。

2.5.14 _SMT_OsCall13 関数

■機能

システムコール番号、呼び出し属性、引数 13 個を出力します。

■呼び出し手順

```
#include "SMTAPI.h"

int osc;
unsigned long arg1,arg2,arg3,arg4,arg5,arg6,arg7,arg8,arg9,arg10
,arg11,arg12,arg13;

_SMT_OsCall12( osc, _SMT_OS_ATTR_CALL,arg1,arg2,arg3,arg4,arg5,arg6
,arg7,arg8,arg9,arg10,arg11,arg12,arg13);
_SMT_OsCall12( osc, _SMT_OS_ATTR_RET,arg1,arg2,arg3,arg4,arg5,arg6
,arg7,arg8,arg9,arg10,arg11,arg12,arg13);
```

■引数

No.	名前	型	意味
1	osc	int	システムコール番号
2	attr	_SMT_OS_CALL	属性 定義値 _SMT_OS_ATTR_CALL : システムコール開始 _SMT_OS_ATTR_RET : システムコール終了
3	arg1 ～ arg13	unsigned long	引数 1～引数 13

■戻り値

型 : int

実行結果を返します。

_SMT_OK（正常終了）又は_SMT_NG（異常終了）を返します。

2.5.15 _SMT_OsCall14 関数

■機能

システムコール番号、呼び出し属性、引数 14 個を出力します。

■呼び出し手順

```
#include "SMTAPI.h"

int osc;
unsigned long arg1,arg2,arg3,arg4,arg5,arg6,arg7,arg8,arg9,arg10
,arg11,arg12,arg13,arg14;

_SMT_OsCall12( osc, _SMT_OS_ATTR_CALL,arg1,arg2,arg3,arg4,arg5,arg6
,arg7,arg8,arg9,arg10,arg11,arg12,arg13,arg14);
_SMT_OsCall12( osc, _SMT_OS_ATTR_RET,arg1,arg2,arg3,arg4,arg5,arg6
,arg7,arg8,arg9,arg10,arg11,arg12,arg13,arg14);
```

■引数

No.	名前	型	意味
1	osc	int	システムコール番号
2	attr	_SMT_OS_CALL	属性 定義値 _SMT_OS_ATTR_CALL : システムコール開始 _SMT_OS_ATTR_RET : システムコール終了
3	arg1 ～ arg14	unsigned long	引数 1～引数 14

■戻り値

型 : int

実行結果を返します。

_SMT_OK（正常終了）又は_SMT_NG（異常終了）を返します。

2.5.16 _SMT_OsCall15 関数

■機能

システムコール番号、呼び出し属性、引数 15 個を出力します。

■呼び出し手順

```
#include "SMTAPI.h"

int osc;
unsigned long arg1,arg2,arg3,arg4,arg5,arg6,arg7,arg8,arg9,arg10
,arg11,arg12,arg13,arg14,arg15;

_SMT_OsCall12( osc, _SMT_OS_ATTR_CALL,arg1,arg2,arg3,arg4,arg5,arg6
,arg7,arg8,arg9,arg10,arg11,arg12,arg13,arg14,arg15);
_SMT_OsCall12( osc, _SMT_OS_ATTR_RET,arg1,arg2,arg3,arg4,arg5,arg6
,arg7,arg8,arg9,arg10,arg11,arg12,arg13,arg14,arg15);
```

■引数

No.	名前	型	意味
1	osc	int	システムコール番号
2	attr	_SMT_OS_CALL	属性 定義値 _SMT_OS_ATTR_CALL : システムコール開始 _SMT_OS_ATTR_RET : システムコール終了
3	arg1 ～ arg15	unsigned long	引数 1～引数 15

■戻り値

型 : int

実行結果を返します。

_SMT_OK（正常終了）又は_SMT_NG（異常終了）を返します。

2.5.17 _SMT_OsCall16 関数

■機能

システムコール番号、呼び出し属性、引数 16 個を出力します。

■呼び出し手順

```
#include "SMTAPI.h"

int osc;
unsigned long arg1,arg2,arg3,arg4,arg5,arg6,arg7,arg8,arg9,arg10
,arg11,arg12,arg13,arg14,arg15,arg15;

_SMT_OsCall12( osc, _SMT_OS_ATTR_CALL,arg1,arg2,arg3,arg4,arg5,arg6
,arg7,arg8,arg9,arg10,arg11,arg12,arg13,arg14,arg15,arg16);
_SMT_OsCall12( osc, _SMT_OS_ATTR_RET,arg1,arg2,arg3,arg4,arg5,arg6
,arg7,arg8,arg9,arg10,arg11,arg12,arg13,arg14,arg15,arg16);
```

■引数

No.	名前	型	意味
1	osc	int	システムコール番号
2	attr	_SMT_OS_CALL	属性 定義値 _SMT_OS_ATTR_CALL : システムコール開始 _SMT_OS_ATTR_RET : システムコール終了
3	arg1 ～ arg16	unsigned long	引数 1～引数 16

■戻り値

型 : int

実行結果を返します。

_SMT_OK（正常終了）又は_SMT_NG（異常終了）を返します。

3 API ライブラリ使用上の注意事項

3.1 デバッグプリント関数について

デバッグプリント関数に出力可能な文字列長は、必ず、書式変換後の長さで、終端文字を含め 124byte 以内としてください。
それ以上の文字列を指定した場合、意図しない動作をする可能性があります。

4 API ライブラリの便利な使用方法

4.1 デバッグ関数の無効化

デバッグ関数をソース中に記述する際に、すべて大文字で記述しておくと、それら大文字で記述されたデバッグ関数は、マクロ定義を使用して一度に無効化できます。
システムマクロトレース API ライブラリヘッダのインクルード定義行の前に､マクロ定義を行います。（マクロ定義変更後にリビルドが必要です）

```
_SMT_PORT_DIS
_SMT_PRINT_DIS
#include "SMTAPI.h"
```

以下のマクロ定義を行うことで、関数グループ毎に無効化できます。

マクロ定義	無効になる関数グループ
_SMT_ALL_DIS	すべてのデバッグ関数
_SMT_PRINT_DIS	デバッグプリント関数
_SMT_PORT_DIS	ポートアウト関数
_SMT_OS_DIS	コンテキストスイッチ関数及びシステムコール関数

※マクロ定義値に関わらず、マクロ定義の有無で判断します。

例えば、以下のようにソースコードに記述すると

```
_SMT_Puts("Message1");       // マクロ定義に依存せず常に有効にします
_SMT_PUTS("Message2");       // マクロ定義によって有効/無効の切り替えをします
```

マクロ定義無しの場合の出力結果

```
Message1
Message2
```

_SMT_ALL_DIS または_SMT_PRINT_DIS の定義がある場合の出力結果

```
Message1
```

となります。