RX ファミリ

SX-ULPGN-2000 Wi-Fi モジュール制御モジュール

Firmware Integration Technology

要旨

本アプリケーションノートは、Firmware Integration Technology (FIT)に準拠した SX-ULPGN-2000 Wi-Fi モジュール制御モジュールの使用方法について説明します。

以降、SX-ULPGN-2000 Wi-Fi モジュール制御モジュールのソフトウェアを総じて”SX-ULPGN Wi-Fi FIT モジュール”、または”本 FIT モジュール”と称します。

本 FIT モジュールがサポートしている Wi-Fi モジュールは以下です。

Silex 社製 ULPGN（SX-ULPGN-2000）

以降、Silex 社製 ULPGN（SX-ULPGN-2000）を”Wi-Fi モジュール”と称します。

なお、本 FIT モジュールでは、RTOS の機能を使用しています。RTOS と合わせて使用してください。また、MCU のシリアル通信機能を制御するデバイスドライバは含まないため、別途、以下のアップリケーションノートを取得してください。

- SCI モジュール Firmware Integration Technology
  R01AN1815

対象デバイス

RX ファミリ

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

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^2 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^2 studio スマート・コンフィグレータ ユーザーガイド(R20AN0451)
RX ファミリ SCI モジュール Firmware Integration Technology (R01AN1815)
RX ファミリ バイト型キューバッファ（BYTEQ）モジュール Firmware Integration Technology (R01AN1683)

目次

1. 概要

1.1 SX-ULPGN Wi-Fi FIT モジュールとは

本 FIT 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 FIT モジュールの組み込み方については、「2.10FIT モジュールの追加方法」を参照してください。

1.2 SX-ULPGN Wi-Fi FIT モジュールの概要

本 FIT モジュールでは SX-ULPGN の Transparent mode（1CH 通信モード）と Separate port mode（2CH 通信モード）をサポートします。

1.3 API の概要

表 1.1 に本 FIT モジュールに含まれる API 関数を示します。また、**エラー! 参照元が見つかりません。**に本 FIT モジュールで必要なメモリサイズを示します。

表 1.1 API 関数一覧

関数	関数説明
R_WIFI_SX_ULPGN_Open()	Wi-Fi モジュールの初期化を行います。
R_WIFI_SX_ULPGN_Close()	Wi-Fi モジュールをクローズします。
R_WIFI_SX_ULPGN_SetDnsServerAddress()	DNS サーバアドレスを設定します。
R_WIFI_SX_ULPGN_Scan()	アクセスポイントの一覧を取得します。
R_WIFI_SX_ULPGN_Connect()	アクセスポイントへ接続します。
R_WIFI_SX_ULPGN_Disconnect()	アクセスポイントからクローズします。
R_WIFI_SX_ULPGN_IsConnected()	アクセスポイントへの接続状態を取得します。
R_WIFI_SX_ULPGN_GetMACaddress()	Wi-Fi モジュールの MAC アドレスを取得します。
R_WIFI_SX_ULPGN_GetIPaddress()	Wi-Fi モジュールの IP アドレスを取得します。
R_WIFI_SX_ULPGN_CreateSocket()	ソケットを作成します。
R_WIFI_SX_ULPGN_ConnectSocketct()	ソケット通信を開始します。
R_WIFI_SX_ULPGN_SendSocket()	ソケットのデータ送信を行います。
R_WIFI_SX_ULPGN_ReceiveSocket()	ソケットのデータ受信を行います。
R_WIFI_SX_ULPGN_ShutdownSocket()	ソケット通信を終了します。
R_WIFI_SX_ULPGN_CloseSocket()	ソケットをクローズします。
R_WIFI_SX_ULPGN_DnsQuery()	DNS クエリを実行します。
R_WIFI_SX_ULPGN_Ping()	指定した IP アドレスに Ping を送信します。
R_WIFI_SX_ULPGN_GetVersion()	本モジュールのバージョン情報を返します。
R_WIFI_SX_ULPGN_GetTcpSocketStatus()	WiFi モジュールとの接続状態を取得します。
WiFi モジュール SSL 機能使用に関する関数	
R_WIFI_SX_ULPGN_RequestTlsSocket ()	ソケット通信に SSL 使用をリクエストします。
証明書格納に関する関数	
R_WIFI_SX_ULPGN_WriteServerCertificate ()	WiFI モジュールに証明書を書き込みます。
R_WIFI_SX_ULPGN_EraseServerCertificate ()	WiFI モジュールに格納されている証明書を消去します。
R_WIFI_SX_ULPGN_GetServerCertificate()	WiFI モジュールに格納されている証明書情報を取得します。
R_WIFI_SX_ULPGN_EraseAllCertificate()	WiFI モジュールに格納されているすべての証明書を消去します。

R_WIFI_SX_ULPGN_SetCertificateProfile()	WiFi モジュールに格納する証明書とサーバ情報の紐づけを行います。

処理例

1.3.1　ハードウェア説明

SX ULPGN の接続例を示します。

図 1.1 1CH 通信モード、図 1.2 2CH 通信モード時の接続になります。

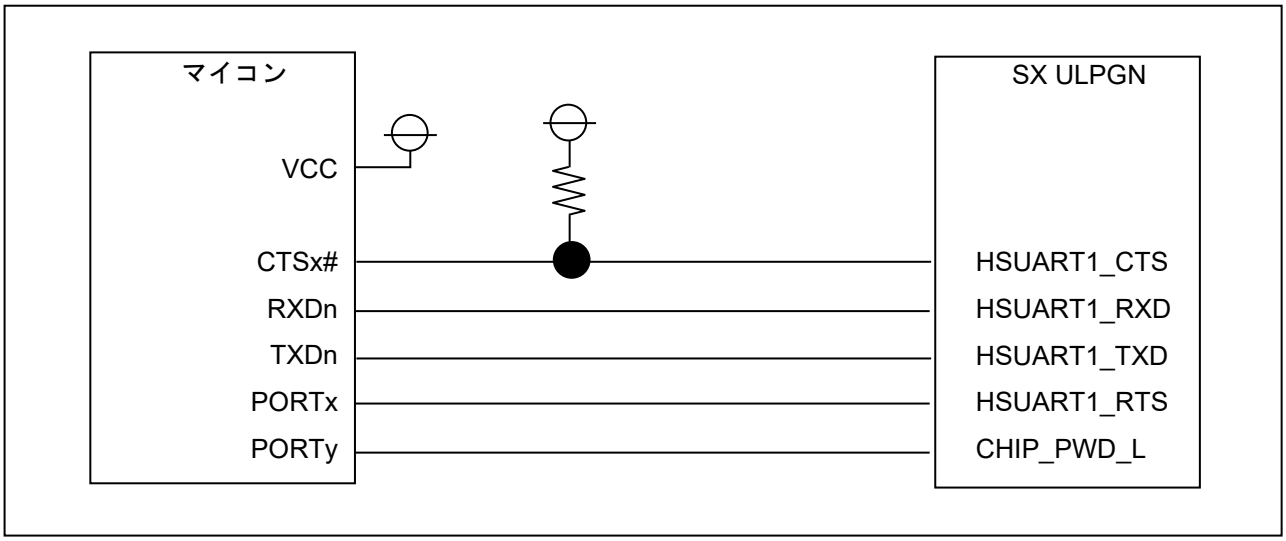

図 1.1　1CH 通信モード時の接続例

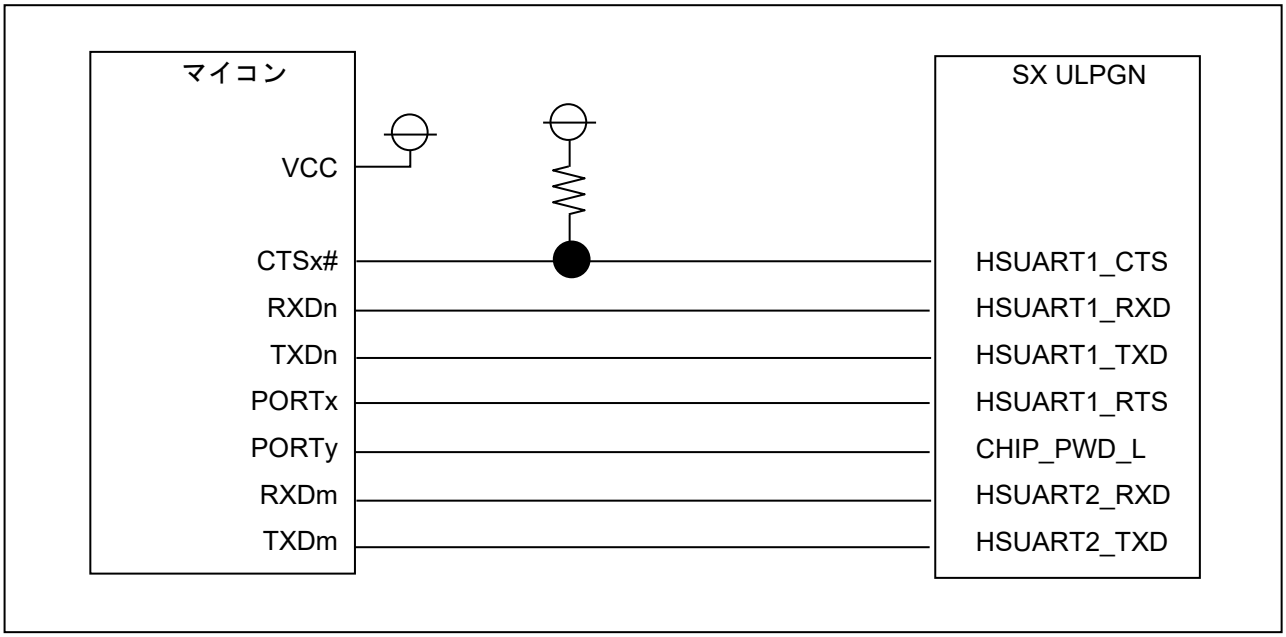

図 1.2　2CH 通信モード時の接続例

1.3.2 ソフトウェア説明

図 1.3 にソフトウェア構成を示します。

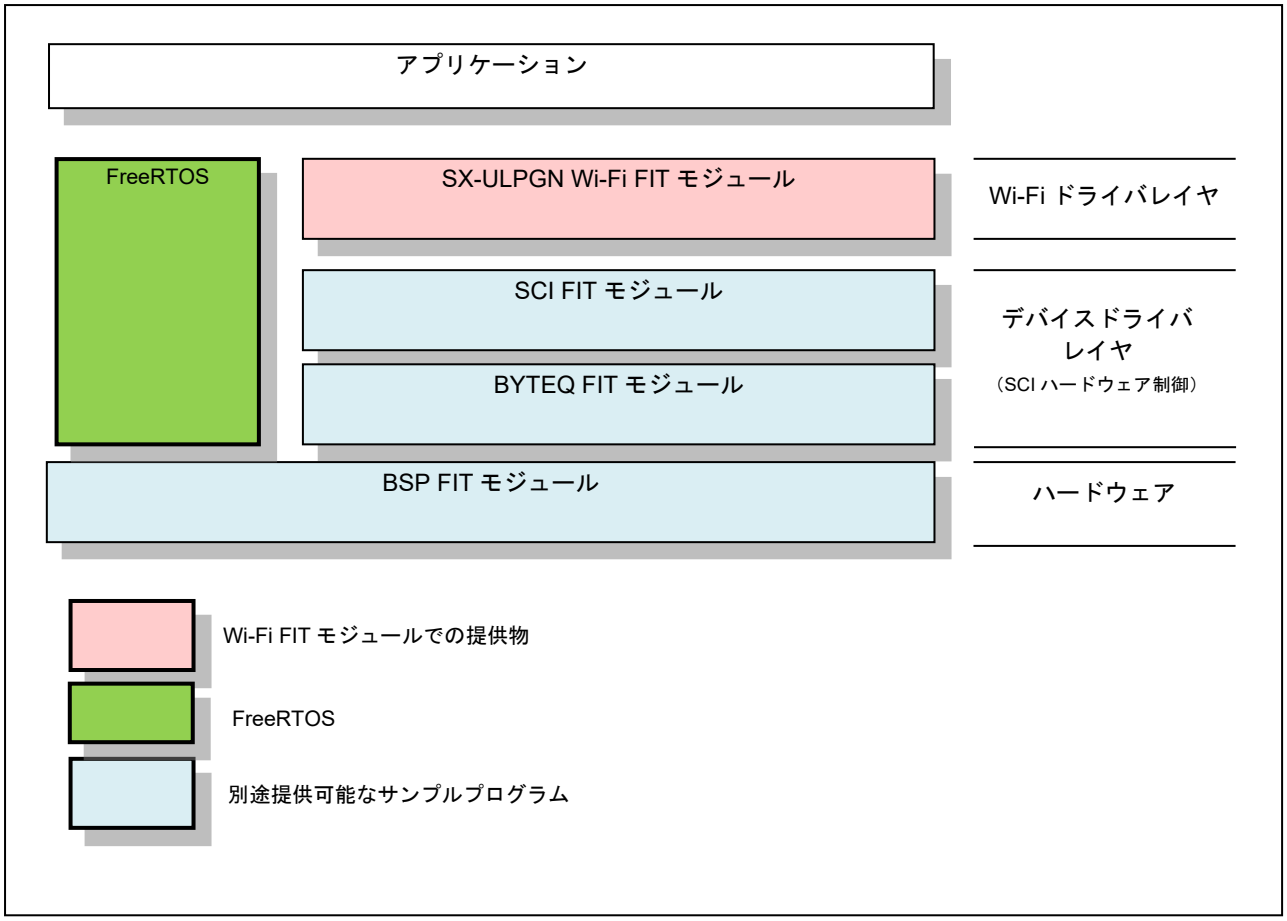

図 1.3 アプリケーション構成図

(1) SX-ULPGN Wi-Fi FIT モジュール

本 FIT モジュールです。Wi-Fi モジュールを制御するために使用するソフトウェアです。

(2) SCI FIT モジュール

Wi-Fi モジュールと MCU 間の通信を行います。サンプルプログラムが入手可能です。先頭ページの「関連ドキュメント」を参照し、入手してください。

(3) 周辺機能モジュール

タイマ制御とバッファ管理を行うソフトウェアです。サンプルプログラムが入手可能です。先頭ページの「関連ドキュメント」を参照し、入手してください。

(4) RTOS

RTOS がシステム全体を管理します。本 FIT モジュールでは Amazon FreeRTOS を使った動作を保証しています。

1.4 状態遷移図

図 1.4 に本 FIT モジュールの通信状態までの状態遷移図を示します。

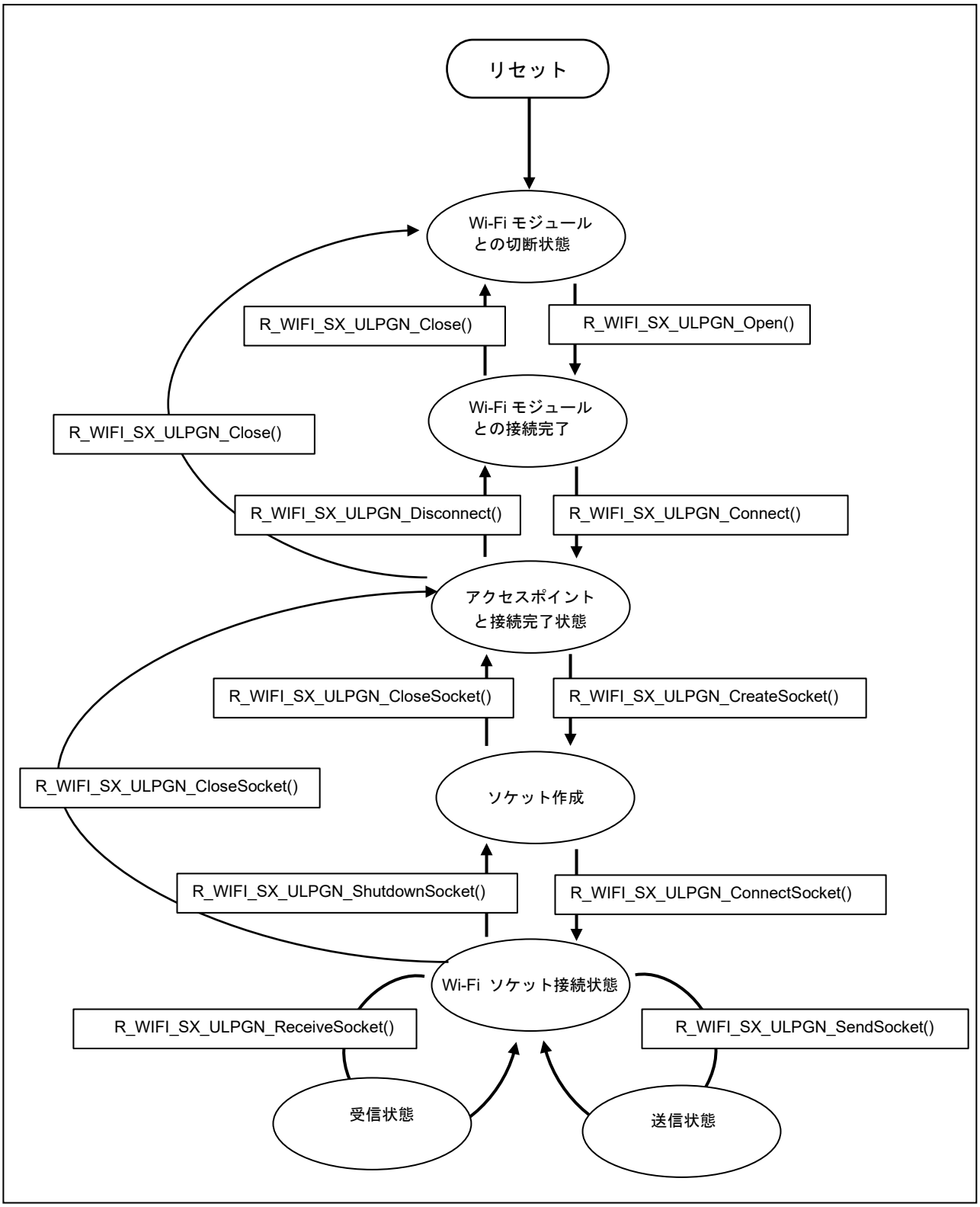

図 1.4 状態遷移図

2. API 情報

本 FIT モジュールは、下記条件で動作を確認しています。

2.1 ハードウェアの要求

ご使用になるマイコンが以下の機能をサポートしている必要があります。

- シリアル通信
- I/O ポート

2.2 ソフトウェアの要求

このドライバは以下のパッケージに依存しています。

- r_bsp
- r_sci_rx
- r_byteq_rx
- FreeRTOS

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 5.1 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

なし

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_wifi_sx_ulpgn_if.h に記載しています。

2.6 整数型

Wi-Fi FIT モジュールは 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 FIT 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_wifi_sx_ulpgn_config.h と r_sci_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。

表 2.1 Configuration options(r_wifi_sx_ulpgn_config.h)

Configuration options in r_wifi_sx_ulpgn_config.h	
WIFI_CFG_SCI_CHANNEL ※デフォルトは“0”	SX-ULPGN の HSUART1 と通信をする SCI ポート番号を指定します。 デフォルト値は SCI ポート番号 0 を使用する場合です。制御する SCI ポートに合わせて設定してください。
WIFI_CFG_SCI_SECOND_CHANNEL ※デフォルトは“1”	SX-ULPGN の HSUART2 と通信をする SCI ポート番号を指定します。 デフォルト値は SCI ポート番号 1 を使用する場合です。制御 SCI ポートに合わせて設定してください。
WIFI_CFG_RTS_PORT ※デフォルトは“2”	SX-ULPGN の RTS 端子を制御する汎用ポートの PDR(ポート方向レジスタ)を設定します。デフォルト値はポート 22 を使用する場合です。制御するポートに合わせて設定してください。
WIFI_CFG_ RTS_PIN ※デフォルトは“2”	SX-ULPGN の RTS 端子を制御する汎用ポートの PODR(ポート出力データレジスタ)を設定します。デフォルト値はポート 22 を使用する場合です。制御するポートに合わせて設定してください。
WIFI_CFG_RESET_PORT ※デフォルトは“D”	SX-ULPGN の PWD_L 端子を制御する汎用ポートの PDR(ポート方向レジスタ)を設定します。デフォルト値はポート D0 を使用する場合です。制御するポートに合わせて設定してください。
WIFI_CFG_RESET_PIN ※デフォルトは“0”	SX-ULPGN の PWD_L 端子を制御する汎用ポートの PODR(ポート出力データレジスタ)を設定します。デフォルト値はポート D0 を使用する場合です。制御するポートに合わせて設定してください。
WIFI_CFG_CREATABLE_SOCKETS ※デフォルトは“4”	SX-ULPGN の作成可能なソケット数を設定します。デフォルト値は"4"です。SX_ULPGN のファームウェア仕様に合わせて設定してください。
WIFI_CFG_SOCKETS_RECEIVE_BUFFER_SIZE ※デフォルトは“8192”	ソケットの受信バッファサイズを設定します。デフォルト値は"8192"です。メモリ使用量、データ受信量に合わせて設定してください。
WIFI_CFG_SCI_INTERRUPT_LEVEL ※デフォルトは“14”	SX-ULPGN と通信を行うシリアルモジュールの割り込み優先度を設定します。デフォルト値は”14”です。システムの優先度に合わせて設定してください。
WIFI_CFG_SCI_BAUDRATE ※デフォルトは“460800”	SX-ULPGN と通信を行うシリアルポートのボーレート（bps）を設定します。デフォルト値は”460800”です。システムに合わせて設定してください。
WIFI_CFG_USE_CALLBACK_FUNCTION ※デフォルトは“0”	コールバック関数を登録するか否かを設定します。 1=有効、0=無効
WIFI_CFG_CALLBACK_FUNCTION_NAME ※デフォルトは“NULL”	(WIFI_CFG_USE_ERROR_REPORT_FUNCTION が 0 の場合は設定不要です。) コールバック関数名を登録します。 コールバック関数はユーザが実装する必要があります。 詳細は 4 章を参照してください。

表 2.2 Configuration options(r_sci_rx_config.h)

Configuration options in r_sci_rx_config.h	
define SCI_CFG_CHx_INCLUDED ※1. CHx = CH0～CH12 ※2. 各デフォルト値は以下のとおり: CH0、CH2～CH12: 0、CH1: 1	チャネルごとに送受信バッファ、カウンタ、割り込み、その他のプログラム、RAM などのリソースを持ちます。このオプションを“1”に設定すると、そのチャンネルに関連したリソースが割り当てられます。
#define SCI_CFG_CHx_TX_BUFSIZ ※1. CHx = CH0～CH12 ※2. 各デフォルト値は(80)	チャネルごとの送信バッファサイズを指定します。 WIFI_CFG_SCI_CHANNEL で指定したチャネルに対応するバッファサイズを 2048 にしてください。
#define SCI_CFG_CHx_RX_BUFSIZ ※1. CHx = CH0～CH12 ※2. 各デフォルト値は(80)	チャネルごとの受信バッファサイズを指定します。 WIFI_CFG_SCI_CHANNEL で指定したチャネルに対応するバッファサイズを 2048 にしてください。
#define SCI_CFG_TEI_INCLUDED ※デフォルト値は″0″	シリアル送信の送信完了割り込みを有効にします。”1”を設定してください。

表 2.3 Configuration options(r_byteq_config.h)

Configuration options in r_byteq_config.h	
#define BYTEQ_CFG_MAX_CTRL_BLKS	WIFI_CFG_CREATABLE_SOCKETS で指定した値を加算してください。

表 2.4 Configuration options(r_bsp_config.h)

Configuration options in r_byteq_config.h	
#define BSP_CFG_RTOS_USED ※デフォルト値は″0″	リアルタイム OS の種類を選択します。 本 FIT モジュールを使用する場合、”1”を設定してください。

2.8 コードサイズ

本 FIT モジュールのコードサイズを下表に示します。

表 2.5 コードサイズ

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
RX65N	ROM	14199 バイト	
	RAM	4826 バイト	
	最大使用スタックサイズ	256 バイト	多重割り込み禁止のため 1 チャネル使用時の最大値のみを記載しています。

2.9 戻り値

以下は API 関数が返すエラーコードです。戻り値の列挙型は、API 関数の宣言とともに r_wifi_sx_ulpgn_if.h に含まれます。

```
typedef enum                                    // Wi-Fi API エラーコード
{
    WIFI_SUCCESS            = 0,        // 成功
    WIFI_ERR_PARAMETER      = -1,       // 引数が無効です。
    WIFI_ERR_ALREADY_OPEN   = -2,       // すでに初期化済みです
    WIFI_ERR_NOT_OPEN       = -3,       // 初期化していません
    WIFI_ERR_SERIAL_OPEN    = -4,       // シリアルの初期化ができません。
    WIFI_ERR_MODULE_COM     = -5,       // WiFiモジュールとの通信に失敗しました。
    WIFI_ERR_NOT_CONNECT    = -6,       // アクセスポイントに接続していません。
    WIFI_ERR_SOCKET_NUM     = -7,       // 利用可能なソケットがありません。
    WIFI_ERR_SOCKET_CREATE  = -8,       // ソケットを作成できません。
    WIFI_ERR_CHANGE_SOCKET  = -9,       // ソケットを切り替えられません。
    WIFI_ERR_SOCKET_CONNECT = -10,      // ソケットに接続できません。
    WIFI_ERR_BYTEQ_OPEN     = -11,      // BYTEQの割り当てに失敗しました。
    WIFI_ERR_SOCKET_TIMEOUT = -12,      // ソケットの送信でタイムアウトしました。
    WIFI_ERR_TAKE_MUTEX     = -13,      // Mutexの取得に失敗しました。
     } wifi_err_t;
```

“security”の構造体を以下に示します。

```
typedef enum
{
   WIFI_SECURITY_OPEN=0,    //セキュリティタイプ OPEN
   WIFI_SECURITY_WEP,       //セキュリティタイプ WEP
   WIFI_SECURITY_WPA,       //セキュリティタイプ WPA
   WIFI_SECURITY_WPA2,      //セキュリティタイプ WPA2
   WIFI_SECURITY_UNDEFINED,  //未定義
} wifi_security_t;

typedef enum
{
    WIFI_SOCKET_STATUS_CLOSED=0,          //クローズ状態
    WIFI_SOCKET_STATUS_SOCKET,            //作成完了状態
    WIFI_SOCKET_STATUS_BOUND,             //BOUND 状態
    WIFI_SOCKET_STATUS_LISTEN,            //LISTEN 状態
    WIFI_SOCKET_STATUS_CONNECTED,         //コネクト状態
    WIFI_SOCKET_STATUS_MAX,               //のステータス数の上限数
} wifi_socket_status_t;
```

```
typedef struct
{
   uint8_t ssid[ 33 ];   //SSID の格納領域
   uint8_t bssid[ 6 ];      //BSSID の格納領域
   wifi_security_t security;      //セキュリティタイプの格納領域
   int8_t rssi;       //信号強度の格納領域
   int8_t channel;       //チャンネル番号の格納領域
   uint8_t hidden;       //Hidden チャンネルの格納領域
} wifi_scan_result_t;


typedef struct
{
   uint32_t ipadress;    //IP アドレス
   uint32_t subnetmask; //サブネットマスク
   uint32_t gateway;     //ゲートウェイ
} wifi_ip_configuration_t;
```

```
typedef enum
{
        WIFI_EVENT_WIFI_REBOOT = 0,
        WIFI_EVENT_WIFI_DISCONNECT,
        WIFI_EVENT_SERIAL_OVF_ERR,
        WIFI_EVENT_SERIAL_FLM_ERR,
        WIFI_EVENT_SERIAL_RXQ_OVF_ERR,
        WIFI_EVENT_RCV_TASK_RXB_OVF_ERR,
        WIFI_EVENT_SOCKET_CLOSED,
        WIFI_EVENT_SOCKET_RXQ_OVF_ERR,
} wifi_err_event_enum_t;


typedef struct
{
        wifi_err_event_enum_t event,
        uint32_t socket_number
} wifi_err_event_t;

typedef enum
{
        ULPGN_SOCKET_STATUS_CLOSED              = 0,
        ULPGN_SOCKET_STATUS_SOCKET,
        ULPGN_SOCKET_STATUS_BOUND,
        ULPGN_SOCKET_STATUS_LISTEN,
        ULPGN_SOCKET_STATUS_CONNECTED,
        ULPGN_SOCKET_STATUS_MAX,
} sx_ulpgn_socket_status_t;;

typedef struct
{
```

```
        uint8_t     certificate_file[20],
        uint8_t     certificate_number,
        wifi_err_t error_flag,
        void        *nextr_certificate_name
} wifi_err_event_t;
```

2.10 FIT モジュールの追加方法

本 FIT 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「CS+ スマート・コンフィグレータ ユーザーガイド (R20AN0470)」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。

2.11 RTOS の使用要件

本 FIT モジュールでは RTOS の機能を使用しています。

2.12 制限事項

本 FIT モジュールには以下の制限があります。

- WIFI_ERR_SERIAL_OPEN が発生した際は R_WIFI_SX_ULPGN_Close()で Wi-Fi FIT モジュールをクローズしてください。
- R_WIFI_SX_ULPGN_WriteServerCertificate()でエラーが発生した際は R_WIFI_SX_ULPGN_EraseAllCertificate()で Wifi モジュールに格納された証明書をすべて削除した後、再度 R_WIFI_SX_ULPGN_WriteServerCertificate()で証明書を書き込んでください。

3. API 関数

3.1 R_WIFI_SX_ULPGN_Open()

SX-ULPGN Wi-Fi FIT モジュールと WI-FI モジュールの初期化を行う関数です。

Format

```
wifi_err_t R_WIFI_SX_ULPGN_Open (
      void
)
```

Parameters

なし

Return Values

WIFI_SUCCESS	*/* 正常終了 */*
WIFI_ERR_TAKE_MUTEX	*/* セマフォの取得に失敗 */*
WIFI_ERR_SERIAL_OPEN	*/* シリアルの初期化失敗 */*
WIFI_ERR_SOCKET_BYTEQ	/* BYTEQ の割り当てに失敗*/
WIFI_ERR_ALREADY_OPEN	*/* すでにオープン済み */*
WIFI_ERR_MODULE_COM	*/* WI-FI モジュールとの通信に失敗 */*

Properties

r_wifi_sx_ulpgn_if.h にプロトタイプ宣言されています。

Description

SX-ULPGN Wi-Fi FIT モジュールの初期化と接続された WI-FI モジュールの初期化を実行します。

SX ULPGN：1CH 通信モード、2CH 通信モードの決定をします。

Reentrant

不可

Example

```
source code
```

Special Notes:

なし

3.2 R_WIFI_SX_ULPGN_Close()

アクセスポイントとの切断、WI-FI モジュールとの切断をする関数です。

Format

```
wifi_err_t R_WIFI_SX_ULPGN_Close (
      void
)
```

Parameters

なし

Return Values

WIFI_SUCCESS */* 正常終了 */*

Properties

r_wifi_sx_ulpgn_if.h にプロトタイプ宣言されています。

Description

アクセスポイントから切断し、WI-FI モジュールとの通信も切断します。

Reentrant

- 不可

Example

```
source code
```

Special Notes:

なし

3.3 R_WIFI_SX_ULPGN_SetDnsServerAddress()

この関数は、DNS サーバの IP アドレスを設定します。

Format

```
wifi_err_t R_WIFI_SX_ULPGN_SetDnsServerAddress (
      uint32_t dns_address1,
      uint32_t dns_address2,
)
```

Parameters

dns_address1

一つ目の DNS サーバ IP アドレス

dns_address2

二つ目の DNS サーバ IP アドレス

Return Values

WIFI_SUCCESS	*/* 正常終了 */*
WIFI_ERR_NOT_OPEN	*/* WiFi モジュールが初期化されていない */*
WIFI_ERR_TAKE_MUTEX	*/* セマフォの取得に失敗 */*
WIFI_ERR_MODULE_COM	*/* WI-FI モジュールとの通信に失敗 */*

Description

dnsaddress1 に 0 以外、かつ dnsaddress2 に 0 を指定した場合

dnsaddress1 で指定されたアドレスを DNS サーバアドレスとします。

dnsaddress1 に 0 以外、かつ dnsaddress2 に 0 以外を指定した場合

dnsaddress1 と dnsaddress2 で指定されたアドレスを DNS サーバアドレスとします。

また、本関数を実行せずに R_WIFI_SX_ULPGN_Connect()を実行してアクセスポイントに接続した場合の DNS サーバアドレスは

R_WIFI_SX_ULPGN_Connect()の auto_ip_assign=0 の場合、設定したゲートウェイアドレスを使用します。

R_WIFI_ SX_ULPGN_Connect()の auto_ip_assign=1 の場合、DHCP サーバから割り当てられた DNS サーバアドレスを使用します。

本関数は、R_WIFI_SX_ULPGN_Open()実行後、R_WIFI_SX_ULPGN_Connect()を実行する前に呼び出してください。

Properties

r_wifi_sx_ulpgn_if.h にプロトタイプ宣言されています。

3.4 R_WIFI_SX_ULPGN_Scan()

この関数は、アクセスポイントをスキャンする関数です。

Format

```
wifi_err_t R_WIFI_SX_ULPGN_Scan (
      wifi_scan_result_t *ap_results,
      uint32_t max_networks、
      uint32_t *exist_ap_count
)
```

Parameters

*ap_results

スキャン結果を格納する wifi_scan_result_t 型配列の先頭アドレスを指すポインタ

max_networks

ap_results に格納可能な個数

exist_ap_count

存在するアクセスポイントの個数

Return Values

WIFI_SUCCESS	*/* 正常終了 */*
WIFI_ERR_PARAMETER	*/* 不正な引数 */*
WIFI_ERR_NOT_OPEN	*/* WiFi モジュールが初期化されていない */*
WIFI_ERR_MODULE_COM	*/* WI-FI モジュールとの通信に失敗 */*

Properties

r_wifi_sx_ulpgn_if.h にプロトタイプ宣言されています。

Description

WiFI モジュール周辺に存在するアクセスポイントをスキャンします。

スキャンした結果は ap_results 引数で指定された領域に最大 maxnetworks 引数の個数分を格納します。

また検出できたアクセスポイントの個数を exist_ap_count に通知します。

Example

```
#define AP_LIST_BUFFER_COUNT 10
wifi_scan_result_t ap_result_buffer[AP_LIST_BUFFER_COUNT];
uint32_t exist_ap;
wifi_err_t err;

err = R_WIFI_SX_ULPGN_Scan (
      ap_result_buffer, AP_LIST_BUFFER_COUNT, &exist_ap);
```

3.5 R_WIFI_SX_ULPGN_Connect()

この関数は、WI-FI モジュールをアクセスポイントに接続します。

Format

```
wifi_err_t R_WIFI_SX_ULPGN_Connect (
      uint8_t *ssid,
      uint8_t *pass,
      uint32_t security,
      uint8_t dhcp_enable
      wifi_ip_configuration_t *ip_config

)
```

Parameters

*ssid

アクセスポイントの SSID のポインタ

*pass

アクセスポイントのパスワードのポインタ

security

セキュリティタイプ情報

WIFI_SECURITY_WPA /* WPA タイプ */

WIFI_SECURITY_WPA2 /* WPA2 タイプ */

dhcp_enable

IP アドレス自動割り当て

0: 無効 (固定 IP アドレスを設定します)

1: 有効 (アクセスポイントから IP アドレスを自動割り当てします)

ip_config

auto_ip_assign が 0 の場合、ip_config に指定した IP アドレス情報を Wi-FI モジュールに設定します。

auto_ip_assign が 1 の場合、自動割り当てされた IP アドレス情報を ip_config に格納します。

Return Values

WIFI_SUCCESS	*/* 正常終了 */*
WIFI_ERR_NOT_OPEN	*/* WiFi モジュールが初期化されていない */*
WIFI_ERR_PARAMETER	*/* 不正な引数 */*
WIFI_ERR_TAKE_MUTEX	*/* セマフォの取得に失敗 */*
WIFI_ERR_MODULE_COM	*/* WI-FI モジュールとの通信に失敗 */*

Properties

r_wifi_sx_ulpgn_if.h にプロトタイプ宣言されています。

Description

pssid で指定したアクセスポイントへ接続します。

DHCP 有効を指定した場合は、アクセスポイントへの接続が成功したあと、IP アドレス割り当てを待ちます。

Reentrant

- 不可

Example

DHCP 有効の場合：

```
wifi_err_t err;
wifi_ip_configuration_t ipconfig;

R_WIFI_SX_ULPGN_Connect (
      “test_SSID”, “test_password”, WIFI_SECURITY_WPA2, 1, &ip_config);
```

DHCP 無効の場合：

```
wifi_err_t err;
wifi_ip_configuration_t ip_config;

ip_config.ipaddr = 0xc0a80003;  //192.168.0.3
ip_config.subnetmask = 0xffffff00;   //255.255.255.0
ip_config.gateway = 0xc0a80001;   //192.168.0.1

R_WIFI_SX_ULPGN_Connect (
      “test_SSID”, “test_password”, WIFI_SECURITY_WPA2, 0, &ip_config);
```

Special Notes:

なし

3.6 R_WIFI_SX_ULPGN_Disconnect ()

この関数は、WI-FI モジュールとアクセスポイントの接続を切断する関数です。

Format

```
wifi_err_t R_WIFI_SX_ULPGN_Disconnect (
      void
)
```

Parameters

なし

Return Values

WIFI_SUCCESS	*/* 正常終了 */*
WIFI_ERR_NOT_OPEN	*/* WiFi モジュールが初期化されていない */*
WIFI_ERR_TAKE_MUTEX	*/* セマフォの取得に失敗 */*

Properties

r_wifi_sx_ulpgn_if.h にプロトタイプ宣言されています。

Description

WI-FI モジュールとアクセスポイントを切断します。

Reentrant

- 不可

Example

```
source code
```

Special Notes:

なし

3.7 R_WIFI_SX_ULPGN_IsConnected()

この関数は、WI-FI モジュールとアクセスポイントの接続状態を取得します。

Format

```
int32_t R_WIFI_SX_ULPGN_IsConnected (
      void
)
```

Parameters

なし

Return Values

0 */* 接続している */*

-1 */* 接続していない */*

Properties

r_wifi_sx_ulpgn_if.h にプロトタイプ宣言されています。

Description

WI-FI モジュールとアクセスポイントの接続状態を返します。

Reentrant

- 不可

Example

```
source code
```

Special Notes:

なし

3.8 R_WIFI_SX_ULPGN_GetMacAddress ()

この関数は、WI-FI モジュールの MAC アドレス値を取得する関数です。

Format

```
wifi_err_t R_WIFI_SX_ULPGN_GetMacAddress (
      uint8_t *mac_address
)
```

Parameters

*mac_address

WI-FI モジュールの MAC アドレスの格納領域へのポインタ（6 バイト）

Return Values

WIFI_SUCCESS	*/* 正常終了 */*
WIFI_ERR_NOT_OPEN	*/* WiFi モジュールが初期化されていない */*
WIFI_ERR_TAKE_MUTEX	*/* セマフォの取得に失敗 */*
WIFI_ERR_PARAMETER	*/* 不正な引数 */*
WIFI_ERR_MODULE_COM	*/* WI-FI モジュールとの通信に失敗 */*

Properties

r_wifi_sx_ulpgn_if.h にプロトタイプ宣言されています。

Description

WI-FI モジュールの MAC アドレス値を取得します。MAC アドレスは mac_address にバイナリデータで格納します。

例

MAC アドレス 11:22:33:44:55:66

mac_address[0] = 0x11, mac_address [1] = 0x22, mac_address [3] = 0x33, …, mac_address [5] = 0x66

Reentrant

- 不可

Example

```
source code
```

Special Notes:

なし

3.9 R_WIFI_SX_ULPGN_GetIpAddress()

この関数は、インターネットサーバから *WI-FI モジュール*の IP アドレスを取得する関数です。

Format

```
wifi_err_t R_WIFI_SX_ULPGN_GetIpAddress (
      wifi_ip_configuration_t *ip_config
)
```

Parameters

* ip_config

IP アドレス格納領域へのポインタ

Return Values

WIFI_SUCCESS	*/* 正常終了 */*
WIFI_ERR_NOT_OPEN	*/* WiFi モジュールが初期化されていない */*
WIFI_ERR_TAKE_MUTEX	*/* セマフォの取得に失敗 */*
WIFI_ERR_PARAMETER	*/* 不正な引数 */*
WIFI_ERR_MODULE_COM	*/* WI-FI モジュールとの通信に失敗 */*

Properties

r_wifi_sx_ulpgn_if.h にプロトタイプ宣言されています。

Description

取得した IP アドレス、サブネットマスク、ゲートウェイアドレスを ip_config に格納します。

例：IP アドレスが 192.168.0.3、サブネットマスクが 255.255.255.0、ゲートウェイが 192.168.0.1 の場合

ip_config ->ipaddr = 0xc0a80003

ip_config ->subnetmask = 0xffffff00

ip_config ->gateway = 0xc0a80001

Reentrant

- 不可

Example

```
source code
```

Special Notes:

なし

3.10 R_WIFI_SX_ULPGN_CreateSocket ()

この関数は、利用可能なソケットに対してソケットタイプ、IP タイプを設定する関数です。

Format

```
int32_t R_WIFI_SX_ULPGN_CreateSocket (
      uint32_t type,
      uint32_t ip_version,
)
```

Parameters

type

ソケットタイプ

WIFI_SOCKET_IP_PROTOCOL /* TCP */

ip_version

IP バージョン

WIFI_SOCKET_IP_VERSION_4 /* IPv4 */

Return Values

正の値	*/* 正常終了（作成したソケットの番号） */*
WIFI_ERR_PARAMETER	*/* 不正な引数 */*
WIFI_ERR_NOT_CONNECT,	*/* アクセスポイントに未接続 */*
WIFI_ERR_SOCKET_CREATE,	*/* ソケットの作成失敗 */*

Properties

r_wifi_sx_ulpgn_if.h にプロトタイプ宣言されています。

Description

ソケットを生成し、ソケット番号を戻り値に返します。設定されるソケット番号は 0 から 3 までの整数値です。

Reentrant

- 不可

Example

```
source code
```

3.11 R_WIFI_SX_ULPGN_ConnectSocket ()

この関数は、指定したアドレスに接続する関数です。

Format

```
wifi_err_t R_WIFI_SX_ULPGN_ConnectSocket (
      int32_t socket_number,
      uint32_t ip_address,
      uint16_t port,
      char    *destination,
)
```

Parameters

socket_number

ソケット番号

ip_address

通信相手の IP アドレス

port

通信相手のポート番号

destination

通信相手のサーバ名

Return Values

WIFI_SUCCESS	*/* 正常終了 */*
WIFI_ERR_PARAMETER	*/* 不正な引数 */*
WIFI_ERR_SOCKET_NUM,	*/* 利用可能なソケット無し */*
WIFI_ERR_TAKE_MUTEX	*/* セマフォの取得失敗 */*
WIFI_ERR_MODULE_COM	*/* WI-FI モジュールとの通信に失敗 */*
WIFI_ERR_NOT_CONNECT,	*/* アクセスポイントに未接続 */*

Properties

r_wifi_sx_ulpgn_if.h にプロトタイプ宣言されています。

Description

指定した socket_number のソケットを利用して指定した IP アドレス、ポート番号に接続します。本 API を使用する前に R_WIFI_SX_SocketCreate()を実行して利用するソケットの設定をしてください。
R_WIFI_SX_SocketCreate()を実行していない場合 WIFI_ERR_SOCKET_NUM を返します。
SSL 通信が有効になっている場合は、SSL に関する設定を行います。SSL を有効にする場合は本 API を使用する前に R_WIFI_SX_ULPGN_SocketOptRequireTls()を実行して SSL 通信を有効に設定し、利用するソケットに対する証明書の設定を行ってください。R_WIFI_SX_ULPGN_SocketOptRequireTls()を実行していない場合は、非 SSL 通信を行います。

Reentrant

- 不可

Example

```
source code
```

Special Notes:

なし

3.12 R_WIFI_SX_ULPGN_SendSocket ()

この関数は、指定したソケットでデータ送信をする関数です。

Format

```
int32_t R_WIFI_SX_ULPGN_SendSocket (
      uint8_t socket_number,
      const uint8_t *data,
      uint32_t length,
      uint32_t timeout_ms,
)
```

Parameters

socket_number

ソケット番号

*data

送信データ格納領域へのポインタ

length

送信したいデータバイト数

timeout_ms（未使用）

送信のタイムアウト時間 [ms]

Return Values

正の値	*/* 正常終了(送信完了したバイト数) */*
WIFI_ERR_PARAMETER	*/* 不正な引数 */*
WIFI_ERR_SOCKET_NUM,	*/* 利用可能なソケット無し */*
WIFI_ERR_NOT_CONNECT,	*/* アクセスポイントに未接続 */*
WIFI_ERR_TAKE_MUTEX	*/* セマフォの取得失敗 */*
WIFI_ERR_CHANGE_SOCKET	*/* ソケット切り替えの失敗 */*
WIFI_ERR_MODULE_COM	*/* WI-FI モジュールとの通信に失敗 */*

Properties

r_wifi_sx_ulpgn_if.h にプロトタイプ宣言されています。

Description

指定したソケットから data に格納されたデータを length で指定したバイト数を送信します。

本 API を使用する前に R_WIFI_SX_ULPGN_SocketConnect()を実行する必要があります。

WIFI_ERR_SOCKET_CONNECT は R_WIFI_SX_ULPGN_SocketConnect()を実行しておらずソケットが接続していないときに返されます。

Reentrant

- 不可

Example

source code

Special Notes:

なし

3.13 R_WIFI_SX_ULPGN_ReceiveSocket ()

この関数は、指定したソケットからデータを受信する関数です。

Format

```
int32_t R_WIFI_SX_ULPGN_ReceiveSocket (
      int32_t socket_number,
      uint8_t *data,
      int32_t length,
      uint32_t timeout_ms
)
```

Parameters

socket_number

ソケット番号

*data

受信データ格納領域へのポインタ

data_length

受信したいデータバイト数

timeout_ms

受信のタイムアウト時間 [ms]

0 設定時はタイムアウトなし

Return Values

正の値	/* 正常終了(受信完了したバイト数) */
WIFI_ERR_PARAMETER	/* 不正な引数 */
WIFI_ERR_NOT_CONNECT,	/* アクセスポイントに未接続 */
WIFI_ERR_SOCKET_NUM	/* 利用可能なソケット無し */
WIFI_ERR_TAKE_MUTEX	/* セマフォの取得失敗 */
WIFI_ERR_CHANGE_SOCKET	/* ソケット切り替えの失敗 */
WIFI_ERR_MODULE_COM	/* WI-FI モジュールとの通信に失敗 */

Properties

r_wifi_sx_ulpgn_if.h にプロトタイプ宣言されています。

Description

指定したソケットから受信格納領域 data へ length で指定した受信バイト数だけデータを取得します。timeout_ms で設定した時間以内に length で指定したサイズのデータが取得できない場合、受信完了したデータサイズを返します。

Reentrant

- 不可

Example

```
source code
```

Special Notes:

なし

3.14 R_WIFI_SX_ULPGN_ShutdownSocket ()

この関数は、指定したソケットの通信を切断します。

Format

```
wifi_err_t R_WIFI_SX_ULPGN_ShutdownSocket (
      int32_t socket_number
)
```

Parameters

socket_number

ソケット番号

Return Values

WIFI_SUCCESS	*/* 正常終了 */*
WIFI_ERR_NOT_CONNECT,	*/* アクセスポイントに未接続 */*
WIFI_ERR_SOCKET_NUM	*/* 利用可能なソケット無し */*
WIFI_ERR_TAKE_MUTEX	*/* セマフォの取得に失敗 */*
WIFI_ERR_CHANGE_SOCKET	*/* ソケット切り替えの失敗 */*
WIFI_ERR_MODULE_COM	*/* WI-FI モジュールとの通信に失敗 */*

Properties

r_wifi_sx_ulpgn_if.h にプロトタイプ宣言されています。

Description

指定した socket_number のソケット通信を切断します。

Reentrant

- 不可

Example

```
source code
```

Special Notes:

なし

3.15 R_WIFI_SX_ULPGN_CloseSocket ()

この関数は、指定したソケットをネットワークから切断する関数です。

Format

```
wifi_err_t R_WIFI_SX_ULPGN_CloseSocket (
      uint8_t socket_number
)
```

Parameters

socket_number

ソケット番号

Return Values

WIFI_SUCCESS	*/* 正常終了 */*
WIFI_ERR_NOT_CONNECT,	*/* アクセスポイントに未接続 */*
WIFI_ERR_SOCKET_NUM	*/* 利用可能なソケット無し */*
WIFI_ERR_TAKE_MUTEX	*/* セマフォの取得に失敗 */*
WIFI_ERR_CHANGE_SOCKET	*/* ソケット切り替えの失敗 */*
WIFI_ERR_MODULE_COM	*/* WI-FI モジュールとの通信に失敗 */*

Properties

r_wifi_sx_ulpgn_if.h にプロトタイプ宣言されています。

Description

socket_number で指定したソケットをクローズします。

Reentrant

- 不可

Example

```
source code
```

Special Notes:

なし

3.16 R_WIFI_SX_ULPGN_DnsQuery()

この関数は、DNS クエリを実行する関数です。

Format

```
wifi_err_t R_WIFI_SX_ULPGN_DnsQuery (
      const uint8_t *domain_name,
      uint32_t *ip_address
)
```

Parameters

* domain_name

ドメイン名

*ip_address

IP アドレス格納領域領域

Return Values

WIFI_SUCCESS	*/* 正常終了 */*
WIFI_ERR_NOT_CONNECT,	*/* アクセスポイントに未接続 */*
WIFI_ERR_PARAMETER	*/* 不正な引数 */*
WIFI_ERR_MODULE_COM	*/* WI-FI モジュールとの通信に失敗 または存在しないドメイン */*

Properties

r_wifi_sx_ulpgn_if.h にプロトタイプ宣言されています。

Description

DNS クエリを実行し、指定したドメインの IP アドレスを取得します。

Reentrant

- 不可

Example

```
source code
```

Special Notes:

なし

3.17 R_WIFI_SX_ULPGN_Ping()

この関数は、指定した IP アドレスに ping を送信する関数です。

Format

```
wifi_err_t R_WIFI_SX_ULPGN_Ping (
      uint32_t ip_address,
      uint16_t count,
      uint32_t interval_ms
)
```

Parameters

ip_address

IP アドレス

count

ping 送信回数

interval_ms

ping 送信間の待機時間 [ms]

Return Values

WIFI_SUCCESS	*/* 正常終了 */*
WIFI_ERR_PARAMETER	*/* 不正な引数 */*
WIFI_ERR_NOT_CONNECT	*/* アクセスポイントに未接続 */*
WIFI_ERR_MODULE_COM	*/* WI-FI モジュールとの通信に失敗 または応答なし */*
WIFI_ERR_TAKE_MUTEX	*/* Mutex の取得に失敗 */*

Properties

r_wifi_sx_ulpgn_if.h にプロトタイプ宣言されています。

Description

ip_address で指定した IP アドレスに ping を送信します。IP アドレスは、ip_address に以下のように格納します。

(例)IP アドレス ：11.22.33.44

```
ip_address = 0x0b16212c;
R_WIFI_SX_ULPGN_Ping (ip_address, 1, 1000);
```

Reentrant

- 不可

Example

```
source code
```

Special Notes:

なし

3.18 R_WIFI_SX_ULPGN_GetVersion()

この関数は、本 FIT モジュールのバージョン情報を取得します。

Format

```
uint32_t R_WIFI_SX_ULPGN_GetVersion(
      void
)
```

Parameters

なし

Return Values

バージョン番号

Properties

r_wifi_sx_ulpgn_if.h にプロトタイプ宣言されています。

Description

この関数は本 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。

Reentrant

- 不可

Example

```
source code
```

Special Notes:

なし

3.19 R_WIFI_SX_ULPGN_GetTcpSocketStatus()

この関数は、指定したソケットの状態を取得する関数です。

Format

```
int32_t R_WIFI_SX_ULPGN_GetTcpSocketStatus (
      uint8_t socket_number
)
```

Parameters

socket_number

ソケット番号

Return Values

WIFI_SOCKET_STATUS_CLOSED=0,	*//クローズ状態*
WIFI_SOCKET_STATUS_SOCKET,	*//作成完了状態*
WIFI_SOCKET_STATUS_BOUND,	*//BOUND 状態*
WIFI_SOCKET_STATUS_LISTEN,	*//LISTEN 状態*
WIFI_SOCKET_STATUS_CONNECTED,	*//コネクト状態*
-1	*//ソケット情報取得失敗*

Properties

r_wifi_sx_ulpgn_if.h にプロトタイプ宣言されています。

Description

この関数は WiFi モジュールとの接続状態を取得します。

Reentrant

- 不可

Example

```
source code
```

Special Notes:

なし

3.20 R_WIFI_SX_ULPGN_RequestTlsSocket ()

この関数は、ソケット通信で TLS 通信の要求を行う関数です。

Format

```
wifi_err_t R_WIFI_SX_ULPGN_RequestTlsSocket
      int32_t socket_number,
)
```

Parameters

socket_number

Socket 番号

Return Values

WIFI_SUCCESS	*/* 正常終了 */*
WIFI_ERR_SOCKET_NUM	*/* 利用可能なソケット無し */*
WIFI_ERR_NOT_CONNECT,	*/* アクセスポイントに未接続 */*

Properties

r_wifi_sx_ulpgn_if.h にプロトタイプ宣言されています。

Description

この関数はソケット通信で TLS 通信の要求を行います。

R_WIFI_SX_ULPGN_CreateSocket()を実行後、RX_WIFI_SX_ULPGN_ConnectSocket()を実行前に呼びだしてください。

Reentrant

- 不可

Example

ソケット番号 0 に TLS 通信を要求し、ID コード 0 の証明書を割り当てる。

source code

```
R_WIFI_SX_ULPGN_RequestTlsSocket (0);
```

Special Notes:

なし

3.21 R_WIFI_SX_ULPGN_WriteServerCertificate()

この関数は、WiFi モジュールに証明書を格納する関数です。

Format

```
wifi_err_t R_WIFI_SX_ULPGN_WriteServerCertificate
      uint32_t data_id,
      uint32_t data_type,
      uint8_t *certificate,
      uint32_t certificate_length
)
```

Parameters

data_id

証明書 ID コード (0~4)

data_type

証明書タイプ

0：certificate

1：Ca list

certificate

証明書データのポインタ

証明書を格納した変数を指定

certificate_length

証明書サイズ

証明書のサイズを指定

Return Values

WIFI_SUCCESS	*/* 正常終了 */*
WIRI_ERR_PARAMETER	*/* 証明書データが正しくセットされていない */*
WIFI_ERR_NOT_OPEN	*/* WI-FI モジュールが Open していない */*
WIFI_ERR_TAKE_MUTEX	*/* Mutex の取得に失敗 */*
WIFI_ERR_MODULE_COM	*/* WI-FI モジュールとの通信に失敗 */*

Properties

r_wifi_sx_ulpgn_if.h にプロトタイプ宣言されています。

Description

この関数は WiFI モジュールに証明書を書き込みます。

本関数を実行する前に R_WIFI_SX_ULPGN_Open()を実行しておく必要があります。

証明書ファイル名は、証明書 ID コード、証明書タイプにより以下のように設定されます。

証明書タイプ ：0 の場合 （証明書）

cert<証明書 ID＞.crt

証明書タイプ ：1 の場合 （CA リスト）

calist<証明書 ID＞.crt

証明書ファイルは WiFI モジュールに 5 セットまで格納することができます。

証明書データは SharkSSLParseCert バイナリフォーマット、CA リストは SharkSSLPerseCAList バイナリフォーマットに変換しておく必要があります。

証明書の作成、SharkSSLParseCert バイナリフォーマットへの変換、プロジェクトへの組み込み方法は 5.3 Appendix を参照してください。

Reentrant

- 不可

Example

source code

```
void prvWifiSetCertification(void)
{
	/* Get Initial Server Certificate Information */
	R_WIFI_SX_ULPGN_GetServerCertificate(wifi_certificate_information);
	R_WIFI_SX_ULPGN_EraseAllServerCertificate();
	R_WIFI_SX_ULPGN_GetServerCertificate(wifi_certificate_information);
	R_WIFI_SX_ULPGN_EraseAllServerCertificate();
	R_WIFI_SX_ULPGN_WriteServerCertificate (0,1,(uint8_t*)&sharkSslRSACert_PC,
(uint32_t)sharkSslRSACert_PCLength);
	R_WIFI_SX_ULPGN_WriteServerCertificate (0,0,(uint8_t*)&sharkSslCAList_PC,
(uint32_t)sharkSslCAList_PCLength);
	R_WIFI_SX_ULPGN_WriteServerCertificate (1,1,(uint8_t*)&sharkSslRSACert,
(uint32_t)sharkSslRSACertLength);
	R_WIFI_SX_ULPGN_WriteServerCertificate (1,0,(uint8_t*)&sharkSslCAList,
(uint32_t)sharkSslCAListLength);
	/* Get Updated Server Certificate Information */
	R_WIFI_SX_ULPGN_GetServerCertificate(wifi_certificate_information);
}
```

Special Notes:

なし

3.22 R_WIFI_SX_ULPGN_EraseServerCertificate()

この関数は、WiFi モジュールに格納されている証明書を削除する関数です。

Format

```
wifi_err_t R_WIFI_SX_ULPGN_EraseServerCertificate
      uint8_t *certificate_name
)
```

Parameters

certificate_name

証明書ファイル名のポインタ

Return Values

WIFI_SUCCESS	*/* 正常終了 */*
WIRI_ERR_PARAMETER	*/* 証明書ファイル名が正しくセットされていない */*
WIFI_ERR_TAKE_MUTEX	*/* セマフォの取得に失敗 */*
WIFI_ERR_NOT_OPEN	*/* WI-FI モジュールが Open していない */*
WIFI_ERR_MODULE_COM	*/* WI-FI モジュールとの通信に失敗 */*

Properties

r_wifi_sx_ulpgn_if.h にプロトタイプ宣言されています。

Description

この関数は、証明書ファイル名を指定し、WiFI モジュールに格納されている証明書を消去します。

本関数を実行する前に R_WIFI_SX_ULPGN_Open()を実行しておく必要があります。

WiFI モジュールに格納されている証明書は R_WIFI_SX_ULPGN_GetServerCertificate()で確認できます。

Reentrant

- 不可

Example

source code

```
void prvWifiSetCertification(void)
{
      /* Get Initial Server Certificate Information */
      R_WIFI_SX_ULPGN_GetServerCertificate(wifi_certificate_information);
      R_WIFI_SX_ULPGN_EraseAllServerCertificate();
      R_WIFI_SX_ULPGN_GetServerCertificate(wifi_certificate_information);
      R_WIFI_SX_ULPGN_EraseAllServerCertificate();
      R_WIFI_SX_ULPGN_WriteServerCertificate (0,1,(uint8_t*)&sharkSslRSACert_PC,
(uint32_t)sharkSslRSACert_PCLength);
      R_WIFI_SX_ULPGN_WriteServerCertificate (0,0,(uint8_t*)&sharkSslCAList_PC,
(uint32_t)sharkSslCAList_PCLength);
      R_WIFI_SX_ULPGN_WriteServerCertificate (1,1,(uint8_t*)&sharkSslRSACert,
(uint32_t)sharkSslRSACertLength);
      R_WIFI_SX_ULPGN_WriteServerCertificate (1,0,(uint8_t*)&sharkSslCAList,
(uint32_t)sharkSslCAListLength);
```

```
        /* Get Updated Server Certificate Information */
        R_WIFI_SX_ULPGN_GetServerCertificate(wifi_certificate_information);
}
```

Special Notes:
なし

3.23 R_WIFI_SX_ULPGN_GetServerCertificate()

この関数は、WiFi モジュールに格納されている証明書ファイル名を取得する関数です。

Format

```
wifi_err_t R_WIFI_SX_ULPGN_GetServerCertificate
      wifi_certificate_infomation_t *wifi_certificate_information
)
```

Parameters

wifi_certificate_information

証明書情報格納領域へのポインタ

Return Values

WIFI_SUCCESS	*/* 正常終了 */*
WIRI_ERR_PARAMETER	*/* 証明書ファイル名が正しくセットされていない */*
WIFI_ERR_TAKE_MUTEX	*/* Mutex の取得に失敗 */*
WIFI_ERR_MODULE_COM	*/* WI-FI モジュールとの通信に失敗 */*

Properties

r_wifi_sx_ulpgn_if.h にプロトタイプ宣言されています。

Description

この関数は WiFi モジュールに格納された証明書情報を取得し、証明書情報の先頭アドレスを wifi_certificate_information に返します。

本関数を実行する前に R_WIFI_SX_ULPGN_Open()を実行しておく必要があります。

Reentrant

- 不可

Example

source code

```
void prvWifiSetCertification(void)
{
      /* Get Initial Server Certificate Information */
      R_WIFI_SX_ULPGN_GetServerCertificate(wifi_certificate_information);
      R_WIFI_SX_ULPGN_EraseAllServerCertificate();
      R_WIFI_SX_ULPGN_GetServerCertificate(wifi_certificate_information);
      R_WIFI_SX_ULPGN_EraseAllServerCertificate();
      R_WIFI_SX_ULPGN_WriteServerCertificate (0,1,(uint8_t*)&sharkSslRSACert_PC,
(uint32_t)sharkSslRSACert_PCLength);
      R_WIFI_SX_ULPGN_WriteServerCertificate (0,0,(uint8_t*)&sharkSslCAList_PC,
(uint32_t)sharkSslCAList_PCLength);
      R_WIFI_SX_ULPGN_WriteServerCertificate (1,1,(uint8_t*)&sharkSslRSACert,
(uint32_t)sharkSslRSACertLength);
      R_WIFI_SX_ULPGN_WriteServerCertificate (1,0,(uint8_t*)&sharkSslCAList,
```

```
(uint32_t)sharkSslCAListLength);
       /* Get Updated Server Certificate Information */
       R_WIFI_SX_ULPGN_GetServerCertificate(wifi_certificate_information);
}
```

Special Notes:
なし

3.24 R_WIFI_SX_ULPGN_EraseAllServerCertificate()

この関数は、WiFi モジュールに格納されている証明書をすべて削除する関数です。

Format

```
wifi_err_t R_WIFI_SX_ULPGN_EraseAllServerCertificate
      void
)
```

Parameters

なし

Return Values

WIFI_SUCCESS	*/* 正常終了 */*
WIFI_ERR_NOT_OPEN	*/* WI-FI モジュールが Open していない */*
WIFI_ERR_TAKE_MUTEX	*/* Mutex の取得に失敗 */*
WIFI_ERR_MODULE_COM	*/* WI-FI モジュールとの通信に失敗 */*

Properties

r_wifi_sx_ulpgn_if.h にプロトタイプ宣言されています。

Description

この関数は WiFI モジュールに格納されている、すべての証明書を消去します。

本関数を実行する前に R_WIFI_SX_ULPGN_Open()を実行しておく必要があります。

Reentrant

- 不可

Example

source code

```
void prvWifiSetCertification(void)
{
	/* Get Initial Server Certificate Information */
	R_WIFI_SX_ULPGN_GetServerCertificate(wifi_certificate_information);
	R_WIFI_SX_ULPGN_EraseAllServerCertificate();
	R_WIFI_SX_ULPGN_GetServerCertificate(wifi_certificate_information);
	R_WIFI_SX_ULPGN_EraseAllServerCertificate();
	R_WIFI_SX_ULPGN_WriteServerCertificate (0,1,(uint8_t*)&sharkSslRSACert_PC,
(uint32_t)sharkSslRSACert_PCLength);
	R_WIFI_SX_ULPGN_WriteServerCertificate (0,0,(uint8_t*)&sharkSslCAList_PC,
(uint32_t)sharkSslCAList_PCLength);
	R_WIFI_SX_ULPGN_WriteServerCertificate (1,1,(uint8_t*)&sharkSslRSACert,
(uint32_t)sharkSslRSACertLength);
	R_WIFI_SX_ULPGN_WriteServerCertificate (1,0,(uint8_t*)&sharkSslCAList,
(uint32_t)sharkSslCAListLength);
	/* Get Updated Server Certificate Information */
	R_WIFI_SX_ULPGN_GetServerCertificate(wifi_certificate_information);
}
```

Special Notes:
なし

3.25 R_WIFI_SX_ULPGN_SetCertificateProfile()

この関数は、WiFI モジュールに格納した証明書とサーバ情報の紐づけを行います。

Format

```
void R_WIFI_SX_ULPGN_SetCertificateProfile
      uint8_t  certificate_id,
      uint32_t ip_address,
      char     *sever_name
)
```

Parameters

certificate_id

証明書 ID 番号

ip_address

サーバ IP アドレス

server_name

サーバ名のポインタ

Return Values

WIFI_SUCCESS

Properties

r_wifi_sx_ulpgn_if.h にプロトタイプ宣言されています。

Description

この関数は WiFI モジュールに格納した証明書とサーバ情報の紐づけを行います。

証明書 ID は必須の指定項目です。サーバ IP アドレス、サーバ名はどちらか一方を指定してください。

両方に設定がある場合はサーバ IP アドレスが優先されます。

Reentrant

- 不可

Example

証明書 ID0 に IP アドレス、証明書 1 にサーバ名を紐づけする。

source code

```
void prvSetCertificateProfile(void)
{
      uint32_t ipaddress;

      ipaddress =
SOCKETS_inet_addr_quick(tcptestECHO_SERVER_TLS_ADDR3,tcptestECHO_SERVER_TLS_ADDR2,tcpte
stECHO_SERVER_TLS_ADDR1,tcptestECHO_SERVER_TLS_ADDR0);
      R_WIFI_SX_ULPGN_SetCertificateProfile(0,ipaddress,"");
      R_WIFI_SX_ULPGN_SetCertificateProfile(1,0,(char*)clientcredentialMQTT_BROKER_ENDP
OINT);
}
```

Special Notes:
なし

4. コールバック関数

4.1 callback()

WI-FI モジュールからのイベントを通知する関数です。

Format

```
void * callback(
      void * pevent
)
```

Parameters

pevent

エラー情報の領域を指すポインタ

Return Values

なし

Properties

ユーザにて宣言してください。

Description

本 FIT モジュールでは、SCI FIT モジュールからの異常通知を受信したタイミングで、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、「2.7 コンパイル時の設定」に記載されたコンフィグレーション「WIFI_CFG_ERROR_REPORT_FUNCTION_NAME」にユーザ関数のアドレスを格納することで設定されます。関数名は”callback”である必要はありません。

コールバック関数が呼び出されるとき、wifi_err_event_t 型に示す通知内容の先頭アドレスを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以下の例を参考に void 型のポインタ変数としてください。

コールバック関数内部で引数の値を使用する際は wifi_err_event_t 型にキャストして使用してください。

wifi_err_event_t 型の event メンバに設定された値と内容を示します。

- WIFI_EVENT_SERIAL_OVF_ERR
  SCI モジュールが、受信オーバフローエラーを検出した場合に通知します。
  UART 送受信制御が正しく行えない状態に陥っています。Wi-Fi モジュールを再起動してください。
- WIFI_EVENT_SERIAL_FLM_ERR
  SCI モジュールが、受信フレーミングエラーを検出した場合に通知します。
  UART 送受信制御が正しく行えない状態に陥っています。Wi-Fi モジュールを再起動してください。
- WIFI_EVENT_SERIAL_RXQ_OVF_ERR
  SCI モジュールが、受信キュー領域に受信データを設定できないエラーを検出した場合に通知します。UART 送受信制御が正しく行えない状態に陥っています。Wi-Fi モジュールを再起動してください。
- WIFI_EVENT_RCV_TASK_RXB_OVF_ERR
  本 FIT モジュールの受信バッファに受信データを設定できないエラーを検出した場合に通知します。
- WIFI_EVENT_SOCKET_RXQ_OVF_ERR
  ソケットの受信キューに受信データを設定できないエラーを検出したことを通知します。
  ソケット番号は pevent->socket_number で示します。

Reentrant

不可

Example

```
[r_wifi_sx_ulpgn_config.h]
#define WIFI_CFG_USE_CALLBACK_FUNCTION   1
#define WIFI_CFG_CALLBACK_FUNCTION_NAME   wifi_callback

[xxx.c]
void wifi_callback(void *p_args)
{
  wifi_err_event_t *pevent;
  pevent = (wifi_err_event_t *)p_args;

  switch(pevent->event)
  {
    case WIFI_EVENT_WIFI_REBOOT:
       break;
    case WIFI_EVENT_WIFI_DISCONNECT:
       break;
    case WIFI_EVENT_SERIAL_OVF_ERR:
       break;
    case WIFI_EVENT_SERIAL_FLM_ERR:
       break;
    case WIFI_EVENT_SERIAL_RXQ_OVF_ERR:
       break;
    case WIFI_EVENT_RCV_TASK_RXB_OVF_ERR:
       break;
    case WIFI_EVENT_SOCKET_CLOSED:
       switch(pevent->socket_number)
       {
          case 0:
             break;
          case 1:
             break;
         /*  To omit  */
          case 3:
             break;
       }
       break;
    case WIFI_EVENT_SOCKET_RXQ_OVF_ERR:
       switch(pevent->socket_number)
       {
          case 0:
             break;
          case 1:
             break;
         /*  To omit  */
          case 3:
             break;
       }
       break;
  }
}
```

Special Notes:

コールバック関数内で 3.API 関数に示す関数を実行しないでください。

各 RenesasTarget Board を使用する場合の端子設定例を以下に示します。

5. 付録

5.1 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 5.1 動作確認環境（Ver.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^2 studio V7.08.00
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 V3.02.00
	コンパイルオプション：統合開発環境のデフォルト設定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	Rev1.00
使用ボード	・Renesas RX65N Cloud Kit（型名：RTK5RX65N0SxxxxxBE）

5.2 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「[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_wifi_sx_ulpgn_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_wifi_sx_ulpgn_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。

(3) Q：[端子設定していない場合発生する現象]が発生します。

A：正しく端子設定が行われていない可能性があります。本 FIT モジュールを使用する場合は端子設定が必要です。詳細は「4 端子設定」を参照してください。

5.3 Appendix（証明書データの組み込み手順）

TLS 通信を行うために WiFI モジュールに書き込む証明書作成手順を示します。

5.3.1 証明書の作成

証明書は OpenSSL を使って作成します。
ご使用の PC に OpenSSL をインストールしてください。
証明書作成手順は以下のとおりです。

- openssl genrsa –out certs/client.key 2048
- openssl req –new –key certs/client.key –out certs/client.csr ¥
  –subj “/C=JP/L=<States>/O=<Company>/OU=<Department>/CN=<Object>/email=<EmailAddress>”
- openssl x509 –req –in certs/client.csr –CA certs.server.pem –CAkey certs/server.key ¥
  –CAcreateserial –out certs/client.pem –days 365 –sha256”

5.3.2 フォーマットの変換

WiFi モジュールへ書き込みを行う証明書データは SharkSSLParseCert バイナリフォーマット、CA リストは SharkSSLPerseCAList バイナリフォーマットに変換しておく必要があります。
フォーマット変換は下記のフリーソフトウェアで行うことができます。

SharkSSL <https://realtimelogic.com/downloads/sharkssl/>

ダウンロード、インストールはソフトウェアの指示に従ってください。

証明書のフォーマット変換方法は以下の通りです。

フォーマット変換時の出力ファイルは、変換した証明書をプログラムに組み込む場合と、

PC から直接書き込みを行う場合で 2 通りの出力が行えます。

① ルート証明書（Class 2 Root CA）を入手する

② ルート証明書を SharkSSL バイナリフォーマットに変換する

（プログラムに組み込む場合）

> SharkSSLParseCAList.exe xxxx.cer > starfield.c

（PC から直接書き込みを行う場合）

> SharkSSLParseCAList.exe xxxx.cer –b xxxx.bin

③ クライアント証明書と秘密鍵を SharkSSL バイナリフォーマットに変換する

（プログラムに組み込む場合）

> SharkSSLParseCert XXXX-certificate.pem.crt XXXX-private.pem.key > mycert.c
（PC から直接書き込みを行う場合）

> SharkSSLParseCert XXXX-certificate.pem.crt XXXX-private.pem.key –b XXXX-certific

ate.bin

5.3.3 WiFi ドライバへの証明書の登録

本 API で証明書の書き込みを行う場合はご使用のプロジェクトに変換したファイルを組み込んでご使用ください。API を使った証明書の書き込みは、3 API 関数を参照ください。

変換し証明書（バイナリファイル）を PC より直接 WiFi モジュールに書き込む場合は、PC と WiFi モジュールの TX0,RX0 端子を、USB シリアル変換を介して接続し、AT コマンドを用いて書き込みを行ってください。その際、ボーレートは 115200bps としてください。

以下に証明書書き込みを行う場合の AT コマンド例を記載します。

（AT コマンド例）

- ATNSSLCERT=<証明書ファイル名>,<証明書サイズ>

  AT コマンド送信後 30 秒以内にバイナリファイルを送信してください。

証明書ファイル名：WiFi モジュールに登録する証明書ファイル名です。
20 文字以内で設定してください。
CA リストの場合は”calist＜番号>.crt“
クライアント証明書の場合は”cert<番号>.crt”
としてください。
証明書サイズ　　：バイナリデータサイズ（バイト数）を設定してください。

6. 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。）

ホームページとサポート窓口

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

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

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.02	2021/3/15	-	新規作成

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。