

## RX ファミリ

### LPT モジュール Firmware Integration Technology

---

#### 要旨

本アプリケーションノートは、Firmware Integration Technology (FIT)を使用した LPT モジュールについて説明します。本モジュールはローパワータイマ(LPT)を使用して、ソフトウェアスタンバイモードの解除信号を発生させます。本モジュールとソフトウェアスタンバイモード、ELC を使用してソフトウェアスタンバイモードの周期的な解除を実現します。

以降、本モジュールを LPT FIT モジュールと称します。

#### 対象デバイス

この API が現在サポートしているデバイスのリストを以下に示します。

- RX113 グループ
- RX130 グループ
- RX230 グループ、RX231 グループ

本アプリケーションノートを他のマイコンへ適用する場合、そのマイコンの仕様にあわせて変更し、十分評価してください。

#### 関連ドキュメント

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。

- Firmware Integration Technology ユーザーズマニュアル(R01AN1833)
- ボードサポートパッケージモジュール Firmware Integration Technology (R01AN1685)
- e<sup>2</sup> studio に組み込む方法 Firmware Integration Technology (R01AN1723)
- CS+に組み込む方法 Firmware Integration Technology (R01AN1826)
- Renesas e<sup>2</sup> studio スマート・コンフィグレータ ユーザーガイド(R20AN0451)

## 目次

1. 概要 .....	3
1.1 LPT FIT モジュールとは .....	3
1.2 LPT FIT モジュールの概要 .....	3
1.3 API の概要 .....	3
1.4 処理例 .....	4
1.5 状態遷移図 .....	5
2. API 情報 .....	6
2.1 ハードウェアの要求 .....	6
2.2 ソフトウェアの要求 .....	6
2.3 サポートされているツールチェーン .....	6
2.4 使用する割り込みベクタ .....	6
2.5 ヘッダファイル .....	6
2.6 整数型 .....	6
2.7 コンパイル時の設定 .....	7
2.8 コードサイズ .....	7
2.9 引数 .....	8
2.10 戻り値 .....	8
2.11 FIT モジュールの追加方法 .....	8
2.12 for 文、while 文、do while 文について .....	9
3. API 関数 .....	10
R_LPT_Open () .....	10
R_LPT_Close () .....	12
R_LPT_Control () .....	14
R_LPT_GetVersion () .....	16
4. サンプルコード .....	17
5. デモプロジェクト .....	19
5.1 lpt_demo_rskrx231 .....	19
5.2 lpt_demo_rskrx113 .....	19
5.3 ワークスペースへのデモ追加 .....	20
5.4 デモのダウンロード方法 .....	20
6. 付録 .....	21
6.1 動作確認環境 .....	21
6.2 トラブルシューティング .....	23
改訂記録 .....	24

## 1. 概要

本モジュールは、RX ファミリの周辺機能である LPT をサポートし、ソフトウェアスタンバイモードの解除信号を発生させます。

本モジュールとソフトウェアスタンバイモード、および ELC を使用することによりソフトウェアスタンバイモードを一定周期ごとに解除する動作が実現できます。

### 1.1 LPT FIT モジュールとは

本モジュールは API として、プロジェクトに組み込んで使用します。本モジュールの組み込み方については、2.11 FIT モジュールの追加方法を参照してください。

### 1.2 LPT FIT モジュールの概要

本モジュールの R\_LPT\_Open 関数を呼び出すことで、LPT を動作させるために必要な設定を行います。

LPT 周期の設定は R\_LPT\_Open 関数を呼び出すことで実施されます。これにより、LPT を使用するために必要なレジスタ値を設定します。

なお、LPT 動作後に LPT 周期を変更する際は、R\_LPT\_Close 関数を呼び出した後、変更後の情報を引数に設定して R\_LPT\_Open 関数を呼び出してください。

LPT のカウントを開始する場合は、LPT\_CMD\_START コマンドで R\_LPT\_Control 関数を呼び出してください。

LPT のカウントを停止する場合は、LPT\_CMD\_STOP コマンドで R\_LPT\_Control 関数を呼び出してください。

LPT のカウントをリセットする場合は、LPT\_CMD\_COUNT\_RESET コマンドで R\_LPT\_Control 関数を呼び出してください。

コンペアマッチ 0 の発生によりソフトウェアスタンバイモードから ELC 動作可能状態への遷移が行われます。

本モジュールを使用する場合は、本モジュール外で LPT の制御を行わないでください。

### 1.3 API の概要

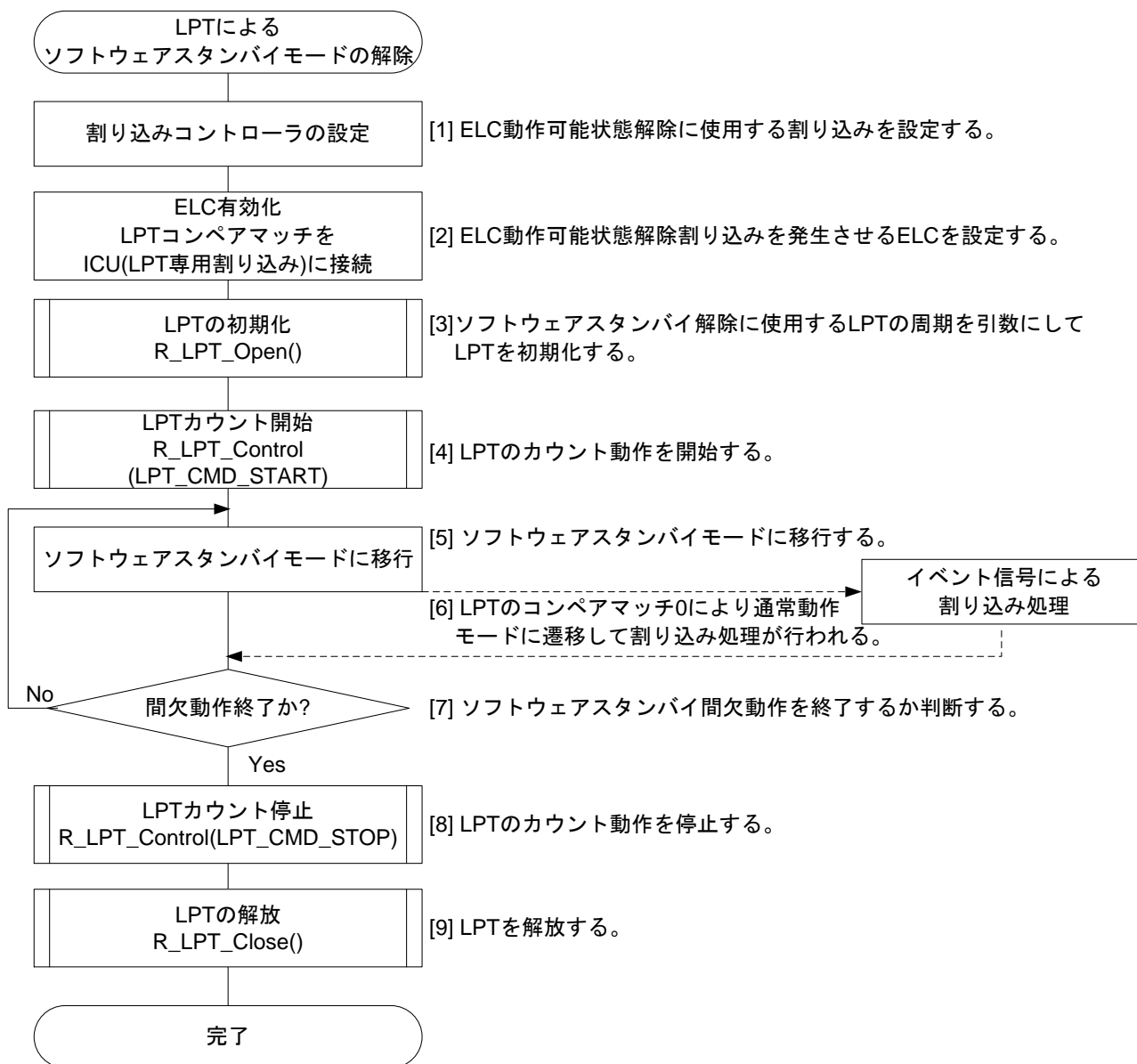
表 1.1 に本モジュールに含まれる API 関数を示します。

表 1.1 API 関数一覧

関数	関数説明
R_LPT_Open	LPT の初期設定を行い、本モジュールを使用できる状態にします。
R_LPT_Close	LPT モジュールを解放します。
R_LPT_Control	LPT のカウント開始、停止、リセット制御を行います。
R_LPT_GetVersion	本モジュールのバージョン番号を返します。

## 1.4 処理例

図 1.1 に本モジュールの処理例を示します。



## 1.5 状態遷移図

図 1.2 に本モジュールの状態遷移図を示します。

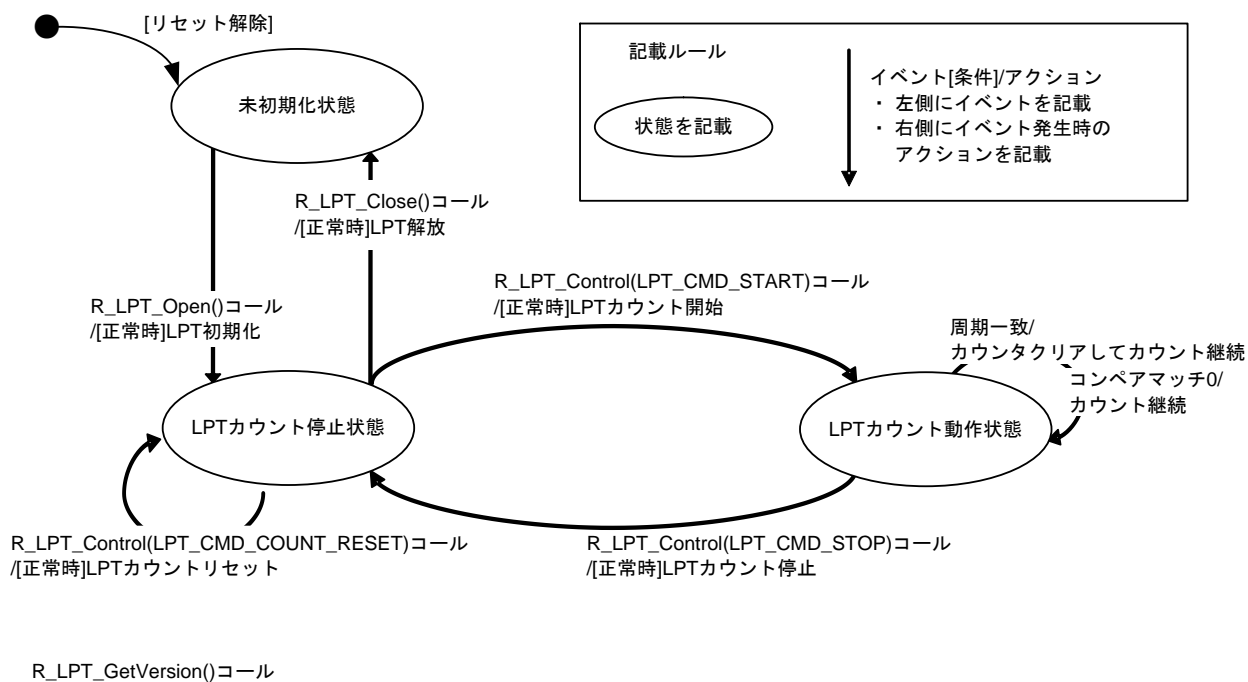


図 1.2 LPT FIT モジュールの状態遷移図

---

## 2. API 情報

本 FIT モジュールは、下記の条件で動作を確認しています。

---

### 2.1 ハードウェアの要求

ご使用になる MCU が以下の機能をサポートしている必要があります。

- ローパワータイマ(LPT)
- イベントリンクコントローラ(ELC)
- 消費電力低減機能(LPC)

---

### 2.2 ソフトウェアの要求

このドライバは以下の FIT モジュールに依存しています。

- ルネサスボードサポートパッケージ (r\_bsp)

---

### 2.3 サポートされているツールチェーン

本 FIT モジュールは「6.1 動作確認環境」に示すツールチェーンで動作確認を行っています。

---

### 2.4 使用する割り込みベクタ

割り込みベクタは使用しません。

---

### 2.5 ヘッダファイル

すべての API 呼び出しとそれをサポートするインタフェース定義は r\_lpt\_rx\_if.h に記載しています。

---

### 2.6 整数型

このプロジェクトは ANSI C99 を使用しています。これらの型は stdint.h で定義されています。

## 2.7 コンパイル時の設定

本モジュールのコンフィギュレーションオプションの設定は、`r_lpt_rx_config.h`で行います。  
オプション名および設定値に関する説明を、下表に示します。

Configuration options in <code>r_lpt_rx_config.h</code>	
<code>#define LPT_CFG_PARAM_CHECKING (1)</code>	各関数の最初にあるパラメータチェック処理(引数の内容を確認する処理)をコード出力するか、しないかを選択します。 “0”の場合、パラメータチェック処理のコードを生成しません。 “1”の場合、パラメータチェック処理のコードを生成し、実行します。 ただし、 <code>r_bsp_config.h</code> ファイルで定義される “BSP_CFG_PARAM_CHECKING_ENABLE”の値が“0”の場合は LPT_CFG_PARAM_CHECKING の値にかかわらずパラメータチェック処理のコードを生成しません。
<code>#define LPT_CFG_LPT_CLOCK_SOURCE (0)</code> ※デフォルト値は <code>r_bsp_config.h</code> ファイルで定義される “BSP_CFG_LPT_CLOCK_SOURCE”の値となります。	ローパワータイマのクロックソースを選択します。 “0”の場合、サブクロック発振器を選択します。 “1”の場合、IWDG 専用オンチップオシレータを選択します。

## 2.8 コードサイズ

本モジュールの ROM サイズ、RAM サイズを下表に示します。

ROM (コードおよび定数) と RAM (グローバルデータ) のサイズは、ビルド時の「2.7 コンパイル時の設定」のコンフィギュレーションオプションによって決まります。

下表の値は下記条件で確認しています。

モジュールリビジョン: `r_lpt_rx rev1.23`

コンパイラバージョン: Renesas Electronics C/C++ Compiler Package for RX Family V3.01.00

(統合開発環境のデフォルト設定に“-lang = c99”オプションを追加)

コンフィギュレーションオプション: デフォルト設定

ROM、RAM およびスタックのコードサイズ				
デバイス	分類	使用メモリ		備考
		パラメータチェック処理あり	パラメータチェック処理なし	
RX113 RX130 RX230 RX231	ROM	480 バイト	469 バイト	
	RAM	0 バイト	0 バイト	
	スタック	60 バイト	60 バイト	

## 2.9 引数

API 関数の引数である列挙型を示します。この列挙型は、API 関数のプロトタイプ宣言とともに `r_lpt_rx_if.h` に記載されています。

```
typedef enum e_lpt_cmd
{
    LPT_CMD_START,          /* LPT カウント開始 */
    LPT_CMD_STOP,           /* LPT カウント停止 */
    LPT_CMD_COUNT_RESET     /* LPT カウントリセット */
} lpt_cmd_t;
```

## 2.10 戻り値

API 関数の戻り値を示します。この列挙型は、API 関数のプロトタイプ宣言とともに `r_lpt_rx_if.h` で記載されています。

```
typedef enum /* LPT API のステータスコード */
{
    LPT_SUCCESS,            /* 正常処理完了 */
    LPT_ERR_LOCK_FUNC,      /* 動作中。既に使用されている */
    LPT_ERR_INVALID_ARG,    /* 構造体の要素に無効な値が含まれている */
    LPT_ERR_CONDITION_NOT_MET /* 実行条件を満たしていない */
} lpt_err_t;
```

## 2.11 FIT モジュールの追加方法

本モジュールは、使用するプロジェクトごとに追加する必要があります。ルネサスでは、Smart Configurator を使用した(1)、(3)の追加方法を推奨しています。ただし、Smart Configurator は、一部の RX デバイスのみサポートしています。サポートされていない RX デバイスについては(2)、(4)の方法を使用してください。

- (1) e<sup>2</sup> studio 上で Smart Configurator を使用して FIT モジュールを追加する場合  
e<sup>2</sup> studio の Smart Configurator を使用して、自動的にユーザプロジェクトに FIT モジュールを追加します。詳細は、アプリケーションノート「Renesas e<sup>2</sup> studio スマート・コンフィグレータ ユーザーガイド (R20AN0451)」を参照してください。
- (2) e<sup>2</sup> studio 上で FIT Configurator を使用して FIT モジュールを追加する場合  
e<sup>2</sup> studio の FIT Configurator を使用して、自動的にユーザプロジェクトに FIT モジュールを追加することができます。詳細は、アプリケーションノート「RX ファミリ e<sup>2</sup> studio に組み込む方法 Firmware Integration Technology (R01AN1723)」を参照してください。
- (3) CS+上で Smart Configurator を使用して FIT モジュールを追加する場合  
CS+上で、スタンドアロン版 Smart Configurator を使用して、自動的にユーザプロジェクトに FIT モジュールを追加します。詳細は、アプリケーションノート「Renesas e<sup>2</sup> studio スマート・コンフィグレータ ユーザーガイド (R20AN0451)」を参照してください。
- (4) CS+上で FIT モジュールを追加する場合  
CS+上で、手動でユーザプロジェクトに FIT モジュールを追加します。詳細は、アプリケーションノート「RX ファミリ CS+に組み込む方法 Firmware Integration Technology (R01AN1826)」を参照してください。



## 2.12 for 文、while 文、do while 文について

本モジュールでは、レジスタの反映待ち処理等で for 文、while 文、do while 文（ループ処理）を使用しています。これらループ処理には、「WAIT\_LOOP」をキーワードとしたコメントを記述しています。そのため、ループ処理にユーザがフェイルセーフの処理を組み込む場合は、「WAIT\_LOOP」で該当の処理を検索できます。

「WAIT\_LOOP」を記述している対象デバイス

- RX113 グループ
- RX130 グループ
- RX230、RX231 グループ

以下に記述例を示します。

```
while 文の例 :
/* WAIT LOOP */
while(0 == SYSTEM.OSCOVFSR.BIT.PLOVF)
{
    /* The delay period needed is to make sure that the PLL has stabilized. */
}

for 文の例 :
/* Initialize reference counters to 0. */
/* WAIT LOOP */
for (i = 0; i < BSP_REG_PROTECT_TOTAL_ITEMS; i++)
{
    g_protect_counters[i] = 0;
}

do while 文の例 :
/* Reset completion waiting */
do
{
    reg = phy_read(ether_channel, PHY_REG_CONTROL);
    count++;
} while ((reg & PHY_CONTROL_RESET) && (count < ETHER_CFG_PHY_DELAY_RESET)); /* WAIT_LOOP */
```

### 3. API 関数

#### R\_LPT\_Open ()

この関数は LPT FIT モジュールを初期化する関数です。この関数は他の API 関数を使用する前に実行される必要があります。

##### Format

```
lpt_err_t R_LPT_Open (  
    uint32_t const period  
)
```

##### Parameters

uint32\_t const period  
LPT の周期(μs 単位で指定)

##### Return Values

LPT_SUCCESS	/* 正常処理完了 */
LPT_ERR_LOCK_FUNC	/* 動作中。既に使用されている */
LPT_ERR_INVALID_ARG	/* 引数に無効な値が含まれている */

##### Properties

r\_lpt\_rx\_if.h にプロトタイプ宣言されています。

##### Description

LPT 動作を開始するため、次に示す初期設定を行います。引数で指定した周期を設定します。

- ・ローパワータイマによるスタンバイ復帰を許可
- ・ローパワータイマクロックソース及び分周比設定
- ・ローパワータイマコンペアマッチ 0 許可
- ・ローパワータイマ周期設定
- ・ローパワータイマコンペアマッチ値 0 設定
- ・ローパワータイマクロック供給
- ・ローパワータイマリセット実施

##### Reentrant

この関数は、再入することはできません。

**Example**

```
void main(void)
{
    lpt_err_t err;
    uint32_t period;

    period = 100000;
    err = R_LPT_Open(period);
    if (LPT_SUCCESS != err)
    {
        while(1) { };
    }

    :
    :
}
```

**Special Notes:**

ローパワータイマのクロックソースの発振が安定している状態で本関数を呼び出してください。

ローパワータイマのクロックソースにサブクロック発振器を選択している場合、LPT の周期は 153～64000488 の値を設定してください。

ローパワータイマクロックソースに IWDTC 専用オンチップオシレータを選択している場合、LPT の周期は 334～139811199 の値を設定してください。

ローパワータイマクロックソースに IWDTC 専用オンチップオシレータを選択している場合、IWDTC オートスタートモード動作時は OFS0.IWDTCSTP ビットに “0” (カウント停止無効)を、それ以外の時は IWDTCSTPR.SLCSTP ビットに “0” (カウント停止無効)を設定してください。

ソフトウェアスタンバイモードを解除した場合、発振安定待ちのためメインクロック発振器ウェイトコントロールレジスタ(MOSCWTCR)で選択した時間待たされるので、プログラムが実行可能になるのはソフトウェアスタンバイモード解除後、発振安定待機時間が経過した後となります。

この関数は、LPT のカウントを停止し、終了処理を実行する関数です。

```
lpt_err_t R_LPT_Close (
    void
)
```

## なし

```
LPT_SUCCESS /* 正常处理完了 */
```

r\_lpt\_rx\_if.h にプロトタイプ宣言されています。

LPT 動作を終了するため、次に示す設定を行います。

- ・ローパワータイマ停止
- ・ローパワータイマクロックが供給されている場合はローパワータイマリセット実施
- ・ローパワータイマクロック停止
- ・ローパワータイマコンペアマッチ値 0 初期化
- ・ローパワータイマ周期初期化
- ・ローパワータイマコントロールレジスタ 1 初期化
- ・ローパワータイマによるスタンバイ復帰を禁止

この関数は、再入することはできません。

**Example**

```
void main(void)
{
    lpt_err_t err;
    uint32_t period;

    period = 100000;
    err = R_LPT_Open(period);
    if (LPT_SUCCESS != err)
    {
        while(1) { };
    }

    :
    :

    err = R_LPT_Close();
    if (LPT_SUCCESS != err)
    {
        while(1) { };
    }

    :
    :

}
```

**Special Notes:**

R\_LPT\_Open 関数により LPT 設定を行った後、ローパワータイマのクロックソースで 1 サイクル以上経過してから本関数を呼び出してください。

---

## R\_LPT\_Control ()

---

この関数は、LPT のカウント開始/停止/リセットするための関数です。

### Format

```
lpt_err_t R_LPT_Control (  
    lpt_cmd_t const cmd      /* コマンド */  
)
```

### Parameters

lpt\_cmd\_t const cmd  
実行するコマンド(「2.9 引数」を参照)

### Return Values

LPT\_SUCCESS                               /\* 正常処理完了 \*/  
LPT\_ERR\_INVALID\_ARG                       /\* 構造体の要素に無効な値が含まれている \*/  
LPT\_ERR\_CONDITION\_NOT\_MET               /\* 実行条件を満たしていない \*/

### Properties

r\_lpt\_rx\_if.h にプロトタイプ宣言されています。

### Description

この API 関数は、ローパワータイマカウント動作/停止/リセット制御を行います。

カウンタリセットコマンド(LPT\_CMD\_COUNT\_RESET)を実行する場合は、LPT のカウントが停止している状態で行ってください。LPT カウント動作中にカウンタリセットコマンドを実行すると、LPT\_ERR\_CONDITION\_NOT\_MET を返します。

### Reentrant

この関数は、再入することはできません。

**Example**

```
void main(void)
{
    lpt_err_t err;
    uint32_t period;

    period = 100000;
    err = R_LPT_Open(period);
    if (LPT_SUCCESS != err)
    {
        while(1) { };
    }

    :
    :

    err = R_LPT_Control(LPT_CMD_START);
    if (LPT_SUCCESS != err)
    {
        while(1) { };
    }

    :
    :

}
```

**Special Notes:**

R\_LPT\_Open 関数により LPT 設定を行った状態で、本関数を呼び出してください。

---

## R\_LPT\_GetVersion ()

---

この関数は本モジュールのバージョン番号を返します。

### Format

```
uint32_t R_LPT_GetVersion (  
    void  
)
```

### Parameters

なし

### Return Values

バージョン番号

### Properties

r\_lpt\_rx\_if.h にプロトタイプ宣言されています。

### Description

この関数はこのモジュールのバージョン番号を返します。バージョン番号は符号化され、最上位の 2 バイトがメジャーバージョン番号を、最下位の 2 バイトがマイナーバージョン番号を示しています。

### Reentrant

この関数は、再入可能な関数です。

### Example

```
void main(void)  
{  
    uint32_t version;  
    :  
    version = R_LPT_GetVersion();  
    while(1) { };  
}
```

### Special Notes:

なし



#### 4. サンプルコード

LPT を使用してソフトウェアスタンバイモードを一定周期ごとに解除する場合のサンプルコードを示します。

次の(1)～(14)の順に動作します。

- (1)ソフトウェアスタンバイモードを使用可能にするため、R\_LPC\_LowPowerModeConfigure 関数を呼び出す。
- (2)ELC のモジュールストップ解除を行う。
- (3)マスカブル割り込み禁止に設定する。
- (4)ELSR19I 割り込みの伝達先を CPU に設定するため、ELSR19I 割り込みの DTC 起動禁止設定を行う。
- (5)ELSR19I 割り込みを禁止する。
- (6)ELSR19I 割り込みの割り込み優先レベルをプロセッサ割り込み優先レベル(IPL)より高い値に設定する。
- (7)ELSR19I 割り込みを許可する。
- (8)ELC 機能を有効に設定する。
- (9)ELC の LPT コンペアマッチを ICU(LPT 専用割り込み)に設定する。
- (10)LPT を LPT FIT モジュールで使用可能にするため、R\_LPT\_Open 関数を呼び出す。
- (11)LPT のカウントを開始するため、R\_LPT\_Control 関数を呼び出す。
- (12)ソフトウェアスタンバイモードに移行するため、R\_LPC\_LowPowerModeActivate 関数を呼び出す。
- (13)LPT のコンペアマッチ 0 が発生すると通常動作モードに遷移して、ELSR19I 割り込み例外処理が呼び出されます。
- (14)ソフトウェアスタンバイモード解除時のユーザ処理を実行後、(12)に戻ります。

- ・本サンプルは RX113 を対象としています。
- ・本サンプルは LPC FIT モジュールを使用しています。LPC FIT モジュールについては、ルネサスエレクトロニクスのホームページを参照してください。

```

#include "platform.h"
#include "r_lpc_rx100_if.h"
#include "r_lpt_rx_if.h"

void main(void);

void main(void)
{
    lpt_err_t lpt_err;
    uint32_t period;
    lpc_err_t lpc_err;

    /* ----- スリープモード時カウント停止無効 ----- */
    IWDTCSTPR.BIT.SLCSSTP = 0;

    /* ----- ソフトウェアスタンバイモード設定 ----- */
    lpc_err = R_LPC_LowPowerModeConfigure(LPC_LP_SW_STANDBY);
    if (LPC_SUCCESS != lpc_err)
    {
        while(1) { };
    }

    /* ELC モジュールストップ解除 */
    R_BSP_RegisterProtectDisable(BSP_REG_PROTECT_LPC_CGC_SWR);
    MSTP(ELC) = 0;
    R_BSP_RegisterProtectEnable(BSP_REG_PROTECT_LPC_CGC_SWR);

    /* ----- マスカブル割り込み禁止 ----- */
    R_BSP_InterruptsDisable();

    /* ----- ELSR19I 割り込み設定 ----- */
    ICU.DTCER[80].BIT.DTCE = 0;
    ICU.IER[0x0A].BIT.IEN0 = 0;
    ICU.IPR[80].BIT.IPR = 15;
    ICU.IER[0x0A].BIT.IEN0 = 1;
    while ( 1 != ICU.IER[0x0A].BIT.IEN0 )
    {
        /* 書き込み値が反映されていることを確認する */
    }

    /* ----- ELC 設定 ----- */
    ELC.ELCR.BIT.ELCON = 1; /* ELC 機能有効 */
    ELC.ELSR[19].BIT.ELS = 0x5D; /* ICU (LPT 専用割り込み) は LPT コンペアマッチ (5Dh) に設定 */

    /* ----- LPT 初期化 ----- */
    period = 100000; /* LPT 周期 = 100000[μs] */
    lpt_err = R_LPT_Open(period);
    if (LPT_SUCCESS != lpt_err)
    {
        while(1) { };
    }

    /* ----- LPT カウント開始 ----- */
    lpt_err = R_LPT_Control(LPT_CMD_START);
    if (LPT_SUCCESS != lpt_err)
    {
        while(1) { };
    }

    while(1)
    {
        /* ----- ソフトウェアスタンバイモードに移行 ----- */
        lpc_err = R_LPC_LowPowerModeActivate(NULL);
        if (LPC_SUCCESS != lpc_err)
        {
            while(1) { };
        }
        /* ----- ソフトウェアスタンバイモード解除時のユーザ処理をここに記述する ----- */
    }
}

```

LPT クロックソースとして IWDTC 専用  
 オンチップオシレータを使用する場合は  
 ソフトウェアスタンバイ中に  
 クロック停止させない設定が必要

ELSR19I 割り込みを使用して  
 ELC 動作可能状態から  
 通常動作モードに遷移させる

ソフトウェアスタンバイ状態から  
 ELC 動作可能状態への遷移完了をトリガ  
 として FI SR19I 割り込みを発生させる

図 4.1 LPT を使用してソフトウェアスタンバイモードを一定周期ごとに解除する例

## 5. デモプロジェクト

デモプロジェクトはスタンドアロンプログラムです。デモプロジェクトには、FIT モジュールとそのモジュールが依存するモジュール（例：r\_bsp）を使用する main()関数が含まれます。本 FIT モジュールには以下のデモプロジェクトが含まれます。

### 5.1 lpt\_demo\_rskrx231

このセクションでは、LPT を使用してソフトウェアスタンバイモードを定期的に終了する（一定間隔動作、間欠動作）サンプルコードについて説明します。

このデモを実行するには、以下のステップを使用します。

- (1) ソフトウェアスタンバイモードに合わせてLPCモジュールを構成します
- (2) ソフトウェアスタンバイモードでカウントを停止しないようにIWDTCロックを構成します
- (3) ELCのモジュール停止状態をキャンセルします
- (4) DTCの有効化動作を無効にします。この結果、割り込み要求はCPU宛に直接送信されるようになります
- (5) ELCモジュールの低消費電力タイマのカウントダウン一致イベントを構成します
- (6) ELCとLPTの動作を開始します
- (7) メインループ：ソフトウェアスタンバイモードに移行して3秒間そのモードにとどまり、その後、通常動作モードに戻ります

#### 注記：

- このデモプログラムは、LPC FITモジュールを使用します。LPC FITモジュールについては、ルネサス エレクトロニクスWebサイトを参照してください
- e<sup>2</sup> studioを使用してデバッグを行う際に、プログラムがソフトウェアスタンバイモードと通常モードのどちらにあるかを判断するには、e<sup>2</sup> studioウィンドウの左下隅にあるプログラム動作状態を参照することができます。
  - 「Standby」：プログラムはソフトウェアスタンバイモードです。
  - 「Running」：プログラムは通常モードです。

#### 確認方法:

- プログラムが、ソフトウェアスタンバイモードに移行したことを通知するメッセージをコンソールに出力するようになります。
- プログラムはソフトウェアスタンバイモードに移行し、3 秒間そのモードにとどまります。モジュールの停止状態を設定していない場合、このモードの間、CPU とすべてのペリフェラルの機能は停止します。
- 3 秒後に、プログラムは通常モードに戻り、通知メッセージをコンソールに出力して、LED 0 を 1 秒間点滅させます。

### 5.2 lpt\_demo\_rskrx113

lpt\_demo\_rskrx113 プログラムは、lpt\_demo\_rskrx231 と同じものです。

**特別な注記：**（RX113 BSP resetprg.c 内の）clock\_source\_select()で、LPT クロックソースを有効にする必要があります。

---

### 5.3 ワークスペースへのデモ追加

---

デモプロジェクトは、本アプリケーションノートで提供されるファイルの FITDemos サブディレクトリにあります。ワークスペースにデモプロジェクトを追加するには、「ファイル」>>「インポート」を選択し、「インポート」ダイアログから「一般」の「既存プロジェクトをワークスペースへ」を選択して「次へ」ボタンをクリックします。「インポート」ダイアログで「アーカイブ・ファイルの選択」ラジオボタンを選択し、「参照」ボタンをクリックして FITDemos サブディレクトリを開き、使用するデモの zip ファイルを選択して「終了」をクリックします。

---

### 5.4 デモのダウンロード方法

---

デモプロジェクトは、RX Driver Package には同梱されていません。デモプロジェクトを使用する場合は、個別に各 FIT モジュールをダウンロードする必要があります。「スマートブラウザ」の「アプリケーションノート」タブから、本アプリケーションノートを右クリックして「サンプル・コード（ダウンロード）」を選択することにより、ダウンロードできます。

## 6. 付録

## 6.1 動作確認環境

このセクションでは、LPT FIT モジュールの動作確認用の環境について説明します。

表 6.1 動作確認環境 (Rev.1.23)

項目	内容
統合開発環境	ルネサスエレクトロニクス製 e <sup>2</sup> studio V7.3.0
C コンパイラ	ルネサスエレクトロニクス製 C/C++ Compiler for RX Family V3.01.00 コンパイルオプション：統合開発環境のデフォルト設定に以下のオプションを追加 -lang = c99
エンディアン	ビッグエンディアン/リトルエンディアン
モジュールのリビジョン	Rev1.23

表 6.2 動作確認環境 (Rev.1.22)

項目	内容
統合開発環境	ルネサスエレクトロニクス製 e <sup>2</sup> studio V7.3.0
C コンパイラ	ルネサスエレクトロニクス製 C/C++ Compiler for RX Family V3.01.00 コンパイルオプション：統合開発環境のデフォルト設定に以下のオプションを追加 -lang = c99
エンディアン	ビッグエンディアン/リトルエンディアン
モジュールのリビジョン	Rev1.22
使用ボード	Renesas Starter Kit+ for RX113 (型名: R0K505113SxxxBE)

表 6.3 動作確認環境 (Rev.1.21)

項目	内容
統合開発環境	ルネサスエレクトロニクス製 e <sup>2</sup> studio V6.0.0
C コンパイラ	ルネサスエレクトロニクス製 C/C++ Compiler for RX Family V2.07.00 コンパイルオプション：統合開発環境のデフォルト設定に以下のオプションを追加 -lang = c99
エンディアン	ビッグエンディアン/リトルエンディアン
モジュールのリビジョン	Rev1.21
使用ボード	Renesas Starter Kit+ for RX130-512KB (型名 : RTK5051308SxxxxxBE)

表 6.4 動作確認環境 (Rev.1.20)

項目	内容
統合開発環境	ルネサスエレクトロニクス製 e <sup>2</sup> studio V6.0.0.001
C コンパイラ	ルネサスエレクトロニクス製 C/C++ Compiler for RX Family V2.07.00 コンパイルオプション：統合開発環境のデフォルト設定に以下のオプションを追加 -lang = c99
エンディアン	ビッグエンディアン/リトルエンディアン
モジュールのリビジョン	Rev1.20
使用ボード	Renesas Starter Kit+ for RX130-512KB (型名 : RTK5051308SxxxxxBE)

表 6.5 動作確認環境 (Rev.1.11)

項目	内容
統合開発環境	ルネサスエレクトロニクス製 e <sup>2</sup> studio V5.0.1.005
C コンパイラ	ルネサスエレクトロニクス製 C/C++ Compiler for RX Family V2.05.00 コンパイルオプション: 統合開発環境のデフォルト設定に以下のオプションを追加 -lang = c99
エンディアン	ビッグエンディアン/リトルエンディアン
モジュールのバージョン	Rev1.11
使用ボード	Renesas Starter Kit for RX113 (型名: R0K505113SxxxBE) Renesas Starter Kit for RX231 (型名: R0K505231SxxxBE) Renesas Starter Kit for RX130 (型名: RTK5005130SxxxxxBE)

表 6.6 動作確認環境 (Rev.1.10)

項目	内容
統合開発環境	ルネサスエレクトロニクス製 e2 studio V5.0.0.043
C コンパイラ	ルネサスエレクトロニクス製 C/C++ Compiler for RX Family V2.04.01 コンパイルオプション: 統合開発環境のデフォルト設定に以下のオプションを追加 -lang = c99
エンディアン	ビッグエンディアン/リトルエンディアン
モジュールのバージョン	Rev1.10
使用ボード	Renesas Starter Kit+ for RX113 (型名: R0K505113SxxxBE) Renesas Starter Kit+ for RX231 (型名: R0K505231SxxxBE) Renesas Starter Kit+ for RX130 (型名: RTK5005130SxxxxxBE)

表 6.7 動作確認環境 (Rev.1.00)

項目	内容
統合開発環境	ルネサスエレクトロニクス製 e <sup>2</sup> studio V4.2.0.012
C コンパイラ	ルネサスエレクトロニクス製 C/C++ Compiler for RX Family V2.04.01 コンパイルオプション: 統合開発環境のデフォルト設定に以下のオプションを追加 -lang = c99
エンディアン	ビッグエンディアン/リトルエンディアン
モジュールのバージョン	Rev1.00
使用ボード	Renesas Starter Kit+ for RX113 (型名: R0K505113SxxxBE)

## 6.2 トラブルシューティング

- (1) Q : プロジェクトに FIT モジュールを追加し、プロジェクトをビルドしました。次のエラーが発生しました。「ソースファイル「platform.h」を開くことができません。」

A : FIT モジュールがプロジェクトに対して正しく追加されていない可能性があります。次の文書で、FIT モジュールを追加した方法が正しいかどうか確認してください。

- CS+を使用している場合 :  
アプリケーションノート『RX ファミリ CS+ に組み込む方法 Firmware Integration Technology (R01AN1826)』
- e<sup>2</sup> studio を使用している場合 :  
アプリケーションノート『RX ファミリ e<sup>2</sup> studio に組み込む方法 Firmware Integration Technology (R01AN1723)』

FIT モジュールを追加する場合、ボードサポートパッケージの FIT モジュール (BSP モジュール) もプロジェクトに追加する必要があります。『ボードサポートパッケージモジュール (R01AN1685)』を参照してください。

- (2) Q : プロジェクトに FIT モジュールを追加し、プロジェクトをビルドしました。次のエラーが発生しました。「現在の r\_lpt\_rx モジュールはこの MCU をサポートしていません。」

A : 追加した FIT モジュールは、現在のプロジェクトで選択されている対象デバイスをサポートしていない可能性があります。追加した FIT モジュールがサポートしているデバイスを確認してください。

- (3) Q : プロジェクトに FIT モジュールを追加し、プロジェクトをビルドしました。構成の設定が誤っている場合に対応するエラーが発生しました。

A : 「r\_lpt\_rx\_config.h」ファイル内の設定が誤っている可能性があります。「r\_lpt\_rx\_config.h」ファイルを確認してください。誤った設定が存在している場合、その設定にとって正しい値を設定してください。詳細については、「2.7 コンパイル時の設定」を参照してください。

## 改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2016.3.1	—	初版発行
1.10	2016.7.1	all	RX130、RX230、RX231 に対応
		all	IWDT 専用オンチップオシレータを選択した場合の LPT の周期の精度を改善
		10	周期の精度改善に伴い、ローパワータイマクロックソースに IWDT 専用オンチップオシレータを選択した場合の LPT の周期の範囲を変更
		10	Special Notes に LPT の周期に設定した時間についての注意事項を追加
1.11	2016.10.1	all	R_LPT_Control 関数のコマンドに LPT カウンタリセット (LPT_CMD_COUNT_RESET) を追加
		8,14	戻り値に「LPT_ERR_CONDITION_NOT_MET」を追加
		プログラム	R_LPT_Control 関数のコマンドに LPT カウンタリセット (LPT_CMD_COUNT_RESET) を追加
		プログラム	戻り値に「LPT_ERR_CONDITION_NOT_MET」を追加
1.20	2017.10.1	all	RX130-512KB に対応
		1	関連アプリケーションノートに以下のアプリケーションノートを追加: Renesas e <sup>2</sup> studio スマート・コンフィグレータ ユーザーガイド(R20AN0451)
		3,7	1.2 章の「必要メモリサイズ」を 2.7 章に移動
		8	2.14 FIT モジュールの追加方法: 変更
		18,19	5.1 動作確認環境追加
		19	5.2 トラブルシューティング追加
1.21	2017.10.31	18	「5. デモプロジェクト」を追加
		20	「6.1 動作確認環境」: Rev.1.21 に対応する表を追加
		22	「6.2 トラブルシューティング」: 質問を 2 つ追加
1.22	2018.11.16	—	XML 内にドキュメント番号を追加。
		8	「2.11 FIT モジュールをプロジェクトに追加する方法」を更新
		9	「2.12 for 文、while 文、do while 文について」を追加
		21	Rev.1.22 に対応する表を追加
1.23	2019.4.1	—	機能関連 Smart Configurator での GUI によるコンフィグオプション設定機能に対応 ■内容 GUI によるコンフィグオプション設定機能に対応するため、設定ファイルを追加。
		3	「1.3 API の概要」を移動
		6	「2.4 使用する割り込みベクタ」を追加
		7	「2.8 コードサイズ」を変更
		8	「2.9 引数」を変更 「コールバック関数」を削除



Rev.	発行日	改訂内容	
		ページ	ポイント
1.23	2019.4.1	9	「2.12 for 文、while 文、do while 文について」を変更
		16	「R_LPT_GetVersion()」を変更
		21	「6.1 動作確認環境」 Rev.1.23 に対応する表を追加

## 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イやマガジンケース、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、入力信号や入出力ブルアップ電源を入れないでください。入力信号や入出力ブ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っていると、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。お客様の機器・システムの設計において、回路、ソフトウェアおよびこれらに関連する情報を使用する場合には、お客様の責任において行ってください。これらの使用に起因して生じた損害（お客様または第三者いずれに生じた損害も含まれます。以下同じです。）に関し、当社は、一切その責任を負いません。
  2. 当社製品、本資料に記載された製品データ、図、表、プログラム、アルゴリズム、応用回路例等の情報の使用に起因して発生した第三者の特許権、著作権その他の知的財産権に対する侵害またはこれらに関する紛争について、当社は、何らの保証を行うものではなく、また責任を負うものではありません。
  3. 当社は、本資料に基づき当社または第三者の特許権、著作権その他の知的財産権を何ら許諾するものではありません。
  4. 当社製品を、全部または一部を問わず、改造、改変、複製、リバースエンジニアリング、その他、不適切に使用しないでください。かかる改造、改変、複製、リバースエンジニアリング等により生じた損害に関し、当社は、一切その責任を負いません。
  5. 当社は、当社製品の品質水準を「標準水準」および「高品質水準」に分類しており、各品質水準は、以下に示す用途に製品が使用されることを意図しております。  
標準水準： コンピュータ、OA 機器、通信機器、計測機器、AV 機器、家電、工作機械、パーソナル機器、産業用ロボット等  
高品質水準： 輸送機器（自動車、電車、船舶等）、交通制御（信号）、大規模通信機器、金融端末基幹システム、各種安全制御装置等  
当社製品は、データシート等により高信頼性、Harsh environment 向け製品と定義しているものを除き、直接生命・身体に危害を及ぼす可能性のある機器・システム（生命維持装置、人体に埋め込み使用するもの等）、もしくは多大な物的損害を発生させるおそれのある機器・システム（宇宙機器と、海底中継器、原子力制御システム、航空機制御システム、プラント基幹システム、軍事機器等）に使用されることを意図しておらず、これらの用途に使用することは想定していません。たとえ、当社が想定していない用途に当社製品を使用したことにより損害が生じても、当社は一切その責任を負いません。
  6. 当社製品をご使用の際は、最新の製品情報（データシート、ユーザーズマニュアル、アプリケーションノート、信頼性ハンドブックに記載の「半導体デバイスの使用上の一般的な注意事項」等）をご確認の上、当社が指定する最大定格、動作電源電圧範囲、放熱特性、実装条件その他指定条件の範囲内でご使用ください。指定条件の範囲を超えて当社製品をご使用された場合の故障、誤動作の不具合および事故につきましては、当社は、一切その責任を負いません。
  7. 当社は、当社製品の品質および信頼性の向上に努めていますが、半導体製品はある確率で故障が発生したり、使用条件によっては誤動作したりする場合があります。また、当社製品は、データシート等において高信頼性、Harsh environment 向け製品と定義しているものを除き、耐放射線設計を行っておりません。仮に当社製品の故障または誤動作が生じた場合であっても、人身事故、火災事故その他社会的損害等を生じさせないよう、お客様の責任において、冗長設計、延焼対策設計、誤動作防止設計等の安全設計およびエージング処理等、お客様の機器・システムとしての出荷保証を行ってください。特に、マイコンソフトウェアは、単独での検証は困難なため、お客様の機器・システムとしての安全検証をお客様の責任で行ってください。
  8. 当社製品の環境適合性等の詳細につきましては、製品個別に必ず当社営業窓口までお問合せください。ご使用に際しては、特定の物質の含有・使用を規制する RoHS 指令等、適用される環境関連法令を十分調査のうえ、かかる法令に適合するようご使用ください。かかる法令を遵守しないことにより生じた損害に関し、当社は、一切その責任を負いません。
  9. 当社製品および技術を国内外の法令および規則により製造・使用・販売を禁止されている機器・システムに使用することはできません。当社製品および技術を輸出、販売または移転等する場合は、「外国為替及び外国貿易法」その他日本国および適用される外国の輸出管理関連法規を遵守し、それらの定めるところに従い必要な手続きを行ってください。
  10. お客様が当社製品を第三者に転売等される場合には、事前に当該第三者に対して、本ご注意書き記載の諸条件を通知する責任を負うものといたします。
  11. 本資料の全部または一部を当社の文書による事前の承諾を得ることなく転載または複製することを禁じます。
  12. 本資料に記載されている内容または当社製品についてご不明な点がございましたら、当社の営業担当者までお問合せください。
- 注 1. 本資料において使用されている「当社」とは、ルネサス エレクトロニクス株式会社およびルネサス エレクトロニクス株式会社が直接的、間接的に支配する会社をいいます。
- 注 2. 本資料において使用されている「当社製品」とは、注 1 において定義された当社の開発、製造製品をいいます。

(Rev.4.0-1 2017.11)

## 本社所在地

〒135-0061 東京都江東区豊洲 3-2-24（豊洲フォレシア）

[www.renesas.com](http://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/](http://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。