



Release Note

for “Release 1.5 for NO_OS”

2006.12.28
OTSL

Release 1.5 No OS 版(slave のみ)について記載する。

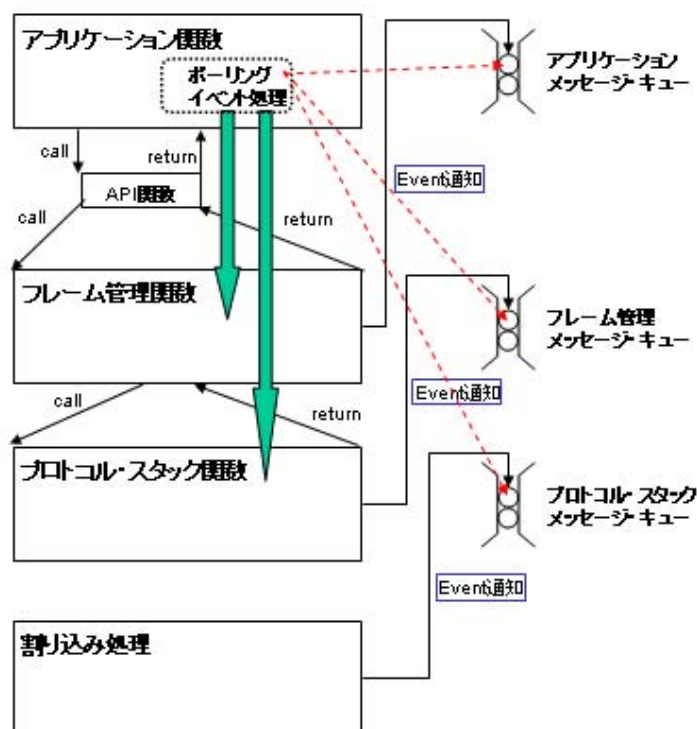
1. RTOS 無し版の同期

メッセージ送信／受信は、OSAL を OS 無し用に記載追加。

OS のタスク・スイッチング機能は、アプリケーション・プログラムで、メッセージのポーリングを行い、フレーム管理関数、プロトコル・スタック関数をコールするものとする。以下の説明図を設計資料 Version 2.2 にも追加している。

OS無しの場合の処理の流れと同期

Page 54



Copyrights © 2006. All rights are reserved by OTSL Inc.



フレーム管理, プロトコル・スタックはタスクであったものを、関数として定義する。

以下のように、フレーム管理タスク(TLinFrame.c), プロトコルスタック・タスク(tlin_stack_main.c)を関数として定義した。

```
/******  
Function Name : tlin_frmcom_tsk(void);  
Function      : TLIN Protocol Stack Task main routine.  
Calling       : void tlin_frmcon_tsk( void );  
Input        : None  
Output       : None  
*****/  
  
#ifdef __NO_OS__  
/******  
OS 無しエントリー  
*****/  
  
void tlin_frmcon_tsk( TLinU32 a_message ) {  
    TLinIfcHnd a_ifc;          /* Interface Handle */  
    TLinU16 a_event;           /* Event ID */  
    TLinU16 a_data;            /* API message Data Field */  
    int a_index;               /* スタック内部で使用する Channel Index */  
    TLinU08 a_exec_mod;        /* 実行モード: Master or Slave ? */  
    TLinU16 a_ercd;            /* エラーコード */  
    T_TLIN_FRMSTM_CNT* p_stm; /* STM Control 構造体へのポインタ */  
    TLinU32 a_tskid;           /* プログラム定義 Task ID */  
  
    a_ercd = TLIN_OSAL_STATUS_OK;  
    a_data = 0;  
  
#else  
/******  
OS 有りエントリー  
*****/  
  
void tlin_frmcon_tsk( void )  
{  
    TLinU32 a_message;          /* 受信メッセージ格納 */  
    TLinIfcHnd a_ifc;          /* Interface Handle */  
    TLinU16 a_event;           /* Event ID */  
    TLinU16 a_data;            /* API message Data Field */  
    int a_index;               /* スタック内部で使用する Channel Index */  
    TLinU08 a_exec_mod;        /* 実行モード: Master or Slave ? */  
    TLinU16 a_ercd;            /* エラーコード */  
    T_TLIN_FRMSTM_CNT* p_stm; /* STM Control 構造体へのポインタ */  
    TLinU32 a_tskid;           /* プログラム定義 Task ID */  
  
    /* 使用変数等初期化 */  
    frmcon_ini_ram();  
  
    a_ercd = TLIN_OSAL_STATUS_OK;  
    a_data = 0;  
  
    /* TLIN Data Structure の初期化 */  
  
    /* Message 待ち MainLoop */  
    while (1) {  
        /*===== メッセージを受け取る =====*/  
        a_ercd = tlin_osal_recv_MailBox( TID_TLIN_FRM_MBXID,  
                                         (TLinU32 *)&a_message,  
                                         TLIN_OSAL_WAIT_TICK );  
  
#ifdef __DEBUG_LOG__  
        __putlog(0xE261);  
        __putlog((TLinU32)a_message);  
#endif  
  
        if ( a_ercd != TLIN_OSAL_STATUS_OK ) {  
            continue;  
        }  
    }  
#endif
```



TlinFrame.c の変更

```
#ifdef __NO_OS__
/*****
OS 無しエントリー
*****/
void tlin_stack_tsk( TLinU32 message )
{
    TLinU16      a_ercd;

#else
/*****
OS 有りエントリー
*****/

void tlin_stack_tsk( void )
{
    TLIN_MSG_PKT  message;
    TLinU16      a_ercd;

    /* 使用変数等初期化 */
    tlin_ini_ram();

    /* HAL initialize */
    tlin_hal_init();

    stack_log_init();

    /* Message 待ち MainLoop */
    while ( 1 ) {
        /*===== メッセージを受け取る =====*/
        a_ercd = tlin_osal_rcv_MailBox ( TID_TLIN_TSK_MBXID, (TLinU32
        *)&message, 65535 );
#ifdef __DEBUG_LOG__
        __putlog(0xA091);
        __putlog((TLinU32)message);
#endif

        /*===== エラーなら無視して、次のメッセージ待ち =====*/
        if ( a_ercd != TLIN_OSAL_STATUS_OK ) {
            continue;
        }

    }

#endif
```

tlin_stack_main.c の変更



また、メッセージ送受信については、OSAL に以下のようなメッセージ・キューを追加した。

```
/*
*****
No Operating System 用 Mail Box / Event Flag 定義
*****
*/

#ifdef __NO_OS__

#define MBX_APP_SIZE 10
#define MBX_FRM_SIZE 20
#define MBX_STK_SIZE 50

static TLinU32 mbx_app[MBX_APP_SIZE];
static TLinU32 mbx_frm[MBX_FRM_SIZE];
static TLinU32 mbx_stk[MBX_STK_SIZE];

static TLinU08 rp_mbx_app;
static TLinU08 rp_mbx_frm;
static TLinU08 rp_mbx_stk;
static TLinU08 wp_mbx_app;
static TLinU08 wp_mbx_frm;
static TLinU08 wp_mbx_stk;
static TLinU08 p_round_app;
static TLinU08 p_round_frm;
static TLinU08 p_round_stk;

static TLinU16 evtflg_frm;
static TLinU16 evtflg_stk;
static TLinU16 evtflg_hal;

#endif
```

これに伴い、tlin_osal_send_MailBox(), tlin_osal_recv_MailBox()が変更されている。(ソースコード参照)

2. コンパイル・スイッチ

OS 有り／無し版を共通のファイルにするために、コンパイル・スイッチ・__NO_OS__を加えた。このスイッチは、TLinCfg.h に含まれている。(以下のソースコード引用参照)

```
/* OS 使用可否のスイッチ */
#define __NO_OS__
/* このスイッチが定義されていると、No OS 用のインプリトなる。 */
```

TLinCfg.h 中のコンパイル・スイッチ記述

ソース・コードの変更は、上記2項に示したものが大きな変更であるが、これに伴い、各所に小さな変更が加えられている。

変更箇所を調べるには、“ __NO_OS__ ”を検索キーにして調べればよい。

3. イベント・ポーリング

OS のタスク・スイッチングが使えないため、イベントの検出は、アプリケーションで行なうことになる。(OSAL のメッセージ受信を用いる。)

以下、サンプル・ソースでのイベント・ポーリングの例を示す。



```
/*--- Signal 書き換え ---*/
#ifdef __NO_OS__

#define NUM_STK_LOOP      20
#define NUM_STK_MBX_LOOP  1
#define NUM_FRM_LOOP      10
#define NUM_FRM_MBX_LOOP  1

while(1) {

    tlin_sig_rw();
    /* プロトコル・スタック・イベント検出 */
    for(i=0; i<NUM_STK_LOOP; ++i) {
        a_ercd = tlin_osal_recv_MailBox( TID_TLIN_TSK_MBXID, &message, NUM_STK_MBX_LOOP);
        if ( a_ercd == TLIN_OSAL_STATUS_OK ) {
            tlin_stack_tsk( message ); /* イベント処理 */
            break;
        }
    }
    /* フレーム管理・イベント検出 */
    for(i=0; i<NUM_FRM_LOOP; ++i) {
        a_ercd = tlin_osal_recv_MailBox( TID_TLIN_FRM_MBXID, &message, NUM_FRM_MBX_LOOP );
        if ( a_ercd == TLIN_OSAL_STATUS_OK ) {
            tlin_frmcon_tsk( message ); /* イベント処理 */
            break;
        }
    }
    heart_beat();
}
```