RX ファミリ

LVD 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 LVD モジュールについて説明します。本モジュールは LVD を使用して、VCC と外部電圧レベルのいずれか、または両方の電圧状態の監視を行います。以降、本モジュールを LVD FIT モジュールと称します。

以降、本モジュールを LVD FIT モジュールと称します。

対象デバイス

- RX110、RX111、RX113 グループ
- RX130 グループ
- RX23T、RX230、RX231 グループ
- RX23W グループ
- RX24T グループ
- RX24U グループ
- RX64M グループ
- RX65N、RX651 グループ
- RX66T グループ
- RX71M グループ
- RX72T グループ
- RX72M 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。

ターゲットコンパイラ

- ルネサスエレクトロニクス製 C/C++ Compiler Package for RX Family
- GCC for Renesas RX
- IAR C/C++ Compiler for Renesas RX

各コンパイラの動作確認環境に関する詳細な内容は、セクション「7.1 動作確認環境」を参照してください。

目次

1. 概要

1.1　LVD 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、「2.12 FIT モジュールの追加方法」を参照してください。

1.2　LVD FIT モジュールの概要

LVD FIT モジュールは、2 チャネルの LVD 回路を使用し、VCC と外部電圧レベルのいずれか、または両方の電圧状態を監視することができます。本モジュールの API 関数を使用することで、低電圧検出に使用可能な LVD 関連レジスタを確認する必要がなくなります。

本モジュールは、チャネルごとに電圧検出条件の設定、および電圧検出時の処理を選択することができます。

- 電圧検出条件の設定
  - ・電圧検出レベル
  - ・監視電圧が電圧検出レベルを超えて上昇した場合、下降した場合のいずれか、または上昇と下降の両方を検出
- 電圧検出時の処理の選択
  - ・リセット
  - ・ノンマスカブル割り込み
  - ・マスカブル割り込み
  - ・処理なし

各機能のサポート範囲の詳細は、該当するユーザーズマニュアル ハードウェア編を参照してください。

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_LVD_Open()	指定されたチャネルを初期化し、LVD を開始します。
R_LVD_Close()	指定されたチャネルの LVD を停止します。
R_LVD_GetStatus()	指定されたチャネルの LVD ステータスを取得します。
R_LVD_ClearStatus()	指定されたチャネルの電圧検出レベル通過状態をクリアします。
R_LVD_GetVersion()	LVD FIT モジュールのバージョン番号を返します。

1.4　制限事項

LVD FIT モジュールでは以下の機能はサポートしていません。

- ELC リンク

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。

- LVD

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。

- ボードサポートパッケージ (r_bsp) v5.20 以上

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「7.1 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

マクロ定義 LVD_CFG_ACTION_CHANNEL_n（n=1、2）が“2”（マスカブル割り込み）の場合、R_LVD_Open 関数を実行したとき、電圧監視 n 割り込みが有効になります。

表 2.1 に本 FIT モジュールが使用する割り込みベクタを示します。

表 2.1　使用する割り込みベクター覧

デバイス	割り込みベクタ
RX110、RX111、RX113、 RX130、RX23T、RX230、 RX231、RX24T、RX24U、 RX64M、RX65N、RX65N_2M、 RX66T、RX71M、RX72T、 RX72M	LVD1 割り込み（ベクタ番号: 88） LVD2 割り込み（ベクタ番号: 89）
RX23W	LVD1 割り込み（ベクタ番号: 88）

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_lvd_rx_if.h に記載しています。

2.6 整数型

このドライバは 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_lvd_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。

コンフィギュレーションオプション（r_lvd_rx_config.h）	
LVD_CFG_PARAM_CHECKING_ENABLE ※デフォルト値は“BSP_CFG_PARAM_CHECKING_ENABLE”	各 API 関数でパラメータチェックを行うかどうかを指定します。 • 0：ビルド時にパラメータチェック処理をコードから省略 • 1：ビルド時にパラメータチェック処理をコードに含む • BSP_CFG_PARAM_CHECKING_ENABLE システムのデフォルト設定を使用 注：ビルド時にパラメータチェックのコードを省略することで、コードサイズを小さくすることができます。
LVD_CFG_CHANNEL_1_USED LVD_CFG_CHANNEL_2_USED ※初期値は“1”	該当チャネルを使用するかどうかを指定します。 • 0：該当チャネルを使用しない • 1：該当チャネルを使用する
LVD_CFG_VDET_TARGET_CHANNEL_1 LVD_CFG_VDET_TARGET_CHANNEL_2 ※初期値は“0”	監視対象をチャネルごとに指定します。 • 0：VCC • 1：CMPA2 端子
LVD_CFG_VOLTAGE_LEVEL_CHANNEL_1 LVD_CFG_VOLTAGE_LEVEL_CHANNEL_2 ※デフォルト値はハードウェア初期値に準拠するため、製品によって異なる	電圧検出レベルをチャネルごとに指定します。 定義値には、小数点第二位までの数値を整数で指定してください。 例： 電圧検出レベルを 3.00V に設定する場合、“300”を指定 電圧検出レベルを 4.29V に設定する場合、“429”を指定
LVD_CFG_DIGITAL_FILTER_CHANNEL_1 LVD_CFG_DIGITAL_FILTER_CHANNEL_2 ※初期値は“0”	デジタルフィルタを、チャネルごとに指定します。 • 0：デジタルフィルタ無効 • 1：デジタルフィルタ有効
LVD_CFG_SAMPLING_CLOCK_CHANNEL_1 LVD_CFG_SAMPLING_CLOCK_CHANNEL_2 ※デフォルト値はハードウェア初期値に準拠するため、製品によって異なる	デジタルフィルタが有効な場合に、チャネルごとに適用するサンプリングクロックとして、LOCO の分周比を指定します。 定義値には、分周値を整数で指定してください。 例： 1 分周を設定する場合、“1”を指定 4 分周を設定する場合、“4”を指定

LVD_CFG_ACTION_CHANNEL_1 LVD_CFG_ACTION_CHANNEL_2 ※デフォルト値は“1”	電圧検出時の処理を、チャネルごとに指定します。 • 0：リセット • 1：ノンマスカブル割り込み • 2：マスカブル割り込み • 3：処理なし 注：リセットはデバイスリセットを指します。リセット選択時は、監視対象電圧が電圧検出レベルより低い場合にリセットが発生します。リセット選択時は、電圧検出条件の設定に依存しません。
LVD_CFG_INT_PRIORITY_CHANNEL_1 LVD_CFG_INT_PRIORITY_CHANNEL_2 ※デフォルト値は“3”	マスカブル割り込みを選択した場合の割り込み優先度を、チャネルごとに指定します。 1 を優先度低、15 を優先度高として優先度を整数で指定してください。 例： 優先度を 3 に設定する場合“3”を指定 優先度を 15 に設定する場合“15”を指定
LVD_CFG_STABILIZATION_CHANNEL_1 LVD_CFG_STABILIZATION_CHANNEL_2 ※デフォルト値は“0” 注：ハードウェア初期値とは異なる	リセットを選択した場合のリセットネゲートタイミングを、チャネルごとに指定します。 • 0：LVD リセット発生後、監視対象電圧が電圧検出レベルを上回ってから一定時間後にネゲート • 1：LVD リセットアサートから一定時間後にネゲート 注：一定時間後とは、電圧監視リセット解除後待機時間です。詳細はユーザーズマニュアル ハードウェア編を参照してください。

2.8　コードサイズ

本モジュールのコード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。掲載した値は、「2.3 サポートされているツールチェーン」の C コンパイラでコンパイルオプションがデフォルト時の参考値です。コンパイルオプションのデフォルトは最適化レベル：2、最適化のタイプ：サイズ優先、データ・エンディアン：リトルエンディアンです。コードサイズは 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 と RAM のコードサイズ

デバイス	分類	使用メモリ						備考
		ルネサス製コンパイラ		GCC		IAR コンパイラ		
		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	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	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	
RX231	ROM サイズ（コード）	1943 バイト	1783 バイト	4008 バイト	3672 バイト	3102 バイト	2834 バイト	
	RAM サイズ	10 バイト	10 バイト	12 バイト	12 バイト	8 バイト	8 バイト	
RX23W	ROM サイズ（コード）	1639 バイト	1196 バイト	-	-	-	-	
	RAM サイズ	5 バイト	5 バイト	-	-	-	-	
RX65N RX65N-2M	ROM サイズ（コード）	2028 バイト	1859 バイト	2380 バイト	2176 バイト	3323 バイト	3055 バイト	
	RAM サイズ	10 バイト	10 バイト	12 バイト	12 バイト	10 バイト	10 バイト	
RX66T	ROM サイズ（コード）	2045 バイト	1876 バイト	4232 バイト	3904 バイト	3322 バイト	3052 バイト	
	RAM サイズ	10 バイト	10 バイト	12 バイト	12 バイト	10 バイト	10 バイト	
RX72T	ROM サイズ（コード）	2045 バイト	1876 バイト	4232 バイト	3904 バイト	3322 バイト	3052 バイト	
	RAM サイズ	10 バイト	10 バイト	12 バイト	12 バイト	10 バイト	10 バイト	
RX72M	ROM サイズ（コード）	2045 バイト	1876 バイト	4408 バイト	4056 バイト	3292 バイト	3020 バイト	
	RAM サイズ	10 バイト	10 バイト	12 バイト	12 バイト	10 バイト	10 バイト	

2.9　引数

API 関数の引数である構造体を示します。この構造体は、API 関数のプロトタイプ宣言とともに r_lvd_rx_if.h に記載されています。

2.9.1　チャネル

この列挙型は MCU で使用できるチャネルを定義します。

```
typedef enum
{
      LVD_CHANNEL_1 = 0,
      LVD_CHANNEL_2,
      LVD_CHANNEL_INVALID
} lvd_channel_t;
```

列挙型名	説明
LVD_CHANNEL_1	LVD channel 1
LVD_CHANNEL_2	LVD channel 2

2.9.2 電圧検出条件

この列挙型は電圧検出条件を定義します。この列挙型は、割り込み発生条件および LVD ステータスに影響を与えます。リセットは監視電圧が電圧検出レベルより低い場合に発生するため、この列挙型の影響を受けません。

```
typedef enum
{
      LVD_TRIGGER_RISE = 0,
      LVD_TRIGGER_FALL,
      LVD_TRIGGER_BOTH,
      LVD_TRIGGER_INVALID
} lvd_trigger_t;
```

列挙型名	説明
LVD_TRIGGER_RISE	電圧上昇
LVD_TRIGGER_FALL	電圧下降
LVD_TRIGGER_BOTH	電圧上昇および下降の両方

2.9.3 電圧検出レベルに対する状態

この列挙型は電圧検出レベルの通過状態を定義します。以降、本ドキュメントではこの状態を" voltage position status"と称します。

```
typedef enum
{
      LVD_STATUS_POSITION_ABOVE = 0,
      LVD_STATUS_POSITION_BELOW,
      LVD_STATUS_POSITION_INVALID
} lvd_status_position_t;
```

列挙型名	説明
LVD_STATUS_POSITION_ABOVE	電圧検出レベルより高い状態
LVD_STATUS_POSITION_BELOW	電圧検出レベルより低い状態

2.9.4 電圧検出レベルの通過状態

この列挙型は電圧検出レベルの通過状態を定義します。以降、本ドキュメントではこの状態を"voltage crossing status "と称します。

```
typedef enum
{
      LVD_STATUS_CROSS_NONE = 0,
      LVD_STATUS_CROSS_OVER,
      LVD_STATUS_CROSS_INVALID
} lvd_status_cross_t;
```

列挙型名	説明
LVD_STATUS_CROSS_NONE	電圧検出レベル未通過の状態
LVD_STATUS_CROSS_OVER	電圧検出レベルを通過した状態

2.9.5 コンフィギュレーション設定

このデータ構造体は、R_LVD_Open()関数に送信される構造体を定義します。

```
typedef struct
{
    lvd_trigger_t trigger;
} lvd_config_t;
```

2.9.6 コールバック

このデータ構造体は、コールバック関数に送信される構造体を定義します。

```
typedef struct
{
    bsp_int_src_t vector;
} lvd_int_cb_args_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_lvd_rx_if.h で記載されています。

```
typedef enum
{
      LVD_SUCCESS = 0,
      LVD_ERR_INVALID_PTR,
      LVD_ERR_INVALID_FUNC,
      LVD_ERR_INVALID_DATA,
      LVD_ERR_INVALID_CHAN,
      LVD_ERR_INVALID_ARG,
      LVD_ERR_UNSUPPORTED,
      LVD_ERR_ALREADY_OPEN,
      LVD_ERR_NOT_OPENED,
      LVD_ERR_LOCO_STOPPED
} lvd_err_t;
```

2.11 コールバック関数

本モジュールでは、LVD 割り込みが発生したタイミングで。ユーザが設定したコールバック関数を呼び出します。

コールバック関数は、「2.9 引数」に記載された構造体メンバ“void (*p_callback)(void *)”に、ユーザの関数のアドレスを格納することで設定されます。コールバック関数が呼び出されるとき、定数が格納された変数が、引数として渡されます。

引数の型は void ポインタ型で渡されるため、コールバック関数の引数は下記の例を参考に void 型のポインタ変数としてください。

コールバック関数は lvd_int_cb_args_t*の型で引数を持ち、vector に以下の割り込み要因が設定されます。使用例は、セクション 5 を参照してください。

- BSP_INT_SRC_LVD1:LVD channel 1
- BSP_INT_SRC_LVD2:LVD channel 2

コールバック関数内部で値を使うときはキャストして値を使用してください。

```
void lvd_isr_callback(void *p_args)
{
  lvd_err_t             err;
  lvd_status_position_t status_position;
  lvd_status_cross_t          status_cross
  lvd_int_cb_args_t           *p_cb_args;

  p_cb_args = (lvd_int_cb_args_t*)p_args;
```

```
    …
}
```

2.12 FIT モジュールの追加方法

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

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

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

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

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

2.13 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。

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_LVD_Open

本関数は指定されたチャネルを初期化し、LVD を開始します。

Format

```
lvd_err_t      R_LVD_Open (
        lvd_channel_t              channel,
        lvd_config_t const         *p_cfg,
        void (*p_callback)(void *)
)
```

Parameters

lvd_channel_t channel
　初期化および LVD を開始するチャネルの列挙型

*lvd_config_t const *p_cfg*
　コンフィギュレーション構造体のアドレス

p_callback
　電圧検出時に割り込みから呼び出されるコールバック関数のアドレス

Return Values

```
[LVD_SUCCESS]                /*成功    : LVD を開始   */
[LVD_ERR_INVALID_PTR]        /*エラー : 引数 p_cfg のアドレスが無効 */
[LVD_ERR_INVALID_FUNC]       /* エラー : 引数 p_callback のアドレスが無効    */
[LVD_ERR_INVALID_DATA]       /* エラー : コンフィギュレーションオプションの定義が無効 */
[LVD_ERR_INVALID_CHAN]       /* エラー : 引数 channel が無効 */
[LVD_ERR_INVALID_ARG]        /* エラー : 引数 p_cfg のデータが無効   */
[LVD_ERR_UNSUPPORTED]        /* エラー : 選択した機能はサポート対象外        */
[LVD_ERR_ALREADY_OPEN]       /* エラー : 指定チャネルはオープンされた状態  */
[LVD_ERR_ LOCO_STOPPED]      /* エラー : LOCO 停止中の設定が無効   */
```

Properties

ファイル r_lvd_rx_if.h にプロトタイプ宣言されています。

Description

本関数は、引数 *p_cfg* およびコンフィギュレーションオプションの設定に従って、指定された LVD チャネルの初期化と電圧検出時の処理設定を行い、LVD を開始します。本関数が成功すると、指定チャネルはオープンされた状態になります。

本関数は、チャネルごとに実行されますが、電圧検出レベルおよび監視対象の選択については、全ての LVD 回路が停止中の場合のみ、コンフィギュレーションオプションの設定が反映されます。

コンフィギュレーションオプションで指定する電圧検出時の処理によって、引数 *p_callback* へのコールバック関数の登録の要否が異なります。下表に詳細を示します。

電圧検出時の処理 （LVD_CFG_ACTION_CHANNEL_1） （LVD_CFG_ACTION_CHANNEL_2）	コールバック関数の要否 （引数 *p_callback*）
リセット	不要：FIT_NO_FUNC を設定
ノンマスカブル割り込み	必要：コールバック関数アドレスを設定
マスカブル割り込み	必要：コールバック関数アドレスを設定
処理なし	不要：FIT_NO_FUNC を設定

Example

電圧検出時の処理にリセットを指定し、本関数を呼び出す場合の例を示します。

他条件での設定例は、セクション 5 を参照してください。

```
lvd_err_t   err;
lvd_config_t      cfg;

cfg.trigger = LVD_TRIGGER_FALL;
err = R_LVD_Open(LVD_CHANNEL_1, &cfg, FIT_NO_FUNC);
```

Special Notes:

LOCO 停止中はコンフィグレーションオプションの定義で以下の設定は使用できません。LOCO 停止中に以下の設定を指定して本関数を実行した場合、*LVD_ERR_LOCO_STOPPED* のエラーを返します。

- 電圧検出時の処理にリセットを選択した場合、LVD_CFG_STABILIZATION_CHANNEL_n (n = 1、2) (リセットネゲートタイミング)を“1”に設定しないでください。
- LVD_CFG_DIGITAL_FILTER_CHANNEL_n (n = 1、2) (デジタルフィルタ有効/無効設定)を“1”に設定しないでください。

リセット発生時は、リセットの種類によって LVD 関連レジスタの初期化対象が異なります。リセット発生後、レジスタを初期化されない LVD は動作を継続します。電圧検出レベルおよび監視対象の選択は、全ての LVD 回路が停止中の場合のみコンフィギュレーションオプションの設定が反映されるため、リセット発生後は必要に応じて R_LVD_Close 関数を実行するなどの処置をとってください。

また、リセット発生後、レジスタを初期化されない LVD が、割り込みによるコールバック関数呼び出し等、ソフトウェアを介在する処理を行っていた場合、正しく継続されません。R_LVD_Open 関数を実行することで、ソフトウェアを介在する処理が有効になります。

一部 MCU では、LVD チャネル 1 と、オプション機能選択レジスタで設定する電圧監視リセットが兼用になっています。これらの MCU では、オプション機能で電圧監視リセットが有効になっている場合、LVD チャネル 2 の電圧監視を開始するためには、LVD チャネル 1 を使用してオプション機能の電圧監視リセットをいったん停止する必要があります。本関数を使用して LVD チャネル 1 の動作を再開する場合、オプション機能の電圧監視リセット設定ではなく、本モジュールの設定値が反映されるため注意が必要です。

各 MCU の制限およびサポート機能については、該当するユーザーズマニュアル ハードウェア編を参照してください。

LVD_CFG_ACTION_CHANNEL_n (n = 1、2)を“1”、もしくは“2”に設定する場合、R_LVD_Open 関数実行後に R_LVD_GetStatus 関数を使用して電源電圧の状態を確認し、電圧検出レベルの通過 (LVD_STATUS_CROSS_OVER)を検出していた場合は R_LVD_ClearStatus 関数を実行してクリアしてください。

R_LVD_Close

本関数は指定されたチャネルの LVD を停止します。

Format

```
lvd_err_t      R_LVD_Close (
        lvd_channel_t    channel
)
```

Parameters

lvd_channel_t channel
LVD を停止するチャネルの列挙型

Return Values

[LVD_SUCCESS] */*成功　: LVD を停止　*/*
[LVD_ERR_INVALID_CHAN] */* エラー：引数 channel が無効 */*

Properties

ファイル r_lvd_rx_if.h にプロトタイプ宣言されています。

Description

本関数は、指定されたチャネルの LVD を停止します。本関数が正常に終了すると、指定チャネルは未オープンの状態に戻ります。

Example

本関数の呼び出し例を示します。

他の条件での設定例は、セクション 5 を参照してください。

```
lvd_err_t err;

err = R_LVD_Close(LVD_CHANNEL_1);
```

Special Notes:

なし

R_LVD_GetStatus

本関数は指定されたチャネルの LVD ステータスを取得します。

Format

```
lvd_err_t      R_LVD_GetStatus       (
        lvd_channel_t             channel,
        lvd_status_position_t     *p_status_position,
        lvd_status_cross_t        *p_status_cross
)
```

Parameters

lvd_channel_t channel
ステータスを取得するチャネルの列挙型

lvd_status_position_t p_status_position
電圧検出レベルに対する状態の列挙型を格納するアドレス

lvd_status_cross_t p_status_cross
電圧検出レベルの通過状態の列挙型を格納するアドレス

Return Values

*[LVD_SUCCESS] /*成功 : LVD ステータスを取得*/*
*[LVD_ERR_INVALID_PTR] /*エラー : 引数 p_status_position、p_status_cross のアドレスが無効*/*
[LVD_ERR_INVALID_CHAN]/ エラー : 引数 channel が無効*/*
[LVD_ERR_NOT_OPENED] / エラー : 指定チャネルがオープンされていない*/*

Properties

ファイル r_lvd_rx_if.h にプロトタイプ宣言されています。

Description

本関数は、指定されたチャネルの LVD ステータスを引数*p_status_position* および*p_status_cross* に格納します。ステータスの詳細は

図 **3.1** を参照してください。

引数*p_status_position* に格納される電圧検出レベルに対する状態は、電圧検出条件の設定に依存せずに状態を取得することができます。引数*p_status_cross* に格納される電圧検出レベルの通過状態は、電圧検出条件の設定に依存し、条件を満たした場合のみ通過状態となります。

本関数実行前に、指定チャネルで R_LVD_Open()関数を実行し、該当チャネルをオープンされた状態にする必要があります。

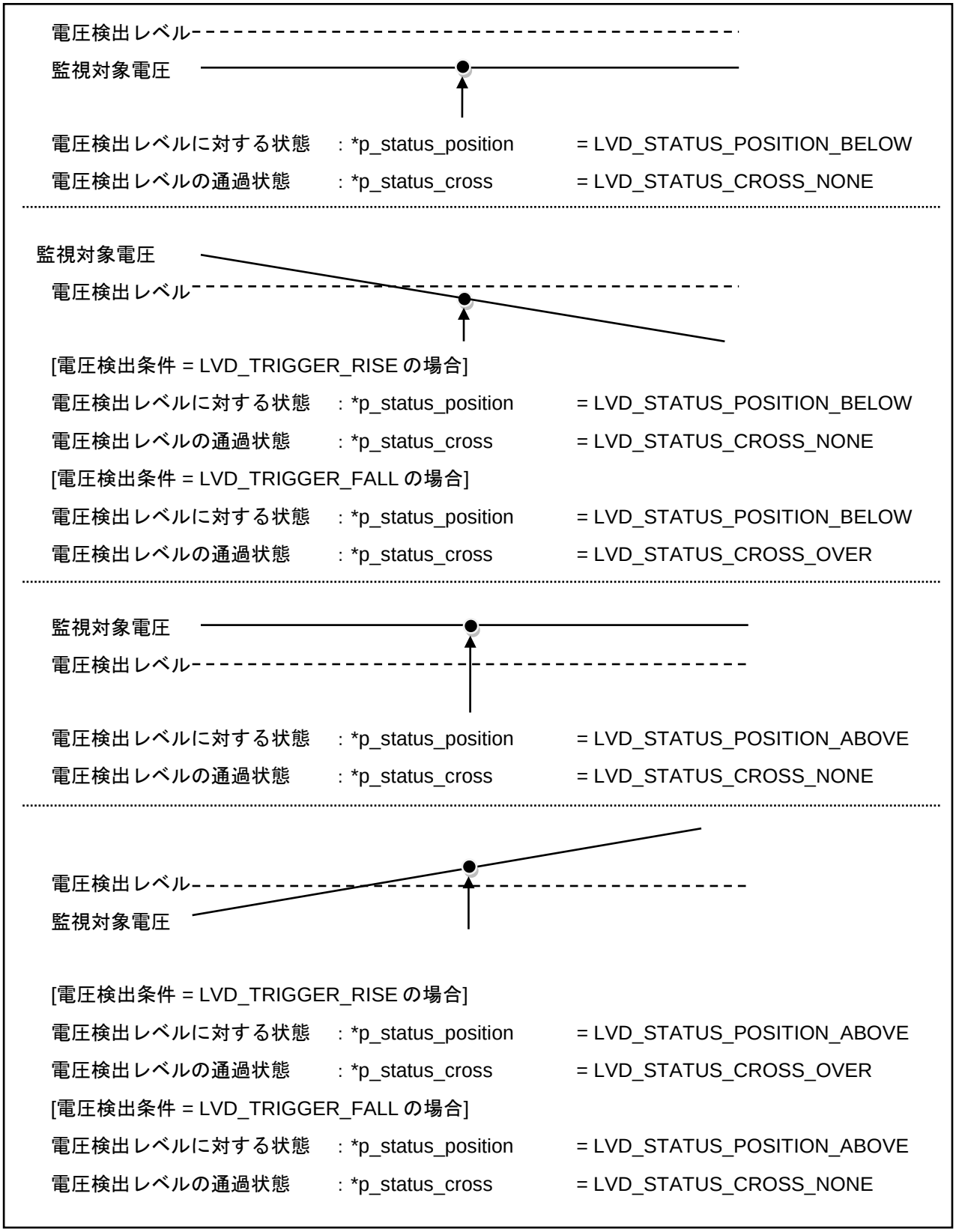

図 3.1 電圧検出レベルに対する監視対象電圧の状態と LVD ステータス

Example

本関数の呼び出し例を示します。

他の条件での設定例は、セクション 5 を参照してください。

```
lvd_err_t err;
lvd_status_position_t status_pos;
lvd_status_cross_t status_cross;

status = R_LVD_GetStatus (LVD_CHANNEL_1, &status_pos, &status_cross);
```

Special Notes:

なし

R_LVD_ClearStatus

本関数は指定されたチャネルの電圧レベル通過状態をクリアします。

Format

```
lvd_err_t      R_LVD_ClearStatus (
        lvd_channel_t    channel
)
```

Parameters

lvd_channel_t channel
電圧レベル通過状態をクリアするチャネルの列挙型

Return Values

[LVD_SUCCESS]	*/*成功　：電圧レベル通過状態をクリア*/*
[LVD_ERR_INVALID_CHAN]	*/* エラー：引数 channel が無効*/*
[LVD_ERR_NOT_OPENED]	*/* エラー：指定チャネルがオープンされていない*/*

Properties

ファイル r_lvd_rx_if.h にプロトタイプ宣言されています。

Description

本関数は、指定された LVD チャネルに対し LVD 電圧レベルの通過状態をクリアし、未通過の状態に戻します。LVD 電圧レベルの通過状態をクリアするために、割り込みおよびリセットを一時的に禁止します。

本関数実行前に、指定チャネルで R_LVD_Open()関数を実行し、該当チャネルをオープンされた状態にする必要があります。

Example

本関数の呼び出し例を示します。

他の条件での設定例は、セクション 5 を参照してください。

```
lvd_err_t err;

err = R_LVD_ClearStatus (LVD_CHANNEL_1);
```

Special Notes:

本関数で割り込みおよびリセットを一時的に禁止している間に電圧検出した場合、割り込みおよびリセットは実行されませんので注意が必要です。

R_LVD_GetVersion

本関数は LVD FIT モジュールのバージョン番号を返します。

Format
uint32_t R_LVD_GetVersion (void)

Parameters
なし

Return Values
バージョン番号

Properties
ファイル r_lvd_rx_if.h にプロトタイプ宣言されています。

Description
この関数は LVD FIT 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。例えば、ver. 4.25 の場合、“0x00040019”が返されます。

Example
本関数の呼び出し例を示します。

```
uint32_t version;

version = R_LVD_GetVersion();
```

Special Notes:
本関数は“#pragma inline”を使用してインライン化されています。

4. 端子設定

LVD FIT モジュールはピン設定を使用しません。

5. 使用例

LVD FIT モジュール使用時の設定例を以下に示します。

5.1 LVD チャネル 1 で VCC を監視しリセットを行う場合

LVD チャネル 1 で VCC が 4.29V 以下に下降した場合に、リセットを行う設定例を以下に示します。

r_lvd_rx_config.h ファイルのコンフィギュレーションオプションで、以下のマクロを設定します。

必要に応じて、リセットネゲートタイミングの指定およびデジタルフィルタの設定も行ってください。

- 使用 LVD チャネル : チャネル 1

```
#define LVD_CFG_CHANNEL_1_USED      (1)
```

- 監視対象 : VCC

```
#define LVD_CFG_VDET_TARGET_CHANNEL_1 (0)
```

- 電圧レベル : 4.29V

```
#define LVD_CFG_VOLTAGE_LEVEL_CHANNEL_1 (429)
```

- アクション : リセット

```
#define LVD_CFG_ACTION_CHANNEL_1 (0)
```

R_LVD_Open()関数を実行して LVD を開始します。

```
void main(void)
{
      lvd_err_t err;
      lvd_config_t cfg;

      cfg.trigger = LVD_TRIGGER_FALL;
      err = R_LVD_Open(LVD_CHANNEL_1, &cfg, FIT_NO_FUNC);
}
```

5.2 LVD チャネル 2 で CMPA2 を監視し割り込みを行う場合

LVD チャネル 2 で CMPA2 が 4.29V より上昇もしくは下降した場合に、マスカブル割り込みを行う設定例を以下に示します。

r_lvd_rx_config.h ファイルのコンフィギュレーションオプションで、以下のマクロを設定します。

必要に応じて、割り込み優先レベルおよびデジタルフィルタの設定も行ってください。

- 使用 LVD チャネル : チャネル 2

```
#define LVD_CFG_CHANNEL_2_USED      (1)
```

- 監視対象 : CMPA2

```
#define LVD_CFG_VDET_TARGET_CHANNEL_2 (1)
```

- 電圧レベル : 4.29V

```
#define LVD_CFG_VOLTAGE_LEVEL_CHANNEL_2 (429)
```

- アクション : マスカブル割り込み

```
#define LVD_CFG_ACTION_CHANNEL_2 (2)
```

R_LVD_Open()関数を実行して LVD を開始します。

```
void main(void)
{
      lvd_err_t err;
      lvd_config_t cfg;

      cfg.trigger = LVD_TRIGGER_BOTH;
      err = R_LVD_Open(LVD_CHANNEL_2, &cfg, (void*)lvd_isr_callback);
}
```

電圧検出時に実行されるコールバック関数を用意し、必要に応じて処理を実行します。

```
void lvd_isr_callback(void *p_args)
{
  lvd_err_t             err;
  lvd_status_position_t status_position;
  lvd_status_cross_t          status_cross
  lvd_int_cb_args_t           *p_cb_args;

  p_cb_args = (lvd_int_cb_args_t*)p_args;
  if (BSP_INT_SRC_LVD2 == p_cb_args->vector)
  {
      err = R_LVD_GetStatus(LVD_CHANNEL_2, &status_position, &status_cross);
      if (status_position == LVD_STATUS_POSITION_ABOVE)
      {
            /* User code */
      }
      else
      {
            /* User code */
      }
      err = R_LVD_ClearStatus(LVD_CHANNEL_2);
  }
}
```

5.3 LVD チャネル 2 のステータスを取得する場合

LVD チャネル 2 で CMPA2 が 4.29V より上昇もしくは下降した場合に、検出時の処理に「処理なし」を設定し、LVD チャネルステータスを取得する方法を以下に示します。

r_lvd_rx_config.h ファイルのコンフィギュレーションオプションで、以下のマクロを設定します。

必要に応じてデジタルフィルタの設定も行ってください。

- 使用 LVD チャネル : チャネル 2

```
#define LVD_CFG_CHANNEL_2_USED (1)
```

- 監視対象 : CMPA2

```
ine LVD_CFG_VDET_TARGET_CHANNEL_2 (1)
```

- 電圧レベル : 4.29V

```
#define LVD_CFG_VOLTAGE_LEVEL_CHANNEL_2 (429)
```

- アクション : 処理なし

```
#define LVD_CFG_ACTION_CHANNEL_2 (3)
```

R_LVD_Open()関数を実行して LVD を開始し、ステータスを取得します。

```
void main(void)
{
  lvd_err_t             err;
  lvd_config_t                cfg;
  lvd_status_position_t status_pos;
  lvd_status_cross_t          status_cross;

  cfg.trigger = LVD_TRIGGER_BOTH;
  err = R_LVD_Open(LVD_CHANNEL_2, &cfg, FIT_NO_FUNC);

  err = R_LVD_GetStatus (LVD_CHANNEL_2, &status_pos, &status_cross);
}
```

5.4　LVD チャネル 1 で電圧検出条件を変更する場合

LVD チャネル 1 で VCC が 4.29V 以下に下降した場合にマスカブル割り込みを行う設定で、LVD 開始後、VCC が 4.29V 以上に上昇した場合に電圧検出条件を変更する設定例を以下に示します。

r_lvd_rx_config.h ファイルのコンフィギュレーションオプションで、以下のマクロを設定します。

必要に応じてリセットネゲートの指定およびデジタルフィルタの設定も行ってください。

- 使用 LVD チャネル　　：チャネル 1

```
#define LVD_CFG_CHANNEL_1_USED      (1)
```

- 監視対象　　：VCC

```
#define LVD_CFG_VDET_TARGET_CHANNEL_1 (0)
```

- 電圧レベル　　：4.29V

```
#define LVD_CFG_VOLTAGE_LEVEL_CHANNEL_1 (429)
```

- アクション　　：マスカブル割り込み

```
#define LVD_CFG_ACTION_CHANNEL_1 (2)
```

R_LVD_Open()関数を実行して LVD を開始後、電圧検出条件を変更して再開します。

```
void main(void)
{
  lvd_err_t err;
  lvd_config_t cfg;

  cfg.trigger = LVD_TRIGGER_FALL;
  err = R_LVD_Open(LVD_CHANNEL_1, &cfg, (void*)lvd_isr_callback);

  err = R_LVD_Close(LVD_CHANNEL_1);

  cfg.trigger = LVD_TRIGGER_RISE;
  err = R_LVD_Open(LVD_CHANNEL_1, &cfg, (void*)lvd_isr_callback);
}
```

電圧検出時に実行されるコールバック関数を用意し、必要に応じて処理を実行します。

```
void lvd_isr_callback(void *p_args)
{
  lvd_err_t             err;
  lvd_status_position_t status_position;
  lvd_status_cross_t          status_cross
  lvd_int_cb_args_t           *p_cb_args;

  p_cb_args = (lvd_int_cb_args_t*)p_args;
  if (BSP_INT_SRC_LVD1 == p_cb_args->vector)
  {
      err = R_LVD_GetStatus(LVD_CHANNEL_1, &status_position, &status_cross);
      if (status_position == LVD_STATUS_POSITION_ABOVE)
      {
            /* User code */
      }
      else
      {
            /* User code */
      }
      err = R_LVD_ClearStatus(LVD_CHANNEL_1);
  }
}
```

6. デモプロジェクト

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。

6.1 lvd_demo_rskrx113

lvd_demo_rskrx113 プロジェクトの動作条件を以下に示します。デモプロジェクトは、LVD 割り込みを扱うコールバック関数の設定方法、またコールバックの引数のチャネル情報を逆引きする方法をデモするものです。プログラム実行時、LVD のコールバック関数で電圧上昇検出時は LED0 を点灯、電圧下降検出時は LED0 を消灯します。デモ動作中は LED1 を点灯し、プログラムが実行中であることを示します。

- 使用 LVD チャネル : チャネル 1
- 監視対象 : VCC
- 電圧レベル : 2.90V
- アクション : ノンマスカブル割り込み
- 電圧検出条件 : 上昇もしくは下降

注記：

- LVD デモは、スタンドアロンモードで実行する必要があります（プログラムをボードにダウンロードし、デバッガを接続解除した後、外部電源を接続します）。
- 可変電源（PSU）を使用して、入力電圧を調整する必要があります。この PSU を、RSK ボードの電源ジャック（PWR）に接続します。

6.2 lvd_demo_rskrx231

lvd_demo_rskrx231 プロジェクトは、lvd_demo_rskrx113 と同じものです。

6.3 lvd_demo_rskrx64m

lvd_demo_rskrx64m プロジェクトは、電圧レベルを除き lvd_demo_rskrx113 と同じものです。

- 電圧レベル：2.99V

6.4 lvd_demo_rskrx65n

lvd_demo_rskrx65n プロジェクトは、電圧レベルを除き lvd_demo_rskrx113 と同じものです。

- 電圧レベル：2.85V

6.5 lvd_demo_rskrx65n_2m

lvd_demo_rskrx65n_2m プロジェクトは、電圧レベルを除き lvd_demo_rskrx113 と同じものです。

- 電圧レベル：2.85V

6.6 ワークスペースにデモを追加する

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。

6.7 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。

7. 付録

7.1 動作確認環境

本 FIT モジュールの動作確認環境を以下に示します。

表 7.1 動作確認環境 (Rev.3.20)

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 V.7.5.0 IAR Embedded Workbench for Renesas RX 4.12.1
C コンパイラ	ルネサスエレクトロニクス製 C/C++ Compiler Package 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
	GCC for Renesas RX 4.8.4.201902 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 -std=gnu99 リンクオプション：「Optimize size (サイズ最適化) (-Os)」を使用する場合、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 -Wl,--no-gc-sections これは、FIT 周辺機器モジュール内で宣言されている割り込み関数をリンカが誤って破棄（discard）することを回避（work around）するための対策です。
	IAR C/C++ Compiler for Renesas RX version 4.12.1 コンパイルオプション：統合開発環境のデフォルト設定
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	Rev.3.20
使用ボード	Renesas Starter Kit+ for RX72M（型名：RTK5572Mxxxxxxxxxx）

表 7.2 動作確認環境 (Rev.3.10)

項目	内容
統合開発環境	ルネサスエレクトロニクス製 e^2 studio V.7.5.0
C コンパイラ	ルネサスエレクトロニクス製 C/C++ Compiler Package 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	Rev.3.10
使用ボード	Renesas Solution Starter Kit for RX23W（型名：RTK5523Wxxxxxxxxx）

表 7.3 動作確認環境 (Rev.3.00)

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 V.7.4.0 IAR Embedded Workbench for Renesas RX 4.10.1
C コンパイラ	ルネサスエレクトロニクス製 C/C++ Compiler Package 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
	GCC for Renesas RX 4.8.4.201803 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 -std=gnu99 リンクオプション：「Optimize size (サイズ最適化) (-Os)」を使用する場合、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 -Wl,--no-gc-sections これは、FIT 周辺機器モジュール内で宣言されている割り込み関数をリンカが誤って破棄（discard）することを回避（work around）するための対策です。
	IAR C/C++ Compiler for Renesas RX version 4.10.1 コンパイルオプション：統合開発環境のデフォルト設定
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	Rev.3.00
使用ボード	Renesas Starter Kit+ for RX65N-2MB（型名：RTK50565Nxxxxxxxxx）

表 7.4 動作確認環境 (Rev.2.50)

項目	内容
統合開発環境	ルネサスエレクトロニクス製 e^2 studio V.7.3.0.
C コンパイラ	ルネサスエレクトロニクス製 C/C++ Compiler Package 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	Rev.2.50
使用ボード	Renesas Starter Kit for RX72T（型名：RTK5572Txxxxxxxxxx）

表 7.5 動作確認環境 (Rev.2.41)

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 V.7.3.0.
C コンパイラ	ルネサスエレクトロニクス製 C/C++ Compiler Package 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	Rev.2.41
使用ボード	Renesas Starter Kit for RX66T（型名：RTK50566T0SxxxxxBE） Renesas Starter Kit+ for RX 65N-2MB（型名：RTK50565N2SxxxxxBE） Renesas Starter Kit+ for RX130-512KB（型名：RTK5051308SxxxxxBE）

表 7.6 動作確認環境 (Rev.2.40)

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 V.7.0.0.
C コンパイラ	ルネサスエレクトロニクス製 C/C++ Compiler Package for RX Family V3.00.00
	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	Rev.2.40
使用ボード	Renesas Starter Kit for RX66T（型名：RTK50566T0SxxxxxBE） Renesas Starter Kit+ for RX 65N-2MB（型名：RTK50565N2SxxxxxBE） Renesas Starter Kit+ for RX130-512KB（型名：RTK5051308SxxxxxBE）

表 7.7 動作確認環境 (Rev.2.31)

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 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	Rev2.31
使用ボード	Renesas Starter Kit+ for RX 65N-2MB（型名：RTK50565N2SxxxxxBE） Renesas Starter Kit+ for RX130-512KB（型名：RTK5051308SxxxxxBE）

表 7.8 動作確認環境 (Rev.2.30)

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 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	Rev2.30
使用ボード	Renesas Starter Kit+ for RX65N-2MB（型名：RTK50565N2SxxxxxBE） Renesas Starter Kit for RX130-512KB（型名：RTK5051308SxxxxxBE）

7.2 トラブルシューティング

(1) Q：本 FIT モジュールをプロジェクトに追加しましたが、ビルド実行すると「Could not open source file "platform.h"」エラーが発生します。

A：FIT モジュールがプロジェクトに正しく追加されていない可能性があります。プロジェクトへの追加方法をご確認ください。

- 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^2 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^2 studio に組み込む方法 Firmware Integration Technology (R01AN1723)」

また、本 FIT モジュールを使用する場合、ボードサポートパッケージ 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。BSP モジュールの追加方法は、アプリケーションノート「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：本 FIT モジュールをプロジェクトに追加しましたが、ビルド実行すると「This MCU is not supported by the current r_lvd_rx module.」エラーが発生します。

A：追加した FIT モジュールがユーザプロジェクトのターゲットデバイスに対応していない可能性があります。追加した FIT モジュールの対象デバイスを確認してください。

(3) Q：本 FIT モジュールをプロジェクトに追加しましたが、ビルド実行すると「コンフィグ設定が間違っている場合のエラーメッセージ」エラーが発生します。

A：“r_lvd_rx_config.h”ファイルの設定値が間違っている可能性があります。
“r_lvd_rx_config.h”ファイルを確認して正しい値を設定してください。詳細は「2.7 コンパイル時の設定」を参照してください。

8. 参考ドキュメント

ユーザーズマニュアル：ハードウェア
最新版をルネサス エレクトロニクスホームページから入手してください。

テクニカルアップデート／テクニカルニュース
最新の情報をルネサス エレクトロニクスホームページから入手してください。

ユーザーズマニュアル：開発環境
RX ファミリ CC-RX コンパイラ ユーザーズマニュアル（R20UT3248）
最新版をルネサス エレクトロニクスホームページから入手してください。

テクニカルアップデートの対応について

本モジュールは以下のテクニカルアップデートの内容を反映しています。
TN-RX*-A137A/E
RX230 グループ、 RX231 グループ ユーザーズマニュアル ハードウェア編 Rev.1.00（Page 1941of 1968）
「表 50.57 パワーオンリセット回路、電圧検出特性 (1)」の検出電圧 の訂正

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
2.00	2016.06.15	—	初版発行
2.10	2016.10.01	—	FIT モジュールの RX65N グループ対応
		1	「RX ファミリ 電圧検出回路 (LVD) モジュール Firmware Integration Technology (R01AN1726)」との互換性に関する注意書きを追加
2.20	2017.02.28	—	FIT モジュールの RX24U グループ対応
		—	誤記修正
		4	「2.5 対応ツールチェーン」に RXC v2.06.00 を追加
		12	3.2 R_LVD_Open: 「Special Notes」に割り込み有効設定時の注意事項を追加
		Program	引数のチェックを、NULL と FIT_NO_PTR、および FIT_NO_FUNC でチェックするように修正
2.30	2017.07.24	—	FIT モジュールの RX130-512KB、RX65N-2MB 対応
		1	「関連ドキュメント」に以下のドキュメントを追加: Renesas e^2 studio スマート・コンフィグレータ ユーザーガイド(R20AN0451)
		5	2.6 使用する割り込みベクタ: 追加
		10	2.14 FIT モジュールの追加方法: 変更
		24	5.5 デモのダウンロード方法: 追加
		25-26	6. 付録: 追加
2.31	2017.10.31	—	RX65N と RX65N-2MB のサポートを追加
		5	2.6 使用する割り込みベクタ：RX65N と RX65N-2MB を追加
		7	2.10 コードサイズ：RX65N と RX65N_2M に対応するコードサイズを追加
		24	5.1 lvd_demo_rskrx113：動作条件を追加
		24	5.4 lvd_demo_rskrx65n を追加
		24	5.5 lvd_demo_rskrx65n_2m を追加
		26	6.1 動作確認環境：
			Rev. 2.31 に対応する表を追加
		27	6.2 トラブルシューティング：質問を 1 つ追加。.
2.40	2018.09.28	1、5	RX66T のサポートを追加。
		8	RX66T に対応するコードサイズを追加。
		28	「7.1 動作確認環境」：Rev.2.40 に対応する表を追加。
2.41	2018.11.16	—	XML 内にドキュメント番号を追加。
		1	RX651 のサポートを追加。
		28	Renesas Starter Kit+ for RX66T の型名を変更。
			Rev.2.41 に対応する表を追加。
2.50	2019.02.01	—	「voltage detection level」(電圧検出レベル) を設定した場合にスマートコンフィギュレータの異常動作を引き起こす MDF ファイル内のバグを修正。
		—	RX72T グループのサポートを追加。
		1、5	RX72T グループのサポートを追加。
		8	RX72T に対応するコードサイズを追加。
		14-20	各 API 関数で「Reentrant」の説明を削除。
		28	「7.1 動作確認環境」Rev 2.50 に対応する表を追加。

3.00	2019.05.20	—	以下のコンパイラをサポート。 - GCC for Renesas RX - IAR C/C++ Compiler for Renesas RX
		1	RX631 と RX63N の更新終了につき、「対象デバイス」からこれらのデバイスを削除。 「ターゲットコンパイラ」のセクションを追加。 関連ドキュメントを削除。
		5	「2.2 ソフトウェアの要求」r_bsp v5.20 以上が必要
		8	「2.8 コードサイズ」セクションを更新。
		28	表 7.1「動作確認環境」： Rev.3.00 に対応する表を追加。
		32	「Web サイトおよびサポート」のセクションを削除。
		プログラム	GCC と IAR コンパイラに関して、以下を変更。 1. R_LVD_GetVersion 関数のインライン展開を削除。 2. 割り込み関数の宣言を、BSP のマクロ定義で置き換えた。
3.10	2019.06.28	1、5	RX23W のサポートを追加。
		8	RX23W に対応するコードサイズを追加。
		28	「7.1 動作確認環境」： Rev.3.10 に対応する表を追加。
		プログラム	RX23W のサポートを追加。
3.20	2019.08.15	1、5	RX72M のサポートを追加。
		8	RX72M に対応するコードサイズを追加。
		28	「7.1 動作確認環境」： Rev.3.20 に対応する表を追加。 表 6.2：RX23W ボード名変更。
		プログラム	RX72M のサポートを追加。

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

注 1. 本資料において使用されている「当社」とは、ルネサス エレクトロニクス株式会社およびルネサス エレクトロニクス株式会社が直接的、間接的に支配する会社をいいます。

注 2. 本資料において使用されている「当社製品」とは、注 1 において定義された当社の開発、製造製品をいいます。

(Rev.4.0-1 2017.11)

本社所在地

〒135-0061 東京都江東区豊洲 3-2-24（豊洲フォレシア）

www.renesas.com

お問合せ窓口

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

www.renesas.com/contact/

商標について

ルネサスおよびルネサスロゴはルネサス エレクトロニクス株式会社の商標です。すべての商標および登録商標は、それぞれの所有者に帰属します。