R01AN0546JJ0215
Rev.2.15
Mar 28, 2016

Renesas USB MCU

USB Peripheral Human Interface Devices Class Driver (PHID) using Basic Mini Firmware

要旨

本資料は、Renesas USB MCUのUSB Basic Mini Firmware を使用したUSB Peripheral Human Interface Devices Class Driver (PHID)のアプリケーションノートです。

動作確認デバイス

RL78/G1C, RL78/L1C, R8C/3MU, R8C/3MK, R8C/34U, R8C/34K

動作確認デバイスと同様の USB モジュールを持つ他の MCU でも本プログラムを使用することができます。このアプリケーションノートのご使用に際しては十分な評価を行ってください。

なお、本プログラムは Renesas Starter Kit 上で動作確認を行っています。

目次

1. はじめに

本アプリケーションノートは、USB-BASIC-FW (1.2章を参照) を使用したUSB Peripheral Human Interface Devices Class Driver (PHID) および、サンプルアプリケーションに関して記述しています。

1.1 機能と特長

USB Peripheral Human Interface Device class driver は、USB Human Interface Device class 仕様(以降HIDと記述) に準拠し、HIDホストとの通信をおこないます。

本クラスドライバは弊社の提供する USB Basic Mini Firmware と組み合わせて使用することを前提にしています。

1.2 関連ドキュメント

1.Universal Serial Bus Revision 2.0 specification
2.USB Device Class Definition for Human Interface Devices (HID) 1.11
3.USB HID Usage Tables Version 1.12
[http://www.usb.org/developers/docs/]
4.Renesas USB MCU ユーザーズマニュアル ハードウェア編
5.Renesas USB MCU USB Basic Mini Firmware アプリケーションノート
ルネサス エレクトロニクスホームページ より入手できます。

・ 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/usb/】

1.3 用語と略語

本書で使用される用語と略語は以下のとおりです。

API	:	Application Program Interface
APL	:	Application program
cstd	:	USB-BASIC-FWの Peripheral & Host共通関数のprefix
HID	:	Human Interface Device class
HID Host	:	HID class USB Host
HM	:	Hardware Manual
KBD	:	Keyboard device
MSE	:	Mouse device
PCD	:	Peripheral control driver of USB-BASIC-F/W
PDCD	:	Peripheral device class driver (device driver and USB class driver)
PHID	:	Peripheral Human Interface Devices
PP	:	プリプロセス定義
pstd	:	Prefix of function and file for Peripheral USB-BASIC-F/W
RSK	:	Renesas Starter Kit
SW1/SW2/SW3	:	User switches on the RSK
USB	:	Universal Serial Bus
USB-BASIC-FW	:	USB-BASIC-FW (Peripheral & Host USB Basic Mini Firmware(USB low level) for Renesas USB MCU)
タスク	:	処理の単位
スケジューラ	:	タスク動作を簡易的にスケジューリングするもの
スケジューラマクロ	:	スケジューラ機能を呼び出すために使用されるもの

1.4 本書の読み方

本書は章の順番通りに読み進める必要はありません。はじめにサンプルプログラムの内容を確認し、ユーザ個別のソリューションに必要な関数およびインタフェースの情報をお読みください。

4.3 章にソース一覧を掲載しています。MCU 固有ソースは、"*devicename*\src\HwResource"にあります。アプリケーションに必要なファイルを確認してください。

ユーザ独自のソリューションを作成するためにはアプリケーションの変更が必要です。5 章はペリフェラル HID アプリケーションの動作を説明しています。

すべてのコードモジュールはタスクに分割されます。タスク間でメッセージの受け渡しが行われていることを予めご理解ください。関数(タスク)の実行順序はスケジューラが決定します。このため重要なタスクに優先権を持たせることができます。また、タスクに登録されたコールバックメカニズムを使用することで、各タスクは並列処理（ノンブロッキング）で動作します。タスクのメカニズムは 1.2 章の"USB-BASIC-FW Application Note"で説明しています。PHID のタスクについては 4.4 章を参照してください。

2. デバイスクラスドライバの登録方法

ユーザが作成したクラスドライバは、USB-BASIC-FW に登録することで USB デバイスクラスドライバとして機能します。*r_usb_phid_apl.c* ファイル内の *usb_phid_driver_registration ()* 関数を参考に USB-BASIC-FW にクラスドライバを登録してください。詳細は、USB-BASIC-FW のアプリケーションノートを参照してください。

3. 動作確認済環境

3.1 コンパイラ

動作確認を行ったコンパイラは以下の通りです。

a. CA78K0R コンパイラ　V.1.71
b. CC-RL コンパイラ V.1.01
c. IAR C/C++ Compiler for RL78 version 2.10.4
d. KPIT GNURL78-ELF v15.02
e. C/C++ Compiler Package for M16C Series and R8C Family V.6.00 Release 00

3.2 評価ボード

動作確認を行った評価ボードは以下の通りです。

a. Renesas Starter Kit for RL78/G1C (型名: R0K5010JGC001BR)
b. Renesas Starter Kit for RL78/L1C (型名: R0K50110PC010BR)
c. R8C/34K Group USB Peripheral 評価ボード(型名: R0K5R8C34DK2PBR)

4. ソフトウェア構成

4.1 モジュール構成

PHID は HID クラスドライバと、マウス、キーボードのデバイスドライバから構成されます。

Figure 4.1 に PHID のソフトウェアモジュール構成図を、Table 4-1 にモジュール説明を示します。

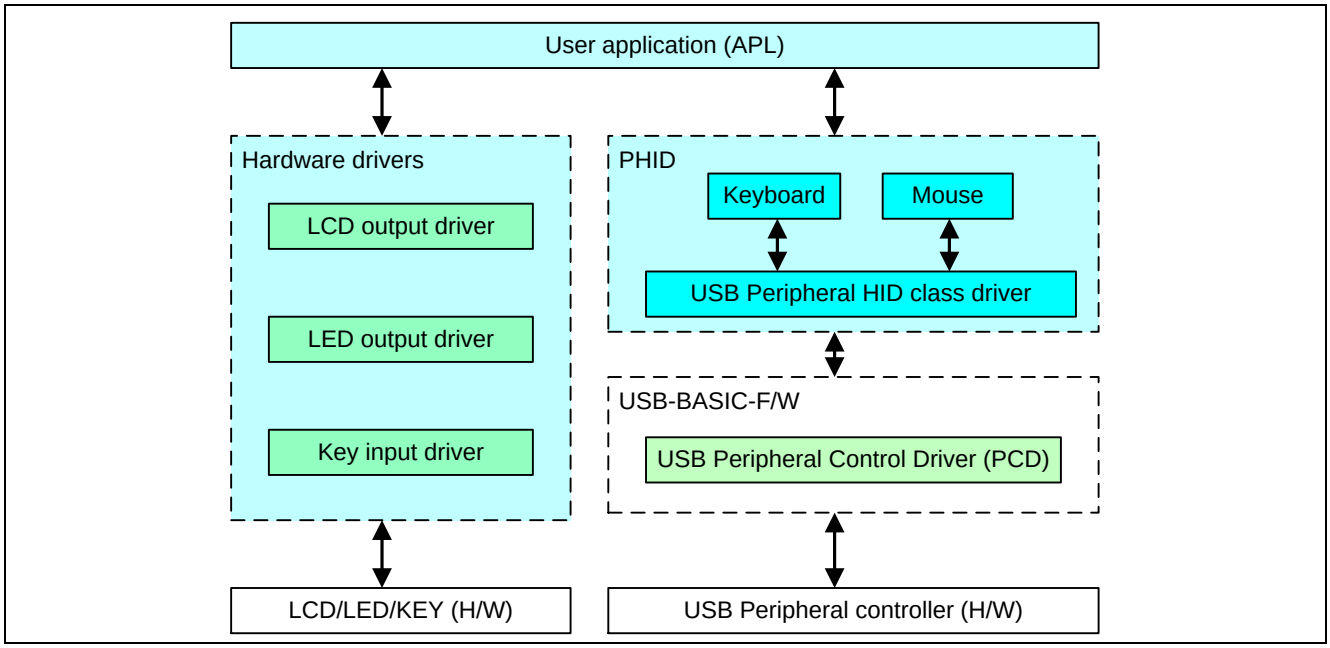


Figure 4.1 モジュール構成図

Table 4-1 モジュール機能説明

モジュール名	機能概要
APL	ユーザアプリケーションプログラムです。 スイッチ 操作により HID ホストと通信します。リモートウェイクアップ制御を行います。 デバイスステートを LCD に表示します。
PHID	RSK ボード上のスイッチ操作（APL の要求）を HID レポートに変換します。 USB-BASICF/W に以下のデータ転送要求を行います。 1) マウス動作用の HID レポート 2) キーボード動作用の HID レポート 転送結果はコールバック関数によって APL に通知します。 また HID ホストから通知された出力レポートを APL に伝えます。 1) キーボード制御用の LED データ
USB-Basic-FW	USB ペリフェラルコントロールドライバです。 (ハードウェア制御とデバイスステート管理を行います。)

4.2 アプリケーションプログラム機能概要

ペリフェラルデモ APL の主な機能は以下のとおりです。

1. HID クラスドライバ
2. デバイスドライバ(マウス、キーボード)

以下のデモオペレーションモードが、 *\devicename\src\PHID\inc\usb_class_usrcfg.h* に定義されています。

1. Mouse
2. Keyboard

デモサンプルアプリケーションは、ハードウェアドライバを介してスイッチ入力を行い、スイッチ入力に対応したレポートデータを作成し、PHID にレポートデータを渡します。

PHID はアプリケーションから受け取ったレポートデータを、PCD を介して HID ホストに通知します。

また HID ホストから要求があれば、PCD を介してアプリケーションに通知します。

PHID サンプル FW では、*\devicename\src\PHID\inc\usb_class_usrcfg.h* ファイルを編集することによりマウス、キーボードの何れかの動作モードで動作します。

4.3 ファイル構成

4.3.1 フォルダの構成

以下に本デバイスクラスで提供するファイルのフォルダ構成を示します。

各 MCU と評価基板に依存するソースコードはそれぞれのハードウェアリソースフォルダ (*\devicename\src\HwResource*) にあります。

```
workspace
    ＋ [ RL78 / R8C ]
      ＋ [ CCRL / CS+ / IAR / e2 studio / HEW ]
        ＋[ RL78G1C / RL78L1C / R8C3MK / R8C3MU / R8C34K / R8C34U ]
            ＋ PHID_KBD                                  Keyboard ビルド結果
            ＋ PHID_MSE                                  Mouse ビルド結果
            ＋ src
              ＋――― PHID [ Human Intergace Device Class driver ]      Table 4-2 参照
              │       ＋――― inc                         HID ドライバ共通ヘッダファイル
              │       ＋――― src                         HID ドライバ
              ＋―――SmplMain [サンプルアプリケーション]
              │       ＋――― APL                         サンプルアプリケーション
              ＋―――USBSTDFW [全ての USB ドライバに共通な基本ファームウェア]
              │       ＋――― inc                         USB ドライバ共通ヘッダファイル
              │       ＋――― src                         USB ドライバ
              ＋――― HwResource [ MCU 初期化等のハードウェアアクセス層 ]
                        ＋――― inc                       H/W リソースヘッダファイル
                        ＋―――src                       H/W リソース
```

[Note]

a. CS+フォルダ下には、CA78K0R コンパイラ用のプロジェクトが格納されています。

b. e² studio フォルダ下には、KPIT GNU コンパイラ用のプロジェクトが格納されています。

c. CS+上で CC-RL コンパイラをご使用になる場合は、「12 e2 studio 用プロジェクトを CS+で使用する場合」を参照してください。

4.3.2 ファイル一覧

Table 4-2 に、PHID が提供するファイル、フォルダ名を示します。

Table 4-2 ファイル構成

フォルダ名	ファイル名	説明	備考
PHID/inc	r_usb_phid_api.h	PHID API 関数定義	
PHID/inc	r_usb_phid_define.h	PHID クラス型定義　PHID マクロ定義	
PHID/inc	r_usb_phid_usrcfg.h	PHID ユーザー定義ファイル	動作モード選択
PHID/src	r_usb_phid_api.c	PHID API 関数	
PHID/src	r_usb_phid_driver.c	PHID ドライバ	
SmplMain	main.c	メインループ処理	
SmplMain/APL	r_usb_phid_apl.c	サンプルアプリケーションプログラム	
SmplMain/APL	r_usb_phid_descriptor.c	PHID クラスディスクリプタテーブル	

4.4 システムリソース

PHID をスケジューラに登録して使用するためのタスク ID とタスク優先度定義を Table 4-3 に示します。

これらについては、 *r_usb_ckernelid.h* で定義します。

Table 4-3 スケジューラ登録 ID 一覧

スケジューラ登録タスク	説明	備考
USB_PHID_TSK	**PHID** (R_usb_phid_task) Task ID: USB_PHID_TSK Task priority: 1	
USB_PCD_TSK	**PCD** (R_usb_pstd_PcdTask) Task ID: USB_PCD_TSK Task priority: 0	
USB_PHIDSMP_TSK	**APL** (usb_phid_main_task) Task ID: USB_PHIDSMP_TSK Task priority: 2	
メールボックス ID / デフォルト受信タスク	**メッセージ名称**	**備考**
USB_PHID_MBX / USB_PHID_TSK	PHID -> PHID / APL -> PHID mailbox ID	
USB_PCD_MBX / USB_PCD_TSK	PCD task mailbox ID	
USB_PHIDSMP_MBX / USB_PHIDSMP_TSK	PHID APL task mailbox ID	

5. 動作環境

ペリフェラル HID サンプルアプリケーション (APL)が動作する環境を以下に示します。

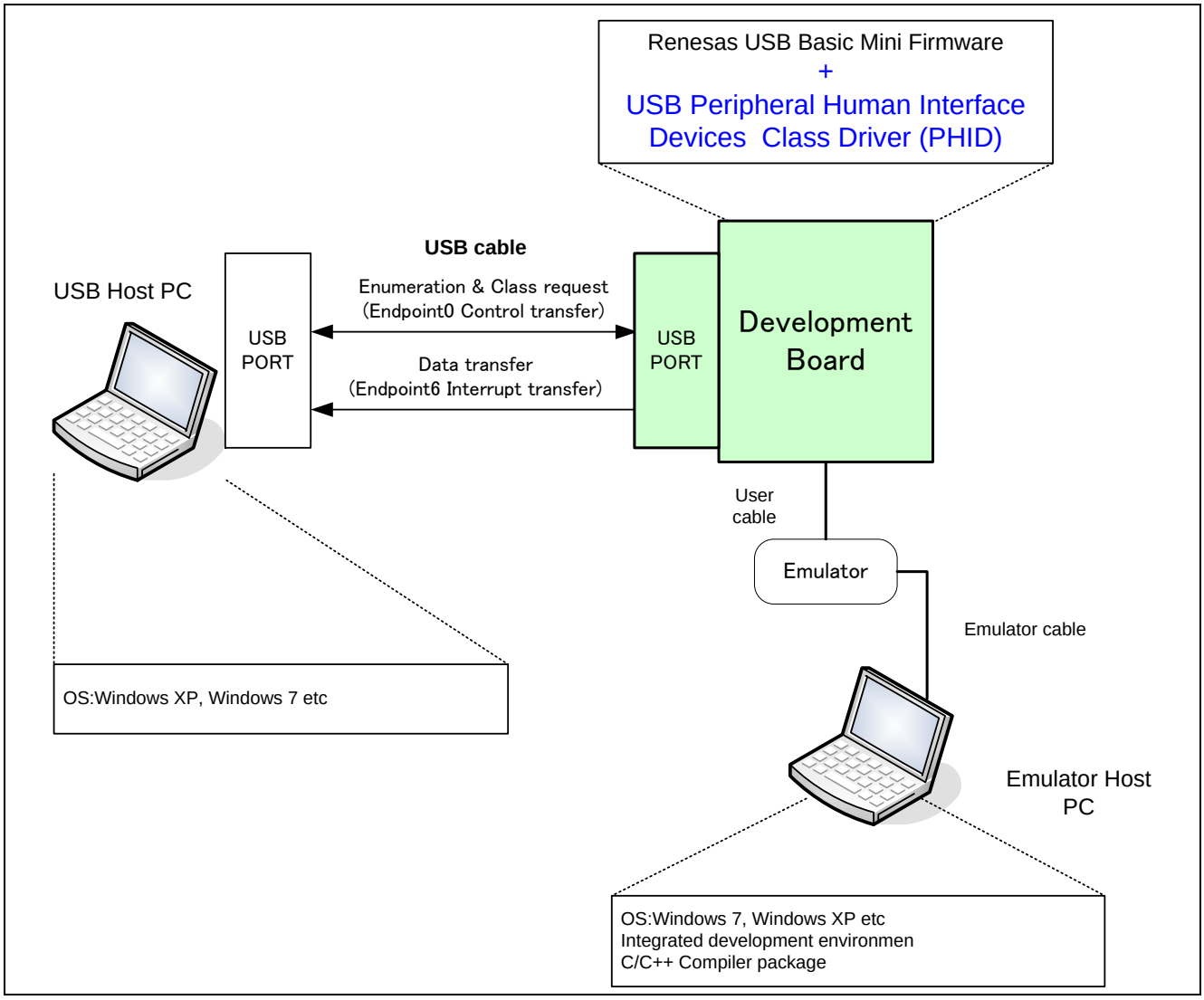


Figure 5-1 動作環境例

6. ペリフェラル HID サンプルアプリケーションプログラム(APL)

6.1 デモ概要

ペリフェラル HID サンプルアプリケーション (APL) は、3 つの動作モードを持っています。

(a) マウスモード

USB マウスとして動作します。ボードのスイッチが押されたとき、ターゲットはマウスの HID レポートを送ります。

(b) キーボードモード

USB キーボードとして動作します。ボードのスイッチが押されたとき、ターゲットはキーボードの HID レポートを送ります。

6.2 HID 動作モード選択について

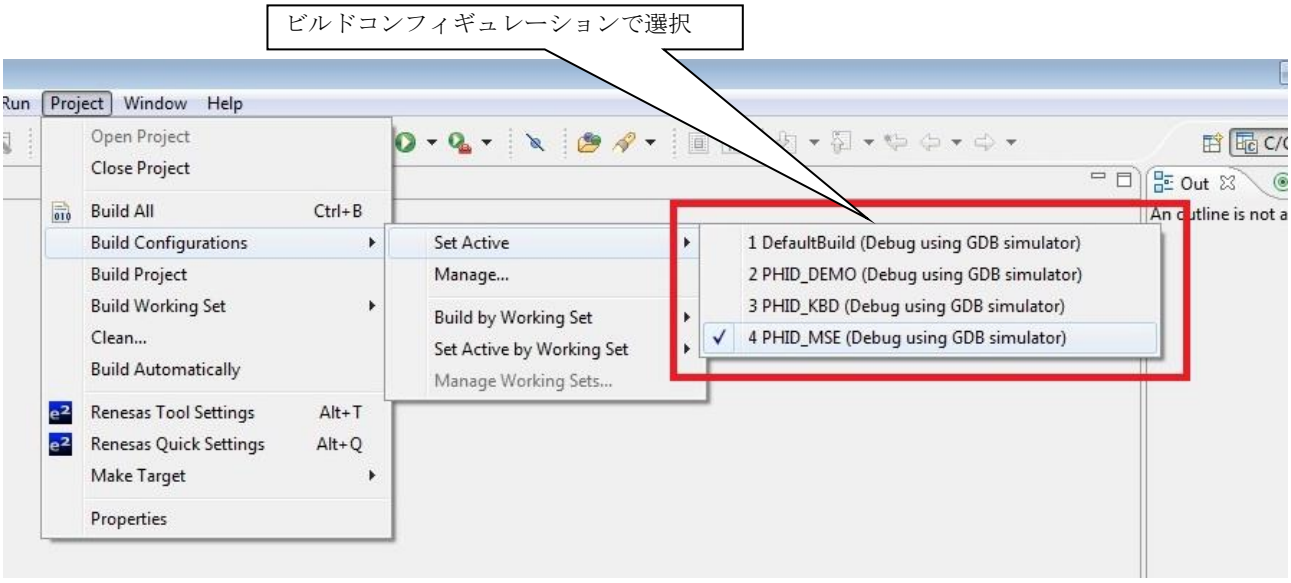
MCU ごとにサポートしている開発環境を立ち上げた後、開発環境上で HID 動作モードを選択します。

キーボードモード : PHID_KBD

マウスモード : PHID_MSE

1) CS+の場合

2) e² studio の場合



6.3 ベンダ ID／プロダクト ID について

ペリフェラルデバイスのベンダ ID(VID) とプロダクト ID(PID)の値は、それぞれ *r_usb_phid_usrcfg.h* の USB_VENDORID と USB_PRODUCTID で設定されています。VID のデフォルト値は 0x0000 で設定されています。また P ID のデフォルト値は、HID デモモードでは 0x0003、キーボードモードでは 0x0013、そしてマウスモードでは 0x0023 に設定されています。

6.4 アプリケーションプログラム関数概要

アプリケーションの主な動作は以下の通りです。

1). キーボードモードの場合は、押下された SW をキーコードに変換し INPUT レポートとして HID ホストに通知します。

2). マウスモードの場合は、押下された SW を X/Y 移動データ、マウスボタンクリックデータに変換し INPUT レポートとして HID ホストに通知します。

6.4.1 アプリケーションプログラム説明

アプリケーションプログラムの動作説明を以下に示します。

(1) マウスモード

マウスモードで RSK の SW1、SW2、SW3 が押されたときの動作を Table 6-1 に示します。

Table 6-1 マウスモード操作

スイッチ番号	動作内容
SW2	上下方向の移動データを通知します。 スイッチを押し続ける時間に応じてマウスカーソルが速度を変えながら移動し、スイッチを離すと停止します。 即座にボタンを再押下すると、移動方向は逆になります。 上方向 スイッチ押下時間 短 : 移動量 1(最小:1～最大:127) 中 : 移動量 2 長 : 移動量 4 下方向 スイッチ押下時間 短 : 移動量 -1 (最小:-1～最大:-127) 中 : 移動量 -2 長 : 移動量 -4
SW3	左右方向の移動データを通知します。 スイッチを押し続ける時間に応じてマウスカーソルが速度を変えながら移動し、スイッチを離すと停止します。 即座にボタンを再押下すると、移動方向は逆になります。 右方向 スイッチ押下時間 短 : 移動量 1(最小:1～最大:127) 中 : 移動量 2 長 : 移動量 4 左方向 スイッチ押下時間 短 : 移動量 -1 (最小:-1～最大:-127) 中 : 移動量 -2 長 : 移動量 -4
SW1	左クリック押下状態を通知します。

HID レポートデータは、Interrupt In 転送を用いて USB ホストへ転送されます。

Table 6-2 にマウス入力レポートフォーマットを示します。

Table 6-2　マウス INPUT レポートフォーマット

offset(Byte)	Value
0	マウスボタンクリック情報 b0：Button 1（左クリック） b1：Button 2（右クリック：非使用） b2-b7：Reserved
1	左右方向の移動量（X displacement：-127～127）
2	上下方向の移動量（Y displacement：-127～127）

Notes:

データ通信で用いるレポートフォーマットはレポートディスクリプタに従う必要があります。

HID クラス仕様を参照し、お客様のシステムに合わせて修正してください。

(2)　キーボードモード

キーボードモードで RSK の SW2 および SW3 を押下された時の動作を Table 6-3 に示します。

Table 6-3 キーボードモード時のスイッチ入力動作

スイッチ番号	動作内容
SW2	SW が押すたびに、”a”-”z”, ”Enter”のキーコードを一つホストへ通知します。
SW3	SW が押すたびに、”1”-”9”, ”0”, ”Enter”のキーコードを一つホストへ通知します。

SW2 及び SW3 はそれぞれ連続して押下している間は次のキーコードに更新しますが、もう一方の SW を押下することによりキーコードをリセットします。

スイッチ押下例：

テキストエディタなどでスイッチ入力操作をした場合の例を以下に示します。

1. SW2 押下 30 回繰り返し -> "a,b,c,d....x,y,z,<Enter>,a,b,c"入力
2. SW3 押下 14 回繰り返し -> "1,2,3,4....8,9,0,<Enter>,1,2,3"入力
3. SW2 押下 3 回繰り返し -> "a,b,c"入力
4. SW3 押下 3 回繰り返し -> "1,2,3" 入力

上記 1～4 の動作を行った後のテキストエディタの入力結果は、以下のようになります。

```
abcdefghijklmnopqrstuvwxyz
abc1234567890
123abc123
```

INPUT レポートデータは、InterruptIn 転送を用いて USB ホストへ転送されます。

Table 6-4 にキーボード INPUT レポートフォーマットを示します。

Table 6-4 キーボード INPUT レポートフォーマット

offset（Byte)	値
0	Modifier keys ： 未使用(=0x00) Bit0:Left Control Bit1:Left Shift Bit2:Left Alt Bit3:Left GUI Bit4:Right Control Bit5:Right Shift Bit6:Right ALT Bit7:Right GUI
1	Reserved ：0x00
2	Keycode 1 ：スイッチ入力に対応するキーコードを設定します。
3	Keycode 2 ： 未使用(=0x00)
4	Keycode 3 ： 未使用 (=0x00)
5	Keycode 4 ： 未使用 (=0x00)
6	Keycode 5 ： 未使用 (=0x00)
7	Keycode 6 ： 未使用 (=0x00)

キーコードは関連ドキュメント”USB HID Usage Tables Version 1.12”で定義されておりますので、そちらを御参照ください。本ソフトウェアで使用しているキーコードを Table 6-5 に示します。

Table 6-5　キーコード

Usage Name	a	b	c	d	e	f	g	h	i	j	k	l	m
Usage ID(HEX)	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10
Usage Name	n	o	p	q	r	s	t	u	v	w	x	y	z
Usage ID(HEX)	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D
Usage Name	1	2	3	4	5	6	7	8	9	0	Enter		
Usage ID(HEX)	1E	1F	20	21	22	23	24	25	26	27	28		

(Usage Name：キー名称，　Usage　ID：　キーコード)

キーボードモードでは HID クラスリクエスト SetReport で転送される OUTPUT レポートによりキーボード LED(RSK LED)制御を行います。

キーボード OUTPUT レポートフォーマットを Table 6-6 に示します。

Table 6-6　キーボード OUTPUT レポートフォーマット

offset（Byte）	値
0	b0：LED 0(NumLock) b1：LED 1(CapsLock) b2：LED 2(ScrollLock) b3：---- b4：---- 0:消灯，1:点灯

Notes:

データ通信で用いるレポートフォーマットはレポートディスクリプタに従う必要があります。HID クラス仕様を参照し、お客様のシステムに合わせて修正してください。

6.5 APL 関数一覧

サンプルアプリケーションの関数一覧を Table 6-7 に示します。

Table 6-7 サンプルアプリケーション関数一覧

	関数名	説明
1	usb_phid_driver_registration	PHID ドライバ登録関数
2	usb_phid_apl_open	USB ペリフェラル HID クラスオープン関数
3	usb_phid_apl_close	USB ペリフェラル HID クラスクローズ関数
4	usb_phid_main_task	ペリフェラル HID サンプルアプリケーションタスク関数
5	usb_phid_apl_trans_cb	HID ホストへの送信完了コールバック関数
6	usb_phid_apl_init	アプリケーション初期化関数
7	usb_phid_apl_clear	アプリケーション初期化関数（デタッチ用）
8	usb_phid_apl_keyboard_led_control	キーボードモード用 SetReport 処理関数
9	usb_phid_apl_kbd_data_set	キーボードモード用キーコードデータ生成処理関数
10	usb_phid_demo_mode_report	HID デモモード用 SetReport 処理関数
11	usb_phid_apl_create_report	INPUT レポートデータ生成処理関数
12	usb_phid_apl_mse_data_set	マウスデータ生成処理関数
13	usb_phid_apl_key_on_wakeup	リモートウェイクアップ用関数
14	usb_phid_smpl_message_send	デモサンプルアプリケーション用メッセージ送信メールボックス処理関数
15	usb_phid_disconnected	USB ケーブル接続状態取得関数
16	usb_psmpl_main_init	アプリケーションプログラムの初期化
17	usb_phid_change_device_state	デバイスステート変更のコールバック
18	usb_phid_apl_DummyFunction	ダミー関数
19	usb_phid_apl_memcpy	メモリのコピー
20	usb_phid_apl_memset	メモリのセット
21	usb_phid_apl_AdData	AD データの読み込み

6.6 シーケンスチャート APL-PHID-PCD

サンプルアプリケーションプログラムのシーケンスを以下に示します。

6.6.1 起動、PHID エニュメレーションのためのセットアップ

Figure 6-1 に、ハードウェアリセットから PHID ドライバへのシーケンスを示します。

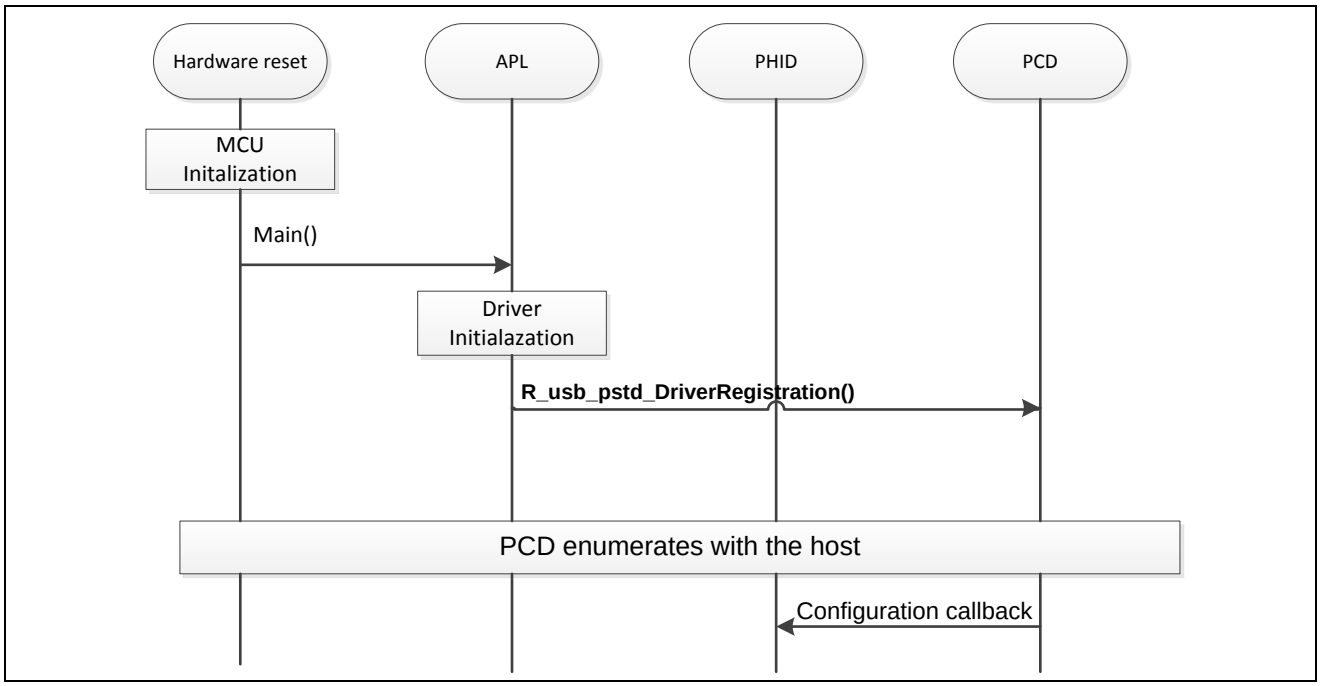


Figure 6-1 起動から PHID エニュメレーションまでのシーケンス

6.6.2 データ通信

Figure 6-2 に、アプリケーションからユーザキー操作・その他イベントが送信される際のデータ通信シーケンスを示します。

レポート転送には、*R_usb_phid_send_data*()が 使用されます。

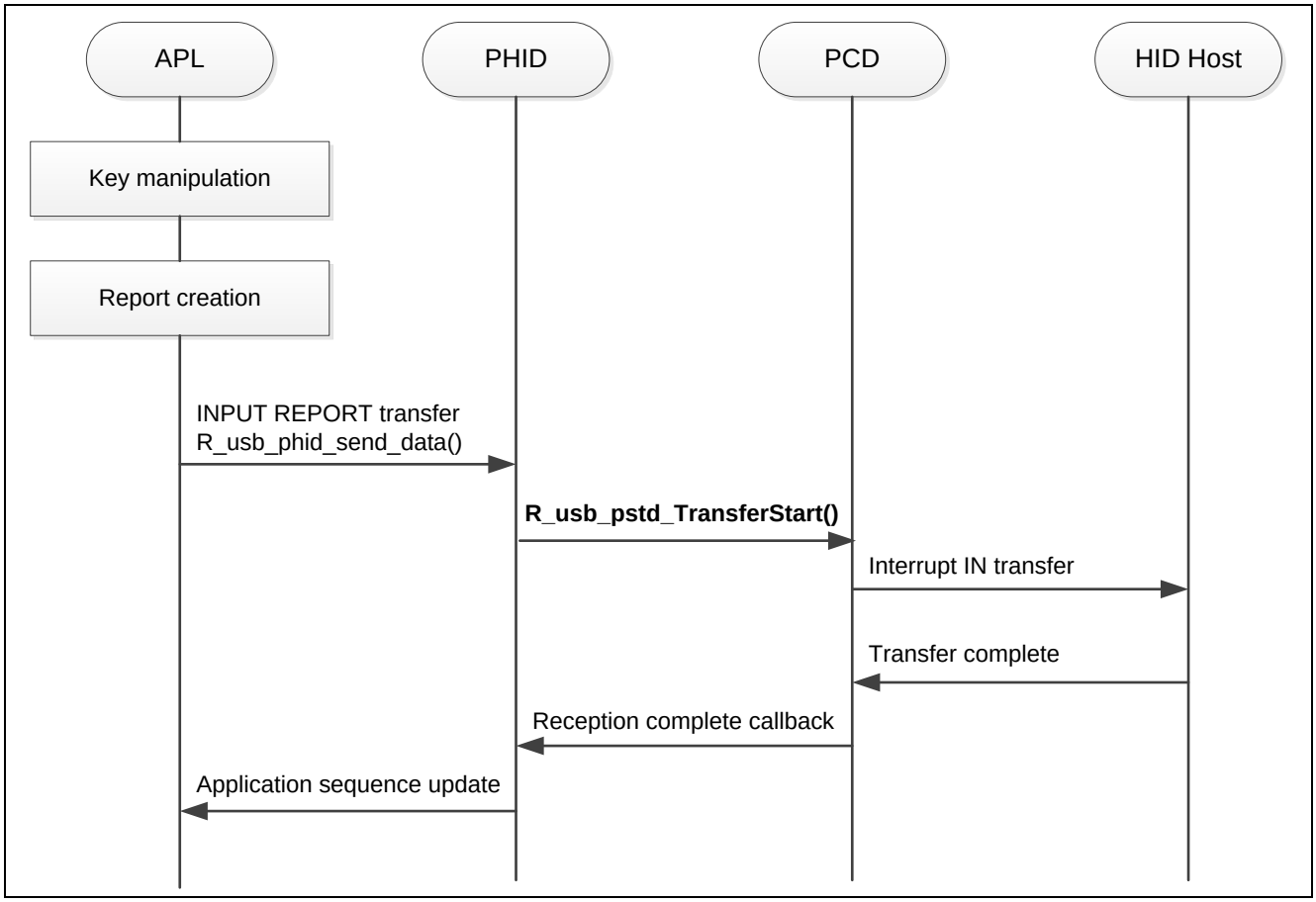


Figure 6-2 リポートデータ送信シーケンス

6.6.3 コントロールリード転送

Figure 6-3 に、HID クラスリクエストの GetReport、GetIdle 、GetProtocol (非サポート)が発生した際のコントロールリード転送シーケンスを示します。

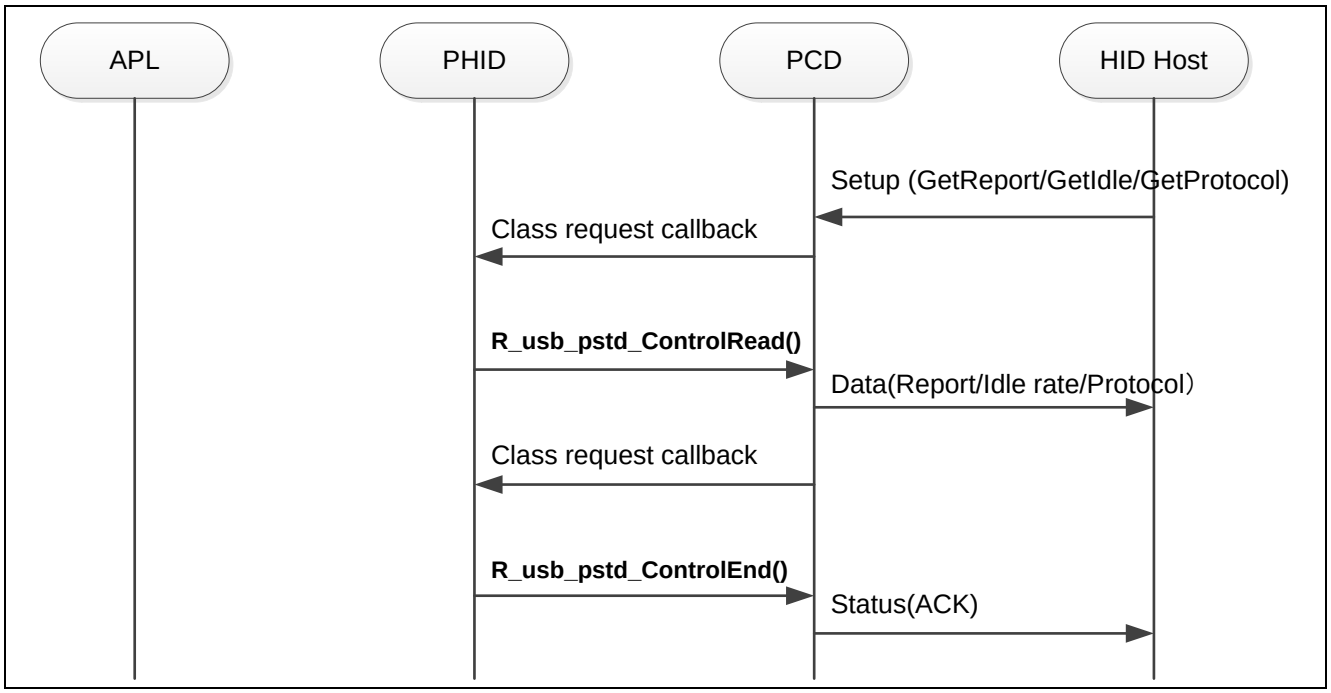


Figure 6-3 コントロールリード転送シーケンス

6.6.4 コントロールライト転送

Figure 6-4 に、HID クラスリクエストの SetReport 発生時の、コントロールライト転送シーケンスを示します。

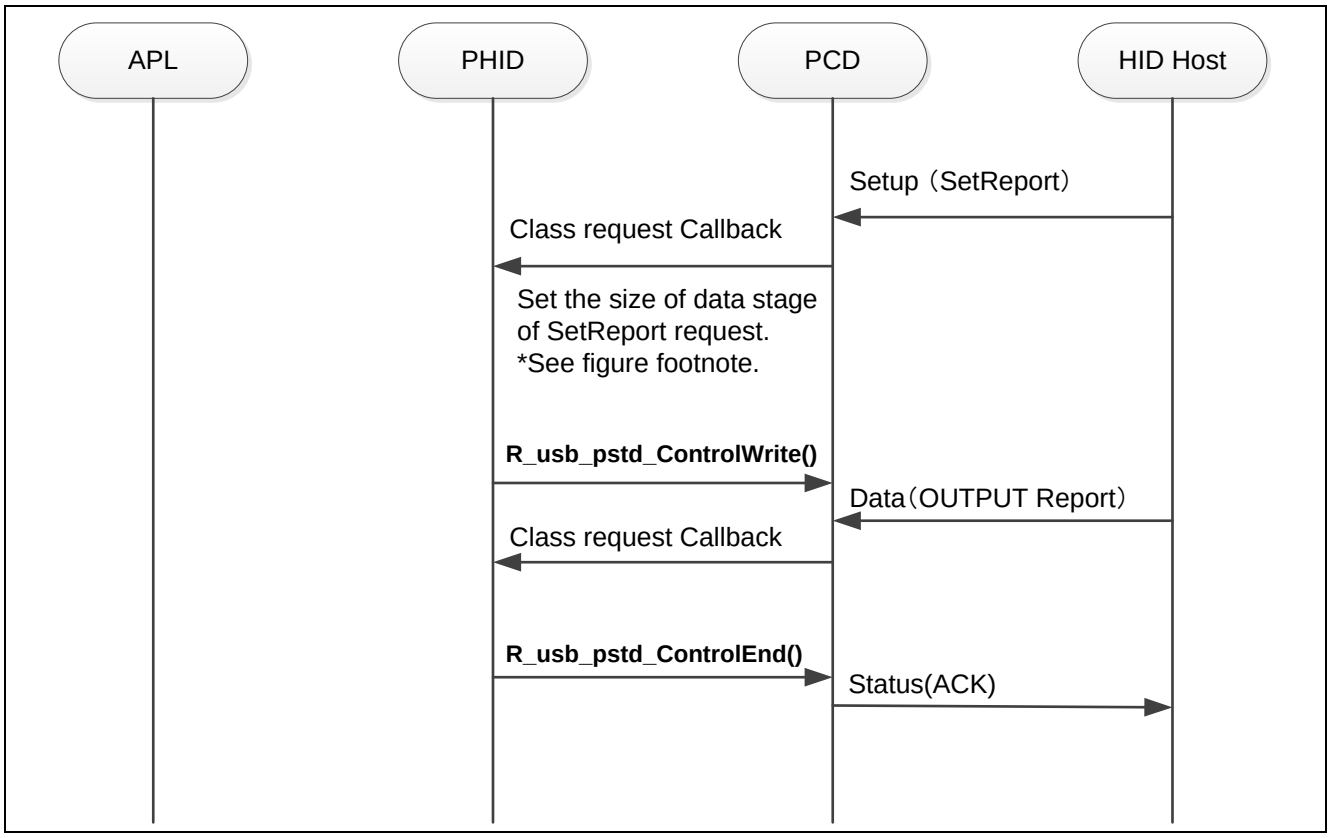


Figure 6-4　コントロールライト転送シーケンス

[Note]

SetReport リクエストのデータステージのサイズは、動作モードで異なります。

以下に各動作モードのサイズを示します。

マウスモード　：　SetReport リクエストで受信無し
キーボードモード　：　キーボード LED コントロールリクエスト(1Byte)
（“Table 6-6　キーボード”を参照してください）

6.6.5 データ無しコントロール転送

Figure 6-5 にデータ無しコントロール転送シーケンスを示します。

SetIdle と SetProtocol (非サポート)は、 データ無しのコントロール転送 HID クラスリクエストです。

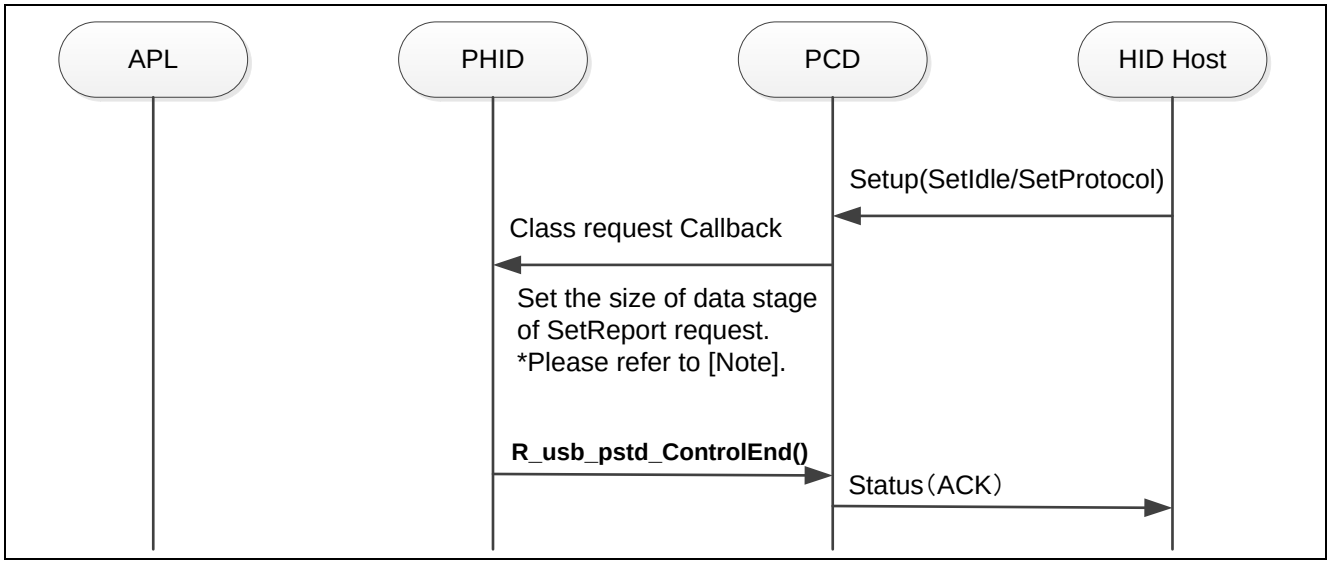


Figure 6-5　データ無しコントロール転送シーケンス

7. Human Interface Device Class (HID)

7.1 基本関数

本ソフトウェアは Human Interface Device Class(HID)仕様に準拠します。PHID の主な機能は、以下のとおりです。

(1) USB ホストからの機能照会に対する応答
(2) USB ホストからのクラスリクエストに対する応答
(3) USB ホストへ INPUT レポート送信

7.2 HID デバイスクラス概要

7.2.1 クラスリクエスト

Table 7-1 に HID で対応しているクラスリクエストを示します。

Table 7-1 HID クラスリクエスト

リクエスト	コード	説明	対応
Get_Report	0x01	HID ホストへレポートを送信する	Yes
Set_Report	0x09	HID ホストからのレポートを受信する	Yes
Get_Idle	0x02	HID ホストへ Duration 時間を送信する	Yes
Set_Idle	0x0A	HID ホストからの Duration 時間設定を受信する	Yes
Get_Protocol	0x03	HID ホストへプロトコルを送信する	No
Set_Protocol	0x0B	HID ホストからのプロトコルを受信する	No
Get_Descriptor Descriptor Type : Class Class Descriptor Type : Report	0x06 (Standard)	レポートディスクリプタを送信する	Yes
Get_Descriptor Descriptor Type : Class Class Descriptor Type : HID	0x06 (Standard)	HID ディスクリプタを送信する	Yes

詳細は“USB Device Class Definitions for Human Interface Devices, Revision 1.1”の 7 章を参照ください。

7.2.2 クラスリクエスト データフォーマット

Table 7-2～Table 7-7 に、本ソフトウェアで対応しているクラスリクエストのフォーマットを示します。

(1) GetReport

ホストが、HID デバイスから Input または Feature レポートを、コントロール転送で行う為のクラスリクエストです。

Table 7-2 GetReport フォーマット

bmRequestType	bRequest	wValue	wIndex	wLength	Data
0xA1	GET_REPORT (0x01)	ReportType & ReportID	Interface	ReportLength	Report

(2) SetReport

ホストが、HID デバイスに対して Output または Feature レポートを、コントロール転送で行う為のクラスリクエストです。

Table 7-3 SetReport フォーマット

bmRequestType	bRequest	wValue	wIndex	wLength	Data
0x21	SET_REPORT (0x09)	ReportType & ReportID	Interface	ReportLength	Report

(3) GetIdle

ホストが HID デバイスからアイドルレートを要求する為のクラスリクエストです。

Table 7-4 GetIdle フォーマット

bmRequestType	bRequest	wValue	wIndex	wLength	Data
0xA1	GET_IDLE (0x02)	0(Zero) & ReportID	Interface	1(one)	Idle rate

(4) SetIdle

ホストが HID デバイスからアイドルレートを要求する為のクラスリクエストです。

Table 7-5 SetIdle フォーマット

bmRequestType	bRequest	wValue	wIndex	wLength	Data
0x21	SET_IDLE (0x0A)	Duration & ReportID	Interface	0(zero)	Not applicable

(5) GetProtocol

ホストが HID デバイスのプロトコル設定（ブートプロトコル／レポートプロトコル）を要求する為のクラスリクエストです。

Table 7-6 GetProtocol フォーマット

bmRequestType	bRequest	wValue	wIndex	wLength	Data
0xA1	GET_PROTOCOL (0x03)	0(Zero)	Interface	1(one)	0(BootProtocol) / 1(ReportProtocol)

(6) SetProtocol

ホストが HID デバイスにプロトコル（ブートプロトコル／レポートプロトコル）を設定する為のクラスリクエストです。

Table 7-7 SetProtocol フォーマット

bmRequestType	bRequest	wValue	wIndex	wLength	Data
0x21	SET_PROTOCOL (0x03)	0(BootProtocol) / 1(ReportProtocol)	Interface	0(zero)	Not applicable

8. パイプ仕様

パイプ仕様は、各 PHID サンプルデモ (マウス、キーボード、HID デモ)により異なります。

各動作モードの仕様は、ファイル *r_usb_phid_descriptor.c* を参照してください。

Table 8-1 マウスモード

パイプ番号	bEndpointAddress		bmAttributes	wMaxPacketSize	説明
	EP 番号	通信方向	転送タイプ	最大パケットサイズ	
PIPE0	EP0	In/Out	Control	64(Full-Speed) 8(Low-Speed)	標準リクエスト、クラスリクエスト
PIPE6	EP6	In	Interrupt	3	デバイスからホストへのデータ転送

Table 8-2 キーボードモード

パイプ番号	bEndpointAddress		bmAttributes	wMaxPacketSize	説明
	EP 番号	通信方向	転送タイプ	最大パケットサイズ	
PIPE0	EP0	In/Out	Control	64(Full-Speed) 8(Low-Speed)	標準リクエスト、クラスリクエスト
PIPE6	EP6	In	Interrupt	8	デバイスからホストへのデータ転送

Table 8-3 HID デモモード

パイプ番号	bEndpointAddress		bmAttributes	wMaxPacketSize	説明
	EP 番号	通信方向	転送タイプ	最大パケットサイズ	
PIPE0	EP0	In/Out	Control	64(Full-Speed) 8(Low-Speed)	標準リクエスト、クラスリクエスト
PIPE6	EP6	In	Interrupt	5	デバイスからホストへのデータ転送

[Notes]

PHID サンプルでは、INPUT レポート転送に PIPE6 を使用します。PIPE6 以外(PIPE7～9) を用いる場合は”USB_PHID_USE_PIPE”の値を変更します。

Pipe7 を使用する例

ファイル *../Workspace/PHID/inc/r_usb_phid_define.h*

```
//#define USB_PHID_USE_PIPE        USB_PIPE6
#define USB_PHID_USE_PIPE          USB_PIPE7
```

9. PHID デバイス クラスドライバ

9.1 基本機能

PHID の基本機能は、以下の通りです。

(1) HID ホストとのデータ送受信
(2) HID クラスリクエストに対する応答

9.2 PHID API 一覧

PHID API 一覧を Table 9-1 に示します。

Table 9-1 PHID API 関数一覧

関数名	機能概要
R_usb_phid_send_data	USB 送信処理
R_usb_phid_DeviceInformation	デバイス状態の情報取得
R_usb_phid_ChangeDeviceState	デバイス状態の変更
R_usb_phid_driver_start	PHID クラスドライバタスクの開始
R_usb_phid_TransferEnd	USB 転送強制終了
R_usb_phid_transfer_length	動作モードに従った USB 送信レポートサイズを取得
R_usb_phid_control_callback	HID 用コントロール転送処理コールバック関数
R_usb_phid_task	PHID タスク処理

9.2.1 API

R_usb_phid_send_data

USB 送信処理

形式

```
void        R_usb_phid_send_data ( uint8_t*     Table,
                                   usb_leng_t   size,
                                   usb_cb_t     complete)
```

引数

*Table	転送データ格納領域の先頭アドレス
size	転送サイズ
complete	処理完了通知コールバック関数

戻り値

―

解説

USB-BASIC-FW に対し、データ転送要求を行ないます。

引数“*Table*”が示すアドレスから引数“*size*” バイトのデータを送信します。

データ送信処理が完了するとコールバック関数を呼び出します。

補足

1. データ転送処理結果はコールバック関数の引数 “*usb_utr_t* *” で通知します。
2. *usb_utr_t* 構造体に関しては USB-BASIC-FW のアプリケーションノートを参照してください。

使用例

```
void usb_apl_task( void )
{
  uint8_t    send_data[] = {0x01,0x02,0x03,0x04,0x05}; /* USB send data */
  uint16_t   size = 5;                                 /* Data size */


  R_usb_phid_send_data((uint8_t *)send_data, size, (USB_CB_t)&usb_complete)
}

/* Callback function */
void  usb_complete( usb_utr_t *mess );
{
  /* Processing at the time of the completion of USB transmitting */
}
```

R_usb_phid_DeviceInformation

デバイス状態の情報取得

形式

void R_usb_phid_DeviceInformation(uint16_t *deviceinfo)

引数

*deviceinfo デバイスステート格納用バッファへのポインタ

戻り値

― ―

解説

与えられた USB ペリフェラルのデバイス情報を取得します。デバイス情報は引数“*deviceinfo*”で指定されたアドレスへ格納されます。

[0] ： USB デバイスステート

b15-b8 未使用

b7 VBSTS VBUS 入力ステータスビット

0：USBm_VBUS 端子が Low

1：USBm_VBUS 端子が High

b6-b4 DVSQ[2:0] デバイスステートビット

0 0 0：パワードステート

0 0 1：デフォルトステート

0 1 0：アドレスステート

0 1 1：コンフィギュレーションステート

1 x x：サスペンドステート x：Don’t care

b3-b0 未使用

[1] ： USB 転送速度

0x0000：未接続

0x00C0：Hi-Speed 接続(非サポート)

0x0080：Full-Speed 接続

0x0040：Low-Speed 接続

[2] ： 使用しているコンフィギュレーション番号

[3] ： 使用しているインタフェース番号

[4] ： リモートウェイクアップフラグ(0:ウェークアップ制御禁止 ,1:ウェークアップ制御許可)

補足

1. ユーザアプリケーションまたはクラスドライバから本関数を呼び出してください。

使用例

```
void  usb_smp_task(void)
{
  uint16_t res[5];
    :
  /* Get USB Device Information */
  R_usb_pstd_DeviceInformation((uint16_t *)res);
    :
}
```

R_usb_phid_ChangeDeviceState

デバイス状態の変更

形式

void	R_usb_phid_ChangeDeviceState(uint16_t msginfo)

引数

msginfo	USB 通信ステータス

戻り値

―	―

解説

デバイスステートの変更を行います。

msginfo で指定できるメッセージは以下のとおりです。

Msginfo	説明
USB_DO_REMOTEWAKEUP	PCD にリモートウェイクアップを要求します。

補足

1. ユーザアプリケーションまたはクラスドライバから本関数を呼び出してください。

使用例

```
void  usb_smp_task( void )
{
    :
  /* Change the device state request */
  R_usb_phid_ChangeDeviceState(USB_DO_REMOTEWAKEUP);
    :
}
```

9.2.2 共通 API

R_usb_phid_driver_start

PHID クラスドライバタスクの開始

形式

void	R_usb_phid_driver_start(void)

引数

―	―

戻り値

―	―

解説

PHID ドライバを開始します。

補足

1. 初期設定時にユーザアプリケーションで呼び出してください。

使用例

```
void usb_pstd_task_start( void )
{
    :
  usb_phid_driver_registration();   /*Peripheral Application Registration*/
  usb_papl_task_start();      /*Peripheral Application Task Start Setting*/
  R_usb_phid_driver_start(); /*Peripheral Class Driver Task Start Setting*/
  R_usb_pstd_usbdriver_start();   /* Peripheral USB Driver Start Setting */
    :
}
```

R_usb_phid_TransferEnd

USB 転送強制終了

形式

USB_ER_t R_usb_phid_TransferEnd(void)

引数

― ―

戻り値

― ―

解説

パイプ経由でのデータ転送を強制終了します。

本関数をコールすると、PCD に対してデータ転送強制終了要求を行います。 PCD は要求を受信すると、データ送信強制終了要求動作を行います。

データ転送が強制的に終了された場合でも、データ転送時に R_usb_phid_send_data でセットされたコールバック関数が呼び出されます。強制終了時の情報として、送信データ長、残りの受信データ長、ステータス、転送エラーのコード番号が、コールバック関数の引数(mess)へ設定されます。

補足

1. ユーザアプリケーションまたはクラスドライバから本関数を呼び出してください。

使用例

```
void  usb_smp_task(void)
{
    :
  /* Transfer end request */
  R_usb_phid_TransferEnd();
    :
}
```

R_usb_phid_transfer_length

動作モードに従った USB 送信レポートサイズを取得

形式

uint16_t R_usb_phid_transfer_length(void)

引数

― ―

戻り値

― PHID 動作モードの INPUT レポート長

解説

PHID 動作モードに対応した INPUT レポートサイズ (Byte)を取得します。

・マウスモード : 3 Byte

・キーボードモード : 8 Byte

補足

1. ユーザアプリケーションまたはクラスドライバから本関数を呼び出してください。

使用例

```
void usb_smp_task( void )
{
  uint16_t  usb_smp_report_length;
    :
  usb_smp_report_length = R_usb_phid_transfer_length();
    :
}
```

R_usb_phid_control_callback

HID 用コントロール転送処理コールバック関数

形式

void R_usb_phid_control_callback(usb_request_t *request, uint16_t ctsq)

引数

*request	クラスリクエストメッセージのポインタ	
ctsq	コントロール転送ステージ情報	
	USB_CS_IDST	Idle or setup stage
	USB_CS_RDDS	Control read data stage
	USB_CS_WRDS	Control write data stage
	USB_CS_WRND	Control write no data status stage
	USB_CS_RDSS	Control read status stage
	USB_CS_WRSS	Control write status stage
	USB_CS_SQER	Sequence error

戻り値

― ―

解説

リクエストタイプが HID クラスリクエストの場合、コントロール転送ステージに対応した処理を呼び出します。

本関数はデバイスクラスドライバ・レジストレーションでコントロール転送時に呼び出すコールバック関数として登録します。

補足

―

使用例

```
void usb_apl_task( void )
{
  usb_pcdreg_t  driver;

        :
  /* Control Transfer */
  driver.ctrltrans = R_usb_phid_control_callback;
  R_usb_pstd_DriverRegistration(&driver);
        :
}
```

R_usb_phid_Task

PHID タスク処理

形式

void	R_usb_phid_task(void)

引数

―	―

戻り値

―	―

解説

PHID タスクはアプリから要求された処理を行い、アプリ（usb_phid_main_task）に処理結果を通知します。

補足

本 API はユーザプログラムで呼び出してください。呼び出し方法については USB-BASIC-FW アプリケーションノートを参照してください。

使用例

```
void usb_apl_task_switch(void)
{
  while( 1 )
  {
    if( USB_FLGSET == R_usb_cstd_Schedule() )
    {
       /* PCD Task */
       R_usb_pstd_PcdTask();

        /* Peripheral HID Task */
       R_usb_phid_task();

       /* Peripheral HID demo sample application Task */
        usb_phid_main_task();
    }
  }
}
```

9.3 パイプ（エンドポイント）情報テーブル

PCD で使用するパイプ情報テーブルとディスクリプタテーブルは以下のとおりです。これらのテーブルはファイル *r_usb_phid_descriptor.c* を参照してください。

詳細は USB-BASIC-FW アプリケーションノートを参照してください。

Table 9-2 パイプ(エンドポイント)情報テーブル

番号	テーブル名	説明	備考
1	usb_gphid_EpTbl	エンドポイントテーブル： エンドポイント設定を定義します	動作モードにより値は異なります
2	usb_gphid_SmplDeviceDescriptor	デバイスディスクリプタ	ー
3	usb_gphid_Configuration	コンフィグレーションディスクリプタ	動作モードにより値は異なります *1
4	usb_gphid_StringDescriptor0	ストリングディスクリプタ 0	言語 ID
5	usb_gphid_StringDescriptor1	ストリングディスクリプタ 1	製造者名 (iManufacturer)
6	usb_gphid_StringDescriptor2	ストリングディスクリプタ 2	製造名 (iProduct)
7	usb_gphid_StringDescriptor3	ストリングディスクリプタ 3	製造番号 (iSerialNumber)
8	usb_gphid_StringDescriptor4	ストリングディスクリプタ 4	コンフィグレーション名 (HID デモ)
9	usb_gphid_StringDescriptor5	ストリングディスクリプタ 5	コンフィグレーション名 (キーボード)
10	usb_gphid_StringDescriptor6	ストリングディスクリプタ 6	コンフィグレーション名 (マウス)
11	usb_gphid_StrPtr	ストリングディスクリプタポインタ： ストリングディスリプタアドレスの配列です。	ー
12	usb_gphid_ReportDiscriptor	レポートディスクリプタ： レポートで扱うデータのフォーマットを定義します。	動作モードにより値は異なります。*2

*1

PHID はデバイス種別によって Configuration Descriptor が異なります。Descriptor の差異はヘッダファイル *r_usb_phid_usrcfg.h* でマクロ指定します。

Configuration Descriptor は、Interface、HID と Endpoint Descriptor を含みます。

Interface Descriptor の bInterface プロトコルは、PHID モードに依存します。

PHID モード	Protocol code	Product ID	Configuration num	Max packet size	Descriptor length
キーボード	1 (Keyboard)	0x0013	5	5	63
マウス	2 (Mouse)	0x0023	6	8	50

*2

キーボードモード及びマウスモードの ReportDescriptor の値は、”USB Device Class Definition for Human Interface Devices (HID) 1.11”の”E.6 Report Descriptor (Keyboard)”及び、”E.10 Report Descriptor (Mouse)”を参照下さい。

10. 制限事項

USB Peripheral Human Interface Devices Class Driver (PHID)には、以下の制限があります。

1. 低消費電力モードへの移行、復帰には対応していません。
2. キーボードモードでの複数キー同時押下や Control キー、Shift キー等の特殊キーは非対応です。

11. e² studio 用プロジェクトのセットアップ

(1). e² studio を起動してください。

※ はじめてe² studio を起動する場合、Workspace 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). [ファイル] → [インポート]を選択してください。インポートの選択ダイアログが表示されます。

(3). インポートの選択画面で、 [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。

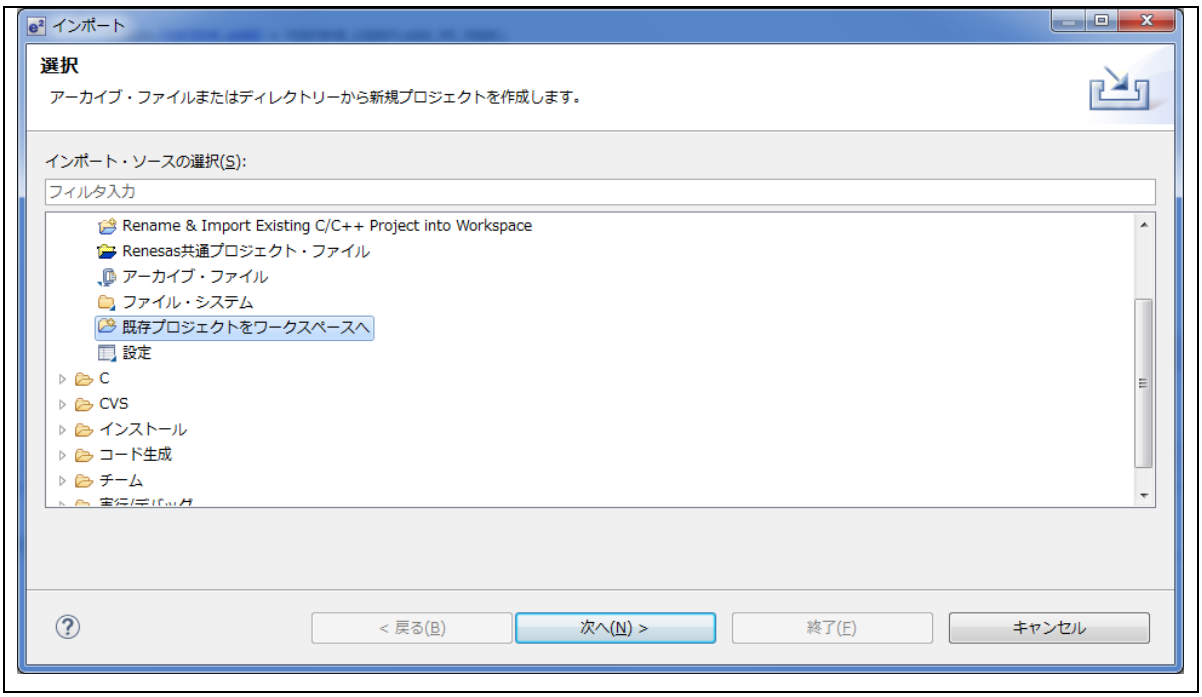


Figure 11-1 インポートの選択

(4). [ルートディレクトリの選択] の [参照] ボタンを押下して、「.cproject 」(プロジェクトファイル) が格納されたフォルダを選択して下さい。

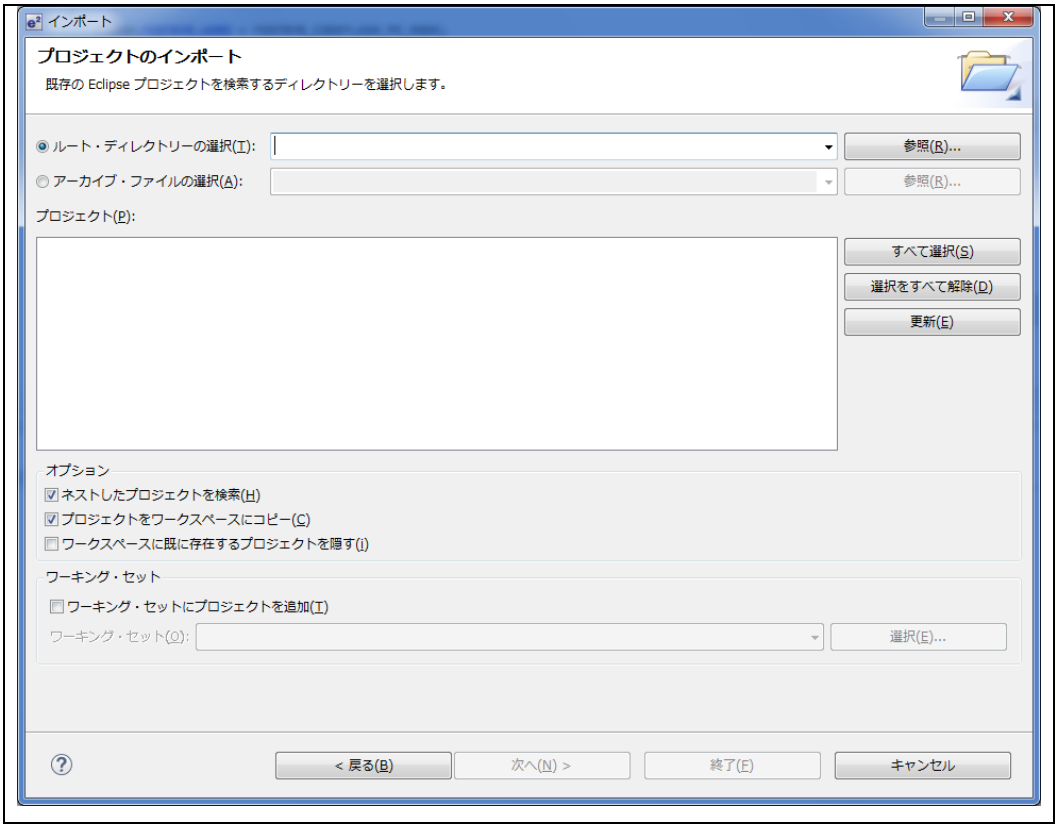


Figure 11-2　プロジェクトのインポート画面

(5). [終了]をクリック して下さい。

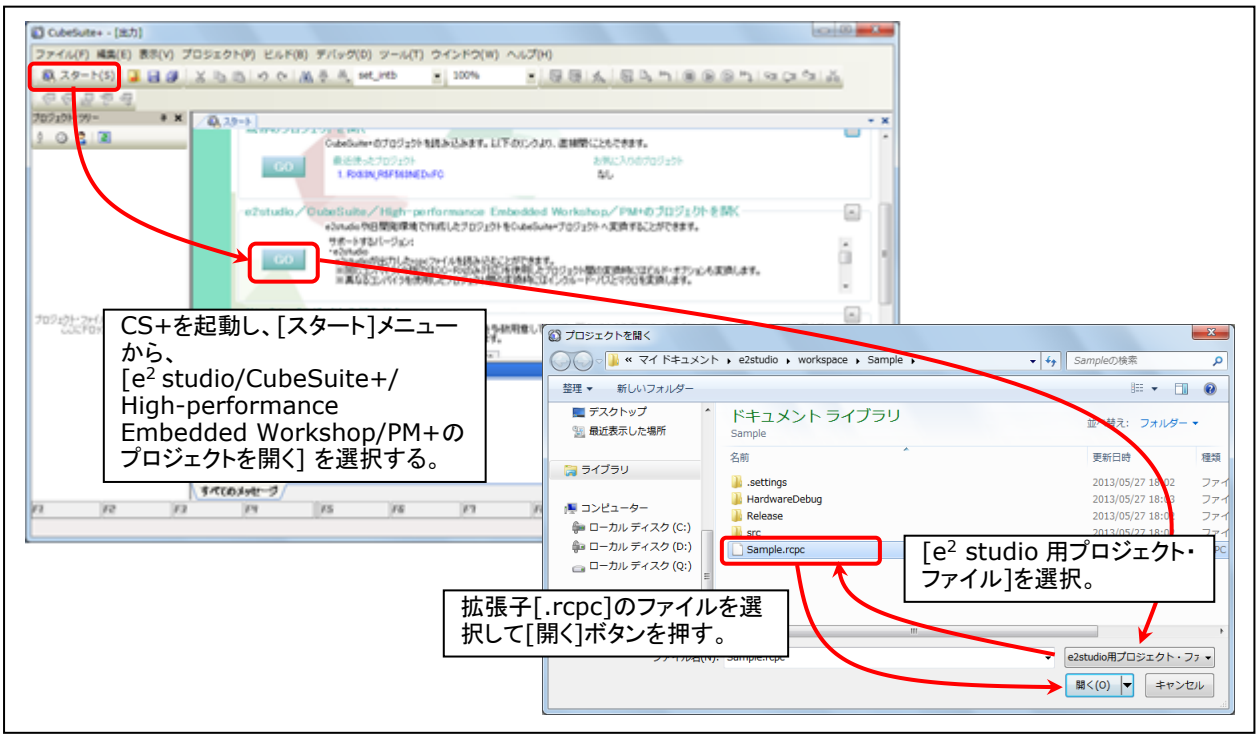
プロジェクトのワークスペースへのインポートが完了します。

12. e² studio 用プロジェクトを CS+で使用する場合

本プロジェクトは、統合環境 e² studio で作成されています。本プロジェクトを CS+で動作させる場合は、下記の手順を行ってください。

[Note]

rcpc ファイルは、workspace\RL78\CCRL\(MCU 名)フォルダ内に用意されています。



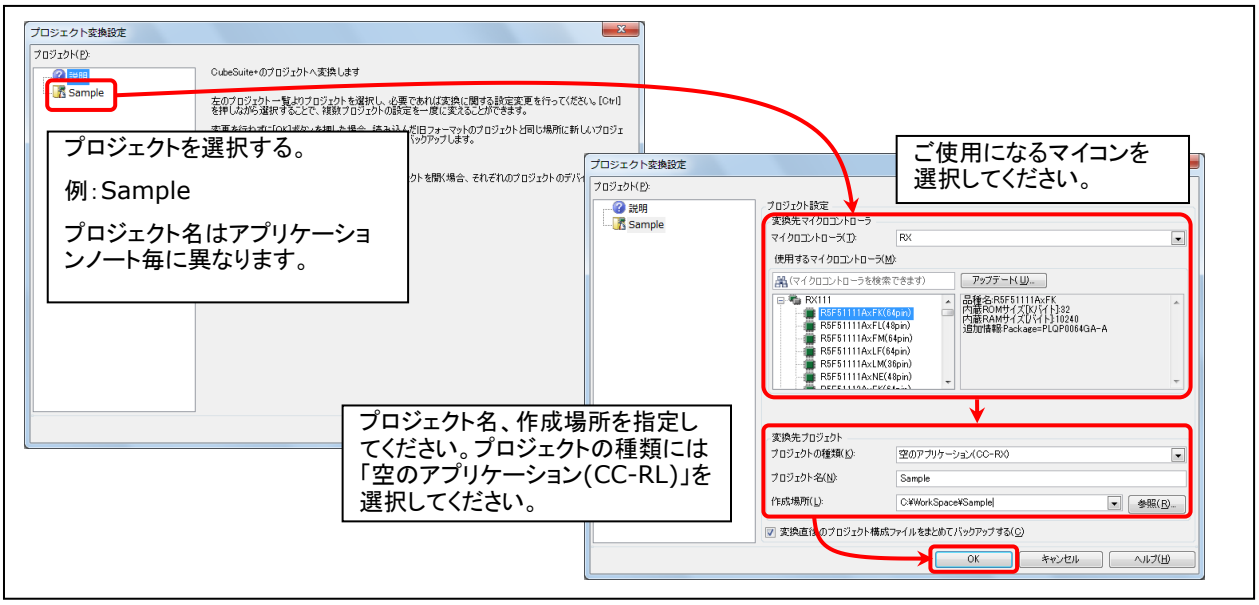


Figure 12-1 e² studio 用プロジェクトの CS+読み込み方法

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/contact/

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2011.05.12	—	初版発行
2.00	2013.01.24	—	ファームウェアアップデートによるドキュメントの改訂
2.10	2013.08.01	—	サポートデバイスに RX111,RL78/L1C を追加
2.11	2013.10.31	—	3.3.1 フォルダ構成を変更。これに伴い、1.4 および 3.2 のパス表記を修正。
2.12	2014.03.31	—	R8C に対応、誤記訂正
2.13	2015.03.16	—	RX111、R8C/3MK および R8C/3MU を動作確認デバイスから削除。
2.14	2016.01.18	—	Technical Update(発行番号: TN-RL*-A055A/J, TN-RL*-A033B/J)に対応しました。
2.15	2016.03.28	—	CC-RL コンパイラをサポートしました。

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/