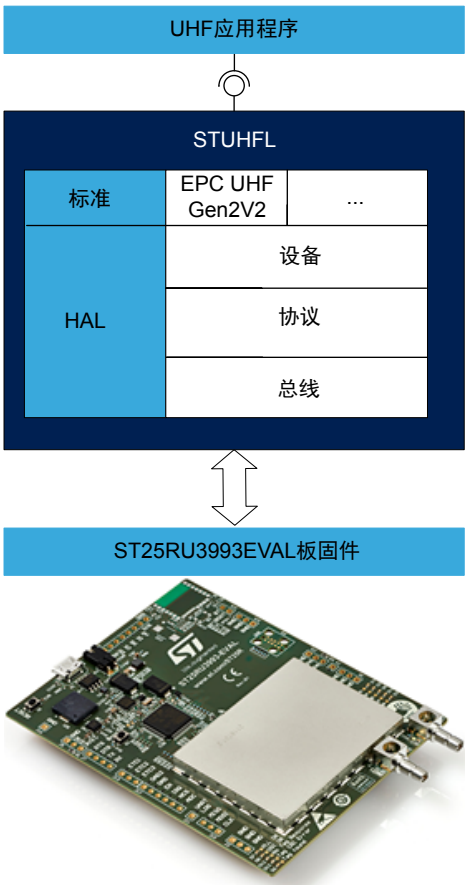
ST25RU3993 ST UHF 库

引言

ST UHF 库（STSW-ST25RU-SDK）是用 ANSI C 编程语言编写的中间件软件栈，可为基于 ST25RU3993 的读写器设备构建支持 RAIN®RFID 的应用。本文档介绍了 ST UHF 库软件 API 的使用方法。

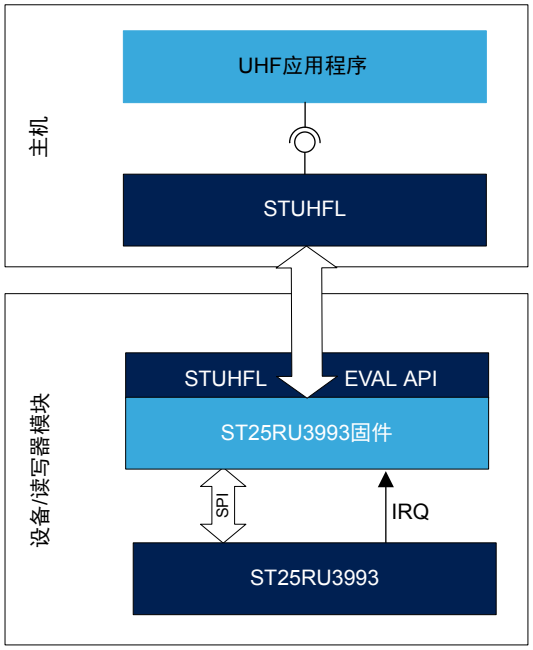
该软件包中还包含完整的演示源代码，用于演示如何在 Windows®和 Linux®平台上使用 ST UHF 库。该完整的软件堆栈兼容 ANSI C 和 POSIX，可以快速、直接地移植到其他操作系统和/或工具链。



1 系统概述

STUHF 库（STUHFL）设计是典型的运行在主机系统上的中间件软件堆栈。它提供简单的软件 API，用于抽象底层通信细节。如简化的系统视图中所示，该系统由运行相关软件组件的两个硬件组件组成。

图 1. 系统概述



第一个组件（设备/读写器模块）运行具有底层驱动程序实现的固件，以操作 ST25RU3993 读写器 IC 和各种协议。固件将一个软件接口实现到 ST UHF 库。

第二个组件（主机端）运行 ST UHF 库并对底层功能进行抽象。它为 UHF 应用提供一个干净的软件 API。

ST UHF 库可以在带有操作系统的主机系统上运行，也可以在资源有限的嵌入式系统（带有操作系统或原生系统）上运行。设计时也可以不配备专用的主机模块，整个系统只在读写器模块上运行。

1.1 概述

本文档中描述的软件支持基于 Arm®的设备。

提示 *Arm is a registered trademark of Arm Limited (or its subsidiaries) in the US and/or elsewhere.*

1.2 特征

- 全面的中间件，可为基于 ST25RU3993 的读写器设备构建支持 RAIN® RFID 的应用
- 镜像的“主机和设备”库 API
- 全部以 ANSI C99 语言编写
- 兼容 POSIX
- 可在不同的平台（MCU/RTOS/OS）之间直接移植
- 兼容主要 UHF 标准
  - EPC UHF Gen2v2
  - GB/T 29768

- 提供源代码示例实现
  - Windows®
  - Linux®（Raspberry Pi 3B）
  - 更多平台实现示例敬请期待

1.3 硬件要求

ST UHF 库的主要目的是简化 ST25RU3993 底层硬件驱动程序，并提供易于使用的软件接口。因此，ST UHF 软件库只能在下列基于 ST25RU3993 的演示板及其相应的固件上运行：

- ST25RU3993 EVAL 板
- ST25RU3993 ELANCE 模块

无论如何都可以将底层硬件驱动程序移植到配备合适 MCU 的其他设备上以驱动固件。

1.4 编译环境

目前所有的主机端软件组件和演示代码均使用 Microsoft Visual Studio 2017（VS2017）生成。该软件包包含一个 VS2017 解决方案，用于面向两个不同平台的项目。Win32 平台的第一个项目适用于生成运行在 Windows 7 平台（以及更新平台）上的代码。第二个项目面向运行在 Linux 设备上的 Arm 平台。Arm 项目使用 VS2017 进行远程编译。这要求 Arm 设备处于同一网络中且正确配置，这样才能确保可访问性。

提示 *面向 Arm 平台的现有 ST UHF 库项目和相应的示例项目使用 Raspberry Pi 3B 硬件进行开发。如需了解有关项目配置的详细信息，请参阅相应的项目解决方案文件。*

2 软件架构

ST UHF 库遵循经典的堆栈中间件软件设计，（作为默认用例）由运行在读写器模块和主机系统上的组件组成。

完整的 ST UHF 库兼容 ANSI-C 和 POSIX，可以轻松移植到其他几个操作系统和平台。

ST UHF 库目前支持最常见的 UHF 标准。未来的版本中将发布扩展和更多协议。

试验样品源代码可用于 Windows 和 Linux，以使用不同的库模块。

2.1 ST UHF 库 API

开放的 ST UHF 库 API 允许用户为 ST25RU3993 读写器设备构建 RAIN 应用。三个堆叠层表示完全开放的 ST UHF 库 API。

- 活动
- 会话
- 设备

在这种排列中，从设备到活动的抽象性和复杂性逐渐增加。设备层的作用是抽象特定于硬件的功能，如直接寄存器访问。会话层使用设备层实现完整的协议，比如 EPC Gen2V2 标准或其他 UHF 标准和协议。活动层使用设备层和会话层进行功能抽象，以便为在后台运行的标签进行自动盘点。

表 1. 主要层

层	说明
活动	通过内置的“Inventory Runner”进行后台库存查询等功能
会话	实现不同的标准或协议，如 EPC Gen2V2
设备	对特定于硬件的功能（如寄存器访问）进行抽象

2.2 STUHFL EVAL API

在设备/读写器模块端，STUHFL EVAL API 为 ST25RU3993 底层驱动程序提供一个基本软件接口。ST UHF 库构建在该底层接口之上，并派生出其依赖于 ST25RU3993 的功能。不涉及主机系统的应用程序直接调用读写器/设备端的底层 STUHFL EVAL API。

提示 *不使用 ST UHF 库堆栈的应用程序可以直接连接到该接口。*

2.3 ST UHF 库封装器

在 ST UHF 库主机接口之上，有两个封装器可用于快速原型设计。

- Java/C#/Python（开发中）
- ST EVAL API

Java、c#和 Python 是最先进的编程语言，使软件开发人员能够快速有效地实现各种 UHF 应用。除了用于不同软件开发语言的封装器外，还可以实现其他自定义接口封装器，如 ST EVAL API 封装器。

提示 *当这些编程语言支持调用本地 C 代码时，可以轻松实现其他语言的软件封装器。*

STUHFL EVAL API 封装器（主机端）完全模拟了设备端的 STUHFL EVAL API。由此用户便能在主机系统上开始开发固件应用，并将固件移植到设备/读写器端以完成自身任务。这种方法的优点在于，主机端能够有效发挥调试器、高级代码跟踪器和许多其他主机系统开发特性。这些主机工具加速并简化了代码开发。当主机端固件代码开发完成后，只需将固件应用程序代码“复制并粘贴”到设备/读写器端即可。在 MCU 上运行固件无需额外的源代码修改。

提示 *STUHFL EVAL API 封装器完全模拟设备端的 STUHFL EVAL API。*

2.4 非开放层

在开放的 ST UHF 库层之下，其他几个层在主机上工作，而对应层则工作在设备端。该层堆栈负责主机和读写器设备之间的数据传输。

- 调度程序
- 协议
- 总线

这些层遵循堆栈设计，仅依靠直接邻层。调度层的主要目的是将 ST UHF 库 API 函数调用捆绑在单一模块中，并将其转发到协议层。

协议层将抽象后的函数调用转换为可转移的数据块，并将这些数据包转发给总线层。

最后，总线层通过物理接口发送数据块，并在设备/读写器模块端进行接收。对于从设备/读写器端到主机端的反向数据传输，协议层和调度层执行相反的任务。

表 2. 非开放层

层	说明
调度程序	与 API 层合并和分离
协议	将数据编码和解码为 TLV 格式的数据块，以便在主机和设备之间传输
总线	平台依赖型传输层处理数据交换

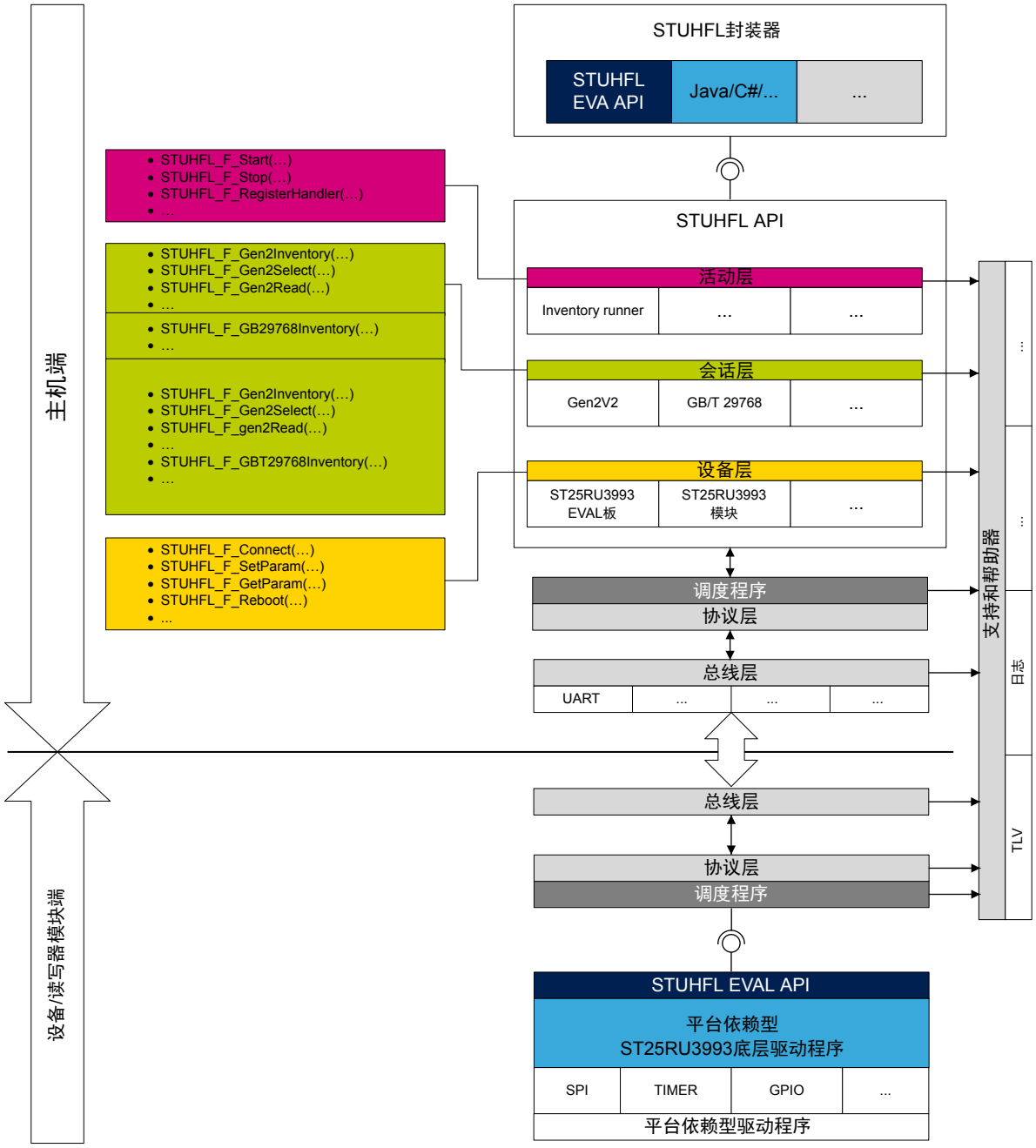
2.4.1 支持和帮助器

除上述非开放层外，库还实现支持和帮助器层，用于提供相关功能，并向几乎所有其他模块开放其接口。

表 3. 帮助器模块

层	说明
TLV	支持 Tag-Length-Value 编码和解码
日志	日志记录帮助器

图 2. **ST UHFL 库系统架构**

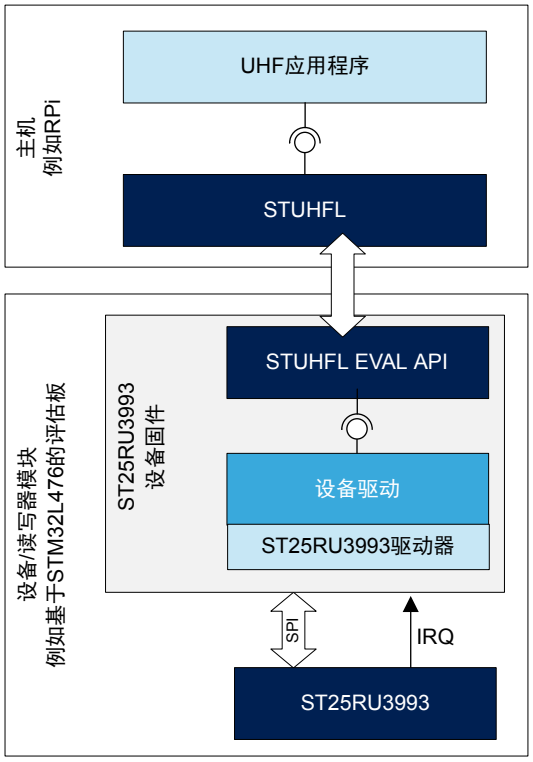


2.5 主机端使用

ST UHF 库 API 公开的功能支持基于 ST25RU3993 读写器 IC 设备或主机设备上的模块开发 RAIN 应用。默认支持运行 Windows 或 Linux 操作系统的主机系统。由于整个库都兼容 ANSI-C 和 POSIX，所以也可以面向其他兼容 POSIX 的平台而轻松构建。

下图显示了一个主机端使用示例。

图 3. ST UHF 库使用概述

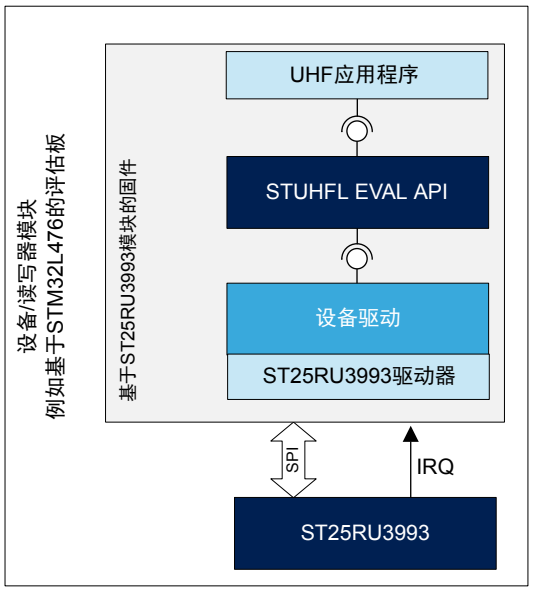


2.6 设备端使用

STUHFL EVAL API 开放的功能为 ST25RU3993 读写器 IC 提供的全部 UHF 功能提供了抽象接口，无需手动处理底层寄存器访问事宜。因此，该接口独立于设备，支持实现任何抽象的 UHF 应用。

下图显示了一个设备端使用示例。

图 4. ST UHF 库 EVAL API 设备端使用



3 源代码

可以从 www.st.com 下载完整的源代码（包括面向 Windows 和 Linux 的示例源代码）。

3.1 目录和文件结构

该封装以 zip 文件格式提供，采用以下目录结构。

图 5. 目录结构

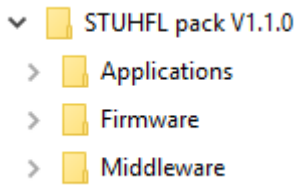


表 4. 目录说明

目录	说明
应用	多个演示项目源代码
固件	多个固件项目
中间件	面向 Windows 和 Linux 的 ST UHF 库项目作为源代码

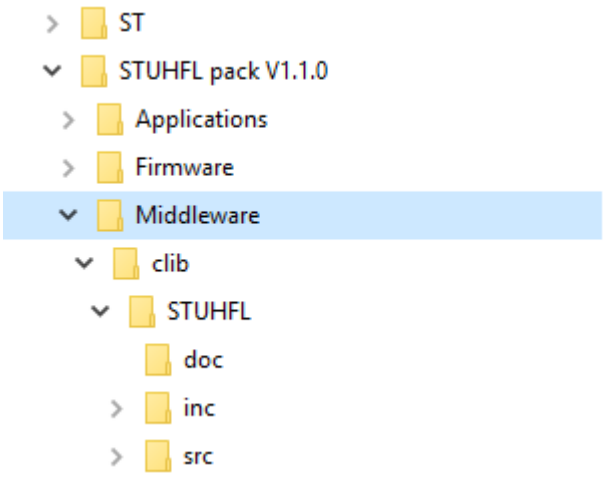
提示 *为了正确操作，用户必须从随库提供的固件包中选择一个适当的。*

“STUHFL”文件位于“middleware/clib” 文件夹中。

其中包含以下文件夹：

- “doc”：包含详细的 API 描述，如 Windows chm 文件。
- “inc”：包含所有相关接口文件
- “src”：包含所有实现文件

图 6. STUHFL 目录结构



“inc”目录包含所有相关接口文件和一个包含平台特定文件的子文件夹。

图 7. "inc"结构

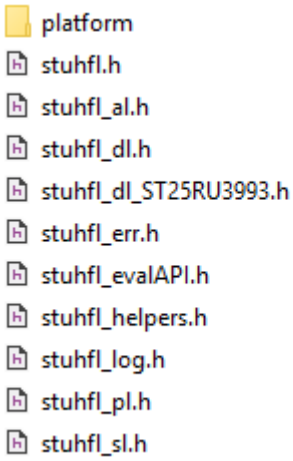


表 5. API 头文件

文件名	说明	注释
stuhfl.h	基础头文件，包含主要类型定义和定义	STUHFL API
stuhfl_al.h	应用模块	STUHFL API
stuhfl_dl.h	设备模块的通用部分	STUHFL API
stuhfl_dl_ST25RU3993.h	依赖于 ST25RU3993 IC 的设备模块层	不开放为 STUHFL API，内部使用
stuhfl_err.h	错误代码定义	STUHFL API
stuhfl_evalAPI.h	STUHFL EVAL API 定义。由启用了 STUHFL 的固件实现使用，作为 STUHFL 封装器实现	STUHFL wrapper - STUHFL EVAL API
stuhfl_helpers.h	辅助模块	不开放为 STUHFL API，内部使用
stuhfl_log.h	日志模块定义	不开放为 STUHFL API，内部使用
stuhfl_pl.h	协议层定义	不开放为 STUHFL API，内部使用
stuhfl_sl.h	会话层	STUHFL API

“平台”目录包含移植到其他平台时所有重要的相关文件。

图 8. “平台”目录

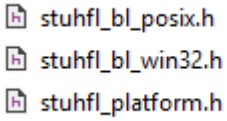


表 6. 平台依赖型头文件

文件名	说明	
Stuhfl_bl_posix.h	兼容 POSIX 的总线层，用于 UART 通信	
stuhfl_bl_win32.h	UART 通信实现，用于 Windows	
stuhfl_platform.h	通用功能实现，用于实现平台独立性	
stuhfl_dl_ST25RU3993.h	依赖于 ST25RU3993 IC 的设备模块层	不开放为 STUHFL API，内部使用

3.2 ST UHF 库解决方案位置

用于构建 Win32 动态链接库或基于 Arm 的共享对象库的解决方案位于./Middleware/clib/STUHFL.sln。

提示 *对于基于 STUHFL 的软件开发，STUHFL 演示解决方案包含 STUHFL 项目和一个完整的演示应用程序。如此一来，用户便能同时为 Win32 和 Raspberry Pi 平台进行构建和调试。*

3.2.1 构建面向 Windows 的 STUHFL（仅 DLL）

用户必须执行以下步骤：

1. 打开 VS2017 solution .\Middleware\clib\STUHFL.sln
2. 右键点击 STUHFL 项目，然后点击“重新构建”

提示 之后会生成一个 Win32 动态链接库，用于与基于 ST25RU3993 读写器 IC 的演示板通信（仅 Windows 主机系统）。

3.2.2 通过 Raspberry Pi 3Bbuild 构建面向 Linux 的 STUHFL（仅共享对象）

用户必须执行以下步骤：

1. 打开 VS2017 solution .\Middleware\clib\STUHFL.sln
2. 在 Tools\Options\Cross platform options 中添加您的 Raspberry Pi 的 IP 地址
3. 在 STUHFL（Linux）项目属性中，将其设置为远程机器
4. 右键点击 STUHFL（Linux）项目，然后点击“重新构建”

提示 之后会生成一个完全共享的 Linux 对象，用于与基于 ST25RU3993 读写器 IC 的演示板通信（仅 Linux 主机系统）。

3.3 示例源代码的实现

3.3.1 构建面向 Windows 的 STUHFL 演示代码

用户必须执行以下步骤：

1. 打开 VS2017 solution ./Applications/STUHFL_demo/STUHFL_demo.sln
2. 重新构建解决方案
3. 启动调试功能，调试演示代码

提示 如需了解关于可用演示的更多信息，请查看 STUHFL_demo 项目的文档。

3.3.2 构建面向 Linux 的 STUHFL 演示代码

用户必须执行以下步骤：

1. 打开 VS2017 solution ./Applications/STUHFL_demo/STUHFL_demo_rpi.sln
2. 在 Tools\Options\Cross platform options 中添加您的 Raspberry Pi 的 IP 地址。
3. 在 STUHFL 和 STUHFL 演示的项目属性中，将其设置为远程机器
4. 更新将 ST25RU3993 EVAL 板连接 Raspberry Pi 的通信端口位置（默认为/dev/ttyUSB0）。
5. 参照 STUHFL_demo.c 并调整 COM_PORT 定义
6. 重新构建解决方案
7. 启动调试功能

提示 如需了解关于可用演示的更多信息，请查看 STUHFL_demo 项目的文档。

3.4 ST UHF 库可移植性

ST UHF 库兼容 ANSI-C 和 POSIX。要将库移植到其他平台或操作系统，只进行一次交叉编译或远程编译即可。也有例外：从主机到设备（以及从设备到主机）的底层通信驱动程序总是依赖于平台。所有需要在成功移植之前进行修改的平台依赖型组件都位于./Middleware/clib/STUHFL/src/platform 中。

提示 为了将 STUHF 库移植到其他平台和/或操作系统，需要在编译前将适合您平台的底层通信驱动程序添加到./Middleware/clib/STUHFL/src/platform 文件夹中。

4 软件接口描述

如需详细的 API 描述，请查看可用的文档，该文档是源代码包的一部分，以编译后的 HTML 文件格式（chm）提供。以下几节只是简要概述了不同模块的不同 API 功能。

提示 *有关详细信息，请参照文件"./Middleware/clib/STUHFL/ doc/STUHFL.chm"*

4.1 设备层

设备层的开放功能。

4.1.1 连接

下表概述了连接功能。

表 7. STUHFL 连接功能

功能	说明
STUHFL_F_Connect	通过 STUHFL 连接到设备
STUHFL_F_Disconnect	断开通过 STUHFL 与当前设备的连接
STUHFL_F_GetCtx	获取当前附加设备的设备描述表
STUHFL_F_Reset	复位当前附加设备

4.1.2 参数存取

下表概述了参数存取功能。

表 8. STUHFL 参数存取功能

功能	说明
STUHFL_F_SetParam	通用参数设置功能，用于设置配置值
STUHFL_F_GetParam	通用参数获取功能，用于获取配置值
STUHFL_F_GetParamInfo	获取参数信息
STUHFL_F_SetMultipleParams	设置包含多个参数的列表
STUHFL_F_GetMultipleParams	获取包含多个参数的列表

4.1.3 数据交换

下表概述了数据交换功能。

表 9. STUHFL 数据交换功能

功能	说明
STUHFL_F_SendCmd	向设备发送指令
STUHFL_F_ReceiveCmdData	接收给设备的指令
STUHFL_F_ExecuteCmd	在一次调用中组合发送和接收
STUHFL_F_GetRawData	从设备接收原始数据

4.1.4 通用维护

下表概述了通用维护功能。

表 10. **STUHFL 维护功能**

功能	说明
STUHFL_F_GetVersion	获取设备版本信息
STUHFL_F_GetInfo	获取设备信息
STUHFL_F_Upgrade	固件升级
STUHFL_F_EnterBootloader	重新启动并进入自举程序
STUHFL_F_Reboot	重启固件

4.2 会话层

4.2.1 STUHFL 的 Gen2V2 功能

下表概述了 gen2V2 功能。

表 11. **STUHFL Gen2V2 功能**

功能	说明
STUHFL_F_Gen2_Inventory	Gen2 Inventory 取决于当前库存和 gen2 配置
STUHFL_F_Gen2_Select	Gen2 Select 指令用于选择或过滤 Gen2 应答器
STUHFL_F_Gen2_Read	Gen2 Read 指令用于从 Gen2 应答器读取信息
STUHFL_F_Gen2_Write	Gen2 Write 指令用于向 Gen2 应答器写入数据
STUHFL_F_Gen2_Lock	Gen2 Lock 指令用于锁定 Gen2 应答器
STUHFL_F_Gen2_Kill	Gen2 Kill 指令用于终止 Gen2 应答器。
STUHFL_F_Gen2_GenericCmd	通用 Gen2 位交换
STUHFL_F_Gen2_QueryMeasureRssi	Gen2 Query 指令执行过程中的 RSSI 测量

4.2.2 STUHFL 的 GB/T 29768

下表概述了 GB/T 29768 功能。

表 12. **STUHFL GB/T 29768 功能**

功能	说明
STUHFL_F_Gb29768_Inventory	GB/T 29768 Inventory 取决于当前库存和 GB/T 29768 配置
STUHFL_F_Gb29768_Sort	GB/T 29768 Sort 指令用于选择或过滤 GB/T 29768 应答器
STUHFL_F_Gb29768_Read	GB/T 29768 Read 指令用于从 GB/T 29768 应答器读取信息
STUHFL_F_Gb29768_Write	GB/T 29768 Write 指令用于向 GB/T 29768 应答器写入数据
STUHFL_F_Gb29768_Lock	GB/T 29768 Lock 指令用于锁定 GB/T 29768 应答器
STUHFL_F_Gb29768_Kill	GB/T 29768 Kill 指令用于灭活 GB/T 29768 应答器。
STUHFL_F_Gb29768_Erase	GB/T 29768 Erase 指令用于擦除 GB/T 29768 应答器的内容。

4.3 活动层

4.3.1 操作

下表概述了操作功能。

表 13. **STUHFL 操作功能**

功能	说明
STUHFL_F_Start	启动前台或后台操作
STUHFL_F_Stop	开始操作

4.4 STUHFL EVAL API 封装器

4.4.1 配置

下表概述了配置功能。

表 14. **STUHFL EVAL API 配置功能**

功能	说明
SetRegister	将值写入 ST25RU3993 寄存器的地址信息中
GetRegister	读取一个 ST25RU3993 寄存器地址，并将值回复给主机
SetRwdCfg	设置读写器配置
GetRwdCfg	获取读写器配置
SetAntennaPower	设置天线功率
GetAntennaPower	获取天线功率
SetGen2Cfg	设置 Gen2 配置
SetTxRxCfg	设置 TxRx 配置
SetPA_Cfg	设置功率放大器配置
SetInventoryCfg	设置通用库存配置
GetGen2Cfg	获取 Gen2 配置
GetGBT29768Cfg	获取 GBT29768 配置
GetTxRxCfg	获取 TxRx 配置
GetPA_Cfg	获取功率放大器配置
GetInventoryCfg	获取通用库存配置

4.4.2 频率

下表概述了频率配置功能。

表 15. **STUHFL EVAL API 频率设置功能**

功能	说明
SetFreqProfile	设置频率配置文件
SetFreqProfileCustomAppend	设置配置文件的附加频率
SetFreqHop	设置跳频时间
SetFreqLBT	设置“先听后说”配置
SetFreqCondMod	设置连续调制配置
GetFreqRSSI	获取频率 RSSI
GetFreqReflectedPower	获取配置文件的附加频率
GetFreqProfileInfo	获取跳频时间
GetFreqHop	获取“先听后说”配置
GetFreqLBT	获取连续调制配置

4.4.3 调谐

下表概述了调谐功能。

表 16. **STUHFL EVAL API 调谐功能**

功能	说明
SetTuning	设置 Cin、Clen 和 Cout
SetTuningTableEntry	设置调谐表条目
SetTuningTableDefault	恢复为默认调谐
SetTuningTableSave2Flash	将当前配置的调谐表保存到闪存
SetTuningTableEmpty	删除调谐表条目
GetTuning	获取 Cin、Clen 和 Cout 配置
GetTuningTableEntry	获取调谐表条目
GetTuningTableInfo	设置当前调谐表信息
调谐	开始调谐

4.4.4 Gen2V2

下表概述了 gen2V2 功能。

表 17. **STUHFL EVAL API Gen2V2 功能**

功能	说明
Gen2_Inventory	Gen2 Inventory 取决于当前库存和 gen2 配置
Gen2_Select	Gen2 Select 指令用于选择或过滤 Gen2 应答器
Gen2_Read	Gen2 Read 指令用于从 Gen2 应答器读取信息
Gen2_Write	Gen2 Write 指令用于向 Gen2 应答器写入数据
Gen2_Lock	Gen2 Lock 指令用于锁定 Gen2 应答器
Gen2_Kill	Gen2 Kill 指令用于终止 Gen2 应答器。

功能	说明
Gen2_GenericCmd	通用 Gen2 位交换
Gen2_QueryMeasureRssi	Gen2 Query 指令执行过程中的 RSSI 测量

4.4.5 GB/T 29768

下表概述了 GB/T 29768 功能。

表 18. STUHFL EVAL API GB/T 29768 功能

功能	说明
Gb29768_Inventory	GB/T 29768 Inventory 取决于当前库存和 GB/T 29768 配置
Gb29768_Sort	GB/T 29768 Sort 指令用于选择或过滤 GB/T 29768 应答器
Gb29768_Read	GB/T 29768 Read 指令用于从 GB/T 29768 应答器读取信息
Gb29768_Write	GB/T 29768 Write 指令用于向 GB/T 29768 应答器写入数据
Gb29768_Lock	GB/T 29768 Lock 指令用于锁定 GB/T 29768 应答器
Gb29768_Kill	GB/T 29768 Kill 指令用于灭活 GB/T 29768 应答器。
Gb29768_Erase	GB/T 29768 Erase 指令用于擦除 GB/T 29768 应答器的内容。

4.4.6 Inventory runner

下表概述了 inventory runner 功能。

表 19. STUHFL EVAL API inventory runner 功能

功能	说明
InventoryRunnerStart	启动 inventory runner
InventoryRunnerStop	停止当前 inventory runner

版本历史

表 20. 文档版本历史

日期	版本	变更
2019 年 9 月 17 日	1	初始版本。
2019 年 11 月 26 日	2	更新了文档标题和第 引言。 对整个文档进行少量文字修订。
2020 年 6 月 3 日	3	更新了： • 第 引言, 第 1.2 节 特征, 第 1.3 节 硬件要求, 第 2.3 节 ST UHF 库封装器 • 图 2. ST UHFL 库系统架构 • 第 4.2.2 节 STUHFL 的 GB/T 29768 的标题和 第 4.4.5 节 GB/T 29768 • 表 12. STUHFL GB/T 29768 功能， 表 18. STUHFL EVAL API GB/T 29768 功能 删除了表 1。硬件要求

目录

表一览

图一览