RL78/G14、RL78/G1C、RL78/L12、RL78/L13、RL78/L1C、RL78/G23 グループ

シリアル・アレイ・ユニットの CSI モードを使ったクロック同期式シングルマスタ制御ソフトウェア

要旨

本アプリケーションノートでは、RL78/G14、RL78/G1C、RL78/L12、RL78/L13、RL78/L1C、RL78/G23 グループ　シリアル・アレイ・ユニット（以下、SAU）の３線シリアル I/O（CSI モード）を使用したクロック同期式シングルマスタ制御方法とサンプルコードの使用方法を説明します。

ポート制御による SPI スレーブデバイスセレクト制御を付加することにより、SPI モード・シングルマスタ制御が可能です。

なお、本サンプルコードは、スレーブデバイスとしての SPI デバイスを制御するための下位層に位置するソフトウェアです。

別途、スレーブデバイス制御のための上位層に位置するソフトウェアを用意していますので、以下の URL から入手してください。なお、スレーブデバイス制御ソフトウェアが追加になった場合、本アプリケーションノートの更新が間に合わないことがあります。最新のスレーブデバイス制御ソフトウェアとの組み合わせ情報は、以下の URL を参照してください。

- SPI シリアル EEPROM 制御
  SPI シリアル EEPROM ドライバ | Renesas
- SPI/QSPI シリアルフラッシュメモリ制御、QSPI シリアル相変化メモリ制御
  SPI/QSPI シリアルフラッシュメモリ・QSPI シリアル相変化メモリドライバ | Renesas
- SPI モードマルチメディアカードドライバ: 導入ガイド
  RL78 ファミリ SPI モードマルチメディアカードドライバ: 導入ガイド (renesas.com)

動作確認デバイス

対応 MCU　RL78/G14、RL78/G1C グループ

RL78/L12、RL78/L13、RL78/L1C グループ

RL78/G23 グループ

動作確認に使用したデバイス

ルネサス エレクトロニクス製 R1EX25xxx シリーズ SPI Serial EEPROM

Macronix International Co., Ltd 社製 MX25/66L family serial NOR Flash memory

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合わせて変更し、十分評価してください。

なお、以降では、対象デバイスが、複数グループ MCU であるため、説明の都合上、“RL78 ファミリ MCU”として記述しています。

目次

1. 仕様

RL78 ファミリ MCU のシリアル・アレイ・ユニット（SAU）の３線シリアル I/O（CSI モード）を使用し、クロック同期式制御を行います。ポート制御による SPI スレーブデバイスセレクト制御を付加することにより、SPI モード・シングルマスタ制御が可能です。

表 1-1 に使用する周辺機器と用途を、図 1-1 に使用例を示します。

以下に、機能概略を示します。

- マスタデバイスを RL78 ファミリ MCU とし、SAU の３線シリアル I/O（CSI モード）を使ったクロック同期式シングルマスタ用ブロック型デバイスドライバです。
- MCU 内蔵のクロック同期式（３線式）シリアル通信機能を使用します。また、ユーザ設定した 1 チャネルの使用が可能です。複数チャネルの使用は、できません。
- 本サンプルコードは、チップセレクト制御をサポートしていません。SPI デバイスを制御する場合、別途、デバイスセレクト制御を組み込む必要があります。
- MSB ファースト転送をサポートしています。
- CPU 転送のみをサポートしています。DMAC 転送をサポートしていません。
- 割り込みによる転送起動をサポートしていません。

表 1-1　使用する周辺機器と用途

周辺機器	用途
SAU	クロック同期式（３線式）シリアル 1ch（必須）
Port	SPI スレーブデバイスセレクト制御信号用 使用デバイス数分のポートが必要（必須） ただし、本サンプルコードでは、扱いません。

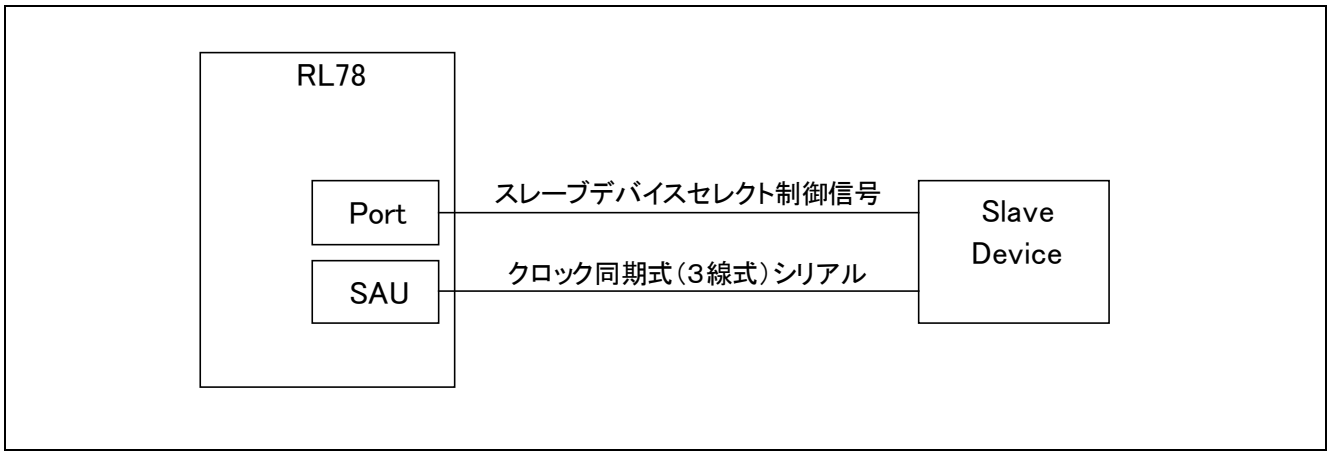


図 1-1　使用例

2. 動作確認条件

本アプリケーションノートのサンプルコードは、以下の動作条件で動作を確認しています。

(1) RL78/G14 SAU 統合開発環境 CS+ for CA,CX（コンパイラ：CA78K0R）

表 2-1 動作確認条件

項目	内容
評価に使用したマイコン	RL78/G14（プログラム ROM 256KB / RAM 24KB）
評価に使用したメモリ	ルネサス エレクトロニクス製 R1EX25xxx シリーズ SPI Serial EEPROM
動作周波数	メイン・システム・クロック：32MHz 周辺ハードウェア・クロック：32MHz シリアル・クロック：4MHz
動作電圧	3.3V
統合開発環境	ルネサス エレクトロニクス製 CS+ for CA,CX V3.01.00
Cコンパイラ、アセンブラ	ルネサス エレクトロニクス製 RL78,78K0R コンパイラ CA78K0R V1.71
	コンパイルオプション 統合開発環境のデフォルト設定（"-qx2"）を使用しています。
サンプルコードのバージョン	Ver.2.05
評価に使用したソフトウェア	RX ファミリ、RL78 ファミリ、78K0R/Kx3-L Renesas R1EX25xxx シリーズ Serial EEPROM 制御ソフトウェア(R01AN0565JJ) Ver.2.04
評価に使用したボード	Renesas Starter Kit for RL78/G14

(2) RL78/G14 SAU 統合開発環境 CS+ for CC（コンパイラ：CC-RL）

表 2-2 動作確認条件

項目	内容
評価に使用したマイコン	RL78/G14（プログラム ROM 256KB / RAM 24KB）
評価に使用したメモリ	ルネサス エレクトロニクス製 R1EX25xxx シリーズ SPI Serial EEPROM
動作周波数	メイン・システム・クロック：32MHz 周辺ハードウェア・クロック：32MHz シリアル・クロック：4MHz
動作電圧	3.3V
統合開発環境	ルネサス エレクトロニクス製 CS+ for CC V3.03.00
Cコンパイラ、アセンブラ	ルネサス エレクトロニクス製 RL78 コンパイラ CC-RL V1.02.00
	コンパイルオプション 統合開発環境のデフォルト設定（既定の最適化を行う(なし)）を使用しています。
サンプルコードのバージョン	Ver.2.05
評価に使用したソフトウェア	RX ファミリ、RL78 ファミリ、78K0R/Kx3-L Renesas R1EX25xxx シリーズ Serial EEPROM 制御ソフトウェア(R01AN0565JJ) Ver.2.04
評価に使用したボード	Renesas Starter Kit for RL78/G14

(3) RL78/G14 SAU 統合開発環境 IAR Embedded Workbench の場合

表 2-3 動作確認条件

項目	内容
評価に使用したマイコン	RL78/G14（プログラム ROM 256KB / RAM 24KB）
評価に使用したメモリ	ルネサス エレクトロニクス製 R1EX25xxx シリーズ SPI Serial EEPROM
動作周波数	メイン・システム・クロック：32MHz 周辺ハードウェア・クロック：32MHz シリアル・クロック：4MHz
動作電圧	3.3V
統合開発環境	IAR Systems 製 IAR Embedded Workbench for Renesas RL78（Ver.1.30.2）
Cコンパイラ	IAR Systems 製 IAR Assembler for Renesas RL78 (Ver.1.30.2.50666) IAR C/C++ Compiler for Renesas RL78（Ver.1.30.2.50666）
	コンパイルオプション 統合開発環境のデフォルト設定（"レベル 低"）を使用しています。
サンプルコードのバージョン	Ver.2.03
評価に使用したソフトウェア	ルネサス エレクトロニクス製 R1EX25xxx シリーズの SPI Serial EEPROM 制御ソフトウェア(R01AN0565JJ) Ver.2.03
評価に使用したボード	Renesas Starter Kit for RL78/G14

(4) RL78/G1C SAU 統合開発環境 CubeSuite+の場合

表 2-4 動作確認条件

項目	内容
評価に使用したマイコン	RL78/G1C（プログラム ROM 32KB / RAM 5.5KB）
評価に使用したメモリ	ルネサス エレクトロニクス製 R1EX25xxx シリーズ SPI Serial EEPROM
動作周波数	メイン・システム・クロック：24MHz 周辺ハードウェア・クロック：24MHz シリアル・クロック：4MHz
動作電圧	3.3V
統合開発環境	ルネサス エレクトロニクス製 CubeSuite+ V2.01.00
Cコンパイラ、アセンブラ	ルネサス エレクトロニクス製 RL78,78K0R コンパイラ CA78K0R V1.70
	コンパイルオプション 統合開発環境のデフォルト設定（"-qx2"）を使用しています。
サンプルコードのバージョン	Ver.2.03
評価に使用したソフトウェア	ルネサス エレクトロニクス製 R1EX25xxx シリーズの SPI Serial EEPROM 制御ソフトウェア(R01AN0565JJ) Ver.2.03.R01
評価に使用したボード	Renesas RL78/G1C ターゲット・ボード QB-R5F10JGC-TB

(5) RL78/G1C SAU 統合開発環境 IAR Embedded Workbench の場合

表 2-5 動作確認条件

項目	内容
評価に使用したマイコン	RL78/G1C（プログラム ROM 32KB / RAM 5.5KB）
評価に使用したメモリ	ルネサス エレクトロニクス製 R1EX25xxx シリーズ SPI Serial EEPROM
動作周波数	メイン・システム・クロック：24MHz 周辺ハードウェア・クロック：24MHz シリアル・クロック：4MHz
動作電圧	3.3V
統合開発環境	IAR Systems 製 IAR Embedded Workbench for Renesas RL78 (Ver.1.30.5)
Cコンパイラ	IAR Systems 製 IAR Assembler for Renesas RL78 (Ver.1.30.4.50715) IAR C/C++ Compiler for Renesas RL78 (Ver.1.30.5.50715)
	コンパイルオプション 統合開発環境のデフォルト設定（"レベル 低"）を使用しています。
サンプルコードのバージョン	Ver.2.03
評価に使用したソフトウェア	ルネサス エレクトロニクス製 R1EX25xxx シリーズの SPI Serial EEPROM 制御ソフトウェア(R01AN0565JJ) Ver.2.03.R01
評価に使用したボード	Renesas RL78/G1C ターゲット・ボード QB-R5F10JGC-TB

(6) RL78/L12 SAU 統合開発環境 CubeSuite+の場合

表 2-6 動作確認条件

項目	内容
評価に使用したマイコン	RL78/L12（プログラム ROM 32KB / RAM 1.5KB）
評価に使用したメモリ	ルネサス エレクトロニクス製 R1EX25xxx シリーズ SPI Serial EEPROM
動作周波数	メイン・システム・クロック：24MHz 周辺ハードウェア・クロック：24MHz シリアル・クロック：4MHz
動作電圧	3.3V
統合開発環境	ルネサス エレクトロニクス製 CubeSuite+ V2.01.00
Cコンパイラ、アセンブラ	ルネサス エレクトロニクス製 RL78,78K0R コンパイラ CA78K0R V1.70
	コンパイルオプション 統合開発環境のデフォルト設定（"-qx2"）を使用しています。
サンプルコードのバージョン	Ver.2.03
評価に使用したソフトウェア	ルネサス エレクトロニクス製 R1EX25xxx シリーズの SPI Serial EEPROM 制御ソフトウェア(R01AN0565JJ) Ver.2.03.R01
評価に使用したボード	Renesas Starter Kit for RL78/L12

(7) RL78/L12 SAU 統合開発環境 IAR Embedded Workbench の場合

表 2-7 動作確認条件

項目	内容
評価に使用したマイコン	RL78/L12（プログラム ROM 32KB / RAM 1.5KB）
評価に使用したメモリ	ルネサス エレクトロニクス製 R1EX25xxx シリーズ SPI Serial EEPROM
動作周波数	メイン・システム・クロック：24MHz 周辺ハードウェア・クロック：24MHz シリアル・クロック：4MHz
動作電圧	3.3V
統合開発環境	IAR Systems 製 IAR Embedded Workbench for Renesas RL78 (Ver.1.30.5)
Cコンパイラ	IAR Systems 製 IAR Assembler for Renesas RL78 (Ver.1.30.4.50715) IAR C/C++ Compiler for Renesas RL78 (Ver.1.30.5.50715)
	コンパイルオプション 統合開発環境のデフォルト設定（"レベル 低"）を使用しています。
サンプルコードのバージョン	Ver.2.03
評価に使用したソフトウェア	ルネサス エレクトロニクス製 R1EX25xxx シリーズの SPI Serial EEPROM 制御ソフトウェア(R01AN0565JJ) Ver.2.03.R01
評価に使用したボード	Renesas Starter Kit for RL78/L12

(8) RL78/L13 SAU 統合開発環境 CubeSuite+の場合

表 2-8 動作確認条件

項目	内容
評価に使用したマイコン	RL78/L13（プログラム ROM 128KB / RAM 8KB）
評価に使用したメモリ	ルネサス エレクトロニクス製 R1EX25xxx シリーズ SPI Serial EEPROM
動作周波数	メイン・システム・クロック：24MHz 周辺ハードウェア・クロック：24MHz シリアル・クロック：4MHz
動作電圧	3.3V
統合開発環境	ルネサス エレクトロニクス製 CubeSuite+ V2.01.00
Cコンパイラ、アセンブラ	ルネサス エレクトロニクス製 RL78,78K0R コンパイラ CA78K0R V1.70
	コンパイルオプション 統合開発環境のデフォルト設定（"-qx2"）を使用しています。
サンプルコードのバージョン	Ver.2.03
評価に使用したソフトウェア	ルネサス エレクトロニクス製 R1EX25xxx シリーズの SPI Serial EEPROM 制御ソフトウェア(R01AN0565JJ) Ver.2.03.R01
評価に使用したボード	Renesas Starter Kit for RL78/L13

(9) RL78/L13 SAU 統合開発環境 IAR Embedded Workbench の場合

表 2-9 動作確認条件

項目	内容
評価に使用したマイコン	RL78/L13（プログラム ROM 128KB / RAM 8KB）
評価に使用したメモリ	ルネサス エレクトロニクス製 R1EX25xxx シリーズ SPI Serial EEPROM
動作周波数	メイン・システム・クロック：24MHz 周辺ハードウェア・クロック：24MHz シリアル・クロック：4MHz
動作電圧	3.3V
統合開発環境	IAR Systems 製 IAR Embedded Workbench for Renesas RL78 (Ver.1.30.5)
Cコンパイラ	IAR Systems 製 IAR Assembler for Renesas RL78 (Ver.1.30.4.50715) IAR C/C++ Compiler for Renesas RL78 (Ver.1.30.5.50715)
	コンパイルオプション 統合開発環境のデフォルト設定（"レベル 低"）を使用しています。
サンプルコードのバージョン	Ver.2.03
評価に使用したソフトウェア	ルネサス エレクトロニクス製 R1EX25xxx シリーズの SPI Serial EEPROM 制御ソフトウェア(R01AN0565JJ) Ver.2.03.R01
評価に使用したボード	Renesas Starter Kit for RL78/L13

(10) RL78/L1C SAU 統合開発環境 CubeSuite+の場合

表 2-10 動作確認条件

項目	内容
評価に使用したマイコン	RL78/L1C（プログラム ROM 256KB / RAM 16KB）
評価に使用したメモリ	ルネサス エレクトロニクス製 R1EX25xxx シリーズ SPI Serial EEPROM
動作周波数	メイン・システム・クロック：24MHz 周辺ハードウェア・クロック：24MHz シリアル・クロック：4MHz
動作電圧	3.3V
統合開発環境	ルネサス エレクトロニクス製 CubeSuite+ V2.01.00
Cコンパイラ、アセンブラ	ルネサス エレクトロニクス製 RL78,78K0R コンパイラ CA78K0R V1.70
	コンパイルオプション 統合開発環境のデフォルト設定（"-qx2"）を使用しています。
サンプルコードのバージョン	Ver.2.03
評価に使用したソフトウェア	ルネサス エレクトロニクス製 R1EX25xxx シリーズの SPI Serial EEPROM 制御ソフトウェア(R01AN0565JJ) Ver.2.03.R01
評価に使用したボード	Renesas Starter Kit for RL78/L1C

(11) RL78/L1C SAU 統合開発環境 IAR Embedded Workbench の場合

表 2-11　動作確認条件

項目	内容
評価に使用したマイコン	RL78/L1C（プログラム ROM 256KB / RAM 16KB）
評価に使用したメモリ	ルネサス エレクトロニクス製 R1EX25xxx シリーズ SPI Serial EEPROM
動作周波数	メイン・システム・クロック：24MHz 周辺ハードウェア・クロック：24MHz シリアル・クロック：4MHz
動作電圧	3.3V
統合開発環境	IAR Systems 製 IAR Embedded Workbench for Renesas RL78 (Ver.1.30.5)
Cコンパイラ	IAR Systems 製 IAR Assembler for Renesas RL78 (Ver.1.30.4.50715) IAR C/C++ Compiler for Renesas RL78 (Ver.1.30.5.50715)
	コンパイルオプション 統合開発環境のデフォルト設定（"レベル 低"）を使用しています。
サンプルコードのバージョン	Ver.2.03
評価に使用したソフトウェア	ルネサス エレクトロニクス製 R1EX25xxx シリーズの SPI Serial EEPROM 制御ソフトウェア(R01AN0565JJ) Ver.2.03.R01
評価に使用したボード	Renesas Starter Kit for RL78/L1C

(12) RL78/G23 SAU 統合開発環境 CS+ for CC (コンパイラ：CC-RL)

表 2-12 動作確認条件

項目	内容
評価に使用したマイコン	RL78/G23（プログラム ROM 128KB / RAM 16KB）
評価に使用したメモリ	Micronix International Co., Ltd.社製 MX25/66L family serial NOR Flash memory
動作周波数	メイン・システム・クロック：32MHz 周辺ハードウェア・クロック：32MHz シリアル・クロック：8MHz
動作電圧	3.3V
統合開発環境	ルネサス エレクトロニクス製 CS+ for CC V8.05.00
C コンパイラ、アセンブラ	ルネサス エレクトロニクス製 RL78 コンパイラ CC-RL V1.10.00
	コンパイルオプション 統合開発環境のデフォルト設定（既定の最適化を行う(なし)）を使用しています。
評価に使用したソフトウェア	RX ファミリ、RL78 ファミリ、78K0R/Kx3-L Macronix International 社製 MX25/66L family serial NOR Flash Memory 制御ソフトウェア(R01AN1967JJ)
評価に使用したボード	RL78/G23-64p Fast Prototyping Board

(13) RL78/G23 SAU 統合開発環境 IAR Embedded Workbench の場合

表 2-13 動作確認条件

項目	内容
評価に使用したマイコン	RL78/G23（プログラム ROM 128KB / RAM 16KB）
評価に使用したメモリ	Micronix International Co., Ltd.社製 MX25/66L family serial NOR Flash memory
動作周波数	メイン・システム・クロック：32MHz 周辺ハードウェア・クロック：32MHz シリアル・クロック：8MHz
動作電圧	3.3V
統合開発環境	IAR Systems 製 IAR Embedded Workbench for Renesas RL78（Ver.4.21.1.2409）
C コンパイラ	IAR Systems 製 IAR Assembler for Renesas RL78（Ver.4.21.1.2409） IAR C/C++ Compiler for Renesas RL78（Ver. 4.21.1.2409）
	コンパイルオプション 統合開発環境のデフォルト設定（”レベル 低”）を使用しています。
評価に使用したソフトウェア	RX ファミリ、RL78 ファミリ、78K0R/Kx3-L Macronix International 社製 MX25/66L family serial NOR Flash Memory 制御ソフトウェア(R01AN1967JJ)
評価に使用したボード	RL78/G23-64p Fast Prototyping Board

(14) RL78/G23 SAU 統合開発環境 e2 studio の場合（コンパイラ：LLVM）

表 2-14 動作確認条件

項目	内容
評価に使用したマイコン	RL78/G23（プログラム ROM 128KB / RAM 16KB）
評価に使用したメモリ	Micronix International Co., Ltd.社製 MX25/66L family serial NOR Flash memory
動作周波数	メイン・システム・クロック：32MHz 周辺ハードウェア・クロック：32MHz シリアル・クロック：8MHz
動作電圧	3.3V
統合開発環境	ルネサス エレクトロニクス製 e2 studio 22.4.0.R20220331-2313
C コンパイラ	オープンソースコンパイラ LLVM for Renesas RL78 10.0.0.202203
	コンパイルオプション 統合開発環境のサイズ設定（”Optimize size (-Os)”）を使用しています。
評価に使用したソフトウェア	RX ファミリ、RL78 ファミリ、78K0R/Kx3-L Macronix International 社製 MX25/66L family serial NOR Flash Memory 制御ソフトウェア(R01AN1967JJ)
評価に使用したボード	RL78/G23-64p Fast Prototyping Board

3. 関連アプリケーションノート

本アプリケーションノートに関連するアプリケーションノートを以下に示します。合わせて参照してください。

- Renesas R1EX25xxx シリーズ Serial EEPROM 制御ソフトウェア（R01AN0565JJ）
- Micron Technology 社製 M25P シリーズ Serial Flash memory 制御ソフトウェア（R01AN0566JJ）
- Micron Technology 社製 M45PE シリーズ Serial Flash memory 制御ソフトウェア（R01AN0567JJ）
- Micron Technology 社製 P5Q Serial Phase Change Memory 制御ソフトウェア（R01AN1439JJ）
- Micron Technology 社製 N25Q Serial NOR Flash Memory 制御ソフトウェア（R01AN1528JJ）
- Spansion 社製 S25FLxxxS MirrorBit® Flash Non-Volatile Memory 制御ソフトウェア（R01AN1529JJ）
- Macronix International 社製 MX25/66L family serial NOR Flash Memory 制御ソフトウェア（R01AN1967JJ）

4. ハードウェア説明

4.1 使用端子一覧

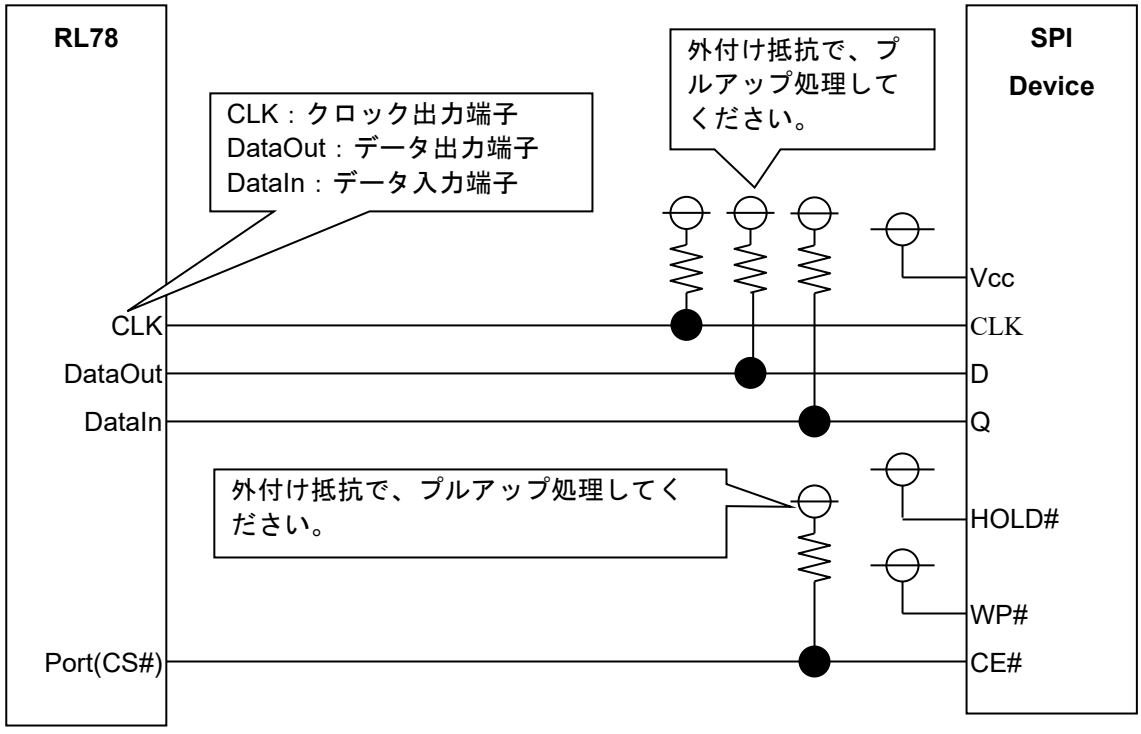
表 4-1 に、使用端子と機能を示します。

表 4-1 使用端子と機能

端子名	入出力	内容
SCK （図 4-1 の CLK）	出力	クロック出力
SO （図 4-1 の DataOut）	出力	マスタデータ出力
SI （図 4-1 の DataIn）	入力	マスタデータ入力
Port （図 4-1 の Port(CS#)）	出力	スレーブデバイスセレクト出力 ただし、本サンプルコードでは、扱いません。

4.2 参考回路

図 4-1 に接続図を示します。



・MCU 上のシリアル I/O に使用される端子名は、MCU に依存します。
・本アプリケーションノートでは、サンプルコードの記述に合わせて、CLK 端子、DataIn 端子、DataOut 端子、Port(CS#)端子として、表します。

図 4-1　RL78 ファミリ MCU シリアル・アレイ・ユニットと SPI スレーブデバイスの接続例

5. ソフトウェア説明

5.1 動作説明

SAU の３線シリアル I/O（CSI モード）を使って、クロック同期式シングルマスタ制御を実現します。

本サンプルコードでは、以下の制御を行っています。

- データの入出力を、クロック同期式モード（内部クロック使用）で、制御する。

本サンプルコードは、以下のように、デバイス上のデータのバイトオフセット値と、転送元／先のメモリのバイトオフセット値が合致するようにしたものです。

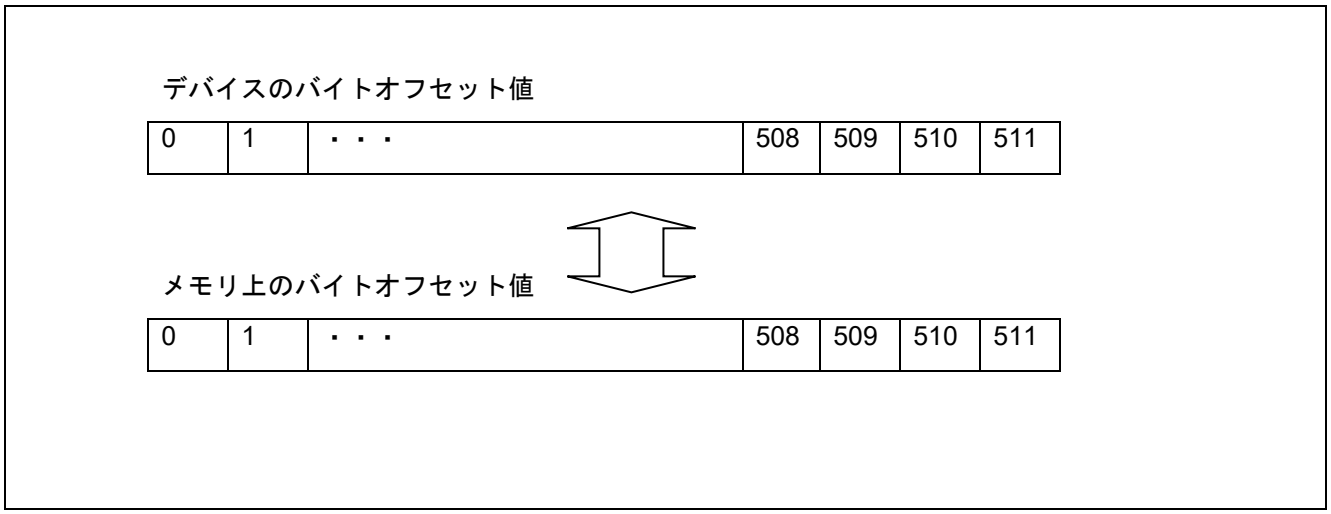
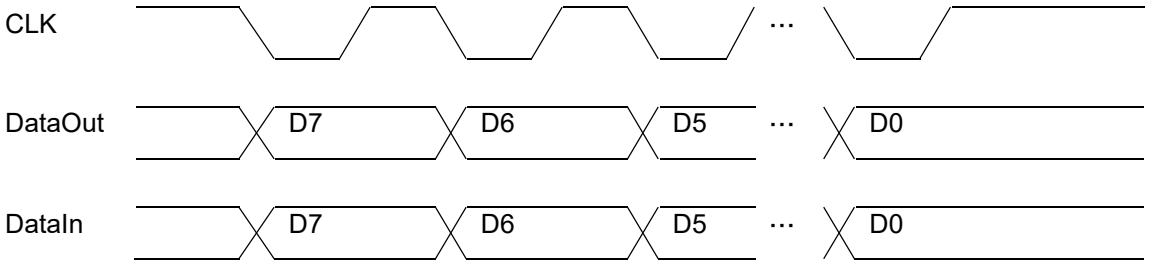


図 5-1 転送データの格納

5.1.1　クロック同期式モードで発生させるタイミング

SPI スレーブデバイス制御のため、図 5-2 に示す SPI モード 3（CPOL=1、CPHA=1）のタイミングを発生します。そのため、本 MCU では、シリアル通信動作設定レジスタ（SCRmn）の位相選択ビット（DAPmn、CKPmn）をタイプ 1（DAPmn=0、CKPmn=0）に設定します。



・MCU-> SPI スレーブデバイスの送信時：転送クロックの立ち下がりで送信データ出力開始
・SPI スレーブデバイス->MCU の受信時：転送クロックの立ち上がりで受信データの入力取り込み
・MSB ファーストでの転送

転送を行っていないときの CLK 端子のレベルは、”H”です。

図 5-2　クロック同期式モード　タイミング設定

使用可能なシリアル・クロック周波数は、MCU および SPI スレーブデバイスのデータシートで、確認してください。

5.1.2　SPI スレーブデバイスの CE#端子制御

SPI スレーブデバイスの CE#端子を MCU の Port に接続し、MCU 汎用ポート出力で、制御させることを推奨します。

また、SPI デバイスの CE#（MCU の Port(CS#)）信号の立ち下がりから、SPI デバイスの CLK（MCU の CLK）信号の立ち下がりまでの時間（SPI デバイスの CE#セットアップ時間）を設けてください。

同様に、SPI デバイスの CLK（MCU の CLK）信号の立ち上がりから、SPI デバイスの/S（MCU の Port(CS#)）信号の立ち上がりまでの時間時間（SPI デバイスの CE#ホールド時間）を設けてください。

SPI デバイスのデータシートを確認して、システムに応じたソフトウェア・ウェイト時間を設定してください。

5.2　ソフトウェア制御概要

5.2.1　ソフトウェア構成

本サンプルコードは、スレーブデバイスとしての SPI デバイスを制御するための下位層に位置するソフトウェアです。

本サンプルコードでは、SPI スレーブデバイスの CE#端子制御無しの SPI モード 3（CPOL=1、CPHA=1）を使った制御を実現しています。

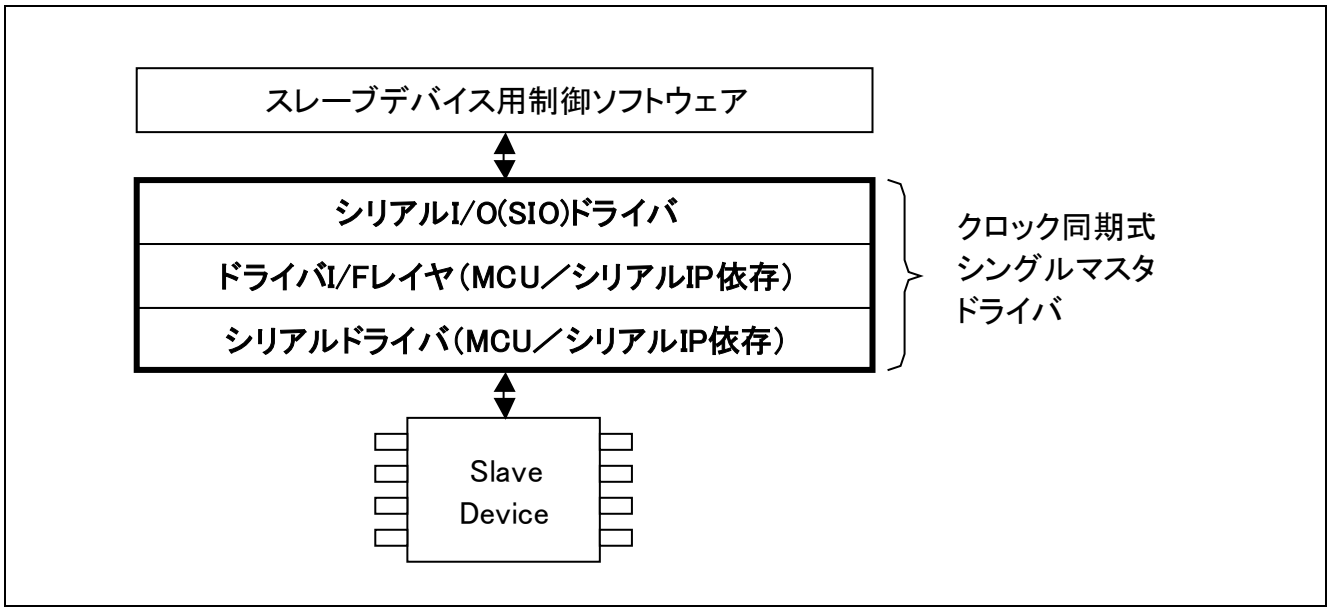


図 5-3　ソフトウェア構成

以下の送信／受信の動作を実現しています。

①クロック同期式シングルマスタソフトウェアを使ったデータ送信

②クロック同期式シングルマスタソフトウェアを使ったデータ受信

本サンプルコードは、以下の 5 つの基本処理で構成しています。

- シリアル許可　－DataIn 端子のポート入力化、DataOut 端子と CLK 端子のポート”H”出力
  シリアル I/O 有効化、ボー・レート設定
- シリアル禁止　－シリアル I/O 無効化、DataIn 端子のポート入力化、
  DataOut 端子と CLK 端子のポート”H”出力化
- シリアル開放　－シリアル I/O 無効化、DataIn 端子のポート入力化、
  DataOut 端子と CLK 端子のポート入力化

データ送信　－SPI デバイスへのデータ送信処理

データ受信　－SPI デバイスからのデータ受信処理

5.2.2　シリアル許可（R_SIO_Enable()）

シリアル I/O で使用する DataIn 端子をポート入力、DataOut 端子と CLK 端子をポート”H”出力にします。

その後、シリアル I/O 機能を有効にし、DataIn 端子をデータ入力、DataOut 端子をデータ出力、CLK 端子をクロック出力に切り替えます。

シリアル I/O で使用する通信速度（ボー・レート）を設定します。

5.2.3　シリアル禁止（R_SIO_Disable()）

シリアル I/O で使用する端子をポートに切り替えて、DataIn 端子をポート入力、DataOut 端子と CLK 端子をポート”H”出力にします。

5.2.4　シリアル開放（R_SIO_Open_Port()）

シリアル I/O で使用する端子をポートに切り替えて、DataIn 端子と DataOut 端子と CLK 端子をポート入力にします。

5.2.5　データ送信（R_SIO_Tx_Data ()）

シリアル I/O を使って、データを送信します。

送信設定にて、送信します。

5.2.6　データ受信（R_SIO_Rx_Data ()）

シリアル I/O を使って、データを受信します。

送受信設定にて、受信します。

5.2.7　データ送受信（R_SIO_TRx_Data ()）

シリアル I/O を使って、データを送受信します。

送受信設定にて、送受信します。

5.3 必要メモリサイズ

命令の異なる MCU 毎にメモリサイズを示します。使用 MCU の命令を調査し参照してください。

環境は、「2. 動作確認条件」を参照してください。

(1) RL78/G14 SAU 統合開発環境 CS+ for CA,CX の場合（コンパイラ：CA78K0R）

表 5-1 必要メモリサイズ

使用メモリ	サイズ	備考
ROM	653 バイト	R_SIO_csi.c
RAM	0 バイト	R_SIO_csi.c
最大使用ユーザスタック	24 バイト	
最大使用割り込みスタック	―	割り込み未使用

必要メモリサイズは、C コンパイラのバージョンやコンパイルオプションにより異なります。

(2) RL78/G14 SAU 統合開発環境 CS+ for CC の場合（コンパイラ：CC-RL）

表 5-2 必要メモリサイズ

使用メモリ	サイズ	備考
ROM	598 バイト	R_SIO_csi.c
RAM	0 バイト	R_SIO_csi.c
最大使用ユーザスタック	20 バイト	
最大使用割り込みスタック	―	割り込み未使用

必要メモリサイズは、C コンパイラのバージョンやコンパイルオプションにより異なります。

(3) RL78/G14 SAU 統合開発環境 IAR Embedded Workbench の場合

表 5-3 必要メモリサイズ

使用メモリ	サイズ	備考
ROM	547 バイト	R_SIO_csi.c
RAM	0 バイト	R_SIO_csi.c
最大使用ユーザスタック	94 バイト	
最大使用割り込みスタック	―	割り込み未使用

必要メモリサイズは、C コンパイラのバージョンやコンパイルオプションにより異なります。
最大使用ユーザスタックサイズは、プロジェクト全体のスタックサイズです。

(4) RL78/L13 SAU 統合開発環境 CubeSuite+の場合

表 5-4 必要メモリサイズ

使用メモリ	サイズ	備考
ROM	525 バイト	R_SIO_csi.c
RAM	0 バイト	R_SIO_csi.c
最大使用ユーザスタック	22 バイト	
最大使用割り込みスタック	―	割り込み未使用

必要メモリサイズは、C コンパイラのバージョンやコンパイルオプションにより異なります。

(5) RL78/L13 SAU 統合開発環境 IAR Embedded Workbench の場合

表 5-5 必要メモリサイズ

使用メモリ	サイズ	備考
ROM	516 バイト	R_SIO_csi.c
RAM	0 バイト	R_SIO_csi.c
最大使用ユーザスタック	94 バイト	
最大使用割り込みスタック	―	割り込み未使用

必要メモリサイズは、C コンパイラのバージョンやコンパイルオプションにより異なります。
最大使用ユーザスタックサイズは、プロジェクト全体のスタックサイズです。

(6) RL78/G23 SAU 統合開発環境 CS+ for CC の場合（コンパイラ：CC-RL）

表 5-6 必要メモリサイズ

使用メモリ	サイズ	備考
ROM	711 バイト	R_SIO_csi.c
RAM	0 バイト	R_SIO_csi.c
最大使用ユーザスタック	18 バイト	―
最大使用割り込みスタック	―	割り込み未使用

必要メモリサイズは、C コンパイラのバージョンやコンパイルオプションにより異なります。

(7) RL78/G23 SAU 統合開発環境 IAR Embedded Workbench の場合

表 5-7 必要メモリサイズ

使用メモリ	サイズ	備考
ROM	465 バイト	R_SIO_csi.c
RAM	0 バイト	R_SIO_csi.c
最大使用ユーザスタック	22 バイト	—
最大使用割り込みスタック	—	割り込み未使用

必要メモリサイズは、C コンパイラのバージョンやコンパイルオプションにより異なります。

最大使用ユーザスタックサイズは、プロジェクト全体のスタックサイズです。

(8) RL78/G23 SAU 統合開発環境 e^2 studio の場合（コンパイラ：LLVM）

表 5-8 必要メモリサイズ

使用メモリ	サイズ	備考
ROM	454 バイト	R_SIO_csi.c
RAM	0 バイト	R_SIO_csi.c
最大使用ユーザスタック	12 バイト	—
最大使用割り込みスタック	—	割り込み未使用

必要メモリサイズは、C コンパイラのバージョンやコンパイルオプションにより異なります。

5.4 ファイル構成

表 5-8 に、サンプルコードで使用するファイルを示します。なお、統合開発環境で自動生成するファイルを除きます。

表 5-9 ファイル構成

¥r01an1195xx0107-rl78-serial <DIR>			サンプルコードのフォルダ
r01an1195jj0107-rl78-serial.pdf			アプリケーションノート（本書）
r01an1195ej0107-rl78-serial.pdf			アプリケーションノート英語版
¥ source <DIR>			プログラム格納用フォルダ
	¥com <DIR>		共通関数格納用フォルダ
	（注 1）	mtl_com.c	共通関数の各種定義
		mtl_com.h.common	共通ヘッダファイル
		mtl_com.h.RL78	共通関数のヘッダファイル
		mtl_endi.c	共通ファイル（エンディアン設定関連）
		mtl_mem.c	共通ファイル（標準ライブラリ関数）
		mtl_os.c mtl_os.h	共通ファイル（標準ライブラリ関数）
		mtl_str.c	共通ファイル（標準ライブラリ関数）
		mtl_tim.c mtl_tim.h	共通ファイル（ループタイマ関連）
		mtl_tim.h.sample	ループタイマの設定値サンプル
	¥r_sio_csi_rl78 <DIR>		RL78 CSI 用クロック同期式シングルマスタ制御ソフトウェアのフォルダ
		R_SIO.h	ヘッダファイル
		R_SIO_csi.c	I/F モジュール
		R_SIO_csi.c.rl78g23	I/F モジュール（RL78/G23 用）
		R_SIO_csi.h.rl78g1c	I/F モジュール共通定義（RL78/G1C 用）
		R_SIO_csi.h.rl78g14	I/F モジュール共通定義（RL78/G14 用）
		R_SIO_csi.h.rl78l1c	I/F モジュール共通定義（RL78/L1C 用）
		R_SIO_csi.h.rl78l12	I/F モジュール共通定義（RL78/L12 用）
		R_SIO_csi.h.rl78l13	I/F モジュール共通定義（RL78/L13 用）
		R_SIO_csi.h.rl78g23	I/F モジュール共通定義（RL78/G23 用）

注 1. com フォルダに含まれるファイルは、スレーブデバイス用制御ソフトウェアでも使用するものです。最新のものを使用してください。

5.5 定数一覧

5.5.1 戻り値

表 5-9 に、サンプルコードで使用する戻り値を示します。

表 5-10 戻り値

定数名	設定値	内容
SIO_OK	(error_t)(0)	Successful Operation
SIO_ERR_PARAM	(error_t)(-1)	Parameter Error
SIO_ERR_HARD	(error_t)(-2)	Hardware Error
SIO_ERR_OTHER	(error_t)(-7)	Other Error

5.5.2 各種定義

表 5-10 に、サンプルコードで使用する各種定義した値を示します。

表 5-11 各種定義値

定数名	設定値	内容
SIO_LOG_ERR	1	Log type : Error
SIO_TRUE	(uint8_t)0x01	Flag "ON"
SIO_FALSE	(uint8_t)0x00	Flag "OFF"
SIO_HI	(uint8_t)0x01	Port "H"
SIO_LOW	(uint8_t)0x00	Port "L"
SIO_OUT	(uint8_t)0x01	Port output setting
SIO_IN	(uint8_t)0x00	Port input setting
SIO_TX_WAIT	(uint16_t)50000	SIO transmission completion waiting time 50000* 1us = 50ms
SIO_RX_WAIT	(uint16_t)50000	SIO receive completion waiting time 50000* 1us = 50ms
SIO_DMA_TX_WAIT	(uint16_t)50000	DMA transmission completion waiting time 50000* 1us = 50ms
SIO_DMA_RX_WAIT	(uint16_t)50000	DMA receive completion waiting time 50000* 1us = 50ms
SIO_T_SIO_WAIT	(uint16_t)MTL_T_1US	SIO transmit&receive completion waiting polling time
SIO_T_DMA_WAIT	(uint16_t)MTL_T_1US	DMA transmit&receive completion waiting polling time
SIO_T_BRR_WAIT	(uint16_t)MTL_T_10US	BRR setting wait time

5.6 構造体／共用体一覧

以下に、サンプルコードで使用する構造体を示します。

```
/* uint32_t <-> uint8_t conversion */
typedef union {
  uint32_t   ul;
  uint8_t    uc[4];
} SIO_EXCHG_LONG;                          /* total 4byte                */


/* uint16_t <-> uint8_t conversion */
typedef union {
  uint16_t   us;
  uint8_t    uc[2];
} SIO_EXCHG_SHORT;                         /* total 2byte                */
```

5.7 関数一覧

表 5-11 に関数一覧を示します。

表 5-12 関数一覧

関数名	説明
R_SIO_Init_Driver()	ドライバ初期化処理
R_SIO_Disable()	シリアル I/O 禁止設定処理
R_SIO_Enable()	シリアル I/O 許可設定処理
R_SIO_Open_Port()	シリアル I/O 開放設定処理
R_SIO_Tx_Data()	シリアル I/O データ送信処理
R_SIO_Rx_Data()	シリアル I/O データ受信処理
R_SIO_TRx_Data()	シリアル I/O データ送受信処理

5.8 関数仕様

本サンプルコードでは、シリアル・アレイ・ユニットの入力クロック供給を許可しますが、入力クロック供給の停止制御は、含まれていません。

したがって、低消費電力化とノイズ低減を図るためにユニット単位での動作停止制御を行う場合、本サンプルコードで使用する以外のチャネルの制御を考慮し、ユーザプログラムで制御してください。

なお、本サンプルコードは、ユニット単位での動作停止制御を行っていませんが、チャネル単位での動作停止制御が可能です。

本サンプルコードの動作クロックは、シリアル・クロック選択レジスタ(SPSm)で選択した動作クロック CKm0 を使用する設定です。必要に応じて、シリアル・モード・レジスタ(SMRmn)、シリアル・クロック選択レジスタ(SPSm)の設定を見直してください。

5.8.1 ドライバ初期化処理

R_SIO_Init_Driver	
概要	ドライバ初期化処理
ヘッダ	R_SIO.h, R_SIO_csi.h, mtl_com.h
宣言	error_t R_SIO_Init_Driver(void)
説明	・ドライバの初期化を行います。シリアル I/O 機能を無効化し、端子をポートに設定します。 ・システム起動時に一度だけ呼び出してください。 ・本関数コール前に、スレーブデバイスセレクト制御信号を”H”にしてください。
引数	なし
リターン値	SIO_OK ; Successful operation
備考	前の使用状態を考慮し、以下の処理を行います。 ・シリアル・アレイ・ユニットの入力クロック供給を許可します。 ・送信／受信を停止します。 ・シリアル I/O で使用する端子をポート設定にします。

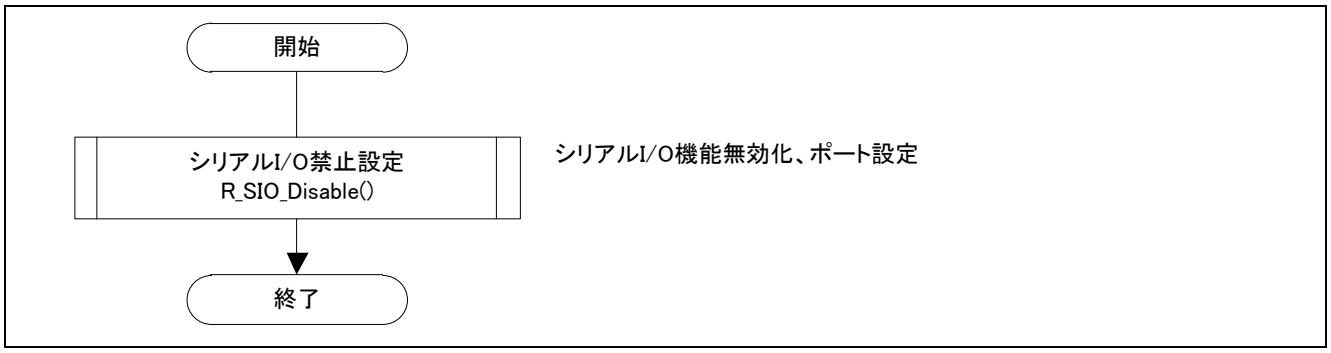


図 5-4 ドライバ初期化処理概要

5.8.2　シリアル I/O 禁止設定処理

R_SIO_Disable	
概要	シリアル I/O 禁止設定処理
ヘッダ	R_SIO.h, R_SIO_csi.h, mtl_com.h
宣言	error_t R_SIO_Disable(void)
説明	・シリアル I/O 機能を無効化し、端子をポートに設定します。 シリアル・アレイ・ユニットの入力クロック供給を許可します。 シリアル I/O を無効化します。 シリアル I/O で使用する端子をポート設定にします。 ・本関数コール前に、スレーブデバイスセレクト制御信号を”H”にしてください。
引数	なし
リターン値	SIO_OK ; Successful operation
備考	・シリアル・アレイ・ユニットの入力クロック供給を許可します。 ・fCLK の 4 クロック以上の設定待ちます。 ・STm, SOm, SOEm を設定し、動作停止状態にし、ポート機能に切り替えます。 ・SCRmn を設定し、通信モードを通信禁止にします。 ・SMRmn に 0020h（リセット時の値）を書き込み、初期化します。 ・SOLm を設定 ・使用しない場合、本関数をコールし、シリアル I/O 機能を無効化することができます。 ・シリアル・アレイ・ユニットの入力クロック供給停止制御は行いません。

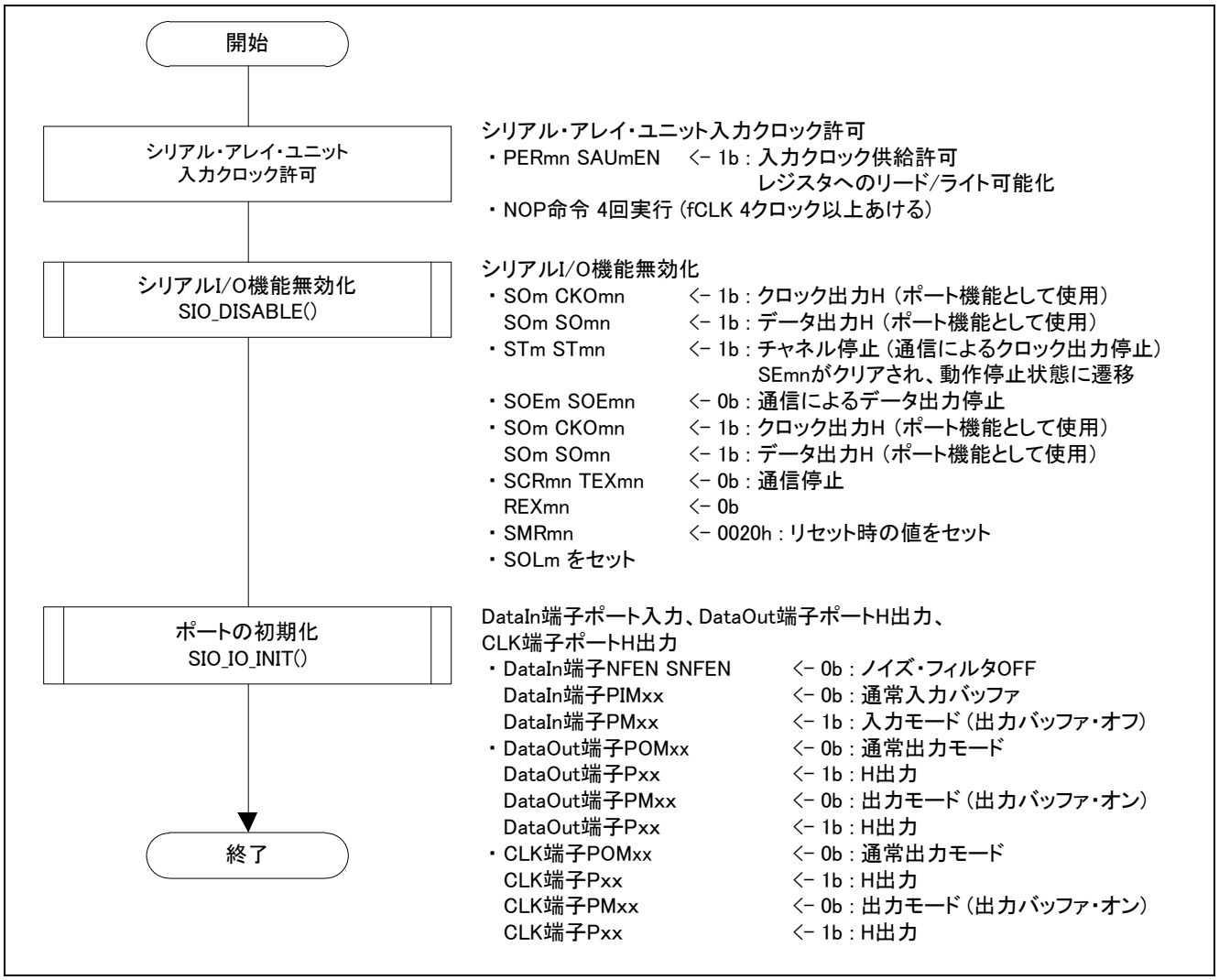


図 5-5　シリアル I/O 禁止設定処理概要

5.8.3　シリアル I/O 許可設定処理

R_SIO_Enable	
概要	シリアル I/O 許可設定処理
ヘッダ	R_SIO.h, R_SIO_csi.h, mtl_com.h
宣言	error_t R_SIO_Enable(uint8_t BrgData)
説明	・シリアル I/O 機能を有効化し、ボー・レートを設定します。 シリアル・アレイ・ユニットに入力クロック供給を許可します。 シリアル I/O で使用する端子をポート設定にします。 シリアル I/O を有効化し、ボー・レートを設定します。 ・R_SIO_Disable()コール後に、本関数をコールしてください。 ・シリアル I/O データ送信処理とシリアル I/O データ受信処理実行前に、一度、本関数をコールしてください。 ・ボー・レートを変更したい場合、本関数を使用してください。事前に、シリアル I/O 禁止設定処理を実行してください。
引数	uint8_t　BrgData　;　ボー・レート設定値
リターン値	SIO_OK　;　Successful operation
備考	・ハードウェアマニュアル記載のマスタ送信／マスタ送受信の初期化設定手順にしたがって、以下を実行します。（R_SIO_Disable()がコールされた状態を想定します。） ①PER SAUmEN を設定 シリアル・アレイ・ユニットに入力クロック供給を許可 ②fCLK の 4 クロック以上の設定待ち ③ポートを初期化 ④SPSm に動作クロックを設定 ⑤SMRmn に動作モードを設定 ⑥SCRmn に通信フォーマットを設定 ⑦SDRmn にボー・レートを設定 ⑧SIRmn のエラーフラグをクリア ⑨SOLm を設定

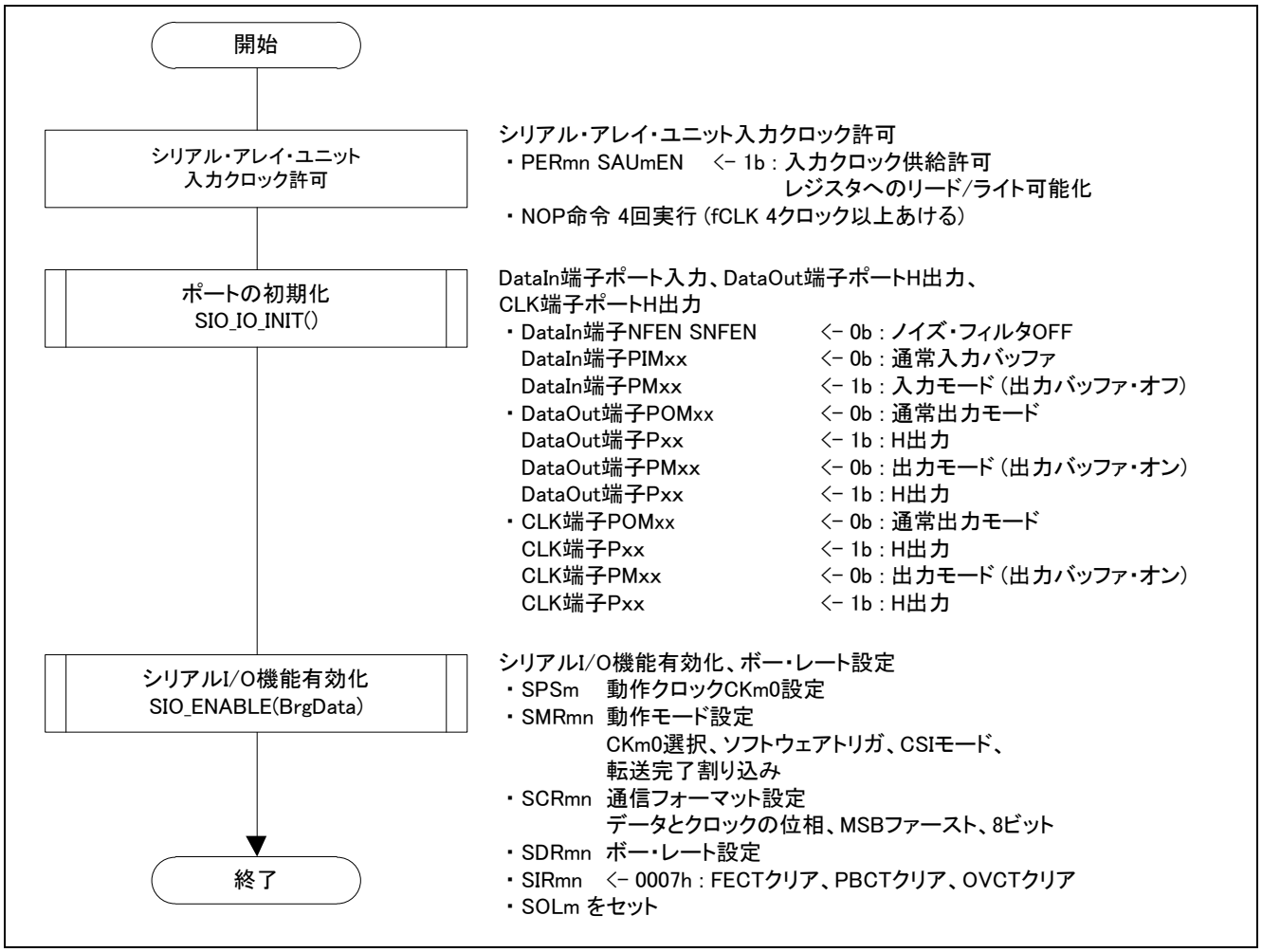


図 5-6 シリアル I/O 許可設定処理概要

5.8.4　シリアル I/O 開放設定処理

R_SIO_Open_Port	
概要	シリアル I/O 開放設定処理
ヘッダ	R_SIO.h, R_SIO_csi.h, mtl_com.h
宣言	error_t R_SIO_Open_Port(void)
説明	・シリアル I/O に使用する端子をオープン（入力状態）にします。 ・本関数コール前に、スレーブデバイスセレクト制御信号を”H”にしてください。
引数	なし
リターン値	SIO_OK　;　Successful operation
備考	・リムーバブルメディアの挿抜目的で、用意した関数です。リムーバブルメディアの挿入前、およびリムーバブルメディアの抜去前に、本関数を使用してください。リムーバブルメディアの抜去前には、シリアル I/O 禁止設定処理を実行してください。

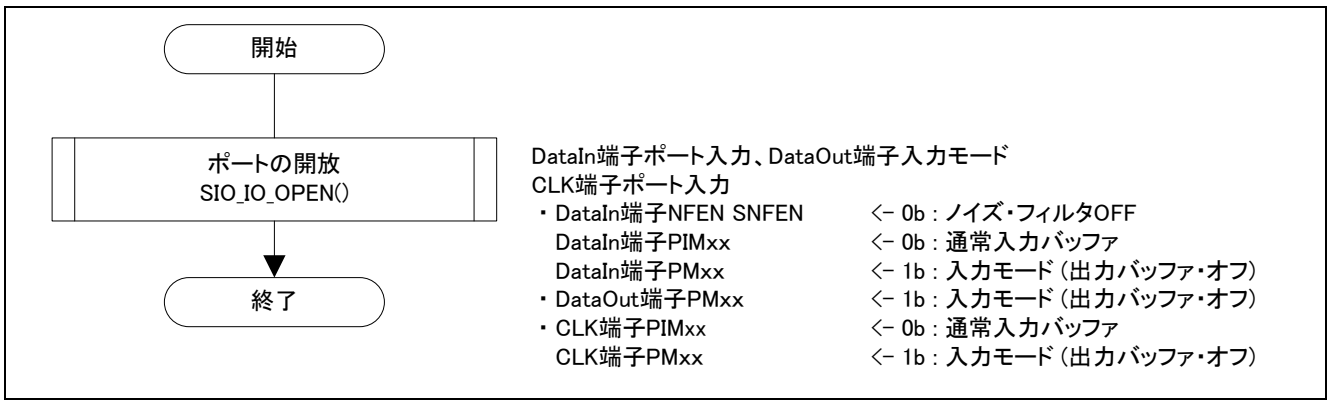


図 5-7　シリアル I/O 開放設定処理概要

5.8.5　シリアル I/O データ送信処理

R_SIO_Tx_Data				
概要	シリアル I/O データ送信処理			
ヘッダ	R_SIO.h, R_SIO_csi.h, mtl_com.h			
宣言	error_t R_SIO_Tx_Data(uint16_t TxCnt, uint8_t FAR* pData)			
説明	・pData のデータを指定バイト数分送信します。 ・本関数コール前に、シリアル I/O 許可設定処理を実行してください。 ・本関数の実行の結果、異常終了であれば、シリアル I/O 禁止設定処理を実行してください。			
引数	uint16_t	TxCnt	;	送信バイト数
	uint8_t FAR*	pData	;	送信データ格納バッファポインタ
リターン値	SIO_OK		;	Successful operation
	SIO_ERR_HARD		;	Hardware error
備考	・ハードウェアマニュアル記載のマスタ送信の初期化設定手順にしたがって、以下のシリアル I/O 許可設定処理以降の初期化設定を実行します。 ①SCRmn に TEXmn=1b, REXmn=0b を設定し、送信を許可 ②SOm にデータ出力 H、クロック出力 H を設定 ③SOEm にシリアル通信動作によるデータ出力許可を設定 ④SSm にシリアル通信によるクロック出力許可を設定 ・送信完了後、ハードウェアマニュアル記載のマスタ送信の中断手順にしたがって、以下を実行します。 ①STm にシリアル通信によるクロック出力停止を設定 ②SOEm にシリアル通信動作によるデータ出力停止を設定 ③SCRmn に TEXmn=0b, REXmn=0b を設定し、通信を禁止 ・継続使用しない場合、シリアル I/O 禁止設定処理を実行することを推奨します。			

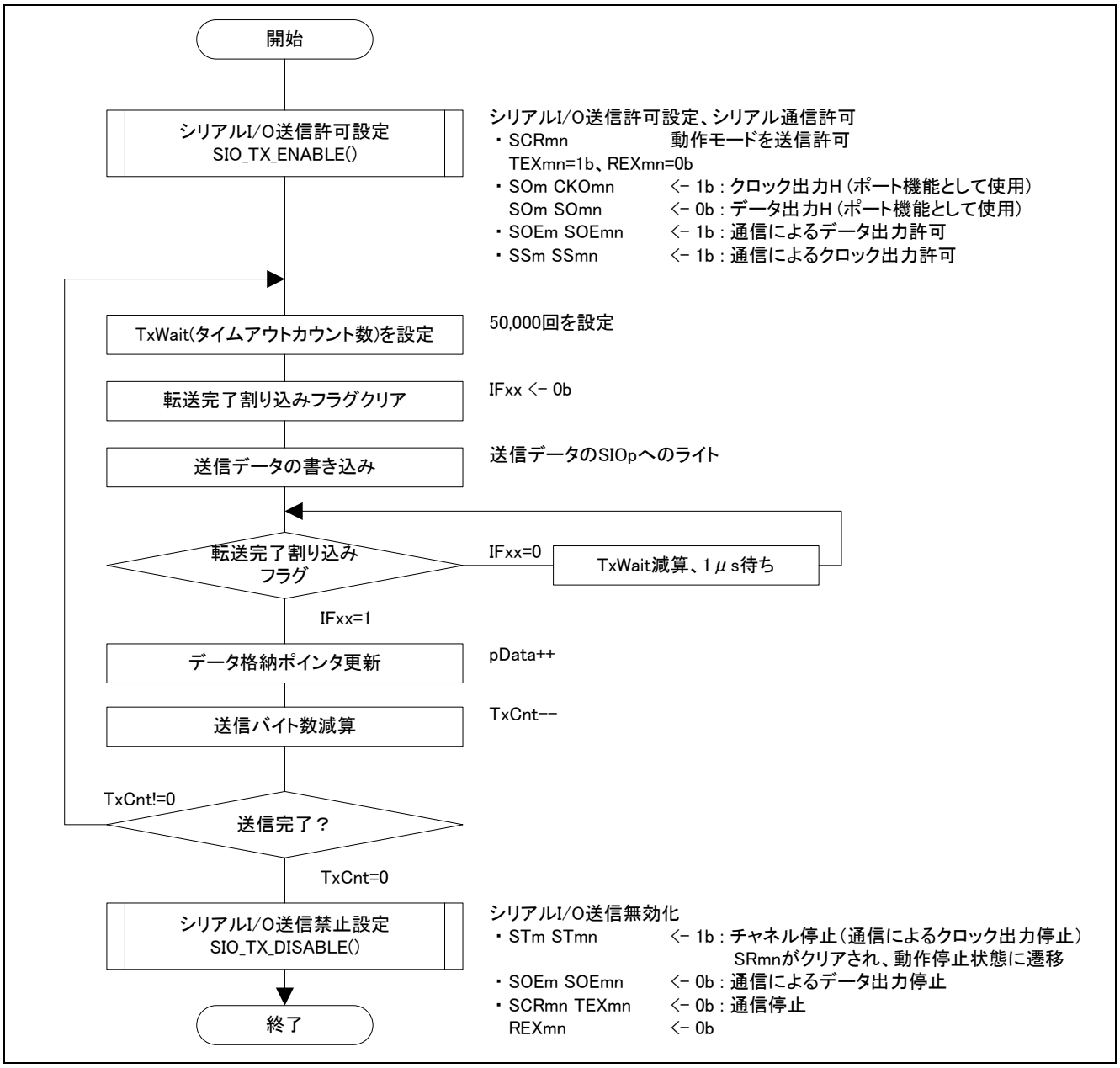


図 5-8 シリアル I/O データ送信処理概要

5.8.6　シリアル I/O データ受信処理

R_SIO_Rx_Data				
概要	シリアル I/O データ受信処理			
ヘッダ	R_SIO.h, R_SIO_csi.h, mtl_com.h			
宣言	error_t R_SIO_Rx_Data(uint16_t RxCnt, uint8_t FAR* pData)			
説明	・指定バイト数分でデータを受信し、pData に格納します。 ・本関数コール前に、シリアル I/O 許可設定処理を実行してください。 ・本関数の実行の結果、異常終了であれば、シリアル I/O 禁止設定処理を実行してください。			
引数	uint16_t	RxCnt	;	受信バイト数
	uint8_t FAR*	pData	;	受信データ格納バッファポインタ
リターン値	SIO_OK		;	Successful operation
	SIO_ERR_HARD		;	Hardware error
備考	・ハードウェアマニュアル記載のマスタ送受信の初期化設定手順にしたがって、以下のシリアル I/O 許可設定処理以降の初期化設定を実行します。 ①SCRmn に TEXmn=1b, REXmn=1b を設定し、送受信を許可 ②SOm にデータ出力 H、クロック出力 H を設定 ③SOEm にシリアル通信動作によるデータ出力許可を設定 ④SSm にシリアル通信によるクロック出力許可を設定 ・送信完了後、ハードウェアマニュアル記載のマスタ送受信の中断手順にしたがって、以下を実行します。 ①STm にシリアル通信によるクロック出力停止を設定 ②SOEm にシリアル通信動作によるデータ出力停止を設定 ③SCRmn に TEXmn=0b, REXmn=0b を設定し、通信を禁止 ・継続使用しない場合、シリアル I/O 禁止設定処理を実行することを推奨します。			

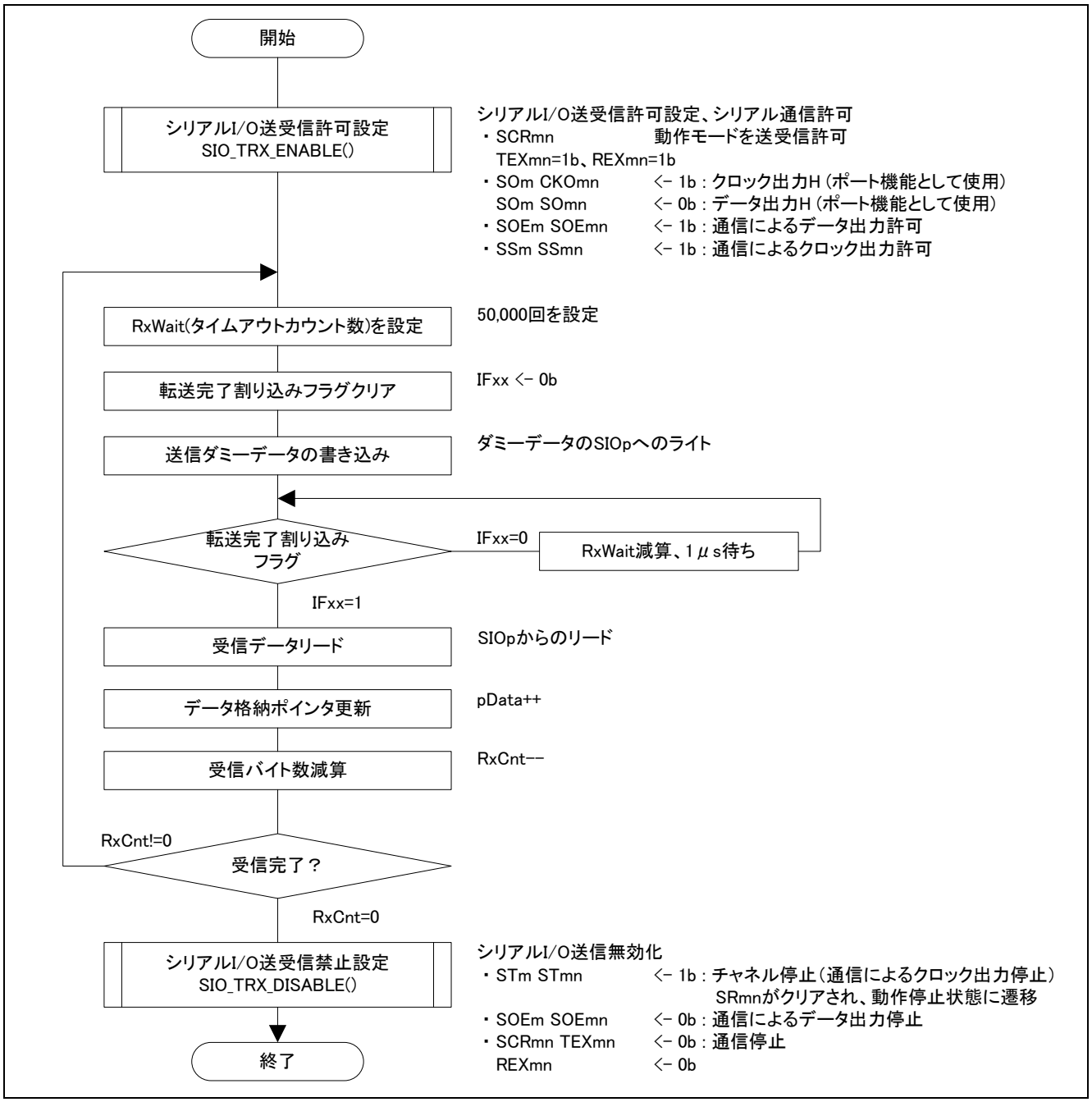


図 5-9 シリアル I/O データ受信処理概要

5.8.7　シリアル I/O データ送受信処理

R_SIO_TRx_Data				
概要	シリアル I/O データ送受信処理			
ヘッダ	R_SIO.h, R_SIO_csi.h, mtl_com.h			
宣言	error_t R_SIO_TRx_Data(uint16_t TRxCnt, uint8_t FAR* pTxData, uint8_t FAR* pRxData)			
説明	・指定バイト数分 pTxData のデータを送信し、受信したデータを pRxData に格納します。 ・本関数コール前に、シリアル I/O 許可設定処理を実行してください。 ・本関数の実行の結果、異常終了であれば、シリアル I/O 禁止設定処理を実行してください。			
引数	uint16_t	TRxCnt	;	送受信バイト数
	uint8_t FAR*	pTxData	;	送信データ格納バッファポインタ
	uint8_t FAR*	pRxData	;	受信データ格納バッファポインタ
リターン値	SIO_OK		;	Successful operation
	SIO_ERR_HARD		;	Hardware error
備考	・ハードウェアマニュアル記載のマスタ送受信の初期化設定手順にしたがって、以下のシリアル I/O 許可設定処理以降の初期化設定を実行します。 ①SCRmn に TEXmn=1b, REXmn=1b を設定し、送受信を許可 ②SOm にデータ出力 H、クロック出力 H を設定 ③SOEm にシリアル通信動作によるデータ出力許可を設定 ④SSm にシリアル通信によるクロック出力許可を設定 ・送受信完了後、ハードウェアマニュアル記載のマスタ送受信の中断手順にしたがって、以下を実行します。 ①STm にシリアル通信によるクロック出力停止を設定 ②SOEm にシリアル通信動作によるデータ出力停止を設定 ③SCRmn に TEXmn=0b, REXmn=0b を設定し、通信を禁止 ・継続使用しない場合、シリアル I/O 禁止設定処理を実行することを推奨します。			

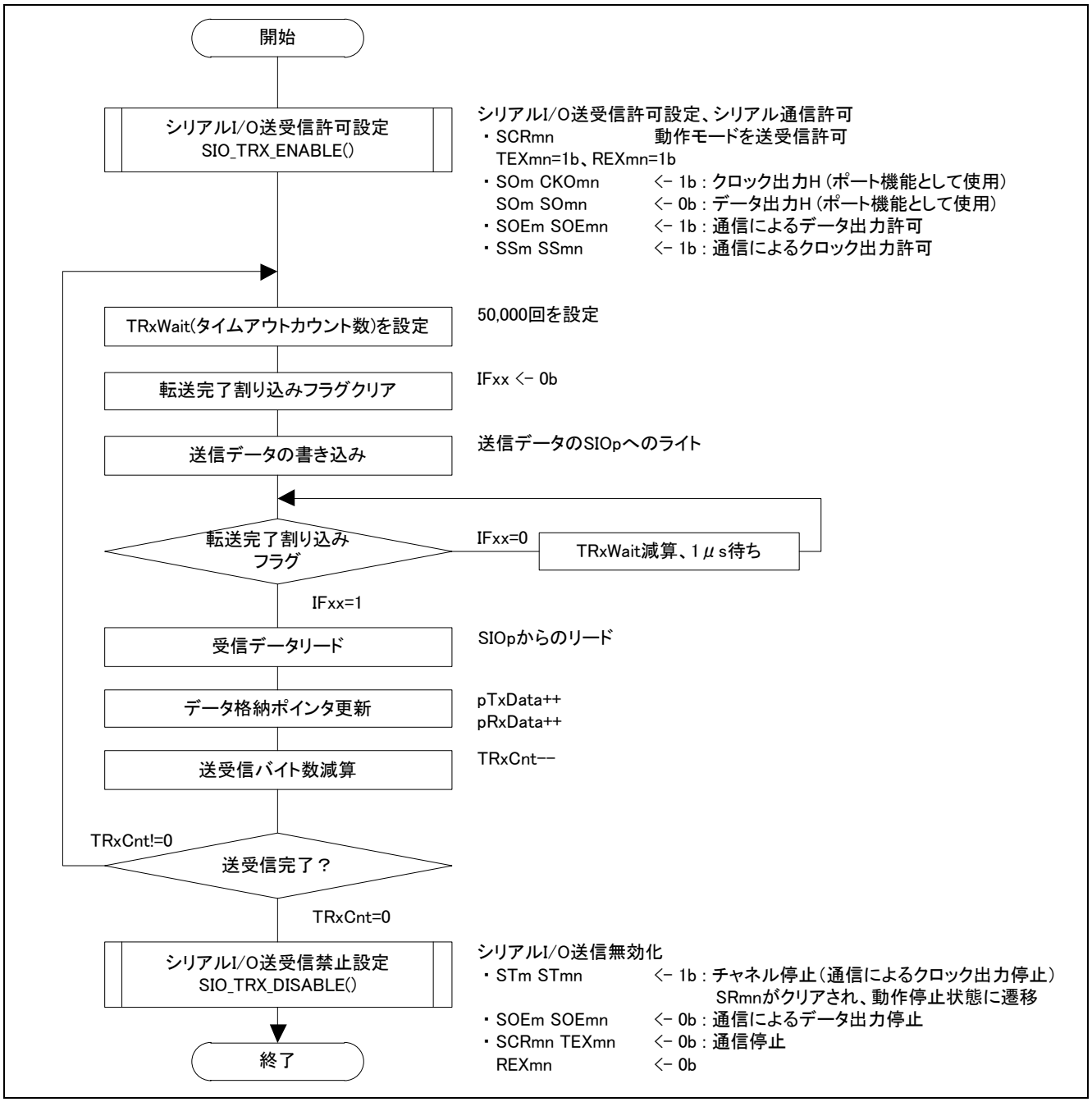


図 5-10 シリアル I/O データ受信処理概要

5.9 マクロ関数仕様

以下に、本サンプルコードで使用するマクロ関数を示します。

5.9.1 マクロ関数 SIO_IO_INIT()

(1) 目的

入力端子をポート入力状態、出力端子をポート出力状態にします。

(2) 機能

DataIn 端子をポート入力状態、DataOut 端子と CLK 端子をポート出力状態にします。

以下の処理を実現します。必要に応じて、処理を見直してください。

①DataIn 端子をポート入力に設定する。

②DataOut 端子をポート”H”出力に設定する。

③CLK 端子をポート”H”出力に設定する。

(3) 備考

本関数実行前に、端子をポート機能使用可能状態にしてください。

シリアル通信出力停止状態の出力端子は、シリアル出力レジスタ(SOm)とポート・レジスタ(Pxx)で設定した出力ラッチの論理積(AND)によって決まります。本関数を実行することで、ポート・レジスタ(Pxx)により H 出力に設定するため、出力端子は、シリアル出力レジスタ(SOm)の該当する CKOmn ビット、SOmn ビットに依存します。SIO_DISABLE()を実行し、シリアル出力レジスタ(SOm)の該当する CKOmn ビットと SOmn ビットを”1”に設定し、ポート機能を有効した後に、本関数を実行してください。

シリアル・アレイ・ユニットのレジスタ操作のため、事前に、PERn（n は、レジスタ番号）の SAUmEN 制御によるクロック供給を許可してください。

5.9.2 マクロ関数 SIO_IO_OPEN()

(1) 目的

入力端子と出力端子をポート入力状態もしくは出力バッファオフ状態にします。

(2) 機能

DataIn 端子と DataOut 端子と CLK 端子入力端子をポート入力状態にします。

以下の処理を実現します。必要に応じて、処理を見直してください。

①DataIn 端子をポート入力に設定する。

②DataOut 端子を入力モード（出力バッファオフ）に設定する。

③CLK 端子をポート入力に設定する。

(3) 備考

リムーバブルメディアの挿入前および抜去前での、全端子の Hi-z 化をするために使用してください。

SIO_IO_INIT()を実行後に、本関数を実行してください。

シリアル・アレイ・ユニットのレジスタ操作のため、事前に、PERn（n は、レジスタ番号）の SAUmEN 制御によるクロック供給を許可してください。

5.9.3 マクロ関数 SIO_DATAI_INIT()

(1) 目的

DataIn 端子をポート入力状態にします。

(2) 機能

以下の処理を実現します。必要に応じて、処理を見直してください。

①RL78/L1x の場合、LCD ポート・ファンクション・レジスタ x(PFSEGx)を使って、DataIn 端子をポート(セグメント出力以外)として使用、に設定する。

②CSI モードのため、DataIn 端子をノイズ・フィルタ OFF に設定する。

③ポート入力モード・レジスタ(PIMxx)を使って、DataIn 端子の入力バッファを通常入力バッファに設定する。

④ポート・モード・レジスタ(PMxx)を使って、DataIn 端子をポート入力に設定する。

(3) 備考

接続するデバイスによって、ポート入力モード・レジスタ(PIMxx)の値を見直してください。

シリアル・アレイ・ユニットのレジスタ操作のため、事前に、PERn（n は、レジスタ番号）の SAUmEN 制御によるクロック供給を許可してください。

5.9.4 マクロ関数 SIO_DATAO_INIT()

(1) 目的

DataOut 端子をポート”H”出力にします。

(2) 機能

以下の処理を実現します。必要に応じて、処理を見直してください。

①RL78/L1x の場合、LCD ポート・ファンクション・レジスタ x(PFSEGx)を使って、DataOut 端子をポート(セグメント出力以外)として使用、に設定する。

②ポート出力モード・レジスタ(POMxx)を使って、DataOut 端子の出力モードを通常出力モードに設定する。

③ポート・モード・レジスタ(PMxx)とポート・レジスタ(Pxx)を使って、DataOut 端子をポート”H”出力に設定する。

(3) 備考

接続するデバイスによって、ポート出力モード・レジスタ(POMxx)の値を見直してください。

シリアル通信出力停止状態の出力端子は、シリアル出力レジスタ(SOm)とポート・レジスタ(Pxx)で設定した出力ラッチの論理積(AND)によって決まります。本関数を実行することで、ポート・レジスタ(Pxx)により H 出力に設定するため、出力端子は、シリアル出力レジスタ(SOm)の該当する SOmn ビットに依存します。SIO_DISABLE()を実行し、シリアル出力レジスタ(SOm)の該当する SOmn ビットを”1”に設定し、ポート機能を有効した後に、本関数を実行してください。

5.9.5 マクロ関数 SIO_DATAO_OPEN()

(1) 目的

DataOut 端子をポート入力状態にします。

(2) 機能

以下の処理を実現します。必要に応じて、処理を見直してください。

①ポート・モード・レジスタ(PMxx)を使って、DataOut 端子をポート入力状態もしくは出力バッファオフ状態に設定する。

(3) 備考

なし

5.9.6 マクロ関数 SIO_CLK_INIT()

(1) 目的

CLK 端子をポート”H”出力にします。

(2) 機能

以下の処理を実現します。必要に応じて、処理を見直してください。

①RL78/L1x の場合、LCD ポート・ファンクション・レジスタ x(PFSEGx)を使って、CLK 端子をポート(セグメント出力以外)として使用、に設定する。

②ポート出力モード・レジスタ(POMxx)を使って、DataOut 端子の出力モードを通常出力モードに設定する。

③ポート・モード・レジスタ(PMxx)とポート・レジスタ(Pxx)を使って、CLK 端子をポート”H”出力に設定する。

(3) 備考

接続するデバイスによって、ポート出力モード・レジスタ(POMxx)の値を見直してください。

シリアル通信出力停止状態の出力端子は、シリアル出力レジスタ(SOm)とポート・レジスタ(Pxx)で設定した出力ラッチの論理積(AND)によって決まります。本関数を実行することで、ポート・レジスタ(Pxx)により H 出力に設定するため、出力端子は、シリアル出力レジスタ(SOm)の該当する CKOmn ビットに依存します。SIO_DISABLE()を実行し、シリアル出力レジスタ(SOm)の該当する CKOmn ビットを”1”に設定し、ポート機能を有効した後に、本関数を実行してください。

5.9.7 マクロ関数 SIO_CLK_OPEN()

(1) 目的

CLK 端子をポート入力状態にします。

(2) 機能

以下の処理を実現します。必要に応じて、処理を見直してください。

①ポート入力モード・レジスタ(PIMxx)を使って、CLK 端子の入力バッファを通常入力バッファに設定する。

②ポート・モード・レジスタ(PMxx)を使って、CLK 端子をポート入力に設定する。

(3) 備考

接続するデバイスによって、ポート入力モード・レジスタ(PIMxx)の値を見直してください。

5.9.8　マクロ関数　SIO_ENABLE()

(1)　目的

シリアル I/O を初期化し、機能を有効化します。ただし、送信許可／受信許可／送受信許可にするまでの共通処理を実行します。また、ボー・レートを設定します。

(2)　機能

ハードウェアマニュアルにしたがって、シリアル I/O を初期化します。必要に応じて、処理を見直してください。RL78 ファミリ MCU の場合、以下の処理を行います。

①送信設定と送受信設定の有効化手順において、共通の処理を行う。

・SPSm に動作クロックを設定する。

CKm0 を設定する。本レジスタは 2 種類の動作クロック(CKm0,CKm1)の設定が可能であるため、論理和を取って設定する。

・SMRmn に動作モードを設定する。

CKSmn に、SPSm で設定したプリスケーラ出力クロック”CKm0”を設定する。

MDmn2-1 に、CSI モードを設定する。

MDmn0 に、割り込み要因として転送完了割り込みを設定する。

・SCRmn に通信フォーマットを設定する。

DAPmn,CKPmn に、データとクロックの位相（DAPmn=0, CKPmn=0：SPI モード 3 互換）を設定する。

DIRmn に、データ転送手順（MSB ファースト）を設定する。

DLSmn2-DLSmn0 に、データ長（8 ビット・データ長）を設定する。

・SDRmn の動作クロック(fMCK)の分周設定部（SDRmn のビット 15-9）を書き込み、ボー・レートを設定する。

・SIRmn の FECTmn、PECTmn、OVCTmn の各フラグに、”1”を書き込んで各フラグのクリアを行う。

・SOLm の SOLmn=0b を設定する。【チャネル依存】

(3)　備考

SIO_DISABLE()と対となるものです。本関数を実行した場合、SIO_DISABLE()を実行して、処理を終了してください。

SPSm、SMRm、SOLm を設定するため、SEmn=0 に設定する必要があります。本関数の実行前に、SIO_DISABLE()を実行してください。

動作クロックとして、CKm0 を使用します。

5.9.9　マクロ関数　SIO_DISABLE()

(1)　目的

シリアル I/O 機能を無効化します。

(2)　機能

シリアル I/O を無効化します。送信設定／送受信設定の無効化手順において、共通の処理を行います。必要に応じて、処理を見直してください。RL78 ファミリ MCU の場合、以下の処理を行います。

①チャネルの動作停止モードに設定し、端子をポート機能に切り替える。

・SOm の CKOmn=1b, SOmn=1b を設定し、ポート機能として使用する。（※１）
・STm の STmn=1b を設定する。
⇒SEmn ビットが”0”にクリアされ、シリアル通信動作によるクロック出力を停止する。
⇒チャネルが動作停止状態になる。
⇒SOm の CKOmn ビットに設定された値が、シリアル・クロック出力端子から出力される
・SOEm の SOEmn=0b を設定し、シリアル通信動作によるデータ出力を停止する。
・SOm の CKOmn=1b, SOmn=1b を設定し、ポート機能として使用する。（※２）
・SOLm の SOLmn=0b を設定する。【チャネル依存】

②SCRmn に TEXmn=0b, REXmn=0b を設定し、動作モードを通信禁止にする。

③SMRmn に 0020h（リセット時の値）を設定する。

(3)　備考

SIO_ENABLE()と対となるものです。SIO_ENABLE()を実行した場合、本関数を実行して、処理を終了してください。

SIO_TX_DISABLE()/SIO_TRX_DISABLE()にて、STm 制御による通信動作停止を行いますが、本関数においても STm 制御による通信動作停止を行います。

シリアル通信出力停止状態の出力端子は、シリアル出力レジスタ(SOm)とポート・レジスタ(Pxx)で設定した出力ラッチの論理積(AND)によって決まります。本関数を実行することで、ポート・レジスタ(Pxx)により H 出力に設定するため、出力端子は、シリアル出力レジスタ(SOm)の該当する CKOmn ビット、SOmn ビットに依存します。

シリアル・アレイ・ユニットのレジスタ操作のため、事前に、PERn（n は、レジスタ番号）の SAUmEN 制御によるクロック供給を許可してください。

初期に SEmn=0 に設定することにより、SPSm、SMRm、SDRm 等のレジスタへの書き込みを許可にします。

本関数がコールされるタイミングは、初期化処理時と、送信／受信の完了後を想定しています。

※１：STm 制御によるクロック出力停止と SOEm 制御によるデータ出力停止の前に、SOm を設定する目的で、実行します。ただし、SEmn と SOEmn が 1 の場合、SOm への書き込みが無視さるため、直前までに設定された SEmn と SOEmn の状態に依存します。

初期化時には、前の状態に依存するため、STm 制御によるクロック出力停止と SOEm 制御によるデータ出力停止の後（※２）に、再度 SOm を設定します。

送信／受信の完了時には、SIO_TX_DISABLE()/SIO_TRX_DISABLE()により、STm 制御によるクロック出力停止と SOEm 制御によるデータ出力停止状態になるため、SOm が設定されます。

※２：STm 制御によるクロック出力停止、SOEm 制御によるデータ出力停止後に、SOm を設定します。

このため、確実に SOm が設定されます。

5.9.10 マクロ関数　SIO_TX_ENABLE()

(1)　目的

シリアル I/O を送信許可にします。

(2)　機能

ハードウェアマニュアルにしたがって、シリアル I/O を送信許可設定します。端子をポート機能からシリアル I/O 機能への切り替え後、送信許可設定します。必要に応じて、処理を見直してください。

本処理では、SIO_ENABLE()の後の初期化手順から、送信設定専用の初期化処理を行います。RL78 ファミリ MCU の場合、以下の処理を行います。

①動作モードを送信に設定する。

SCRmn の TEXmn=1b, REXmn=0b を設定し、送信を許可する。

②端子をシリアル I/O 機能に切り替える。

・SOm をデータ出力：H、クロック出力：H に設定し、端子から出力する。

・SOEm の SOEmn=1b を設定し、シリアル通信動作によるデータ出力を許可する。

⇒通信動作によって反映された値が、シリアルデータ出力端子から出力される。

③シリアル通信動作許可状態に設定する。

SSm の SSmn=1b を設定する。

⇒SEmn ビットが”1”にセットされ、シリアル通信によるクロック出力が許可になる。

⇒通信動作によって反映された値が、シリアル・クロック出力端子から出力される。

(3)　備考

SIO_TX_DISABLE()と対となるものです。本関数を実行した後は、SIO_TX_DISABLE()を実行して、処理を終了してください。

本関数実行前には、SIO_DISABLE()/SIO_TX_DISABLE()/SIO_TRX_DISABLE()（STm 制御による通信動作停止）のいずれかを実行し、通信動作を停止させてください。

シリアル通信出力停止状態の出力端子は、シリアル出力レジスタ(SOm)とポート・レジスタ(Pxx)で設定した出力ラッチの論理積(AND)によって決まります。本関数実行前に、一度、SIO_DISABLE()と SIO_IO_INIT()を実行し、シリアル出力レジスタ(SOm)の該当する CKOmn ビット、SOmn ビットとポート・レジスタ(Pxx)を”1”に設定してください。

シリアル・アレイ・ユニットのレジスタ操作のため、事前に、PERn（n は、レジスタ番号）の SAUmEN 制御によるクロック供給を許可してください。

5.9.11 マクロ関数　SIO_TX_DISABLE()

(1)　目的

シリアル I/O の送信機能を停止します。

(2)　機能

SIO_TX_ENABLE()の処理の逆手順により、送信機能を停止します。送信停止設定処理後、端子をシリアル I/O 機能からポート機能へ切り替えます。必要に応じて、処理を見直してください。RL78 ファミリ MCU の場合、以下の処理を行います。

①シリアル通信動作停止状態に設定する。

STm の STmn=1b を設定する。

⇒SEmn ビットが”0”にクリアされ、シリアル通信動作によるクロック出力を停止する。

⇒チャネルが動作停止状態になる。

⇒SOm の CKOmn ビットに設定された値が、シリアル・クロック出力端子から出力される。

②シリアル通信動作による出力を停止する。

SOEm の SOEmn=0b を設定し、シリアル通信動作によるデータ出力を停止する。

⇒SOm の SOmn ビットに設定された値が、シリアルデータ出力端子から出力される。

③動作モードを通信禁止に設定する。

SCRmn の TEXmn=0b, REXmn=0b を設定し、通信を禁止する。

(3)　備考

SIO_TX_ENABLE()と対となるものです。SIO_TX_ENABLE()を実行した後は、本関数を実行して、処理を終了してください。

シリアル通信出力停止状態の出力端子は、シリアル出力レジスタ(SOm)とポート・レジスタ(Pxx)で設定した出力ラッチの論理積(AND)によって決まります。本関数実行前に、一度、SIO_DISABLE()と SIO_IO_INIT()を実行し、シリアル出力レジスタ(SOm)の該当する CKOmn ビット、SOmn ビットとポート・レジスタ(Pxx)を”1”に設定してください。

シリアル・アレイ・ユニットのレジスタ操作のため、事前に、PERn（n は、レジスタ番号）の SAUmEN 制御によるクロック供給を許可してください。

5.9.12 マクロ関数　SIO_TRX_ENABLE()

(1)　目的

シリアル I/O を送受信許可にします。

(2)　機能

ハードウェアマニュアルにしたがって、シリアル I/O を送受信許可設定します。端子をポート機能からシリアル I/O 機能への切り替え後、送受信許可設定します。必要に応じて、処理を見直してください。

本処理では、SIO_ENABLE()の後の初期化手順から、送受信設定専用の初期化処理を設定します。RL78 ファミリ MCU の場合、以下の処理を行います。

①動作モードを送受信に設定する。

SCRmn の TEXmn=1b, REXmn=1b を設定し、送受信を許可する。

②端子をシリアル I/O 機能に切り替える。

・SOm をデータ出力：H、クロック出力：H に設定し、端子から出力する。

・SOEm の SOEmn=1b を設定し、シリアル通信動作によるデータ出力を許可する。

⇒通信動作によって反映された値が、シリアルデータ出力端子から出力される。

③シリアル通信動作許可状態に設定する。

SSm の SSmn=1b を設定する。

⇒SEmn ビットが”1”にセットされ、シリアル通信によるクロック出力が許可になる。

⇒通信動作によって反映された値が、シリアル・クロック出力端子から出力される。

(3)　備考

SIO_TRX_DISABLE()と対となるものです。本関数を実行した後は、SIO_TRX_DISABLE()を実行して、処理を終了してください。

本関数実行前には、SIO_DISABLE()/SIO_TX_DISABLE()/SIO_TRX_DISABLE()（STm 制御による通信動作停止）のいずれかを実行し、通信動作を停止させてください。

シリアル通信出力停止状態の出力端子は、シリアル出力レジスタ(SOm)とポート・レジスタ(Pxx)で設定した出力ラッチの論理積(AND)によって決まります。本関数実行前に、一度、SIO_DISABLE()と SIO_IO_INIT()を実行し、シリアル出力レジスタ(SOm)の該当する CKOmn ビット、SOmn ビットとポート・レジスタ(Pxx)を”1”に設定してください。

シリアル・アレイ・ユニットのレジスタ操作のため、事前に、PERn（n は、レジスタ番号）の SAUmEN 制御によるクロック供給を許可してください。

5.9.13 マクロ関数　SIO_TRX_DISABLE()

(1)　目的

シリアル I/O の送受信機能を停止します。

(2)　機能

SIO_TRX_ENABLE()の処理の逆手順により、送受信機能を停止します。送受信停止設定処理後、端子をシリアル I/O 機能からポート機能へ切り替えます。必要に応じて、処理を見直してください。

RL78 ファミリ MCU の場合、以下の処理を行います。①シリアル通信動作停止状態に設定する。

STm の STmn=1b を設定する。

⇒SEmn ビットが”0”にクリアされ、シリアル通信動作によるクロック出力を停止する。

⇒チャネルが動作停止状態になる。

⇒SOm の CKOmn ビットに設定された値が、シリアル・クロック出力端子から出力される。

②シリアル通信動作による出力を停止する。

SOEm の SOEmn=0b を設定し、シリアル通信動作によるデータ出力を停止する。

⇒SOm の SOmn ビットに設定された値が、シリアルデータ出力端子から出力される。

③動作モードを通信禁止に設定する。

SCRmn の TEXmn=0b, REXmn=0b を設定し、通信を禁止する。

(3)　備考

SIO_TRX_ENABLE()と対となるものです。SIO_TRX_ENABLE()を実行した後は、本関数を実行して、処理を終了してください。

シリアル通信出力停止状態の出力端子は、シリアル出力レジスタ(SOm)とポート・レジスタ(Pxx)で設定した出力ラッチの論理積(AND)によって決まります。本関数実行前に、一度、SIO_DISABLE()と SIO_IO_INIT()を実行し、シリアル出力レジスタ(SOm)の該当する CKOmn ビット、SOmn ビットとポート・レジスタ(Pxx)を”1”に設定してください。

シリアル・アレイ・ユニットのレジスタ操作のため、事前に、PERn（n は、レジスタ番号）の SAUmEN 制御によるクロック供給を許可してください。

5.10 状態遷移図

図 5-11 に、状態遷移図を示します。

シリアル I/O 機能を有効化していない状態でのシリアル送受信は行わないでください。詳しくは「7.6 データ送信／データ受信時の禁止事項」を参照ください。

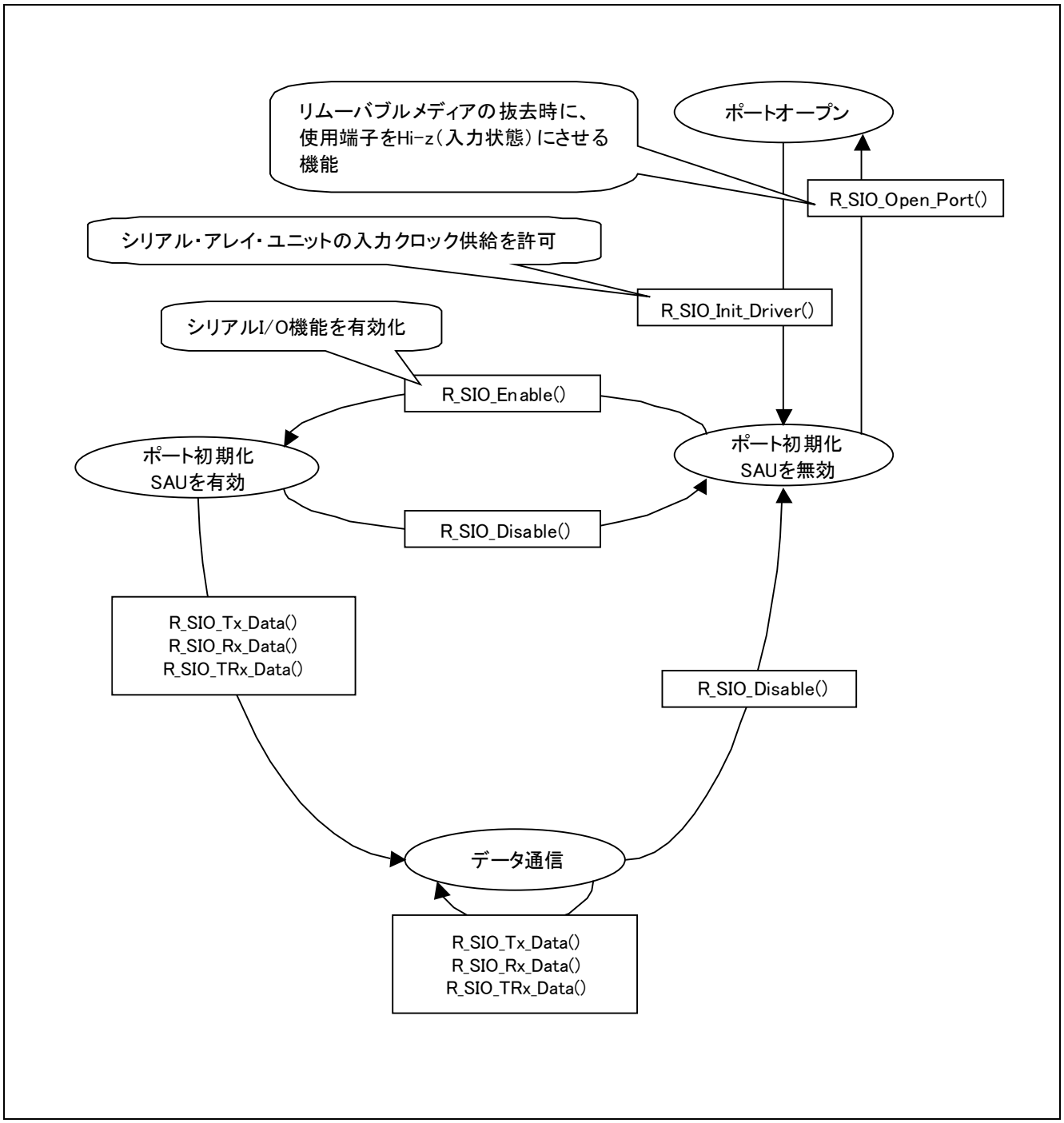


図 5-11　状態遷移図

6. 応用例

シリアル I/O 制御部分の設定例を示します。

使用する上での設定例を以下に示します。

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

6.1 mtl_com.h（共通ヘッダファイル）

共通で使用される共通関数のヘッダです。

mtl_com.h.XXX（mtl_com.h.common を除く）は、MCU 毎に評価目的で作成したものです。どれか一つを mtl_com.h にリネームして使用してください。対象 MCU のものが無い場合には、参照して、mtl_com.h を作成してください。

(1) OS のヘッダファイル定義

本サンプルコードは、OS 非依存です。

下記の例は、OS を使用しない場合の例です。

本サンプルコードは、OS 非依存のため、使用しない設定にしてください。他のソフトウェアに依存します。

```
/* システムコールを使用するため、                                          */
/* プロトタイプ宣言のある OS のヘッダファイルをインクルードしてください。  */
/* OS を使用しない場合は、下記デファインとインクルードをコメントにしてください。 */
//#define MTL_OS_USE                    /* Use OS                        */
//#include <RTOS.h>                     /* OS header file                */
//#include "mtl_os.h"
```

(2) 共通アクセス領域を定義したヘッダファイル定義

MCU の機能レジスタの定義がされているヘッダファイルをインクルードできます。

主にデバイスドライバがポート制御等に使用するため、インクルードする必要があります。

RL78 では別の方法でインクルード行うため、本ヘッダファイルではコメントアウトにしてください。

下記の例は、ヘッダファイルをインクルードしない場合の例です。

```
/* MCU の SFR 領域のデファイン値を使用しているため、                       */
/* I/O 周りのデファイン定義のあるヘッダファイルをインクルードしてください。 */
//#include "iodefine.h"                 /* definition of MCU SFR         */
```

(3) ループタイマの定義

ソフトウェア・ループタイマを使用する場合、以下のヘッダをインクルードします。

主にデバイスドライバが待ち時間を確保するために、使用します。

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

下記の例は、ソフトウェア・ループタイマを使用する場合の例です。

本サンプルコード使用時は、インクルードしてください。

```
/* ループタイマを使用しない場合は、下記インクルードをコメントにしてください。            */
#include "mtl_tim.h"
```

(4) エンディアンタイプ定義

リトルエンディアン／ビッグエンディアンのどちらかの指定が可能です。

RL78 ファミリ MCU の場合は、リトルエンディアンを定義してください。

```
/* MCU が、(1)SuperH でリトルエンディアン設定、(2)M16C  の場合、定義を有効にしてください。
  */
/* その他の MCU を指定する時はリトルエンディアンの定義をコメントにしてください。          */
#define MTL_MCU_LITTLE                  /* Little Endian                        */
```

(5) エンディアン処理の高速化の定義

mtl_end.c の処理の高速化指定が可能です。M16C を使用する場合、高速になります。

RL78 ファミリ MCU の場合は、コメントアウトし、定義しないでください。

```
/* M16C を使用する場合、定義を有効にしてください。                                    */
/* mtl_endi.c の処理の高速化が可能です。                                             */
//#define MTL_ENDI_HISPEED              /* Uses the high-speed function.     */
```

(6) 使用する標準ライブラリのタイプの定義

使用する標準ライブラリのタイプを定義してください。

下記に示す処理をコンパイラ添付のライブラリで使用する場合は、下記デファイン定義をコメントにしてください。

下記の例は、コンパイラ添付のライブラリを使用する場合の例です。

```
/* 使用する標準ライブラリのタイプを指定してください。                                */
/* 下記に示す処理をコンパイラ添付のライブラリで使用する場合は、                      */
/* 下記デファイン定義をコメントにしてください。                                      */
/* memcmp() / memmove() / memcpy() / memset() / strcat() / strcmp() / strcpy()
/ strlen()              */
//#define MTL_USER_LIB                    /* use optimized library          */
```

(7) アクセスする RAM 領域の定義

使用する RAM 領域を定義してください。

標準関数や一部の処理に効率の良い処理を適用します。

RL78 ファミリ MCU の場合は、MTL_MEM_NEAR を定義してください。

```
/* 使用する処理群がアクセスする RAM 領域を定義してください。                          */
/* 標準関数や一部の処理に効率の良い処理を適用します。                              */
//#define MTL_MEM_FAR    /* Defines 'FAR' as 'far' attribute for RAM area.(For M16C Family)*/
#define MTL_MEM_NEAR     /* No far/near attribute for RAM area.                   */
```

6.1.2 mtl_tim.h

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

使用する MCU、クロック、コンパイルオプション等に依存します。

また、命令キャッシュを有効にしているシステムでは、キャッシュ内に、ループタイマ処理が格納されている場合を想定して、設定してください。

使用環境に応じて、測定し直してください。

```
/* タイマのカウンタ値を定義してください。                                          */
/* MCU 及びクロック、ウエイトに応じて設定してください。                            */
/* Define the counter value for the timer.                                    */
/* Specify according to the user MCU, clock and wait requirements.            */
#if 1
/* Setting for 32MHz no wait (Compile Option : "-qx" at CubeSuite+ V1.01.01a,
CA78K0R Ver.1.30)*/
#define MTL_T_1US                          4     /* loop Number of   1us      */
#define MTL_T_2US                          8     /* loop Number of   2us      */
#define MTL_T_4US                         16     /* loop Number of   4us      */
#define MTL_T_5US                         20     /* loop Number of   5us      */
#define MTL_T_10US                        40     /* loop Number of  10us      */
#define MTL_T_20US                        80     /* loop Number of  20us      */
#define MTL_T_30US                       120     /* loop Number of  30us      */
#define MTL_T_50US                       200     /* loop Number of  50us      */
#define MTL_T_100US                      400     /* loop Number of 100us      */
#define MTL_T_200US                      800     /* loop Number of 200us      */
#define MTL_T_300US                     1200     /* loop Number of 300us      */
#define MTL_T_400US      ( MTL_T_200US   *  2 )   /* loop Number of 400us      */
#define MTL_T_1MS                       4000     /* loop Number of   1ms      */
#endif
```

上記は、未測定のため、妥当な値が設定されていませんので、評価してください。

6.2 クロック同期式シングルマスタ制御ソフトウェアの設定

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

6.2.1 R_SIO.h

(1) BRR 設定後のウェイトの定義

SAU の BRR 設定後、転送 1bit 期間をソフトウェア・ウェイトにより待ちます。ウェイト時間を設定してください。

初期値として、10μs を設定しています。

マルチメモリカードを使用する場合、100kHz 通信を想定し、10μs を設定してください。

```
#define SIO_T_BRR_WAIT        (uint16_t)MTL_T_10US /* BRR setting wait time */
```

RL78 ファミリ MCU の場合、BRR 設定後のウェイトは不要です。サンプルコード上、ウェイト処理を設定していないため、無視されます。

6.2.2 R_SIO_csi.h

SAU 用の定義ファイルです。

R_SIO_csi.h.XXX は、各 MCU に評価目的で作成したものです。どれか一つを R_SIO_csi.h にリネームして使用してください。対象 MCU のものが無い場合には、参照して、R_SIO_csi.h を作成してください。

(1) 使用する動作モードの定義

使用する MCU のリソースの設定が可能です。

下記は、MSB ファーストでの CRC-CCITT 演算処理を行う場合に、SIO_OPTION_2 を指定してください。

シリアル EEPROM、シリアルフラッシュメモリを制御する場合は、CRC-CCITT 演算処理は不要です。コメントアウトしてください。

また、マルチメディアカード制御にて CRC-CCITT 演算処理を行う場合、別途 R_SIO_csi_rx_mmc.c が必要です。

```
/*------------------------------------------------------------------------*/
/*  Define the combination of the MCU's resources.                        */
/*------------------------------------------------------------------------*/
#define SIO_OPTION_1        /* Low speed*/ /* SI/O                        */
//#define SIO_OPTION_2      /*          */ /* SI/O  + CRC calculation (S/W)  */
```

(2) 使用する CRC 演算処理の定義

使用する CRC 演算処理を定義してください。

シリアル EEPROM、シリアルフラッシュメモリを制御する場合は、CRC-CCITT 演算処理を使用しません。コメントアウトしてください。

マルチメディアカードを制御する場合、同時に両方を定義してください。

```
/*------------------------------------------------------------------------------*/
/*  Define the CRC calculation.                                                 */
/*------------------------------------------------------------------------------*/
#define SIO_CRCCCITT_USED     /* CRC-CCITT used                                 */
#define SIO_CRC7_USED         /* CRC7 used                                      */
```

(3) 使用する端子の定義

使用する端子を指定してください。

以下は統合開発環境 CubeSuite+（ルネサス エレクトロニクス製 RL78,78K0R コンパイラ CA78K0R）を使用した場合の例です。

```
/*------------------------------------------------------------------------------*/
/*  Define the control port.                                                    */
/*  Delete comment of a related macrodefinition, and please validate setting.
  */
/*------------------------------------------------------------------------------*/

#define SIO_PM_DATAO    PM14.4      /* SIO DataOut                              */
#define SIO_PM_DATAI    PM14.3      /* SIO DataIn                               */
#define SIO_PM_CLK      PM14.2      /* SIO CLK                                  */
#define SIO_P_DATAO     P14.4       /* SIO DataOut                              */
#define SIO_P_DATAI     P14.3       /* SIO DataIn                               */
#define SIO_P_CLK       P14.2       /* SIO CLK                                  */
#define SIO_PIM_DATAI   PIM14.3     /* SIO DataIn                               */
#define SIO_PIM_CLK     PIM14.2     /* SIO CLK                                  */
#define SIO_POM_DATAO   POM14.4     /* SIO DataOut                              */
#define SIO_POM_CLK     POM14.2     /* SIO CLK                                  */
```

(4) 周辺イネーブル・レジスタの定義

使用する SAU に関する周辺イネーブル・レジスタを指定してください。

```
#define SIO_SAUEN       SAU1EN      /* Control of CSI30 input clock supply */
```

(5) 使用する CSI チャネルの定義

使用する CSI チャネルを指定してください。以下は、統合開発環境 CubeSuite+（ルネサス エレクトロニクス製 RL78,78K0R コンパイラ CA78K0R）、CSI30 を使用する場合の例です。

```
/*----------------- SIO definitions -----------------*/

/* CSI30 setting example - Set the following for the system. */
#define SIO_SPS      SPS1     /* Serial clock select register              */
#define SIO_SMR      SMR12    /* Serial mode register                      */
#define SIO_SCR      SCR12 /* Serial communication operation setting register
  */
#define SIO_SDR      SDR12    /* Serial data register                      */
#define SIO_TXBUF    SIO30    /* SIOp data register                        */
#define SIO_RXBUF    SIO30    /* SIOp data register                        */
#define SIO_SIR      SIR12    /* Serial flag clear trigger register        */
#define SIO_SSR      SSR12    /* Serial status register                    */
#define SIO_SS       SS1L.2   /* Serial channel start register SSmn        */
#define SIO_ST       ST1L.2   /* Serial channel stop register STmn         */
#define SIO_SE       SE1L.2   /* Serial channel enable status register SEmn*/
#define SIO_SOE      SOE1L.2  /* Serial output enable register SOEmn       */
#define SIO_SO       SO1      /* Serial output register SOmn               */
#define SIO_SOL      SOL1     /* Serial output level register              */
#define SIO_SNFEN    NFEN0.6  /* Use of noise filter of RXD pin SNFEN      */

#define SIO_TXNEXT   (SSR12L & 0x20)/* CSI Transmit data empty             */
#define SIO_RXNEXT   IF1H.4   /* CSI Receive completion                    */
#define SIO_TXEND    IF1H.4   /* CSI Transmit completion                   */
```

(6) シリアル・クロック選択レジスタ(SPSm)による使用する動作クロック選択の定義

シリアル・クロック選択レジスタ(SPSm)の動作クロック選択を指定してください。以下は、CKm0 を使用する場合の例です。

```
#define SIO_USPS_INIT      (uint16_t)0x0000
                 /* 0000000000000000B */ /* SPS CSI initial setting       */
                 /* ||||||||||||++++-- CKm0:No division of fclk           */
                 /* ||||||||++++------ CKm1:No division of fclk           */
                 /* ++++++++---------- Reserved      : 0   Fixed         */
```

(7) 使用するチャネルの動作クロック(fMCK)選択の定義

シリアル・モード・レジスタ(SMRmn)の CKSmn ビットに使用するチャネルの動作クロック(fMCK)選択を指定してください。以下は、CKm0 を使用する場合の例です。

```
#define SIO_USMR_INIT       (uint16_t)0x0020
            /* 0000000000100000B */ /* SMR CSI initial setting              */
            /* ||||||||||||||||+-- Interrupt source : Transfer end interrupt */
            /* ||||||||||||||++--- Operation mode : CSI mode                 */
            /* ||||||||||||++----- Reserved       : 0      Fixed             */
            /* ||||||||||+-------- Reserved       : 1      Fixed             */
            /* ||||||||+---------- Reserved (Controls in UART mode)         */
            /* ||||||||+---------- Reserved       : 0      Fixed             */
            /* |||||||+----------- Start trigger source : Software trigger   */
            /* ||+++++------------ Reserved       : 0      Fixed             */
            /* |+----------------- ftclk clock channel setting   : Divided fmck
  */
            /* +------------------ fMCK clock channel setting : CKm0 set      */
```

(8) 使用するシリアル出力の出力値の定義

使用するチャネルのシリアル出力レジスタを 1 に設定してください。

そのため、使用するチャネルのシリアル出力レジスタ(SOm)の SOmn ビットを 1 に設定してください。1 に設定された箇所の SOmn ビットを 1 に設定します。以下は、CSI01 のデータ出力端子、クロック出力端子を使用する場合の例です。

```
#define SIO_USO_INIT        (uint16_t)0x0404                                */
            /* 0000010000000100B */ /* SO0m initial setting                */
            /* ||||||||||||||||+-- SOm0  output                            */
            /* |||||||||||||||+--- SOm1  output                            */
            /* ||||||||||||||+---- SOm2  output                            */
            /* |||||||||||||+----- SOm3  output                            */
            /* ||||||||++++------- Reserved       : 0   Fixed              */
            /* |||||||+----------- CKOm0 output                            */
            /* ||||||+------------ CKOm1 output                            */
            /* |||||+------------- CKOm2 output                            */
            /* ||||+-------------- CKOm3 output                            */
            /* ++++--------------- Reserved       : 0   Fixed              */
```

(9)　シリアル出力レベル・レジスタ(SOLm)の定義

CSI モードで使用する場合は、反転設定は禁止されています。対象の SOLmn ビットと予約ビットに 0 を書き込むために、1 を設定してください。

1 を設定する理由は、サンプルコードに、設定値を反転させ、1 に設定された箇所に 0 を書き込む処理が含まれているためです。

以下の例は、CSI30 を使用する場合の例です。

```
#define SIO_USOL_INIT       (unit16_t)0xFFFE
            /* 111111111111110B  */ /* SOLm initial setting(CSI mode setting)
  */
            /* ||||||||||||||+-- SOLm0 Communication data is output :      */
            /* ||||||||||||||+--- Reserved    : 1      Fixed               */
            /* ||||||||||||||+---- SOLm2 Communication data is output :    */
            /* ++++++++++++++----- Reserved    : 1      Fixed              */

            /* Caution: Refer to the application note for Setting method.  */
            /*          Set Unit/Channel No. and reserved bit to use to 1. */
            /*          Because 0 is written to a register by setting 1.   */
```

(10) ポート入力モード・レジスタ(PIM)、ポート出力モード・レジスタ(POM)の定義

使用する端子にしたがって、PIM、POM 設定を定義してください。

```
/*------------------ DataIn control ------------------*/
#define SIO_DATAI_INIT() do {       /* DataIn initial setting          */ ¥
    SIO_SNFEN      = 0;             /* Noise filter OFF                */ ¥
    SIO_PIM_DATAI   = 0; /** SET **/ /* Normal input buffer             */ ¥
    SIO_PM_DATAI    = 1;             /* DataIn          Input           */ ¥
} while (0)

/*----------------- DataOut control ------------------*/
#define SIO_DATAO_INIT() do {       /* DataOut initial setting         */ ¥
    SIO_POM_DATAO   = 0; /** SET **/ /* Normal output mode              */ ¥
    SIO_P_DATAO    = SIO_HI;        /* DataOut         "H"             */ ¥
    SIO_PM_DATAO    = 0;            /* DataOut         Output          */ ¥
    SIO_P_DATAO    = SIO_HI;        /* DataOut         "H"             */ ¥
} while (0)

#define SIO_DATAO_OPEN() do {       /* DataOut open setting            */ ¥
    SIO_PM_DATAO    = 1;            /* DataOut         Input           */ ¥
} while (0)

/*------------------- CLK control ---------------------*/
#define SIO_CLK_INIT() do {         /* CLK initial setting             */ ¥
    SIO_POM_CLK    = 0;  /** SET **/ /* Normal output mode             */ ¥
    SIO_P_CLK      = SIO_HI;        /* CLK             "H"             */ ¥
    SIO_PM_CLK     = 0;             /* CLK             Output          */ ¥
    SIO_P_CLK      = SIO_HI;        /* CLK             "H"             */ ¥
} while (0)

#define SIO_CLK_OPEN() do {         /* CLK open setting                */ ¥
    SIO_PIM_CLK    = 0;  /** SET **/ /* Normal input buffer            */ ¥
    SIO_PM_CLK     = 1;             /* CLK             Input           */ ¥
} while (0)
```

6.3 R_SIO_csi.c

R_SIO_csi.c.rl78g23 は、RL78/G23 用に評価目的で作成したものです。RL78/G23 を使用する場合は、R_SIO_csi.c と置き換えて使用してください。

使用する上での設定例を以下に示します。

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

(1) SFR 領域用デファインの設定

使用する C コンパイラには、定義済プリプロセッサシンボルがあります。この定義済プリプロセッサシンボルを使用し、プログラムを記述済です。

また、IAR Systems 社製の統合開発環境を使用する場合には、使用する MCU の SFR が定義されているヘッダファイルを設定する必要があります。

表 6-1 SFR 領域用デファインの設定

統合開発環境	SFR 設定の要・不要	設定方法
CubeSuite+ CS+	不要	不要
IAR Embedded Workbench	要	#ifdef __ICCRL78__ #include <ior5f104pj.h> ←MCU に合わせて変更 #include <ior5f104pj_ext.h> ←MCU に合わせて変更 #endif
e^2 studio	不要	不要

以下は、RL78/G14 100pin を使用する場合の例です。

```
#ifdef __ICCRL78__                      /* IAR RL78 Compiler                */
   #include <ior5f104pj.h>              /* for RL78/G14 100pin (R5F104PJ)   */
   #include <ior5f104pj_ext.h>          /* for RL78/G14 100pin (R5F104PJ)   */
#endif   /* __ICCRL78__ */
```

7. 使用上の注意事項

7.1 組み込み時の注意事項

本サンプルコードを組み込む場合は、R_SIO.h、R_SIO_csi.h（R_SIO_csi.h.XXX をリネーム）をインクルードしてください。

e² studio（CC-RL コンパイラ）でスマート・コンフィグレータを使用する場合は、iodefine.h をインクルードしてください。RL78/G23 の場合、パスは以下になります。

“/src/smc_gen/r_bsp/mcu/rl78_g23/register_access/ccrl”

7.2 不必要な関数について

使用されない関数は、ROM を不必要に消費しますので、コメントアウト等の処理にて、組み込まれないことを推奨します。

7.3 他 MCU を使用する場合

他 MCU を使用する場合、容易に対応が可能です。

準備するファイルは、

- R_SIO_csi.h.XXX に相当する I/O モジュール共通定義
- mtl_com.h.XXX に相当するヘッダ定義

です。添付のものを参考に、作成してください。

7.4 シリアル・データ及びクロック出力端子のポート制御方法

本端子をポート機能で使用する場合、シリアル出力レジスタ(SOm)の CKOmn ビット、SOmn ビットを”1”に設定してください。本端子の出力は、シリアル出力レジスタ(SOm)とポートレジスタ(Pxx)で設定した出力ラッチの論理積(AND)によって決まります。CKOmn ビット、SOmn ビットを”1”に設定することで、ポート・レジスタ(Pxx)で設定した値を、そのままポート出力値にすることができます。

7.5 シリアル・アレイ・ユニットのクロック供給の供給／停止制御について

本サンプルコードは、シリアル I/O 許可設定処理（R_SIO_Enable()）にてクロック供給開始制御を行いますが、シリアル I/O 禁止設定処理（R_SIO_Disable()）ではクロック供給停止制御を行いません。これは他のプログラムが同一ユニット内の他チャネルを使用している可能性があるためです。

低消費電力化とノイズ低減を図るためにユニット単位での動作停止制御を行う場合、本サンプルコードで使用する以外のチャネルの制御を考慮し、ユーザプログラムで制御してください。

なお、本サンプルコードは、チャネルの動作停止制御が可能です。

7.6 データ送信／データ受信時の禁止事項

シリアル I/O 機能を有効化していない状態でのシリアル送受信は行わないでください。

本サンプルコードは、ドライバ初期化処理（R_SIO_Init_Driver()）を行うと、シリアル・アレイ・ユニットへのクロック供給を開始します。この状態でシリアル I/O データ送信処理（R_SIO_Tx_Data()）またはシリアル I/O データ受信処理（R_SIO_Rx_Data()）を実行すると、シリアル I/O 機能に対して正しいレジスタ設定がされていないにも関わらず、送信処理／受信処理を開始します。この状態ではボー・レート等のレジスタ設定が正しく行われていないため、正常な送受信処理ができません。

シリアル I/O データ送信処理（R_SIO_Tx_Data()）、またはシリアル I/O データ受信処理（R_SIO_Rx_Data()）を行う際は、一度、シリアル I/O 許可設定処理（R_SIO_Enable()）を実行し、シリアル I/O に対してレジスタ設定してください。また、「5.10 状態遷移図」を参照してください。

7.7 シリアル出力レベル・レジスタ(SOLm)の設定について

CSI モードで使用する場合は、反転設定は禁止されています。対象の SOLmn ビットと予約ビットに 0 を書き込むために、1 を設定してください。

1 を設定する理由は、サンプルコードに、設定値を反転させ、1 に設定された箇所に 0 を書き込む処理が含まれているためです。

詳しくは、「6.2.2(9) シリアル出力レベル・レジスタ(SOLm)の定義」を参照してください。

7.8 型宣言の重複によるワーニングについて

本ドライバでは stdint.h で宣言される intN_t、uintN_t（N はビット長）を定義しています。stdint.h をインクルードした場合や、既に型を定義している場合、コンパイル時にワーニングが発生する可能性があります。本ドライバにある型宣言が不要な場合には、定義を削除してください。

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/contact/

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.02	2012.08.31	—	初版発行
1.03	2013.11.29	6	2. 動作確認条件　に、以下を追加した。 (2) RL78/G14 SAU 統合開発環境 IAR Embedded Workbench の場合
		7	3. 関連アプリケーションノート　に、以下を追加した。 Micron Technology 社製 P5Q Serial Phase Change Memory 制御ソフトウェア（R01AN1439JJ） Micron Technology 社製 N25Q Serial NOR Flash Memory 制御ソフトウェア（R01AN1528JJ） Spansion 社製 S25FLxxxS MirrorBit® Flash Non-Volatile Memory 制御ソフトウェア（R01AN1529JJ）
		13	5.3 必要メモリサイズ　に、以下を追加した。 (2) RL78/G14 SAU 統合開発環境 IAR Embedded Workbench の場合
		14	5.4 ファイル構成 アプリケーションノート番号を変更 フォルダ名を変更
		41,42	6.2.2 (3),(5)　内容を修正した。
		46	6.3 R_SIO_csi.c　新規に追加した。
		47	元 7.4 SFR 領域操作方法を削除した。
1.04	2014.4.30	—	MCU 追加に伴い、タイトルの対象グループ名を変更。　元は、「RL78/G14 グループ」であった。
		—	アプリケーションノート番号、作成日付け、Rev を更新
		1	要旨　の対象 MCU　元は、「RL78/G14 グループ」であった。
		1	最新のスレーブデバイス制御ソフトウェアの組み合わせ情報 URL を追加した。
		1	動作確認デバイス　元は、「RL78/G14 グループ」であった。
		1	「なお、以降では、対象デバイスが、複数グループ MCU であるため、説明の都合上、“RL78 ファミリ MCU”として記述しています。」を追記した。
		2-	「RL78 ファミリ MCU」　元は「RL78/G14」であった。
		5	2. 動作確認条件 (1) RL78/G14 SAU 統合開発環境 CubeSuite+の場合 動作確認表から　「エンディアン：リトルエンディアン」を削除。 (2) RL78/G14 SAU 統合開発環境 IAR Embedded Workbench の場合 動作確認表から　「エンディアン：リトルエンディアン」を削除。

		6 - 9	2. 動作確認条件　に、以下を追加した。 (3) RL78/G1C SAU 統合開発環境 CubeSuite+の場合 (4) RL78/G1C SAU 統合開発環境 IAR Embedded Workbench の場合 (5) RL78/L12 SAU 統合開発環境 CubeSuite+の場合 (6) RL78/L12 SAU 統合開発環境 IAR Embedded Workbench の場合 (7) RL78/L13 SAU 統合開発環境 CubeSuite+の場合 (8) RL78/L13 SAU 統合開発環境 IAR Embedded Workbench の場合 (9) RL78/L1C SAU 統合開発環境 CubeSuite+の場合 (10) RL78/L1C SAU 統合開発環境 IAR Embedded Workbench の場合
		16	5.3 必要メモリサイズ　に、以下を追加した。 「命令の異なる MCU 毎にメモリサイズを示します。使用 MCU の命令を調査し参考にてください。環境は、「2.動作確認条件」を参照してください。」
		17	5.3 必要メモリサイズ　に、以下を追加した。 (3) RL78/L13 SAU 統合開発環境 CubeSuite+の場合 (4) RL78/L13 SAU 統合開発環境 IAR Embedded Workbench の場合
		18	5.4 ファイル構成 アプリケーションノート番号を変更
		18	5.4 ファイル構成 R_SIO_csi.h.rl78g14　元は R_SIO_csi.h.rl78 であった。 以下のファイルを追加 R_SIO_csi.h.rl78g1c、R_SIO_csi.h.rl78l12、 R_SIO_csi.h.rl78g13、R_SIO_csi.h.rl78l1c、
		32	5.9.3 マクロ関数 SIO_DATAI_INIT() (2)機能　①を追加した。
		32	5.9.4 マクロ関数 SIO_DATAO_INIT() (2)機能　①を追加した。
		33	5.9.6 マクロ関数 SIO_CLK_INIT() (2)機能　①を追加した。
		50	6.3 R_SIO_csi.c IAR Systems 社　元は IAR 社であった。 表 6-1　より、RX 関連の記述を削除した。
1.05	2016.3.31	5	2.動作確認条件 (1)RL78/G14 SAU 統合開発環境 CS+ for CA, CX の動作確認環境を更新した。 (2) RL78/G14 SAU 統合開発環境 CS+ for CC を追加した。
		15	5.2 ソフトウェア制御概要　に、以下を追加した。 5.2.7 データ送受信（R_SIO_TRx_Data ()）
		17	5.3 必要メモリサイズ (1) RL78/G14 SAU 統合開発環境 CS+ for CA, CX の場合を更新した。 (2) RL78/G14 SAU 統合開発環境 CS+ for CC の場合を追加した。

		19	5.4 ファイル構成のアプリケーションノート番号を変更した。
		21	5.7 関数一覧　の表に、以下を追加した。 R_SIO_TRx_Data ()
		32	5.8 関数仕様　に、以下を追加した。 5.8.7 シリアル I/O データ送受信処理
		43	5.10 状態遷移図　に、以下を追加した。 R_SIO_TRx_Data ()
		55	7.8 型宣言の重複によるワーニングについて を追加した。
1.06	2021.7.14	1	タイトルと要旨に RL78/G23 を追加 文書のリンクを最新に変更 動作確認デバイスに MX25R1635F、MX25L3233F を追加 動作確認に使用した MCU に RL78/G23 を追加 動作確認に使用したデバイスに Micronix International 社製デバイスを追加
		11	2.動作確認条件 に、RL78/G23 の動作確認条件を追加
		12	3.関連アプリケーションノート に R01AN1967JJ のアプリケーションノートタイトルを追加
		20,21	5.3 必要メモリサイズに、RL78/G23 のメモリサイズを追加
		22	5.4 ファイル構成 に、RL78/G23 のファイルを追加
1.07	2022.8.4	12	2. 動作確認条件 (14) RL78/G1C SAU 統合開発環境 e^2 studio の場合（コンパイラ：LLVM）を追加
		22	5.3 必要メモリサイズ (8) RL78/G1C SAU 統合開発環境 e^2 studio の場合（コンパイラ：LLVM）を追加
		23	5.4 ファイル構成 サンプルコードのフォルダ名を変更 アプリケーションノート番号を更新
		57	6.3 (1) SFR 領域用デファインの設定 MCU の記載を削除 e^2 studio を追加
		58	7.1 組み込み時の注意事項 e2 studio（CC-RL コンパイラ）でスマート・コンフィグレータを使用する場合の注意事項を追加

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. 静電気対策
   CMOS 製品の取り扱いの際は静電気防止を心がけてください。CMOS 製品は強い静電気によってゲート絶縁破壊を生じることがあります。運搬や保存の際には、当社が出荷梱包に使用している導電性のトレーやマガジンケース、導電性の緩衝材、金属ケースなどを利用し、組み立て工程にはアースを施してください。プラスチック板上に放置したり、端子を触ったりしないでください。また、CMOS 製品を実装したボードについても同様の扱いをしてください。
2. 電源投入時の処置
   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. 電源オフ時における入力信号
   当該製品の電源がオフ状態のときに、入力信号や入出力プルアップ電源を入れないでください。入力信号や入出力プルアップ電源からの電流注入により、誤動作を引き起こしたり、異常電流が流れ内部素子を劣化させたりする場合があります。資料中に「電源オフ時における入力信号」についての記載のある製品は、その内容を守ってください。
4. 未使用端子の処理
   未使用端子は、「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。
5. クロックについて
   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。
6. 入力端子の印加波形
   入力ノイズや反射波による波形歪みは誤動作の原因になりますので注意してください。CMOS 製品の入力がノイズなどに起因して、V_{IL}（Max.）から V_{IH}（Min.）までの領域にとどまるような場合は、誤動作を引き起こす恐れがあります。入力レベルが固定の場合はもちろん、V_{IL}（Max.）から V_{IH}（Min.）までの領域を通過する遷移期間中にチャタリングノイズなどが入らないように使用してください。
7. リザーブアドレス（予約領域）のアクセス禁止
   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、将来の拡張機能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。
8. 製品間の相違について
   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、フラッシュメモリ、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、ノイズ幅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。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、お客様の責任において、お客様の機器・システムを設計ください。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. 当社製品または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. 当社製品を組み込んだ製品の輸出入、製造、販売、利用、配布その他の行為を行うにあたり、第三者保有の技術の利用に関するライセンスが必要となる場合、当該ライセンス取得の判断および取得はお客様の責任において行ってください。
5. 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。
6. 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。
7. あらゆる半導体製品は、外部攻撃からの安全性を 100%保証されているわけではありません。当社ハードウェア／ソフトウェア製品にはセキュリティ対策が組み込まれているものもありますが、これによって、当社は、セキュリティ脆弱性または侵害（当社製品または当社製品が使用されているシステムに対する不正アクセス・不正使用を含みますが、これに限りません。）から生じる責任を負うものではありません。当社は、当社製品または当社製品が使用されたあらゆるシステムが、不正な改変、攻撃、ウイルス、干渉、ハッキング、データの破壊または窃盗その他の不正な侵入行為（「脆弱性問題」といいます。）によって影響を受けないことを保証しません。当社は、脆弱性問題に起因しまたはこれに関連して生じた損害について、一切責任を負いません。また、法令において認められる限りにおいて、本資料および当社ハードウェア／ソフトウェア製品について、商品性および特定目的との合致に関する保証ならびに第三者の権利を侵害しないことの保証を含め、明示または黙示のいかなる保証も行いません。
8. 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。
9. 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。
10. 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。
11. 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。
12. 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。
13. 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。
14. 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.5.0-1　2020.10)

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。