RX ファミリ

R01AN1644JJ0100
Rev.1.00
2013.03.29

オープンソースFATファイルシステム M3S-TFAT-Tinyへの SPI モード MMC/SD メモリカード・ドライバ・ソフトウェアの組み込み例

要旨

本アプリケーションノートでは、オープンソース FAT ファイルシステム M3S-TFAT-Tiny（以下、TFAT ライブラリと呼ぶ）と SPI モード MMC/SD メモリカード・ドライバ・ソフトウェア（以下、MMC/SD メモリカード・ドライバと呼ぶ）を組み合わせて使用する場合の組み込み方法について説明します。

対象デバイス

対象 MCU

「RX ファミリ　オープンソース FAT ファイルシステム M3S-TFAT-Tiny」と「RX ファミリ　SPI モード・MMC/SD メモリカード・ドライバ・ソフトウェア Ver.2.11」が共に動作する MCU

本アプリケーションノートを他のマイコンへ適用する場合、そのマイコンの仕様にあわせて変更し、十分評価してください。

目次

1. 仕様 3

2. 動作確認条件 4

3. 関連アプリケーションノート／ユーザーズマニュアル 5

4. ソフトウェア説明 6
4.1 動作概要 6
4.2 必要メモリサイズ 7
4.3 ファイル構成 7
4.4 関数一覧 8
4.4.1 メモリドライバインタフェース関数 8
4.5 関数仕様 9
4.5.1 メモリドライバインタフェース関数の修正 9
4.6 その他修正箇所 17
4.6.1 r_main.c（TFAT 動作確認用ファイル） 17
4.6.2 mtl_com.h（共通関数ヘッダファイル） 18
4.6.3 mtl_com.h.common（共通ヘッダファイル） 18

5. 使用上の注意事項 19
5.1 ヘッダファイルのインクルード 19
5.2 MMC/SD メモリカード・ドライバを組み込み時に必要なファイル 19
5.3 共通関数のヘッダファイル 19
5.4 使用制限事項 19
5.5 MMC/SD メモリカード・ドライバの初期化手順 19
5.6 動作確認 19

1. 仕様

記憶メディアのファイル操作を可能にするための TFAT ライブラリに含まれるメモリドライバインタフェース関数のサンプルコードです。

以下に、機能概略を示します。

- MMC/SD メモリカード・ドライバを使用する場合のメモリドライバインタフェース関数を提供し、ファイル操作を可能にします。
- TFAT ライブラリの R_tfat_f_sync()（ディスクキャッシュ制御）は、未サポートです。

2. 動作確認条件

本アプリケーションノートのサンプルコードは、下記の条件で動作を確認しています。

表2.1　動作確認条件

項目	内容
使用マイコン	RX62N グループ（プログラム ROM 512KB / RAM 96KB）
動作周波数	ICLK：96MHz、PCLK：48MHz
動作電圧	3.3V
統合開発環境	**ルネサス エレクトロニクス製** High-performance embedded Workshop Version 4.09.01.007
C コンパイラ	**ルネサス エレクトロニクス製** RX ファミリ用 C/C++コンパイラケッケージ （ツールチェーン 1.2.1.0）
	コンパイルオプション **総合開発環境のデフォルト設定（※１）を使用しています。** **※１：最適化レベル**”2”、**最適化方法**”**サイズ優先**”
サンプルコードのバージョン	1.01
使用ボード	Renesas Starter Kit for RX62N
使用デバイス	Panasonic SDHC class10 32GB

3. 関連アプリケーションノート／ユーザーズマニュアル

本アプリケーションノートに関連するアプリケーションノート／ユーザーズマニュアルを以下に示します。併せて参照してください。

- FAT ファイルシステム（M3S-TFAT-Tiny）（R20UW0078JJ）
- RX ファミリ　FAT ファイルシステムソフトウェア M3S-TFAT-Tiny：導入ガイド（R20AN0038JJ）
- RX ファミリ（RX600 シリーズ、RX200 シリーズ）　SPI モード・MMC/SD メモリカード・ドライバ・ソフトウェア Ver.2.11（R01UW0076JJ）
- RX610 グループ　SCI を使ったクロック同期式シングルマスタ制御ソフトウェア（R01AN0534JJ）
- RX62N グループ　RSPI を使ったクロック同期式シングルマスタ制御ソフトウェア（R01AN0323JJ）
- RX62N グループ　SCI を使ったクロック同期式シングルマスタ制御ソフトウェア（R01AN1088JJ）
- RX63N グループ　RSPI を使ったクロック同期式シングルマスタ制御ソフトウェア（R01AN0812JJ）
- RX63N グループ　SCI を使ったクロック同期式シングルマスタ制御ソフトウェア（R01AN1089JJ）
- RX210,RX21A,RX220 グループ　RSPI を使ったクロック同期式シングルマスタ制御ソフトウェア（R01AN1196JJ）
- RX210,RX21A,RX220 グループ　SCI を使ったクロック同期式シングルマスタ制御ソフトウェア（R01AN1229JJ）

4. ソフトウェア説明

4.1 動作概要

TFAT ライブラリに含まれるメモリドライバインタフェース関数を MMC/SD メモリカード・ドライバ用に修正し、記憶メディアのファイル操作を可能にします。

図 4.1にTFAT ライブラリと MMC/SD メモリカード・ドライバの構成を示します。

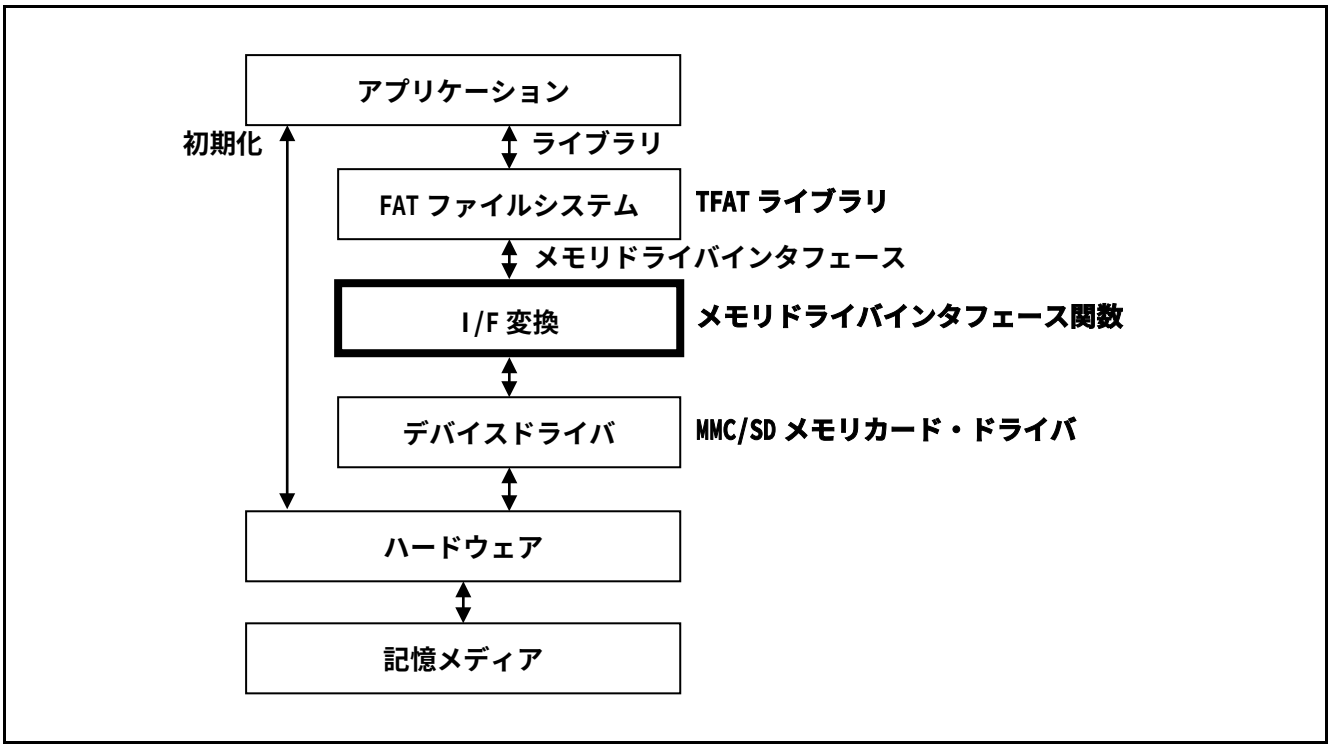

図4.1　TFAT **ライブラリと** MMC/SD **メモリカード・ドライバの構成**

4.2　必要メモリサイズ

表 4.1に必要メモリサイズを示します。

表4.1　**必要メモリサイズ**

使用メモリ	サイズ	備考
ROM	1,467 **バイト**	r_tfat_drv_if.c
RAM	26 **バイト**	r_tfat_drv_if.c
最大使用ユーザスタック	268 **バイト**	
最大使用割り込みスタック	-	**割り込み未使用**

【注１】必要メモリサイズは C **コンパイラのバージョンやコンパイルオプションにより異なります。**
【注２】スタックサイズは、MMC/SD **メモリカード・ドライバとクロック同期式シングルマスタドライバの スタック** RAM **サイズを含みます。**
【注３】TFAT **動作確認用ファイル（**r_main.c**）は含みません。**

4.3　ファイル構成

表 4.2にサンプルコードで使用するファイルを示します。なお、統合開発環境で自動生成されるファイルは除きます。

表4.2　**サンプルコードで使用するファイル**

¥an_r01an1644jj0100_rx_file_system		<DIR>	**サンプルコードのフォルダ**
	r01an1644jj0100_rx.pdf		**アプリケーションノート**
	¥source	<DIR>	**プログラム格納用フォルダ**
		¥tfat_sample <DIR>	**メモリドライバインタフェース用フォルダ**
		r_main.c	TFAT **動作確認用ファイル**
		r_tfat_drv_if.c	**メモリドライバインタフェース関数ファイル**
		¥com <DIR>	**共通関数格納用フォルダ**
		mtl_com.h.common	**共通のヘッダファイル【注１】**
		mtl_com.h.RX	**共通関数のヘッダファイル【注１】**

【注１】MCU **個別のクロック同期式シングルマスタ制御ソフトウェアに含まれる** com **フォルダのファイルです。**

4.4 関数一覧

4.4.1 メモリドライバインタフェース関数

表 4.3にメモリドライバインタフェース関数を示します。

表4.3 メモリドライバインタフェース関数

関数名	概要
R_tfat_disk_initialize()	ディスク・ドライブの初期化
R_tfat_disk_read()	ディスクからの読み込み
R_tfat_disk_write()	ディスクへの書き込み
R_tfat_disk_ioctl()	その他のドライブ制御
R_tfat_disk_status()	ディスク・ドライブの状態取得
R_tfat_get_fattime()	日付・時刻の取得
R_card_insertion_chk()	カードの挿入チェック処理
R_init_card_detect_chat()	チャタリングカウンタ初期化

R_tfat_outstream 関数（TFAT ライブラリのユーザーズマニュアルを参照）は、R_tfat_f_forward 関数を使用する場合に必要です。必要に応じて作成してください。

4.5　関数仕様

TFAT ライブラリのユーザーズマニュアルを参照してください。

4.5.1　メモリドライバインタフェース関数の修正

(1)　インクルードファイルの修正

MMC/SD メモリカード・ドライバのヘッダファイルと共通関数のヘッダファイルを設定してください。なお、共通関数のヘッダファイルについては、MCU 個別のクロック同期式シングルマスタ制御ソフトウェアから入手してください。

■r_tfat_drv_if.c　36 行目付近

```
/************************************************************************
Includes <System Includes> , "Project Includes"
************************************************************************/
#include "r_tfat_lib.h"
#include "mtl_com.h"
#include "R_SPI_MMC.h"
```

(2)　#define 定義の追加

MMC/SD メモリカード検出用の#define 定義を行ってください。

■r_tfat_drv_if.c　50 行目付近

```
/************************************************************************
Macro definitions
************************************************************************/
#define MMC_INS_CHAT    20 /* counter value of chattering detection */
#define DISABLE_IRQ()
#define ENABLE_IRQ()
#define MMC_CARD_CHK_MSK    0x00FF
```

(3)　変数定義の修正

配列のインデックスを変更してください。

■r_tfat_drv_if.c　58 行目付近

```
/************************************************************************
Exported global variables and functions (to be accessed by other files)
************************************************************************/
/* Real Time Clock - Time set to 23 FEB 2009 15:02:20 */
uint8_t g_rtcYear = 109, g_rtcMon = 2, g_rtcMday = 23;
uint8_t g_rtcHour = 15, g_rtcMin = 2, g_rtcSec = 20;

uint16_t gDetChatCnt[MMC_DEV_NUM]; /* Counter of chattering detection */
uint16_t gDetSts_Old[MMC_DEV_NUM]; /* Previous status of detection   */
MMC_INFO gMmcInfo[MMC_DEV_NUM];
```

(4) R_tfat_disk_initialize()の修正

本関数は、MMC/SD メモリカード検出処理（内部で R_SPI_MMC_Chk_Detect()関数をコール）とデバイス初期化設定（R_SPI_MMC_Init_Device()関数をコール）を実行するものです。

したがって、TFAT ライブラリから MMC/SD メモリカード・ドライバ API がコールされる前に、MMC/SD メモリカード・ドライバの初期化処理(R_SPI_MMC_Init_Driver())を事前に行ってください。

MMC/SD メモリカード・ドライバの初期化方法は、MMC/SD メモリカード・ドライバのユーザーズマニュアルを参照してください。

■r_tfat_drv_if.c　82 行目付近

```
DSTATUS R_tfat_disk_initialize(void)
{
  /*Please put the code for disk_initalize driver interface function over here */
  /*Please refer the application note for details                                */
  /*Please put the code for memory driver initialization, if any, over here    */
  int mmc_ret;
  DSTATUS ret_value  = TFAT_STA_NODISK | TFAT_STA_NOINIT;

  mmc_ret = R_card_insertion_chk(MMC_DEV0);
  if (mmc_ret < MMC_OK)
  {
   R_init_card_detect_chat(MMC_DEV0);
   return ret_value;
  }
  if (mmc_ret == MMC_FALSE)
  {
   return ret_value;
  }

  ret_value &= ~TFAT_STA_NODISK;
  mmc_ret = R_SPI_MMC_Init_Device(MMC_DEV0);
  if (mmc_ret < MMC_OK)
  {
   R_init_card_detect_chat(MMC_DEV0);
   return ret_value;
  }

  mmc_ret = R_SPI_MMC_Get_MmcInfo(MMC_DEV0, &gMmcInfo[MMC_DEV0]);
  if (mmc_ret < MMC_OK)
  {
   R_init_card_detect_chat(MMC_DEV0);
   return ret_value;
  }

  ret_value &= ~TFAT_STA_NOINIT;
  return  ret_value;
}
```

(5) R_tfat_disk_read()の修正

MMC/SD メモリカード・ドライバの読み出し関数をコールします。

■r_tfat_drv_if.c　131 行目付近

```
DRESULT R_tfat_disk_read(
    uint8_t Drive,        /* Physical drive number          */
    uint8_t* Buffer,      /* Pointer to the read data buffer   */
    uint32_t SectorNumber, /* Start sector number           */
    uint8_t SectorCount    /* Number of sectors to read     */
)
{
  /*   Please put the code for R_tfat_disk_read driver interface function over he
re */
  /*   Please refer the application note for details                          */
  DRESULT func_ret;
  int16_t mmc_ret;
  mmc_ret = R_SPI_MMC_Read_Data(MMC_DEV0, SectorNumber, SectorCount, Buffer, MMC
_MODE_NORMAL);
  switch (mmc_ret)
  {
   case MMC_OK:
      func_ret = TFAT_RES_OK;
      break;
   case MMC_ERR_PARAM:
      func_ret = TFAT_RES_PARERR;
      break;
   case MMC_ERR_OTHER:
   default:
      func_ret = TFAT_RES_ERROR;
      break;
  }
  return func_ret;
}
```

(6) R_tfat_disk_write()の修正

MMC/SD メモリカード・ドライバの書き込み関数をコールします。

以下の修正により MMC メモリカードに加え、SD メモリカードと SDHC メモリカードの制御が可能になります。ただし、Read Only や One Time Program のカードチェックは実施しません。

■r_tfat_drv_if.c　170 行目付近

```
DRESULT R_tfat_disk_write(
    uint8_t Drive,          /* Physical drive number          */
    const uint8_t* Buffer,  /* Pointer to the write data       */
    uint32_t SectorNumber,  /* Sector number to write          */
    uint8_t SectorCount     /* Number of sectors to write      */
)
{
  /*   Please put the code for R_tfat_disk_write driver interface function over h
ere    */
  /*   Please refer the application note for details                          */
  DRESULT func_ret;
  int16_t mmc_ret;

  /* Assumes using Regular RD/WR card. */
  /* Doesn't judge the card attributes such as Read Only or One time program. */
  if (((gMmcInfo[MMC_DEV0].Card & MMC_CARD_CHK_MSK) != MMC_CARD_MMC) &&
      ((gMmcInfo[MMC_DEV0].Card & MMC_CARD_CHK_MSK) != MMC_CARD_SD) &&
     ((gMmcInfo[MMC_DEV0].Card & MMC_CARD_CHK_MSK) != MMC_CARD_SDHC)   )
  {
    return TFAT_RES_NOTRDY;
  }
  if (gMmcInfo[MMC_DEV0].WProtect != MMC_FALSE)
  {
    return TFAT_RES_WRPRT;
  }

  mmc_ret = R_SPI_MMC_Write_Data(MMC_DEV0, SectorNumber, SectorCount, (uint8_t
*)Buffer, MMC_MODE_NORMAL);
  switch (mmc_ret)
  {
    case MMC_OK:
       func_ret = TFAT_RES_OK;
       break;
    case MMC_ERR_PARAM:
       func_ret = TFAT_RES_PARERR;
       break;
    case MMC_ERR_WP:
       func_ret = TFAT_RES_WRPRT;
       break;
    case MMC_ERR_OTHER:
    default:
       func_ret = TFAT_RES_ERROR;
       break;
  }
  return func_ret;
}
```

(7) R_tfat_disk_ioctl()の修正

ディスクキャッシュ機能を持たせる場合に、適切な処理を記述してください。

詳細は、TFAT ライブラリのユーザーズマニュアルを参照してください。

以下は、何もせずに正常終了させる場合の処理例です。したがって、R_tfat_f_sync()を使用しないでください。

■r_tfat_drv_if.c　225 行目付近

```
DRESULT R_tfat_disk_ioctl(
    uint8_t Drive,         /* Drive number         */
    uint8_t Command,       /* Control command code   */
    void* Buffer          /* Data transfer buffer    */
)
{
  /*   Please put the code for R_tfat_disk_ioctl driver interface function over h
ere   */
  /*   Please refer the application note for details                          */
  return TFAT_RES_OK;
}
```

(8)　R_tfat_disk_status ()の修正

ディスク・ドライブの状態を取得させる場合に、適切な処理を記述してください。

詳細は、TFAT ライブラリのユーザーズマニュアルを参照してください。

以下の修正により MMC メモリカードに加え、SD メモリカードと SDHC メモリカードの制御が可能になります。ただし、Read Only や One time program のカードチェックは実施しません。

■r_tfat_drv_if.c　244 行目付近

```
DSTATUS R_tfat_disk_status(
    uint8_t Drive          /* Physical drive number   */
)
{
  /*   Please put the code for R_tfat_disk_status driver interface function over
here  */
  /*   Please refer the application note for details                          */
  uint8_t cdetect;

  /* Assumes using Regular RD/WR card. */
  /* Doesn't judge the card attributes such as Read Only or One time program. */
  if (gDetSts_Old[MMC_DEV0] == MMC_FALSE)
  {
    return TFAT_STA_NODISK | TFAT_STA_NOINIT;
  }
  if (MMC_OK != R_SPI_MMC_Chk_Detect(MMC_DEV0, &cdetect))
  {
    return TFAT_STA_NODISK | TFAT_STA_NOINIT;
  }
  if (cdetect == MMC_FALSE)
  {
    return TFAT_STA_NODISK | TFAT_STA_NOINIT;
  }
  if (((gMmcInfo[MMC_DEV0].Card & MMC_CARD_CHK_MSK) != MMC_CARD_MMC) &&
      ((gMmcInfo[MMC_DEV0].Card & MMC_CARD_CHK_MSK) != MMC_CARD_SD) &&
       ((gMmcInfo[MMC_DEV0].Card & MMC_CARD_CHK_MSK) != MMC_CARD_SDHC)   )
  {
    return TFAT_STA_NOINIT;
  }
  if (gMmcInfo[MMC_DEV0].WProtect != MMC_FALSE)
  {
    return TFAT_STA_PROTECT;
  }
  return 0;

}
```

(9) R_tfat_get_fattime ()の修正

MMC/SD メモリカード・ドライバが関連しない処理のため、説明を省略します。

(10) R_card_insertion_chk()の修正

MMC/SD メモリカード・ドライバの挿入チェック関数をコールします。

■drv_if_sub.c　323 行目付近

```
int16_t R_card_insertion_chk(uint8_t slot_num)
{
  uint8_t  DetSts;  /* Detection status */
  int16_t  api_ret;

  /* Check MMC insertion. */
  api_ret = R_SPI_MMC_Chk_Detect(slot_num, &DetSts); /* Get status of MMC insert
ion. */
  if (api_ret < MMC_OK)
  {
    return api_ret;
  }
  if (DetSts != gDetSts_Old[slot_num]) /* Status Changed! */
  {
    if (DetSts == MMC_TRUE) /* Removal -> Insertion */
    {
       gDetChatCnt[slot_num]--;       /* Chattering counter decrement */
       if (gDetChatCnt[slot_num] == 0) /* counter =0                */
       {
          /* Initialize MMC slot. */
          gDetChatCnt[slot_num] = MMC_INS_CHAT;
          gDetSts_Old[slot_num] = DetSts;
       }
    }
    else
    {
       /* Insertion -> Removal    */
       /* Do not care chattering. */
       gDetChatCnt[slot_num] = MMC_INS_CHAT;
       gDetSts_Old[slot_num] = DetSts;
    }
  }
  else
  {
    /* No change */
    gDetChatCnt[slot_num] = MMC_INS_CHAT;
  }
  return gDetSts_Old[slot_num];
}
```

(11) R_init_card_detect_chat()の修正

対象の MMC/SD メモリカードのチャタリングカウンタの初期化を行います。

■drv_if_sub.c　370 行目付近

```
void R_init_card_detect_chat(uint8_t drv) /* Initial process */
{
  uint8_t slot_num; /* Slot number */

  if (drv >= MMC_DEV_NUM)
  {
    for (slot_num = MMC_DEV0; slot_num < MMC_DEV_NUM; slot_num++)
    {
       gDetChatCnt[slot_num] = MMC_INS_CHAT;
                        /* Reset the counter of chattering detection.           */
       gDetSts_Old[slot_num] = MMC_FALSE;
                       /* Set the previous status of detection  to "removal".   */
    }
  }
  else
  {
    gDetChatCnt[drv] = MMC_INS_CHAT;
                       /* Reset the counter of chattering detection.*/
    gDetSts_Old[drv] = MMC_FALSE;
                       /* Set the previous status of detection  to "removal".   */
  }
}
```

4.6　その他修正箇所

4.6.1　r_main.c（TFAT 動作確認用ファイル）

(1)　**インクルードファイルの修正**

MMC/SD メモリカード・ドライバのヘッダファイルと共通関数のヘッダファイルを設定してください。なお、共通関数のヘッダファイルについては、MCU 個別のクロック同期式シングルマスタ制御ソフトウェアから入手してください。

■r_main.c　43 行目付近

```
/***********************************************************************
Include files
***********************************************************************/
#include "iodefine.h"
#include "r_stdint.h"
#include "r_tfat_lib.h"
#include "r_data_file.h"
#include "r_board.h"
#include "mtl_com.h"
#include "R_SPI_MMC.h"
```

(2)　main()**の修正**

MMC/SD メモリカード・ドライバの初期化関数をコールします。

■r_main.c　90 行目付近

```
  /* Initialize MMC driver. */
  R_SPI_MMC_Init_Driver();

  /* Initialize card detect chattering. */
  R_init_card_detect_chat(MMC_DEV_NUM);
```

4.6.2　mtl_com.h（共通関数ヘッダファイル）

共通で使用される共通関数のヘッダファイルです。

通常は、3項に示す各クロック同期式シングルマスタ制御ソフトウェアから入手してください。

本アプリケーションノートでは、RX 用の MMC/SD メモリカード・ドライバ・ソフトウェアを組み込むために修正したヘッダファイル mtl_com.h.RX を同梱しています。使用する場合は、mtl_com.h にリネームしてご使用ください。

■mtl_com.h　120 行目付近

```
#include "mtl_com.h.common"
#include "r_stdint.h"
```

4.6.3　mtl_com.h.common（共通ヘッダファイル）

共通で使用されるヘッダファイルです。

通常は、3項に示す各クロック同期式シングルマスタ制御ソフトウェアから入手してください。

本アプリケーションノートでは、RX 用の MMC/SD メモリカード・ドライバ・ソフトウェアを組み込むために修正したヘッダファイル mtl_com.h.common を同梱しています。

■mtl_com.h.common　101 行目付近

```
/*-----------------------------------------------------------------------*
   Types
*-----------------------------------------------------------------------*/
typedef unsigned char   uchar;              /* For Legacy S/W                */
typedef unsigned short  ushort;             /* For Legacy S/W                */
typedef unsigned int    uint;               /* For Legacy S/W                */
typedef unsigned long   ulong;              /* For Legacy S/W                */

/* typedef unsigned char   uint8_t;  */     /* For RPDL                      */
/* typedef unsigned short  uint16_t; */     /* For RPDL                      */
/* typedef unsigned long   uint32_t; */     /* For RPDL                      */
/* typedef signed char     int8_t;   */     /* For RPDL                      */
/* typedef signed short    int16_t;  */     /* For RPDL                      */
/* typedef signed long     int32_t;  */     /* For RPDL                      */
```

5. 使用上の注意事項

5.1 ヘッダファイルのインクルード

アプリケーション・ソフトウェアに MMC/SD メモリカード・ドライバのヘッダファイル、共通関数のヘッダファイル、TFAT ライブラリのヘッダファイルをインクルードしてください。

```
#include "mtl_com"
#include "R_SPI_MMC.h"
#include "r_stdint.h"
```

5.2 MMC/SD メモリカード・ドライバを組み込み時に必要なファイル

MMC/SD メモリカード・ドライバは、MMC/SD メモリカードを制御するためのソフトウェアです。MCU と通信を行うためには、別途、MCU 個別のクロック同期式シングルマスタ制御ソフトウェアが必要です。

このソフトウェアはサンプルプログラムとして用意していますので、別途、ルネサスエレクトロニクスのホームページから入手してください。

5.3 共通関数のヘッダファイル

共通関数のヘッダファイルは、MCU 個別のクロック同期式シングルマスタ制御ソフトウェアに同梱されています。

5.4 使用制限事項

ディスクキャッシュ機能を持たせる場合のメモリドライバインタフェース処理としての R_tfat_disk_ioctl () 記述ではありません。本資料の記述内容の場合、TFAT ライブラリの使用上、R_tfat_f_sync()をコールしないでください。

5.5 MMC/SD メモリカード・ドライバの初期化手順

以下の順で MMC/SD メモリカード・ドライバの初期化とデバイスの初期化を実行してください。

①MMC/SD メモリカード・ドライバのドライバ初期化関数(R_SPI_MMC_Init_Driver())の実行
②R_tfat_disk_initialize()関数の実行
　MMC/SD メモリカードを検出し、デバイス初期化関数(R_SPI_MMC_Init_Driver())を実行します。

5.6 動作確認

TFAT ライブラリに含まれる r_main.c を使用し、動作確認済みです。

ホームページとサポート窓口

ルネサス エレクトロニクスホームページ
http://japan.renesas.com

お問合せ先
http://japan.renesas.com/contact/

改訂記録	RX ファミリ　アプリケーションノート　オープンソース FAT ファイルシステム M3S-TFAT-Tiny への SPI モード MMC/SD メモリカード・ドライバ・ソフトウェアの組み込み例

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2013.03.29	—	初版発行

製品ご使用上の注意事項

ここでは、マイコン製品全体に適用する「使用上の注意事項」について説明します。個別の使用上の注意事項については、本ドキュメントおよびテクニカルアップデートを参照してください。

1. 未使用端子の処理
   【注意】未使用端子は、本文の「未使用端子の処理」に従って処理してください。
   CMOS 製品の入力端子のインピーダンスは、一般に、ハイインピーダンスとなっています。未使用端子を開放状態で動作させると、誘導現象により、LSI 周辺のノイズが印加され、LSI 内部で貫通電流が流れたり、入力信号と認識されて誤動作を起こす恐れがあります。未使用端子は、本文「未使用端子の処理」で説明する指示に従い処理してください。
2. 電源投入時の処置
   【注意】電源投入時は，製品の状態は不定です。
   電源投入時には、LSI の内部回路の状態は不確定であり、レジスタの設定や各端子の状態は不定です。
   外部リセット端子でリセットする製品の場合、電源投入からリセットが有効になるまでの期間、端子の状態は保証できません。
   同様に、内蔵パワーオンリセット機能を使用してリセットする製品の場合、電源投入からリセットのかかる一定電圧に達するまでの期間、端子の状態は保証できません。
3. リザーブアドレス（予約領域）のアクセス禁止
   【注意】リザーブアドレス（予約領域）のアクセスを禁止します。
   アドレス領域には、将来の機能拡張用に割り付けられているリザーブアドレス（予約領域）があります。これらのアドレスをアクセスしたときの動作については、保証できませんので、アクセスしないようにしてください。
4. クロックについて
   【注意】リセット時は、クロックが安定した後、リセットを解除してください。
   プログラム実行中のクロック切り替え時は、切り替え先クロックが安定した後に切り替えてください。
   リセット時、外部発振子（または外部発振回路）を用いたクロックで動作を開始するシステムでは、クロックが十分安定した後、リセットを解除してください。また、プログラムの途中で外部発振子（または外部発振回路）を用いたクロックに切り替える場合は、切り替え先のクロックが十分安定してから切り替えてください。
5. 製品間の相違について
   【注意】型名の異なる製品に変更する場合は、製品型名ごとにシステム評価試験を実施してください。
   同じグループのマイコンでも型名が違うと、内部 ROM、レイアウトパターンの相違などにより、電気的特性の範囲で、特性値、動作マージン、ノイズ耐量、ノイズ輻射量などが異なる場合があります。型名が違う製品に変更する場合は、個々の製品ごとにシステム評価試験を実施してください。

ご注意書き

1. 本資料に記載された回路、ソフトウェアおよびこれらに関連する情報は、半導体製品の動作例、応用例を説明するものです。お客様の機器・システムの設計において、回路、ソフトウェアおよびこれらに関連する情報を使用する場合には、お客様の責任において行ってください。これらの使用に起因して、お客様または第三者に生じた損害に関し、当社は、一切その責任を負いません。
2. 本資料に記載されている情報は、正確を期すため慎重に作成したものですが、誤りがないことを保証するものではありません。万一、本資料に記載されている情報の誤りに起因する損害がお客様に生じた場合においても、当社は、一切その責任を負いません。
3. 本資料に記載された製品データ、図、表、プログラム、アルゴリズム、応用回路例等の情報の使用に起因して発生した第三者の特許権、著作権その他の知的財産権に対する侵害に関し、当社は、何らの責任を負うものではありません。当社は、本資料に基づき当社または第三者の特許権、著作権その他の知的財産権を何ら許諾するものではありません。
4. 当社製品を改造、改変、複製等しないでください。かかる改造、改変、複製等により生じた損害に関し、当社は、一切その責任を負いません。
5. 当社は、当社製品の品質水準を「標準水準」および「高品質水準」に分類しており、
   各品質水準は、以下に示す用途に製品が使用されることを意図しております。
   標準水準： コンピュータ、OA機器、通信機器、計測機器、AV機器、
   家電、工作機械、パーソナル機器、産業用ロボット等
   高品質水準：輸送機器（自動車、電車、船舶等）、交通用信号機器、
   防災・防犯装置、各種安全装置等
   当社製品は、直接生命・身体に危害を及ぼす可能性のある機器・システム（生命維持装置、人体に埋め込み使用するもの等）、もしくは多大な物的損害を発生させるおそれのある機器・システム（原子力制御システム、軍事機器等）に使用されることを意図しておらず、使用することはできません。たとえ、意図しない用途に当社製品を使用したことによりお客様または第三者に損害が生じても、当社は一切その責任を負いません。なお、ご不明点がある場合は、当社営業にお問い合わせください。
6. 当社製品をご使用の際は、当社が指定する最大定格、動作電源電圧範囲、放熱特性、実装条件その他の保証範囲内でご使用ください。当社保証範囲を超えて当社製品をご使用された場合の故障および事故につきましては、当社は、一切その責任を負いません。
7. 当社は、当社製品の品質および信頼性の向上に努めていますが、半導体製品はある確率で故障が発生したり、使用条件によっては誤動作したりする場合があります。また、当社製品は耐放射線設計については行っておりません。当社製品の故障または誤動作が生じた場合も、人身事故、火災事故、社会的損害等を生じさせないよう、お客様の責任において、冗長設計、延焼対策設計、誤動作防止設計等の安全設計およびエージング処理等、お客様の機器・システムとしての出荷保証を行ってください。特に、マイコンソフトウェアは、単独での検証は困難なため、お客様の機器・システムとしての安全検証をお客様の責任で行ってください。
8. 当社製品の環境適合性等の詳細につきましては、製品個別に必ず当社営業窓口までお問合せください。ご使用に際しては、特定の物質の含有・使用を規制するRoHS指令等、適用される環境関連法令を十分調査のうえ、かかる法令に適合するようご使用ください。お客様がかかる法令を遵守しないことにより生じた損害に関して、当社は、一切その責任を負いません。
9. 本資料に記載されている当社製品および技術を国内外の法令および規則により製造・使用・販売を禁止されている機器・システムに使用することはできません。また、当社製品および技術を大量破壊兵器の開発等の目的、軍事利用の目的その他軍事用途に使用しないでください。当社製品または技術を輸出する場合は、「外国為替及び外国貿易法」その他輸出関連法令を遵守し、かかる法令の定めるところにより必要な手続を行ってください。
10. お客様の転売等により、本ご注意書き記載の諸条件に抵触して当社製品が使用され、その使用から損害が生じた場合、当社は何らの責任も負わず、お客様にてご負担して頂きますのでご了承ください。
11. 本資料の全部または一部を当社の文書による事前の承諾を得ることなく転載または複製することを禁じます。

注1. 本資料において使用されている「当社」とは、ルネサスエレクトロニクス株式会社およびルネサスエレクトロニクス株式会社がその総株主の議決権の過半数を直接または間接に保有する会社をいいます。
注2. 本資料において使用されている「当社製品」とは、注１において定義された当社の開発、製造製品をいいます。

ルネサス エレクトロニクス株式会社

■営業お問合せ窓口 http://www.renesas.com

※営業お問合せ窓口の住所・電話番号は変更になることがあります。最新情報につきましては、弊社ホームページをご覧ください。

ルネサス エレクトロニクス販売株式会社 〒100-0004 千代田区大手町 2-6-2（日本ビル） (03)5201-5307

■技術的なお問合せおよび資料のご請求は下記へどうぞ。
総合お問合せ窓口：http://japan.renesas.com/contact/