



Realtek iOS Simple Configure Wizard Guide

Realtek iOS Simple Configure Wizard Guide

Date: 2016/05/03

Version: 2.0.5

This document is subject to change without notice. The document contains Realtek confidential information and must not be disclosed to any third party without appropriate NDA.

1. INTRODUCTION.....1

2. SOURCE CODE LOCATION AND DESCRIPTION2

2.1 SIMPLE CONFIG MAIN APPLICATION2

2.2 PROVIDING RESOURCES.....2

2.3 SIMPLE CONFIG LIBRARY2

3 SIMPLE CONFIG LIBRARY3

3.1 DESCRIPTION3

3.2 KEY STRUCTURE.....3

3.3 SIMPLECONFIG (EXTERNAL API)4

4 SIMPLE CONFIG WORKING FLOW6

4.1 DEVICE CONFIGURATION6

4.2 DEVICE DISCOVERY6

4.3 DEVICE CONTROL7

5 REFERENCE.....8

1. Introduction

This is the document describes how to use Realtek iOS Simple Config Wizard APP to configure WiFi Speaker and introduce simple config API.

2. Source Code Location and Description

2.1 Simple Config main application

SimpleConfig*: main package

2.2 Providing Resources

SimpleConfig\Resource: Bitmap files

2.3 Simple Config library

SimpleConfig\Lib\SimpleConfigLib: Simple config library

SimpleConfig\Lib\ZBarSDK: Library for QRCode scanner

3 Simple Config Library

3.1 Description

Simple Config for IOS contains two major interfaces: **PatternBase** and **SimpleConfig**.

PatternBase is the underlying class which implements Pattern 2 、Pattern 3 and Pattern 4.

SimpleConfig is an API, which inherits from PatternBase. SimpleConfig supplies developers with external APIs for Simple Config further development.

3.2 Key Structure

3.2.1 Device Information

To record device information, the following structure is defined.

```
struct dev_info{
    unsigned char    status;                //BIT(0):connected BIT(1):profile saved
    unsigned short dev_type;                //device type, e.g.: refrigerator
    unsigned char    mac[6];                //device MAC address
    unsigned int     ip;                    //device IP address
    unsigned char    extra_info[64];        //device name
    unsigned char    require_pin;           //1-require PIN, 0-no need for PIN
};
```

3.2.2 Operation Mode

```
typedef enum{
    MODE_INIT = 0,        //initial mode
    MODE_CONFIG,          //start to configure
    MODE_WAIT_FOR_IP,     //got the first ACK packet from DUT, which containing with IP value 0
    MODE_DISCOVER,        //device discovery mode
    MODE_CONTROL,         //device control mode
    MODE_ALERT,           //alert mode
}PatternModes;
```

3.3 SimpleConfig (External API)

All configuration pattern class are derived from PatternBase class. The following table shows external APIs defined in PatternBase, which must all be overwritten in its derived class.

SimpleConfig supplies the API to developers.

Member Functions of Interface PatternFactory(A): Device Configure	
API	Description
- (id) init: (unsigned int)pattern_flag	Initial function. To initial this pattern class.
- (int) rtk_pattern_build_profile: (NSString *)ssid psw: (NSString *)password pin: (NSString *)pin	Build Wi-Fi profile about to send, including SSID, password and possible PIN code.
- (int) rtk_pattern_send: (NSNumber *)times	Send Wi-Fi profile multiple times. Times: send times.
- (void) rtk_pattern_stop	Stop Wi-Fi profile
- (void) rtk_sc_close_sock	Close I/O socket. Usage: When switching configuration pattern, this function must be called before start the new pattern class.
- (NSMutableArray *) rtk_pattern_get_config_list	Return device list that are successfully configured. This functions returns a NSMutableArray containing values with structure dev_info, which is introduced in 3.2.1.

Member Functions of Interface PatternFactory(B): Device Control	
-(void)rtk_sc_build_scan_data: (unsigned int)security_level	Generate device discover UDP packets
-(int)rtk_sc_start_scan	Send scan data
-(NSMutableArray *)rtk_sc_get_scan_list	Return device list that are successfully configured. This functions returns a NSMutableArray containing values with structure dev_info, which is introduced in 3.2.1.
-(void)rtk_sc_gen_control_data: (unsigned int)type pin: (NSString *)pin name: (NSString *)name	Generate device control UDP packets, whose control type is control_type

-(int) rtk_sc_send_control_data: (unsigned int)ip	Send device control packets data to ip using UDP. (Unicast)
--	--

4 Simple Config Working Flow

Simple Config can be used to:

1. Configure a client device;
2. Discover devices;
3. Control devices, including delete profile and rename device.

The working flow of each function will be introduced in this chapter, so that developer can call API correctly.

4.1 Device configuration

The working flow of device configure is shown as Fig 4-1.

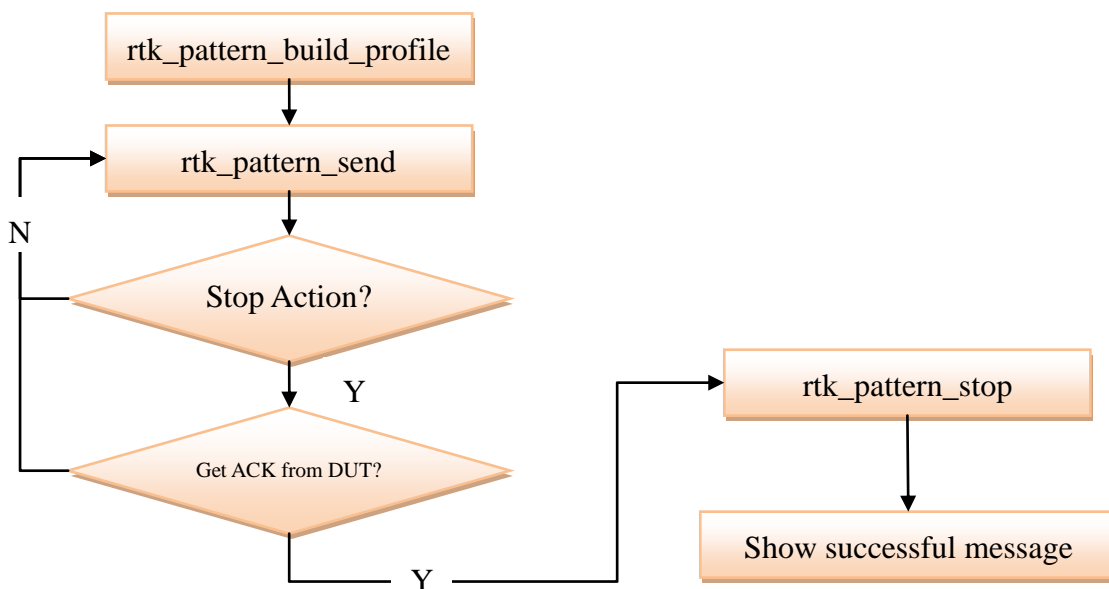


Figure 4-1 Device Configure working flow

4.2 Device discovery

The working flow of device configure is shown as Fig 4-2.

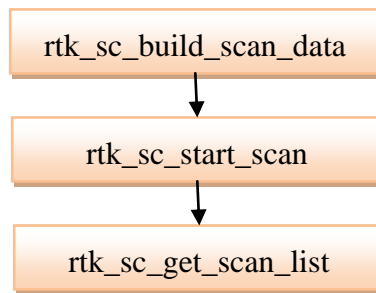


Figure 4-2 Device discovery working flow

4.3 Device control

Device control includes two parts: delete profile and rename device. Rename device requires user to input device new name. So rename device has a different API with the former two.

This general working flow is shown as Fig 4-3.

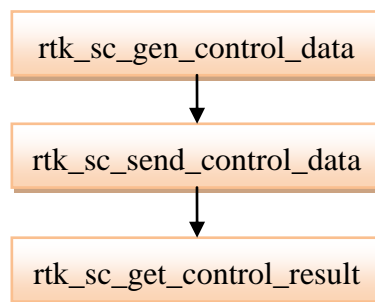


Figure 4-3 Delete profile working flow

Rename device working flow is shown in Fig 4-4.

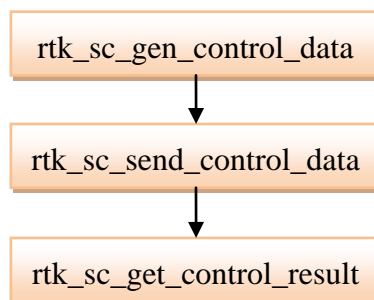


Figure 4-4 Rename device working flow

Also developers can use API *rtk_sc_get_control_result* to get the control result.

5 Reference

N/A