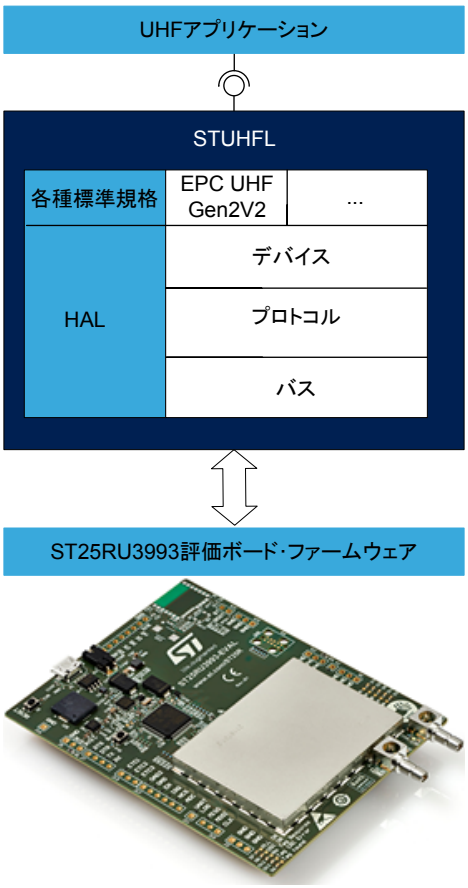
ST25RU3993 ST UHF ライブラリ

Introduction

ST UHF ライブラリ（STSW-ST25RU-SDK）は、ANSI C で記述したミドルウェア・ソフトウェア・スタックであり、ST25RU3993 ベースのリーダ・デバイスで使用する RAIN® RFID 対応アプリケーションの作成を目的としています。本書では、ST UHF ライブラリ・ソフトウェア API の使用方法について説明します。

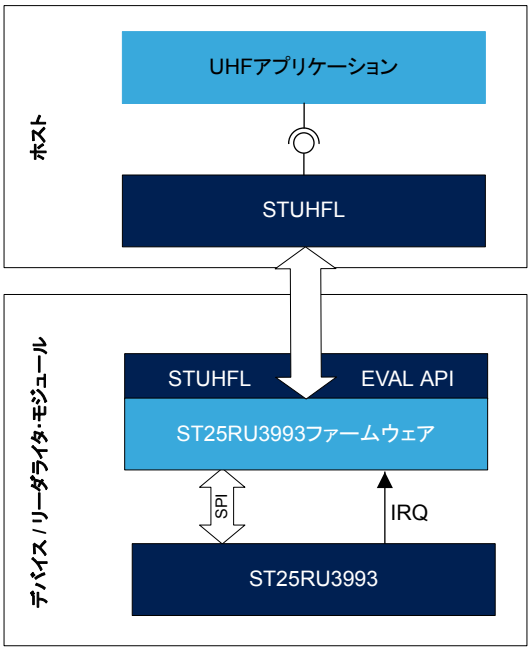
このパッケージと共に包括的なデモンストレーションのソース・コードも用意されているので、Windows®プラットフォームと Linux®プラットフォームで ST UHF ライブラリを使用する方法を知ることができます。このソフトウェア・スタック全体は ANSI C 互換および POSIX 互換なので、他のオペレーティング・システムとツールチェーンに迅速、容易に移植できます。



1 システムの概要

ST UHF ライブラリ（STUHFL）は、ホスト・システム上で動作する従来のミドルウェア・ソフトウェア・スタックによる設計を採用しています。低レベル通信の詳細を抽象化する簡潔なソフトウェア API を提供します。簡略化したシステムの概要に示すように、このシステムは、関連するソフトウェア・コンポーネントを実行する 2 つのハードウェア・コンポーネントで構成されています。

図 1. システムの概要



そのコンポーネントの 1 つ（デバイス / リーダライタ・モジュール）は、低レベル・ドライバの実装を使用してファームウェアを実行し、ST25RU3993 リーダライタ IC とさまざまなプロトコルを操作します。このファームウェアは、ST UHF ライブラリにソフトウェア・インタフェースを実装します。

もう 1 つのコンポーネント（ホスト側）は、ST UHF ライブラリを実行し、低レベル機能を抽象化します。UHF アプリケーション向けにクリーンなソフトウェア API を提供します。

ST UHF ライブラリは、オペレーティング・システムによってホスト・システム上で動作するほか、限られたリソースを使用して内蔵システム（OS による動作またはネイティブ動作）上でも動作します。専用のホスト・モジュールを持たない設計も可能で、その場合はリーダライタ・モジュール上でのみ完成状態のシステムが動作します。

1.1 一般情報

本書で取り上げているソフトウェアは Arm®ベースのデバイスをサポートしています。

注 Arm は、米国その他の地域における Arm Limited 社（およびその子会社）の登録商標です。

arm

1.2 機能

- ST25RU3993 を使用したリーダライタ・デバイス向けに RAIN® RFID 対応アプリケーションを作成するための包括的なミドルウェア
- ミラーリングしたホストとデバイス・ライブラリ API
- ANSI C99 のみによる記述
- POSIX 互換
- さまざまなプラットフォーム（MCU、RTOS、OS）にわたる容易な移植性

- 主な UHF 標準規格との互換性
  - EPC UHF Gen2v2
  - GB/T 29768
- ソース･コード例の実装を用意
  - Windows®
  - Linux®（Raspberry Pi 3B）
  - さらに多くのプラットフォーム実装例について開発中

1.3 ハードウェア要件

ST UHF ライブラリの主な目的は、ST25RU3993 の低レベル･ハードウェア･ドライバを簡素化すること、および使いやすいソフトウェア･インタフェースを提供することにあります。したがって、ST UHF ソフトウェア･ライブラリは、以下に挙げる ST25RU3993 ベースのデモンストレーション･ボードとそれに対応するファームウェアでのみ動作します。

- ST25RU3993 評価ボード
- ST25RU3993 ELANCE モジュール

この低レベル･ハードウェア･ドライバは、ファームウェアの動作に適切な MCU をホストする他のデバイスに容易に移植できます。

1.4 動作環境の構築

現在のところ、すべてのホスト側ソフトウェア･コンポーネントとデモンストレーション･コードは、Microsoft Visual Studio 2017（VS2017）を使用して作成されています。本パッケージには、2 種類のプラットフォーム向けプロジェクトをホストする VS2017 ソリューションが用意されています。その 1 つは Win32 プラットフォーム向けプロジェクトで、Windows 7 以降のプラットフォームで実行するコードの生成に適しています。もう 1 つは、Linux デバイス上で実行する Arm プラットフォーム向けプロジェクトです。Arm プロジェクトでは、VS2017 でリモート･コンパイルを実行する機能を使用します。そのためには、同じネットワークの中でアクセス可能な、適切に設定した Arm デバイスが必要です。

注 Arm 向けに用意されている ST UHF ライブラリ･プロジェクトとそれに該当するプロジェクト例では、開発に Raspberry Pi 3B ハードウェアを使用しています。プロジェクトの設定詳細については、該当するプロジェクト･ソリューションの各種ファイルを参照してください。

2 ソフトウェア·アーキテクチャ

ST UHF ライブラリは、従来のスタック形式ミドルウェアのソフトウェア設計に従い、リーダ·モジュール上で動作するコンポーネントおよびデフォルトのユース·ケースとしてホスト上で動作するコンポーネントで構成されています。

ST UHF ライブラリ全体は ANSI C 互換および POSIX 互換なので、他の各種オペレーティング·システムとプラットフォームへの移植が容易です。

広く普及している各種 UHF 標準規格が、現在の ST UHF ライブラリでサポートされています。今後のバージョンで、拡張機能とさらに多くのプロトコルをリリースする予定です。

Windows と Linux 向けに実動作するサンプル·ソース·コードが用意され、さまざまなライブラリ·モジュールを使用できます。

2.1 ST UHF ライブラリ API

公開されている ST UHF ライブラリ API を使用すると、ST25RU3993 リーダライタ IC 上で動作する RAIN アプリケーションを作成できます。公開されている ST UHF ライブラリ API には、次に挙げる 3 つのスタックされた層があります。

- アクティビティ
- セッション
- デバイス

この構成では、デバイスが動作状態に移行した時点から抽象化と複雑化の程度が高くなっていきます。デバイス層の目的は、レジスタへの直接アクセスのようなデバイス固有の機能を抽象化することにあります。セッション層では、デバイス層を使用して、EPC Gen2V2 標準や他の UHF の標準とプロトコルなどのプロトコル全体を実装します。アクティビティ層では、デバイス層とセッション層を使用して、タグの自動的な在庫管理のためにバックグラウンドで動作している機能を抽象化します。

表 1. 主要な層

層	説明
アクティビティ	組込みの“Inventory Runner”を介した、バックグラウンドでの在庫検索などの機能
セッション	EPC Gen2V2 などのさまざまな標準規格やプロトコルの実装
デバイス	レジスタへのアクセスなどのハードウェア固有機能の抽象化

2.2 STUHFL EVAL API

デバイス / リーダライタ·モジュール側では、STUHFL EVAL API が、ST25RU3993 の低レベル·ドライバに基本的なソフトウェア·インタフェースを提供します。ST UHF ライブラリは、この低レベル·インタフェース上に構築され、その ST25RU3993 依存機能を実現します。ホスト·システムが関与しないアプリケーションでは、リーダライタ / デバイス側で低レベルの STUHFL EVAL API を直接呼び出します。

注 ST UHF ライブラリ·スタックを使用しないアプリケーションは、このインタフェースに直接接続できます。

2.3 ST UHF ライブラリ·ラッパー

ST UHF ライブラリのホスト·インタフェースのほかに、迅速な試作を目的として次の 2 つのラッパーが用意されています。

- Java、C#、Python（開発中）
- ST EVAL API

Java、C#、Python は最先端のプログラミング言語であり、これらの言語をソフトウェア開発で使用すると、多彩な UHF アプリケーションを迅速、効率的に実装できます。さまざまなソフトウェア開発言語向けのラッパーのほか、ST EVAL API ラッパーなどの他のカスタム·インタフェース·ラッパーを実装することもできます。

注 他のプログラミング言語であっても、ネイティブな C コードの呼び出しに対応していれば、その言語向けのソフトウェア·ラッパーを容易に実装できます。

ホスト側の STUHFL EVAL API ラッパーは、デバイス側の STUHFL EVAL API を全面的にエミュレートします。これにより、ファームウェア·アプリケーションの開発をホスト·システム上で開始し、そのファームウェアをデバイス / リーダ側に移植することで開発タスクを完了できます。この方法の利点は、ホスト側であればデバッガ、高度なコード·トレーサ、他の多数のホスト·システム開発機能を利用できることです。このようなホスト·ツールを使用することで、コード開発を迅速なプロセスにして簡素化できます。ホスト側でファームウェアのコード開発が完了すれば、あとはそのファームウェア·アプリケーションをデバイス / リーダ側に“コピー·アンド·ペースト”するだけです。MCU 上でファームウェアを実行するうえで、ソース·コードの余分な修正が必要になることはありません。

注 STUHFL EVAL API ラッパーは、STUHFL EVAL API を全面的にエミュレートして、デバイス側で作成した場合と同様に使用できるようにします。

2.4 非公開層

公開されている ST UHF ライブラリ層の下位では、ホスト上で他のいくつかの層が、そのデバイス側の同等層と共に機能しています。この層スタックが、ホストとリーダライタ·デバイス間のデータ転送を扱っています。

- ディスパッチャ
- プロトコル
- バス

これらの層は、スタック設計に従い、それぞれに直接隣接する層にのみ依存しています。ディスパッチャ層の主な目的は、ST UHF ライブラリ API の各関数呼び出しを 1 つのモジュールにまとめ、プロトコル層に転送することにあります。

プロトコル層は、抽象化した関数呼び出しを転送可能なデータ·チャンクに変換し、これらのデータ·パケットをバス層に転送します。

最後に、バス層はこのデータ·チャンクを物理インタフェース上で送信し、デバイス / リーダライタ·モジュール側で受信できるようにします。デバイス / リーダライタ側からホスト側に向かう逆方向のデータ転送では、プロトコル層とディスパッチャ層が送信とは逆の順序でそれぞれのタスクを実行します。

表 2. 非公開層

層	説明
ディスパッチャ	API 層からのデータをマージし、API 層へのデータを分割します。
プロトコル	ホストとデバイス間で転送できるように、データを TLV フォーマットのデータ·チャンクにエンコードし、またそこからデコードします。
バス	データ交換を扱うプラットフォーム依存の転送層です。

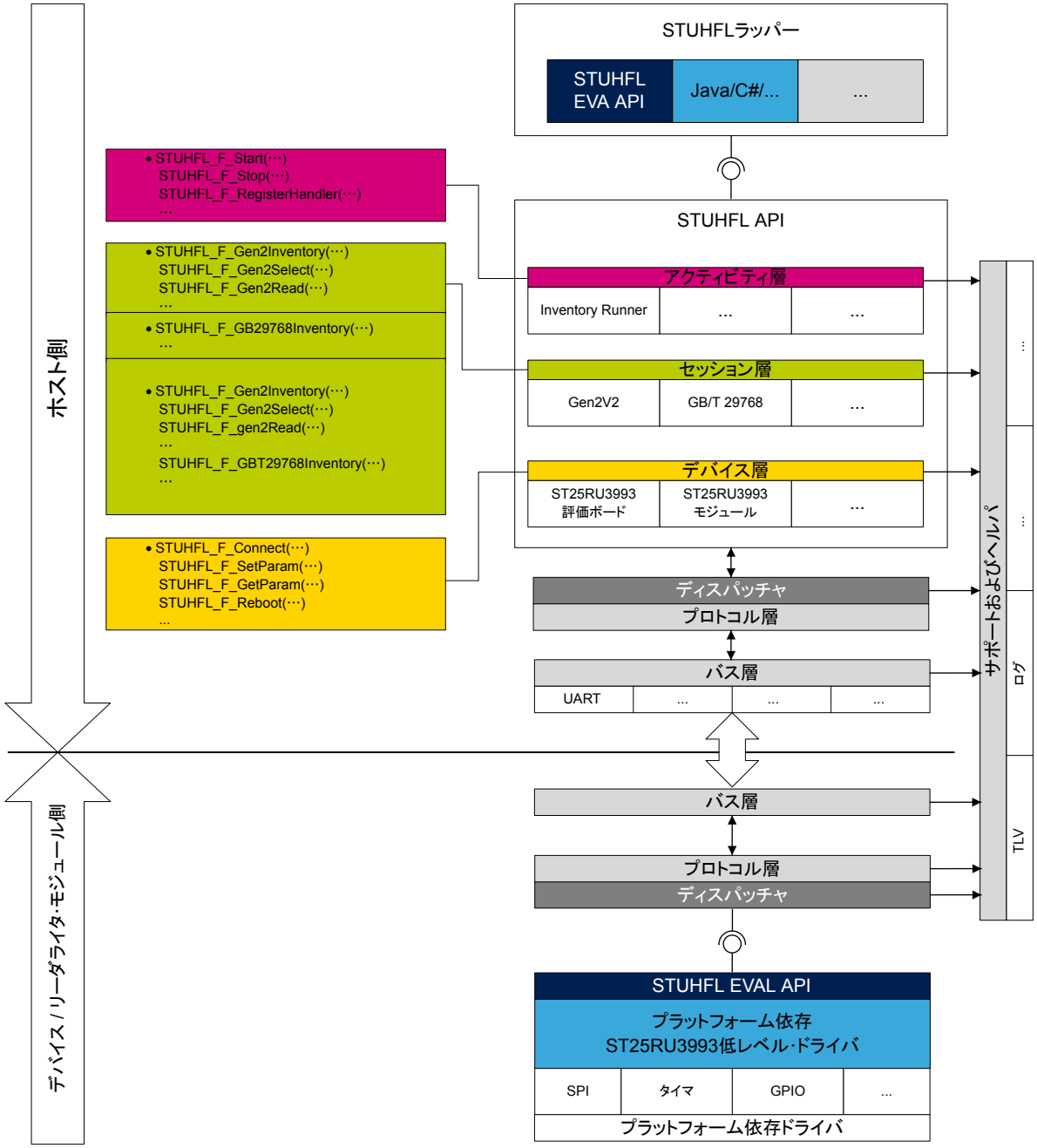
2.4.1 サポートおよびヘルパ

上記の非公開層のほか、このライブラリではサポート層とヘルパ層を実装します。これらの層は関連性のある各種機能を提供し、そのインタフェースを他のほとんどすべてのモジュールに公開します。

表 3. ヘルパ·モジュール

層	説明
TLV	“Tag-Length-Value”フォーマットのデータのエンコードとデコードに対するサポート
ログ	ログ·ヘルパ

図 2. ST UHFL ライブラリのシステム・アーキテクチャ

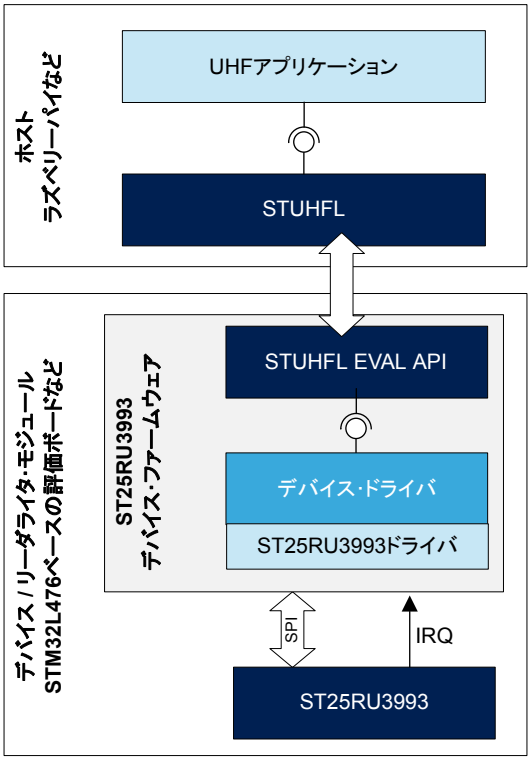


2.5 ホスト側での使用

ST UHF ライブラリ API で公開されている機能により、ST25RU3993 リーダライタ IC デバイスまたはモジュールを使用する RAIN アプリケーションをホスト側で開発できます。Windows または Linux の各 OS を実行するホスト・システムがデフォルトでサポートされています。このライブラリ全体に ANSI C および POSIX との互換性があるので、POSIX 互換の他のプラットフォーム向けにも容易にアプリケーションを作成できます。

ホスト側での使用例を以下の図に示します。

図 3. ST UHF ライブラリの使用方法概要

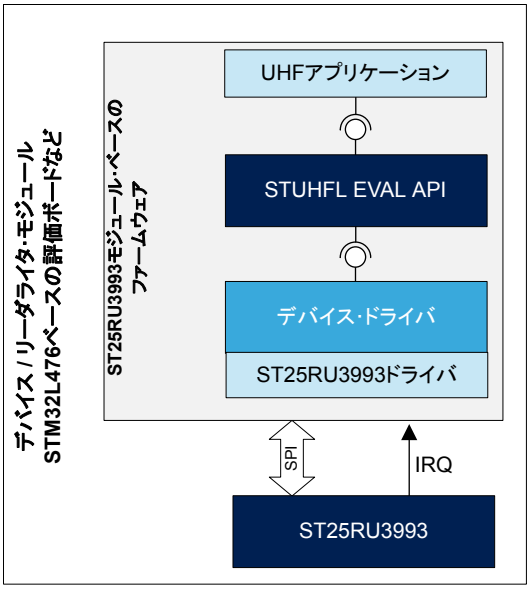


2.6 デバイス側での使用

STUHFL EVAL API で公開されている各種機能は、ST25RU3993 リーダライタ IC に用意されているすべての UHF 機能との抽象化インタフェースを提供します。これにより、低レベルのレジスタ・アクセスを手動で扱う必要がありません。したがって、このインタフェースはデバイスに依存せず、あらゆる抽象化 UHF アプリケーションの実装を実現します。

デバイス側での使用例を以下の図に示します。

図 4. デバイス側での ST UHF ライブラリ EVAL API の使用



3 ソース·コード

Windows と Linux 向けのサンプル·ソースをはじめとするソース·コード全体を www.st.com からダウンロードできます。

3.1 ディレクトリとファイルの構造

本パッケージは、以下のディレクトリ構造を収めた zip ファイルで提供されています。

図 5. ディレクトリ構造

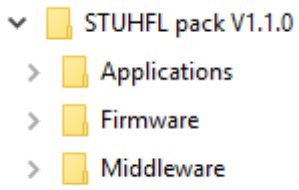


表 4. ディレクトリの説明

ディレクトリ	説明
Application	各種デモンストレーション·プロジェクトのソース·コード
Firmware	各種ファームウェア·プロジェクト
Middleware	Windows 向けと Linux 向けの ST UHF ライブラリ·プロジェクトのソース·コード

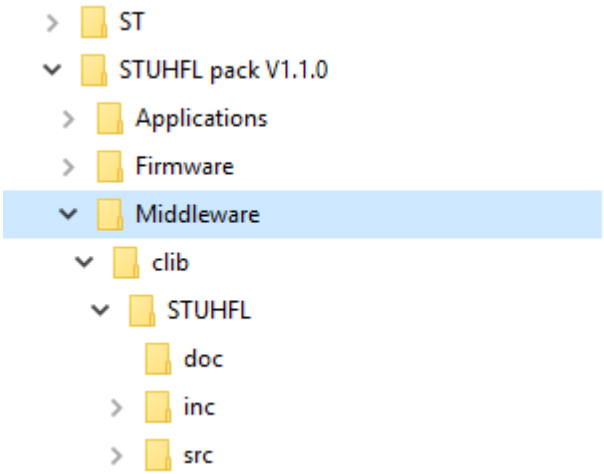
注 正しい動作とするには、ライブラリと共に提供されている該当のファームウェア·パッケージからいずれかを選択する必要があります。

“middleware/clib”フォルダの下に“STUHFL”フォルダがあります。

このフォルダには以下の各フォルダがあります。

- “doc”: API の詳しい説明を Windows の chm ファイルとして収めています。
- “inc”: 関連するすべてのインタフェース·ファイルを収めています。
- “src”: すべての実装ファイルを収めています。

図 6. STUHFL のディレクトリ構造



“inc”ディレクトリには、関連するすべてのインタフェース·ファイルと、プラットフォーム固有のファイルを収めたサブフォルダがあります。

図 7. “inc”の構造

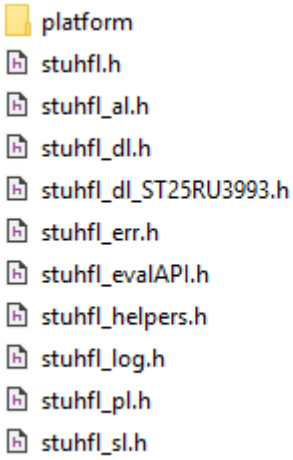


表 5. API の各ヘッダ

ファイル名	説明	注記
stuhfl.h	主な typedef と define を収めた基本的なヘッダ·ファイル	STUHFL API
stuhfl_al.h	アプリケーション·モジュール	STUHFL API
stuhfl_dl.h	デバイス·モジュールの汎用部分	STUHFL API
stuhfl_dl_ST25RU3993.h	ST25RU3993 IC 依存のデバイス·モジュール層	内部使用専用で、STUHFL API としては非公開
stuhfl_err.h	エラー·コードの定義	STUHFL API
stuhfl_evalAPI.h	STUHFL EVAL API の定義。STUHFL 対応ファームウェア実装により、STUHFL ラッパーの実装として使用されます。	STUHFL ラッパー - STUHFL EVAL API
stuhfl_helpers.h	ヘルパ·モジュール	内部使用専用で、STUHFL API としては非公開
stuhfl_log.h	ログ·モジュールの定義	内部使用専用で、STUHFL API としては非公開
stuhfl_pl.h	プロトコル層の定義	内部使用専用で、STUHFL API としては非公開
stuhfl_sl.h	セッション層	STUHFL API

“platform”ディレクトリは、他のプラットフォームへの移植で重要になる関連ファイルをすべて収めています。

図 8. “platform”ディレクトリ

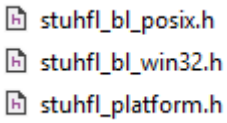


表 6. プラットフォーム依存の各ヘッダ

ファイル名	説明
Stuhfl_bl_posix.h	UART 通信向けの POSIX 互換バス層
stuhfl_bl_win32.h	Windows 向け UART 通信の実装

ファイル名	説明	
stuhfl_platform.h	プラットフォームとの非依存性を実現する汎用機能の実装	
stuhfl_dl_ST25RU3993.h	ST25RU3993 IC 依存のデバイス·モジュール層	内部使用専用で、STUHFL API としては非公開

3.2 ST UHF ライブラリ·ソリューションの場所

Win32 ダイナミック·リンク·ライブラリまたは Arm ベースの共有オブジェクト·ライブラリを作成するためのソリューションは./Middleware/clib/STUHFL.sln にあります。

注 STUHFL デモンストレーション·ソリューションには、STUHFL に基づくソフトウェア開発を考慮して、STUHFL プロジェクトと完成状態のデモンストレーション·アプリケーションが用意されています。これにより、Win32 プラットフォーム向けと Raspberry Pi プラットフォーム向けのアプリケーションを同時にビルドし、デバッグできます。

3.2.1 Windows 向け STUHFL の作成（DLL のみ）

以下の手順を実行する必要があります。

1. VS2017 ソリューション.\Middleware\clib\STUHFL.sln を開きます。
2. STUHFL プロジェクトを右クリックして[Rebuild]をクリックします。

注 これにより、ST25RU3993 リーダライタ IC ベースのデモンストレーション·ボード（Windows ホスト·システム専用）との通信に使用できる Win32 ダイナミック·リンク·ライブラリが生成されます。

3.2.2 Raspberry Pi 3B を使用した Linux 向け STUHFL の作成（共有オブジェクトのみ）

以下の手順を実行する必要があります。

1. VS2017 ソリューション.\Middleware\clib\STUHFL.sln を開きます。
2. Tools/Options/Cross platform のオプションで、使用している Raspberry Pi の IP アドレスを追加します。
3. STUHFL（Linux）プロジェクトのプロパティで、この Raspberry Pi をリモート·マシンとして設定します。
4. STUHFL（Linux）プロジェクトを右クリックしてリビルドのコマンドをクリックします。

注 これにより、ST25RU3993 リーダ IC ベースのデモンストレーション·ボード（Linux ホスト·システム専用）と通信する Linux 共有オブジェクト全体が生成されます。

3.3 サンプル·ソース·コードの実装

3.3.1 Windows 向け STUHFL デモの作成

以下の手順を実行する必要があります。

1. VS2017 ソリューション./Applications/STUHFL_demo/STUHFL_demo.sln を開きます。
2. ソリューションをリビルドします。
3. デモンストレーションのデバッグを開始します。

注 使用できるデモンストレーションの詳細については、STUHFL_demo プロジェクトの文書を参照してください。

3.3.2 Linux 向け STUHFL デモの作成

以下の手順を実行する必要があります。

1. VS2017 ソリューション./Applications/STUHFL_demo/STUHFL_demo_rpi.sln を開きます。
2. Tools\Options\Cross platform のオプションで、使用している Raspberry Pi の IP アドレスを追加します。
3. STUHFL と STUHFL デモンストレーション·プロジェクトのプロパティで、この Raspberry Pi をリモート·マシンとして設定します。
4. 通信ポートの場所を更新します（デフォルトの場所は/dev/ttyUSB0）。このポートは、ST25RU3993 評価ボードを Raspberry Pi に接続する場所です。
5. STUHFL_demo.c を参考にして、COM_PORT の define を調整します。
6. ソリューションをリビルドします。
7. デバッグを開始します。

注 使用できるデモンストレーションの詳細については、STUHFL_demo プロジェクトの文書を参照してください。

3.4 ST UHF ライブラリの移植性

ST UHF ライブラリは ANSI C 互換および POSIX 互換です。このライブラリを他のプラットフォームやオペレーティング·システムに移植するには、簡単なクロスコンパイルまたはリモートコンパイルを実行するだけで十分です。ホストからデバイスおよびデバイスからホストへの通信に使用する低レベル通信ドライバはこの例外であり、プラットフォームに依存します。移植を問題なく完了するために作業が別途必要になるプラットフォーム依存コンポーネントはすべて、./Middleware/clib/STUHFL/src/platform にあります。

注 ST UHF ライブラリを他のプラットフォームやオペレーティング·システムに移植するには、目的のプラットフォーム向けの低レベル通信ドライバを./Middleware/clib/STUHFL/src/platform フォルダに追加したうえでコンパイルに移ります。

4 ソフトウェア·インタフェースの説明

API の詳しい説明については、ソース·コード·パッケージにコンパイル済み HTML ファイル·フォーマット（chm）として用意されている文書を参照してください。以降の各セクションでは、さまざまなモジュールに使用する各種 API 機能を簡単に紹介します。

注 詳細については、ファイル“./Middleware/clib/STUHFL/ doc/STUHFL.chm”を参照してください。

4.1 デバイス層

デバイス層で公開されている機能。

4.1.1 接続

接続機能の概要を以下の表に示します。

表 7. STUHFL の接続機能

機能	説明
STUHFL_F_Connect	STUHFL を介してデバイスに接続します。
STUHFL_F_Disconnect	STUHFL を介して現在接続しているデバイスとの接続を解除します。
STUHFL_F_GetCtx	現在接続しているデバイスのデバイス·コンテキストを取得します。
STUHFL_F_Reset	現在接続しているデバイスをリセットします。

4.1.2 パラメータ·アクセス

パラメータ·アクセス機能の概要を以下の表に示します。

表 8. STUHFL のパラメータ·アクセス機能

機能	説明
STUHFL_F_SetParam	設定の値を設定するための汎用パラメータ設定機能です。
STUHFL_F_GetParam	設定の値を取得するための汎用パラメータ取得機能です。
STUHFL_F_GetParamInfo	パラメータ情報を取得します。
STUHFL_F_SetMultipleParams	複数のパラメータのリストを設定します。
STUHFL_F_GetMultipleParams	複数のパラメータのリストを取得します。

4.1.3 データ交換

データ交換機能の概要を以下の表に示します。

表 9. STUHFL のデータ交換機能

機能	説明
STUHFL_F_SendCmd	デバイスにコマンドを送信します。
STUHFL_F_ReceiveCmdData	デバイスへのコマンドを受信します。
STUHFL_F_ExecuteCmd	1 回の呼び出しで送信と受信を組み合わせて実行します。
STUHFL_F_GetRawData	デバイスから生データを受信します。

4.1.4 汎用管理

汎用管理機能の概要を以下の表に示します。

表 10. STUHFL の管理機能

機能	説明
STUHFL_F_GetVersion	デバイスのバージョン情報を取得します。
STUHFL_F_GetInfo	デバイス情報を取得します。
STUHFL_F_Upgrade	ファームウェアのアップグレード
STUHFL_F_EnterBootloader	再起動してブートローダを開始します。
STUHFL_F_Reboot	ファームウェアを再起動します。

4.2 セッション層

4.2.1 STUHFL の Gen2V2

Gen2V2 機能の概要を以下の表に示します。

表 11. STUHFL の Gen2V2 機能

機能	説明
STUHFL_F_Gen2_Inventory	現在の Inventory 設定と Gen2 設定に応じた Gen2 Inventory コマンド
STUHFL_F_Gen2_Select	Gen2 対応タグを選択またはフィルタ処理する Gen2 Select コマンド
STUHFL_F_Gen2_Read	Gen2 対応タグから読み取る Gen2 Read コマンド
STUHFL_F_Gen2_Write	Gen2 対応タグにデータを書き込む Gen2 Write コマンド
STUHFL_F_Gen2_Lock	Gen2 対応タグをロックする Gen2 Lock コマンド
STUHFL_F_Gen2_Kill	Gen2 対応タグを停止する Gen2 Kill コマンド
STUHFL_F_Gen2_GenericCmd	汎用の Gen2 ビット交換
STUHFL_F_Gen2_QueryMeasureRssi	Gen2 Query コマンド実行中の RSSI 測定

4.2.2 STUHFL の GB/T 29768

GB/T 29768 機能の概要を以下の表に示します。

表 12. STUHFL の GB/T 29768 機能

機能	説明
STUHFL_F_Gb29768_Inventory	現在の Inventory と GB/T 29768 設定に応じた GB/T 29768 Inventory コマンド
STUHFL_F_Gb29768_Sort	GB/T 29768 対応タグを選択またはフィルタ処理する GB/T 29768 Sort コマンド
STUHFL_F_Gb29768_Read	GB/T 29768 対応タグからデータを読み取る GB/T 29768 Read コマンド
STUHFL_F_Gb29768_Write	GB/T 29768 対応タグにデータを書き込む GB/T 29768 Write コマンド
STUHFL_F_Gb29768_Lock	GB/T 29768 対応タグをロックする GB/T 29768 Lock コマンド
STUHFL_F_Gb29768_Kill	GB/T 29768 対応タグを完全停止する GB/T 29768 Kill コマンド
STUHFL_F_Gb29768_Erase	GB/T 29768 対応タグのデータを消去する GB/T 29768 Erase コマンド

4.3 アクティビティ層

4.3.1 アクション

アクション機能の概要を以下の表に示します。

表 13. STUHFL のアクション機能

機能	説明
STUHFL_F_Start	フォアグラウンドまたはバックグラウンドのアクションを開始します。
STUHFL_F_Stop	アクションを停止します。

4.4 STUHFL EVAL API ラッパー

4.4.1 設定

設定機能の概要を以下の表に示します。

表 14. STUHFL EVAL API の設定機能

機能	説明
SetRegister	指定されたアドレスの ST25RU3993 レジスタに値を書き込みます。
GetRegister	指定されたアドレスの ST25RU3993 レジスタから値を読み取ってホストに返します。
SetRwdCfg	リーダライタの設定を変更します。
GetRwdCfg	リーダライタの設定を取得します。
SetAntennaPower	アンテナ電力を変更します。
GetAntennaPower	アンテナ電力を取得します。
SetGen2Cfg	Gen2 の設定を変更します。
SetTxRxCfg	送受信の設定を変更します。
SetPA_Cfg	電力増幅器の設定を変更します。
SetInventoryCfg	一般的な在庫の設定を変更します。
GetGen2Cfg	Gen2 の設定を取得します。
GetGBT29768Cfg	GBT29768 の設定を取得します。
GetTxRxCfg	送受信の設定を取得します。
GetPA_Cfg	電力増幅器の設定を取得します。
GetInventoryCfg	一般的な在庫の設定を取得します。

4.4.2 周波数

周波数設定機能の概要を以下の表に示します。

表 15. STUHFL EVAL API の周波数設定機能

機能	説明
SetFreqProfile	周波数プロファイルを設定します。
SetFreqProfileCustomAppend	プロファイルの対象とする周波数を設定します。
SetFreqHop	周波数ホッピング時間を設定します。
SetFreqLBT	Listen Before Talk の設定を変更します。
SetFreqCondMod	連続変調の設定を変更します。
GetFreqRSSI	周波数の RSSI を取得します。
GetFreqReflectedPower	プロファイルの対象とする周波数を取得します。
GetFreqProfileInfo	周波数ホッピング時間を取得します。
GetFreqHop	Listen Before Talk の設定を取得します。
GetFreqLBT	連続変調の設定を取得します。

4.4.3 チューニング

チューニング機能の概要を以下の表に示します。

表 16. STUHFL EVAL API のチューニング機能

機能	説明
SetTuning	Cin、Clen、および Cout を設定します。
SetTuningTableEntry	チューニング·テーブルのエントリを設定します。
SetTuningTableDefault	デフォルトのチューニングに戻します。
SetTuningTableSave2Flash	現在設定されているチューニング·テーブルを保存し、テーブルをフラッシュします。
SetTuningTableEmpty	チューニング·テーブルのエントリを削除します。
GetTuning	Cin、Clen、および Cout のコンフィギュレーションを取得します。
GetTuningTableEntry	チューニング·テーブルのエントリを取得します。
GetTuningTableInfo	現在のチューニング·テーブル情報を設定します。
Tune	チューニングを開始します。

4.4.4 Gen2V2

Gen2V2 機能の概要を以下の表に示します。

表 17. STUHFL EVAL API の Gen2V2 機能

機能	説明
Gen2_Inventory	現在の Inventory 設定と Gen2 設定に応じた Gen2 Inventory コマンド
Gen2_Select	Gen2 対応タグを選択またはフィルタ処理する Gen2 Select コマンド
Gen2_Read	Gen2 対応タグから読み取る Gen2 Read コマンド
Gen2_Write	Gen2 対応タグにデータを書き込む Gen2 Write コマンド
Gen2_Lock	Gen2 対応タグをロックする Gen2 Lock コマンド
Gen2_Kill	Gen2 対応タグを完全休眠させる Gen2 Kill コマンド
Gen2_GenericCmd	汎用的な Gen2 ビット交換

機能	説明
Gen2_QueryMeasureRssi	Gen2 Query コマンド実行中の RSSI 測定

4.4.5 GB/T 29768

GB/T 29768 機能の概要を以下の表に示します。

表 18. STUHFL EVAL API の GB/T 29768 機能

機能	説明
Gb29768_Inventory	現在の Inventory 設定と GB/T 29768 設定に応じた GB/T 29768 Inventory コマンド
Gb29768_Sort	GB/T 29768 対応タグを選択またはフィルタ処理する GB/T 29768 Sort コマンド
Gb29768_Read	GB/T 29768 対応タグからデータを読み取る GB/T 29768 Read コマンド
Gb29768_Write	GB/T 29768 対応タグにデータを書き込む GB/T 29768 Write コマンド
Gb29768_Lock	GB/T 29768 対応タグをロックする GB/T 29768 Lock コマンド
Gb29768_Kill	GB/T 29768 対応タグを完全休眠させる GB/T 29768 Kill コマンド
Gb29768_Erase	GB/T 29768 対応タグのデータを消去する GB/T 29768 Erase コマンド

4.4.6 Inventory Runner

Inventory Runner 機能の概要を以下の表に示します。

表 19. STUHFL EVAL API の Inventory Runner 機能

機能	説明
InventoryRunnerStart	Inventory Runner を開始します。
InventoryRunnerStop	現在の Inventory Runner を停止します。

Revision history

表 20. 文書改版履歴

日付	版	変更内容
2019 年 9 月 17 日	1	初版発行
2019 年 11 月 26 日	2	文書タイトルとセクション概要を更新。 文書全体に軽微な編集を適用。
2020 年 6 月 3 日	3	更新: • セクション概要、セクション 1.2 機能、セクション 1.3 ハードウェア要件、セクション 2.3 ST UHF ライブラリ·ラッパー • 図 2. ST UHFL ライブラリのシステム·アーキテクチャ • セクション 4.2.2 STUHFL の GB/T 29768 および セクション 4.4.5 GB/T 29768 • 表 12. STUHFL の GB/T 29768 機能、表 18. STUHFL EVAL API の GB/T 29768 機能 表 1.ハードウェア要件を削除。

目次

1 システムの概要 2
1.1 一般情報 2
1.2 機能 2
1.3 ハードウェア要件 3
1.4 動作環境の構築 3
2 ソフトウェア・アーキテクチャ 4
2.1 ST UHF ライブラリ API 4
2.2 STUHFL EVAL API 4
2.3 ST UHF ライブラリ・ラッパー 4
2.4 非公開層 5
2.4.1 サポートおよびヘルパ 5
2.5 ホスト側での使用 7
2.6 デバイス側での使用 8
3 ソース・コード 9
3.1 ディレクトリとファイルの構造 9
3.2 ST UHF ライブラリ・ソリューションの場所 11
3.2.1 Windows 向け STUHFL の作成（DLL のみ） 12
3.2.2 Raspberry Pi 3B を使用した Linux 向け STUHFL の作成（共有オブジェクトのみ） 12
3.3 サンプル・ソース・コードの実装 12
3.3.1 Windows 向け STUHFL デモの作成 12
3.3.2 Linux 向け STUHFL デモの作成 12
3.4 ST UHF ライブラリの移植性 12
4 ソフトウェア・インタフェースの説明 13
4.1 デバイス層 13
4.1.1 接続 13
4.1.2 パラメータ・アクセス 13
4.1.3 データ交換 13
4.1.4 汎用管理 14
4.2 セッション層 14
4.2.1 STUHFL の Gen2V2 14
4.2.2 STUHFL の GB/T 29768 14
4.3 アクティビティ層 15
4.3.1 アクション 15
4.4 STUHFL EVAL API ラッパー 15
4.4.1 設定 15

4.4.2 周波数 16
4.4.3 チューニング 16
4.4.4 Gen2V2 16
4.4.5 GB/T 29768 17
4.4.6 Inventory Runner 17
改訂履歴 18

表一覧

表 1. 主要な層 . . . 4
表 2. 非公開層 . . . 5
表 3. ヘルパ·モジュール . . . 5
表 4. ディレクトリの説明 . . . 9
表 5. API の各ヘッダ . . . 10
表 6. プラットフォーム依存の各ヘッダ . . . 10
表 7. STUHFL の接続機能 . . . 13
表 8. STUHFL のパラメータ·アクセス機能 . . . 13
表 9. STUHFL のデータ交換機能 . . . 13
表 10. STUHFL の管理機能 . . . 14
表 11. STUHFL の Gen2V2 機能 . . . 14
表 12. STUHFL の GB/T 29768 機能 . . . 14
表 13. STUHFL のアクション機能 . . . 15
表 14. STUHFL EVAL API の設定機能 . . . 15
表 15. STUHFL EVAL API の周波数設定機能 . . . 16
表 16. STUHFL EVAL API のチューニング機能 . . . 16
表 17. STUHFL EVAL API の Gen2V2 機能 . . . 16
表 18. STUHFL EVAL API の GB/T 29768 機能 . . . 17
表 19. STUHFL EVAL API の Inventory Runner 機能 . . . 17
表 20. 文書改版履歴 . . . 18

図一覧

図 1. システムの概要 2
図 2. ST UHFL ライブラリのシステム·アーキテクチャ 6
図 3. ST UHF ライブラリの使用方法概要 7
図 4. デバイス側での ST UHF ライブラリ EVAL API の使用 8
図 5. ディレクトリ構造 9
図 6. STUHFL のディレクトリ構造 9
図 7. “inc”の構造 10
図 8. “platform”ディレクトリ 10

重要なお知らせ（よくお読みください）

STMicroelectronics NV およびその子会社（両者を総称して以下、ST とします）は、ST 製品および本書の内容をいつでも予告なく変更、修正、改善、改定および改良する権利を留保します。購入される方は、発注前に ST 製品に関する最新の関連情報を必ず入手してください。ST 製品は、注文請書発行時点で有効な ST の販売条件に従って販売されます。

ST 製品の選択並びに使用については購入されるお客様がすべての責任を負うものとします。お客様の製品のアプリケーション支援や設計に関して ST は一切の責任を負いません。

明示又は黙示を問わず、ST は本書においていかなる知的財産権の実施権も許諾致しません。

本書で説明されている情報とは異なる条件で ST 製品が再販された場合、その製品について ST が提供するいかなる保証も無効となります。

ST および ST ロゴは STMicroelectronics の商標です。ST の商標に関する詳細は、www.st.com/trademarks をご覧ください。その他の製品またはサービスの名称は、それぞれの所有者に帰属します。

本書の情報は本書の以前のバージョンで提供されたすべての情報に優先し、それに代わるものです。

© 2021 STMicroelectronics – All rights reserved