RXファミリ

R01AN0513JJ0233
Rev.2.33
Jul 31, 2019

USB Host Mass Storage Class Driver (HMSC)

要旨

本アプリケーションノートでは、USB Host マスストレージクラスドライバ(HMSC)について説明します。本ドライバは USB Basic Host Driver(USB-BASIC-FW)と組み合わせることで動作します。以降、本ドライバを HMSC と称します。

対象デバイス

RX62N/RX621 グループ
RX63N/RX631 グループ
RX63T グループ

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。

関連ドキュメント

1. Universal Serial Bus Revision 2.0 specification
2. USB Mass Storage Class Specification Overview Revision 1.1
3. USB Mass Storage Class Bulk-Only Transport Revision 1.0
   【http://www.usb.org/developers/docs/】
4. RX62N/RX621 グループユーザーズマニュアル ハードウェア編 (ドキュメント No.R01UH0033)
5. RX63N/RX631 グループユーザーズマニュアル ハードウェア編 (ドキュメント No.R01UH0041)
6. RX63T グループユーザーズマニュアル ハードウェア編 (ドキュメント No.R01UH0238)
7. RX ファミリ M3S-TFAT-Tiny : FAT ファイルシステムソフトウェア (ドキュメント No. R20AN0038)
8. USB Basic Host and Peripheral Driver アプリケーションノート(ドキュメント No.R01AN0512)

— ルネサス エレクトロニクスホームページ
【http://japan.renesas.com/】
— USB デバイスページ
【http://japan.renesas.com/prod/usb/】

目次

1. 概要 3
2. ソフトウェア構成 5
3. API情報 6
4. ターゲットペリフェラルリスト（TPL） 8
5. クラスドライバ概要 9
6. API 10
7. R_USB_GetEvent関数の戻り値 (USB_STS_MSC_CMD_COMPLETE) 15
8. サンプルアプリケーション 16
9. セットアップ 18
10. アプリケーションの作成方法 21
11. e² studio用プロジェクトをCS+で使用する場合 22

1. 概要

HMSC は、 と組み合わせることで、USB Host マスストレージクラスドライバ（以降 HMSC と記述）として動作します。

HMSC は、USB マスストレージクラスの Bulk-Only Transport (BOT) プロトコルで構築されています。ファイルシステム、ストレージデバイスドライバと組み合わせることで BOT 対応の USB ストレージ機器と通信を行うことが可能です。

また、本ドライバでは、M3S-TFAT-Tiny（以下 TFAT と略します）を使用しています。そのため、RX ファミリ M3S-TFAT-Tiny : FAT ファイルシステムソフトウェア(ドキュメント No. R20AN0038JJ)も併せてご使用ください。

以下に、本モジュールがサポートしている機能を示します。

1. 接続された USB ストレージ機器のデバイス照合（動作可否判定）を行う。
2. BOT プロトコルによるストレージコマンド通信を行う。
3. USB マスストレージサブクラスの SFF-8070i(ATAPI)に対応。
4. データ転送に使用するパイプを、IN/OUT 転送で共有。複数のデバイスを接続した場合も 1 本のパイプを共有。
5. 最大 4 つのストレージ機器を接続することができます。

1.1 必ずお読みください

このドライバを使ってアプリケーションプログラムを作成する場合は、USB Basic Host and Peripheral Driver アプリケーションノート(ドキュメント No.R01AN0512)を参照いただきますようお願いします。このアプリケーションノートは、パッケージ内の"reference_documents"フォルダにあります。

1.2 注意事項

1. このドライバは、USB 通信動作を保証するものではありません。システムに適用される場合は、お客様における動作検証はもとより、多種多様なデバイスに対する接続確認を実施してください。
2. RX62N/RX621/RX63T/RX630 をご使用の場合、必ず"reference_documents"フォルダ下のドキュメント(r01an0513jj0232_usb.pdf, r01an0512jj0232_usb.pdf)をご使用いただきますようお願いします。

1.3 制限事項

本ドライバは以下の制限事項があります。

1. ストレージ機器として認識できない MSC デバイスがあります。
2. 本ドライバは、GetMaxLun コマンド(Mass Storage Class コマンド)に対する応答が"1"以上の MSC デバイスをサポートしておりません。
3. 最大 4 つの USB ストレージ機器を接続することができます。
4. セクタサイズが 512 バイトの USB ストレージ機器と接続可能です。
5. READ_CAPACITY コマンドに応答しないデバイスはセクタサイズを 512 バイトとして動作します。
6. DMA/DTC をサポートしていません。

1.4 用語一覧

APL	:	Application program
BOT	:	Mass storage class Bulk Only Transport
FSL	:	FAT File System Library
HCD	:	Host Control Driver of
HDCD	:	Host Device Class Driver (device driver and USB class driver)

MGR	:	Peripheral device state manager of HCD
MSC	:	Mass Storage Class
RSK	:	Renesas Starter Kits
TFAT	:	Tiny FAT file system software for microcontrollers (M3S-TFAT-Tiny-RX)
	:	USB Basic Host Driver for (non-OS)
USB	:	Universal Serial Bus

2. ソフトウェア構成

HDCD(ホストデバイスクラスドライバ)は HMSDD（ホストマスストレージデバイスドライバ）と HMSCD（USB ホストマスストレージクラスドライバ）の総称です。
Figure 2-1に HMSC のモジュール構成、Table 2-1にモジュール機能概要を示します。

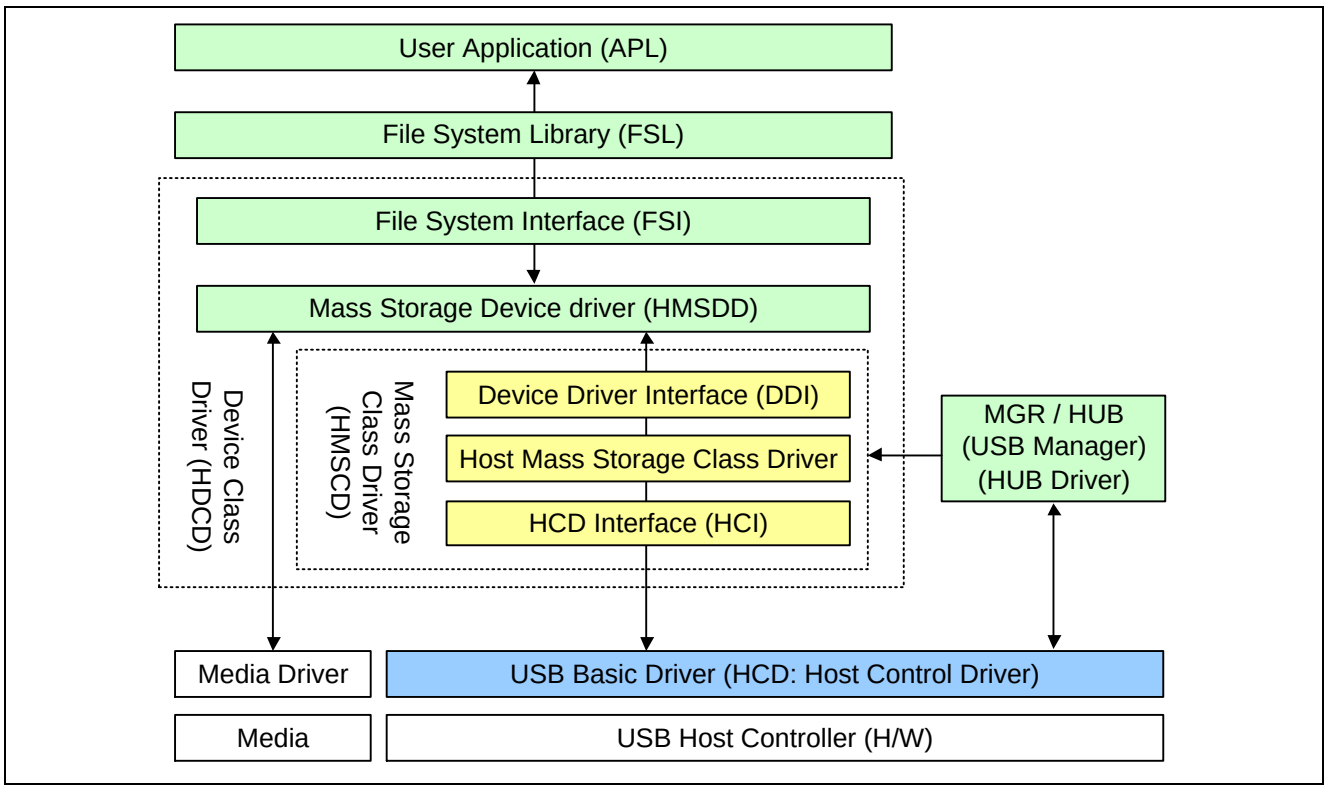


Figure 2-1 ソフトウェア構成図

Table 2-1 モジュール機能概要

モジュール名	機能概要
FSI	FSL－HMSDD 間のインタフェース関数 FSL にあわせて改変してください。
HMSDD	マスストレージデバイスドライバ
DDI	HMSDD－HMSCD 間のインタフェース関数 HMSDD に合わせて改変してください。
HMSCD	マスストレージクラスドライバ ストレージコマンドに BOT プロトコルを付加して HCD へ要求します。また、BOT のシーケンスを管理します。 システム仕様にあわせてストレージコマンドを追加（改変）してください。
HCI	HMSCD－HCD 間のインタフェース関数です。
MGR / HUB	接続されたデバイスとエニュメレーションをして HMSCD を起動します。またデバイスの状態管理も行います。
HCD	USB Host H/W 制御ドライバ

3. API 情報

本ドライバの API はルネサスの API の命名基準に従っています。

3.1 ハードウェアの要求

ご使用になる MCU が以下の機能をサポートしている必要があります。

- USB

3.2 動作確認環境

このドライバの動作確認環境を以下に示します。

Table 3-1 動作確認環境

項目	内容
C コンパイラ	ルネサスエレクトロニクス製 C/C++ Compiler for RX Family V.3.01.00
	コンパイルオプション：統合開発環境のデフォルト設定に以下のオプションを追加 -lang = c99
エンディアン	リトルエンディアン / ビッグエンディアン
使用ボード	Renesas Starter Kits for RX63N

3.3 使用する割り込みベクタ

このドライバが使用する割り込みベクタを以下に示します。

Table 3-2 使用する割り込みベクター一覧

デバイス	割り込みベクタ
RX63N/RX631	USBI0 割り込み(ベクタ番号: 35) / USBR0 割り込み(ベクタ番号: 90)

3.4 ヘッダファイル

すべての API 呼び出しとそれをサポートするインタフェース定義は r_usb_basic_if.h と r_usb_hmsc_if.h に記載されています。

3.5 整数型

このプロジェクトは ANSI C99 を使用しています。これらの型は stdint.h で定義されています。

3.6 コンパイル時の設定

コンパイル時の設定については、USB Basic Host and Peripheral Driver アプリケーションノート(ドキュメント No.R01AN0512)内の「コンフィグレーション」の章を参照してください。

3.7 ROM/RAM サイズ

本ドライバの ROM/RAM サイズを以下に示します。

	引数チェック実施時	引数チェック非実施時

ROM サイズ	40.2K バイト (Note 4)	39.6K バイト (Note 5)
RAM サイズ	27.5K バイト	27.5K バイト

[Note]

1. 上記のサイズには、USB Basic Driver のコードサイズが含まれています。
2. 上記のサイズには、TFAT の ROM/RAM サイズは含まれていません。
3. コンパイラの最適化オプションには、Default オプションが指定されています。
4. 「引数チェック実施時」の ROM サイズは、r_usb_basic_config.h ファイル内の USB_CFG_PARAM_CHECKING 定義に対し USB_CFG_ENABLE を指定した時の値です。
5. 「引数チェック非実施時」の ROM サイズは、r_usb_basic_config.h ファイル内の USB_CFG_PARAM_CHECKING 定義に対し USB_CFG_DISABLE を指定した時の値です。

3.8 引数

API 関数の引数に使用される構造体については、USB Basic Host and Peripheral Driver アプリケーションノート(ドキュメント No.R01AN0512)内の「構造体」の章を参照してください。

4. ターゲットペリフェラルリスト（TPL）

TPL については、USB Basic Host and Peripheral Driver アプリケーションノート(ドキュメント No.R01AN0512)内の「ターゲットペリフェラルリスト(TPL)の設定方法」の章を参照してください。

5. クラスドライバ概要

5.1 クラスリクエスト

本ドライバがサポートしているクラスリクエストを以下に示します。

Table 5-1 クラスリクエスト

リクエスト	説明
GetMaxLun	サポートする最大ユニット数を取得する。
MassStrageReset	プロトコルエラーを解除する。

5.2 ストレージコマンド

本ドライバは、以下のストレージコマンドをサポートしています。

1. TEST_UNIT_READY
2. REQUEST_SENSE
3. MODE_SELECT10
4. MODE_SENSE10
5. PREVENT_ALLOW
6. READ_FORMAT_CAPACITY
7. READ10
8. WRITE10

6. API

Host Mass Storage Class 固有の API を以下に示します。

API	説明
R_USB_HmscStrgCmd()	Host Mass Storage コマンドを発行
R_USB_HmscGetDriveNo()	ドライブ番号を取得

[Note]

1. ストレージメディアに対するアクセスを行う場合は、FAT(File Allocation Table)の API を使用してください。
2. その他の API を使用する場合は USB Basic Host and Peripheral Driver Firmware Integration Technology アプリケーションノート(ドキュメント No.R01AN2025)内の「API」の章を参照してください。

6.1 R_USB_HmscStrgCmd

Mass Storage コマンドを発行する。

形式

usb_err_t R_USB_HmscStrgCmd(usb_ctrl_t *p_ctrl, uint8_t *p_buf, uint16_t command)

引数

p_ctrl	usb_ctrl_t 構造体領域へのポインタ
p_buf	バッファ領域へのポインタ
command	Mass Storage コマンド

戻り値

USB_SUCCESS	正常終了
USB_ERR_PARA	パラメータエラー
USB_ERR_NG	その他のエラー

解説

引数(p_ctrl)のメンバ module とメンバ address によって指定された MSC デバイスに対し、引数 command に指定された Mass Storage コマンドを発行します。アプリケーションプログラムは、R_USB_GetEvent 関数の戻り値(USB_STS_MSC_CMD_COMPLETE)により Mass Storage コマンドの完了を確認することができます。

応答データがある Mass Storage コマンドを発行した場合、アプリケーションプログラムでは、R_USB_GetEvent 関数の戻り値(USB_STS_MSC_CMD_COMPLETE)を確認後、第 2 引数(p_buf)が示す領域から応答データを取得することができます。受信した応答データのサイズは、第 1 引数(p_ctrl)のメンバ size を参照してください。

引数 command には、以下を指定してください。

Table 6-1 MassStorage コマンド

MassStorage コマンド
USB_ATAPI_TEST_UNIT_READY
USB_ATAPI_REQUEST_SENSE
USB_ATAPI_INQUIRY
USB_ATAPI_MODE_SELECT10
USB_ATAPI_PREVENT_ALLOW
USB_ATAPI_READ_FORMAT_CAPACITY
USB_ATAPI_READ_CAPACITY
USB_ATAPI_MODE_SENSE10

リエントラント

本関数は再入不可能（リエントラント不可）です。

補足

1. 本 API をコールする前に usb_ctrl_t 構造体のメンバ module に USB モジュール番号を指定し、メンバ address に対しデバイスアドレスを指定してください。なお、メンバ module に対し USB_IP0/USB_IP1 以外を指定した場合、戻り値に USB_ERR_PARA が返されます。
2. ご使用の MCU が USB モジュールを 1 つしかサポートしていない場合、メンバ module に対し USB_IP1 を指定しないでください。USB_IP1 を指定した場合は、戻り値に USB_ERR_PARA が返されます。
3. 引数 p_ctrl に対し USB_NULL を指定した場合は、戻り値に USB_ERR_PARA が返されます。

4. 引数 p_buf には自動変数(スタック)領域へのポインタは指定しないでください。
5. MSC デバイスからの応答データがないコマンドを発行する場合、引数 p_buf に対し USB_NULL を指定してください。
6. 引数 command に対しTable 6-1の MassStorage コマンド以外のコマンドを指定した場合、戻り値に USB_ERR_PARA が返されます。
7. 本 API による Mass Storage コマンド発行後、FAT API や本 API をコールする場合、必ず、R_USB_GetEvent 関数の戻り値(USB_STS_CMD_COMPLETE)を確認した後でこれらの API をコールしてください。
8. CSW については、「7. **R_USB_GetEvent関数の戻り値 (USB_STS_MSC_CMD_COMPLETE)**」を参照してください。
9. CSW 情報は、usb_ctrl_t 構造体のメンバ status に設定されます。メンバ status の値が USB_CSW_FAIL の場合、本 API を使って"Request Sense"コマンドを MSC デバイスに発行してください。
10. Mode Sense10 コマンドのページコード(1 バイト)は、第 2 引数(p_buf)が示す領域の先頭アドレスに指定してください。
11. Mode Select10 コマンドのパラメータデータは USB マスストレージサブクラス(SFF-8070i など)の規格書をもとに第 2 引数(p_buf)が示す領域に指定してください。
12. USB デバイスが CONFIGURED 状態の場合に、本 API をコールすることができます。CONFIGURED 以外の状態で本 API をコールすると戻り値に USB_ERR_NG が返されます。

使用例

```
uint8_t g_buf[64];
void usb_application( void )
{
    usb_ctrl_t ctrl;
    usb_err_t err;
                    :
    while (1)
    {
        switch (R_USB_GetEvent(&ctrl))
        {
                    :
            case USB_STS_CONFIGURED:
                    :
                g_buf[0] = 0x3F             /* Page Code */
                ctrl.module = USB_IP1;
                ctrl.address = adr;
                R_USB_HmscStrgCmd( &ctrl, &g_buf, USB_ATAPI_MODE_SENSE10 );
                    :
            break;
            case USB_STS_MSC_CMD_COMPLETE:
                if (ctrl.status == USB_CSW_FAIL)
                {
                    R_USB_HmscStrgCmd(&ctrl, &g_buf, USB_ATAPI_REQUEST_SENSE);
                }
                    :
            break;
                    :
        }
    }
}
```

6.2 R_USB_HmscGetDriveNo

ドライブ番号を取得する

形式

usb_err_t R_USB_HmscGetDriveNo(usb_ctrl_t *p_ctrl, uint8_t *p_drive)

引数

p_ctrl	usb_ctrl_t 構造体領域へのポインタ
p_drive	ドライブ番号を格納する領域へのポインタ

戻り値

USB_SUCCESS	正常終了
USB_ERR_PARA	パラメータエラー
USB_ERR_NG	その他のエラー

解説

usb_ctrl_t 構造体に指定された情報(メンバ module およびメンバ address)をもとに関連するドライブ番号を取得します。ドライブ番号は、引数 p_drive が示す領域に格納されます。

リエントラント

本 API は再入可能（リエントラント）です。

補足

1. 本 API をコールする前に usb_ctrl_t 構造体のメンバ address およびメンバ module に対し、ドライブ番号を取得したい MSC デバイスのデバイスアドレスおよびその MSC デバイスが接続された USB モジュール番号(USB_IP0/USB_IP1)を指定してください。なお、これらのメンバに対する指定に問題がある場合は、戻り値に USB_ERR_PARA が返されます。
2. 引数 p_ctrl に対し USB_NULL を指定した場合は、戻り値に USB_ERR_PARA が返されます。
3. USB デバイスが CONFIGURED 状態の場合に、本 API をコールすることができます。CONFIGURED 以外の状態で本 API をコールすると戻り値に USB_ERR_NG が返されます。

使用例

```
void usb_application( void )
{
    usb_ctrl_t ctrl;
    uint8_t     drive;
                    :
    while (1)
    {
        switch (R_USB_GetEvent(&ctrl))
        {
                    :
            case USB_STS_CONFIGURED:
                    :
                ctrl.module = USB_IP0;
                ctrl.address = adr;
                R_USB_HmscGetDriveNo( &ctrl, &drive );
                    :
            break;
                    :
        }
    }
}
```

7. R_USB_GetEvent 関数の戻り値 (USB_STS_MSC_CMD_COMPLETE)

R_USB_HmscStrgCmd 関数による Mass Storage コマンドの完了を認識した後で R_USB_GetEvent 関数をコールすると戻り値 USB_STS_MSC_CMD_COMPLETE が返されます。このほか、usb_ctrl_t 構造体の以下メンバにも情報が設定されます。

module	:	Mass Storage コマンドが完了した USB モジュール番号
address	:	Mass Storage コマンドが完了した USB デバイスのデバイスアドレス
size	:	応答データサイズ
status	:	CSW 情報

[Note]

1. usb_ctrl_t 構造体のメンバ module にはその USB デバイスが接続されている USB モジュール番号 (USB_IP0 / USB_IP1)が設定され、メンバ address には Mass Storage コマンドが完了した USB デバイスのデバイスアドレスが設定されます。
2. メンバ size には MSC デバイスから送信される応答データのデータサイズが設定されます。
3. メンバ status には CSW(Command Status Wrapper)の bCSWStatus が設定されます。

USB_CSW_SUCCESS	(値:00H)	:	成功
USB_CSW_FAIL	(値:01H)	:	失敗
USB_CSW_PHASE	(値:02H)	:	フェーズエラー

8. サンプルアプリケーション

8.1 アプリケーション仕様

APL の主な機能を以下に示します。

1. MSC デバイスに対するエニュメレーション処理およびドライブ認識処理を行います。
2. 上記完了後、ファイル'hmscdemo.txt'を一回、MSC デバイスに対し書き込みます。
3. 上記書き込み後、ファイル'hmscdemo.txt'のリード処理を繰り返します。

[Note]

ファイル'hmscdemo.txt'が格納された MSC デバイスを接続した場合、'hmscdemo.txt'が上書きされます。

8.2 アプリケーション処理概要

APL は、初期設定、メインループの 2 つの部分から構成されます。以下にそれぞれの処理概要を示します。

8.2.1 初期設定

初期設定では、MCU の端子設定、USB ドライバの設定、USB コントローラの初期設定を行います。

8.2.2 メインループ

このメインループでは、R_USB_GetEvent 関数の戻り値とステート変数を使ってプログラム制御を行っています。USB のイベント管理(アタッチ/デタッチ等)を、R_USB_GetEvent 関数の戻り値により行い、MSC デバイスに対するファイルライト/ファイルリードをステート変数の状態参照により行います。

以下に、APL の処理概要を示します。

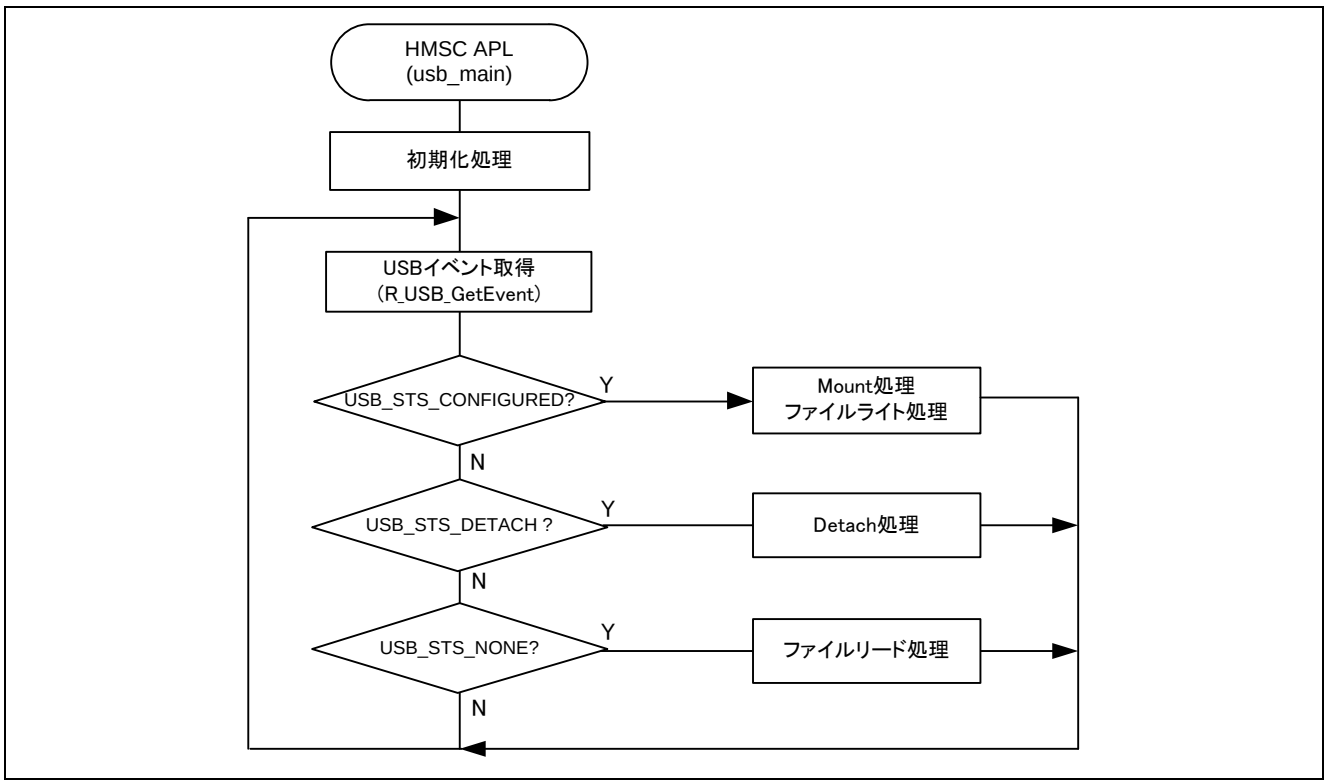


Figure 8-1 メインループ処理

1) Mount 処理 /ファイルライト処理 (USB_STS_CONFIGURED)

RSK に MSC デバイスが ATTACH され、Enumeration およびドライブ認識処理完了後に R_USB_GetEvent 関数をコールすると戻り値に USB_STS_CONFIGURED がセットされます。アプリケーションプログラムでは、戻り値に USB_STS_CONFIGURED がセットされていることを確認すると、Mount 処理およびファイルライト処理を行った後、ステート管理用変数に対し STATE_FILE_READ を設定します。

2) ファイルリード処理 (USB_STS_NONE)

USB 関連のイベントが無い状態で、R_USB_GetEvent 関数をコールすると戻り値に USB_STS_NONE がセットされます。アプリケーションプログラムでは、戻り値に USB_STS_NONE がセットされていることを確認すると以下のファイルリード処理を行います。

a. ファイルリード処理

FAT API を使って MSC デバイスに対しファイルリード処理を行います。MSC デバイスがデタッチされるまで MSC デバイスに対するファイルリード処理が継続されます。

3) Detach 処理 (USB_STS_DETACH)

RSK から MSC デバイスが DETACH された後、R_USB_GetEvent 関数をコールすると戻り値に USB_STS_DETACH がセットされます。アプリケーションプログラムでは、戻り値に USB_STS_DETACH がセットされていることを確認すると、ステート管理用変数に対し STATE_DETACH を設定します。

8.3 アプリケーションプログラム用コンフィグレーションファイル (r_usb_hmsc_apl_config.h)

以下の各定義に対する設定を行ってください。

1. USE_USBIP 定義

使用する USB モジュールのモジュール番号を指定してください。USB_IP0/USB_IP1 のいずれかを指定してください。

```
#define    USE_USBIP    USE_USBIP0                  // USB0 モジュールを使用する場合
#define    USE_USBIP    USE_USBIP1                  // USB1 モジュールを使用する場合
#define    USE_USBIP    (USE_USBIP0|USE_USBIP1)     // USB0 と USB1 モジュールを使用する場合
```

2. 注意事項

上記はアプリケーションプログラム用のコンフィグレーション設定です。上記の設定の他に USB ドライバのコンフィグレーション設定が必要です。USB ドライバのコンフィグレーション設定については、「USB Basic Host and Peripheral Driver アプリーションノート」(Document No. R01AN0512JJ)を参照してください。

8.4 複数の MSC デバイスを接続する場合

USB Hub 等を使用し、複数の MSC デバイスを接続するアプリケーションプログラムを開発する場合は、以下の参考プログラムを参照してください。

r_usb_hmsc_apl_multi.c

9. セットアップ

9.1 ハードウェア

HMSC の動作環境例をFigure 9-1に示します。評価ボードのセットアップ、エミュレータなどの使用方法については各取扱説明書を参照ください。

9.1.1 動作環境例

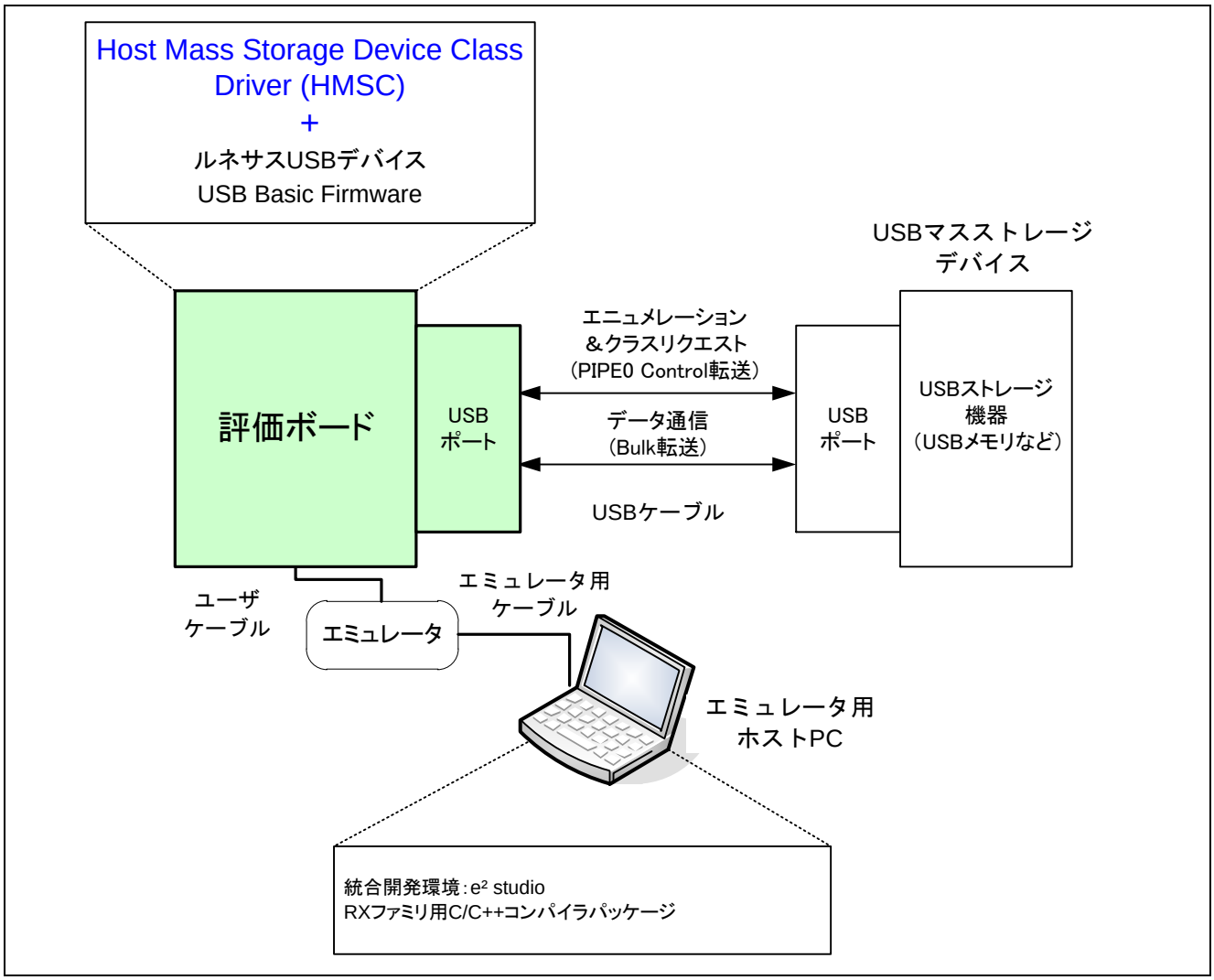


Figure 9-1 動作環境例

9.1.2 RSK 設定

RSK を USB Host モードに設定する必要があります。設定内容は以下を参照してください。

Table 9-1 **RSK 設定**

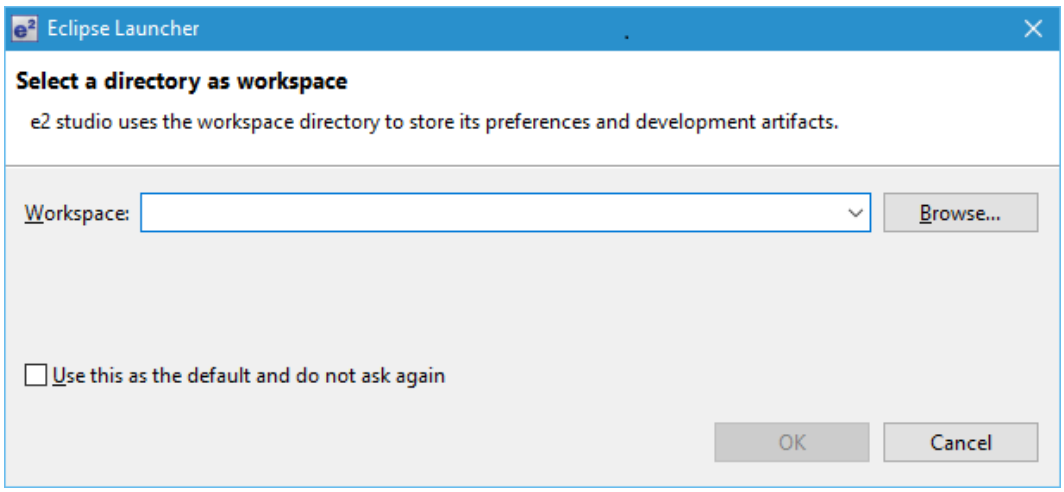
RSK	ジャンパ設定
RSK+RX63N	J3: Shorted Pin 2-3 J4: Shorted Pin 2-3 J18:Shorted Pin1-2

9.2 ソフトウェア

(1). e² studio を起動

a) e² studio を起動してください。

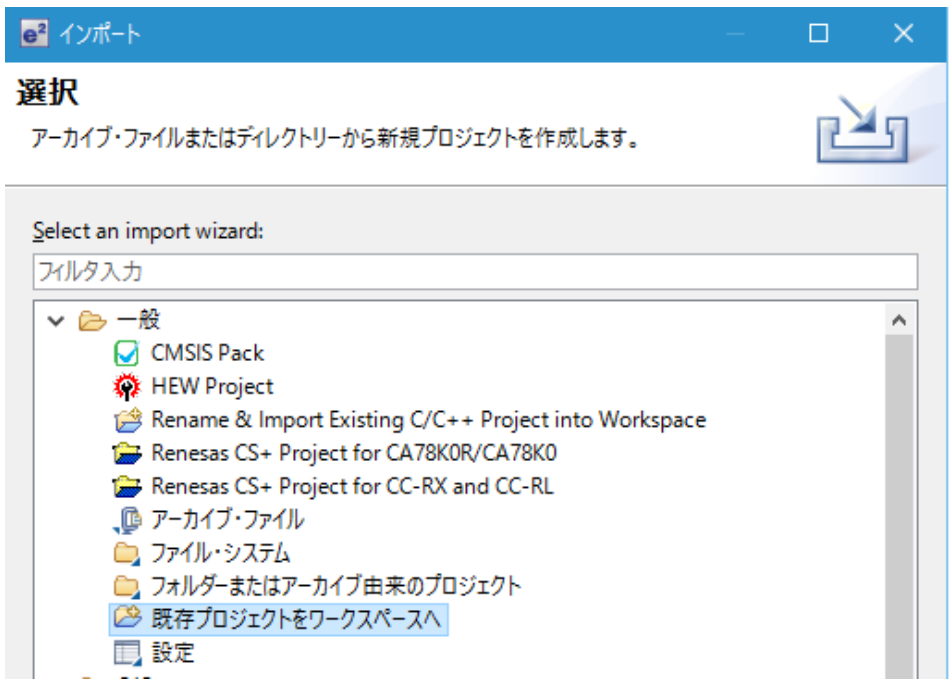
b) はじめて e² studio を起動する場合、Eclipse Launcher ダイアログが表示されますので、プロジェクトを格納するためのフォルダを指定してください。



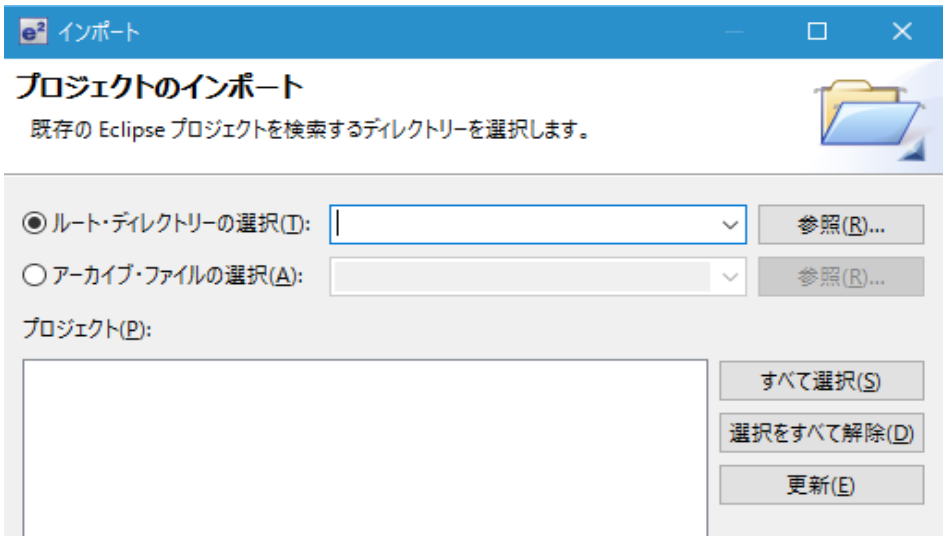
(2). ワークスペースへのプロジェクトの登録

a) [ファイル] --> [インポート]を選択してください。

b) [一般] => [既存プロジェクトをワークスペースへ]を選択してください。



c) プロジェクトファイル".cproject"が格納されたフォルダを”ルート・ディレクトリの選択”に入力してください。



d) “終了”をクリック

プロジェクトのワークスペースへのインポートが完了しました。同様の方法で他のプロジェクトを同一のワークスペースへインポートすることができます。

(3). “Build”ボタンをクリックし、実行プログラムを生成してください。

(4). デバッガへの接続を行い、実行プログラムをダウンロードしてください。“Run”ボタンをクリックすると、プログラムが実行されます。

10. アプリケーションの作成方法

USB Basic Host and Peripheral Driver アプリケーションノート(ドキュメント No.R01AN0512)内の「**アプリケーションプログラムの作成方法**」の章を参照してください。

11. e² studio 用プロジェクトを CS+で使用する場合

HMSC のプロジェクトは、統合環境 e² studio で作成されています。HMSC を CS+で動作させる場合は、下記の手順にて読み込んでください。

[Note]

「プロジェクト変換設定」ウィンドウ内の「変換直前のプロジェクト構成ファイルをまとめてバックアップする」のチェックを外してください。

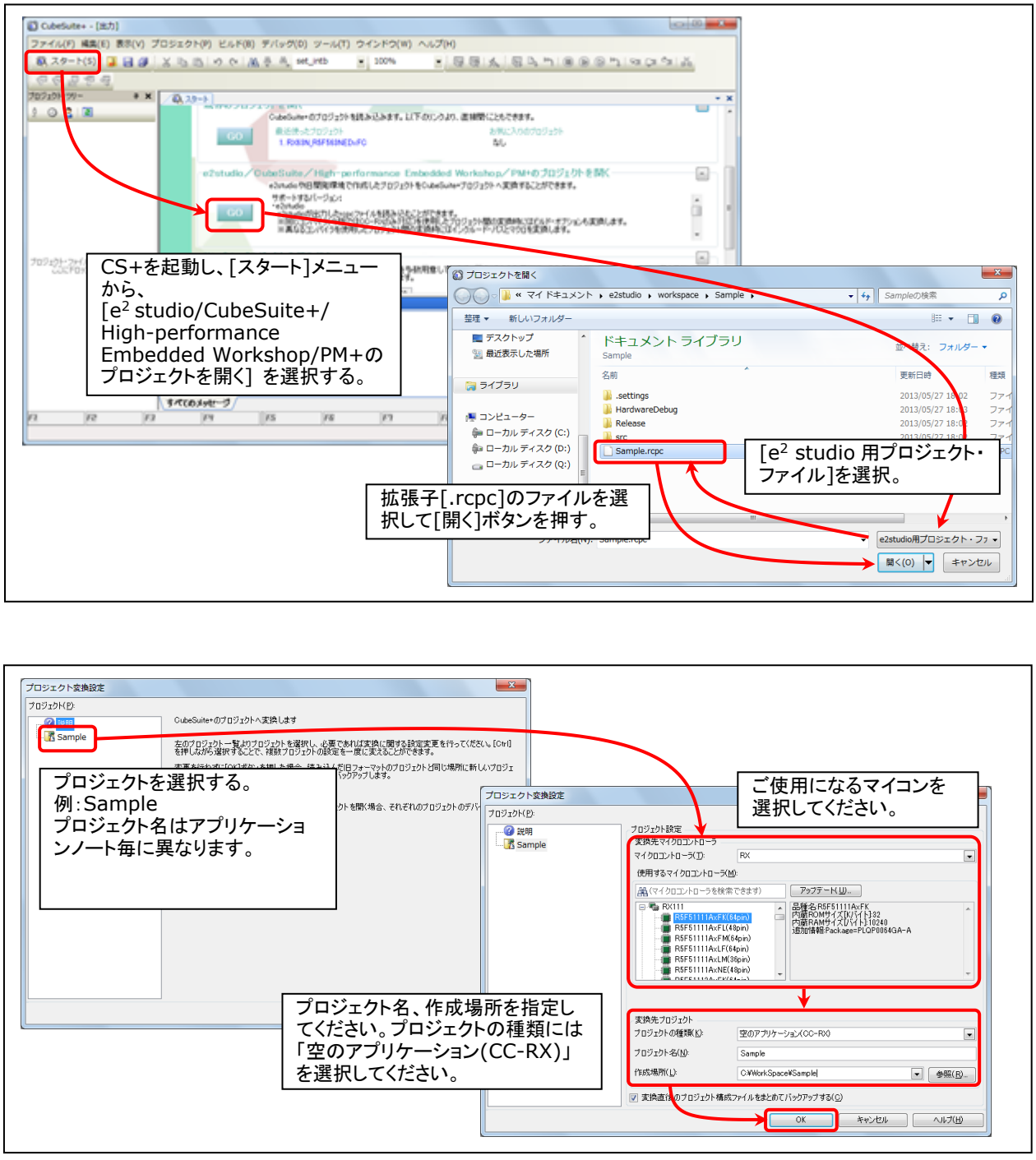


Figure 11-1　e² studio 用プロジェクトの CS+読み込み方法

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/inquiry

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2010.11.29	—	初版発行
1.10	2011.8.10	—	動作確認デバイスに R8A66597 を追加
		—	誤記修正:unit8/16/32_t の unit 表記を uint に修正,
		5	2.4.1 フォルダ構成追加
		8	2.5.2 non-OS 動作時のシステムリソース定義追加
		9	図 2.2 更新
		44-56	8 章、9 章追加
2.00	2012.6.1	—	V.2.00 用 First Release
2.01	2013.2.1	—	“1.4 本書の読み方”追加およびケアレスミス修正
2.10	2013.4.1	—	V.2.10 用 First Release 動作確認デバイスに RX63T を追加。この追加に伴い RX63T に関する内容を追加
2.20	2015.9.30	—	アプリケーションプログラムを変更 フォルダ構成を変更 対象デバイスに RX63N と RX631 を追加。 対象デバイスから R8A66597 を削除 以下の API を追加。 R_usb_hmsc_alloc_drvno, R_usb_hmsc_free_drvno, R_usb_hmsc_ref_drvno 以下の API に対し引数 drvno を追加。 R_usb_hmsc_SetDevSts, R_usb_hmsc_GetDevSts 以下の API に対し引数 ipno を追加。 R_usb_hmsc_Information MSC デバイスの複数接続をサポート。
2.30	Sep 30, 2016	—	1. DMA 転送をサポート 2. USB Host and Peripheral Interface Driver アプリケーションノート(ドキュメント No.R01AN3293JJ)に対応
2.31	Sep 30, 2017	—	1. DMA/DTC 転送を非サポートに変更 2. USB Host and Peripheral Interface Driver アプリケーションノート(ドキュメント No.R01AN3293JJ)の記載内容を本ドキュメントに移行し、USB Host and Peripheral Interface Driver アプリケーションノートを削除した。
2.32	Mar 31, 2018	—	USB Basic Driver のリビジョンを Up しました。
2.33	Jul 31, 2019	—	対象デバイスに RX63N/RX631 を追加

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生じた損害（お客様または第三者いずれに生じた損害も含みます。以下同じです。）に関し、当社は、一切その責任を負いません。
2. 当社製品、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またはこれらに関する紛争について、当社は、何らの保証を行うものではなく、また責任を負うものではありません。
3. 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、全部または一部を問わず、改造、改変、複製、リバースエンジニアリング、その他、不適切に使用しないでください。かかる改造、改変、複製、リバースエンジニアリング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）、交通制御（信号）、大規模通信機器、
   金融端末基幹システム、各種安全制御装置等

   当社製品は、データシート等により高信頼性、Harsh environment向け製品と定義しているものを除き、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（宇宙機器と、海底中継器、原子力制御システム、航空機制御システム、プラント基幹システム、軍事機器等）に使用されることを意図しておらず、これらの用途に使用することは想定していません。たとえ、当社が想定していない用途に当社製品を使用したことにより損害が生じても、当社は一切その責任を負いません。
6. 当社製品をご使用の際は、最新の製品情報（データシート、ユーザーズマニュアル、アプリケーションノート、信頼性ハンドブックに記載の「半導体デバイスの使用上の一般的な注意事項」等）をご確認の上、当社が指定する最大定格、動作電源電圧範囲、放熱特性、実装条件その他指定条件の範囲内でご使用ください。指定条件の範囲を超えて当社製品をご使用された場合の故障、誤動作の不具合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、データシート等において高信頼性、Harsh environment向け製品と定義しているものを除き、耐放射線設計を行っておりません。仮に当社製品の故障または誤動作が生じた場合であっても、人身事故、火災事故その他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。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. 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。当社製品および技術を輸出、販売または移転等する場合は、「外国為替及び外国貿易法」その他日本国および適用される外国の輸出管理関連法規を遵守し、それらの定めるところに従い必要な手続きを行ってください。
10. お客様が当社製品を第三者に転売等される場合には、事前に当該第三者に対して、本ご注意書き記載の諸条件を通知する責任を負うものといたします。
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。
12. 本資料に記載されている内容または当社製品についてご不明な点がございましたら、当社の営業担当者までお問合せください。

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直接的、間接的に支配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。

(Rev.4.0-1 2017.11)

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：https://www.renesas.com/contact/

Colophon 6.0