

---

## RL78 ファミリ

R20AN0158JJ0103

### SPI モードマルチメディアカードドライバ: 導入ガイド

Rev.1.03

2015.10.01

---

#### 要旨

本書は、RL78 ファミリ用マルチメディアカードドライバのソフトウェア構成、使用方法について記述します。

また応用例として、マルツエレクトロニクス株式会社より販売しているRL78/G14マイコン・トレーニング・キット [MTK-RL78G14](#) に本ドライバを組み込んだ音声再生・録音デモソフトウェアを別途用意しています。詳細は下記URLをご参照ください。

[http://japan.renesas.com/products/tools/middleware/tiny\\_soft/adpcm/m3s\\_s2\\_tiny/app\\_notes.jsp](http://japan.renesas.com/products/tools/middleware/tiny_soft/adpcm/m3s_s2_tiny/app_notes.jsp)

(ドキュメント No. : R20AN0194)

#### 動作確認デバイス

RL78/G13、RL78/G14

## 目次

1. 概要.....	4
1.1 目的.....	4
1.2 機能概要.....	4
1.3 ファイルの構成.....	5
2. プログラム型定義.....	7
3. デバイスドライバ.....	8
3.1 デバイスドライバ関数概要.....	8
3.2 関数詳細.....	9
3.2.1 MMC ドライバ初期化処理(R_mmc_Init_Driver).....	9
3.2.2 MMC スロット初期化処理(R_mmc_Init_Slot).....	9
3.2.3 MMC スロット停止処理(R_mmc_Detach).....	9
3.2.4 MMC 挿入チェック処理(R_mmc_Chk_Detect).....	10
3.2.5 データ読み出し処理(R_mmc_Read_Data).....	10
3.2.6 データ書き込み処理(R_mmc_Write_Data).....	11
3.2.7 MMC 情報取得処理(R_mmc_Get_MmcInfo).....	12
3.3 データ構造体.....	13
3.4 マクロ定義.....	13
4. 設定例.....	14
4.1 共通関数 r_mtl_XXX の可変データ設定例.....	14
4.1.1 r_mtl_com.h.....	14
4.1.2 r_mtl_tim.h.....	15
4.2 MMC ドライバ可変データ設定例.....	16
4.2.1 r_mmc.h(ドライバ共通定義).....	16
4.2.2 r_mmc_user_config.h (MCU 個別定義).....	17
4.3 RL78/G13 使用時のポーティング方法.....	22
4.3.1 r_mmc_user_config.h.....	22
4.3.2 r_mmc_sfr.h.....	24
5. MCU との接続方法と使用する MCU リソース.....	25
5.1 使用する MCU リソース.....	25
5.2 MCU との接続方法.....	26
6. アプリケーション作成時の注意事項.....	27
6.1 使用上の注意事項.....	27
6.2 開発環境.....	27
6.2.1 CS+ for CA, CX.....	27
6.2.2 CS+ for CC.....	27
6.2.3 IAR Embedded Workbench.....	27
6.3 組み込み時の注意事項.....	28
6.3.1 インクルードするファイル.....	28
6.3.2 コンフィグ設定 (r_mmc_user_config.h) での注意事項.....	28
6.3.3 プログラム上の制限事項.....	28

---

6.4	ROM/RAM/スタックサイズ.....	28
6.4.1	CS+ for CA, CX.....	28
6.4.2	CS+ for CC.....	28
6.4.3	IAR Embedded Workbench .....	28
6.5	カードの挿抜検出に関する注意事項.....	29
6.6	ポートの割り当てとカードの挿抜に関するポートの Hi-z 化処理の注意事項.....	29
7.	MMC ドライバ改訂履歴 .....	30

## 1. 概要

### 1.1 目的

RL78 ファミリ MCU とマルチメディアカード(以下、MMC と略す。)を SPI モードで通信させるためのインタフェースを提供することを目的としています。

本アプリケーションノートでは、アプリケーションを作成するための情報を提供します。

### 1.2 機能概要

本デバイスドライバ(以下、MMC ドライバと略す。)は、MMC との通信を RL78 ファミリ MCU で実現するためのソフトウェアです。

本ソフトウェアは、RL78 ファミリ MCU に内蔵されている通信機能のうち、シリアル・アレイ・ユニットの 3 線シリアル(以下、CSI と略す。)を使い、SPI モードでの MMC へのアクセスを実現しています。

#### MMC ドライバの仕様

- 参照 MMCA 規格 Ver. 3.2
- MMC の SPI モード専用です。
- 1 セクタ=512Byte とするブロック型デバイスドライバです。  
READ\_MULTIPLE\_BLOCK コマンドと WRITE\_MULTIPLE\_BLOCK コマンドを使用しています。  
上記 2 つの MULTIPLE\_BLOCK コマンド非対応カードの場合は、READ\_SINGLE\_BLOCK コマンドと WRITE\_SINGLE\_BLOCK コマンドで対応します。
- CS 制御による複数デバイスをサポートしています。
- OS 非依存です。
- 本マニュアルの対象 MMC ドライバ : RL78 ファミリ SPI モードマルチメディアカードドライバ V.1.03 Release 00

## 1.3 ファイルの構成

ディレクトリおよびファイルの構成を示します。

ディレクトリ構成、 ディレクトリ名/ファイル名	備考
¥doc <DIR>	ドキュメントディレクトリ
r20an0158jj0103_rl78_mmc.pdf	アプリケーションノート(本書)
¥mmc_driver <DIR>	MMC ドライバソースプログラム
¥com <DIR>	共通関数のディレクトリ
r_mtl_com.c	共通関数(ログ記録)
r_mtl_com2.h	共通関数の各種定義
r_mtl_endi.c	共通関数(エンディアン関連)
r_mtl_mem.c	共通関数(標準ライブラリ関数)
r_mtl_str.c	共通関数(標準ライブラリ関数)
r_mtl_tim.c, r_mtl_tim.h	共通関数(ソフトウェアループタイマ)、各種定義
r_stdint.h	標準整数型定義ヘッダファイル
¥r178_32MHz <DIR>	CS+ for CA, CX 32MHz 動作用定義のディレクトリ
r_mtl_com.h	CS+ for CA, CX 32MHz 動作用定義ヘッダファイル
¥r178_24MHz <DIR>	CS+ for CA, CX 24MHz 動作用定義のディレクトリ
r_mtl_com.h	CS+ for CA, CX 24MHz 動作用定義ヘッダファイル
¥r178_20MHz <DIR>	CS+ for CA, CX 20MHz 動作用定義のディレクトリ
r_mtl_com.h	CS+ for CA, CX 20MHz 動作用定義ヘッダファイル
¥r178_32MHz_CCRL <DIR>	CS+ for CC 32MHz 動作用定義のディレクトリ
r_mtl_com.h	CS+ for CC 32MHz 動作用定義ヘッダファイル
¥r178_24MHz_CCRL <DIR>	CS+ for CC 24MHz 動作用定義のディレクトリ
r_mtl_com.h	CS+ for CC 24MHz 動作用定義ヘッダファイル
¥r178_20MHz_CCRL <DIR>	CS+ for CC 20MHz 動作用定義のディレクトリ
r_mtl_com.h	CS+ for CC 20MHz 動作用定義ヘッダファイル
¥r178_32MHz_IAR <DIR>	IAR 32MHz 動作用定義のディレクトリ
r_mtl_com.h	IAR 32MHz 動作用定義ヘッダファイル
¥r178_24MHz_IAR <DIR>	IAR 24MHz 動作用定義のディレクトリ
r_mtl_com.h	IAR 24MHz 動作用定義ヘッダファイル
¥r178_20MHz_IAR <DIR>	IAR 20MHz 動作用定義のディレクトリ
r_mtl_com.h	IAR 20MHz 動作用定義ヘッダファイル

ディレクトリ構成、 ディレクトリ名/ファイル名	備考
¥mmc <DIR>	<b>MMC 用デバイスドライバのディレクトリ</b>
r_mmc.h	MMC ドライバ共通定義
r_mmc_io.c, r_mmc_io.h	MMC ドライバ SPI モード I/O モジュール
r_mmc_mmc.c	MMC ドライバ SPI モード MMC モジュール
r_mmc_sub.c, r_mmc_sub.h	MMC ドライバ SPI モード・サブモジュール
r_mmc_usr.c	MMC ドライバ SPI モード・API ソースプログラム
r_mmc_mcu_pragma.h	MMC ドライバ プラグマ定義 ヘッダファイル
¥rl78 <DIR>	<b>RL78 ファミリ ディレクトリ</b>
r_mmc_csi.c	MMC ドライバ SPI モード CSI を用いた通信モジュール
¥rl78g14_csi <DIR>	<b>RL78/G14 グループ CSI 用 SFR 定義ディレクトリ (CS+ for CA, CX 対応)</b>
r_mmc_sfr.h	MMC ドライバ SFR 定義
r_mmc_user_config.h	MMC ユーザ定義ヘッダファイル
¥rl78g14_csi_ccrl <DIR>	<b>RL78/G14 グループ CSI 用 SFR 定義ディレクトリ (CS+ for CA, CX 対応)</b>
r_mmc_sfr.h	MMC ドライバ SFR 定義
r_mmc_user_config.h	MMC ユーザ定義ヘッダファイル
¥rl78g14_csi_iar <DIR>	<b>RL78/G14 グループ CSI 用 SFR 定義ディレクトリ (IAR 対応)</b>
r_mmc_sfr.h	MMC ドライバ SFR 定義
r_mmc_user_config.h	MMC ユーザ定義ヘッダファイル

## 2. プログラム型定義

本プログラムの整数型の定義は以下のとおりとします。

Datatype	Typedef
unsigned char	uint8_t
unsigned short	uint16_t
unsigned long	uint32_t
signed char	int8_t
signed short	int16_t
signed long	int32_t
int16_t	natural_int_t
uint16_t	natural_uint_t

## 3. デバイスドライバ

## 3.1 デバイスドライバ関数概要

## 初期化関数

関数名	機能概要
R_mmc_Init_Driver ()	MMC ドライバ初期化処理

## デバイス操作関数

関数名	機能概要
R_mmc_Init_Slot()	MMC スロット初期化処理
R_mmc_Detach()	MMC スロット停止処理
R_mmc_Chk_Detect()	MMC 挿入チェック処理

## データ・アクセス操作関数

関数名	機能概要
R_mmc_Read_Data()	データ読み出し処理
R_mmc_Write_Data()	データ書き込み処理
R_mmc_Get_MmcInfo()	MMC 情報取得処理

## 内部使用コマンド一覧

コマンドインデックス	コマンド名
CMD0	GO_IDLE_STATE
CMD1	SEND_OP_COND
CMD9	SEND_CSD
CMD10	SEND_CID
CMD12	STOP_TRANSMISSION
CMD13	SEND_STATUS
CMD17	READ_SINGLE_BLOCK
CMD18	READ_MULTIPLE_BLOCK
CMD24	WRITE_BLOCK
CMD25	WRITE_MULTIPLE_BLOCK
CMD58	READ_OCR
CMD59	CRC_ON_OFF

【注】 未サポート・コマンドに対しては、ユーザ側で対応してください。



## 3.2 関数詳細

### 3.2.1 MMC ドライバ初期化処理(R\_mmc\_Init\_Driver)

項目	内容
プロトタイプ	void R_mmc_Init_Driver(void)
引数	なし
説明	MMC ドライバの初期化を行います。 MMC 制御用 SFR の初期化、スロット毎の制御ポートと RAM を初期化します。 本関数はシステム起動時に一度だけ呼び出してください。
戻り値	なし

### 3.2.2 MMC スロット初期化処理(R\_mmc\_Init\_Slot)

項目	内容
プロトタイプ	int16_t R_mmc_Init_Slot(uint8_t SlotNo)
引数	uint8_t SlotNo : スロット番号
説明	引数で指定されたスロットに対する RAM およびポートの初期化を行います。 また、MMC に対して初期化処理を行います。 本関数はカード挿入検出時に呼び出してください。
戻り値	初期化結果を返します。 MMC_OK : 正常終了 MMC_ERR_PARAM : パラメータエラー MMC_ERR_HARD : ハードウェアエラー MMC_ERR_CRC : CRC エラー MMC_ERR_IDLE : Idle state エラー MMC_ERR_OTHER : その他エラー

### 3.2.3 MMC スロット停止処理(R\_mmc\_Detach)

項目	内容
プロトタイプ	int16_t R_mmc_Detach(uint8_t SlotNo)
引数	uint8_t SlotNo : スロット番号
説明	指定スロット MMC 引抜き時の処理を行います。 MMC 制御用 SFR の初期化、制御 port の開放、制御用 RAM の初期化を行います。 本関数はカード引抜き検出時に呼び出してください。
戻り値	引き抜き処理の結果を返します。 MMC_OK : 正常終了 MMC_ERR_PARAM : パラメータエラー

## 3.2.4 MMC 挿入チェック処理(R\_mmc\_Chk\_Detect)

項目	内容
プロトタイプ	int16_t R_mmc_Chk_Detect(uint8_t SlotNo, uint8_t *pDetSts)
引数	uint8_t SlotNo : スロット番号 uint8_t *pDetSts : MMC 挿入状態格納先バッファポインタ
説明	<p>引数で指定されたスロットに対して MMC 挿入状態のチェックを行います。戻り値が MMC_OK の場合、MMC 挿入状態格納先バッファ(pDetSts)には MMC 挿入検出端子状態が格納されます。</p> <ul style="list-style-type: none"> <li>• MMC_TRUE : MMC 挿入検出端子 Active</li> <li>• MMC_FALSE : MMC 挿入検出端子 Non Active</li> </ul> <p>この処理内でチャタリングの除去は行いません。 上位にて必要なチャタリングの除去を行ってください。 定周期でのポーリングによるメディアの挿入確認を推奨します。</p>
戻り値	<p>チェック結果を返します。</p> <p>MMC_OK : 正常終了 MMC_ERR_PARAM : パラメータエラー</p>

## 3.2.5 データ読み出し処理(R\_mmc\_Read\_Data)

項目	内容
プロトタイプ	int16_t R_mmc_Read_Data(uint8_t SlotNo, uint32_t BlkNo, uint32_t BlkCnt, uint8_t *pData, uint8_t Mode)
引数	uint8_t SlotNo : スロット番号 uint32_t BlkNo : 読み出し開始ブロック番号 uint32_t BlkCnt : 読み出しブロック数 uint8_t *pData : 読み出しデータ格納バッファポインタ uint8_t Mode : 読み出しデータ転送モード
説明	<p>MMC からブロック(512byte)単位でデータの読み出しを行います。指定ブロックから指定ブロック数分、データを読み出します。転送モード(Mode)は MMC_MODE_NORMAL(データを引数のデータ格納バッファ(pData)に転送するモード)を選択してください。</p> <p>MMC からの読み出しは、R_mmc_Get_MmcInfo()関数から渡される MMC 情報のうち、カード種別(MmcInfo.Card)が 'MMC_CARD_UNDETECT' でない場合のみ可能です。最大ブロック番号は、R_mmc_Get_MmcInfo()関数から渡される 'pMmcInfo.MaxBlkNum' です。最大ブロック数は、'pMmcInfo.MaxBlkNum'+1 です。</p>
戻り値	<p>読み出し結果を返します。</p> <p>MMC_OK : 正常終了 MMC_ERR_PARAM : パラメータエラー MMC_ERR_HARD : ハードウェアエラー MMC_ERR_CRC : CRC エラー MMC_ERR_OTHER : その他エラー</p>

## 3.2.6 データ書き込み処理(R\_mmc\_Write\_Data)

項目	内容
プロトタイプ	int16_t R_mmc_Write_Data(uint8_t SlotNo, uint32_t BlkNo, uint32_t BlkCnt, uint8_t *pData, uint8_t Mode)
引数	uint8_t SlotNo : スロット番号 uint32_t BlkNo : 書き込み開始ブロック番号 uint32_t BlkCnt : 書き込みブロック数 uint8_t *pData : 書き込みデータ格納バッファポインタ uint8_t Mode : 書き込みデータ転送モード
説明	MMC ヘブロック(512byte)単位でデータの書き込みを行います。 指定ブロックから指定ブロック数分、データを書き込みます。 転送モード(Mode)は MMC_MODE_NORMAL(データを引数のデータ格納バッファ(pData)から転送するモード)を選択してください。 MMC への書き込みは、R_mmc_Get_MmcInfo()関数から渡される MMC 情報のうち、カード種別(MmcInfo.Card)が 'MMC_CARD_UNDETECT' でない場合のみ可能です。 最大ブロック番号は、R_mmc_Get_MmcInfo()関数から渡される 'pMmcInfo.MaxBlkNum' です。 最大ブロック数は、'pMmcInfo.MaxBlkNum'+1 です。
戻り値	MMC 情報取得結果を返します。 MMC_OK : 正常終了 MMC_ERR_PARAM : パラメータエラー MMC_ERR_HARD : ハードウェアエラー MMC_ERR_WP : ライトプロテクトエラー MMC_ERR_OTHER : その他エラー

## 3.2.7 MMC 情報取得処理(R\_mmc\_Get\_MmcInfo)

項目	内容
プロトタイプ	int16_t R_mmc_Get_MmcInfo(uint8_t SlotNo, MMC_INFO* pMmcInfo)
引数	uint8_t SlotNo : スロット番号 MMC_INFO* pMmcInfo : MMC 情報格納バッファポインタ
説明	<p>MMC 情報を返します。 MMC 情報格納バッファ(pMmcInfo)には MMC 情報が格納されます。</p> <ul style="list-style-type: none"> <li>• pMmcInfo.Card : カード種別 <ul style="list-style-type: none"> <li>— MMC_CARD_UNDETECT : カード未検出</li> <li>— MMC_CARD_MMC : MMC</li> <li>— MMC_CARD_OTHER : その他カード</li> </ul> </li> <li>• pMmcInfo.WProtect : ライトプロテクト状態 <ul style="list-style-type: none"> <li>— MMC_NO_PROTECT : ライトプロテクト解除</li> <li>— MMC_W_PROTECT_SOFT : ソフトライトプロテクト</li> </ul> </li> <li>• pMmcInfo.MemSize : カード容量(byte)</li> <li>• pMmcInfo.MaxBlkNum : 最大ブロック番号(メディアの最大ブロック番号)</li> </ul> <p>'pMmcInfo.MemSize'が 0xFFFFFFFF の場合は、'pMmcInfo.MaxBlkNum'+1 が、メディアの持つブロック数 (1 ブロック=512 バイト) を示します。 カード容量の計算が必要な場合、('pMmcInfo.MaxBlkNum'+1)×512 で計算してください。</p>
戻り値	<p>書き込み結果を返します。</p> <ul style="list-style-type: none"> <li>MMC_OK : 正常終了</li> <li>MMC_ERR_PARAM : パラメータエラー</li> <li>MMC_ERR_OTHER : その他エラー</li> </ul>

### 3.3 データ構造体

データ構造体を以下に示します。

MMC 情報構造体定義

```
typedef struct {
    uint8_t    Card;           /* Card type                */
    uint8_t    WProtect;      /* Write-protection status  */
    uint32_t   MemSize;       /* Card capacity            */
    uint32_t   MaxBlkNum;     /* The number of the max blocks */
} MMC_INFO;
```

### 3.4 マクロ定義

マクロ定義を以下に示します。

```
/*----- Definitions of return value -----*/
#define MMC_OK          (int16_t)( 0)      /* Successful operation     */
#define MMC_ERR_PARAM   (int16_t)(-1)     /* Parameter error         */
#define MMC_ERR_HARD    (int16_t)(-2)     /* Hardware error          */
#define MMC_ERR_CRC     (int16_t)(-3)     /* CRC error               */
#define MMC_ERR_WP      (int16_t)(-4)     /* Write-protection error  */
#define MMC_ERR_MBLKCMD (int16_t)(-5)     /* Multi-block command error */
#define MMC_ERR_IDLE    (int16_t)(-6)     /* Idle state error        */
#define MMC_ERR_OTHER   (int16_t)(-7)     /* Other error             */

/*----- Definitions of flag -----*/
#define MMC_TRUE        (uint8_t)0x01     /* Flag "ON"               */
#define MMC_FALSE      (uint8_t)0x00     /* Flag "OFF"              */

/*----- Definition of card type -----*/
#define MMC_CARD_UNDETECT (uint8_t)0x00   /* Card is not found       */
#define MMC_CARD_MMC     (uint8_t)0x01   /* MMC                     */
#define MMC_CARD_OTHER   (uint8_t)0xFF   /* Other card              */

/*----- Definitions of write-protection status -----*/
#define MMC_NO_PROTECT   (uint8_t)0x00   /* None setting           */
#define MMC_W_PROTECT_SOFT (uint8_t)0x02 /* Software write-protection */
```

## 4. 設定例

設定例は、RL78/G14 使用時の説明になります。RL78/G13 使用時のポーティング方法は、4.3 節に記載しています。

### 4.1 共通関数 r\_mtl\_XXX の可変データ設定例

各システムのリソースに合わせて設定をする部分です。

設定箇所は、各ファイル中の「/\*\* SET \*\*/」というコメントの部分です。

ファイル毎に抜粋を示し、詳細な解説を加えます。

#### 4.1.1 r\_mtl\_com.h

共通で使用される共通関数のヘッダです。

r\_mtl\_com.h は MCU およびシステムクロック別に個々に用意しています。

ご使用の環境に合わせて下記の表に示すディレクトリにある r\_mtl\_com.h をインクルードしてください。

異なる MCU やシステムクロックで動作させる場合は、ユーザでご用意ください。

使用 MCU – システムクロック	インクルードディレクトリ(source ディレクトリ以降)
RL78/ G14 – 32MHz (CS+ for CA, CX 対応)	¥com¥ r178_32MHz
RL78/ G14 – 24MHz (CS+ for CA, CX 対応)	¥com¥ r178_24MHz
RL78/ G14 – 20MHz (CS+ for CA, CX 対応)	¥com¥ r178_20MHz
RL78/ G14 – 32MHz (CS+ for CC 対応)	¥com¥ r178_32MHz_CCRL
RL78/ G14 – 24MHz (CS+ for CC 対応)	¥com¥ r178_24MHz_CCRL
RL78/ G14 – 20MHz (CS+ for CC 対応)	¥com¥ r178_20MHz_CCRL
RL78/ G14 – 32MHz (IAR 対応)	¥com¥ r178_32MHz_IAR
RL78/ G14 – 24MHz (IAR 対応)	¥com¥ r178_24MHz_IAR
RL78/ G14 – 20MHz (IAR 対応)	¥com¥ r178_20MHz_IAR

#### (1) ループタイマの定義

— ソフトウェア・ループタイマを使用する場合、r\_mtl\_tim.h をインクルードします。

主にデバイスドライバが待ち時間を確保するために、使用します。

ソフトウェア・ループタイマを使用しない場合は、下記インクルードをコメントにしてください。

下記の例は、ソフトウェア・ループタイマを使用する場合の例です。

本ドライバ使用時は、インクルードしてください。また、r\_mtl\_tim.h からシステムクロックに合ったマクロを定義してください。RL78 ファミリの MCU を 32MHz で動作する場合は "MTL\_TIM\_RL78\_\_32MHz\_noWait" を定義します。

```
/* When not using the loop timer, put the following 'include' as comments. */
#define MTL_TIM_RL78__32MHz_noWait

#include "r_mtl_tim.h"
```

#### (2) エンディアンタイプ定義

— RL78 ファミリでは、リトルエンディアン固定です。

```
#define MTL_MCU_LITTLE /* Little Endian */ /** SET **/
```

## (3) 使用する標準ライブラリのタイプの定義

— 使用する標準ライブラリのタイプを定義してください。

下記に示す処理を標準ライブラリで使用する場合は、下記マクロ定義をコメントにしてください。

下記の例は、コンパイラ添付のライブラリを使用しない場合の例です。

```
/* Specify the type of user standard library.                *** SET **/
/* When using the compiler-bundled library for the following processes, *** SET **/
/* put the following 'define' as comments.                  *** SET **/
/* memcmp()/memmove()/memcpy()/memset()/strcat()/strcmp()/strcpy()/strlen()*** SET **/
#define MTL_USER_LIB          /* use optimized library      */ /** SET **/
```

## 4.1.2 r\_mtl\_tim.h

r\_mtl\_com.h にて、ループタイマを定義した場合に、インクルードされます。

使用する MCU やクロック、ウェイトに依存します。

環境に合うものがない場合は、ユーザにて定義を作成してください。

下記記載のカウント数は、実測値での値になっています。

```
/* Define the counter value for the timer.                  */
/* Specify according to the user MCU, clock and wait requirements. */
/*                                                                */
/* Set the reference value to 10% more than the actual calculated value. */
/*=====*/

/*=====*/
#ifdef MTL_TIM_RL78__32MHz_noWait
/* Setting for 32.0MHz no wait */
#define MTL_T_1US      3          /* loop Number of 1us */ /** SET **/
#define MTL_T_2US      8          /* loop Number of 2us */ /** SET **/
#define MTL_T_4US     17          /* loop Number of 4us */ /** SET **/
#define MTL_T_5US     21          /* loop Number of 5us */ /** SET **/
#define MTL_T_10US    44          /* loop Number of 10us */ /** SET **/
#define MTL_T_20US    90          /* loop Number of 20us */ /** SET **/
#define MTL_T_30US   136          /* loop Number of 30us */ /** SET **/
#define MTL_T_50US   227          /* loop Number of 50us */ /** SET **/
#define MTL_T_100US  456          /* loop Number of 100us */ /** SET **/
#define MTL_T_200US  913          /* loop Number of 200us */ /** SET **/
#define MTL_T_300US 1370          /* loop Number of 300us */ /** SET **/
#define MTL_T_400US 1827          /* loop Number of 400us */ /** SET **/
#define MTL_T_1MS   4572          /* loop Number of 1ms */ /** SET **/
#endif

#ifdef MTL_TIM_RL78__24MHz_noWait
/* Setting for 24.0MHz no wait */
(省略)
#endif

#ifdef MTL_TIM_RL78__20MHz_noWait
/* Setting for 20.0MHz no wait */
(省略)
#endif
```

## 4.2 MMC ドライバ可変データ設定例

各システムのリソースに合わせて設定をする部分です。

設定箇所は、各ファイル中の「`/** SET **/`」というコメントの部分です。

ファイル毎に抜粋を示し、詳細な解説を加えます。

### 4.2.1 r\_mmc.h(ドライバ共通定義)

#### (1) デバイス数、デバイス番号の定義

— メモリカードのスロット数を定義してください。

```

/* Define number of required card slots. (1-N slots) */
/* Define slot number in accordance with the number of card slots to be connected. */
/*-----*/
/* Define number of slots (devices). */
#define MMC_SLOT_NUM      1                /* 1slots          */ /** SET **/

/* Define slot number. */
#define MMC_SLOT0        0                /* Slot 0          */ /** SET **/
#define MMC_SLOT1        1                /* Slot 1          */ /** SET **/

```

#### (2) SPI モード マルチブロックコマンド非対応カードの場合の定義

— デフォルトでは、`MULTIPLE_BLOCK` コマンド非対応カードの場合は、`READ_SINGLE_BLOCK` コマンドと `WRITE_SINGLE_BLOCK` コマンドで対応する設定のため、このままの設定を推奨します。

```

/* When use the card which does not support a multi-block command, please define it.*/
/* Use single block commands in the case of the card which does not support multiple block*/
/* commands. */
#define MMC_SBLK_CMD      /* Support single block commands */ /** SET **/

```

#### (3) 対象メディアの定義

— `MMC_SUPPORT_MMC` を定義してください。

```

/*-----*/
/* Please define the media to support. */
/*-----*/
#define MMC_SUPPORT_MMC      /* MMC          */ /** SET **/

```



## 4.2.2 r\_mmc\_user\_config.h (MCU 個別定義)

## (1) 使用 MCU の選択

r\_mmc\_user\_config.h は MCU で個々に用意しています。ご使用の環境に合わせて下記の表に示すディレクトリにある r\_mmc\_user\_config.h をインクルードしてください。

使用 MCU - 通信モジュール	インクルードディレクトリ(source ディレクトリ以降)
RL78G14 – CSI (CS+ for CA, CX)	¥mmc¥rl78¥rl78g14_csi
RL78G14 – CSI (CS+ for CC)	¥mmc¥rl78¥rl78g14_csi_ccrl
RL78G14 – CSI (IAR)	¥mmc¥rl78¥rl78g14_csi_iar

## (2) 通信ユニットのチャンネル番号の定義

MMC\_SAU\_UNIT マクロで使用する通信ユニットを、MMC\_SAU\_CHANNEL マクロで使用する通信ユニットのチャンネル番号を定義してください。

通信ユニット SAU0 の CSI00 チャンネルもしくは CSI10 チャンネルを使用する場合、使用する端子を選択する必要があります。MMC\_CSI\_PIN マクロで使用する端子を選択してください。

```

/* Serial Array Unit(SAU) Select ( 0 or 1 )*/
#define MMC_SAU_UNIT          1          /** SET **/
/* SAU Channel Select ( 0 or 1 or 2 or 3 ) */
#define MMC_SAU_CHANNEL      1          /** SET **/
/* CSI PIN select ( 'A' or 'B' ) */
#define MMC_CSI_PIN          'A'        /** SET **/

```

```

/*

```

			Select Port		
MMC_SAU_	MMC_SAU_	MMC_CSI_	SI	SCK	S0
UNIT	CHANNEL	PIN	Select	Select	Select
Value	Value	Value	port	port	port
0 (=Use SAU0)	0	'A'	P50	P30	P51
	(=Use CSI00)	'B'	P16	P55	P17
	1	(invalid)	P74	P75	P73
	(=Use CSI01)				
	2	'A'	P03	P04	P02
	(=Use CSI10)	'B'	P81	P80	P82
1 (=Use SAU1)	0	(invalid)	P14	P15	P13
	(=Use CSI20)				
	1	(invalid)	P71	P70	P72
	(=Use CSI21)				
	2	(invalid)	P143	P142	P144
	(=Use CSI30)				
3	(invalid)	P53	P54	P52	
	(=Use CSI31)				

```

*/

```

## (3)使用制御ポートの定義

- DETECT(カード検出) 信号および CS(カードセレクト)信号を接続回路に合わせて定義してください。  
MMC スロットとマイコン間で DETECT 信号が配線されている場合、MMC\_DETECT0\_CONNECTION マクロを定義してください。

```

/*-----*/
/* Define the control port. */
/*-----*/
#define MMC_CS0_PORTNO      0          /* CS0 Port No. */ /** SET **/
#define MMC_CS0_BITNO      5          /* CS0 Bit No. */ /** SET **/

/* Please define the MMC_DETECT0_CONNECTION macro when the MMC slot Card detect
pin connect to MCU. */
#define MMC_DETECT0_CONNECTION      /* DETECT0 Port Connection */ /** SET **/

#if defined(MMC_DETECT0_CONNECTION)
#define MMC_DETECT0_PORTNO      0          /* DETECT0 Port No. */ /** SET **/
#define MMC_DETECT0_BITNO      0          /* DETECT0 Bit No. */ /** SET **/
#endif /* #if defined(MMC_DETECT0_CONNECTION) */

#if (MMC_SLOT_NUM > 1)

#define MMC_CS1_PORTNO          /* CS1 Port No. */ /** SET **/
#define MMC_CS1_BITNO          /* CS1 Bit No. */ /** SET **/

#define MMC_DETECT1_CONNECTION      /* DETECT1 Port Connection */ /** SET **/

#if defined(MMC_DETECT1_CONNECTION)
#define MMC_DETECT1_PORTNO      /* DETECT1 Port No. */ /** SET **/
#define MMC_DETECT1_BITNO      /* DETECT1 Bit No. */ /** SET **/
#endif /* #if defined(MMC_DETECT0_CONNECTION) */

```

## (4) 通信タイムアウト検出処理の定義

— 通信中のタイムアウト検出処理を省略することが出来ます。

省略する場合、MMC\_NOCHK\_TIMEOUT マクロを定義してください。定義した場合は処理速度が向上するメリットがありますが、通信機能に異常が発生した場合にプログラムが停止する可能性があるデメリットがあります。

省略しない場合は、タイムアウト時間を設定してください。

- MMC\_T\_CSI\_WAIT マクロで時間の単位を設定します。設定するマクロは r\_mtl\_tim.h から選択してください。
- MMC\_CSI\_TX\_WAIT マクロでデータ送信時のタイムアウト時間を定義してください。
- MMC\_CSI\_RX\_WAIT マクロでデータ受信時のタイムアウト時間を定義してください。
- 各タイムアウト時間マクロの設定値は (タイムアウト時間 / 単位) となります。

```

/*-----*/
/* Macro "MMC_NOCHK_TIMEOUT" omits detecting timeout during communication. */
/* If user omits detecting timeout, please define this macro. */
/* If this macro is defined, processing speed would be increased. */
/*-----*/
#define MMC_NOCHK_TIMEOUT /* No Check Communication Timeout */ /** SET **/

/*-----*/
/* If MMC_NOCHK_TIMEOUT would be not defined, please set timeout time. */
/* MMC_T_CSI_WAIT is unit of measuring timeout. */
/* Please select value from "r_mtl_tim.h" */
/* Please set value of (timeout time/unit) to MMC_CSI_TX_WAIT(transmitting) */
/* and MMC_CSI_RX_WAIT(receiving). */
/*-----*/
/* CSI transmit&receive completion waiting polling time */
#define MMC_T_CSI_WAIT (natural_uint_t)MTL_T_1US /** SET **/

/* CSI transmission completion waiting time 50000 * lus = 50ms */
#define MMC_CSI_TX_WAIT (natural_uint_t)50000 /** SET **/
/* CSI receive completion waiting time 50000 * lus = 50ms */
#define MMC_CSI_RX_WAIT (natural_uint_t)50000 /** SET **/

```

## (5) 使用リソースの定義

— MCU に応じて、最適化した MCU リソースの組み合わせを定義しています。

以下の定義のうち、1つ定義してください。

```

/*-----*/
/* Define the combination of the MCU's resources. */
/*-----*/
// #define MMC_OPTION_1 /* CSI */ /** SET **/
#define MMC_OPTION_2 /* CSI + CRC calculation circuit */ /** SET **/

```

## (6) 通信ボーレートの定義

— 通信ボーレートを定義してください。

使用する CSI の定義を修正する場合は、各々の CSI レジスタに応じた SFR 設定が必要になりますので、MCU のデータシートを参照し、システムに応じた SFR 設定を行ってください。

特に、通信速度設定に関して、Identification mode/Data Transfer mode それぞれのカード仕様書の tODLY を満たすように設定する必要があります。

さらに、カード仕様書では、Identification mode では、tOD ( $100\text{KHz} \leq tOD \leq 400\text{KHz}$ )、Data Transfer mode では、tPP ( $0.1\text{MHz} \leq tPP \leq 20\text{MHz}$ ) を満たすように設定する必要があります。

なお、tOD, tPP は、本システムにおいて、クロック周波数を示すことになります。

MCU により、サポート可能な CSI のクロック周波数が変わりますので、MCU のデータシートを参照してください。

- MMC\_FCLK\_PRESCALER\_SELECT マクロ

fCLK(源クロック)から分周して fMCK を設定します。設定値は SPSm レジスタの PRSm3~PRSm0 ビットに反映されます。MMC\_FCLK\_PRESCALER\_SELECT マクロには 0~15 の値を設定してください。

- MMC\_OPERATION\_CLK\_SELECT マクロ

SAU(シリアル・アレイ・ユニット)のチャンネル(最大 4 チャンネル)毎に fMCK の選択をおこないます。設定値は SMSm レジスタの CKSmn ビットに反映されます。MMC\_OPERATION\_CLK\_SELECT マクロには 0(CKm0)もしくは 1(CKm1)を任意で設定してください。

- MMC\_UBRG\_IDENTIFICATION マクロ

Identification mode 時の転送クロックを設定します。設定値は SDRmn レジスタの上位 7 ビット側に反映され、fMCK から分周された転送クロック(fTCLK)となります。MMC\_UBRG\_IDENTIFICATION マクロには、0~63 の値を設定してください。

- MMC\_UBRG\_D\_TRANSFER マクロ

Transfer mode 時の転送クロックを設定します。設定値は SDRmn レジスタの上位 7 ビット側に反映され、fMCK から分周された転送クロック(fTCLK)となります。

MMC\_UBRG\_D\_TRANSFER マクロには、0~63 の値を設定してください。

- MMC\_CLK\_D\_TRANSFER マクロ

MMC\_CLK\_D\_TRANSFER マクロには、Transfer mode 時のクロック周波数を設定してください。設定値はカード仕様書 N<sub>AC</sub> Cycles の監視に使われます。

— 通信ボーレートの設定と定義例を示します。

- fCLK=32MHz(HOCO), (fMCK=16MHz,) fTCLK=8MHz

r\_mmc\_user\_config.h の通信ボーレートを定義してください。下記に定義例を示します。

```

/* Define the value of the bit rate register according to a communication baud rate.          */
(省略)

/* fCLK = 32MHz , fMCK = 16MHz , fTCLK = 8MHz */
#define MMC_FCLK_PRESCALER_SELECT 1                /* SPSm.PRSmk[3:0]          */ /** SET **/

#define MMC_OPERATION_CLK_SELECT 0                /* select SMRm.CKmX 0:CKm0 1:CKm1 */ /** SET **/

#define MMC_UBRG_IDENTIFICATION (uint8_t)19      /* BRR identification mode setting*/ /** SET **/
/*                +-+----- 400KHz                */ /** SET **/
#define MMC_UBRG_D_TRANSFER (uint8_t)0          /* BRR data Transfer mode setting */ /** SET **/
/*                +-+----- 8.0MHz                */ /** SET **/
#define MMC_CLK_D_TRANSFER (uint32_t)8000000 /* Data Transfer mode clock frequency */ /** SET **/

```

- fCLK=24MHz(HOCO), (fMCK=24MHz,) fTCLK=12MHz

r\_mmc\_user\_config.h の通信ボーレートを定義してください。下記に定義例を示します。

```

/*-----*/
/* Define the value of the bit rate register according to a communication baud rate.          */
(省略)

/* fCLK = 24MHz , fMCK = 24MHz , fTCLK = 12MHz */
#define MMC_FCLK_PRESCALER_SELECT 0                /* SPSm.PRSmk[3:0]                */ /** SET **/

#define MMC_OPERATION_CLK_SELECT 0                /* select SMRm.CKmX 0:CKm0 1:CKm1 */ /** SET **/

#define MMC_UBRG_IDENTIFICATION (uint8_t)29      /* BRR identification mode setting*/ /** SET **/
/*                +----- 400KHz                */ /** SET **/
#define MMC_UBRG_D_TRANSFER (uint8_t)0          /* BRR data Transfer mode setting */ /** SET **/
/*                +----- 12.0MHz                */ /** SET **/
#define MMC_CLK_D_TRANSFER (uint32_t)12000000 /* Data Transfer mode clock frequency*/ /** SET **/

```

- fCLK=20MHz(XIN), (fMCK=20MHz,) fTCLK=10MHz

r\_mmc\_user\_config.h の通信ボーレートを定義してください。下記に定義例を示します。

```

/*-----*/
/* Define the value of the bit rate register according to a communication baud rate.          */
(省略)

/* fCLK = 20MHz , fMCK = 20MHz , fTCLK = 10MHz */
#define MMC_FCLK_PRESCALER_SELECT 0                /* SPSm.PRSmk[3:0]                */ /** SET **/

#define MMC_OPERATION_CLK_SELECT 0                /* select SMRm.CKmX 0:CKm0 1:CKm1 */ /** SET **/

#define MMC_UBRG_IDENTIFICATION (uint8_t)24      /* BRR identification mode setting */ /** SET **/
/*                +----- 400KHz                */ /** SET **/
#define MMC_UBRG_D_TRANSFER (uint8_t)0          /* BRR data Transfer mode setting */ /** SET **/
/*                +----- 10.0MHz                */ /** SET **/
#define MMC_CLK_D_TRANSFER (uint32_t)10000000 /* Data Transfer mode clock frequency*/ /** SET **/

```

### 4.3 RL78/G13 使用時のポーティング方法

RL78/G14 使用時との違いが発生するのは、4.2.2 節の(1)と(2)のみです。

それ以外は、4.1 節、4.2 節に記載されている内容を参照してください。

本節では、4.2.2 節「(2) 通信ユニットのチャンネル番号の定義」に該当する部分について説明します。変更するファイルは、`r_mmc_user_cofing.h` と `r_mmc_sfr.h` になります。

4.2.2 節「(1) 使用 MCU の選択」のインクルードディレクトリ名については、任意でリネームしてください。

#### 4.3.1 `r_mmc_user_config.h`

通信ユニットや通信ユニットのチャンネル番号と、使用可能な端子の組み合わせは、MCU や対応ピン数で異なります。

お使いの MCU の「ユーザーズマニュアル ハードウェア編」のシリアル・アレイ・ユニットの章に、通信ユニット(MMC\_SAU\_UNIT)、通信ユニットのチャンネル番号(MMC\_SAU\_CHANNEL)、CSI チャンネルとの組み合わせが記載されています。

それを元に 4.2.2 節にも記載した

MMC_SAU_UNIT Value	MMC_SAU_CHANNEL Value	MMC_CSI_PIN Value	SI Select port	SCK Select port	S0 Select port
0 (=Use SAU0)	0	'A'	P50	P30	P51
	(=Use CSI00)	'B'	P16	P55	P17
	1	(invalid)	P74	P75	P73
	(=Use CSI01)				
	2	'A'	P03	P04	P02
	(=Use CSI10)	'B'	P81	P80	P82
1 (=Use SAU1)	0	(invalid)	P14	P15	P13
	(=Use CSI20)				
	1	(invalid)	P71	P70	P72
	(=Use CSI21)				
	2	(invalid)	P143	P142	P144
	(=Use CSI30)				
3	(invalid)	P53	P54	P52	
	(=Use CSI31)				

の更新を行います。

例えば、RL78/G13 32 ピン製品の場合は、RL78/G13 ユーザーズマニュアル「第 12 章 シリアル・アレイ・ユニット」に記載されている表(表 1参照)を元に、下記のように更新できます。

MMC_SAU_ UNIT Value	MMC_SAU_ CHANNEL Value	MMC_CSI_ PIN Value	SI Select port	SCK Select port	SO Select port
0 (=Use SAU0)	0	(invalid)	P11	P10	P12
	1	---	---	---	---
	2	---	---	---	---
	3 (=Use CSI11)	(invalid)	P50	P30	P51
1 (=Use SAU1)	0	(invalid)	P14	P15	P13
	1	---	---	---	---

SI、SCK、SO へのポート割り当ては、RL78/G13 ユーザーズマニュアル「第 1 章 概説」などに記載されています。

○30, 32ピン製品

ユニット	チャンネル	CSIとして使用	UARTとして使用	簡易I <sup>2</sup> Cとして使用
0	0	CSI00	UART0	IIC00
	1	—		—
	2	—	UART1	—
	3	CSI11		IIC11
1	0	CSI20	UART2 (LIN-bus対応)	IIC20
	1	—		—

表 1 通信機能割り当て(RL78/G13 ユーザーズマニュアル Rev.3.10 から抜粋)

接続回路に合わせて、通信ユニットのチャンネル番号定義を更新します。

今回は、CSI20 チャンネルを使用とします。

```

/* Serial Array Unit(SAU) Select ( 0 or 1 )*/
#define MMC_SAU_UNIT          1          /** SET **/
/* SAU Channel Select ( 0 or 1 or 2 or 3 ) */
#define MMC_SAU_CHANNEL      0          /** SET **/
    
```

RL78/G13 32 ピン製品では CSI チャンネルに対して、割り当てられている端子は 1 組のみです。

“#define MMC\_CSI\_PIN”は使用しません。

### 4.3.2 r\_mmc\_sfr.h

select port などの設定を行います。

変更対象はファイル上部にある下記 12 定義です。それ以外は変更せずに使用してください。

#### 1. #define MMC\_CSI\_UNIT

CSI のユニット番号を設定します(CSI<sub>mn</sub> の m)。  
CSI20 ならば”2”。

#### 2. #define MMC\_CSI\_CHANNEL

CSI のチャンネル番号を設定します(CSI<sub>mn</sub> の n)。  
CSI20 ならば”0”。

#### 3. #define MMC\_DATAI\_PORTNO

DATA 入力ライン(SI)として割り当てたポート番号を設定します。  
P14 ならば”1”。

#### 4. #define MMC\_DATAI\_BITNO

DATA 入力ライン(SI)として割り当てたポートのビット番号を設定します。  
P14 ならば”4”。

#### 5. #define MMC\_CLK\_PORTNO

CLOCK ラインとして割り当てたポート番号を設定します。  
P15 ならば”1”。

#### 6. #define MMC\_CLK\_BITNO

CLOCK ライン(SCK)として割り当てたポートのビット番号を設定します。  
P15 ならば”5”。

#### 7. #define MMC\_DATAO\_PORTNO

DATA 出力ライン(SO)として割り当てたポート番号を設定します。  
P13 ならば”1”。

#### 8. #define MMC\_DATAO\_BITNO

DATA 出力ライン(SO)として割り当てたポートのビット番号を設定します。  
P13 ならば”3”。

#### 9. #define MMC\_CSI\_SIR\_CLEAR

シリアル・フラグ・クリア・トリガ・レジスタ(SIR) のフラグクリアをする為の設定値になります。  
対象ビットに 1 を設定するとフラグがクリアされます。

FECT ビット(フレーミング・エラー・フラグのクリア・トリガ)は、有効となるユニット/チャンネルの組み合わせに限られます。詳細は、MCU のユーザーズマニュアルでご確認ください。

FECT ビットが有効となる組み合わせ時は”7”を設定してください。

FECT ビットが有効でない組み合わせ時は”3”を設定してください。



## 5. MCU との接続方法と使用する MCU リソース

### 5.1 使用する MCU リソース

本プログラムは、以下の制御を行っています。

データの入出力を、3 線シリアル(内部クロックを使用)で、制御します。

3 線シリアルを割り当てる際には、高速動作させるため CMOS 出力可能な端子割り当てと CMOS 出力設定をしてください。

送信制御は、送信バッファの空きを検出して、割り込み使用せずに送信割り込み要求ビットを利用しています。したがって、割り込み関連を以下のように設定しています。

- 割り込み処理を、“1”(割り込み処理禁止)に設定。
- MMC の CS#端子を MCU の Port に接続し、MCU 汎用ポート出力で制御する。

使用するリソース	RL78/G14
3 線シリアル(CSI)	◎
CRC 演算回路	○
CS#用ポート 1 本/Card	◎
カード検出用ポート 1 本/Card	○
電源制御用ポート 1 本/Card	◎

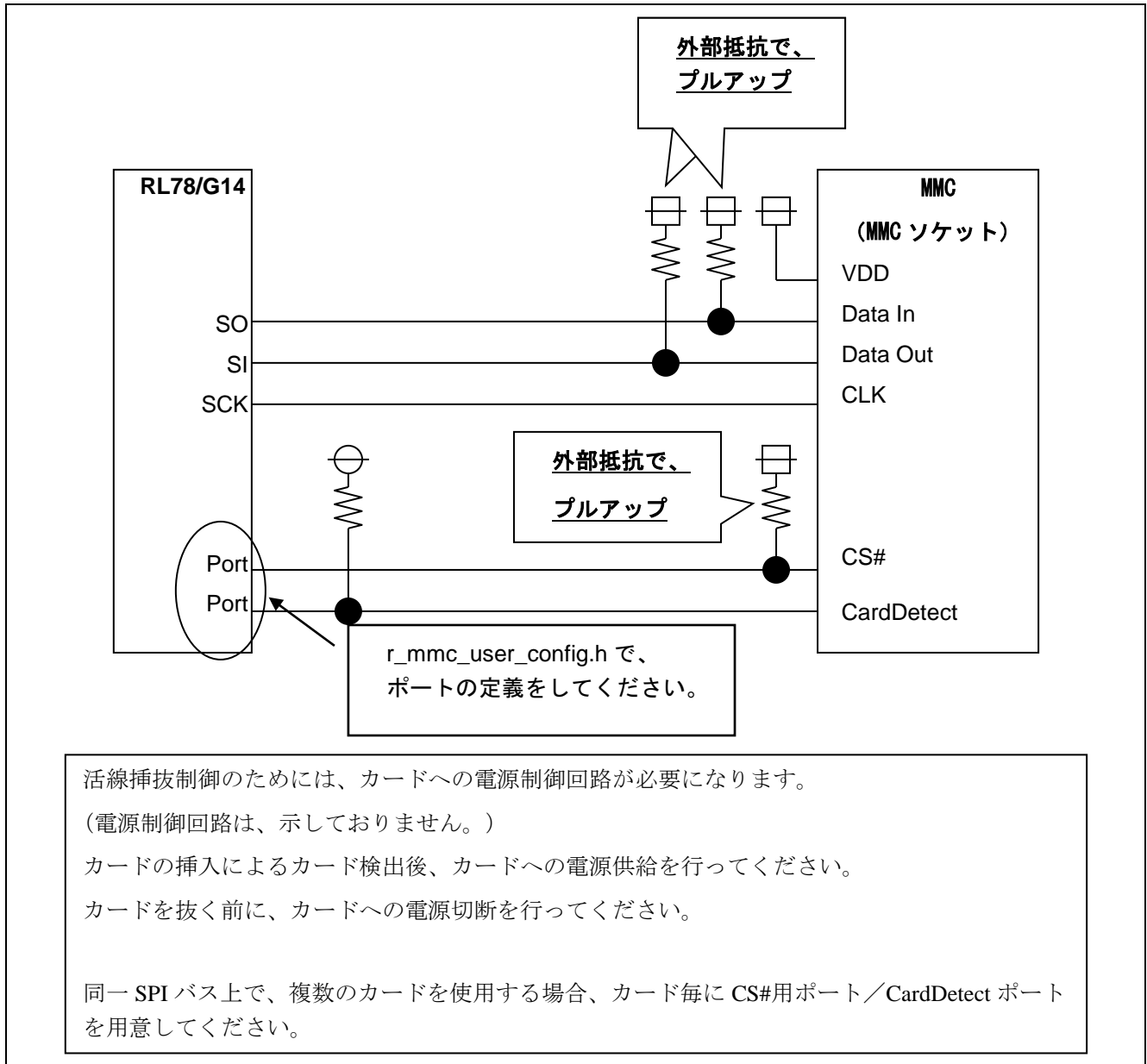
◎ : 必須

○ : 使用することを推奨。(RL78 ファミリのリソースを使う場合、高速化が可能)

## 5.2 MCU との接続方法

RL78/G14 との接続例を示します。

他 RL78 ファミリ MCU の場合であっても、同様の接続となります。



## 6. アプリケーション作成時の注意事項

### 6.1 使用上の注意事項

- 使用にあたっては、ハードウェアに合わせてソフトウェアを設定してください。
- メモリカードを非アクティブにし、MCU-メモリカード間の信号を Hi-z にし、デバイスへの電源供給を停止させた後、メモリカードを抜いてください。動作中にデバイスを抜いた場合、デバイスが壊れる可能性があります。
- 活線挿抜対応の回路を実現していない場合、活線挿抜を行うと電源等が不安定になり MCU がリセット状態になる可能性があります。

### 6.2 開発環境

弊社の開発環境を以下に示します。

ユーザアプリケーション開発時は以下のバージョンより新しいものをご使用下さい。

#### 6.2.1 CS+ for CA, CX

[統合開発環境]

CS+ for CA,CX V3.00.01

[C コンパイラ]

CA78K0R V1.71

#### 6.2.2 CS+ for CC

[統合開発環境]

CS+ for CC V3.01.00

[C コンパイラ]

CC-RL V1.01

#### 6.2.3 IAR Embedded Workbench

[ソフトウェアツール]

— -統合開発環境 および コンパイラ

IAR Embedded Workbench for Renesas RL78 V.2.10.1

-コード生成ツール

Applilet3 for RL78 V1.09.00 ルネサス製

[評価ボード]

IAR 版 MMC ドライバを用いたサンプルプログラムは、下記アプリケーションノートで使用しています。

ドキュメントタイトル: 「RL78/G14 CPU ボードを用いた音声再生・録音デモ」

(ドキュメント No. R20AN0194)

サンプルコードは下記 URL からダウンロードしてください。

[http://japan.renesas.com/products/tools/middleware/tiny\\_soft/adpcm/m3s\\_s2\\_tiny/app\\_notes.jsp](http://japan.renesas.com/products/tools/middleware/tiny_soft/adpcm/m3s_s2_tiny/app_notes.jsp)

## 6.3 組み込み時の注意事項

### 6.3.1 インクルードするファイル

MMC ドライバを組み込む場合は、`r_mtl_com.h` と `r_mmc.h` をインクルードしてください。

`r_mtl_com.h` を先にインクルードする必要があります。

### 6.3.2 コンフィグ設定 (`r_mmc_user_config.h`) での注意事項

- `MMC_CSI_PIN` マクロで 'B' を選択する場合の注意点

MCU のピン数によっては、CSI チャンネル(CSI00、CSI10)の使用端子を変更することができます。

`MMC_CSI_PIN` マクロの値を変更することで使用端子の選択を可能としています。'B' を選択する場合には、`PIOR0` レジスタで該当する端子のビットを設定する必要があります。

### 6.3.3 プログラム上の制限事項

SAUmEN ビットは通信開始時に 1(許可)に設定しますが通信終了時に 0(禁止)には設定しません。必要に応じて、ユーザにて 0(禁止)に設定してください。

## 6.4 ROM/RAM/スタックサイズ

MMC ドライバが使用する ROM/RAM サイズおよびスタックサイズは以下のとおりです。

### 6.4.1 CS+ for CA, CX

ROM size : 約 9KByte

RAM size : 約 1KByte (\*)

Stack size : 約 120byte

### 6.4.2 CS+ for CC

ROM size : 約 7KByte

RAM size : 約 1KByte (\*)

Stack size : 約 100byte

### 6.4.3 IAR Embedded Workbench

ROM size : 約 7KByte

RAM size : 約 1KByte (\*)

Stack size : 約 80byte

(\*) ユーザプログラムからの書込み、読出し用の各 512byte の領域確保分を含む

## 6.5 カードの挿抜検出に関する注意事項

カードのコネクタにあるカード検出端子を使用することで、`R_mmc_Chk_Detect()`を使ってカード挿抜の検出が可能です。したがって、カード検出のために定周期でのポーリングによるカードの挿入確認を推奨しません。

また、通信中においてカードが抜かれた状態になった場合、コマンドに対するレスポンス異常となり、結果的にドライバがエラーを返します。

最もケアしなければならない項目として、通信中に一瞬挿抜が行われた場合が挙げられます。

以下のような場合、ドライバがエラーを返さない可能性があります。

- 周期内の抜挿はドライバでは検出できず、カードからレスポンス異常が無い場合は正常動作を行います。
- 書き込み中に一瞬抜挿すると、ドライバが書き込み完了と認識してしまう可能性があります。これは、書き込み `Busy` 信号解除は `DataIn` 端子の "H" で検出する仕様であるためです。(DataIn 端子はプルアップされています。)

ハードウェア割り込み制御やポーリング周期の見直し等でシステムに適した方法で解決してください。

## 6.6 ポートの割り当てとカードの挿抜に関するポートの Hi-z 化処理の注意事項

- カードの挿入においては、カードの `CS#`, `DataIn`, `DataOut`, `CLK` 信号を Hi-z 状態にし、カードを挿入してください。その後、カードへの電源を供給してください。
- カードの抜去においては、カードへの電源供給停止後、カードの `CS#`, `DataIn`, `DataOut`, `CLK` 信号を Hi-z 状態にしてからカードを抜いてください。
- カードの `CS#`, `DataIn`, `DataOut`, `CLK` は、MCU の CSI、ポート端子に割り当てられていますが、そのポートが他のリソースに割り当てられている場合が想定されますので、本ドライバでは Hi-z 処理を行っておりません。したがって、カードの挿抜においては上位側で MCU 端子の Hi-z 処理をお願いします。

## 7. MMC ドライバ改訂履歴

Ver	変更点	リリース日時
1.03	<ul style="list-style-type: none"><li>・ CS+ for CC 対応</li><li>・ r_mmc_user_config.h から MMC_OPTION_3, MMC_OPTION_4 の選択を削除しました。</li></ul>	2015/10/01
1.02	<ul style="list-style-type: none"><li>・ RL78/G14 以外使用時のポーティング方法追加</li></ul>	2015/04/01
1.01	<ul style="list-style-type: none"><li>・ IAR Embedded Workbench に対応しました</li><li>・ Detect 端子が接続されていない場合でも通信可能となるようにコンフィグ設定を追加しました。</li></ul>	2014/09/01
1.00	新規リリース	2012/3/31

## 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/>

お問い合わせ先

<http://japan.renesas.com/contact/>

すべての商標および登録商標は、それぞれの所有者に帰属します。

## 改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.03	2015.10.01	—	CubeSuite+から CS+ for CA,CX に変更 CS+ for CC 対応 サンプルを削除 DTC 機能を用いた通信方法を削除
1.02	2015.04.01	—	RL78/G14 以外へのポーティング方法追加
1.01	2014.09.01	—	Ver.1.01 Release 00 に合わせてリリース
1.00	2012.03.31	—	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に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っていると、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、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と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 エレクトロニクス株式会社 〒135-0061 東京都江東区豊洲3-2-24（豊洲フォレシア）

■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/>