

RXファミリ

R01AN2030JJ0140

Rev.1.40

Jun 30, 2022

USB Peripheral Communications Device Class Driver (PCDC) Firmware Integration Technology

要旨

本アプリケーションノートでは、Firmware Integration Technology (FIT) を使用した、USB Peripheral コミュニケーションデバイスクラスドライバ (PCDC) について説明します。本モジュールは USB Basic Host and Peripheral Driver(USB-BASIC-FW FIT モジュール)と組み合わせることで動作します。以降、本モジュールを USB PCDC FIT モジュールと称します。

対象デバイス

RX65N/RX651 グループ
RX64M グループ
RX71M グループ
RX66T グループ
RX72T グループ
RX72M グループ
RX66N グループ
RX72N グループ
RX671 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 Class Definitions for Communications Devices Revision 1.2
3. USB Communications Class Subclass Specification for PSTN Devices Revision 1.2
【<http://www.usb.org/developers/docs/>】
4. RX64M 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.R01UH0377)
5. RX71M 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.R01UH0493)
6. RX65N/RX651 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. R01UH0590)
7. RX65N/RX651-2M 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. R01UH0659)
8. RX66T 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. R01UH0749)
9. RX72T 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. R01UH0803)
10. RX72M 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. R01UH0804)
11. RX66N 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. R01UH0825)
12. RX72N 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. R01UH0824)
13. RX671 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. R01UH0899)
14. 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)

— 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..... | 4 |
| 3. API情報 | 5 |
| 4. コミュニケーションデバイスクラス (CDC) 、PSTN and ACM..... | 9 |
| 5. API..... | 13 |
| 6. コンフィグレーション (r_usb_pcdc_config.h)..... | 14 |
| 7. CDCドライバのインストール..... | 15 |
| 8. アプリケーションの作成方法 | 18 |

1. 概要

USB PCDC FIT モジュールは、USB-BASIC-FW FIT モジュールと組み合わせることで、USB Peripheral コミュニケーションデバイスクラスドライバ（以降 PCDC と記述）として動作します。PCDC は、USB コミュニケーションデバイスクラス仕様（以降 CDC と記述）の Abstract Control Model に準拠し、USB ホストとの通信を行うことができます。

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。

- USB ホストとのデータ転送
- CDC クラスリクエストに応答
- コミュニケーションデバイスクラスノーティフィケーション送信サービスの提供

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 Firmware Integration Technology アプリケーションノート(ドキュメント No.R01AN2025)を参照いただきますようお願いいたします。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

本ドライバは、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。

1.3 制限事項

Azure RTOS では、Composite Device (PCDC+PCDC)をサポートしていません。

1.4 用語一覧

本資料で使用される用語と略語は以下のとおりです。

| | | |
|--------------|---|---|
| ACM | : | Abstract Control Model. |
| APL | : | Application program |
| CDC | : | Communications Devices Class |
| CDCC | : | Communications Devices Class Communications Interface Class |
| CDCD | : | Communications Devices Class Data Class Interface |
| CPD | : | Serial Communication Port Driver |
| H/W | : | Renesas USB device |
| Non-OS | : | USB Driver for OS-less |
| PCD | : | Peripheral Control Driver for USB-BASIC-FW |
| PCDC | : | Peripheral Communications Devices Class |
| PCDCD | : | Peripheral Communications Devices Class Driver |
| PDCD | : | Peripheral Device Class Driver (Device Driver and USB Class Driver) |
| PSTN | : | Public Switched Telephone Network, contains the ACM (above) standard. |
| RTOS | : | USB Driver for the real-time OS |
| RSK | : | Renesas Starter Kits |
| USB-BASIC-FW | : | USB Basic Host and Peripheral Driver |

1.5 USB PCDC FIT モジュール

本モジュールは、r_usb_basic を使用したプロジェクトに組み込む必要があります。プロジェクトに組み込み後、API を使用することで USB の H/W 制御を行います。

2. ソフトウェア構成

Figure 2-1に PCDC 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。

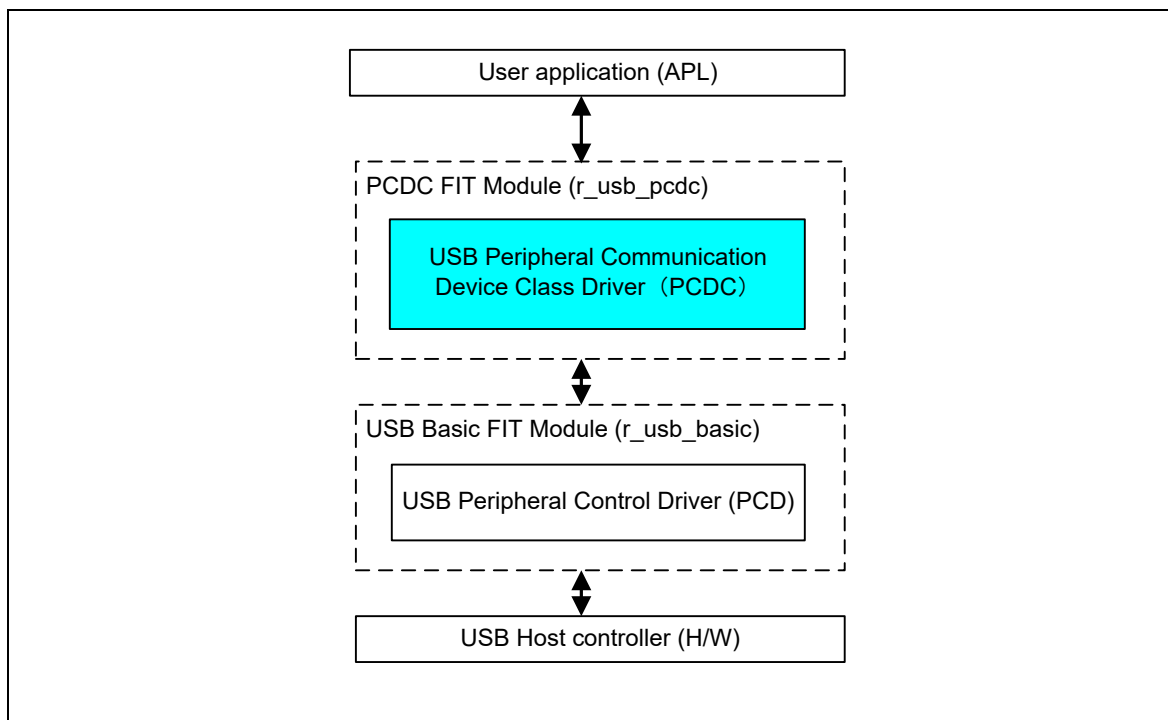


Figure 2-1 モジュール構成図

Table 2-1 各モジュール機能概要

| モジュール名 | 機能概要 |
|--------|---|
| PCDC | APL からの CDC に関するリクエストおよび、データ通信を PCD へ要求します。 |
| PCD | USB Peripheral H/W 制御ドライバです。 |
| CPD | シリアルポートの制御ドライバです。 |

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 ソフトウェアの要求

このドライバは以下のパッケージに依存しています。

- r_bsp
- r_usb_basic

3.3 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.3.03.00 (統合開発環境のデフォルト設定に"-lang = c99"オプションを追加) |
| | GCC for Renesas RX 4.08.04.201902 (統合開発環境のデフォルト設定に"-std = gnu99"オプションを追加) |
| | IAR C/C++ Compiler for Renesas version 4.12.01 |
| リアルタイム OS | FreeRTOS V.10.0.0 RI600V4 Azure RTOS (USBX) 6.1.10 |
| エンディアン | リトルエンディアン / ビッグエンディアン |
| モジュールのリビジョン | Rev.1.40 |
| 使用ボード | Renesas Starter Kits for RX64M Renesas Starter Kits for RX71M Renesas Starter Kits for RX65N, Renesas Starter Kits for RX65N-2MB Renesas Starter Kits for RX72T Renesas Starter Kits for RX72M Renesas Starter Kits for RX72N Renesas Starter Kits for RX671 |
| ホスト環境 | 下記の OS に接続し動作確認を行っています。 1. Windows® 8.1 2. Windows® 10 |

3.4 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一覧

| デバイス | 割り込みベクタ |
|------|---------|
| | |

| | |
|---|---|
| RX64M RX71M | USBIO 割り込み(ベクタ番号: 189, 割り込み要因番号 : 62, 選択型割り込み B) USB D0FIFO0 割り込み(ベクタ番号: 34) / USB D1FIFO0 割り込み(ベクタ番号: 35) USBR0 割り込み(ベクタ番号:90) |
| | USBAR 割り込み(ベクタ番号: 94) USB D0FIFO2 割り込み(ベクタ番号: 32) / USB D1FIFO2 割り込み(ベクタ番号: 33) |
| RX65N RX651 RX72M RX66N RX72N | USBIO 割り込み(ベクタ番号: 185, 割り込み要因番号 : 62, 選択型割り込み B) USB D0FIFO0 割り込み(ベクタ番号: 34) / USB D1FIFO0 割り込み(ベクタ番号: 35) USBR0 割り込み(ベクタ番号:90) |
| RX66T RX72T | USBIO 割り込み(ベクタ番号: 174) / USBR0 割り込み(ベクタ番号: 90) USB D0FIFO0 割り込み(ベクタ番号: 34) / USB D1FIFO0 割り込み(ベクタ番号: 35) |
| RX671 | USBIO 割り込み(ベクタ番号: 185, 割り込み要因番号 : 62, 選択型割り込み B) USB D0FIFO0 割り込み(ベクタ番号: 34) / USB D1FIFO0 割り込み(ベクタ番号: 35) USBR0 割り込み(ベクタ番号:90) |
| | USB11 割り込み(ベクタ番号: 182, 割り込み要因番号 : 63, 選択型割り込み B) USB D0FIFO1 割り込み(ベクタ番号: 36) / USB D1FIFO1 割り込み(ベクタ番号: 37) |

3.5 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_pcdc_if.h` に記載されています。

3.6 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。

3.7 コンパイル時の設定

コンパイル時の設定については、「6. コンフィグレーション (`r_usb_pcdc_config.h`)」章および 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)の「コンフィグレーション」章を参照してください。

3.8 ROM / RAM サイズ

本ドライバの ROM/RAM サイズを以下に示します。

1. CC-RX (最適化レベル: Default)

(1). Non-OS

| | 引数チェック実施時 | 引数チェック非実施時 |
|---------|--------------------|--------------------|
| ROM サイズ | 22.0K バイト (Note 3) | 21.6K バイト (Note 4) |
| RAM サイズ | 5.5K バイト | 5.5K バイト |

(2). RTOS

a. FreeRTOS

| | 引数チェック実施時 | 引数チェック非実施時 |
|---------|--------------------|--------------------|
| ROM サイズ | 35.4K バイト (Note 3) | 35.0K バイト (Note 4) |
| RAM サイズ | 21.0K バイト | 21.0K バイト |

b. RI600V4

| | 引数チェック実施時 | 引数チェック非実施時 |
|---------|--------------------|--------------------|
| ROM サイズ | 38.6K バイト (Note 3) | 38.2K バイト (Note 4) |
| RAM サイズ | 11.4K バイト | 11.4K バイト |

c. Azure RTOS

| | |
|---------|-----------|
| ROM サイズ | 51.2K バイト |
| RAM サイズ | 17.4K バイト |

2. GCC (最適化レベル: -O2)

a. Non-OS

| | 引数チェック実施時 | 引数チェック非実施時 |
|---------|--------------------|--------------------|
| ROM サイズ | 27.1K バイト (Note 3) | 26.6K バイト (Note 4) |
| RAM サイズ | 5.3K バイト | 5.3K バイト |

b. Azure RTOS

| | |
|---------|-----------|
| ROM サイズ | 59.7K バイト |
| RAM サイズ | 16.7K バイト |

3. IAR (最適化レベル: Medium)

a. Non-OS

| | 引数チェック実施時 | 引数チェック非実施時 |
|---------|--------------------|--------------------|
| ROM サイズ | 21.4K バイト (Note 3) | 20.9K バイト (Note 4) |
| RAM サイズ | 4.0K バイト | 4.0K バイト |

b. Azure RTOS

| | |
|---------|-----------|
| ROM サイズ | 41.0K バイト |
| RAM サイズ | 11.4K バイト |

[Note]

- 上記のサイズには、BSP および USB Basic Driver の ROM/RAM サイズが含まれています。
- 上記は V2 コアオプション指定時のサイズです。
- 「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。
- 「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。
- RTOS には、リアルタイム OS の ROM/RAM サイズが含まれています。
- Azure RTOS には USBX の ROM/RAM サイズが含まれています。

3.9 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 Firmware Integration Technology アプリケーションノート(ドキュメント No.R01AN2025)内の「構造体」の章を参照してください。

3.10 FIT モジュールの追加方法

本モジュールは、使用するプロジェクトごとに追加する必要があります。ルネサスでは、Smart

Configurator を使用した(1)、(3)の追加方法を推奨しています。ただし、Smart Configurator は、一部の RX デバイスのみサポートしています。サポートされていない RX デバイスについては(2)、(4)の方法を使用してください。

(1) e² studio 上で Smart Configurator を使用して FIT モジュールを追加する場合

e² studio の Smart Configurator を使用して、自動的にユーザプロジェクトに FIT モジュールを追加します。詳細は、アプリケーションノート「Renesas e² studio スマート・コンフィグレータ ユーザーガイド (R20AN0451)」を参照してください。

(2) e² studio 上で FIT Configurator を使用して FIT モジュールを追加する場合

e² studio の FIT Configurator を使用して、自動的にユーザプロジェクトに FIT モジュールを追加することができます。詳細は、アプリケーションノート「RX ファミリ e² studio に組み込む方法 Firmware Integration Technology (R01AN1723)」を参照してください。

(3) CS+上で Smart Configurator を使用して FIT モジュールを追加する場合

CS+上で、スタンドアロン版 Smart Configurator を使用して、自動的にユーザプロジェクトに FIT モジュールを追加します。詳細は、アプリケーションノート「Renesas e² studio スマート・コンフィグレータ ユーザーガイド (R20AN0451)」を参照してください。

(4) CS+上で FIT モジュールを追加する場合

CS+上で、手動でユーザプロジェクトに FIT モジュールを追加します。詳細は、アプリケーションノート「RX ファミリ CS+に組み込む方法 Firmware Integration Technology (R01AN1826)」を参照してください。

4. コミュニケーションデバイスクラス (CDC) 、PSTN and ACM

4.1 基本機能

CDC は、コミュニケーションデバイスクラス仕様 Abstract Control Model サブクラス (4.2章参照) に準拠しています。

4.2 Abstract Control Model 概要

Abstract Control Model サブクラスは、USB 機器と従来のモデム (RS-232C 接続) との間を埋める技術で、従来のモデムを使用するアプリケーションプログラムが使用可能です。

以下に本 S/W でサポートするクラスリクエスト・クラスノーティフィケーションを記します。

4.2.1 クラスリクエスト (ホスト→デバイスへの通知)

本ドライバは以下のクラスリクエストを受信するとアプリケーションプログラムに通知します。

アプリケーションプログラムでのクラスリクエスト処理については、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「クラスリクエスト」の章を参照してください。

Table 4-1 CDC クラスリクエスト

| リクエスト | コード | 説明 |
|---------------------|------|---|
| SetLineCoding | 0x20 | 通信回線設定を行う。(通信速度, データ長, パリティビット, ストップビット長) |
| GetLineCoding | 0x21 | 通信回線設定状態を取得する。 |
| SetControlLineState | 0x22 | 通信回線制御信号 RTS、DTR の設定を行う。 |

Abstract Control Model リクエストについては、USB Communications Class Subclass Specification for PSTN Devices Revision 1.2 の Table11 : Requests-Abstract Control Model を参照して下さい。

4.2.2 クラスリクエストのデータフォーマット

CDC が対応するクラスリクエストのデータフォーマットを以下に記します。

1. SetLineCoding

UART 回線設定を行う為にホストがデバイスに対して送信するクラスリクエストです。
SetLineCoding データフォーマットを以下に示します。

Table 4-2 SetLineCoding フォーマット

| bmRequestType t | bReques | wValue | wIndex | wLength | Data |
|-----------------|---------------------------|--------|--------|---------|---|
| 0x21 | SET_LINE_CODING (0x20) | 0x00 | 0x00 | 0x07 | Line Coding Structure Table 4-3 Line Coding Structureフォーマット参照 |

Table 4-3 Line Coding Structure フォーマット

| Offset | Field | Size | Value | Description |
|--------|-------------|------|--------|---|
| 0 | DwDTERate | 4 | Number | データ端末の速度 (bps) |
| 4 | BcharFormat | 1 | Number | ストップビット 0 - 1 Stop bit 1 - 1.5 Stop bit 2 - 2 Stop bit |
| 5 | BparityType | 1 | Number | パリティ 0 - None 1 - Odd 2 - Even |
| 6 | BdataBits | 1 | Number | データビット (5、6、7、8) |

2. GetLineCoding

UART 回線設定状態を要求する為にホストがデバイスに対して送信するクラスリクエストです。
GetLineCoding データフォーマットを以下に示します。

Table 4-4 GetLineCoding フォーマット

| bmRequestType t | bReques | wValue | wIndex | wLength | Data |
|-----------------|---------------------------|--------|--------|---------|---|
| 0xA1 | GET_LINE_CODING (0x21) | 0x00 | 0x00 | 0x07 | Line Coding Structure Table 4-3 Line Coding Structureフォーマット参照 |

3. SetControlLineState

UART のフロー制御用信号を設定する為にホストがデバイスに対して送信するクラスリクエストです。
本 S/W では RTS/DTR の制御をサポートしていません。
SET_CONTROL_LINE_STATE データフォーマットを以下に示します。

Table 4-5 SET_CONTROL_LINE_STATE フォーマット

| bmRequestType t | bReques | wValue | wIndex | wLength | Data |
|-----------------|-----------------------------------|---|--------|---------|------|
| 0x21 | SET_CONTROL_ LINE_STATE (0x22) | Control Signal Bitmap Table 4-6 Control Signal Bitmapフォーマット参照 | 0x00 | 0x00 | None |

Table 4-6 Control Signal Bitmap フォーマット

| Bit Position | Description |
|--------------|---|
| D15~D2 | 予約 (0 にリセット) |
| D1 | DCE の送信機能を制御 0 - RTS OFF 1 - RTS ON |
| D0 | DTE がレディ状態かの通知 0 - DTR OFF 1 - DTR ON |

4.2.3 クラスノーティフィケーション（デバイス→ホストへの通知）

本 S/W のクラスノーティフィケーション対応/非対応をTable 4-7下表に示します。

Table 4-7 CDC クラスノーティフィケーション

| ノーティフィケーション | コード | 説明 | 対応 |
|--------------------|------|-------------------------------|----|
| NETWORK_CONNECTION | 0x00 | ネットワーク接続状況を通知する | × |
| RESPONSE_AVAILABLE | 0x01 | GET_ENCAPSLATED_RESPONSE への応答 | × |
| SERIAL_STATE | 0x20 | シリアル回線状態を通知する | ○ |

1. SerialState

UART ポートに状態変化を検出した場合、ホストへ状態通知を行います。

本 S/W ではオーバーランエラー、パリティエラー、フレーミングエラー検出をサポートしています。状態通知は正常状態からエラー検出した場合に行います。エラーを連続検出しても状態通知を連続送信しません。

SerialState データフォーマットを以下に示します。

Table 4-8 SerialState フォーマット

| bmRequestType | bReques | wValue | wIndex | wLength | Data |
|---------------|------------------------|--------|--------|---------|---|
| 0xA1 | SERIAL_STATE (0x20) | 0x00 | 0x00 | 0x02 | UART State bitmap Table 4-9 UART State bitmapフォーマット参照 |

Table 4-9 UART State bitmap フォーマット

| Bits | Field | Description | 対応 |
|--------|---------------|---------------------------------|----|
| D15~D7 | | 予約 | — |
| D6 | b_over_run | オーバーランエラー検出 | ○ |
| D5 | b_parity | パリティエラー検出 | ○ |
| D4 | b_framing | フレーミングエラー検出 | ○ |
| D3 | b_ring_signal | 着信 (Ring signal) を感知した | × |
| D2 | b_break | ブ레이크信号検出 | × |
| D1 | btx_carrier | Data Set Ready : 回線が接続されて通信可能 | × |
| D0 | brx_carrier | Data Carrier Detect : 回線にキャリア検出 | × |

4.3 PC の仮想 COM ポートについて（参考）

Windows OS 搭載 PC は CDC デバイスを仮想 COM ポートとして利用することが可能です。

Windows OS 搭載 PC に本 S/W を実装した RSK ボードを接続すると、エnumレーション設定に続き、CDC クラスリクエストの `GetLineCoding` 及び `SetControlLineState` を行った後、仮想 COM デバイスとしてデバイスマネージャに登録されます。Windows デバイスマネージャに仮想 COM ポートとして登録された後は、WindowsOS 標準搭載のハイパーターミナル等のターミナルアプリで CDC デバイスとデータ通信が可能です。

ターミナルアプリのシリアルポート設定を行うことで、クラスリクエスト `SetLineCoding` による UART 設定が可能です。ターミナルアプリのウインドウから入力したデータ（又はファイル送信）は EP2 を使用して RSK ボードへ転送され、RSK ボード側から PC へのデータ転送は EP1 を使用して行われます。ターミナルアプリによっては最後に受信したデータが MAX パケットサイズの場合、継続するデータがあると判断して受信データをターミナルに表示しないことがあります。この場合、MAX パケットサイズ未満のデータを受信することで、それまでに受信したデータがターミナルに表示されます。

5. API

アプリケーションプログラム内で使用する API については、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. コンフィグレーション (r_usb_pcdc_config.h)

お客様のシステムにあわせて以下の設定をお願いします。

[Note]

必ず r_usb_basic_config.h ファイルに対する設定もお願いします。r_usb_basic_config.h については、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「コンフィグレーション」の章を参照してください。

1. Bulk IN/OUT 転送用パイプ設定

Bulk IN, Bulk OUT 転送で使用するパイプ番号(PIPE1 から PIPE5)を指定してください。

```
#define USB_CFG_PCDC_BULK_IN      パイプ番号 (USB_PIPE1 から USB_PIPE5)
#define USB_CFG_PCDC_BULK_OUT    パイプ番号 (USB_PIPE1 から USB_PIPE5)
#define USB_CFG_PCDC_BULK_IN2    パイプ番号 (USB_PIPE1 から USB_PIPE5)
#define USB_CFG_PCDC_BULK_OUT2   パイプ番号 (USB_PIPE1 から USB_PIPE5)
```

[Note]

- a. 上記の各マクロには同じパイプ番号と指定しないで下さい。
- b. お客様のシステムが Comoposite Device(PCDC+PCDC)ではない場合、USB_CFG_PCDC_BULK_IN2 と USB_CFG_PCDC_BULK_OUT2 マクロにはパイプ番号を指定する必要はありません。

2. Interrupt IN 転送用パイプ設定

Interrupt IN 転送で使用するパイプ番号(PIPE6 から PIPE9)を指定してください。

```
#define USB_CFG_PCDC_INT_IN      パイプ番号 (USB_PIPE6 から USB_PIPE9)
#define USB_CFG_PCDC_INT_IN2    パイプ番号 (USB_PIPE6 から USB_PIPE9)
```

[Note]

- a. 上記の各マクロには同じパイプ番号と指定しないで下さい。
- b. お客様のシステムが Comoposite Device(PCDC+PCDC)ではない場合、USB_CFG_PCDC_INT_IN2 マクロにはパイプ番号を指定する必要はありません。

7. CDC ドライバのインストール

USB HostがPC(Windows®)の場合、そのPCに対しCDCドライバをインストールする必要があります。本サンプルプログラムの書き込みを行ったRSKをPCに接続すると、Figure 7-1に示すウィザードが表示され、CDCドライバのインストールが行われます。

- (1). デバイス・マネージャより、ドライバーソフトウェアの更新を選択します。
- (2). 《コンピューターを参照してドライバーソフトウェアを検索します (R) 》を選択します。

Note:

- (1). PCのOSがWindows® 10の場合、CDCドライバのインストール作業は不要です。
- (2). PCのOSがWindows® 8.1の場合、デジタル署名済のカタログファイルが必要になります。デジタル署名済のカタログファイルはお客様により作成いただく必要があります。

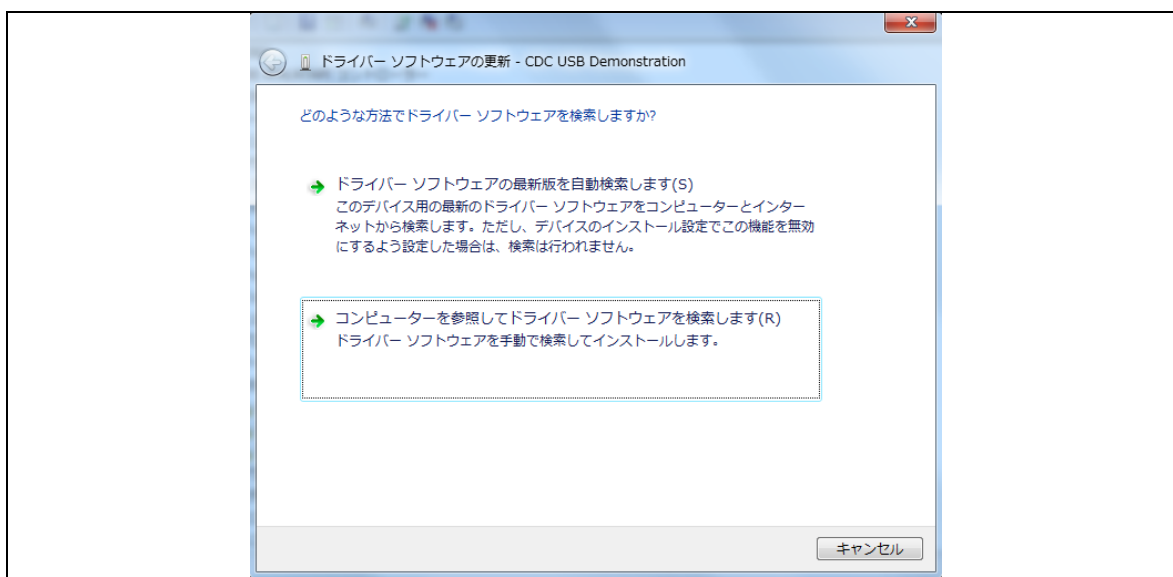


Figure 7-1 新しいハードウェアの検索ウィザード

- (3). 《次の場所で最適のドライバーソフトウェアを検索します》を選択します。
“参照 (R) ” をクリックして“CDC_Demo.inf”の存在するフォルダを指定し、“次へ (N) ” をクリックしてください。

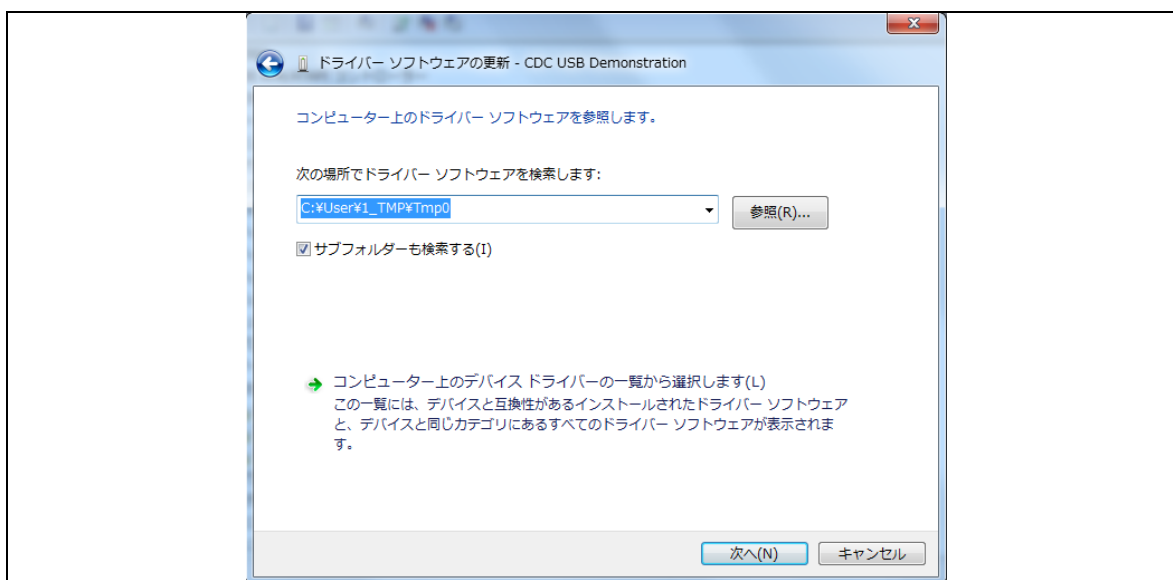


Figure 7-2 ドライバの場所の選択

Note:

CDC_Demo.inf ファイルは、パッケージ内の "r_usb_pcdc_mini\utilities" フォルダに格納されています。

- (4). 次のインストール確認画面が表示される場合は、“このドライバーソフトウェアをインストールします (I)” をクリックしてください。

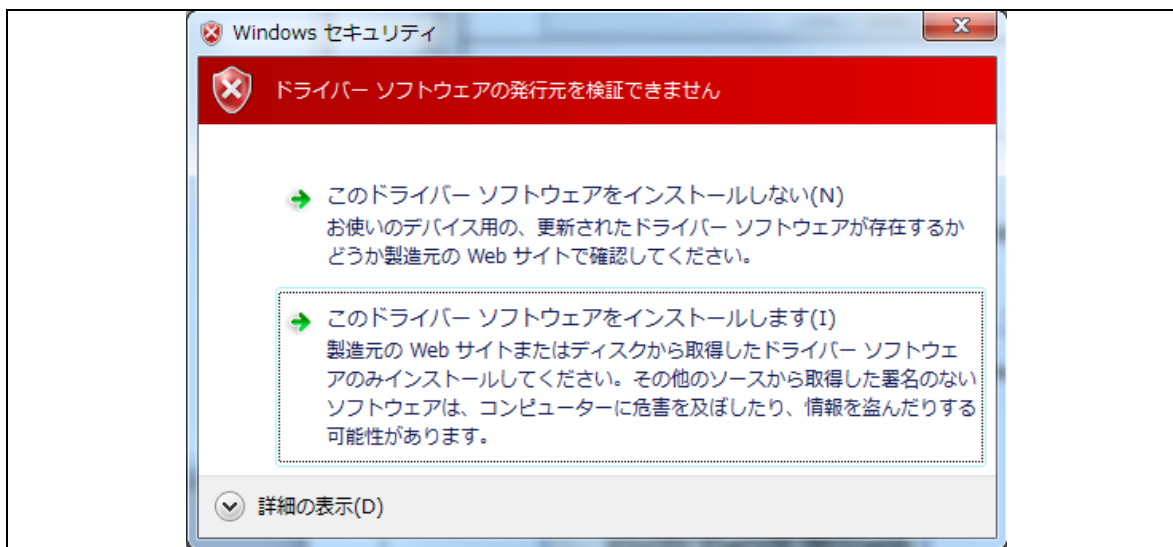


Figure 7-3 インストール確認

- (5). 次のウィンドウが表示されたら、CDC ドライバのインストールは完了です。“閉じる” をクリックしてください。

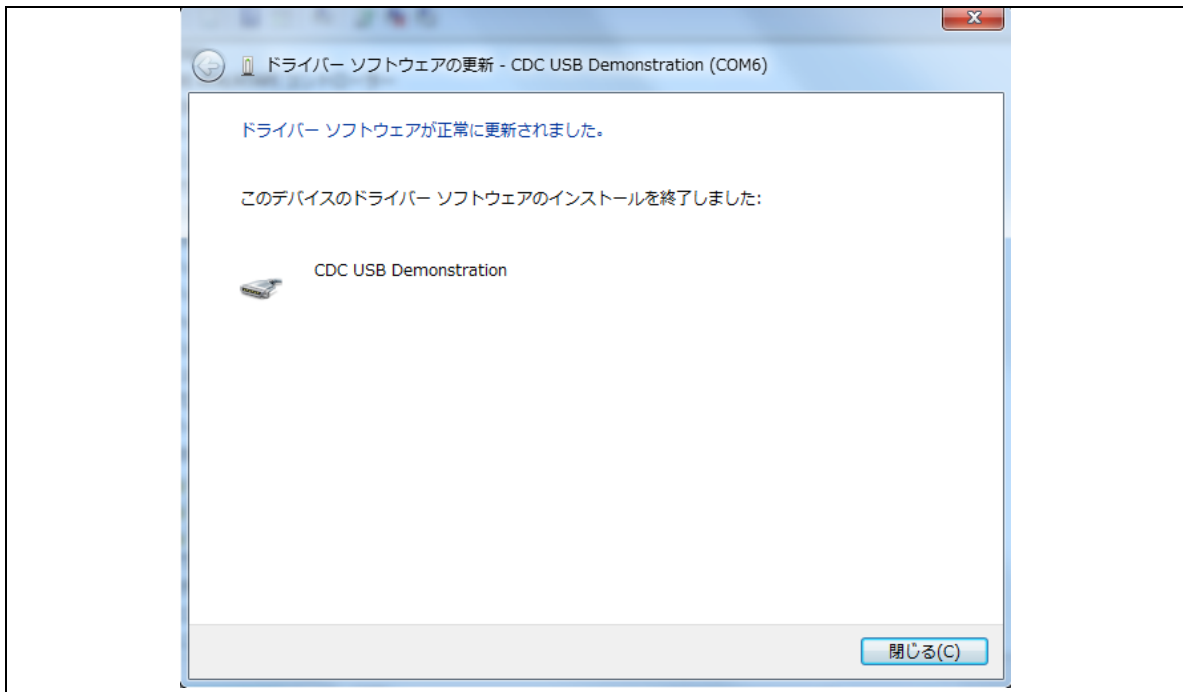


Figure 7-4 インストール完了

8. アプリケーションの作成方法

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「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。

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>

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 |
| 1.00 | Aug 1, 2014 | — | 初版発行 |
| 1.10 | Sep 30, 2015 | — | 対象デバイスに RX71M を追加。 |
| 1.20 | Sep 30, 2016 | — | 1. 対象デバイスに RX65N/RX651 を追加 2. DMA 転送をサポート 3. 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 |
| 1.21 | Mar 31, 2017 | — | 1. Technical Update(発行番号: TN-RX*-A172A/J)に対応しました。 2. 「API」の章を USB Basic Host and Peripheral Driver Firmware Integration Technology (ドキュメント番号:R01AN2025)のドキュメントへ移しました。 |
| 1.22 | Sep 30, 2017 | — | RX65N/RX651-2M をサポート |
| 1.23 | Mar 31, 2018 | — | Smart Configurator に対応しました。 |
| 1.24 | Dec 28, 2018 | — | RTOS をサポート |
| 1.25 | Apr 16, 2019 | — | 対象デバイスに RX66T/RX72T を追加 |
| 1.26 | May 31, 2019 | — | 1. GCC/IAR コンパイラをサポートしました。 2. 対象デバイスから RX63N を削除しました。 |
| 1.27 | Jul 31, 2019 | — | 対象デバイスに RX72M を追加 |
| 1.30 | Mar 1, 2020 | — | 1. リアルタイム OS(uITRON:RI600V4)をサポートしました。 2. 対象デバイスに RX72N/RX66N を追加 |
| 1.31 | Mar 1, 2021 | — | 対象デバイスに RX671 を追加 |
| 1.40 | Jun 30, 2022 | — | Azure RTOS(USBX PCDC)をサポートしました。 |
| | | | |
| | | | |
| | | | |