Renesas Synergy™プラットフォーム
UART HAL モジュールガイド

R11AN0085JU0101
Rev.1.01
2019.09.11

（注 1）本資料は英語版を翻訳した参考資料です。内容に相違がある場合には英語版を優先します。資料によっては英語版のバージョンが更新され、内容が変わっている場合があります。日本語版は、参考用としてご使用のうえ、最新および正式な内容については英語版のドキュメントを参照ください。

（注 2）本資料の第 6 章まで（要旨除く）の日本語訳は、「Synergy™ Software Package (SSP) v1.5.0 ユーザーズマニュアルモジュール概要編（参考資料）」の第 4 章「モジュールの概要」に掲載されていますのでそちらを参照ください。

要旨（Introduction）

本モジュールガイドは、ユーザが UART HAL モジュールを効果的に使用してシステムが開発できるようになることを目的としています。このモジュールガイドの内容を習得することで、開発システムへのモジュールの追加とターゲットアプリケーション向けの正確な設定（configuration）ができ、さらに付属のアプリケーションプロジェクトコードを参照して、効率的なコード記述が行えるようになります。より詳細な API や、より高度なモジュール使用法を記述した他のアプリケーションプロジェクト例もルネサス WEB サイト（本書末尾の「参考情報」の章を参照）から入手でき、より複雑な設計に役立ちます。

UART HAL モジュールは、UART（汎用非同期送受信）アプリケーション向けのハイレベル API（high-level API）であり、r_sci_uart に実装されています。UART HAL モジュールは、Synergy MCU 上にある SCI 周辺回路（peripheral）を使用します。必要な場合、ハードウェアハンドシェイク（hardware-handshake）とデータ操作を管理する目的で、ユーザ定義のコールバック関数（user-defined call back function）を作成することもできます。

加えて UART HAL モジュールの応用として、DK-S7G2 を使用した RS-485 通信について説明します。

目次

1. UART HAL Module Features
2. UART HAL Module APIs Overview
3. UART HAL Module Operational Overview
4. Including the UART HAL Module in an Application
5. Configuring the UART HAL Module
6. Using the UART HAL Module in an Application
7. UART HAL モジュールのアプリケーションプロジェクト（The UART HAL Module Application Project）

このモジュールガイドで説明するアプリケーションプロジェクトを実際に使うことで、設計全体の手順を体験することができます。このプロジェクトは、このドキュメントの末尾にある「参考情報」章に掲載されているリンクから入手することができます。ISDE でアプリケーションプロジェクトをインポートして開き、UART HAL モジュールに対応する設定項目を表示することができます。また、システムにおける UART API を理解するために、コード `uart_hal_mg.c` を確認することもできます。

本アプリケーションプロジェクトは、2 種類の UART API の使用方法を示します。1 つは汎用操作（generic operation）に関するもので、もう 1 つは外部 GPIO 端子を RTS 信号として使用し、CTS/RTS ハードウェアフロー制御（hardware flow-control）を使用するものです。フロー制御のサンプルを実行するためには、データ（規定したバイト）を SK-S7G2 に送信するために 2 つ目のボードが必要になります。

以下の表に、このアプリケーションプロジェクトが使用するのに必要となるソフトウェアおよびハードウェアのバージョンを示します。

表 9　このアプリケーションプロジェクトが使用するソフトウェアとハードウェアのリソース

リソース	リビジョン	説明
e² studio	5.3.1 またはそれ以降	統合ソリューション開発環境（ISDE）
SSP	1.2.0 またはそれ以降	Synergy ソフトウェアプラットフォーム
IAR EW for Renesas Synergy	7.71.2 またはそれ以降	IAR Embedded Workbench for Renesas Synergy
SSC	5.3.1 またはそれ以降	Synergy Standalone Configurator
SK-S7G2（2 個）	v3.0、v3.1 またはそれ以降	スタータキット

重要注意事項：フロー制御サンプルのデモを実行するには、SK-S7G2 キットが 2 つ必要です。

汎用操作（generic operation）のフローは、以下の通りです（UART4 が対象）。

- `open` API を使用して、UART HAL モジュールを初期化します。
- `read` API を使用して、外部デバイスから受信するデータを格納するためのデータバッファを初期化します。
- イベントコード UART_EVENT_RX_COMPLETE が設定され、データ受信が完了した時点で、割り込みコールバック関数（interrupt callback function）内でフラグをセット（1 に設定）します。
- 受信完了フラグ（received completion flag）がセットされた時点で、受信したデータをアプリケーションの必要に応じて操作します。
- アプリケーションに応じて、データを外部デバイスに送信します。
- 送信が完了するまで待機します。最後のデータを送信した後、割り込みコールバック関数内でイベントコード UART_EVENT_TX_COMPLETE を設定します。
- データ送信が終了した後、`close` API を使用して、UART HAL モジュールを閉じます。

外部 GPIO 端子を RTS 信号として使用し、CTS/RTS ハードウェアフロー制御を行う特定操作（specific operation）の流れは以下のとおりです（UART0 が対象）。

- `open` API を使用して、UART HAL モジュールを初期化します。
- `baudSet` API を使用し、必要に応じてボーレートを変更します。
- （イベントコード UART_EVENT_RX_CHAR が設定された時点で）割り込みコールバック関数内で、受信したデータをユーザ定義バッファに格納します。
- データ受信が完了した時点で、割り込みコールバック関数内でフラグをセットします。
- 受信完了フラグがセットされた時点で、受信したデータをアプリケーションの必要に応じて操作します。
- アプリケーションに応じて、データを外部デバイスに送信します。
- 送信が完了するまで待機します。最後のデータを送信した後、割り込みコールバック関数内でイベントコード UART_EVENT_TX_COMPLETE を設定します。
- データ送信が終了した後、`close` API を使用して、UART HAL モジュールを閉じます。

以下の図に、このアプリケーションプロジェクトのシンプルなフローを示します。

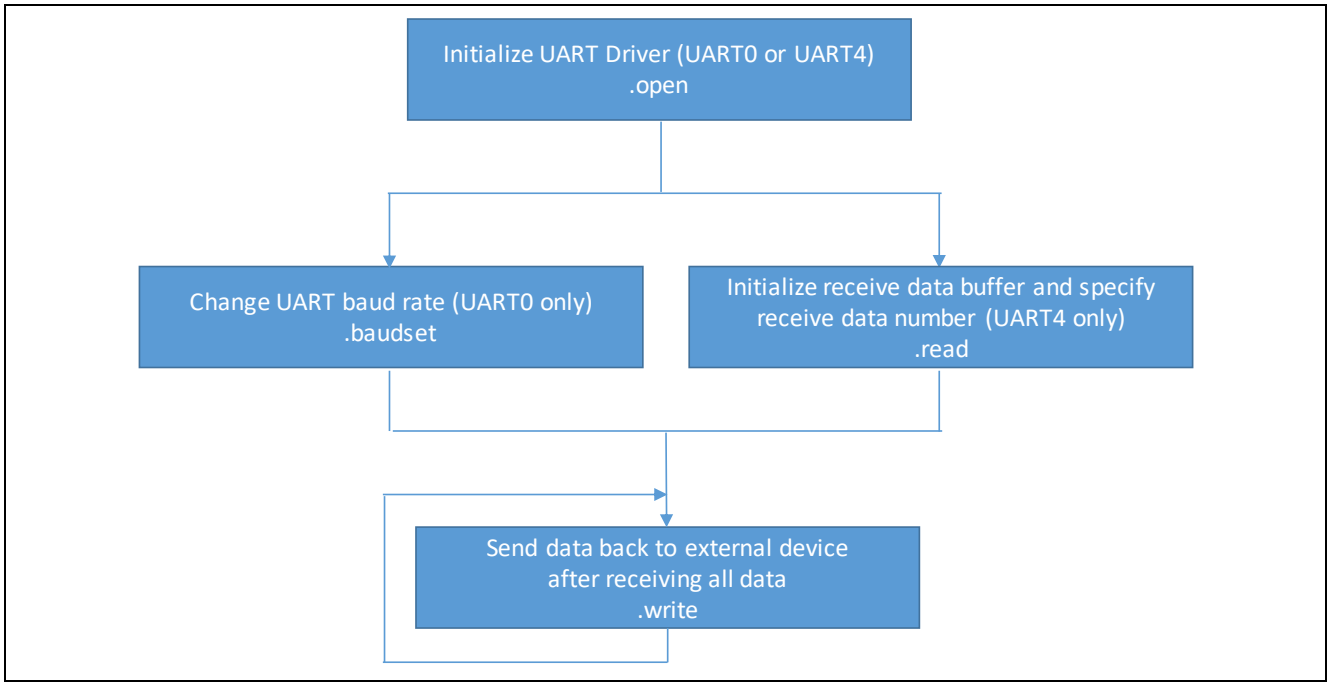

図 6　メインループ内での UART アプリケーションプロジェクトのフロー

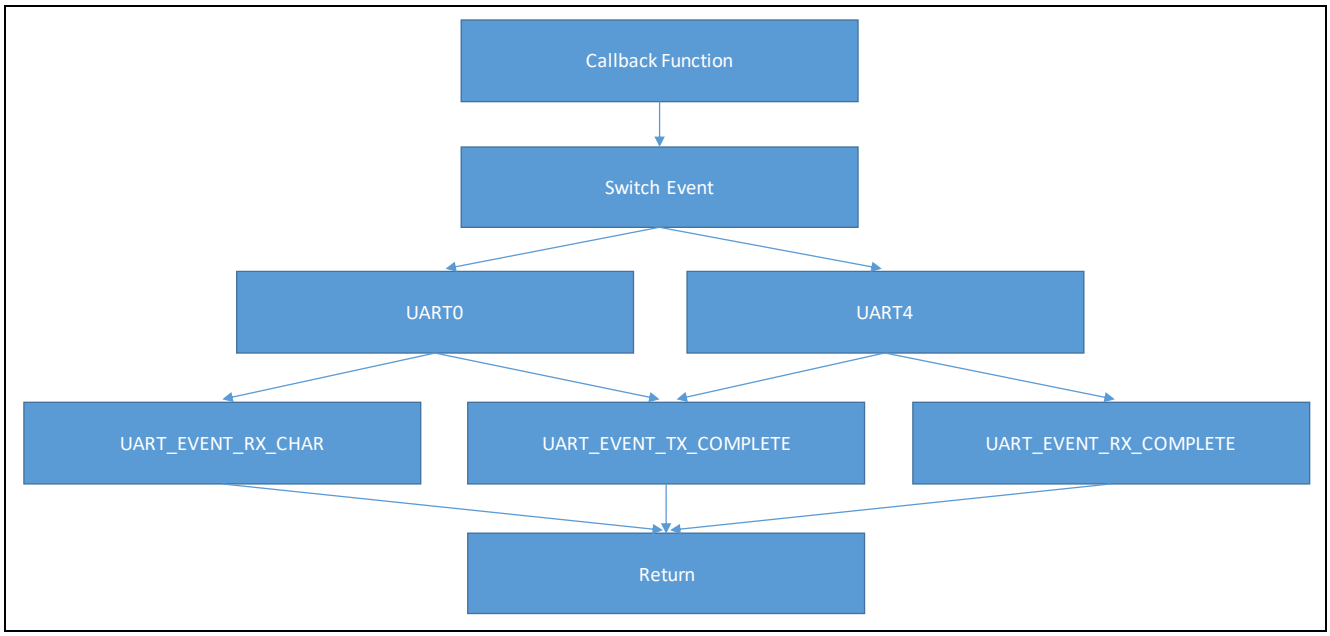

図 7　コールバック関数の詳細なフローチャート

uart_hal_mg.c ファイルは、このプロジェクトを ISDE にインポートすることにより、プロジェクト内に配置されます。ISDE でこのファイルを開き、API の使い方のガイドを受けることができます。

uart_hal_mg.c の最初のセクションはヘッダファイルであり、UART インスタンス構造体（UART instance structure）と、外部関数宣言（external function declaration）を参照しています。hal_entry.c は、uart_hal_demo()関数を呼び出します。この関数は、uart_hal_mg.c 内にあります。このデモプロジェクトでは、汎用操作（UART4）と特定操作（UART0）の 2 種類の操作を実行しています。

- 汎用モード（generic mode）で read API を使用してバイト数を設定する方法により、複数のデータ（16 バイト）をユーザ定義バッファ内に自動的に格納することができます。この場合、DTC を使用してデータを連続して受信します。一方、特定モード（specific mode）ではコールバック関数内で、各データをユーザ定義バッファに手動で格納する必要があります。
- 汎用モードでのデータトランザクション（data transaction）ステータスは、コールバック関数内でイベント列挙型（event enumeration）UART_EVENT_RX_COMPLETE と UART_EVENT_TX_COMPLETE を使用して確認できます。一方、特定モードの場合、イベント列挙型 UART_EVENT_RX_CHAR と UART_EVENT_TX_COMPLETE を使用して、このステータスを確認することができます。
- 特定操作では、どの端子を RTS 信号として使用するかを指定する目的で、コールバック関数 user_rts_callback も使用します。

セミホスト機能（semi-hosting function）が有効な場合、printf 関数はすべての受信データと送信データをデバッグ仮想コンソール（Debug Virtual Console）に出力します。

注記：この説明は、Synergy ソフトウェアパッケージ内のデバッグコンソールで printf() を使用する方法をユーザが理解していることを想定しています。このような経験がない場合は、下記 WEB サイトの FAQ 2000008「Synergy ソフトウェアパッケージのデバッグコンソールで Printf_使用方法」という記事を参照してください。デバッグモードで変数ウォッチ機能を使用して結果を表示することもできます。
https://ja-support.renesas.com/knowledgeBase/17792531

このアプリケーションプロジェクトでは、ターゲットボードや MCU の物理プロパティ（physical property）に加えいくつかの重要なプロパティに対し、必要な操作をサポートするために設定しています。以下の表に、このプロジェクトで設定したそれらのプロパティの値を示します。このアプリケーションプロジェクトを開き、[Properties]（プロパティ）ウィンドウでこれらの設定を表示することもできます。

特定モードの UART0：

表 10　アプリケーションプロジェクトに対応する UART HAL モジュールの設定項目

ISDE のプロパティ	設定値
External RTS Operation（外部 RTS の操作）	Enable（有効）
Name（名前）	g_uart0
Channel（チャネル）	0
Baud Rate（ボーレート）	115200
CTS/RTS Selection（CTS/RTS の選択）	CTS（External RTS Operation（外部 RTS の操作）モードを有効にして 1 本の GPIO 端子を使用する場合、RTS が使用可能になることに注意してください）
Name of UART callback function to be defined by user（ユーザが定義する UART コールバック関数の名前）	user_uart0_callback
Name of UART callback function for the RTS external pin control to be defined by user（ユーザが定義する RTS 外部端子制御に対応する UART コールバック関数の名前）	user_rts_callback
Receive Interrupt Priority（受信割り込みの優先順位）	Priority 2（優先順位 2）
Transmit Interrupt Priority（送信割り込みの優先順位）	Priority 2（優先順位 2）
Transmit End Interrupt Priority（送信終了割り込みの優先順位）	Priority 2（優先順位 2）
Error Interrupt Priority（エラー割り込みの優先順位）	Priority 2（優先順位 2）

汎用モードの UART4：

表 11　アプリケーションプロジェクトに対応する UART HAL モジュールの設定項目

ISDE のプロパティ	設定値
External RTS Operation（外部 RTS の操作）	Enable（有効）
Name（名前）	g_uart4
Channel（チャネル）	4
Baud Rate（ボーレート）	9600
CTS/RTS Selection（CTS/RTS の選択）	RTS (CTS is disabled)（CTS は無効）
Name of UART callback function to be defined by user（ユーザが定義する UART コールバック関数の名前）	user_uart4_callback
Receive Interrupt Priority（受信割り込みの優先順位）	Priority 2（優先順位 2）
Transmit Interrupt Priority（送信割り込みの優先順位）	Priority 2（優先順位 2）
Transmit End Interrupt Priority（送信終了割り込みの優先順位）	Priority 2（優先順位 2）
Error Interrupt Priority（エラー割り込みの優先順位）	Priority 2（優先順位 2）

【注意】アプリケーション内の「#define UartNormal」を使用して、このモードを定義しています。特定モード（specific mode）を使用する場合、この文をコメントアウト（頭の#をとる）してください。

特定のチャネルまたは端子にアクセスするには、SCI 端子を ISDE の[Pins]（端子）タブで設定する必要があります。

以下の表は、[SSP configuration]（SSP 設定）ウィンドウ内での端子の選択方法と、SCI 端子の選択例を示しています。

特定モードの UART0：

表 12　SCI 上での UART HAL モジュールの端子選択シーケンス

Resource（リソース）	ISDE Tab（ISDE タブ）	Pin selection Sequence（端子選択シーケンス）
SCI	Pins（端子）	Select Peripherals > Connectivity: SCI > SCI0

表 13　SCI 上での UART HAL モジュールの端子設定項目

Pin Configuration Property（端子構成のプロパティ）	値
Pin Group Selection（端子グループの選択）	Mixed（組み合わせ）
Operation Mode（動作モード）	Custom（カスタム）
TXD_MOSI	P411
RXD_MISO	P410
SCK, SDA, SCL（SCK、SDA、SCL）	None（なし）
CTS_RTS_SS	P413

【注意】P413 を CTS_RTS_SS に設定するために、GPIO 出力モード（初期値はロー）で使用する P413 のポート設定を無効にする必要があります。

表 14　外部 RTS 端子の端子選択シーケンス

Resource（リソース）	ISDE Tab（ISDE タブ）	Pin selection Sequence（端子選択シーケンス）
GPIO	Pins（端子）	Select Ports > P1 > P100 （[ポート] > [P1] > [P100]を選択）

表 15　外部 RTS 端子の端子設定項目

Pin Configuration Property（端子構成のプロパティ）	値
Mode（モード）	Output mode (Initial Low) （出力モード（初期はロー））
Pull up（プルアップ）	None（なし）
IRQ	None（なし）
Driver Capacity（ドライブ能力）	Low（ロー）
Output type（出力タイプ）	CMOS

【注意】P100 を出力モードに設定するために、SPI0 の動作モード設定を無効にする必要があります。

汎用モード（generic operation）の UART4：

表 16　SCI 上での UART HAL モジュールの端子選択シーケンス

Resource（リソース）	ISDE Tab（ISDE タブ）	Pin selection Sequence（端子選択シーケンス）
SCI	Pins（端子）	Select Peripherals > Connectivity: SCI > SCI4 （周辺装置の選択 > 接続: SCI > SCI4）

表 17　SCI 上での UART HAL モジュールの端子設定項目

Pin Configuration Property（端子構成のプロパティ）	値
Pin Group Selection（端子グループの選択）	Mixed（組み合わせ）
Operation Mode（動作モード）	Asynchronous UART（非同期 UART）
TXD_MOSI	P512
RXD_MISO	P511

【注意】前の設定例は、Synergy S7G2 および SK-S7G2 キットを使用するプロジェクトに対するものです。他の Synergy キットと他の Synergy MCU では使用可能な端子設定項目が異なる可能性があります。端子の使用法については、以下の図を参照してください。

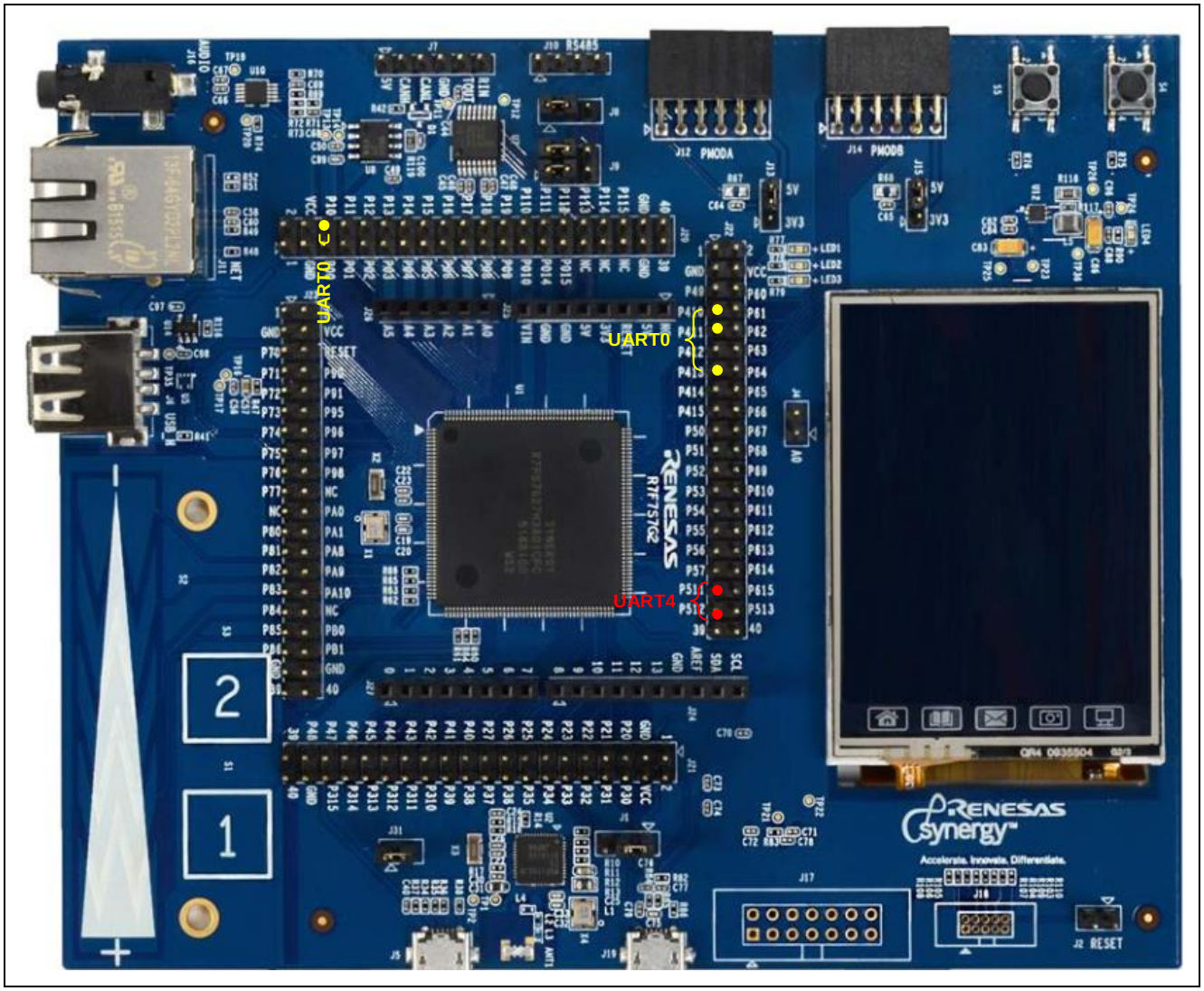

図 8　ハードウェア接続の画像

8.　ターゲットアプリケーションに対応する UART HAL モジュールのカスタマイズ（Customizing the UART HAL Module for a Target Application）

いくつかの設定項目は通常、アプリケーションプロジェクトの値に対し、ユーザが変更を加えます。たとえば、ユーザは[Clock]タブで[PCLKA/PCLKB]を更新する方法により、UART クロックの設定項目を容易に変更することができます。また、ユーザ所望のアナログ入力を選択するために、UART ポートの端子を変更することもできます。この作業を行うには、コンフィギュレータの[Pins]（端子）タブを使用します。SCI をベースとする UART モードで CTS 機能と RTS 機能を同時に使用することはできません。

9. UART HAL モジュールのアプリケーションプロジェクトの実行（Running the UART HAL Module Application Project）

UART HAL モジュールのアプリケーションプロジェクトを実行し、ターゲットキットでその動作を確認するために、本プロジェクトの ISDE へのインポート、コンパイル（compile）、およびデバッグ（debug）を容易に実行することができます。

パッケージ付属のサンプルプロジェクトをインポートして、ビルドしてください。実行する手順については、『Synergy プロジェクトインポートガイド』（下記 WEB）を参照してください。

- 英語版：
  https://www.renesas.com/jp/ja/doc/products/renesas-synergy/apn/r11an0023eu0121-synergy-ssp-import-guide.pdf
- 日本語版（参考資料）：
  https://www.renesas.com/jp/ja/doc/products/renesas-synergy/apn/r11an0023ju0121-synergy-ssp-import-guide.pdf

新しいプロジェクト内で UART HAL モジュールアプリケーションを実装する場合、ターゲットキット上で行う定義、設定、ファイルの自動生成、コードの追加、コンパイル、デバッグは、以下の手順に従います。このガイドに示す手順に従うことで SSP での開発プロセスをより実践的に習得するのに役立ちます。

【注】 Renesas Synergy™ プラットフォーム開発プロセスの基本的な流れを経験したことのあるユーザにとって、以下の手順は十分詳細なものです。これらの手順をまだ理解していない場合、このドキュメントの末尾にある「参考情報」の章に掲載されている『SSP ユーザーズマニュアル』の最初にあるいくつかの章を参照してください。

UART HAL モジュールのアプリケーションプロジェクトを作成し、実行するには、以下の手順に従ってください。

1. UART_HAL という名称で SK-S7G2 ボード（S7G2-BSP）グループに対応する新しいプロジェクトを作成します。
2. プロジェクトを作成するときに、[Project Template Selection]ページで[BSP]を選択します。その後、新しいプロジェクトの設定を完了させます。
3. [Threads]タブ -> [HAL/Common]を選択します。
4. [UART HAL]モジュールを[HAL/Common]スタックに追加します。
5. 割り込みの有効化を含め、パラメータを設定します。
6. [Generate Project Content]ボタンをクリックします。
7. 付属のプロジェクトファイル `art_hal_mg.c`、`uart_hal_mg.h`、および `hal_entry.c` からコードを追加します。
8. プロジェクトをコンパイルします。
9. micro USB ケーブルを使い、ホスト PC と SK-S7G2 キットの J19 を接続します。
10. もう 1 つのボードを使って、SCI の汎用関数（generic function）と特定関数（specific function）を使用して SK-S7G2 との通信を行います。UART0 に対して、RxD、TxD、外部 RTS の各端子（P100）をもう 1 つの SK-S7G2 ボードに接続する必要があります。UART4 に対して接続する必要があるのは、RxD と TxD のみです。
11. プロジェクト内で定義した「UartNormal」を使用して、汎用操作（UART4）または特定操作（UART0）を選択します。
12. アプリケーションのデバッグを開始します。もう 1 つのボードから送信された 16 バイトのデータは SCI に入力されます。すべてのバイトを受信した後、これらのバイトは通信相手に返送されます。
13. セミホスト機能が有効になっている場合、Renesas Debug Virtual Console（Renesas デバッグ仮想コンソール）で出力を確認できます。最初の画像は、UART0（特定操作）の出力値を、2 枚目の画像は UART4（汎用操作）の出力値を示しています。

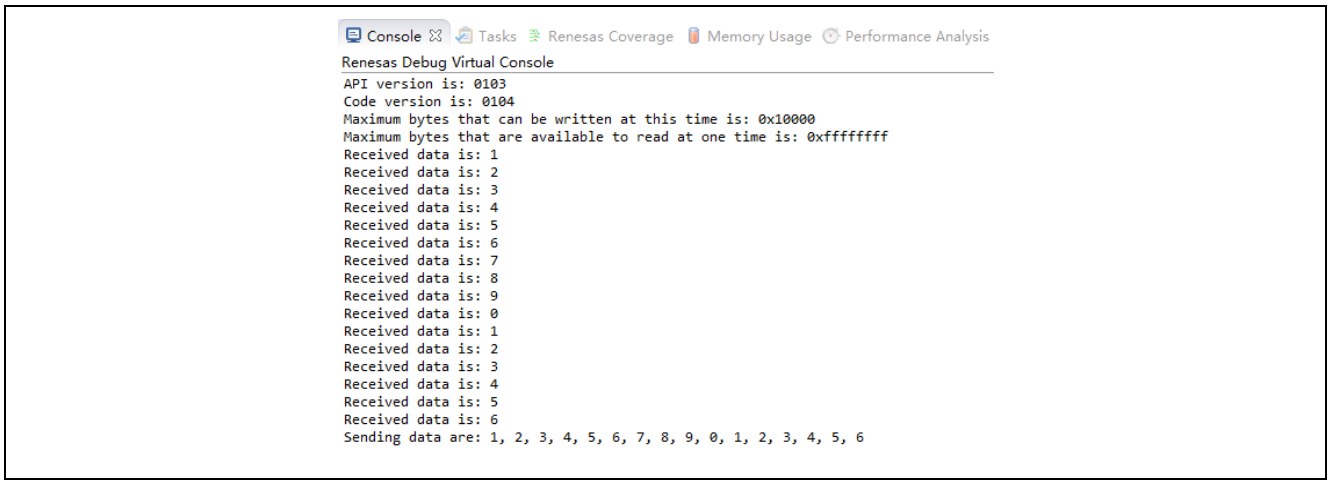

図 9　UART0 のサンプル出力

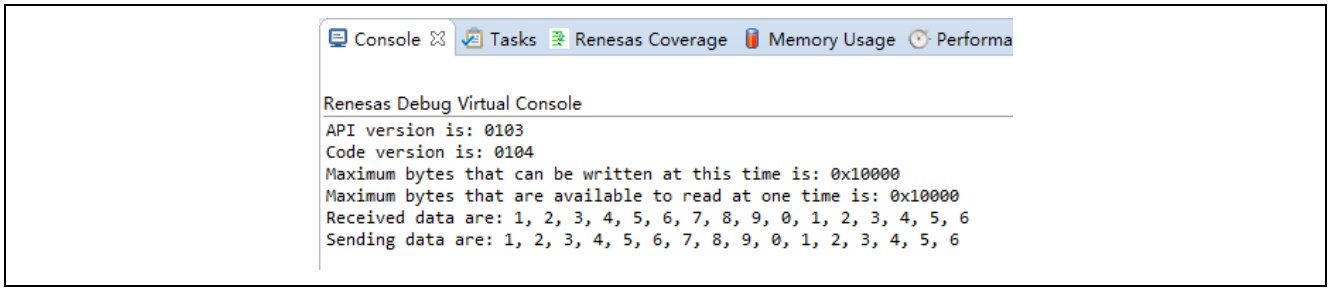

図 10　UART4 のサンプル出力

14. また、出力をオシロスコープで表示することもできます。

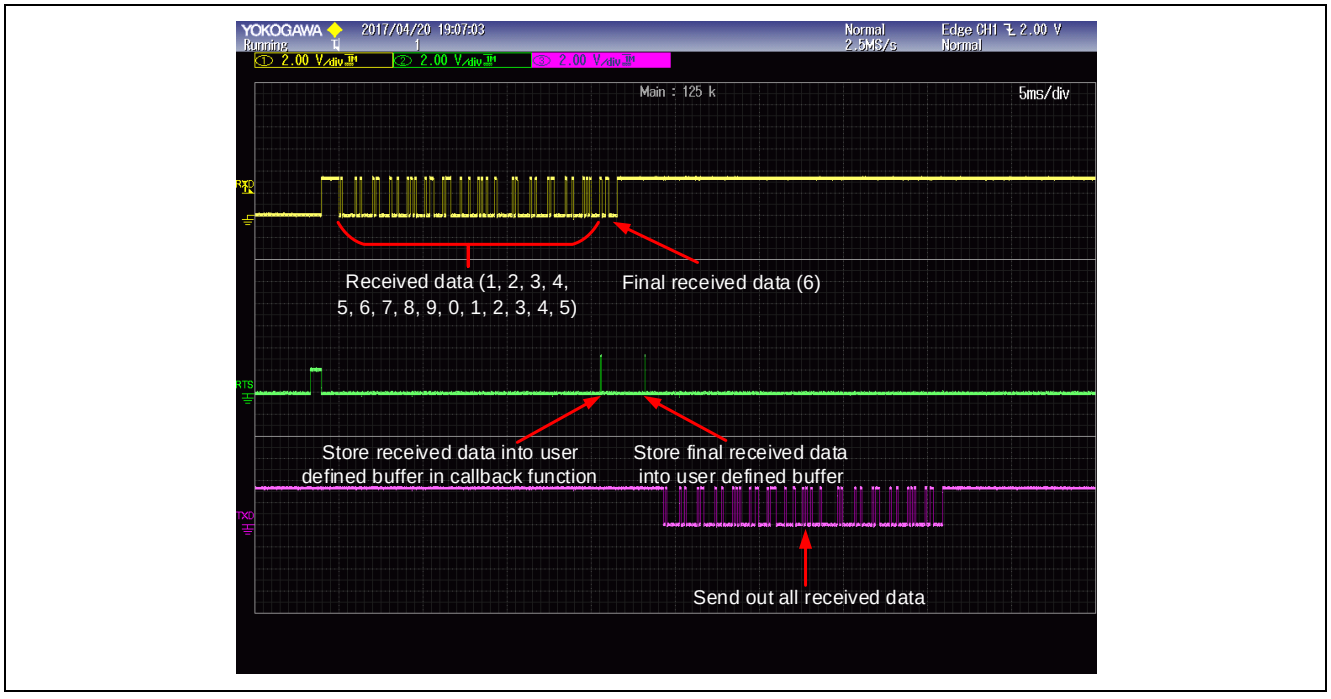

図 11　UART0 の波形（セミホスト機能を使用していない場合）

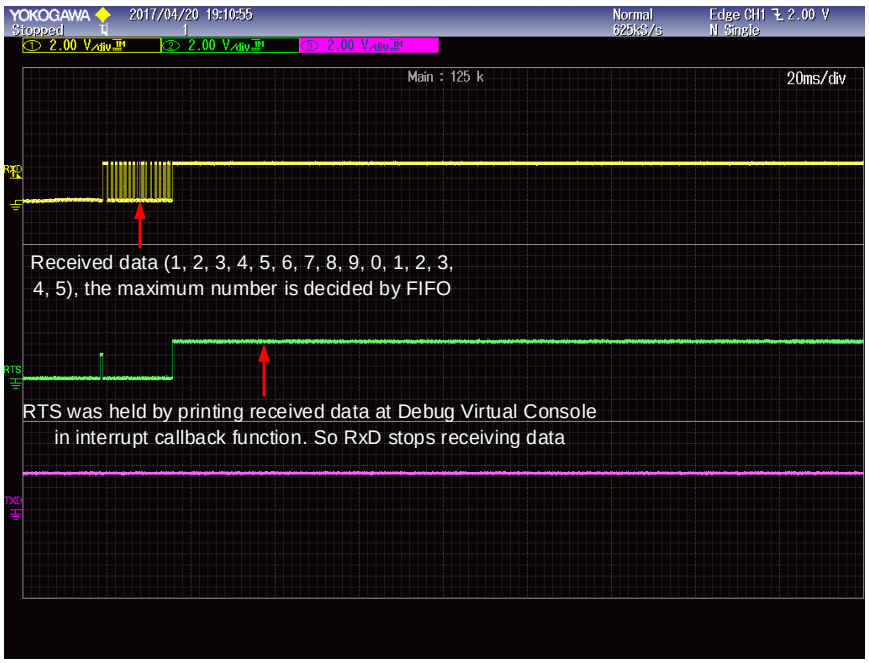

図 12　UART0 の波形（セミホスト機能を使用した場合）—最初のフレーム

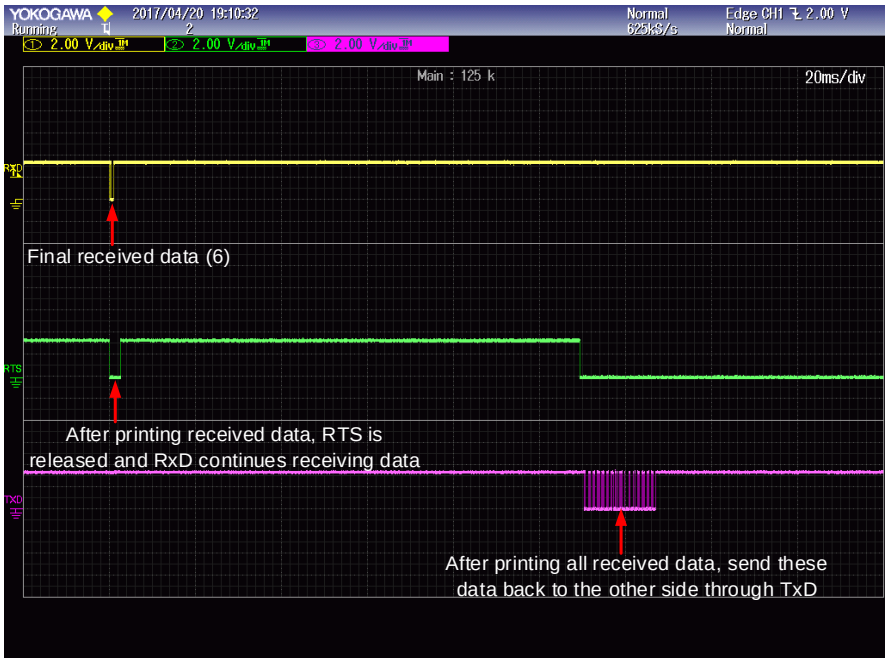

図 13　UART0 の波形（セミホスト機能を使用した場合）—2 番目のフレーム

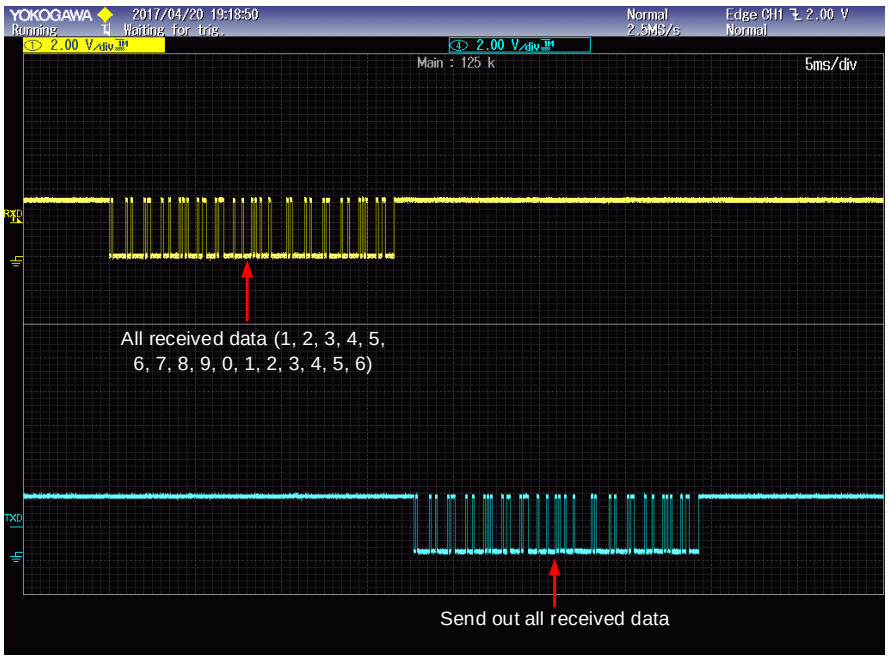

図 14　UART4 の波形（セミホスト機能を使用していない場合）

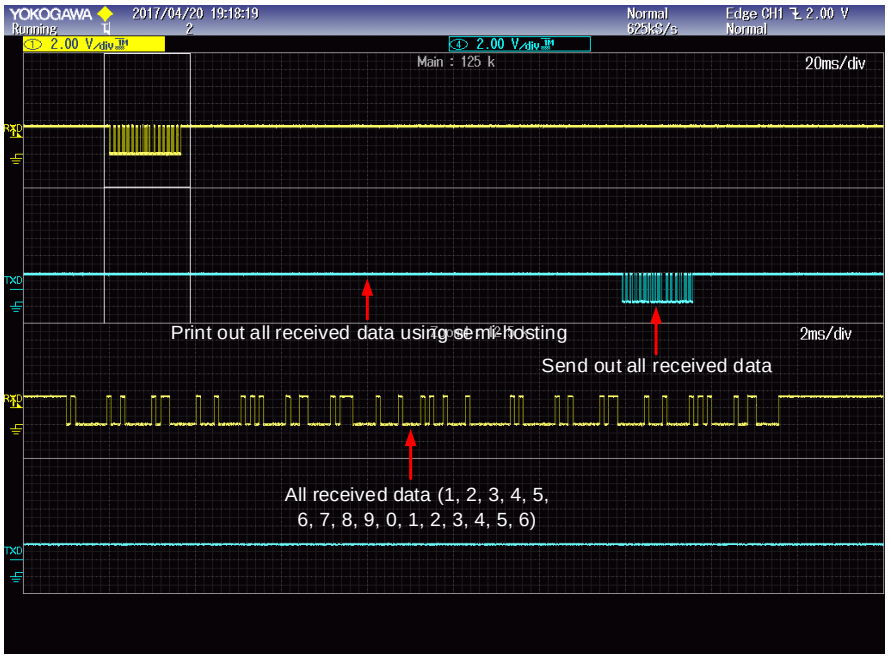

図 15　UART4 の波形（セミホスト機能を使用した場合）

【注意】セミホスト機能を使用して受信データを Renesas Debug Virtual Console（ルネサスデバッグ仮想コンソール）に表示する場合、ISR を呼び出すときに、`user_uart_callback` 関数内で使用する printf 関数が RTS を保持します。セミホスト機能を使用しない場合、ISR に必要な時間が短くなる可能性があります。RTS のハイレベルは、通信先のボードが連続的にデータを送信することを禁止します。

10. UART HAL モジュールのまとめ（UART HAL Module Conclusion）

このモジュールガイドでは、サンプルプロジェクトでモジュールの選択、追加、設定、使用を行うために必要な情報全般を説明しました。従来の組み込みシステムでは、これらの手順を理解することに多くの時間を必要とし、また間違いが起こりやすい操作でした。Renesas Synergy プラットフォームにより、これら手順の所要時間が短くなり、設定項目の競合や、ローレベルドライバの誤った選択など、誤りが防止できるようになりました。アプリケーションプロジェクトで示したように、ハイレベル API を使用することで高いレベルの開発からスタートできるため、ローレベルドライバを作成する時間が不要になり、開発時間を短縮することができます。

11. UART HAL モジュールの次の手順（UART HAL Module Next Steps）

シンプルな UART HAL モジュールのプロジェクトをマスタすればより複雑なサンプルがレビューでき、RS-485 も使用できるようになります。このモジュールガイドには、RS-485 のデモを示す追加のアプリケーションプロジェクトも付属しています。このサンプルは、DK-S7G2 キットを使用します。

DK-S7G2 ボードの SCI モジュール上にある UART は、PORT9 の pin14（P914 端子）、S101、および S102 を使用して、デュアルプロトコルの RS-232／RS-485 トランシーバ上で RS-485 ポートをアクティブにします。

S5 DIP スイッチが実装されているメインボードで、JTAG を ON の位置に、また PBS を ON の位置に設定し、PORT8 の pin7（P807 端子）と pin10（P810 端子）を有効にして、緑と赤の LED を点灯させます。また、S1 を有効にしてマスタをトリガし、データを送信します。

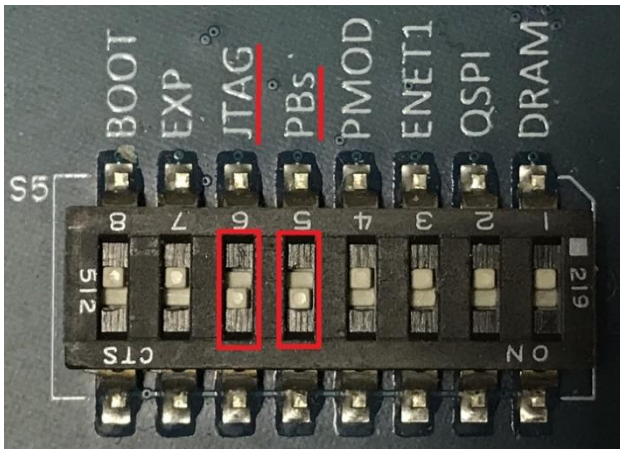

図 16　DIP スイッチ S5 の設定

ベースボードでは、DIP スイッチの S101、つまり RS が ON の位置になっています。DIP スイッチ S102 で、232（RS-232）を OFF の位置に設定します。その後、SLEW と SPB を使用してトランシーバのスルーレート（slew rate）を選択することができます（詳細については、トランシーバのデータシートを参照してください）。S101 と S102 で、他のすべてのスイッチを OFF にします。

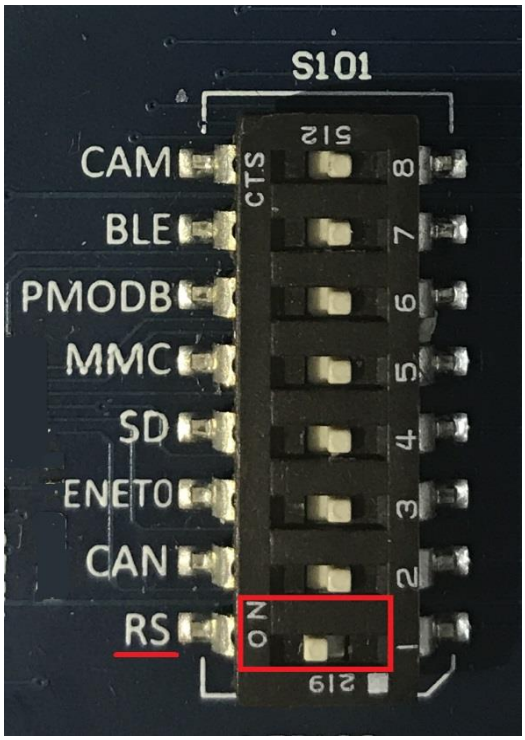

図 17　DIP スイッチ S101 の設定

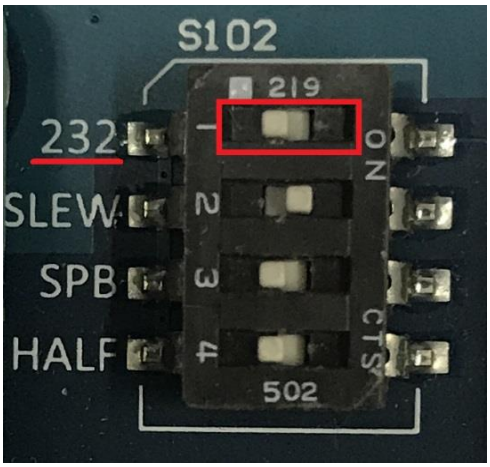

図 18　DIP スイッチ S102 の設定

ベースボード上の J112 は、RS-485 用の出力コネクタです。以下の図に従って、追加のキットに接続します。

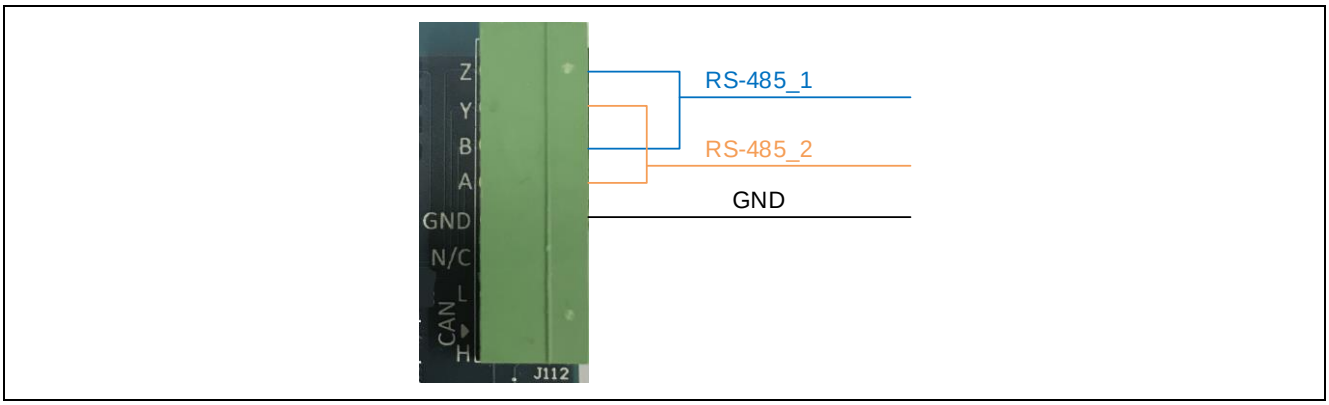

図 19　ベースボード上の J112 に関する接続例

RS-485 通信機能は、SCI1 上で汎用操作モード（generic operation mode）を使用します。トランシーバでアクティブハイドライバ（active high driver）の出力イネーブル（output enable）とアクティブローレシーバ（active low receiver）の出力イネーブルが互いに接続されているほか、P914 を使用して受信と送信の方向を制御します。S1 ボタンをクリックすると、マスタデバイスからのデータ転送がトリガされます。スレーブデバイスがマスタデバイスから正しいデータを受信した時点で、「Received data is right!」（受信データは正しい）が送り返され、LED2 が緑色で点灯します。それ以外の場合、「Received data is wrong!」（受信データは誤り）が送り返され、LED2 が赤く点灯します。

以下の図に、RS-485 アプリケーションプロジェクトのシンプルなフローを示します。

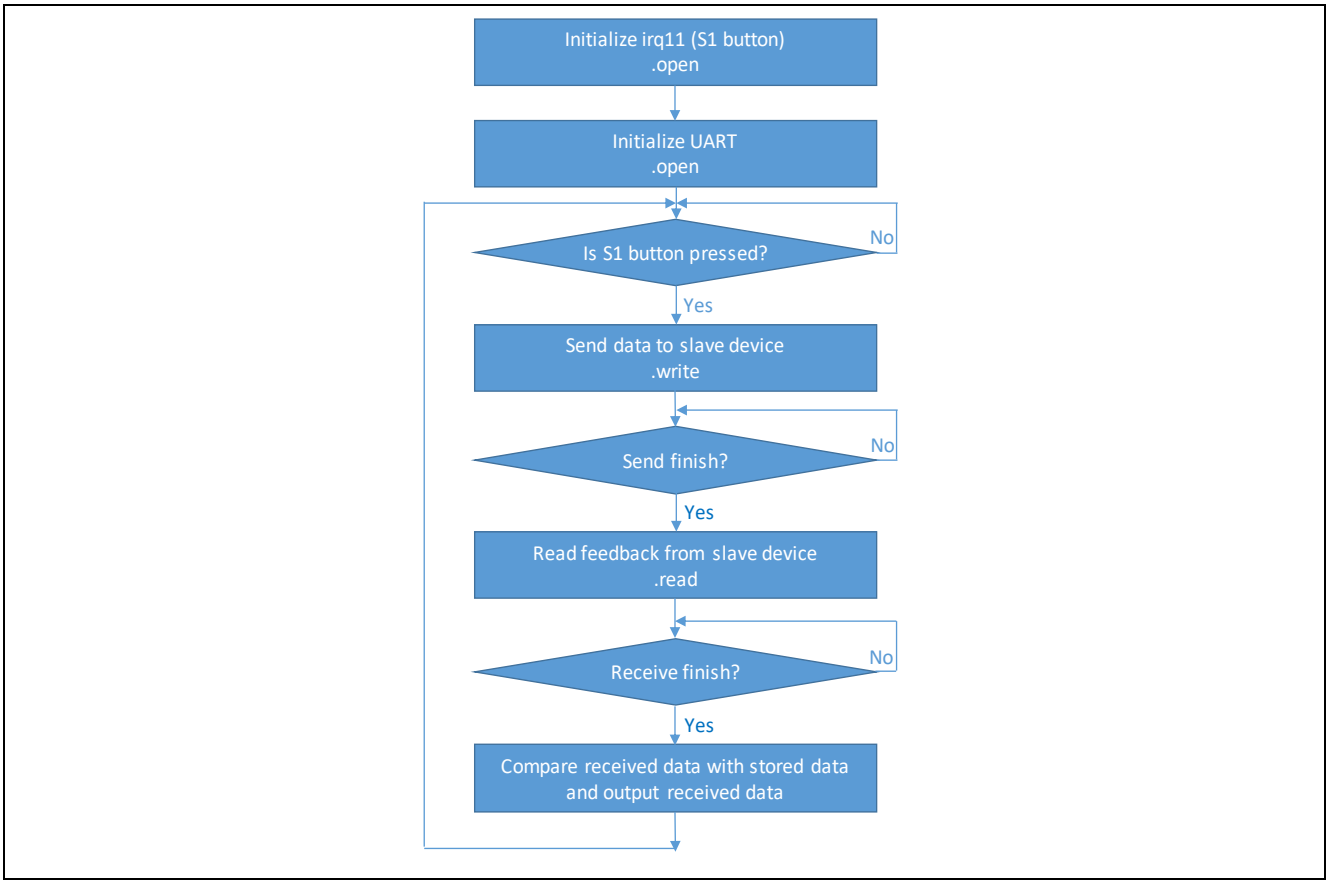

図 20　RS-485 マスタプロジェクトのフロー

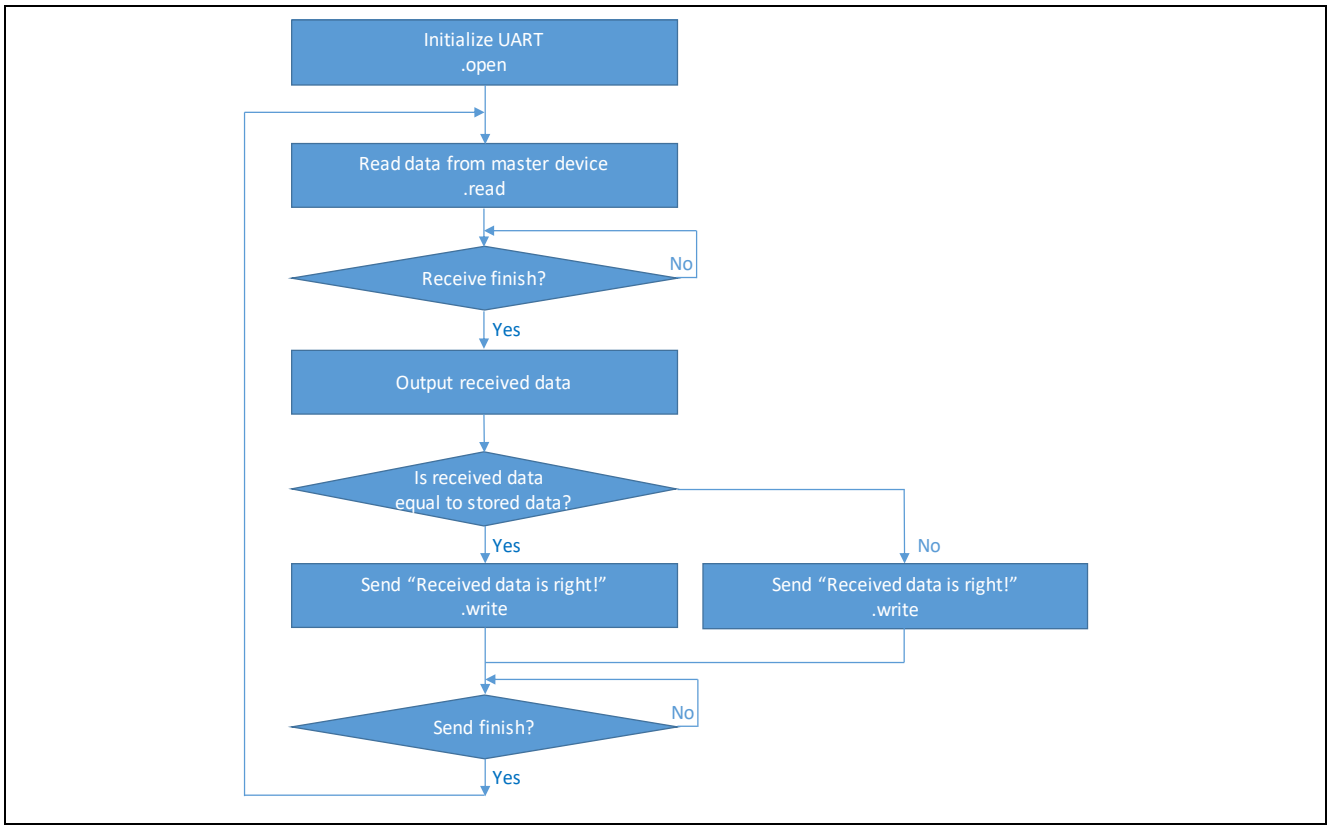

図 21　RS-485 スレーブプロジェクトのフロー

【注意】マスタプロジェクトを使用する場合、`uart_hal_mg_RS485_slave.c` をビルドから除外する必要があります。スレーブの場合も同様です。不要なファイルを除外するには、「*.c」ファイルを右クリックし、[exclude from build]（ビルドから除外）をクリックします。マスタプロジェクトでは、`hal_entry.h` 内の`#define MASTER` を使用します。スレーブプロジェクトをビルドする場合、この行をコメントアウトしてください。

出力は、Renesas Debug Console（Renesas デバッグコンソール）に表示されます。最初の画像はマスタの出力値を、2 枚目の画像はスレーブの出力値を示しています。

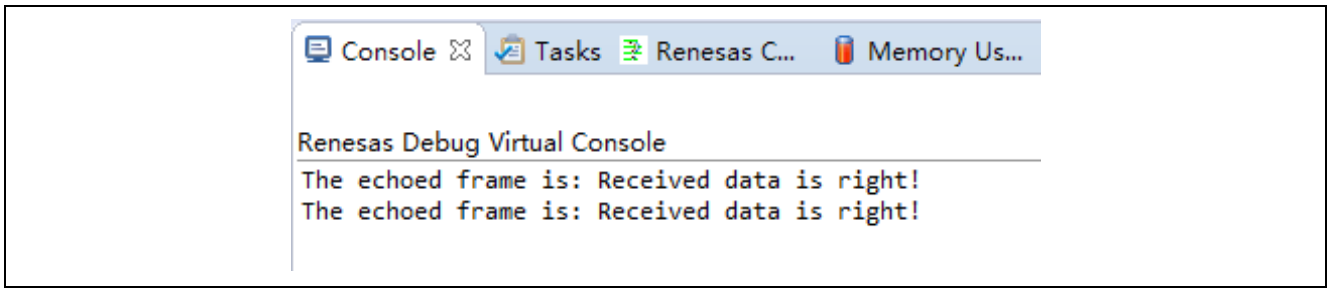

図 22　受信データが正しい場合の RS-485 マスタプロジェクトのサンプル出力

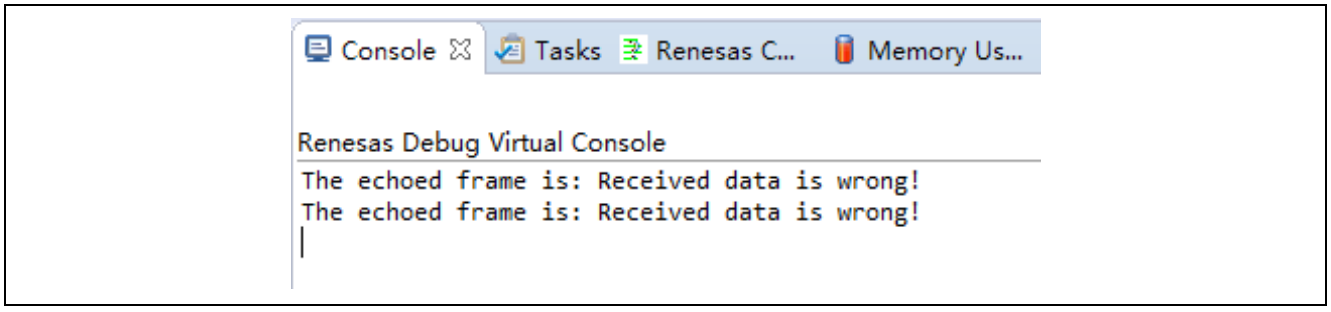

図 23　受信データが誤っている場合の RS-485 マスタプロジェクトのサンプル出力

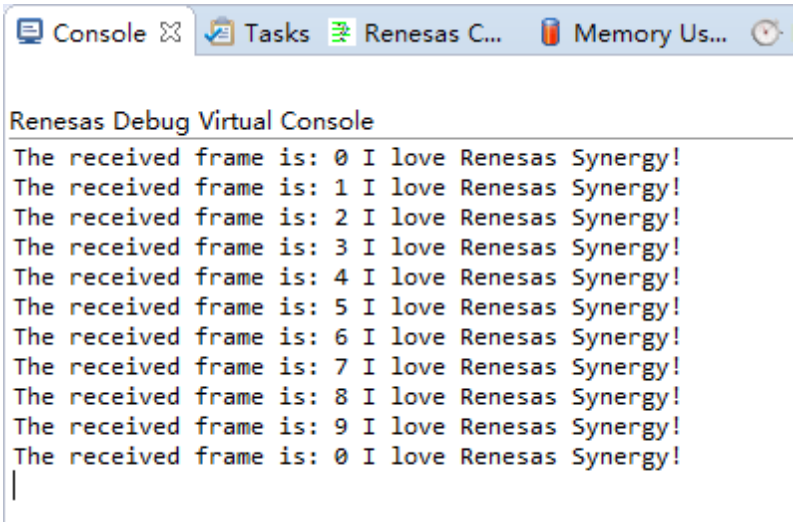

図 24　RS-485 スレーブプロジェクトのサンプル出力

また、出力をオシロスコープで表示することもできます。

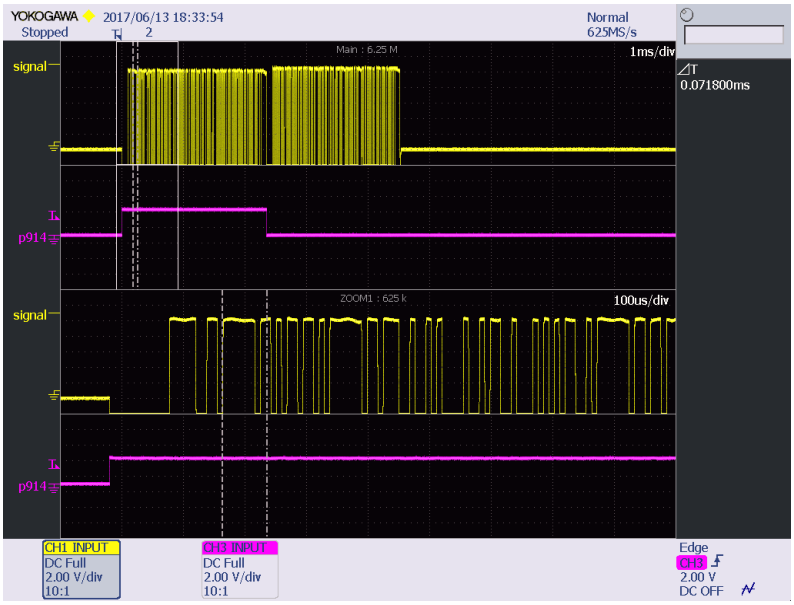

図 25　差動の波形

RS-485 通信以外で、通信フレームワーク（Communications Framework）がアプリケーションに適していると判断されることがあります。通信フレームワークを UART フレームワークインタフェースに実装し、少なくとも 1 個の UART HAL モジュールを使用して UART アプリケーションを作成することもできます。これは、ThreadX RTOS を使用した通信アプリケーション向け汎用 API です。コンソールフレームワーク（Console Framework）は、ThreadX RTOS を使用したコンソールコマンドラインインタフェース（CLI）向け汎用 API です。この実装は、通信インタフェースとして UART を使用することもできます。これらに関する参考情報や他の役立つリソースは、このドキュメントの末尾にある「参考情報」の章に掲載されています。

12. UART HAL モジュールの参考情報（UART HAL Module Reference Information）

『SSP ユーザーズマニュアル』: SSP ディストリビューションパッケージの一部として HTML 形式が入手できるほか、Renesas Synergy™ WEBサイトのSSPページ https://www.renesas.com/jp/ja/products/synergy/software/ssp.htmlから pdf を入手することもできます。

最新版の r_sci_uartモジュールの参考資料やリソースへのリンクは、以下の Synergy WEBサイトから入手できます。

https://www.renesas.com/jp/ja/products/synergy.html

Web サイトおよびサポート

サポート： https://synergygallery.renesas.com/support

テクニカルサポート：

- アメリカ： https://www.renesas.com/en-us/support/contact.html
- ヨーロッパ： https://www.renesas.com/en-eu/support/contact.html
- 日本： https://www.renesas.com/ja-jp/support/contact.html

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.01	2019.09.11	—	• 初版 • 英語版（R11AN0085EU0101 Rev.1.01, 2017.09.15）の巻頭と第 7 章以降を翻訳

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)

本社所在地

〒135-0061 東京都江東区豊洲 3-2-24（豊洲フォレシア）

www.renesas.com

お問合せ窓口

弊社の製品や技術、ドキュメントの最新情報、最寄の営業お問合せ窓口に関する情報などは、弊社ウェブサイトをご覧ください。

www.renesas.com/contact/

商標について

ルネサスおよびルネサスロゴはルネサス エレクトロニクス株式会社の商標です。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。