RX ファミリ、RL78 ファミリ

Renesas R1EX24xxx シリーズ Serial EEPROM 制御ソフトウェア

R01AN1075JJ0103
Rev.1.03
2016.03.31

要旨

本アプリケーションノートでは、ルネサス エレクトロニクス製 MCU を使用したルネサス エレクトロニクス製 R1EV24xxx/R1EX24xxx/HN58X24xxx シリーズの I^2C Serial EEPROM 制御方法とサンプルコードの使用方法を説明します。

本サンプルコードでは、スレーブデバイスを制御するためのソフトウェアを上位層、I^2C シングルマスタ基本プロトコル制御を実現するソフトウェアを下位層としています。上位層は下位層で用意されているプロトコルを組み合わせてスレーブデバイスを制御します。

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

別途、マスタデバイスとしての MCU 個別の I^2C シングルマスタ制御のための下位層に位置するソフトウェアを用意していますので、以下から入手してください。なお、I^2C シングルマスタ制御ソフトウェアにおいて、新規 MCU が対応可能になった場合でも、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。最新の制御ソフトウェアの情報は、以下の URL のページに記載されている「I^2C シングルマスタドライバ（下位層ソフトウェア）」を参照してください。

- I^2C シリアル EEPROM 制御
  http://japan.renesas.com/driver/i2c_serial_eeprom

対象デバイス

Serial EEPROM

ルネサス エレクトロニクス製 R1EV24xxx/R1EX24xxx/HN58X24xxx シリーズ I^2C Serial EEPROM

動作確認に使用した MCU

RL78/G1x シリーズ	：RL78/G14、RL78/G1C（IICA を使用）
RL78/L1x シリーズ	：RL78/L12、RL78/L13、RL78/L1C（IICA を使用）
RX600 シリーズ	：RX62N、RX63N、RX63T（RIIC を使用）
RX200 シリーズ	：RX210、RX21A（RIIC を使用）

本アプリケーションノートを他のマイコンや他メモリへ適用する場合、そのマイコンやメモリの仕様に合わせて変更し、十分評価してください。

目次

1. 仕様

ルネサス エレクトロニクス製 MCU を使用したルネサス エレクトロニクス製 R1EV24xxx/R1EX24xxx/HN58X24xxx シリーズの I²C Serial EEPROM 制御方法とサンプルコードの使用方法を説明します。

本ソフトウェアと下位層にあたる I²C シングルマスタ制御ソフトウェアを使用することで、I²C Serial EEPROM と通信を行うことができます。本ソフトウェアは通信を制御するためのものであり、書き込み/読み出し等の処理は下位層のソフトウェアが行います。

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。

- マスタデバイスをルネサス エレクトロニクス製 MCU、スレーブデバイスをルネサス エレクトロニクス製 R1EV24xxx/R1EX24xxx/HN58X24xxx シリーズの I²C Serial EEPROM としたブロック型デバイスドライバです。
- I²C Serial EEPROM の Write 動作,Acknowledge polling（EEPROM 書き換え状態確認）,Read 動作をサポートします。
- EEPROM Write 動作は I²C シングルマスタ制御ソフトウェアのマスタ送信モード（パターン１）（※１）を使用します。送信完了後（ストップコンディション生成後）にバスを解放します。EEPROM のデータ書き換え完了の判定は Acknowledge polling で行います。
- Acknowledge polling は I²C シングルマスタ制御ソフトウェアのマスタ送信モード（パターン３）（※１）を使用します。スレーブアドレス送信時の ACK もしくは NACK 受信により、EEPROM への書き換え完了判定を行います。
- EEPROM Read 動作は I²C シングルマスタ制御ソフトウェアのマスタ複合モード（マスタ送信→マスタ受信）（※１）を使用します。
- 各制御を開始する開始関数と、通信を監視しプロトコル処理を進めるアドバンス関数により通信を実現します。通信開始後は、アドバンス関数をコールすることで通信を完了させる必要があります。
- 複数のチャネルをサポートします。
- １つのチャネル・バス上の複数かつ型名が異なるスレーブデバイスを制御できます。（※２）ただし、通信中（スタートコンディション生成から、ストップコンディション生成完了までの期間）は、そのデバイス以外の通信はできません。
- I²C Serial EEPROM は、使用する EEPROM の容量により、“スレーブデバイスを指定するアドレス”と“EEPROM 内部アドレス”の指定方法が異なります。本サンプルコードでは、必要な情報を設定することでアドレス変換を行います。
- ライトプロテクト（WP=H）状態制御機能をサポートしていません。ライトプロテクト状態であっても通信は開始しますが、データ送信時に EEPROM から NACK によりエラーが返ります。
- 通信中に通信が一定時間止まった場合や、ノイズ等の影響によりバスハングアップが発生した場合、EEPROM 復帰処理を行い、バスを開放することができます。

※１：詳細は、個別の I²C シングルマスタ制御ソフトウェアの仕様に依存します。
※２：Serial EEPROM は使用するデバイス容量により接続可能なスレーブデバイス数が異なります。１つのチャネル上に複数のスレーブデバイスを接続する場合はご注意ください。接続可能なスレーブデバイス数については表 5-2 を参照ください。

表 1-1　使用する周辺機器と用途

周辺機器	用途
MCU 内蔵の I^2C バス制御機能	I^2C バス通信機能 1ch（必須）

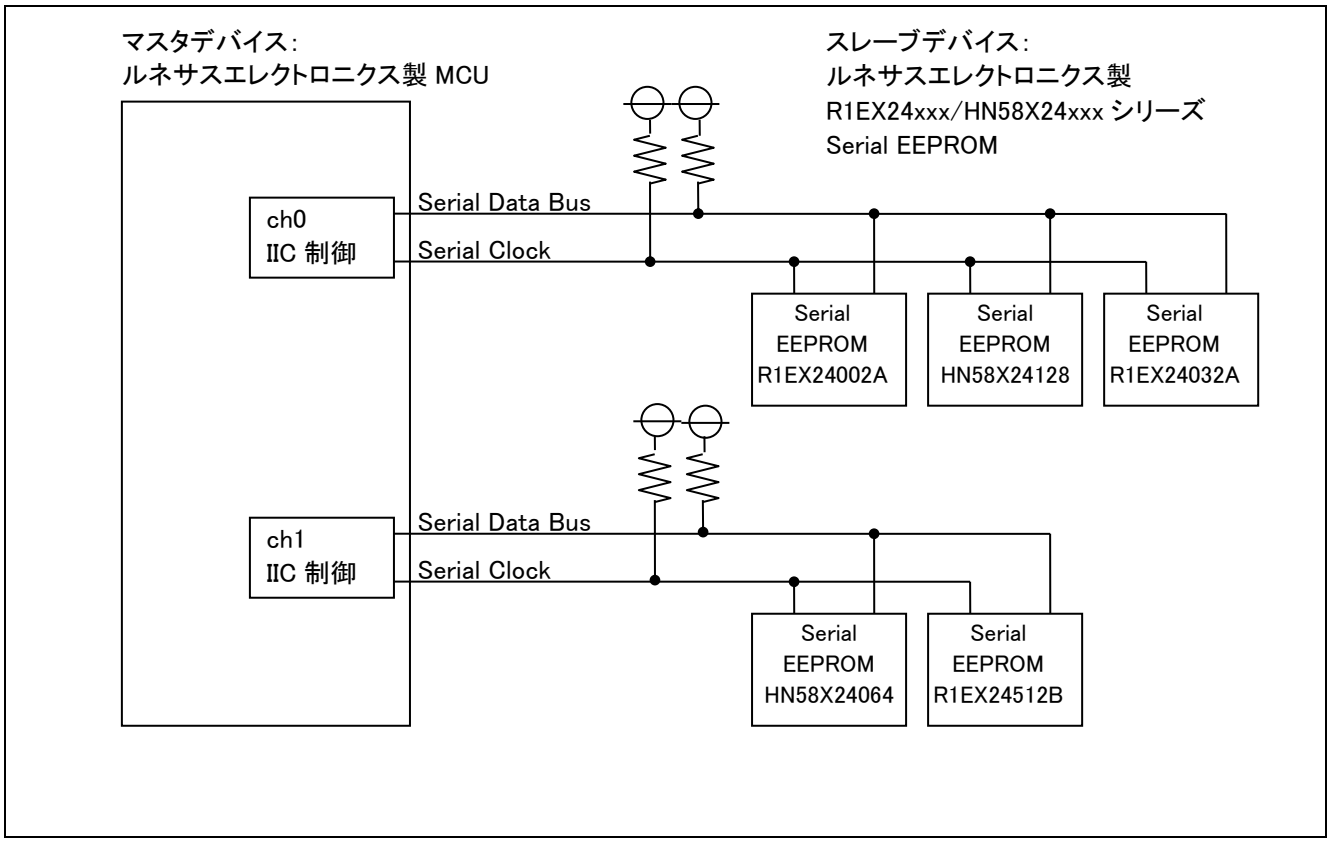


図 1-1　使用例

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。

2.1 RL78 の場合

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

表 2-1 動作確認条件

項目	内容
評価に使用したメモリ	ルネサス エレクトロニクス製 R1EX24xxx/HN58X24xxx シリーズ I^2C Serial EEPROM
評価に使用したマイコン	RL78/G14 グループ（プログラム ROM 256KB、RAM 24KB）
動作周波数（マイコン）	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、転送クロック：400KHz
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	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.1.01
評価に使用したソフトウェア	RL78/G14、RL78/G1C、RL78/L12、RL78/L13、RL78/L1C グループ IICA を使った I^2C シングルマスタ制御ソフトウェア (R01AN1074JJ) Ver.1.03
評価に使用したボード	Renesas Starter Kit for RL78/G14

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

表 2-2 動作確認条件

項目	内容
評価に使用したメモリ	ルネサス エレクトロニクス製 R1EX24xxx/HN58X24xxx シリーズ I^2C Serial EEPROM
評価に使用したマイコン	RL78/G14 グループ（プログラム ROM 256KB、RAM 24KB）
動作周波数（マイコン）	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、転送クロック：400KHz
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	ルネサス エレクトロニクス製 RL78 コンパイラ CC-RL
	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.1.01
評価に使用したソフトウェア	RL78/G14、RL78/G1C、RL78/L12、RL78/L13、RL78/L1C グループ IICA を使った I^2C シングルマスタ制御ソフトウェア (R01AN1074JJ) Ver.1.03
評価に使用したボード	Renesas Starter Kit for RL78/G14

(3) RL78/G14 IICA 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
評価に使用したメモリ	ルネサス エレクトロニクス製 R1EX24xxx/HN58X24xxx シリーズ I^2C Serial EEPROM
評価に使用したマイコン	RL78/G14 グループ（プログラム ROM 256KB、RAM 24KB）
動作周波数（マイコン）	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、転送クロック：400KHz
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.1.01
評価に使用したソフトウェア	RL78/G1x, RL78/L1x シリーズ IICA を使った I^2C シングルマスタ制御ソフトウェア(R01AN1074JJ) Ver.1.01
評価に使用したボード	Renesas Starter Kit for RL78/G14

(4) RL78/G1C IICA 統合開発環境 CubeSuite+の場合

表 2-4 動作確認条件

項目	内容
評価に使用したメモリ	ルネサス エレクトロニクス製 R1EX24xxx/HN58X24xxx シリーズ I^2C Serial EEPROM
評価に使用したマイコン	RL78/G1C（プログラム ROM 32KB、RAM 5.5KB）
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、転送クロック：400KHz
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 CubeSuite+ 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.1.01
評価に使用したソフトウェア	RL78/G1x, RL78/L1x シリーズ IICA を使った I^2C シングルマスタ制御ソフトウェア(R01AN1074JJ) Ver.1.02
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 IICA 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
評価に使用したメモリ	ルネサス エレクトロニクス製 R1EX24xxx/HN58X24xxx シリーズ I^2C Serial EEPROM
評価に使用したマイコン	RL78/G1C（プログラム ROM 32KB、RAM 5.5KB）
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、転送クロック：400KHz
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、アセンブラ	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.1.01
評価に使用したソフトウェア	RL78/G1x, RL78/L1x シリーズ IICA を使った I^2C シングルマスタ制御ソフトウェア(R01AN1074JJ) Ver.1.02
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 IICA 統合開発環境 CubeSuite+の場合

表 2-6 動作確認条件

項目	内容
評価に使用したメモリ	ルネサス エレクトロニクス製 R1EX24xxx/HN58X24xxx シリーズ I^2C Serial EEPROM
評価に使用したマイコン	RL78/L12（プログラム ROM 32KB、RAM 1.5KB）
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、転送クロック：400KHz
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 CubeSuite+ 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.1.01
評価に使用したソフトウェア	RL78/G1x, RL78/L1x シリーズ IICA を使った I^2C シングルマスタ制御ソフトウェア(R01AN1074JJ) Ver.1.02
評価に使用したボード	Renesas Starter Kit for RL78/L12

(7) RL78/L12 IICA 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
評価に使用したメモリ	ルネサス エレクトロニクス製 R1EX24xxx/HN58X24xxx シリーズ I^2C Serial EEPROM
評価に使用したマイコン	RL78/L12（プログラム ROM 32KB、RAM 1.5KB）
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、転送クロック：400KHz
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、アセンブラ	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.1.01
評価に使用したソフトウェア	RL78/G1x, RL78/L1x シリーズ IICA を使った I^2C シングルマスタ制御ソフトウェア(R01AN1074JJ) Ver.1.02
評価に使用したボード	Renesas Starter Kit for RL78/L12

(8) RL78/L13 IICA 統合開発環境 CubeSuite+の場合

表 2-8 動作確認条件

項目	内容
評価に使用したメモリ	ルネサス エレクトロニクス製 R1EX24xxx/HN58X24xxx シリーズ I^2C Serial EEPROM
評価に使用したマイコン	RL78/L13（プログラム ROM 128KB、RAM 8KB）
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、転送クロック：400KHz
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 CubeSuite+ 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.1.01
評価に使用したソフトウェア	RL78/G1x, RL78/L1x シリーズ IICA を使った I^2C シングルマスタ制御ソフトウェア(R01AN1074JJ) Ver.1.02
評価に使用したボード	Renesas Starter Kit for RL78/L13

(9) RL78/L13 IICA 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
評価に使用したメモリ	ルネサス エレクトロニクス製 R1EX24xxx/HN58X24xxx シリーズ I^2C Serial EEPROM
評価に使用したマイコン	RL78/L13（プログラム ROM 128KB、RAM 8KB）
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、転送クロック：400KHz
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、アセンブラ	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.1.01
評価に使用したソフトウェア	RL78/G1x, RL78/L1x シリーズ IICA を使った I^2C シングルマスタ制御ソフトウェア(R01AN1074JJ) Ver.1.02
評価に使用したボード	Renesas Starter Kit for RL78/L13

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

表 2-10 動作確認条件

項目	内容
評価に使用したメモリ	ルネサス エレクトロニクス製 R1EX24xxx/HN58X24xxx シリーズ I^2C Serial EEPROM
評価に使用したマイコン	RL78/L1C（プログラム ROM 256KB、RAM 16KB）
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、転送クロック：400KHz
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 CubeSuite+ 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.1.01
評価に使用したソフトウェア	RL78/G1x, RL78/L1x シリーズ IICA を使った I^2C シングルマスタ制御ソフトウェア(R01AN1074JJ) Ver.1.02
評価に使用したボード	Renesas Starter Kit for RL78/L1C

(11) RL78/L1C IICA 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
評価に使用したメモリ	ルネサス エレクトロニクス製 R1EX24xxx/HN58X24xxx シリーズ I^2C Serial EEPROM
評価に使用したマイコン	RL78/L1C（プログラム ROM 256KB、RAM 16KB）
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、転送クロック：400KHz
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、アセンブラ	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.1.01
評価に使用したソフトウェア	RL78/G1x, RL78/L1x シリーズ　IICA を使った I^2C シングルマスタ制御ソフトウェア(R01AN1074JJ)　Ver.1.02
評価に使用したボード	Renesas Starter Kit for RL78/L1C

2.2 RX の場合

(1) RX62N RIIC の場合

表 2-12 動作確認条件

項目	内容
評価に使用したメモリ	ルネサス エレクトロニクス製 R1EX24xxx/HN58X24xxx シリーズ I^2C Serial EEPROM
評価に使用したマイコン	RX62N グループ（プログラム ROM 512KB、RAM 64KB）
動作周波数	ICLK：96MHz、PCLK：48MHz、転送クロック：400kHz
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 High-performance embedded Workshop Version 4.09.01.007
Cコンパイラ、アセンブラ	ルネサスエレクトロニクス製 RX ファミリ用 C/C++コンパイラケッケージ（ツールチェーン 1.2.1.0）
	コンパイルオプション 総合開発環境のデフォルト設定（※１）を使用しています。 ※１：最適化レベル”2”、最適化方法”サイズ優先”
エンディアン	ビッグエンディアン／リトルエンディアン
サンプルコードのバージョン	Ver.1.00
評価に使用したソフトウェア	RX600, RX200 シリーズ　RIIC を使った I^2C シングルマスタ制御ソフトウェア（R01AN1254JJ）　Ver.1.13
評価に使用したボード	Renesas Starter Kit for RX62N

(2) RX63N RIIC の場合

表 2-13 動作確認条件

項目	内容
評価に使用したメモリ	ルネサス エレクトロニクス製 R1EX24xxx/HN58X24xxx シリーズ I^2C Serial EEPROM
評価に使用したマイコン	RX62N グループ（プログラム ROM 1MB、RAM 128KB）
動作周波数	ICLK：96MHz、PCLK：48MHz、転送クロック：400kHz
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 High-performance embedded Workshop Version 4.09.01.007
Cコンパイラ、アセンブラ	ルネサスエレクトロニクス製 RX ファミリ用 C/C++コンパイラケッケージ（ツールチェーン 1.2.1.0）
	コンパイルオプション 総合開発環境のデフォルト設定（※１）を使用しています。 ※１：最適化レベル”2”、最適化方法”サイズ優先”
エンディアン	ビッグエンディアン／リトルエンディアン
サンプルコードのバージョン	Ver.1.00
評価に使用したソフトウェア	RX600, RX200 シリーズ　RIIC を使った I^2C シングルマスタ制御ソフトウェア（R01AN1254JJ）　Ver.1.13
評価に使用したボード	Renesas Starter Kit for RX63N

(3) RX63T RIIC の場合

表 2-14 動作確認条件

項目	内容
評価に使用したメモリ	ルネサス エレクトロニクス製 R1EX24xxx/HN58X24xxx シリーズ I^2C Serial EEPROM
評価に使用したマイコン	RX63T グループ（プログラム ROM 512KB、RAM 48KB）
動作周波数	ICLK：96MHz、PCLK：48MHz、転送クロック：400kHz
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+ V2.00.00
Cコンパイラ、アセンブラ	ルネサスエレクトロニクス製 RX ファミリ用 C/C++コンパイラケッケージ（ツールチェーン 2.00.000）
	コンパイルオプション 総合開発環境のデフォルト設定（※１）を使用しています。 ※１：最適化レベル”2”、最適化方法”サイズ優先”
エンディアン	ビッグエンディアン／リトルエンディアン
サンプルコードのバージョン	Ver.1.00
評価に使用したソフトウェア	RX600, RX200 シリーズ　RIIC を使った I^2C シングルマスタ制御ソフトウェア（R01AN1254JJ）　Ver.1.13
評価に使用したボード	Renesas Starter Kit for RX63T

(4) RX210 RIIC の場合

表 2-15 動作確認条件

項目	内容
評価に使用したメモリ	ルネサス エレクトロニクス製 R1EX24xxx/HN58X24xxx シリーズ I^2C Serial EEPROM
評価に使用したマイコン	RX210 グループ（プログラム ROM 512KB、RAM 64KB）
動作周波数	ICLK：50MHz、PCLK：25MHz、転送クロック：400kHz
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 High-performance embedded Workshop Version 4.09.01.007
Cコンパイラ、アセンブラ	ルネサスエレクトロニクス製 RX ファミリ用 C/C++コンパイラケッケージ（ツールチェーン 1.2.1.0）
	コンパイルオプション 総合開発環境のデフォルト設定（※１）を使用しています。 ※１：最適化レベル”2”、最適化方法”サイズ優先”
エンディアン	ビッグエンディアン／リトルエンディアン
サンプルコードのバージョン	Ver.1.00
評価に使用したソフトウェア	RX600, RX200 シリーズ　RIIC を使った I^2C シングルマスタ制御ソフトウェア（R01AN1254JJ）　Ver.1.13
評価に使用したボード	Renesas Starter Kit for RX210

(5) RX21A RIIC の場合

表 2-16 動作確認条件

項目	内容
評価に使用したメモリ	ルネサス エレクトロニクス製 R1EX24xxx/HN58X24xxx シリーズ I^2C Serial EEPROM
評価に使用したマイコン	RX21A グループ（プログラム ROM 512KB、RAM 64KB）
動作周波数	ICLK：50MHz、PCLK：25MHz、転送クロック：400kHz
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 High-performance embedded Workshop Version 4.09.01.007
Cコンパイラ、アセンブラ	ルネサスエレクトロニクス製 RX ファミリ用 C/C++コンパイラケッケージ（ツールチェーン 1.2.1.0）
	コンパイルオプション 総合開発環境のデフォルト設定（※１）を使用しています。 ※１：最適化レベル”2”、最適化方法”サイズ優先”
エンディアン	ビッグエンディアン／リトルエンディアン
サンプルコードのバージョン	Ver.1.00
評価に使用したソフトウェア	RX600, RX200 シリーズ　RIIC を使った I^2C シングルマスタ制御ソフトウェア（R01AN1254JJ）　Ver.1.13
評価に使用したボード	Renesas Starter Kit for RX210

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。併せて参照してください。

- RL78/G14、RL78/G1C、RL78/L12、RL78/L13、RL78/L1C グループ　IICA を使った I^2C シングルマスタ制御ソフトウェア(R01AN1074JJ)
- RX600、RX200 シリーズ　RIIC を使った I^2C シングルマスタ制御ソフトウェア(R01AN1254JJ)

4. ハードウェア説明

4.1 使用端子一覧

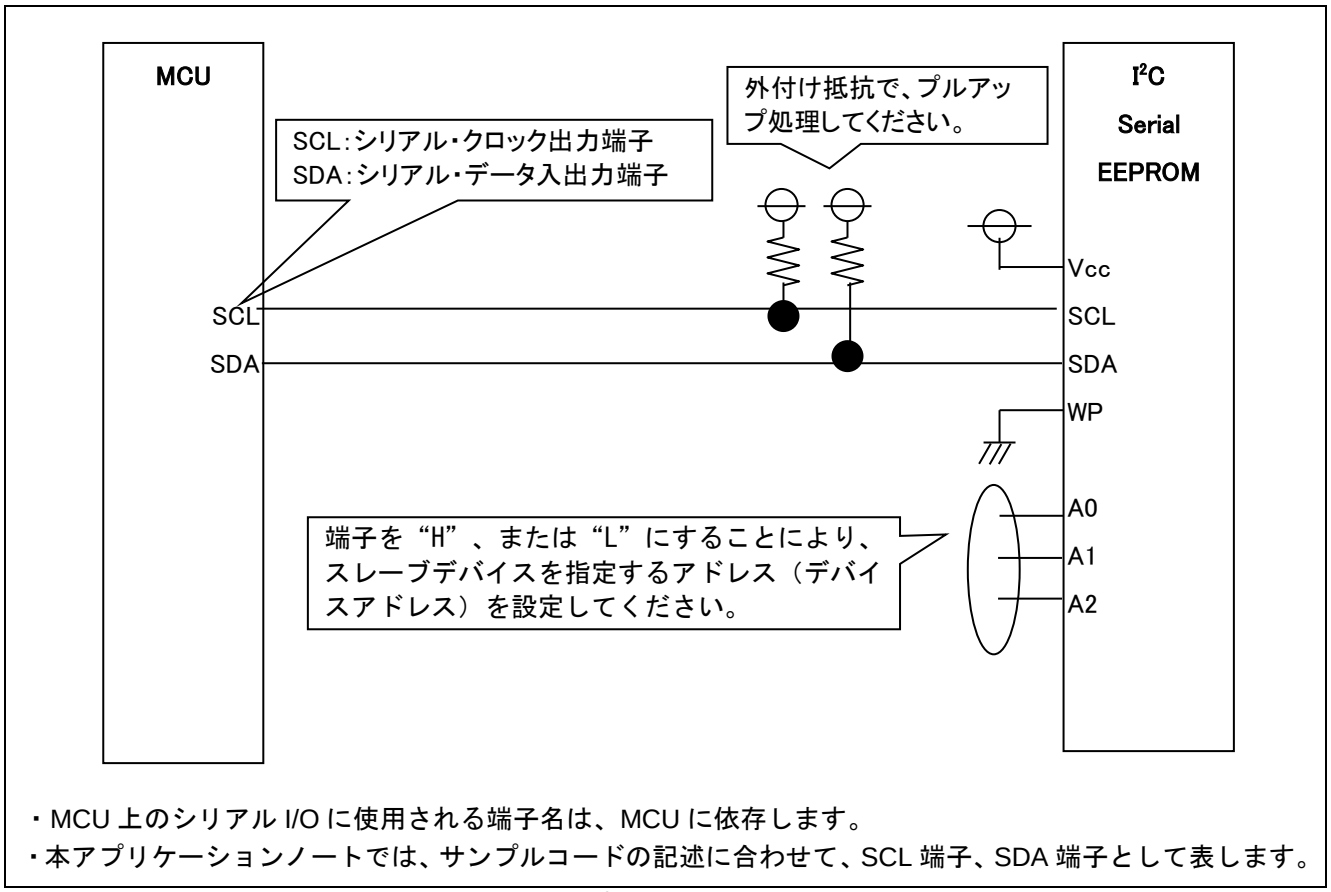
表 4-1 に、使用端子と機能を示します。

表 4-1 使用端子と機能

端子名	入出力	内容
SCL （図 4-1 の SCL）	出力	シリアル・クロック出力
SDA （図 4-1 の SDA）	入出力	シリアル・データ入出力

4.2 参考回路

図 4-1 に接続図を示します。シリアル・クロック・ラインおよびシリアル・データ・バス・ラインは、出力が N-ch オープン・ドレーンのため、外部にプルアップ抵抗が必要です。システムに応じて適正な抵抗を検討してください。また、各信号ラインの回路的マッチングを取るためのダンピング抵抗を検討してください。



・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、SCL 端子、SDA 端子として表します。

図 4-1 MCU と I²C Serial EEPROM の接続例

4.3 複数スレーブデバイス制御

本サンプルコードでは複数のチャネルをサポートします。また、１つのチャネル・バス上に、複数かつ型名が異なるスレーブデバイスを搭載し、制御できます。ただし、スタートコンディション生成から、ストップコンディション生成完了までの期間は、そのデバイス以外の通信はできません。

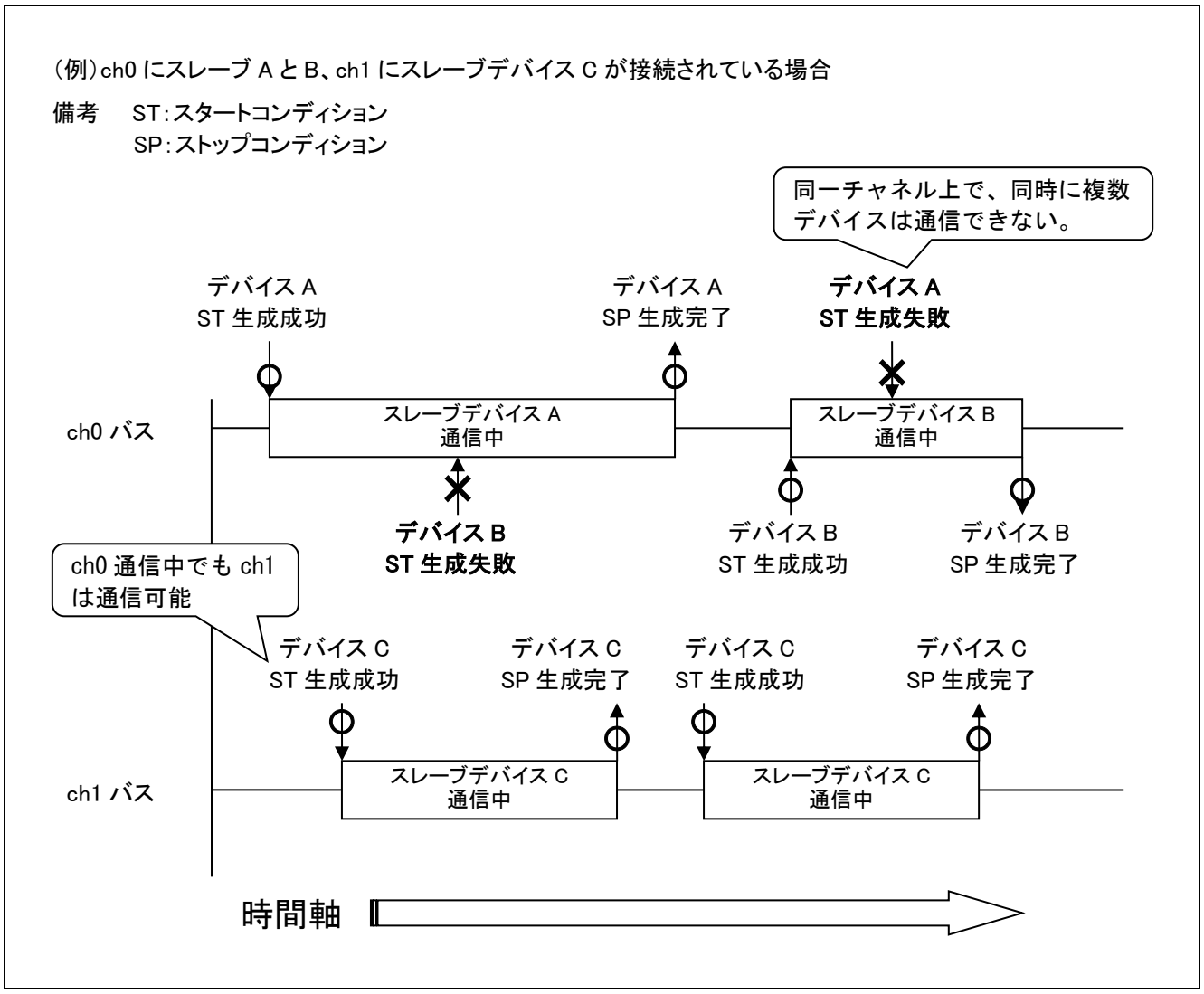


図 4-2　複数スレーブデバイス制御例

4.4 最大転送速度

最大 400kHz までの設定可能です。ただし、標準モード・デバイスとファーストモード・デバイスが混載されるバスでは、標準モードの最大 100kHz に設定する必要があります。

以下に、混載バス・システムでの最大転送速度を示します。

表 4-2　混載バス・システムでの最大転送速度

通信デバイス	混載デバイス	
	ファーストモード	標準モード
ファーストモード	0 – 400kHz	0 – 100kHz
標準モード	0 – 100kHz	0 – 100kHz

5. ソフトウェア説明

5.1 ソフトウェア構成

スレーブデバイスを制御するためのソフトウェアを上位層、I²C シングルマスタ基本プロトコル制御を実現するソフトウェアを下位層とします。上位層は下位層で用意されているプロトコルを組み合わせてスレーブデバイスを制御します。

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

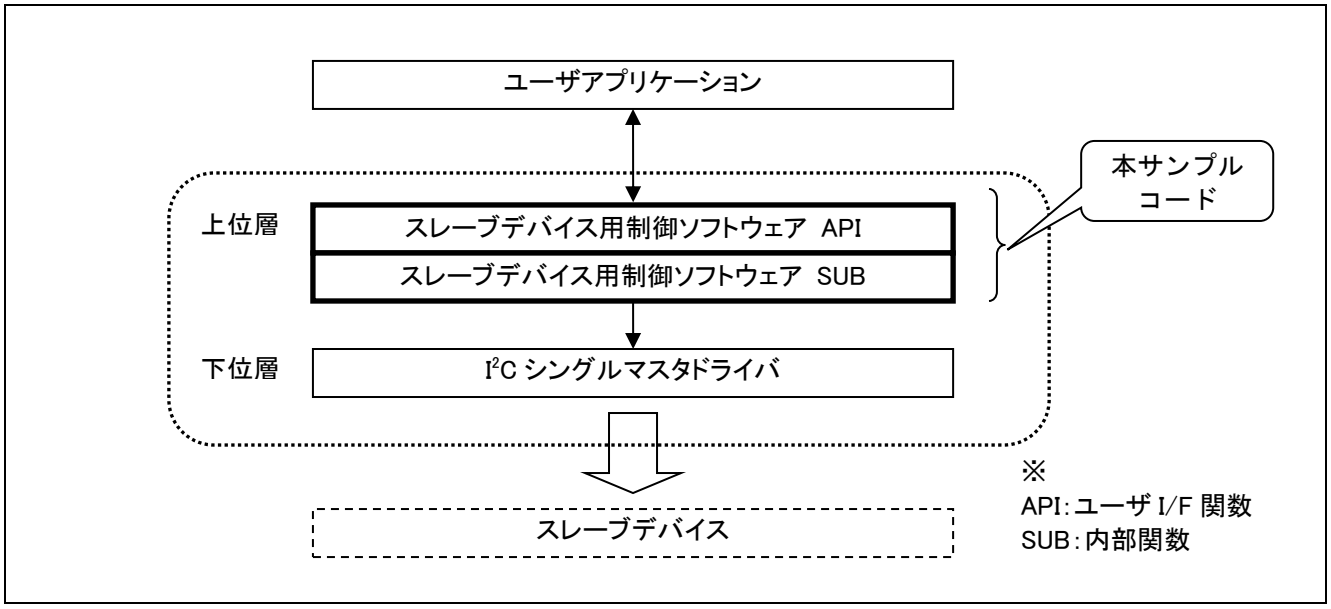


図 5-1 ソフトウェア構成

5.2 動作概要

本項では、MCU 内蔵の I²C バス制御機能を使用してシングルマスタ I²C バスのマスタとして、スレーブデバイスの EEPROM を制御するための方法を示します。

5.2.1 アドレス指定

I²C Serial EEPROM は、使用する EEPROM の容量により、“スレーブデバイスを指定するアドレス”と“EEPROM 内部アドレス”の指定方法が異なります。以下に、アドレスの指定方法を示します。

(1) スレーブデバイスを指定するアドレス

スタートコンディション生成後、EEPROM に対して 8 ビットのデータ（デバイス・アドレス・ワード）を送信します。EEPROM では、その上位 4 ビットは 1010 に固定されています。残りの 3 ビット（デバイスアドレスコード）については、使用するデバイスにより異なります。詳しくは表 5-2 を参照ください。

表 5-1 EEPROM のデバイス・アドレス・ワード

Device code（fixed）				Device address code			R/W code
bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
1	0	1	0	A2	A1	A0	R/W

(2) EEPROM の容量別アドレス指定

表 5-2 に対象とする EEPROM の仕様を示します。ここで、表中の Device address code 欄の‘A’はスレーブデバイスを指定するアドレス、‘a’は EEPROM 内部アドレスを示します。表中の Device address code 欄で a16 や a10～a8 と書かれているデバイスは、Device address code で EEPROM 内部アドレスの指定が必要です。

例えば、8K ビットの EEPROM の場合、Device address code の bit3（A2）は接続されている EEPROM を指定するアドレスを設定します。bit2（a9）と bit1（a8）は EEPROM 内部アドレスの上位ビットを指定します。仮に、EEPROM 内部アドレスを“11 1001 0101h”とした場合、上位 2 ビット（a9～a8）の値“11”が Device address code の bit2 と bit1 に設定する値になります。

表 5-2 EEPROM の容量別アドレス指定一覧（※1）

製品型名	容量(bit)	EEPROM 内部アドレス長	最大接続スレーブ数（個）	Device address code			EEPROM 内部アドレス	
				bit3	bit2	bit1	1 バイト目	2 バイト目
HN58W241000I	1M	17 ビット（a16～a0)	4	A2	A1	a16	a15～a8	a7～a0
R1EX24512B	512K	16 ビット（a15～a0)	8	A2	A1	A0	a15～a8	a7～a0
R1EX24512A HN58X24512	512K	16 ビット（a15～a0)	4	0 (※2)	A1	A0	a15～a8	a7～a0
R1EX24256B R1EX24256A HN58X24256	256K	15 ビット（a14～a0)	8	A2	A1	A0	a14～a8	a7～a0
R1EX24128B R1EX24128A HN58X24128	128K	14 ビット（a13～a0)	8	A2	A1	A0	a13～a8	a7～a0
R1EV24064A R1EX24064A HN58X2464	64K	13 ビット（a12～a0)	8	A2	A1	A0	a12～a8	a7～a0
R1EX24032A HN58X2432	32K	12 ビット（a11～a0)	8	A2	A1	A0	a11～a8	a7～a0
R1EX24016A HN58X2416	16K	11 ビット（a10～a0)	1	a10	a9	a8	a7～a0	
R1EX24008A HN58X2408	8K	10 ビット（a9～a0)	2	A2	a9	a8	a7～a0	
R1EV24004A R1EX24004A HN58X2404	4K	9 ビット（a8～a0)	4	A2	A1	a8	a7～a0	
R1EV24002A R1EX24002A HN58X2402	2K	8 ビット（a7～a0)	8	A2	A1	A0	a7～a0	

※1：本書 Rev.1.00 現在の EEPROM 製品一覧です。新規デバイスについては、そのデータシートを参照ください。

※2：Don’t care bit

5.2.2 Write 動作

Write 動作の前提として、本サンプルコードでは、マスタ（MCU）からスレーブ（EEPROM）にデータを送信することを“書き込み”と定義します。また、EEPROM へ書き込まれたデータを EEPROM 内部で書き換えることを“書き換え”と定義します。

本サンプルコードでは、Byte Write、Page Write（※）をサポートします。

※：使用する EEEPROM が定めるページサイズに対して、任意のバイト数を一度に書き換えられる機能。

(1) データ書き込み（WP=L 時）

I^2C シングルマスタ制御ソフトウェアのマスタ送信モード（パターン 1）を使用します。初めにスタートコンディション（ST）を生成し、次にスレーブデバイスのアドレスを送信します。このとき、8 ビット目は転送方向指定ビットになりますので、データ送信時には“0”（Write）を送信します。次に EEPROM 内部アドレスを送信し、データ送信を開始します。

ページサイズ内で書き換え可能なデータ数分、連続でデータ送信を行います。データ送信が完了すると、ストップコンディション（SP）を生成してバスを解放します。

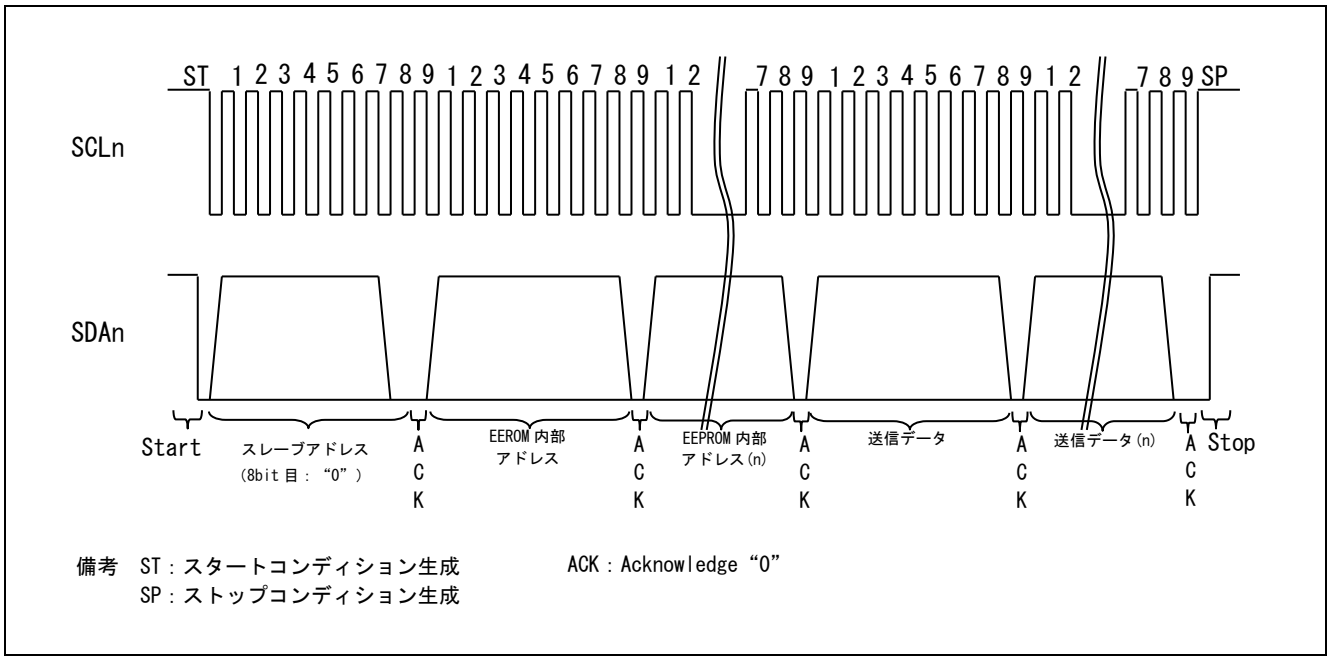


図 5-2 データ書き込み動作

(2) データ書き込み（WP=H 時）

ライトプロテクト端子（WP）が High 状態で Write 動作を行った場合、EEPROM のデータ書き換えはできません。

スタートコンディション→スレーブアドレス送信→EEPROM 内部アドレス送信では、9 ビットごとに Acknowledge“0”を受信するため正常動作しますが、データの送信後は Acknowledge“1”を受信し、NACK エラーとなります。NACK エラー後、ストップコンディションを生成し、バスを解放します。

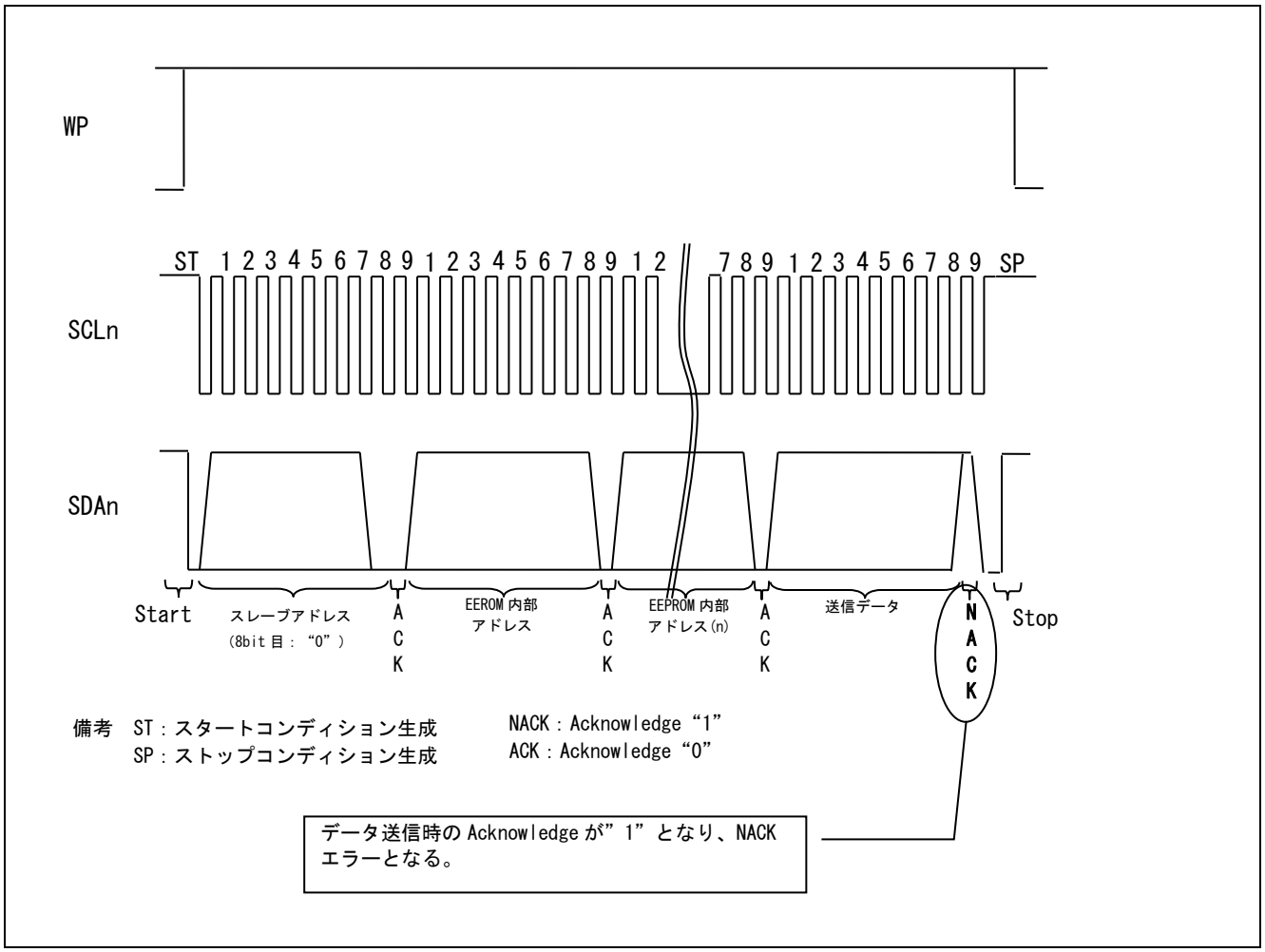


図 5-3 ライトプロテクト（WP=H）時の Write 動作

(3) Acknowledge polling

EEPROM は、データ書き込みが完了し、ストップコンディションが受信されると、書き換え動作に入ります。EEPROM の書き換え完了を判定する機能として、Acknowledge polling があります。

本サンプルコードでは、I^2C シングルマスタ制御ソフトウェアのマスタ送信モード（パターン 3）を使用します。初めにスタートコンディション（ST）を生成し、次にスレーブデバイスのアドレスを送信します。このときの 8 ビット目の転送方向指定ビットは“0”（Write）を送信します。9 ビット目の Acknowledge で書き換え完了の判断を行います。Acknowledge“1”（NACK）は書き換え中、Acknowledge“0”（ACK）は書き換え完了を示します。最後にストップコンディション（SP）を生成してバスを解放します。

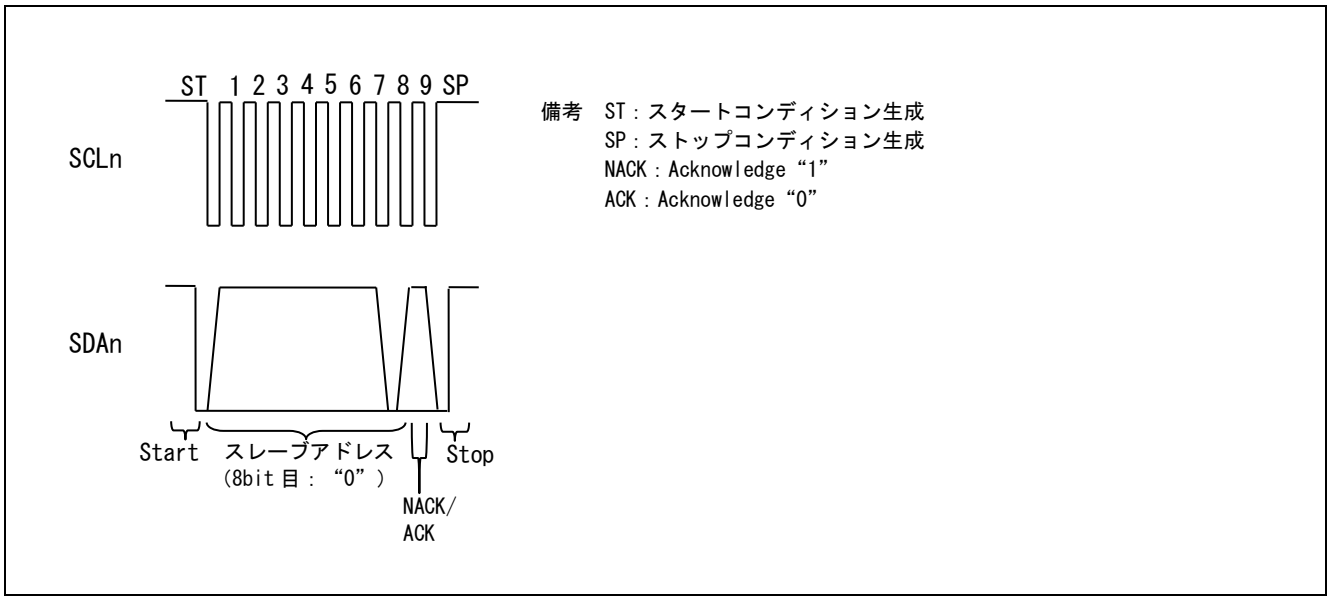


図 5-4 Acknowledge polling（書き換え完了判定）

(4) ブロック書き換え

本サンプルコードでは、書き換えたいデータの総数を指定すると、ソフトウェア内で書き込みサイズを計算しブロック書き換えを行います。ブロック書き換えについて、以下に説明します。

EEPROM は、一度に書き換えられるバイト数は 1 ページです。ページ内のアドレスがページの最終番地に達した場合は、アドレスは“Roll Over”して、ページの先頭に戻ります。このため、ページの境界を越えてデータを書き換える（ブロック書き換える）場合には、書き換えるデータがページ内に収まるように何回かに分けて書き換える必要があります。表 5-3 に EEPROM 容量別のページサイズを示します。このページサイズは使用する EEPROM によって異なります。使用するデバイスに合った設定を行ってください。

表 5-3　EEPROM の容量別ページサイズ一覧（※）

製品型名	ページサイズ
HN58W241000I	256 バイト
R1EX24512B、R1EX24512A、HN58X24512	128 バイト
R1EX24256B、R1EX24128B、R1EX24256A、R1EX24128A、HN58X24256 HN58X24128	64 バイト
R1EV24064A、R1EX24064A、R1EX24032A、HN58X2464、HN58X2432、HN58X2416、HN58X2408	32 バイト
R1EX24016A、R1EX24008A、R1EV24004A、R1EX24004A、R1EX24002A	16 バイト
R1EV24002A、HN58X2402、HN58X2404	8 バイト

※：本書 Rev.1.00 現在の EEPROM 製品一覧です。新規デバイスについては、そのデータシートを参照ください。

5.2.3 Read 動作

Read 動作の前提として、本サンプルコードでは、マスタ（MCU）がスレーブ（EEPROM）からデータを受信することを“読み出し”と定義します。

本サンプルコードでは、Random Read、Sequential Read をサポートします。

(1) データ読み出し

I^2C シングルマスタ制御ソフトウェアのマスタ複合モードを使用します。初めにスタートコンディション（ST）を生成し、次にスレーブデバイスのアドレスを送信します。このとき、8 ビット目は転送方向指定ビットになりますので、データ送信時には“0”（Write）を送信します。次に EEPROM 内部アドレスを送信します。次にリスタートコンディション（RST）を生成し、スレーブアドレスを送信します。このとき、転送方向指定ビットは“1”（Read）を送信します。次にデータ受信を開始します。

Random Read の場合、EEPROM から 8 ビットのデータを受信した後、MCU から EEPROM に Acknowledge “1”を送信し、ストップコンディション（SP）を生成してバスを解放します。

Sequential Read の場合、連続受信中はデータ受信後、Acknowledge “0”を送信します。これにより、EEPROM 内部アドレスはインクリメントされ、次のデータ受信が可能となります。最終データを受信後、MCU から EEPROM に Acknowledge “1”を送信し、ストップコンディション（SP）を生成してバスを解放します。

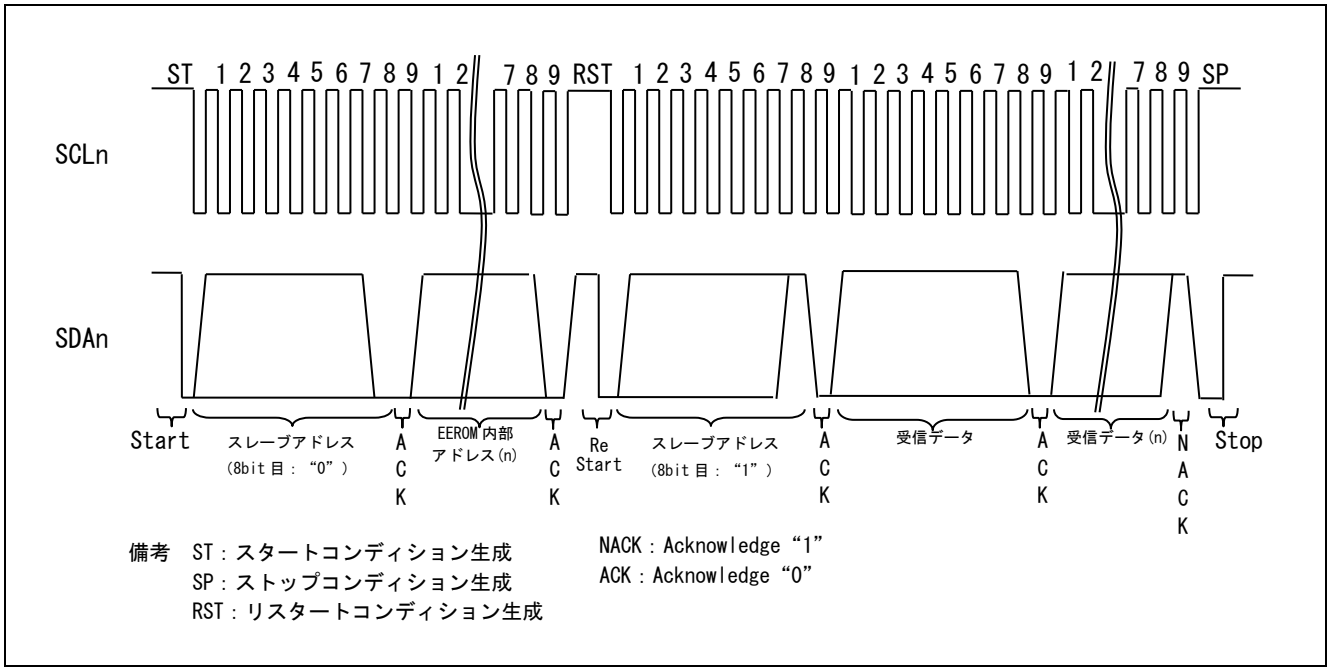


図 5-5 データ読み出し動作

5.3 ソフトウェア動作

本サンプルコードは、OS 制御（※１）を考慮した仕様としています。それぞれの処理方法を以下に示します。

(1) 通常制御（OS 無し）

開始関数をコールすることで通信を開始します。その後、ユーザがコールする EEPROM アドバンス関数により、I²C 通信を進める処理を行います。I²C 通信を進める処理を行うかは、I²C 割り込みの発生有無により、EEPROM アドバンス関数が判断します。EEPROM アドバンス関数コールを割り込み処理ではなく、メイン処理で行うことで I²C チャネルの多重コールにも対応可能な仕様としています。

OS 無し制御では、割り込みが発生するとイベントフラグ（g_iic_Event[]）（※２）が セットされます。そのフラグを確認した EEPROM アドバンス関数が通信を実行します。

通信中の状態は、EEPROM アドバンス関数のリターン値により確認することができます。

(2) 通常制御（OS 有り）

本制御は動作未確認のため、使用される場合、十分に評価して必要に応じて修正してください。

OS 有りの場合、イベントフラグに代わり、OS のシステムコールがイベントになります。

開始関数コール後、EEPROM アドバンス関数をコールすると、イベントが発生するまではシステムコール待ち状態になります。割り込みが発生すると OS のシステムコールが発生し、EEPROM アドバンス関数でタスク（I²C 通信を進める処理）を実行します。

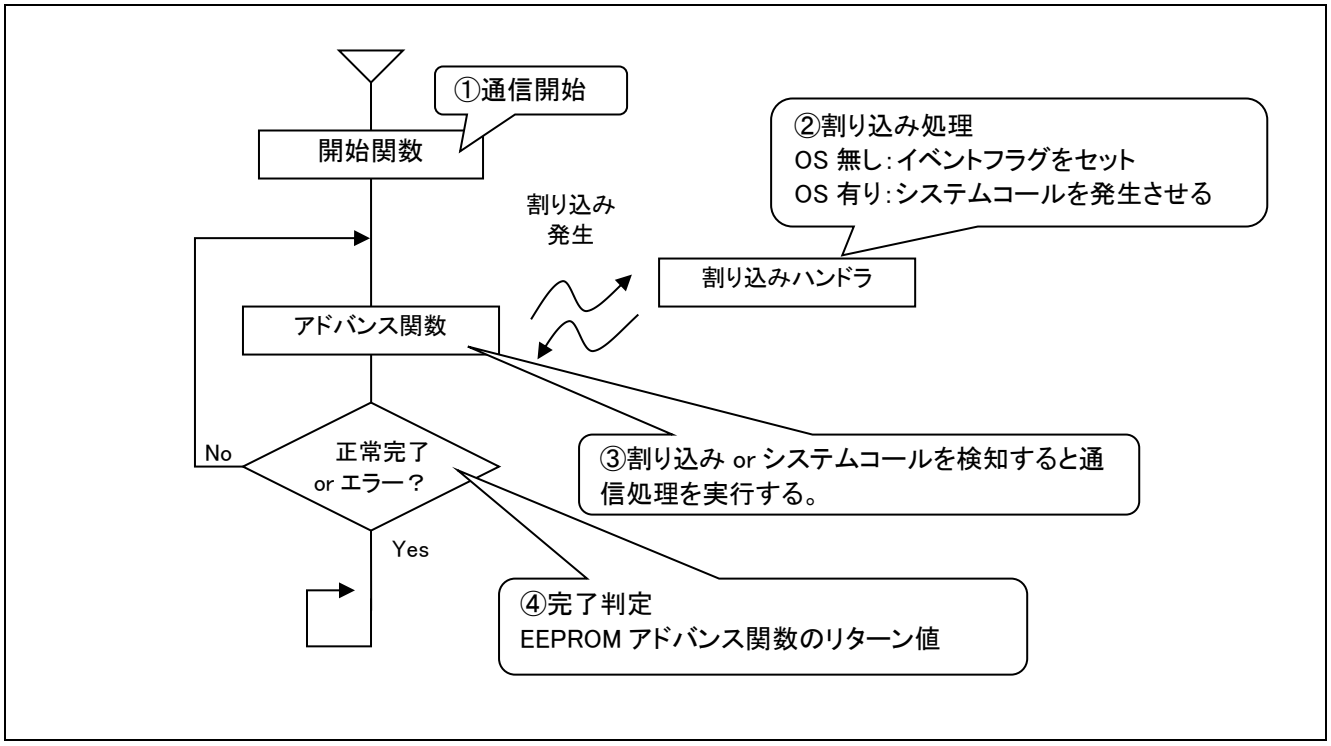


図 5-6 ソフトウェア動作概略（OS 無し/有り制御）

※１：本サンプルコードの OS 制御は、μITRON4.0 を想定しています。
※２：詳しくは、I²C シングルマスタ制御ソフトウェアをご参照ください。

5.4 ソフトウェア動作シーケンス

(1) 通常動作（OS 無し/OS 有り）

以下に、通常動作（OS 無し/OS 有り）時の動作シーケンスを示します。

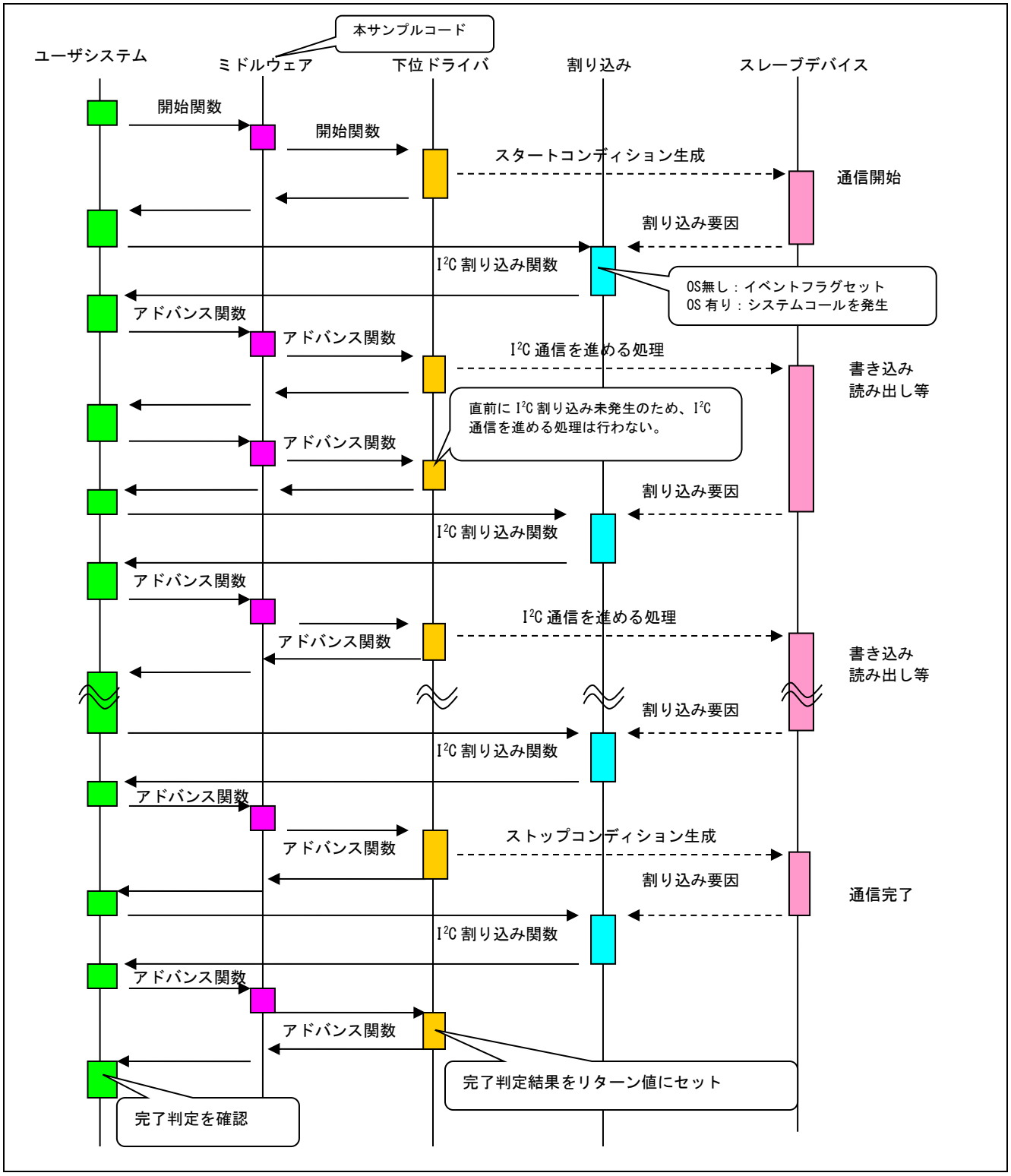


図 5-7 通常動作（OS 有り/OS 無し）シーケンス図

5.5 ブロック書き換えの実現方法

Write 開始関数をコールすると 1 ページ書き込みを行います。ページ境界を跨いだデータ書き換えを行う場合は、EEPROM アドバンス関数で通信を完了させた後、再度 Write 開始関数をコールする必要があります。

本サンプルコードでは、Write 開始関数をコールし、EEPROM アドバンス関数でその通信が完了した時に、ページ境界を跨いだデータ（以降、残データ）があるかチェックを行います。残データがある場合、残データ有りのリターン値が返ります。残データ無しのリターン値が返るまで、Write 開始関数をコールすることで、ブロック書き換えを実現します。

5.6 動作フロー

5.6.1 Write 動作フロー

(1) Acknowledge polling を使用した場合

図 5-8 に Acknowledge polling を使用した場合の Write 動作フローを示します。

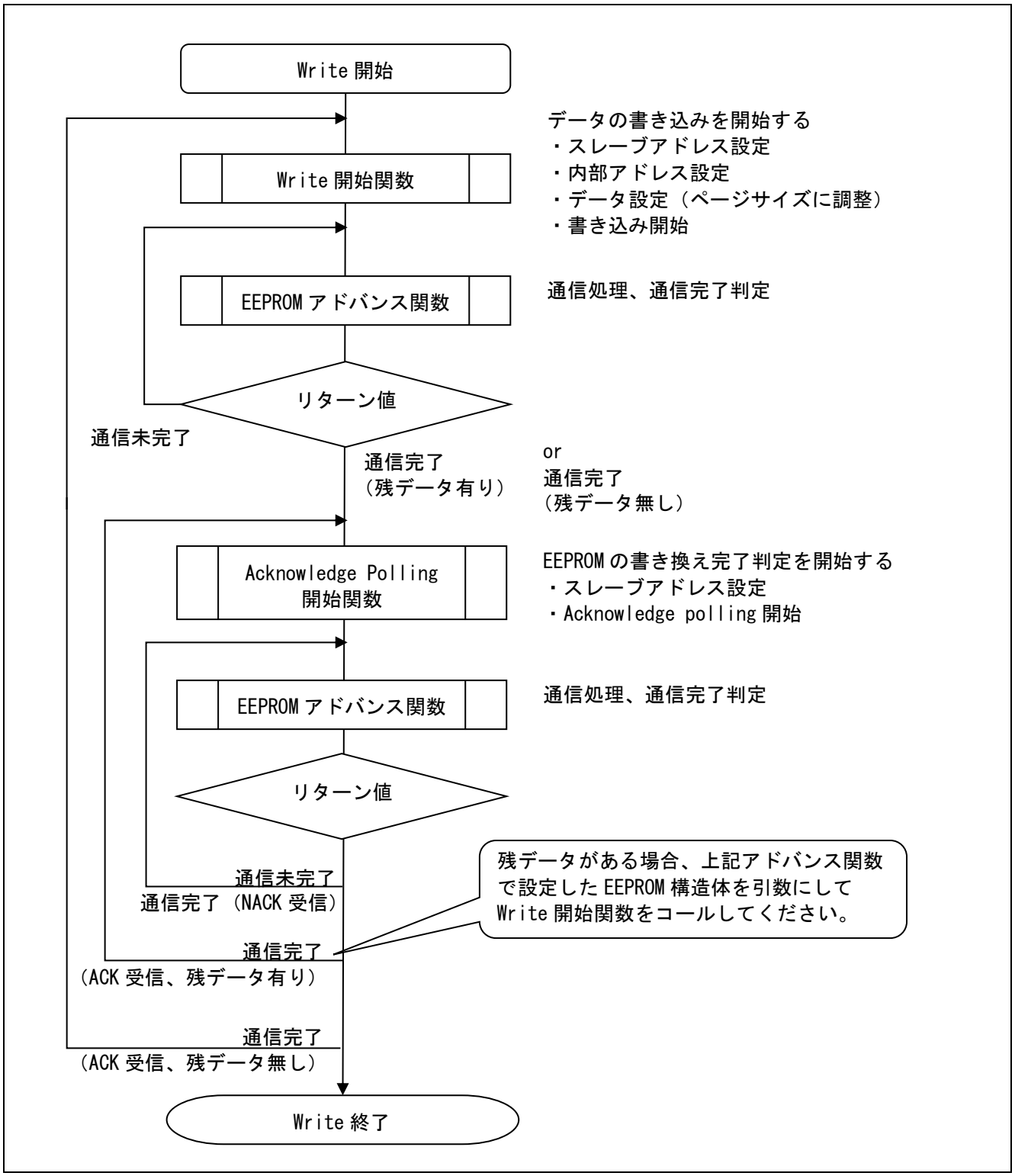


図 5-8 Write 動作フロー（Acknowledge polling 使用）

(2) Acknowledge polling を使用しない場合

図 5-9 に Acknowledge polling を使用しない場合の Write 動作フローを示します。

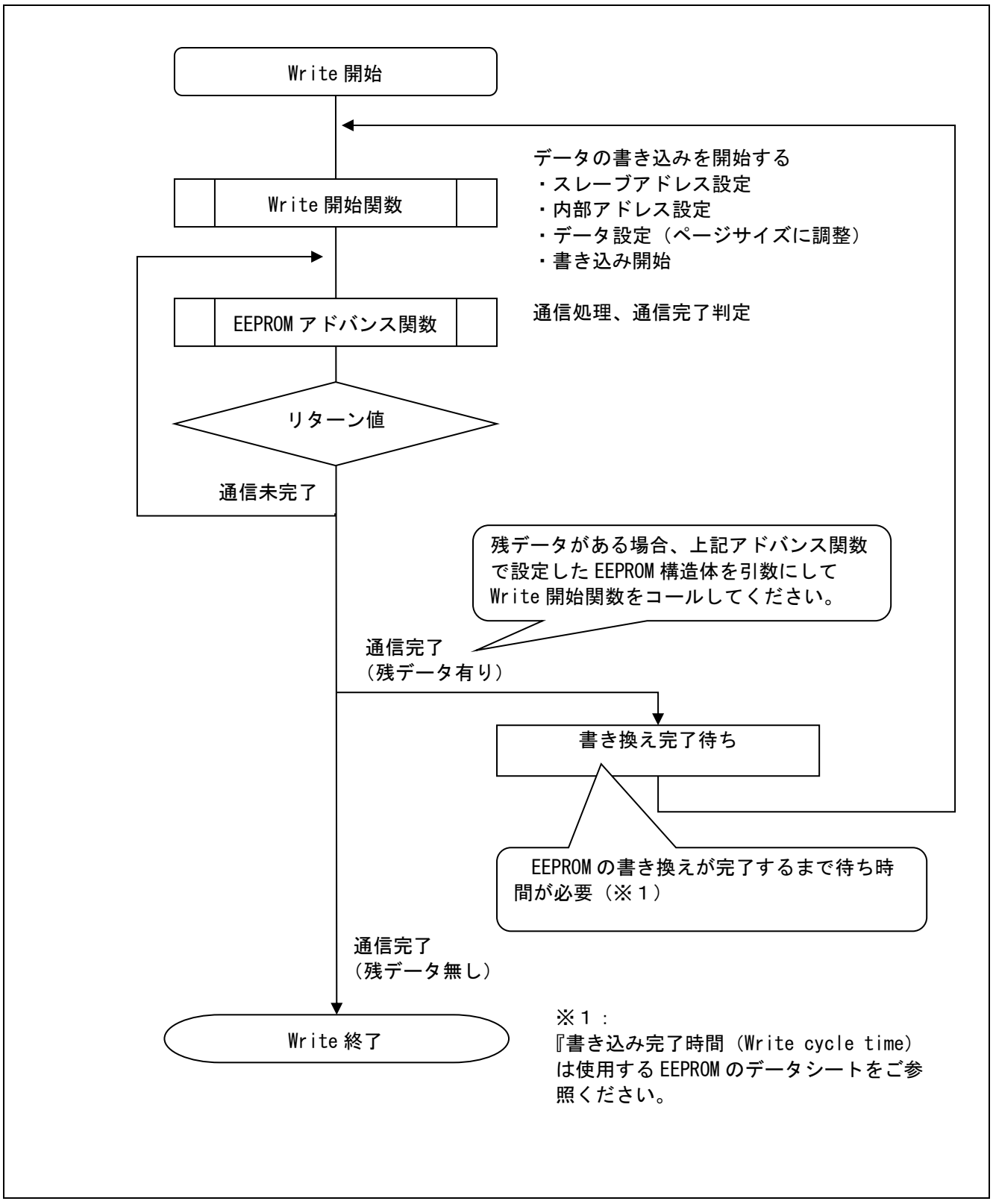


図 5-9 Write 動作フロー（Acknowledge polling 未使用）

5.6.2 Read 動作フロー

図 5-10 に Read 動作フローを示します。

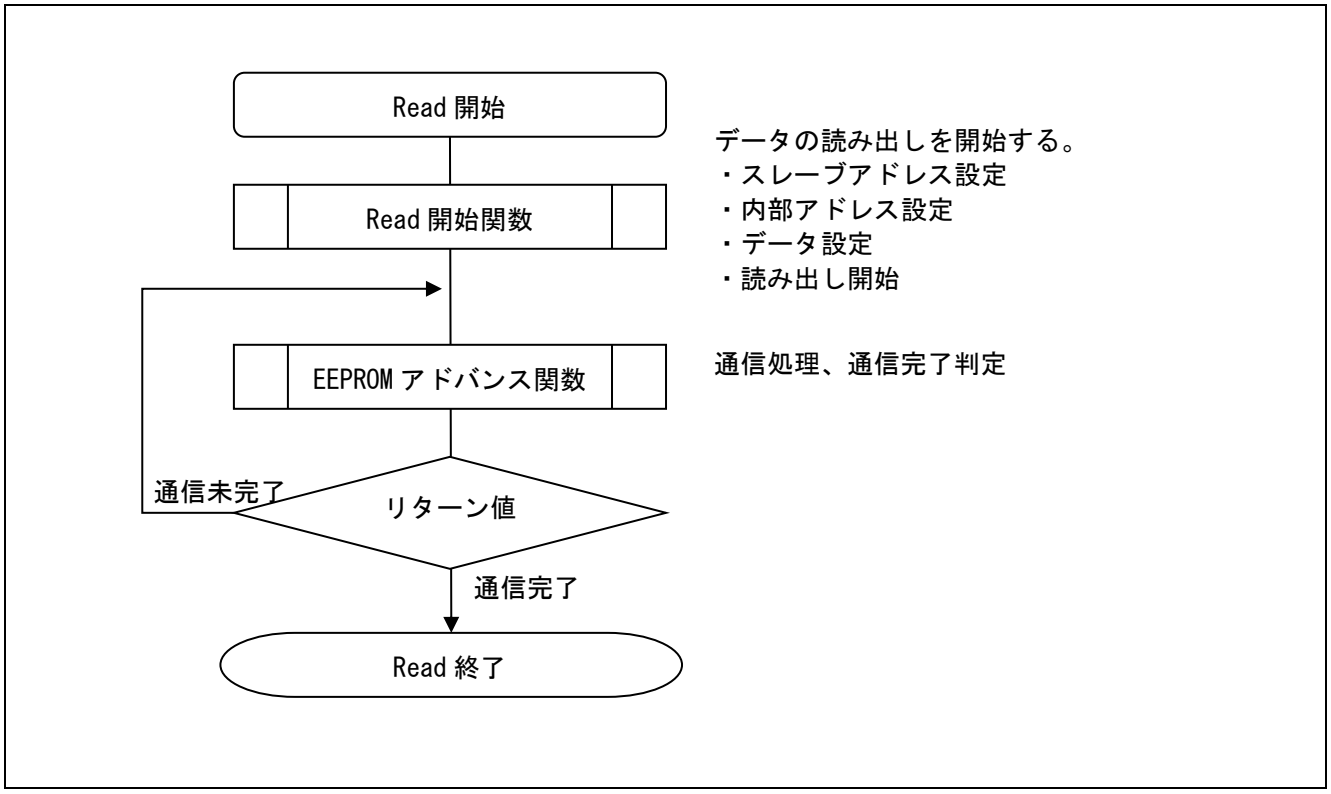


図 5-10 Read 動作フロー

5.7 データバッファと送信／受信データの関係

本サンプルコードは、ブロック型デバイスドライバであり、送信／受信データポインタを引数として設定します。RAM 上のデータバッファのデータ並びと送信／受信順番の関係は、以下のとおりで、エンディアンや使用するシリアル通信機能に関係なく、送信データバッファの並びの順に送信し、また、受信の順に受信データバッファに書き込みます。

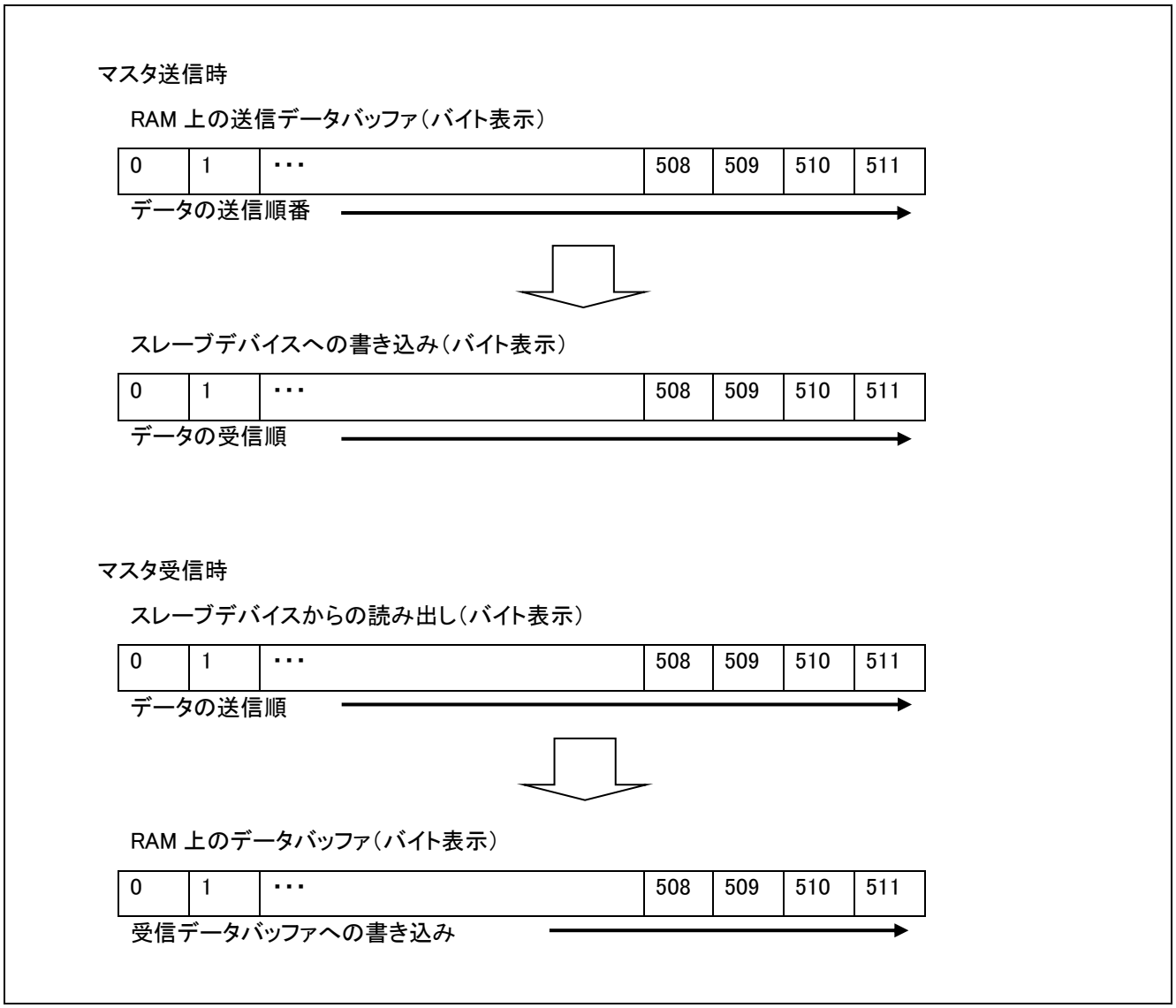


図 5-11 転送データの格納

5.8 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。

5.8.1 RL78 の場合

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

表 5-4 必要メモリサイズ

使用メモリ	サイズ	備考
ROM	2,047 バイト	r_iic_eepmdl_api.c r_iic_eepmdl_sub.c
RAM	2 バイト	r_iic_eepmdl_api.c r_iic_eepmdl_sub.c
最大使用ユーザスタック	90 バイト	
最大使用割り込みスタック	―	

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。
ROM/RAM サイズは下位層の I^2C シングルマスタ制御ソフトウェアに使用されるメモリサイズを含みません。
使用する MCU により、上記メモリサイズは異なります。
最大使用ユーザスタックサイズは、下位層の I^2C シングルマスタ制御ソフトウェアのスタックサイズも含みます。

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

表 5-5 必要メモリサイズ

使用メモリ	サイズ	備考
ROM	1,329 バイト	r_iic_eepmdl_api.c r_iic_eepmdl_sub.c
RAM	2 バイト	r_iic_eepmdl_api.c r_iic_eepmdl_sub.c
最大使用ユーザスタック	70 バイト	
最大使用割り込みスタック	―	

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。
ROM/RAM サイズは下位層の I^2C シングルマスタ制御ソフトウェアに使用されるメモリサイズを含みません。
使用する MCU により、上記メモリサイズは異なります。
最大使用ユーザスタックサイズは、下位層の I^2C シングルマスタ制御ソフトウェアのスタックサイズも含みます。

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

表 5-6 必要メモリサイズ

使用メモリ	サイズ	備考
ROM	3,892 バイト	r_iic_eepmdl_api.c r_iic_eepmdl_sub.c
RAM	4 バイト	r_iic_eepmdl_api.c r_iic_eepmdl_sub.c
最大使用ユーザスタック	272 バイト	
最大使用割り込みスタック	―	

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。
ROM/RAM サイズは下位層の I²C シングルマスタ制御ソフトウェアに使用されるメモリサイズを含みません。
使用する MCU により、上記メモリサイズは異なります。
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 IICA 統合開発環境 CubeSuite+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	1,963 バイト	r_iic_eepmdl_api.c r_iic_eepmdl_sub.c
RAM	4 バイト	r_iic_eepmdl_api.c r_iic_eepmdl_sub.c
最大使用ユーザスタック	78 バイト	
最大使用割り込みスタック	―	

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。
ROM/RAM サイズは下位層の I²C シングルマスタ制御ソフトウェアに使用されるメモリサイズを含みません。
使用する MCU により、上記メモリサイズは異なります。
最大使用ユーザスタックサイズは、下位層の I²C シングルマスタ制御ソフトウェアのスタックサイズも含みます。

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

表 5-8 必要メモリサイズ

使用メモリ	サイズ	備考
ROM	3,482 バイト	r_iic_eepmdl_api.c r_iic_eepmdl_sub.c
RAM	4 バイト	r_iic_eepmdl_api.c r_iic_eepmdl_sub.c
最大使用ユーザスタック	146 バイト	
最大使用割り込みスタック	―	

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。
ROM/RAM サイズは下位層の I²C シングルマスタ制御ソフトウェアに使用されるメモリサイズを含みません。
使用する MCU により、上記メモリサイズは異なります。
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。

5.8.2 RX の場合

(1) RX63N RIIC の場合

表 5-9 必要メモリサイズ

使用メモリ	サイズ	備考
ROM	1,155 バイト（リトルエンディアン）	r_iic_eepmdl_api.c r_iic_eepmdl_sub.c
RAM	1 バイト（リトルエンディアン）	r_iic_eepmdl_api.c r_iic_eepmdl_sub.c
最大使用ユーザスタック	136 バイト	
最大使用割り込みスタック	―	

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。
ROM/RAM サイズは下位層の I^2C シングルマスタ制御ソフトウェアに使用されるメモリサイズを含みません。
使用する MCU により、上記メモリサイズは異なります。
最大使用ユーザスタックサイズは、下位層の I^2C シングルマスタ制御ソフトウェアのスタックサイズも含みます。

(2) RX210 RIIC の場合

表 5-10 必要メモリサイズ

使用メモリ	サイズ	備考
ROM	1,126 バイト（リトルエンディアン）	r_iic_eepmdl_api.c r_iic_eepmdl_sub.c
RAM	1 バイト（リトルエンディアン）	r_iic_eepmdl_api.c r_iic_eepmdl_sub.c
最大使用ユーザスタック	148 バイト	
最大使用割り込みスタック	―	

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。
ROM/RAM サイズは下位層の I^2C シングルマスタ制御ソフトウェアに使用されるメモリサイズを含みません。
使用する MCU により、上記メモリサイズは異なります。
最大使用ユーザスタックサイズは、下位層の I^2C シングルマスタ制御ソフトウェアのスタックサイズも含みます。

5.9 ファイル構成

表 5-11 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 5-11 ファイル構成

¥an_r01an1075jj0103_mcu_seep <DIR>			サンプルコードのフォルダ
r01an1075jj0103_mcu.pdf			アプリケーションノート
¥ source <DIR>			プログラム格納用フォルダ
¥ r_iic_eepmdl <DIR>			Serial EEPROM 制御ソフトウェア用フォルダ
	r_iic_eepmdl_api.c		API ソースファイル
	r_iic_eepmdl_api.h		API ヘッダファイル
	r_iic_eepmdl_sub.c		内部関数ソースファイル
	r_iic_eepmdl_sub.h		内部関数ヘッダファイル
¥ sample <DIR>			動作確認プログラム格納用フォルダ
	testmain.c		デバイス 1 個接続動作検証用のサンプルソースファイル
	testmain.h		デバイス 1 個接続動作検証用のサンプルヘッダファイル
	testmain2.c		デバイス 2 個接続動作検証用のサンプルソースファイル
	testmain2.h		デバイス 2 個接続動作検証用のサンプルヘッダファイル

5.10 定数一覧

5.10.1 各種定義

以下に、サンプルコードで使用する各種定義を示します。

表 5-12 マクロ定義（リターン値）

定数名	設定値	内容
R_IIC_EEP_NO_INIT	(error_t)(0)	未初期化状態
R_IIC_EEP_IDLE	(error_t)(1)	アイドル状態
R_IIC_EEP_COMMUNICATION	(error_t)(4)	通信中： Write 動作/Acknowledge polling/Read 動作のいずれかの通信を実行中
R_IIC_EEP_LOCK_FUNC	(error_t)(5)	API 処理中 以下の場合に発生します。 ・API 処理中に他の API をコールした場合
R_IIC_EEP_BUS_BUSY	(error_t)(6)	バスビジー 以下の場合に発生します。 ・通信中に EEPROM 初期化関数または各開始関数をコールした場合 ・同一チャネル上の他のデバイスが通信中に、各開始関数または EEPROM アドバンス関数をコールした場合
R_IIC_EEP_FINISH_WRITE	(error_t)(21)	アイドル状態：Write 完了 全データの書き込み完了
R_IIC_EEP_FINISH_WRITE_AGN	(error_t)(22)	アイドル状態：Write 完了 ページ書き込み完了、残データ有り
R_IIC_EEP_FINISH_ACKPOL	(error_t)(23)	アイドル状態：Acknowledge polling 完了 全データの書き換え完了
R_IIC_EEP_FINISH_ACKPOL_AGN	(error_t)(24)	アイドル状態：Acknowledge polling 完了 ページ書き換え完了、残データ有り
R_IIC_EEP_FINISH_ACKPOL_NACK	(error_t)(25)	アイドル状態：Acknowledge polling 完了 ページ書き換え未完了（NACK 受信）
R_IIC_EEP_FINISH_READ	(error_t)(26)	アイドル状態：Read 完了 全データの読み出し完了
R_IIC_EPP_ERR_PARAM	(error_t)(-1)	パラメータエラー
R_IIC_EPP_ERR_AL	(error_t)(-2)	アービトレーションロストエラー
R_IIC_EPP_ERR_NON_REPLY	(error_t)(-3)	未応答エラー
R_IIC_EPP_ERR_SDA_LOW_HOLD	(error_t)(-4)	EEPROM 復帰処理関数コール時、SDA Low ホールドエラー
R_IIC_EPP_ERR_OTHER	(error_t)(-5)	その他エラー
R_IIC_EEP_ERR_NACK	(error_t)(-21)	Write 通信中に NACK を受信

表 5-13　マクロ定義（変更禁止）

定数名	設定値	内容
R_IIC_EEP_DEVCODE	(uint8_t)(0xa0)	EEPROM 固定のデバイスコード
R_IIC_EEP_FALSE	(uint8_t)(0x00)	Flag "ON"
R_IIC_EEP_TRUE	(uint8_t)(0x01)	Flag "OFF"

表 5-14　マクロ定義（ユーザ変更可能）

定数名	設定値	内容
SCL_CLK_CNT	(uint8_t)(0x09)	SCL 疑似クロック生成カウンタ EEPROM 復帰関数コール時、SCL に疑似クロックを生成する回数を定義します。 通信単位が 9 クロックの場合が多いため、カウンタ値は 9 に設定しています。

5.11　構造体／共用体一覧

5.11.1　EEPROM 通信情報構造体

サンプルコードで使用する EEPROM 通信情報構造体を以下に示します。本構造体には、EEPROM と通信するために必要な情報が格納されます。使用するスレーブデバイス毎に設定する必要があります。

```
typedef struct
{
  r_iic_eepmdl_mode_t          EepMode;     /* Mode of EEPROM                    */
  r_iic_eepmdl_eepsize_t       EepSize;     /* Size of EEPROM                    */
  r_iic_eepmdl_pagesize_t      PageSize;    /* Size of write page                */
  uint32_t                     EepIntAdr;   /* Internal address of EEPROM        */
  uint32_t                     EepRWCnt;    /* R/W data counter                  */
  r_iic_drv_info_t             RIic_Info;   /* IIC information                   */
  uint8_t                      DevAdr;      /* Address to appoint a slave device */
  uint8_t                      rsv1;
  uint8_t                      rsv2;
  uint8_t                      rsv3;
} r_iic_eepmdl_info_t;
```

図 5-12　EEPROM 通信情報構造体

(1) メンバ説明

表 5-15 に構造体“r_iic_eepmdl_info_t”の説明を以下に示します。

表 5-15 構造体“r_iic_eepmdl_info_t”のメンバー覧

構造体メンバ	設定可能範囲	説明
EepMode	（設定禁止）	動作モード 「未通信状態」「Write 中」「Acknowledge polling 中」「Read 中」の 4 つのモードがあります。本ソフトウェアで管理するため、設定は禁止です。
EepSize	—	EEPROM 容量 2K ビットから 1M ビットまでサポートします。列挙型“r_iic_eepmdl_romsize_t”で定義する値（表 5-17）を参照し、定義から 1 つ設定してください。
PageSize	—	EEPROM ページサイズ（※１、※２） 8 バイトから 256 バイトまでサポートします。列挙型“r_iic_eepmdl_pagesize_t”で定義する値（表 5-18）を参照し、定義から 1 つ設定してください。
EepIntAdr	0000 0000h～FFFF FFFFh	EEPROM 内部アドレス（an）（※１） EEPROM へ書き込みまたは読み出しを行う際の先頭アドレスを指定してください。使用する EEPROM の最大サイズに収まる値に設定してください。
EepRWCnt	0000 0000h～FFFF FFFFh	データカウンタ（バイト数）（※１） 書き込み時：送信データカウンタ（送信したいデータの総数を設定） 読み出し時：受信データカウンタ（受信したいデータの総数を設定）
RIic_Info	—	I^2C シングルマスタ制御ソフトウェアの I^2C 通信情報構造体 詳細は表 5-16 を参照ください。
DevAdr	00h～07h	EEPROM デバイスアドレスコード（An）（※３） EEPROM のデバイスアドレスコードとして、EEPROM の物理ピンで示される A0-A2 で示されるデバイスアドレスコードを指定してください。 表 物理ピン接続と DevAdr 設定値の関係 A2 / A1 / A0 / DevAdr L / L / L / 0 L / L / H / 1 L / H / L / 2 L / H / H / 3 H / L / L / 4 H / L / H / 5 H / H / L / 6 H / H / H / 7
rsv1 rsv2 rsv3	（設定無効）	アライメント用。

※１：EEPROM 容量により制限値が異なります。詳しくはブロック書き換え 5.2.2(4)を参照ください。
※２：使用する EEPROM で規定されているページサイズを設定してください。使用する EEPROM のページサイズ以上の値を設定した場合でも、本サンプルコードは通信を行います。この場合、書き込みで Roll Over が発生しますので、ご注意ください。
※３：製品型名により最大接続スレーブ数が異なります。詳しくは 5.2.1(2)を参照ください。

表 5-16　I²C シングルマスタ制御ソフトウェアの I²C 通信情報構造体メンバー一覧

構造体メンバ	設定可能範囲	内容
*pSlvAdr	—	スレーブアドレス格納バッファポインタ スレーブアドレスを指定するデータの格納元です。データの格納元のアドレスを指定してください。 1 バイト確保してください。
*pData1st	—	EEPROM 内部アドレス格納バッファポインタ EEPROM 内部アドレスを指定するデータの格納元です。データの格納元のアドレスを指定してください。 書き込み時または読み出し時： EEPROM 容量が 16K ビット以下の場合、1 バイト確保してください。 EEPROM 容量が 32K ビット以上の場合、2 バイト確保してください。 Acknowledge polling 時： 設定不要。設定値は無視されます。
*pData2nd	—	データ格納バッファポインタ 書き込み時：EEPROM へ書き込むデータの格納元 読み出し時：EEPROM から読み出すデータの格納先 Acknowledge polling 時：設定不要。設定値は無視されます。
*pDevStatus	—	デバイス状態フラグポインタ 使用方法は I²C シングルマスタ制御ソフトウェアをご参照ください。
Cnt1st	（設定禁止）	EEPROM 内部アドレスカウンタ EEPROM の内部アドレスを指定するデータのカウンタです。本サンプルコードが設定するため、設定禁止です。
Cnt2nd	（設定禁止）	データカウンタ 書き込み時：1 ページに書き込むデータ数 読み出し時：読み出したいデータ総数 本サンプルコードが設定するため、設定禁止です。
CallBackFunc	—	コールバック関数 使用方法は I²C シングルマスタ制御ソフトウェアをご参照ください。
ChNo	00h～FFh	I²C バス制御機能のチャネル番号 使用するバスのチャネル番号を設定してください。
rsv1 rsv2 rsv3	（設定無効）	アライメント用。

5.12 列挙型

本サンプルコードで使用する列挙型の定義を以下に示します。

表 5-17 使用する EEPROM の容量一覧（enum r_iic_eepmdl_eepsize_t）

定義名	内容
R_IIC_EEP_EEPSIZE_002K	EEPROM 容量 2K ビット
R_IIC_EEP_EEPSIZE_004K	EEPROM 容量 4K ビット
R_IIC_EEP_EEPSIZE_008K	EEPROM 容量 8K ビット
R_IIC_EEP_EEPSIZE_016K	EEPROM 容量 16K ビット
R_IIC_EEP_EEPSIZE_032K	EEPROM 容量 32K ビット
R_IIC_EEP_EEPSIZE_064K	EEPROM 容量 64K ビット
R_IIC_EEP_EEPSIZE_128K	EEPROM 容量 128K ビット
R_IIC_EEP_EEPSIZE_256K	EEPROM 容量 256K ビット
R_IIC_EEP_EEPSIZE_512K	EEPROM 容量 512K ビット
R_IIC_EEP_EEPSIZE_001M	EEPROM 容量 1M ビット

表 5-18 使用する EEPROM のページサイズ一覧（enum r_iic_eepmdl_pagesize_t）

定義名	内容
R_IIC_EEP_PAGESIZE_8B	EEPROM ページサイズ 8 バイト
R_IIC_EEP_PAGESIZE_16B	EEPROM ページサイズ 16 バイト
R_IIC_EEP_PAGESIZE_32B	EEPROM ページサイズ 32 バイト
R_IIC_EEP_PAGESIZE_64B	EEPROM ページサイズ 64 バイト
R_IIC_EEP_PAGESIZE_128B	EEPROM ページサイズ 128 バイト
R_IIC_EEP_PAGESIZE_256B	EEPROM ページサイズ 256 バイト

表 5-19 EEPROM 動作モード（enum r_iic_eepmdl_mode_t）

定義名	内容
R_IIC_EEP_MODE_NONE	未通信状態
R_IIC_EEP_MODE_WRITE	Write 中
R_IIC_EEP_MODE_ACKPOL	Acknowledge polling 中
R_IIC_EEP_MODE_READ	Read 中

5.13 変数一覧

表 5-20 にグローバル変数を示します。

表 5-20 グローバル変数

型	変数名	内容	使用関数
bool	g_iic_EepMdl_Api[MAX_IIC_CH_NUM]（※）	EEPROM API フラグ 本サンプルコードの API の多重コールを防止するために使用します。 API 処理開始時にセットされ、終了後にクリアされます。	R_IIC_EepMdl_Init() R_IIC_EepMdl_Write() R_IIC_EepMdl_AckPolling() R_IIC_EepMdl_Read() R_IIC_EepMdl_Advance() R_IIC_EepMdl_Recovery()

※：「MAX_IIC_CH_NUM」は I^2C シングルマスタ制御ソフトウェアの定義値です。“同時に使用する最大チャネル番号+1”の値が格納されます。

5.14 関数一覧

表 5-21 に関数一覧を示します。

表 5-21 関数一覧

関数名	説明
R_IIC_EepMdl_Init()	EEPROM 初期化関数
R_IIC_EepMdl_Write()	Write 開始関数
R_IIC_EepMdl_AckPolling()	Acknowledge polling 開始関数
R_IIC_EepMdl_Read()	Read 開始関数
R_IIC_EepMdl_Advance()	EEPROM アドバンス関数
R_IIC_EepMdl_Recovery()	EEPROM 復帰関数

5.15 状態遷移図

図 5-13 に、各チャネルの状態遷移図を示します。

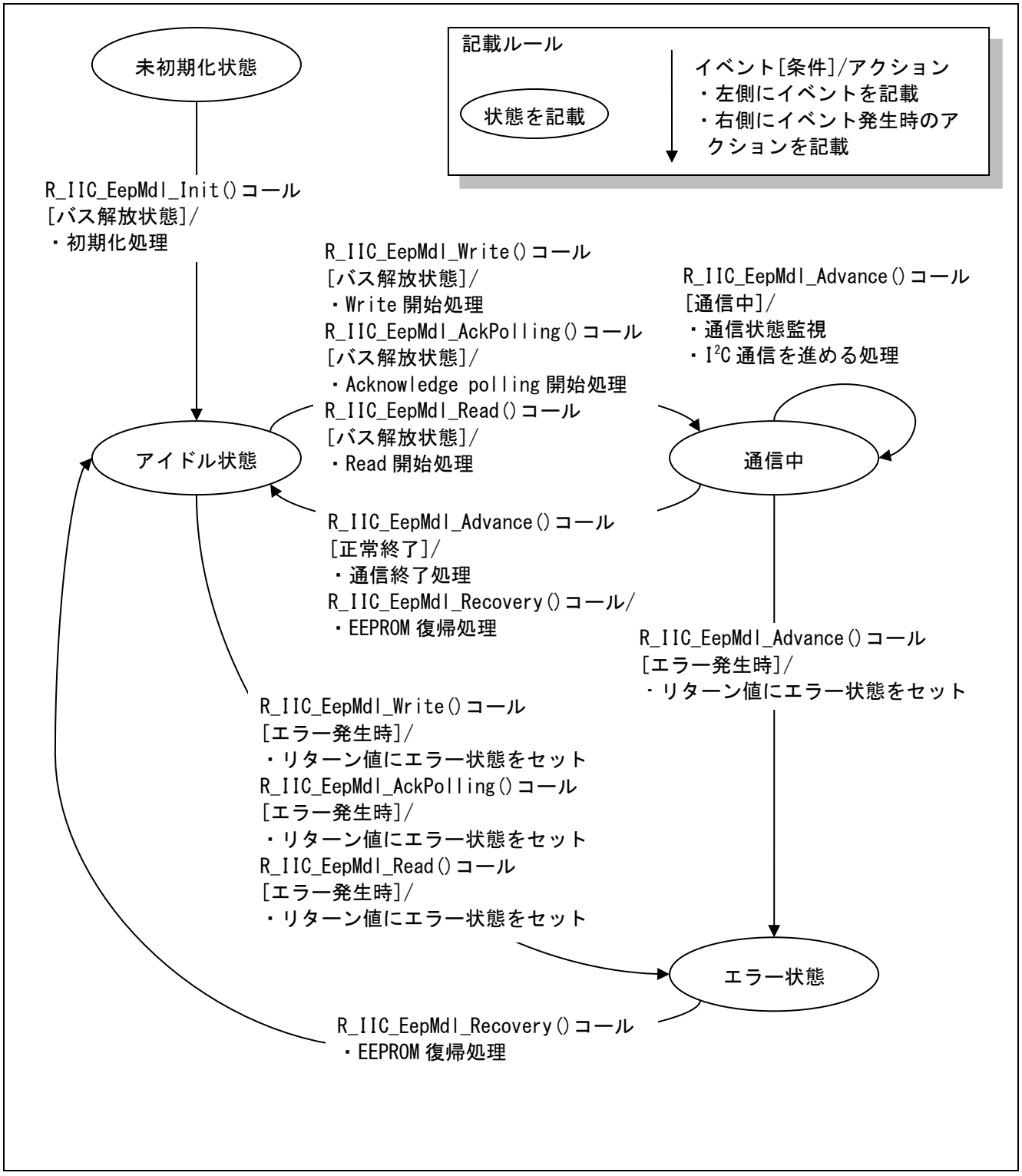


図 5-13 状態遷移図

5.16 関数仕様

ユーザアプリケーションにて、デバイスの通信情報領格納域を確保し、制御時に、パラメータを設定して、コールしてください。

5.16.1 関数共通処理

本サンプルコードでは、1 度にコールできる関数は 1 つであり、関数処理実行中に本サンプルコードの他の関数がコールされた場合、処理を行わずに終了します。この際、“R_IIC_EEP_LOCK_FUNC”がリターン値として返ります。

関数の同時コールを防止するために EEPROM API フラグを用意しています。このフラグは、処理が行われている間セットされます。各関数の最初に必ずチェックされ、フラグがセットされていなければ、その処理を実行する仕組みです。図 5-14 に概略フローを示します。

この処理は、5.14 項で定義している関数に対して実施します。以降、5.16.2 項～では図 5-14 の「API 関数処理」の処理内容について記載します。

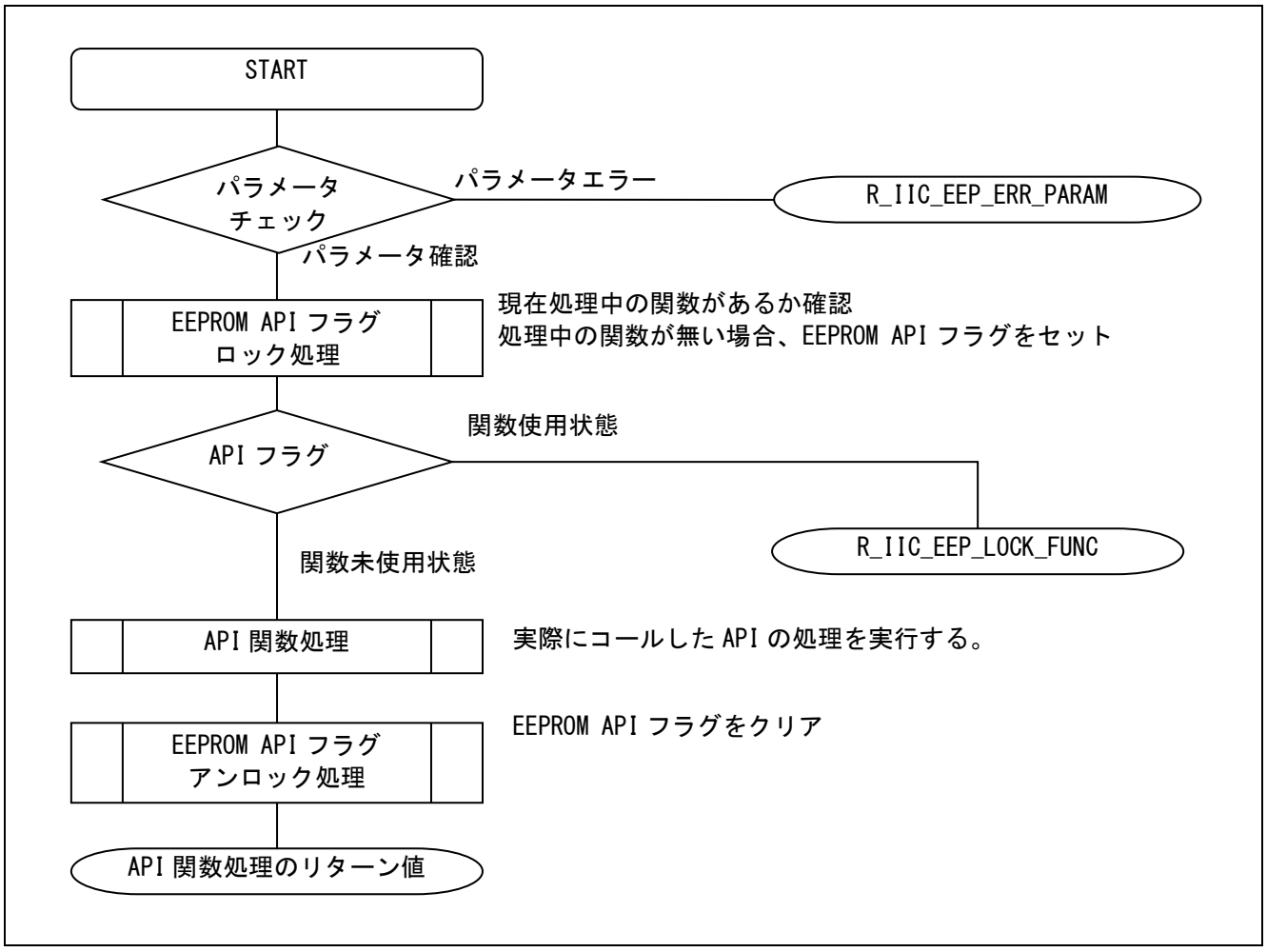


図 5-14 多重コール防止処理概略フロー

5.16.2 EEPROM 初期化関数

R_IIC_EepMdl_Init	
概要	EEPROM 初期化関数
ヘッダ	r_iic_eepmdl_api.h, r_iic_eepmdl_sub.h
宣言	error_t R_IIC_EepMdl_Init(r_iic_eepmdl_info_t FAR *pEep_Info)
説明	・対象チャネルの初期設定を行います。実施後、アイドル状態に遷移し、通信可能状態になります。 ・本処理を行うためには、以下の設定が必要です。 構造体“r_iic_eepmdl_info_t”メンバの RIic_Info.ChNo;使用チャネル番号 チャネル状態フラグ（g_iic_ChStatus[]）;“R_IIC_NO_INIT”をセット（※１） デバイス状態フラグ（*(pEep.Info.RIic_Info.pDevStatus)） ;“R_IIC_NO_INIT”をセット（※１）
引数	r_iic_eepmdl_info_t FAR *pEep_Info ;EEPROM 通信情報構造体ポインタ
リターン値	R_IIC_EEP_IDLE 初期化を行い、アイドル状態に遷移しました。すでに初期化済みの場合、初期化は行いません。 →開始関数をコールすることで通信が可能です。 R_IIC_EEP_LOCK_FUNC 他の API 処理が行われているため、処理は行われませんでした。 →他の API 処理終了後、本関数をコールしてください。 R_IIC_EEP_BUS_BUSY 通信中です。初期化できませんでした。 →EEPROM アドバンス関数をコールして通信を終了させてください。 R_IIC_EEP_ERR_PARAM パラメータエラーです。 →設定値を確認してください。 R_IIC_EEP_ERR_AL アービトレーションロストが発生しています。 →EEPROM 復帰関数をコールすることで復帰処理を行うことができます。 R_IIC_EEP_ERR_NON_REPLY 未応答エラーが発生しています。（※２） →EEPROM 復帰関数をコールすることで復帰処理を行うことができます。 R_IIC_EEP_ERR_SDA_LOW_HOLD SDA は Low ホールドから復帰できていない状態です。 →スレーブデバイスが Low ホールドしていないか、マスタデバイスから Low 信号が出力されていないか等、システムの状態を確認してください。 R_IIC_EEP_ERR_OTHER その他エラーが発生しています。 →以下を確認してください。 ・EEPROM 通信情報構造体の設定が正しく行われているか確認してください。 ・OS 制御でエラーが発生していないか確認してください。
備考	・制御対象のデバイスに対して一度呼び出してください。 ・初期化済みの場合、I^2C ドライバ初期化処理は行いません。 ・エラー等が発生して再初期化を行いたい場合には、EEPROM 復帰関数をコールしてください。 ※１：I^2C シングルマスタ制御ソフトウェアで定義するグローバル変数です。バスの状態（未初期化/アイドル/通信中/エラー）を管理します。EEPROM 初期化関数をコールする場合、その前に“R_IIC_NO_INIT”をセットしてくださ

い。設定しないまま EEPROM 初期化関数をコールしても、初期化処理を行わない場合があります。

※２：未応答エラーは、制御する MCU によって定義が異なります。詳しくは I²C シングルマスタ制御ソフトウェアの未応答エラーの説明をご参照ください。

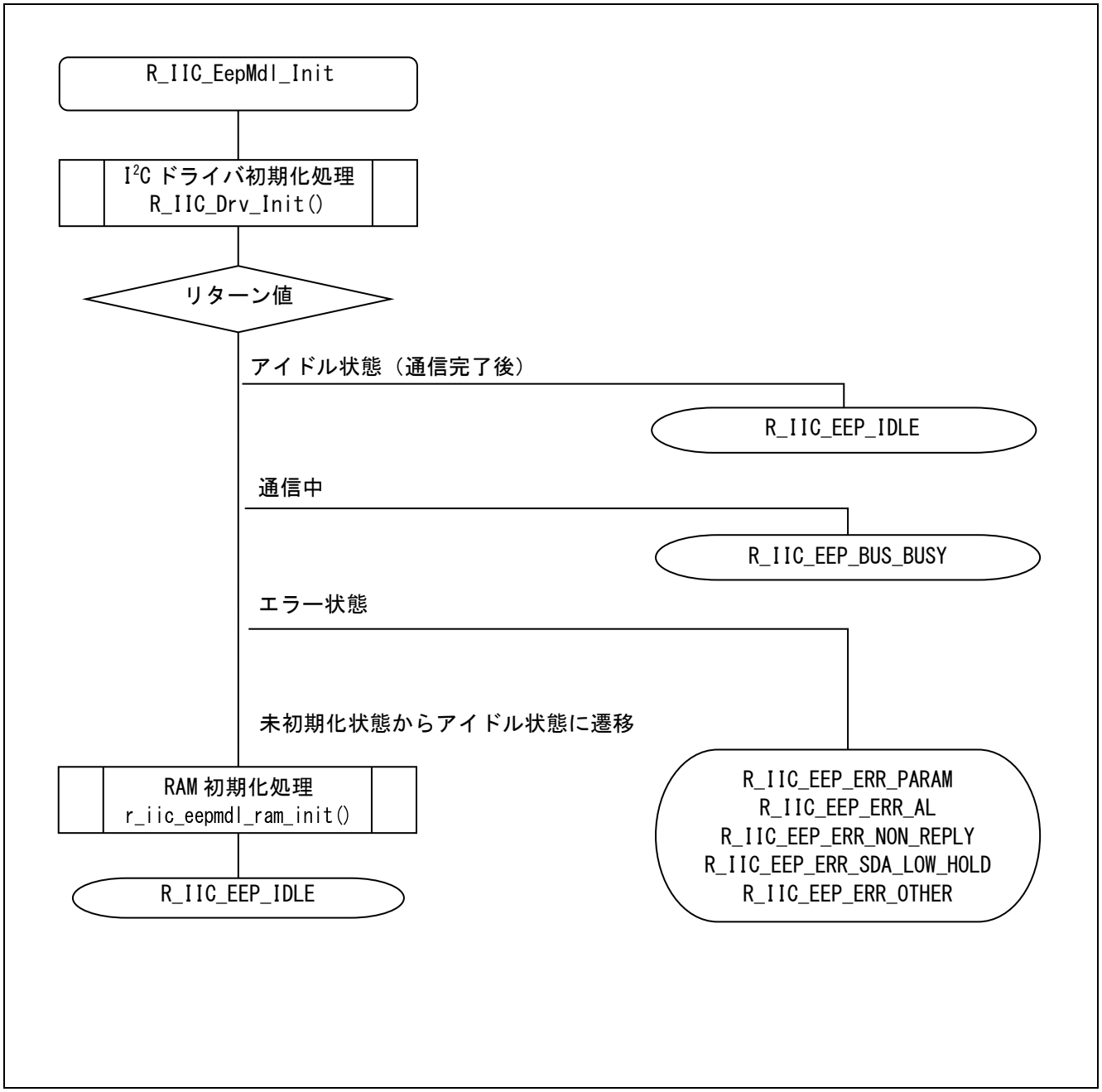


図 5-15　EEPROM ドライバ初期化関数概要

5.16.3 Write 開始関数

R_IIC_EepMdl_Write	
概要	Write 開始関数
ヘッダ	r_iic_eepmdl_api.h, r_iic_eepmdl_sub.h
宣言	error_t R_IIC_EepMdl_Write(r_iic_eepmdl_info_t FAR *pEep_Info)
説明	・EEPROM へのデータ書き込みを開始します。 ・マスタ（MCU）からスレーブ（EEPROM）へデータを書き込みたい場合に使用してください。
引数	r_iic_eepmdl_info_t FAR *pEep_Info ;EEPROM 通信情報格納ポインタ
リターン値	■リターン値（R_IIC_EepMdl_Write()コール時） R_IIC_EEP_COMMUNICATION EEPROM への書き込みを開始しました。 →EEPROM アドバンス関数をコールして通信を終了させてください。 R_IIC_EEP_NO_INIT 未初期化状態です。 →EEPROM 初期化関数をコールして初期化してください。 R_IIC_EEP_LOCK_FUNC 他の API 処理が行われているため、処理は行われませんでした。 →他の API 処理終了後、本関数をコールしてください。 R_IIC_EEP_BUS_BUSY 通信中です。EEPROM への書き込み処理を開始できませんでした。 →EEPROM アドバンス関数をコールして通信を終了させてください。 R_IIC_EEP_ERR_PARAM パラメータエラーです。 →設定値を確認してください。 R_IIC_EEP_ERR_AL アービトレーションロストが発生しています。 →EEPROM 復帰関数をコールすることで復帰処理を行うことができます。 R_IIC_EEP_ERR_NON_REPLY 未応答エラーが発生しています。（※１） →EEPROM 復帰関数をコールすることで復帰処理を行うことができます。 R_IIC_EEP_ERR_SDA_LOW_HOLD SDA は Low ホールドから復帰できていない状態です。 →スレーブデバイスが Low ホールドしていないか、マスタデバイスから Low 信号が出力されていないか等、システムの状態を確認してください。 R_IIC_EEP_ERR_OTHER その他エラーが発生しています。 →以下を確認してください。 ・EEPROM 通信情報構造体の設定が正しく行われているか確認してください。 ・OS 制御でエラーが発生していないか確認してください。 ■リターン値（本関数コール後、R_IIC_EepMdl_Advance()コール時） R_IIC_EEP_COMMUNICATION 通信中です。 →EEPROM アドバンス関数をコールして通信を終了させてください。 R_IIC_EEP_FINISH_WRITE 全データの書き込みが完了しました。 →Acknowledge polling 開始関数をコールすることで、EEPROM への書き換え

が完了したか判定することができます。（※２）

R_IIC_EEP_FINISH_WRITE_AGN

ページ書き込みが完了しました。残データがあります。

→書き換え完了後、Write 開始関数をコールしてください。この際、EEPROM 通信情報構造体メンバの情報は変更しないでください。（※３）

エラー時のリターン値は、EEPROM アドバンス関数を参照ください。

備考

・I^2C シングルマスタ制御ソフトウェアのマスタ送信モード（パターン 1）を使用します。

・本処理を行うためには、構造体“r_iic_eepmdl_info_t”の設定が必要です。設定方法は、5.11.1 を参照ください。

・本関数から戻った時点では、I^2C 通信は完了していません。I^2C 通信を終了させるためには、EEPROM アドバンス関数をコールする必要があります。

・本関数をコールすると 1 ページ書き込みを行います。書き込みデータの最終アドレスがページ境界アドレスより大きい場合には､書き込みデータをページ境界までに変更します。

・ページ境界を跨いだデータ書き換えを行う場合は、EEPROM アドバンス関数で通信を完了させた後、再度 Write 開始関数をコールしてください。

※１：未応答エラーは、制御する MCU によって定義が異なります。詳しくは I^2C シングルマスタ制御ソフトウェアの未応答エラーの説明をご参照ください。

※２：書き換え判定を行わず Write/Read 開始関数をコールすることも可能です。この場合、EEPROM 書き換え完了待ちを行った後、開始関数をコールしてください。尚、書き込み完了時間（Write cycle time）は使用する EEPROM のデータシートをご参照ください。

※３：EEPROM 通信情報構造体メンバには、残データの書き込み情報が格納されているため、メンバの値を変更した場合、正常に通信を行うことができなくなります。

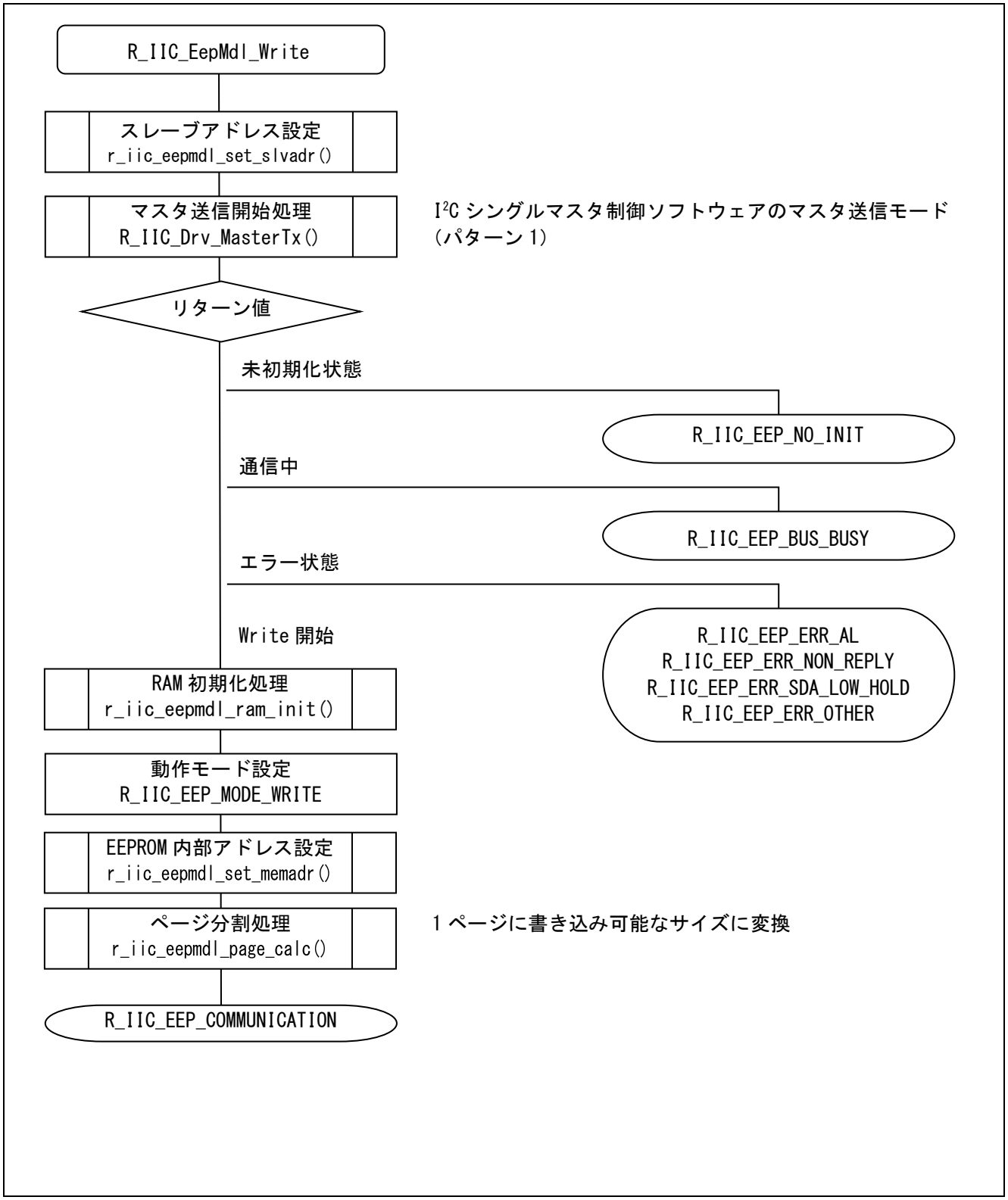


図 5-16　Write 開始関数概要

5.16.4 Acknowledge polling 開始関数

R_IIC_EepMdl_AckPolling	
概要	Acknowledge polling 開始関数
ヘッダ	r_iic_eepmdl_api.h, r_iic_eepmdl_sub.h
宣言	error_t R_IIC_EepMdl_AckPolling(r_iic_eepmdl_info_t FAR *pEep_Info)
説明	・Acknowledge Polling（データ書き換え完了判定）を開始します。 ・マスタ（MCU）からスレーブ（EEPROM）へのデータ書き込みが完了した後、EEPROM のデータ書き換えが完了したかどうか判定したい場合に使用してください。
引数	r_iic_eepmdl_info_t FAR *pEep_Info ;EEPROM 通信情報格納ポインタ
リターン値	■リターン値（R_IIC_EepMdl_AckPolling()コール時） R_IIC_EEP_COMMUNICATION Acknowledge polling を開始しました。 →EEPROM アドバンス関数をコールして通信を終了させてください。 R_IIC_EEP_NO_INIT 未初期化状態です。 →EEPROM 初期化関数をコールして初期化してください。 R_IIC_EEP_LOCK_FUNC 他の API 処理が行われているため、処理は行われませんでした。 →他の API 処理終了後、本関数をコールしてください。 R_IIC_EEP_BUS_BUSY 通信中です。Acknowledge polling 処理を開始できませんでした。 →EEPROM アドバンス関数をコールして通信を終了させてください。 R_IIC_EEP_ERR_PARAM パラメータエラーです。 →設定値を確認してください。 R_IIC_EEP_ERR_AL アービトレーションロストが発生しています。 →EEPROM 復帰関数をコールすることで復帰処理を行うことができます。 R_IIC_EEP_ERR_NON_REPLY 未応答エラーが発生しています。（※１） →EEPROM 復帰関数をコールすることで復帰処理を行うことができます。 R_IIC_EEP_ERR_SDA_LOW_HOLD SDA は Low ホールドから復帰できていない状態です。 →スレーブデバイスが Low ホールドしていないか、マスタデバイスから Low 信号が出力されていないか等、システムの状態を確認してください。 R_IIC_EEP_ERR_OTHER その他エラーが発生しています。 →以下を確認してください。 ・EEPROM 通信情報構造体の設定が正しく行われているか確認してください。 ・OS 制御でエラーが発生していないか確認してください。 ■リターン値（本関数コール後、R_IIC_EepMdl_Advance()コール時） R_IIC_EEP_COMMUNICATION 通信中です。 →EEPROM アドバンス関数をコールして通信を終了させてください。 R_IIC_EEP_FINISH_ACKPOL 全データの書き換えが完了しました。

→開始関数をコールすることで通信が可能です。

R_IIC_EEP_FINISH_ACKPOL_AGN

ページ書き換えが完了しました。残データがあります。

→Write 開始関数をコールしてください。この際、EEPROM 通信情報構造体メンバの情報は変更しないでください。（※２）

R_IIC_EEP_FINISH_ACKPOL_NACK

ページ書き換えは完了していません。（スレーブアドレス送信後、NACK 受信）

→Acknowledge polling 開始関数をコールすることで、EEPROM への書き換えが完了したか判定することができます。（※３）

エラー時のリターン値は、EEPROM アドバンス関数のリターン値を参照ください。

備考

・I^2C シングルマスタ制御ソフトウェアのマスタ送信モード（パターン 3）を使用します。

・本処理を行うためには、構造体“r_iic_eepmdl_info_t”の設定が必要です。設定方法は、5.11.1 を参照ください。

・本関数から戻った時点では、I^2C 通信は完了していません。I^2C 通信を終了させるためには、EEPROM アドバンス関数をコールする必要があります。

・ページ境界を跨いだデータ書き換えを行う場合は、EEPROM アドバンス関数で通信を完了させた後、再度 Write 開始関数をコールしてください。

※１：未応答エラーは、制御する MCU によって定義が異なります。詳しくは I^2C シングルマスタ制御ソフトウェアの未応答エラーの説明をご参照ください。

※２：EEPROM 通信情報構造体メンバには、残データの書き込み情報が格納されているため、メンバの値を変更した場合、正常に通信を行うことができなくなります。

※３：書き換え判定を行わず Write/Read 開始関数をコールすることも可能です。この場合、EEPROM 書き換え完了待ちを行った後、開始関数をコールしてください。尚、書き込み完了時間（Write cycle time）は使用する EEPROM のデータシートをご参照ください。

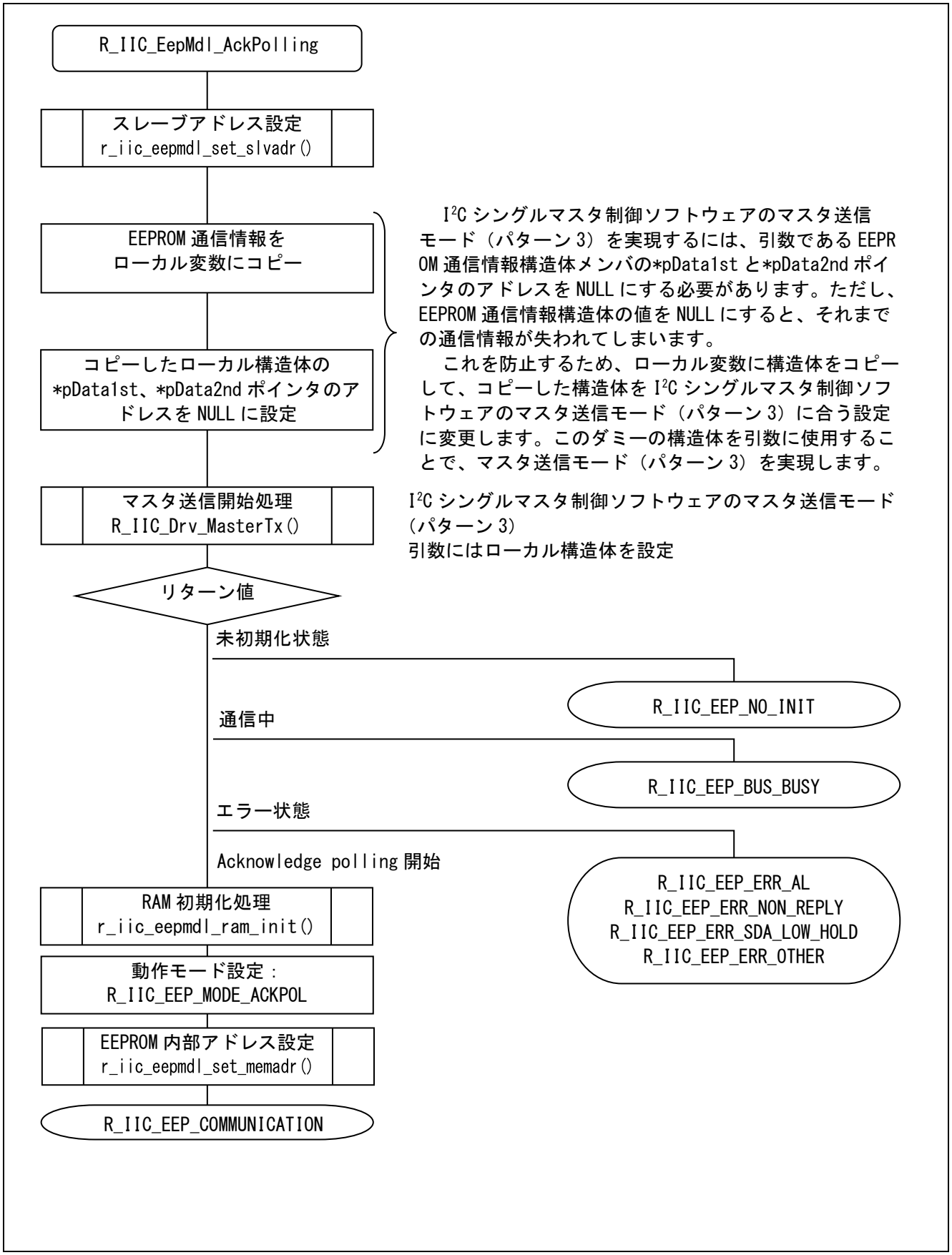


図 5-17　Acknowledge polling 開始関数概要

5.16.5 Read 開始関数

R_IIC_EepMdl_Read	
概要	Read 開始関数
ヘッダ	r_iic_eepmdl_api.h, r_iic_eepmdl_sub.h
宣言	error_t R_IIC_EepMdl_Read(r_iic_eepmdl_info_t FAR *pEep_Info)
説明	・EEPROM からデータ読み出しを開始します。 ・マスタ（MCU）がスレーブ（EEPROM）からデータを読み出すために使用してください。
引数	r_iic_eepmdl_info_t FAR *pEep_Info ;EEPROM 通信情報格納ポインタ
リターン値	■リターン値（R_IIC_EepMdl_Read()コール時） R_IIC_EEP_COMMUNICATION EEPROM から読み出しを開始しました。 →EEPROM アドバンス関数をコールして通信を終了させてください。 R_IIC_EEP_NO_INIT 未初期化状態です。 →EEPROM 初期化関数をコールして初期化してください。 R_IIC_EEP_LOCK_FUNC 他の API 処理が行われているため、処理は行われませんでした。 →他の API 処理終了後、本関数をコールしてください。 R_IIC_EEP_BUS_BUSY 通信中です。EEPROM からの読み出しを開始できませんでした。 →EEPROM アドバンス関数をコールして通信を終了させてください。 R_IIC_EEP_ERR_PARAM パラメータエラーです。 →設定値を確認してください。 R_IIC_EEP_ERR_AL アービトレーションロストが発生しています。 →EEPROM 復帰関数をコールすることで復帰処理を行うことができます。 R_IIC_EEP_ERR_NON_REPLY 未応答エラーが発生しています。（※１） →EEPROM 復帰関数をコールすることで復帰処理を行うことができます。 R_IIC_EEP_ERR_SDA_LOW_HOLD SDA は Low ホールドから復帰できていない状態です。 →スレーブデバイスが Low ホールドしていないか、マスタデバイスから Low 信号が出力されていないか等、システムの状態を確認してください。 R_IIC_EEP_ERR_OTHER その他エラーが発生しています。 →以下を確認してください。 ・EEPROM 通信情報構造体の設定が正しく行われているか確認してください。 ・OS 制御でエラーが発生していないか確認してください。 ■リターン値（本関数コール後、R_IIC_EepMdl_Advance()コール時） R_IIC_EEP_COMMUNICATION 通信中です。 →EEPROM アドバンス関数をコールして通信を終了させてください。 R_IIC_ EEP_FINISH_READ 全データの読み出しが完了しました。 →開始関数をコールすることで通信が可能です。

エラー時のリターン値は、EEPROM アドバンス関数のリターン値を参照ください。

備考
・I^2C シングルマスタ制御ソフトウェアのマスタ複合モードを使用します。
・本処理を行うためには、構造体“r_iic_eepmdl_info_t”の設定が必要です。設定方法は、5.11.1 を参照ください。
・本関数から戻った時点では、I^2C 通信は完了していません。I^2C 通信を終了させるためには、EEPROM アドバンス関数をコールする必要があります。
※１：未応答エラーは、制御する MCU によって定義が異なります。詳しくは I^2C シングルマスタ制御ソフトウェアの未応答エラーの説明をご参照ください。

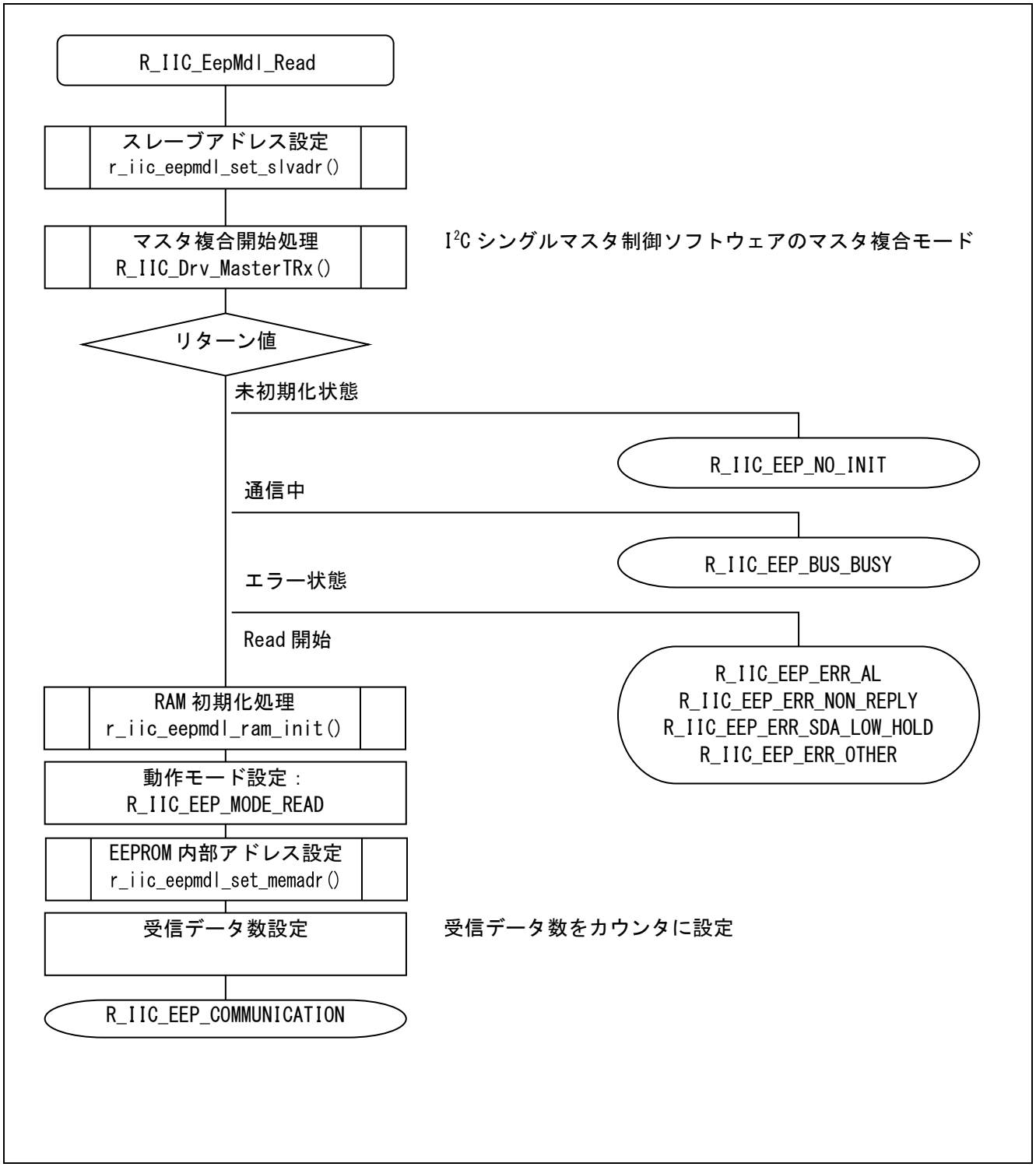


図 5-18　Read 開始関数概要

5.16.6 EEPROM アドバンス関数

R_IIC_EepMdl_Advance	
概要	EERPROM アドバンス関数
ヘッダ	r_iic_eepmdl_api.h, r_iic_eepmdl_sub.h
宣言	error_t R_IIC_EepMdl_Advance(r_iic_eepmdl_info_t FAR *pEep_Info)
説明	・通信を監視し通信を進める処理を実行します。リターン値に通信の状態を返します。
引数	r_iic_eepmdl_info_t FAR *pEep_Info ;EEPROM 通信情報構造体ポインタ
リターン値	R_IIC_EEP_COMMUNICATION 通信中です。 →EEPROM アドバンス関数をコールして通信を終了させてください。 R_IIC_EEP_FINISH_WRITE 全データの書き込みが完了しました。 →Acknowledge polling 開始関数をコールすることで、EEPROM への書き換えが完了したか判定することができます。（※１） R_IIC_EEP_FINISH_WRITE_AGN ページ書き込みが完了しました。残データがあります。 →書き換え完了後、Write 開始関数をコールしてください。この際、EEPROM 通信情報構造体メンバの情報は変更しないでください。（※２） R_IIC_EEP_FINISH_ACKPOL 全データの書き換えが完了しました。 →開始関数をコールすることで通信が可能です。 R_IIC_EEP_FINISH_ACKPOL_AGN ページ書き換えが完了しました。残データがあります。 →Write 開始関数をコールしてください。この際、EEPROM 通信情報構造体メンバの情報は変更しないでください。（※２） R_IIC_EEP_FINISH_ACKPOL_NACK ページ書き換えは完了していません。（スレーブアドレス送信後、NACK 受信） →Acknowledge polling 開始関数をコールすることで、EEPROM への書き換えが完了したか判定することができます。（※１） R_IIC_ EEP_FINISH_READ 全データの読み出しが完了しました。 →開始関数をコールすることで通信が可能です。 R_IIC_EEP_LOCK_FUNC 他の API 処理が行われているため、処理は行われませんでした。 →他の API 処理終了後、本関数をコールしてください。 R_IIC_EEP_BUS_BUSY 同一チャネル上の他のデバイスが通信中のため、処理は行われませんでした。 →他のデバイスの通信を終了させてください。 R_IIC_EEP_NO_INIT 未初期化状態です。 →EEPROM 初期化関数をコールして初期化してください。 R_IIC_EEP_IDLE アイドル状態です。 →開始関数をコールすることで通信が可能です。 R_IIC_EEP_ERR_PARAM パラメータエラーです。 →設定値を確認してください。 R_IIC_EEP_ERR_AL

アービトレーションロストが発生しています。
→EEPROM 復帰関数をコールすることで復帰処理を行うことができます。

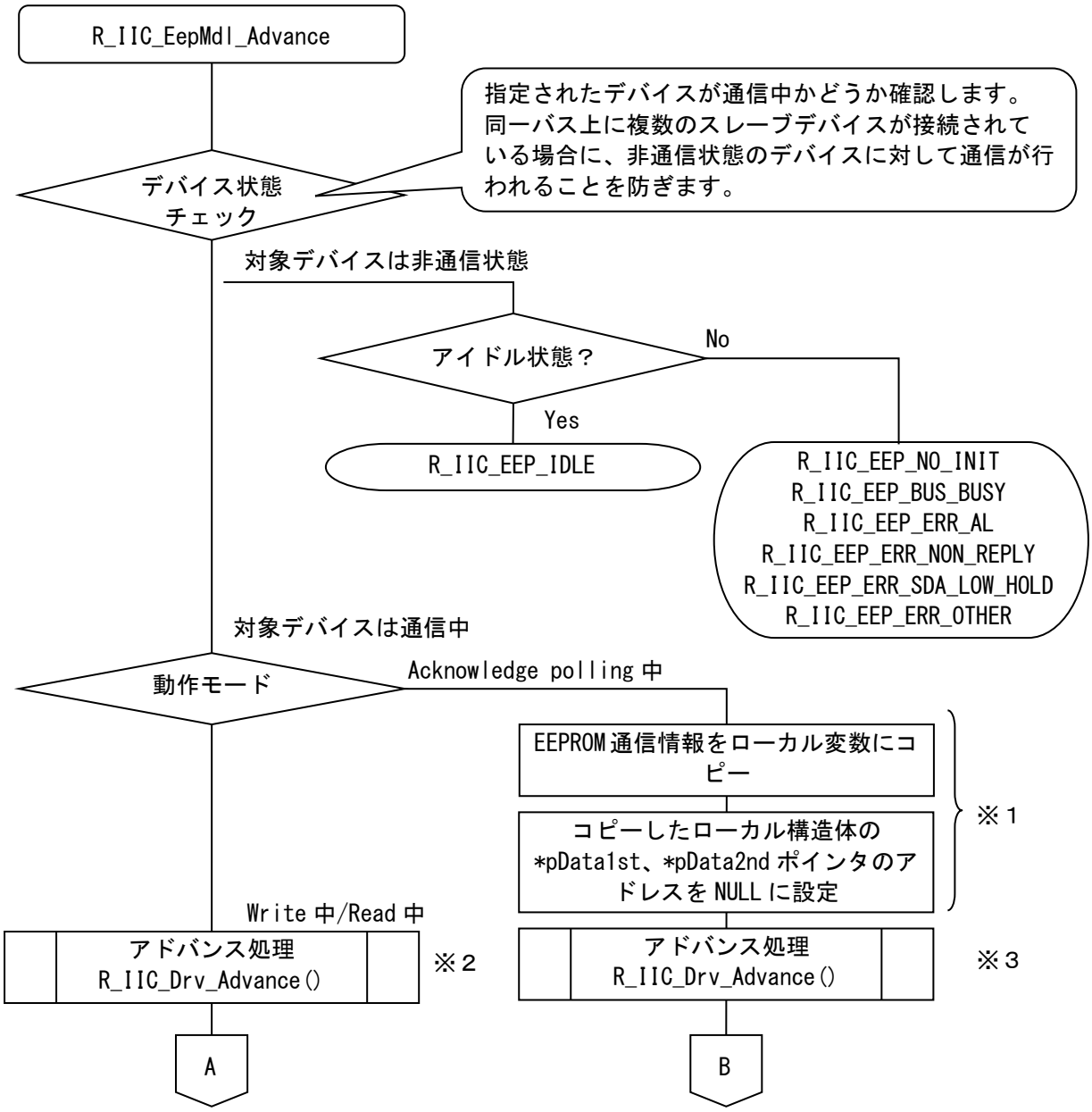
R_IIC_EEP_ERR_NON_REPLY
未応答エラーが発生しています。（※３）
→EEPROM 復帰関数をコールすることで復帰処理を行うことができます。

R_IIC_EEP_ERR_SDA_LOW_HOLD
SDA は Low ホールドから復帰できていない状態です。
→スレーブデバイスが Low ホールドしていないか、マスタデバイスから Low 信号が出力されていないか等、システムの状態を確認してください。

R_IIC_EEP_ERR_OTHER
その他エラーが発生しています。
→以下を確認してください。
・EEPROM 通信情報構造体の設定が正しく行われているか確認してください。
・OS 制御でエラーが発生していないか確認してください。

R_IIC_EEP_ERR_NACK
Write 通信中に NACK を受信しました。
→以下を確認してください。
・ライトプロテクト端子（WP）が High の場合、Low に設定後 Write 開始関数をコールしてください。書き込み予定データ数と書き込んだデータが不一致の場合、EEPROM 復帰関数をコールすることで復帰処理を行うことができます。

備考
・各開始関数コール後は、EEPROM アドバンス関数をコールして通信を完了させてください。
・コールした開始関数により、通信完了時の EEPROM アドバンス関数のリターン値が異なります。本項の説明、もしくは各開始関数で説明している EEPROM アドバンス関数のリターン値をご確認ください。
・通信途中でエラーが発生した場合には、復帰処理後、再度データを設定し直してから開始関数をコールしてください。
※１：書き換え判定を行わず Write/Read 開始関数をコールすることも可能です。この場合、EEPROM 書き換え完了待ちを行った後、開始関数をコールしてください。尚、書き込み完了時間（Write cycle time）は使用する EEPROM のデータシートをご参照ください。
※２：EEPROM 通信情報構造体メンバには、残データの書き込み情報が格納されているため、メンバの値を変更した場合、正常に通信を行うことができなくなります。
※３：未応答エラーは、制御する MCU によって定義が異なります。詳しくは I^2C シングルマスタ制御ソフトウェアの未応答エラーの説明をご参照ください。

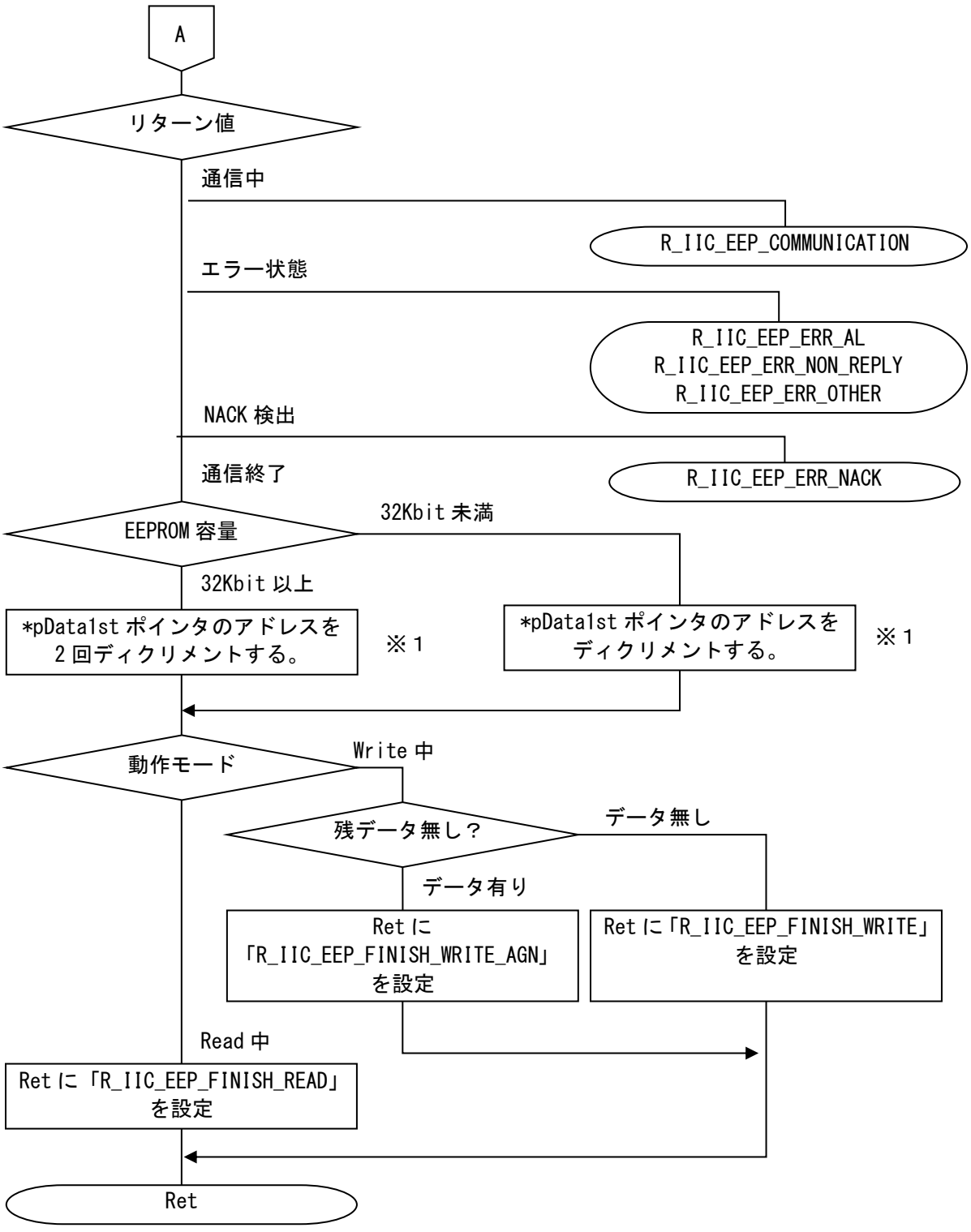


※１：I²C シングルマスタ制御ソフトウェアのマスタ送信モード（パターン 3）を実現するには、引数である EEPROM 通信情報構造体メンバの*pData1st と*pData2nd ポインタのアドレスを NULL にする必要があります。ただし、EEPROM 通信情報構造体の値を NULL にすると、それまでの通信情報が失われてしまいます。

これを防止するため、ローカル変数に構造体をコピーして、コピーした構造体を I²C シングルマスタ制御ソフトウェアのマスタ送信モード（パターン 3）に合う設定に変更します。このダミーの構造体を引数に使用することで、マスタ送信モード（パターン 3）を実現します。

※２：引数には EEPROM 通信情報構造体を設定
※３：引数にはローカル構造体を設定

図 5-19　EEPROM アドバンス関数概要（1/3）



※１：
「*pData1st」は EEPROM 内部アドレス格納バッファポインタです。通信終了時には、ポインタのアドレスがインクリメントされているため、通信開始前の位置に戻す必要があります。
なお、EEPROM 容量によりインクリメントされる値は異なります。32Kbit 以上の場合、内部アドレスは 2 バイト送信するため、2 回ディクリメントを行います。32Kbit 未満では、1 回ディクリメントを行います。

図 5-20　EEPROM アドバンス関数概要（2/3）

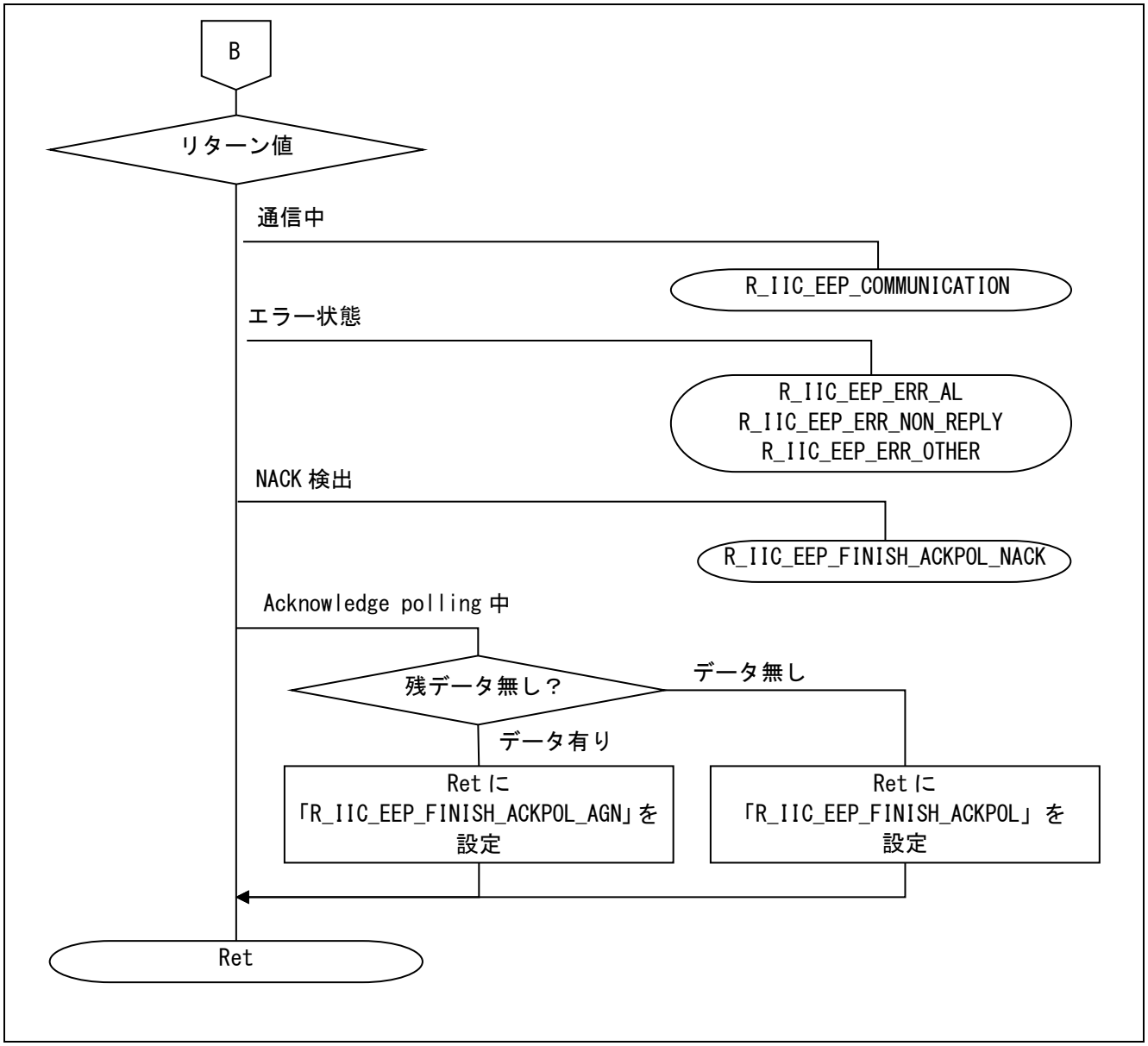
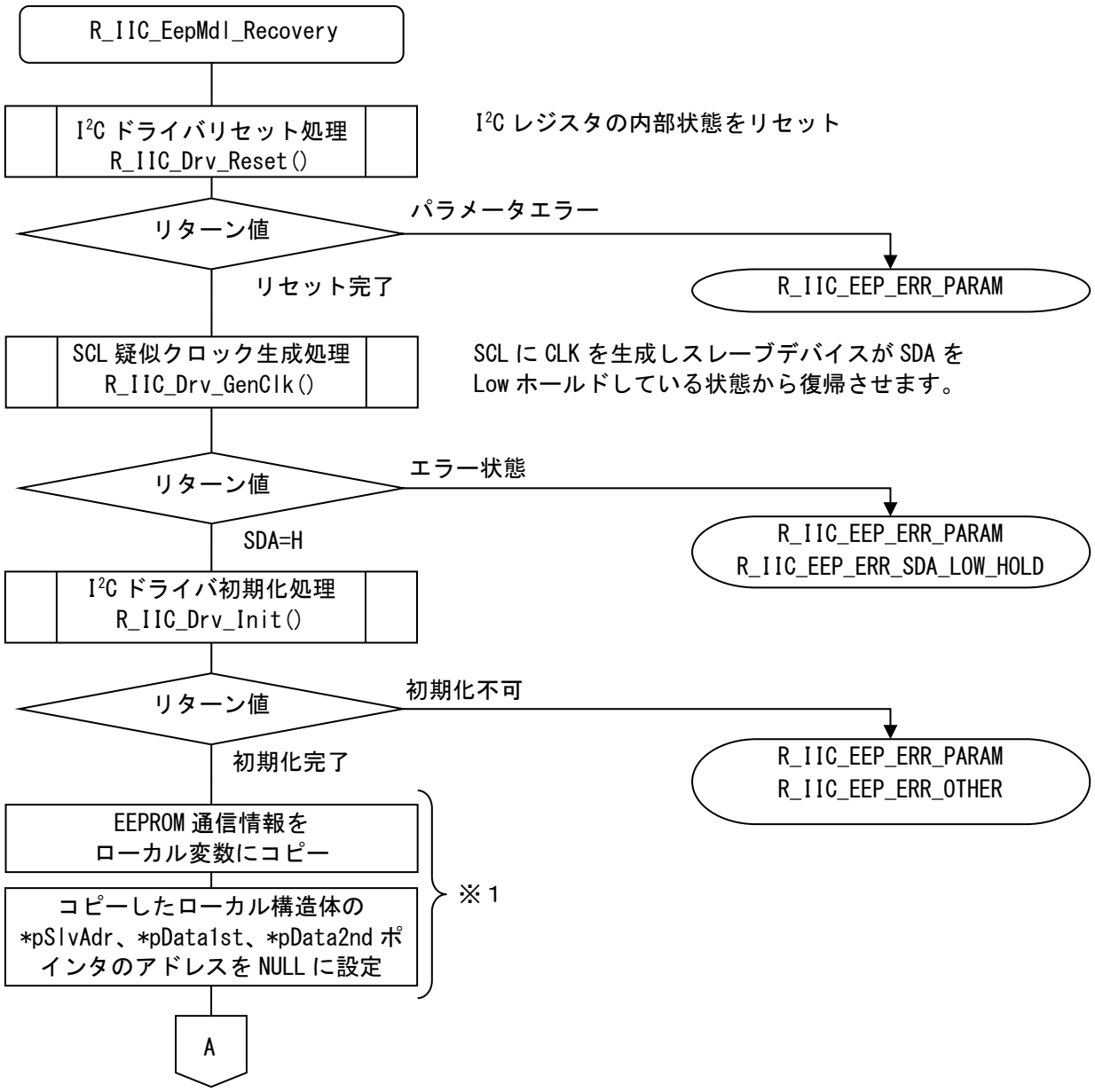


図 5-21　EEPROM アドバンス関数概要（3/3）

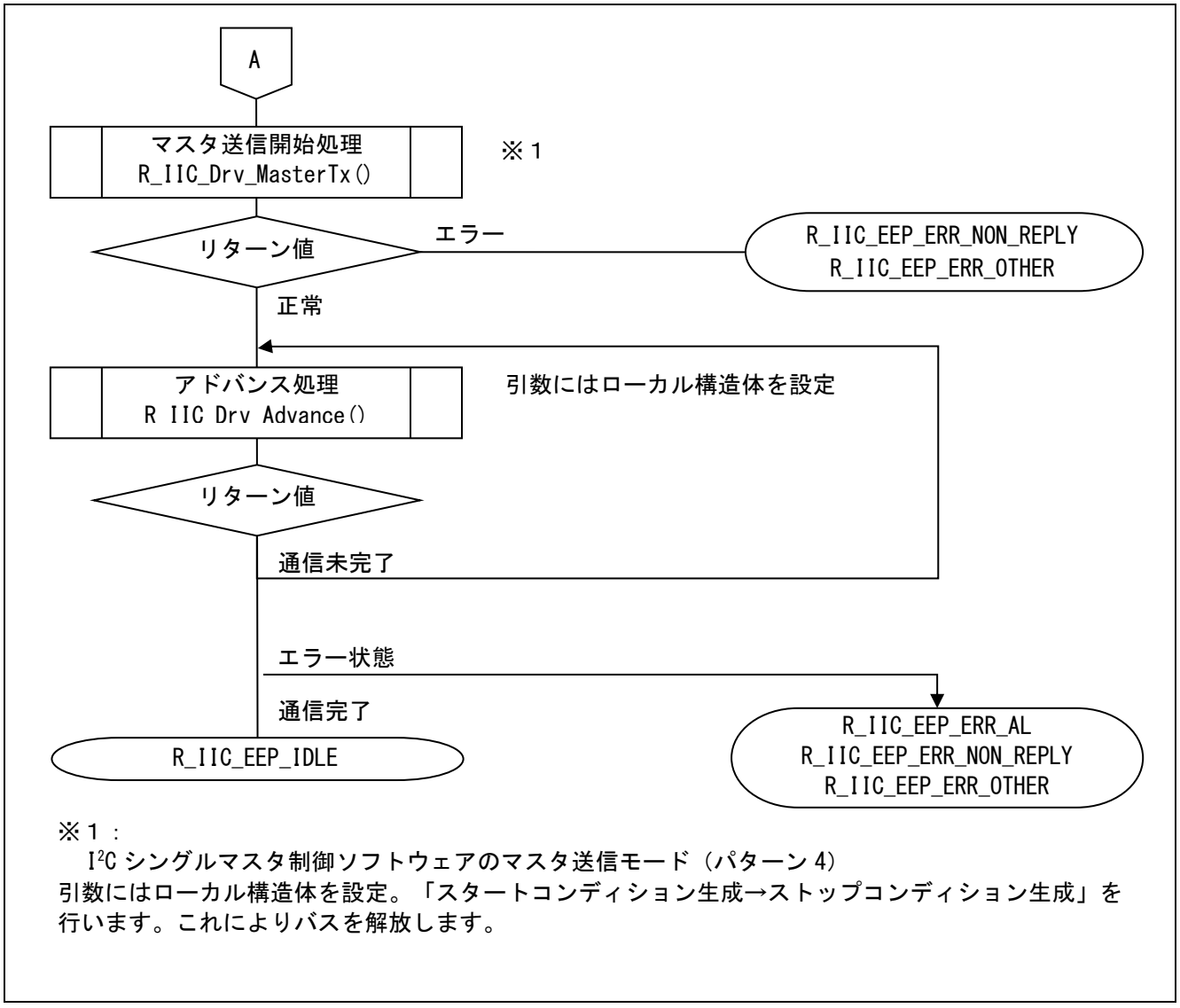
5.16.7 EEPROM 復帰関数

R_IIC_EepMdl_Recovery	
概要	EEPROM 復帰関数
ヘッダ	r_iic_eepmdl_api.h, r_iic_eepmdl_sub.h,
宣言	error_t R_IIC_EepMdl_Recovery(r_iic_eepmdl_info_t FAR *pEep_Info)
説明	・通信エラーとなった場合に、復帰処理を行うことができます。 ・強制的に初期化したい場合、本関数をコールしてください。 ・本関数コール後、アイドル状態になります。
引数	r_iic_eepmdl_info_t FAR *pEep_Info ;EEPROM 通信情報格納ポインタ
リターン値	R_IIC_EEP_IDLE EEPROM 復帰処理が正常に終了し、アイドル状態になりました。 →開始関数をコールすることで通信が可能です。 R_IIC_EEP_LOCK_FUNC 他の API 処理が行われているため、処理は行われませんでした。 →他の API 処理終了後、本関数をコールしてください。 R_IIC_EEP_ERR_PARAM パラメータエラーです。 →設定値を確認してください。 R_IIC_EEP_ERR_AL アービトレーションロストが発生しています。 →再度 EEPROM 復帰関数をコールすることで復帰処理を行うことができます。 R_IIC_EEP_ERR_NON_REPLY 未応答エラーが発生しています。（※１） →再度 EEPROM 復帰関数をコールすることで復帰処理を行うことができます。 R_IIC_EEP_ERR_SDA_LOW_HOLD EEPROM 復帰処理を行いましたが SDA は Low ホールドから復帰できていない状態です。 →スレーブデバイスが Low ホールドしていないか、マスタデバイスから Low 信号が出力されていないか等、システムの状態を確認してください。 R_IIC_EEP_ERR_OTHER その他エラーが発生しています。 →以下を確認してください。 ・スレーブデバイスが SCL を Low ホールドしていないか、マスタデバイスから Low 信号が出力されていないか等、システムの状態を確認してください。 ・EEPROM 通信情報構造体の設定が正しく行われているか確認してください。 ・OS 制御でエラーが発生していないか確認してください。
備考	・I²C 内部リセットを行います。 ・リセット後、SDA が Low ホールドしている場合、SCL に疑似クロックを生成します。生成する回数はマクロ定義 SCL_CLK_CNT で設定可能です。（マクロ定義は 5.10.1 の表 5-14 を参照） ・I²C シングルマスタ制御ソフトウェアのマスタ送信モード（パターン 4）を使用し、スタートコンディションとストップコンディションを生成して、バスを解放します。 ・本処理実行後も通信に復帰できない場合、SDA が GND に固定されている等の異常が考えられます。 ※１：未応答エラーは、制御する MCU によって定義が異なります。詳しくは I²C シングルマスタ制御ソフトウェアの未応答エラーの説明をご参照ください。



※１：
　I²C シングルマスタ制御ソフトウェアのマスタ送信モード（パターン 4）を実現するには、引数である EEPROM 通信情報構造体メンバの*pSlvAdr と*pData1st と*pData2nd ポインタのアドレスを NULL にする必要があります。ただし、EEPROM 通信情報構造体の値を NULL にすると、それまでの通信情報が失われてしまいます。
　これを防止するため、ローカル変数に構造体をコピーして、コピーした構造体を I²C シングルマスタ制御ソフトウェアのマスタ送信モード（パターン 4）に合う設定に変更します。このダミーの構造体を引数に使用することで、マスタ送信モード（パターン 4）を実現します。

図 5-22　EEPROM 復帰関数概要（1/2）



※１：
　I²C シングルマスタ制御ソフトウェアのマスタ送信モード（パターン 4）
引数にはローカル構造体を設定。「スタートコンディション生成→ストップコンディション生成」を行います。これによりバスを解放します。

図 5-23　EEPROM 復帰関数概要（2/2）

6. 応用例

6.1 r_iic_eepmdl_api.h

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) 使用する MCU 選択

使用する MCU を指定してください。

```
/*-------------------------------------------------------------------------*/
/*   Select to use MCU Type.                                               */
/*-------------------------------------------------------------------------*/
#define MCU_RL78
/* #define MCU_RX       */
```

(2) アクセスする RAM 領域の定義

M16C 使用時、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。

M16C 以外での設定は無効です。

下記の例は、FAR 領域を使用する場合です。

```
/*-------------------------------------------------------------------------*/
/* If using a M16C, define the RAM area to be accessed by the user process.*/
/* Please choose one of definitions.                                        */
/* Efficient operations for standard functions and processes are applied.  */
/*-------------------------------------------------------------------------*/
#if defined(MCU_M16C)
#define R_IIC_FAR
/* #define R_IIC_NEAR */
#endif  /* #if defined(M16C) */
```

(3) SCL 疑似クロックカウンタ値の定義

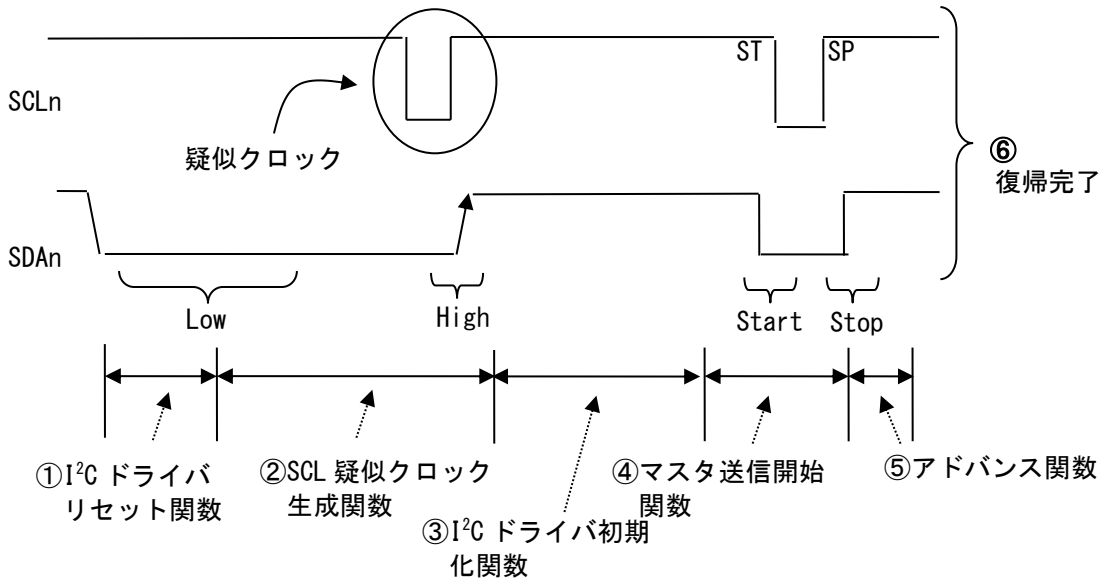
EEPROM 復帰関数では、SDA が Low ホールドしている場合、SCL に疑似クロックを生成します。このクロック回数を定義してください。

尚、I^2C の通信単位が 9 クロックの場合が多いため、本サンプルコードでは 9 に設定しています。

```
/*-------------------------------------------------------------------------*/
/*   Counter                                                               */
/*-------------------------------------------------------------------------*/
#define  SCL_CLK_CNT (uint8_t)(9)  /* Clock counter to SCL when recovers
processing*/
```

6.2 EEPROM 復帰関数について

通信中に意図しない電源の瞬断やノイズの発生により、SDA または SCL が Low ホールドした場合の通信への EEPROM 復帰処理を図 6-1 に示します。処理が完了するとアイドル状態になります。



①I²C 内部リセットを行います。
②SCL に疑似クロックを生成し、スレーブデバイスの Low ホールド状態から解放させます。
③マスタ送信（パターン 4）を行うため、初期化を行います。
④マスタ送信（パターン 4）を開始します。スタートコンディション生成後、ストップコンディションを生成します。
⑤アドバンス関数により通信を終了させます。
⑥アイドル状態になりました。以降、各開始関数をコールすることで通信が可能です。

図 6-1 EEPROM 復帰関数動作概要

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、以下をインクルードしてください。

・r_iic_eepmdl_api.h
・r_iic_eepmdl_sub.h
・r_iic_drv_api.h
・r_iic_drv_sub.h
・r_iic_drv_sfr.h
・r_iic_drv_int.h
・r_iic_drv_os.h

7.2 ページサイズ設定について

EEPROM 通信情報構造体メンバ“PageSize”にページサイズを設定する必要があります。設定する際は、使用する EEPROM で規定されているページサイズを設定してください。同一容量であっても、EEPROM 製品名により、ページサイズが異なる場合があります。

本サンプルコードでは、使用する EEPROM のページサイズとは異なる設定した場合でも通信を行います。このため、EEPROM で規定されているサイズよりも大きい値を設定した場合、書き込みで Roll Over が発生しますのでご注意ください。

7.3 Acknowledge polling 開始関数コール時の構造体の取扱いについて

EEPROM の Acknowledge polling を行う際、I^2C シングルマスタ制御ソフトウェアのマスタ送信パターン 3 を使用しますが、この際、EEPROM 通信情報構造体メンバの*pData1st と*pData2nd ポインタのアドレスを NULL にする必要があります。

Write 動作を完了後、Acknowledge polling 開始関数や EEPROM アドバンス関数をコールする際には、通信中の EEPROM 通信情報構造体を引数に設定する必要があります。これらの関数処理中に、この構造体メンバの*pData1st や*pData2nd ポインタのアドレスを NULL にすると、通信情報が失われてしまいます。

これを防止するため、Acknowledge polling 開始関数や EEPROM アドバンス関数では、引数の構造体をローカル変数へコピーし、コピーした構造体を I^2C シングルマスタ制御ソフトウェアの関数の引数に設定します。コピーした構造体で処理を行うことで、それまでの通信情報が失われるのを防ぎます。

7.4 EEPROM 復帰関数コール時の構造体の取扱について

Acknowledge polling 開始関数と同様、EEPROM 復帰関数では I^2C シングルマスタ制御ソフトウェアのマスタ送信パターン 4 を使用します。この際、EEPROM 通信情報構造体メンバの*pSlvAdr、*pData1st、*pData2nd ポインタのアドレスを NULL にする必要があります。7.2 項の方法と同様、EEPROM 復帰関数内でコピーした構造体で処理を行います。

7.5 EEPROM 復帰関数コール後の通信について

EEPROM 復帰関数コール後に通信を再開する場合、それまでの通信情報が失われている可能性があるため、通信の最初からやり直してください。

7.6 EEPROM アドバンス関数を割り込み内で処理する場合と OS 制御について

EEPROM アドバンス関数を割り込み内で処理する場合と OS 制御（※）については、未検証です。使用される場合は十分に評価を行い、必要に応じて修正してください。

※：本サンプルコードの OS 制御は、μITRON4.0 を想定しています。

7.7 同一バス上に複数デバイスを接続する際の注意事項

Serial EEPROM は使用するデバイス容量により接続可能なスレーブデバイス数が異なります。1 つのチャネル上に複数のスレーブデバイスを接続する場合はご注意ください。接続可能なスレーブデバイス数については表 5-2 を参照ください。

7.8 コンパイル時の注意事項

CC-RL コンパイラでコンパイルした場合、警告メッセージ「W0520111:文は実行されません。」が出力されます。

これは break 文の直前で return しているため break 文が実行されないという警告メッセージです。動作に影響がないため無視して問題ありません。

ホームページとサポート窓口

ルネサス エレクトロニクスホームページ
http://japan.renesas.com/

お問合せ先
http://japan.renesas.com/contact/

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2012.12.28	—	初版発行
1.01	2014.03.31	6	2. 動作確認条件　において、 見出しに「(1) RL78/G14 IICA 統合開発環境 CubeSuite+の場合」を追加した。
		7	2. 動作確認条件　において、以下を追加した。 (2) RL78/G14 IICA 統合開発環境 IAR Embedded Workbench の場合
		9	図 4-2 にて、 「同一チャネル上で、同時に複数デバイスは通信できない。」 元は、「同一バス上で、同時に複数デバイスは通信できない。」であった。
		13	5.2.2. Write 動作　(2)データ書き込み（WP=H 時）　にて、 「データの送信後は」　元は、「送信データ入力後は」であった。
		14	5.2.2. Write 動作　(3)Acknowledge polling　にて、 「ストップコンディションが受信されると」　元は、「ストップコンディションが入力されると」であった。
		20	5.6.1. Write 動作フロー　図 5-8　において、 吹き出し内の「コールしてください」　元は、「コールする」であった。
		21	5.6.1. Write 動作フロー　図 5-9　において、 吹き出し内の「コールしてください」　元は、「コールする」であった。
		24	5.8 必要メモリサイズ　にて、 「(1)RL78/G14 IICA 統合開発環境 CubeSuite+の場合」　元は、「(1)RL78 IICA の場合」であった。
		24	5.8 必要メモリサイズ　にて、以下を追加した。 「(1)RL78/G14 IICA 統合開発環境 IAR Embedded Workbench の場合」
		25	5.9 ファイル構成　にて、アプリケーションノートの更新に伴い、表 5-6 を更新した。
		29	5.11.1 EEPROM 通信情報構造体　表 5-11　構造体メンバ *pData1st　の　内容欄において、 「書き込み時または読み出し時」　元は、「Write 時または Read 時」であった。
		33	5.16.1 関数共通処理　図 5-14 において、関数開始直後にパラメータチェックを追加した。
		34	5.16.2 EEPROM 初期化関数　説明欄において、 「デバイス状態フラグ（*(pEep.Info.RIic_Info.pDevStatus)）」の記述を左に移動させた。
		35	5.16.2 EEPROM 初期化関数　図 5-15 において、関数開始直後のパラメータチェックを削除した。
		38	5.16.3 Write 開始関数　図 5-16 において、関数開始直後のパラメータチェックを削除した。
		41	5.16.4 Acknowledge polling 開始関数　図 5-17 において、関数開始直後のパラメータチェックを削除した。
		44	5.16.5 Read 開始関数　図 5-18 において、関数開始直後のパラメータチェックを削除した。

		46	5.16.6 EEPROM アドバンス関数　リターン値　において、 「ライトプロテクト端子（WP）が High の場合、Low に設定後 Write 開始関数をコールしてください。」　元は、「ライトプロテクト端子（WP）が High の場合、Low にしてから Write 開始関数をコールしてください。」であった。
		47	5.16.6 EEPROM アドバンス関数　図 5-19 において、関数開始直後のパラメータチェックを削除した。
		51	5.16.7 EEPROM 復帰関数　図 5-22 において、関数開始直後のパラメータチェックを削除した。
1.02	2014.9.30	1	要旨　に I^2C シングルマスタ制御ソフトウェア情報の URL を追加した。
		1	対象デバイスに、RL78/G1C、RL78/L1C、RL78/L12、RL78/L13 を追加
		6-10	2. 動作確認条件　に以下を追加 2.1 RL78 の場合 (3) RL78/G1C IICA 統合開発環境 CubeSuite+の場合 (4) RL78/G1C IICA 統合開発環境 IAR Embedded Workbench の場合 (5) RL78/L12 IICA 統合開発環境 CubeSuite+の場合 (6) RL78/L12 IICA 統合開発環境 IAR Embedded Workbench の場合 (7) RL78/L13 IICA 統合開発環境 CubeSuite+の場合 (8) RL78/L13 IICA 統合開発環境 IAR Embedded Workbench の場合 (9) RL78/L1C IICA 統合開発環境 CubeSuite+の場合 (10) RL78/L1C IICA 統合開発環境 IAR Embedded Workbench の場合
		11-13	2. 動作確認条件　に以下を追加 2.2 RX の場合 (1) RX62N RIIC の場合 (2) RX63N RIIC の場合 (3) RX63T RIIC の場合 (4) RX210 RIIC の場合 (5) RX21A RIIC の場合
		13	3. 関連アプリケーションノート　以下を更新した。 RL78/G14,RL78/G1C,RL78/L12,RL78/L13,RL78/L1C グループ IICA を使った I^2C シングルマスタ制御ソフトウェア (R01AN1074JJ)
		13	3.関連アプリケーションノート　以下を追加した。 RX600、RX200 シリーズ　RIIC を使った I^2C シングルマスタ制御ソフトウェア(R01AN1254JJ)
		30-32	5.8 必要メモリサイズ　以下を追加した。 5.8.1.RL78 の場合 (3) RL78/L13 IICA 統合開発環境 CubeSuite+の場合 (4) RL78/L13 IICA 統合開発環境 IAR Embedded Workbench の場合 5.8.2.RX の場合 (1) RX63N RIIC の場合 (2) RX210 RIIC の場合
		33	5.9 ファイル構成 アプリケーションノート番号を変更
1.03	2016.3.31	6	2. 動作確認条件

			(1) RL78/G14 IICA 統合開発環境 CS+ for CA,CX の場合（コンパイラ：CA78K0R） で情報を更新した。 (2) RL78/G14 IICA 統合開発環境 CS+ for CC の場合（コンパイラ：CC-RL） を追加した。
		31	5.8 必要メモリサイズ 5.8.1 RL78 の場合 (1) RL78/G14 IICA 統合開発環境 CS+ for CA,CX の場合（コンパイラ：CA78K0R） で情報を更新した。 (2) RL78/G14 IICA 統合開発環境 CS+ for CC の場合（コンパイラ：CC-RL） を追加した。
		34	5.9 ファイル構成 アプリケーションノート番号を更新した。
		65	7.8 コンパイル時の注意事項 を追加した。

製品ご使用上の注意事項

ここでは、マイコン製品全体に適用する「使用上の注意事項」について説明します。個別の使用上の注意事項については、本ドキュメントおよびテクニカルアップデートを参照してください。

1. 未使用端子の処理
   【注意】未使用端子は、本文の「未使用端子の処理」に従って処理してください。
   CMOS製品の入力端子のインピーダンスは、一般に、ハイインピーダンスとなっています。未使用端子を開放状態で動作させると、誘導現象により、LSI周辺のノイズが印加され、LSI内部で貫通電流が流れたり、入力信号と認識されて誤動作を起こす恐れがあります。未使用端子は、本文「未使用端子の処理」で説明する指示に従い処理してください。
2. 電源投入時の処置
   【注意】電源投入時は，製品の状態は不定です。
   電源投入時には、LSIの内部回路の状態は不確定であり、レジスタの設定や各端子の状態は不定です。
   外部リセット端子でリセットする製品の場合、電源投入からリセットが有効になるまでの期間、端子の状態は保証できません。
   同様に、内蔵パワーオンリセット機能を使用してリセットする製品の場合、電源投入からリセットのかかる一定電圧に達するまでの期間、端子の状態は保証できません。
3. リザーブアドレス（予約領域）のアクセス禁止
   【注意】リザーブアドレス（予約領域）のアクセスを禁止します。
   アドレス領域には、将来の機能拡張用に割り付けられているリザーブアドレス（予約領域）があります。これらのアドレスをアクセスしたときの動作については、保証できませんので、アクセスしないようにしてください。
4. クロックについて
   【注意】リセット時は、クロックが安定した後、リセットを解除してください。
   プログラム実行中のクロック切り替え時は、切り替え先クロックが安定した後に切り替えてください。
   リセット時、外部発振子（または外部発振回路）を用いたクロックで動作を開始するシステムでは、クロックが十分安定した後、リセットを解除してください。また、プログラムの途中で外部発振子（または外部発振回路）を用いたクロックに切り替える場合は、切り替え先のクロックが十分安定してから切り替えてください。
5. 製品間の相違について
   【注意】型名の異なる製品に変更する場合は、製品型名ごとにシステム評価試験を実施してください。
   同じグループのマイコンでも型名が違うと、内部ROM、レイアウトパターンの相違などにより、電気的特性の範囲で、特性値、動作マージン、ノイズ耐量、ノイズ輻射量などが異なる場合があります。型名が違う製品に変更する場合は、個々の製品ごとにシステム評価試験を実施してください。

ご注意書き

1. 本資料に記載された回路、ソフトウェアおよびこれらに関連する情報は、半導体製品の動作例、応用例を説明するものです。お客様の機器・システムの設計において、回路、ソフトウェアおよびこれらに関連する情報を使用する場合には、お客様の責任において行ってください。これらの使用に起因して、お客様または第三者に生じた損害に関し、当社は、一切その責任を負いません。
2. 本資料に記載されている情報は、正確を期すため慎重に作成したものですが、誤りがないことを保証するものではありません。万一、本資料に記載されている情報の誤りに起因する損害がお客様に生じた場合においても、当社は、一切その責任を負いません。
3. 本資料に記載された製品データ、図、表、プログラム、アルゴリズム、応用回路例等の情報の使用に起因して発生した第三者の特許権、著作権その他の知的財産権に対する侵害に関し、当社は、何らの責任を負うものではありません。当社は、本資料に基づき当社または第三者の特許権、著作権その他の知的財産権を何ら許諾するものではありません。
4. 当社製品を改造、改変、複製等しないでください。かかる改造、改変、複製等により生じた損害に関し、当社は、一切その責任を負いません。
5. 当社は、当社製品の品質水準を「標準水準」および「高品質水準」に分類しており、
   各品質水準は、以下に示す用途に製品が使用されることを意図しております。
   標準水準： コンピュータ、OA機器、通信機器、計測機器、AV機器、
   家電、工作機械、パーソナル機器、産業用ロボット等
   高品質水準：輸送機器（自動車、電車、船舶等）、交通用信号機器、
   防災・防犯装置、各種安全装置等
   当社製品は、直接生命・身体に危害を及ぼす可能性のある機器・システム（生命維持装置、人体に埋め込み使用するもの等）、もしくは多大な物的損害を発生させるおそれのある機器・システム（原子力制御システム、軍事機器等）に使用されることを意図しておらず、使用することはできません。たとえ、意図しない用途に当社製品を使用したことによりお客様または第三者に損害が生じても、当社は一切その責任を負いません。なお、ご不明点がある場合は、当社営業にお問い合わせください。
6. 当社製品をご使用の際は、当社が指定する最大定格、動作電源電圧範囲、放熱特性、実装条件その他の保証範囲内でご使用ください。当社保証範囲を超えて当社製品をご使用された場合の故障および事故につきましては、当社は、一切その責任を負いません。
7. 当社は、当社製品の品質および信頼性の向上に努めていますが、半導体製品はある確率で故障が発生したり、使用条件によっては誤動作したりする場合があります。また、当社製品は耐放射線設計については行っておりません。当社製品の故障または誤動作が生じた場合も、人身事故、火災事故、社会的損害等を生じさせないよう、お客様の責任において、冗長設計、延焼対策設計、誤動作防止設計等の安全設計およびエージング処理等、お客様の機器・システムとしての出荷保証を行ってください。特に、マイコンソフトウェアは、単独での検証は困難なため、お客様の機器・システムとしての安全検証をお客様の責任で行ってください。
8. 当社製品の環境適合性等の詳細につきましては、製品個別に必ず当社営業窓口までお問合せください。ご使用に際しては、特定の物質の含有・使用を規制するRoHS指令等、適用される環境関連法令を十分調査のうえ、かかる法令に適合するようご使用ください。お客様がかかる法令を遵守しないことにより生じた損害に関して、当社は、一切その責任を負いません。
9. 本資料に記載されている当社製品および技術を国内外の法令および規則により製造・使用・販売を禁止されている機器・システムに使用することはできません。また、当社製品および技術を大量破壊兵器の開発等の目的、軍事利用の目的その他軍事用途に使用しないでください。当社製品または技術を輸出する場合は、「外国為替及び外国貿易法」その他輸出関連法令を遵守し、かかる法令の定めるところにより必要な手続を行ってください。
10. お客様の転売等により、本ご注意書き記載の諸条件に抵触して当社製品が使用され、その使用から損害が生じた場合、当社は何らの責任も負わず、お客様にてご負担して頂きますのでご了承ください。
11. 本資料の全部または一部を当社の文書による事前の承諾を得ることなく転載または複製することを禁じます。

注1. 本資料において使用されている「当社」とは、ルネサス エレクトロニクス株式会社およびルネサス エレクトロニクス株式会社がその総株主の議決権の過半数を直接または間接に保有する会社をいいます。
注2. 本資料において使用されている「当社製品」とは、注１において定義された当社の開発、製造製品をいいます。

ルネサス エレクトロニクス株式会社

■営業お問合せ窓口 http://www.renesas.com

※営業お問合せ窓口の住所は変更になることがあります。最新情報につきましては、弊社ホームページをご覧ください。

ルネサス エレクトロニクス株式会社　〒135-0061　東京都江東区豊洲3-2-24（豊洲フォレシア）

■技術的なお問合せおよび資料のご請求は下記へどうぞ。
総合お問合せ窓口：http://japan.renesas.com/contact/