GR551x ANCS Profile Example Application

Version: 1.9

Release Date: 2022-02-20

Shenzhen Goodix Technology Co., Ltd.

Trademarks and Permissions

GOODiX and other Goodix trademarks are trademarks of Shenzhen Goodix Technology Co., Ltd. All other trademarks and trade names mentioned in this document are the property of their respective holders.

Disclaimer

Information contained in this document is intended for your convenience only and is subject to change without prior notice. It is your responsibility to ensure its application complies with technical specifications.

Shenzhen Goodix Technology Co., Ltd. (hereafter referred to as “Goodix”) makes no representation or guarantee for this information, express or implied, oral or written, statutory or otherwise, including but not limited to representation or guarantee for its application, quality, performance, merchantability or fitness for a particular purpose. Goodix shall assume no responsibility for this information and relevant consequences arising out of the use of such information.

Without written consent of Goodix, it is prohibited to use Goodix products as critical components in any life support system. Under the protection of Goodix intellectual property rights, no license may be transferred implicitly or by any other means.

Shenzhen Goodix Technology Co., Ltd.

Headquarters: 2F. & 13F., Tower B, Tengfei Industrial Building, Futian Free Trade Zone, Shenzhen, China

TEL: +86-755-33338828 FAX: +86-755-33338099

Website: www.goodix.com

Preface

Purpose

This document introduces how to use and verify the ANCS example in the GR551x Software Development Kit (SDK), to help users quickly get started with secondary development.

Audience

This document is intended for:

- GR551x user
- GR551x developer
- GR551x tester
- iOS developer
- Hobbyist developer
- Technical writer

Release Notes

This document is the seventh release of *GR551x ANCS Profile Example Application*, corresponding to GR551x System-on-Chip (SoC) series.

Revision History

Version	Date	Description
1.0	2019-12-08	Initial release
1.3	2020-03-16	Updated the release time in the footers.
1.5	2020-05-30	Updated code in “Interaction Process and Major Code”.
1.6	2020-06-30	• Modified the pin code in “Bluetooth Connection” and “Connection, Pairing, and Bonding”. • Updated the code in “Control Command”.
1.7	2021-04-20	Optimized descriptions in “Initial Operation” and “Application Details”.
1.8	2021-08-06	Changed the section "Supported Development Platform" into "Preparation".
1.9	2022-02-20	Modified the file name of the example firmware based on SDK changes.

Contents

1 Introduction

Apple Notification Center Service (ANCS) is applied to intelligent Bluetooth-enabled devices, such as wristbands and smart watches that connect to iOS devices. Through a Bluetooth Low Energy (Bluetooth LE) link, the Bluetooth devices can access media notifications from iOS devices, and send ANCS-related control commands to iOS devices.

This document introduces how to implement ANCS Client based on a GR551x System-on-Chip (SoC).

Before getting started, you can refer to the following documents.

Table 1-1 Reference documents

Name	Description
ANCS Specification	Provides ANCS protocols. Available at Apple Notification Center Service (ANCS) Specification.
GR551x Developer Guide	Introduces GR551x Software Development Kit (SDK) and how to develop and debug applications based on the SDK.
Bluetooth Core Spec	Offers official Bluetooth standards and core specification from Bluetooth SIG.
Bluetooth GATT Spec	Provides details about Bluetooth profiles and services. Available at http://www.bluetooth.com/specifications/gatt
J-Link/J-Trace User Guide	Provides J-Link operational instructions. Available at http://www.segger.com/downloads/jlink/UM08001_JLink.pdf..
Keil User Guide	Offers detailed Keil operational instructions. Available at http://www.keil.com/support/man/docs/uv4/.

2 Profile Overview

The ANCS Profile defines two device roles:

1. Server: iOS devices serve as the Central, providing services and data sources.
2. Client: Bluetooth devices serve as the Peripheral capable of detecting services from iOS devices (the Central) as well as reading and writing data after being connected to an iOS device.

The interaction process between the Server and the Client is presented in the figure below:

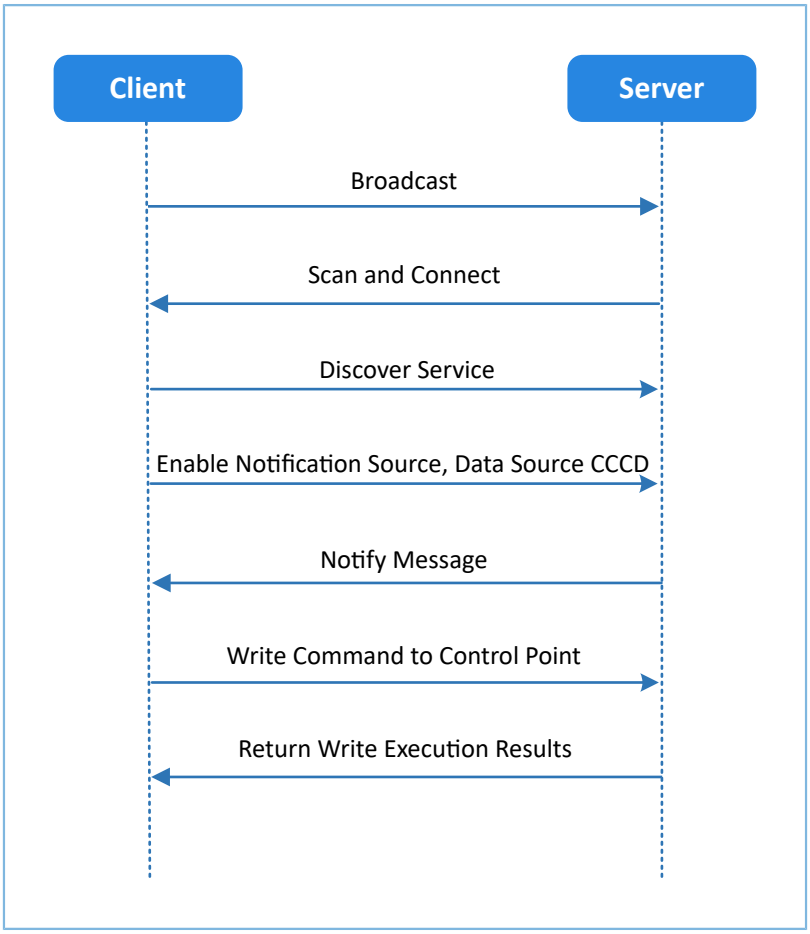

Figure 2-1 Client-Server interaction process

ANCS includes three characteristics as below.

Table 2-1 ANCS characteristics

Characteristic	UUID	Type	Support	Security	Property
Notification Source	9FBF120D-6301-42D9-8C58-25E699A21DBD	128 bits	Mandatory	None	Notify
Control Point	69D1D8F3-45E1-49A8-9821-9BBDFDAAD9D9	128 bits	Mandatory	None	Write
Data Source	22EAC6E9-24D6-4BB5-BE44-B36ACE7C7BFB	128 bits	Mandatory	None	Notify

3 Initial Operation

This chapter introduces how to rapidly verify an ANCS Client example in the GR551x SDK.

Note:

SDK_Folder is the root directory of GR551x SDK.

3.1 Preparation

Perform the following tasks before running an ANCS example.

- **Hardware preparation**

Table 3-1 Hardware preparation

Name	Description
J-Link debug probe	JTAG emulator launched by SEGGER. For more information, visit www.segger.com/products/debug-probes/j-link/.
Development board	GR5515 Starter Kit Board (SK Board)
Connection cable	Micro USB 2.0 cable
iOS device	Any iOS device supporting Bluetooth LE 4.0 and later, such as iPhone 4S and iPad 3

- **Software preparation**

Table 3-2 Software preparation

Name	Description
Windows	Windows 7/Windows 10
J-Link driver	A J-Link driver. Available at www.segger.com/downloads/jlink/.
Keil MDK5	An integrated development environment (IDE). MDK-ARM Version 5.20 or later is required. Available at www.keil.com/download/product/.
GProgrammer (Windows)	A programming tool. Available in SDK_Folder\tools\GProgrammer.
GRUart (Windows)	A serial port debugging tool. Available in SDK_Folder\tools\GRUart.

3.2 Firmware Programming

The source code of the ANCS example is in SDK_Folder\projects\ble\ble_peripheral\ble_app_ancs_c.

You can programme *ble_app_ancs_c.bin* to the SK Board through GProgrammer. For details, see *GProgrammer User Manual*.

Note:

- The *ble_app_ancs_c.bin* is in `SDK_Folder\projects\ble\ble_peripheral\ble_app_ancs_c\build\`.

3.3 Test and Verification

Follow the steps below to test the ANCS example:

1. Establish connection.

   Power the SK Board on. Turn on **Bluetooth** on an iOS device to scan nearby Bluetooth devices. The device discovers an SK Board with an advertising name of **Goodix_ANCS_C** as shown in Figure 3-1.

Note:

This document is based on tests on an iPhone 7 running on iOS 11.03. The interface can be different depending on the device and operating system in use.

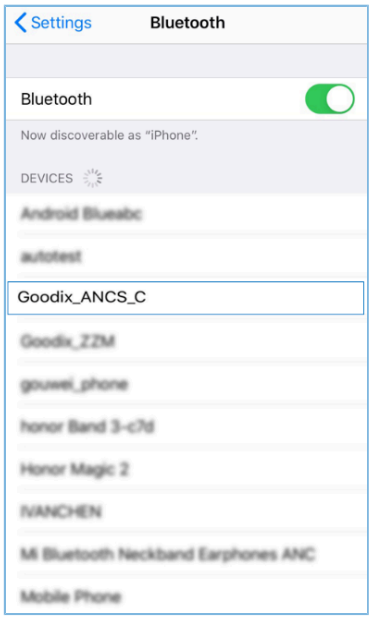

Figure 3-1 Discovering **Goodix_ANCS_C**

Tap **Goodix_ANCS_C** to connect the device to the SK Board. As a pairing request box pops up as below, enter **123456**, and tap **Pair**.

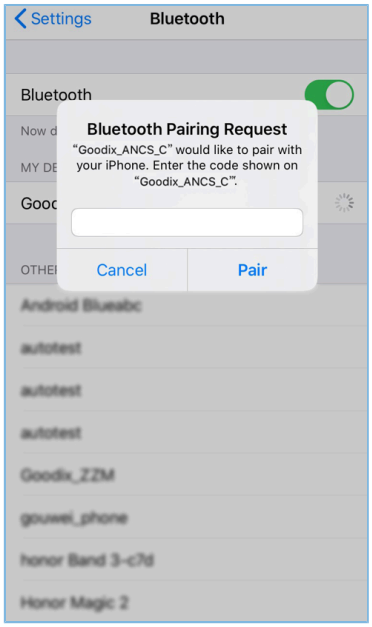

Figure 3-2 Entering pairing password

After pairing, **Goodix_ANCS_C** displays as **Connected** under **MY DEVICES**.

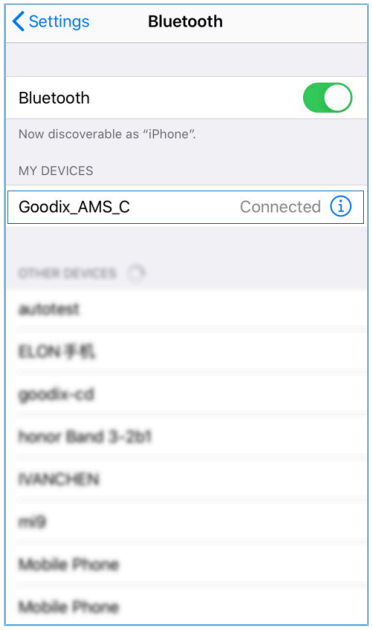

Figure 3-3 Successful pairing

2. Test and Verification

   Users can verify whether ANCS runs normally according to serial port printing information on GRUart.

   This section describes how to verify ANCS operations by taking a notification as an example, as shown in the figure below. For details, refer to *Apple Notification Center Service (ANCS) Specification*.

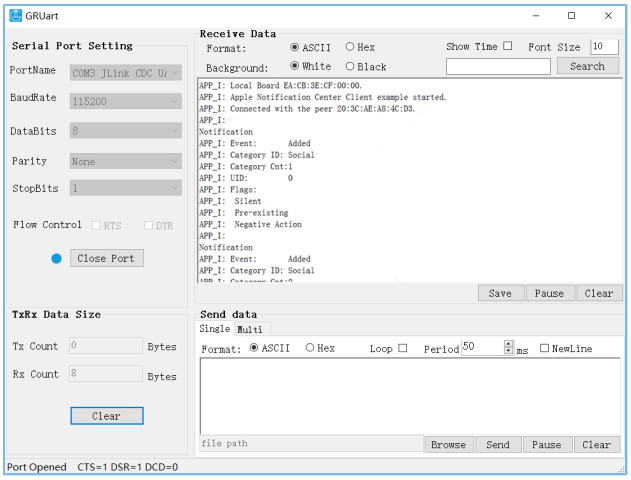

Figure 3-4 Serial port printing information on GRUart

Output information from serial ports is described as below:

Table 3-3 Notification description

Name	Description
Notification	Indicates this is a notification
Event: Added	Event type: Added
Category ID: Social	Information category: Social
Category Cnt: 1	Type quantity: 1
UID: 0	Unique identifier (UID) of the notification: 0
Flags: Silent	Information type: Silent
Pre-existing	Exists in buffer
Negative Action	Indicates delete operation is allowed

Take making and answering a phone call for example. Make a phone call on another phone to the tested iPhone 7. When the call is put through, the tested iPhone 7 sends a notification to the SK Board immediately; the board then processes the received notification, and the ANCS Client example prints information on GRUart.

To check the incoming call number, users can press **OK** on the board, and GRUart displays the following information (including the incoming call number).

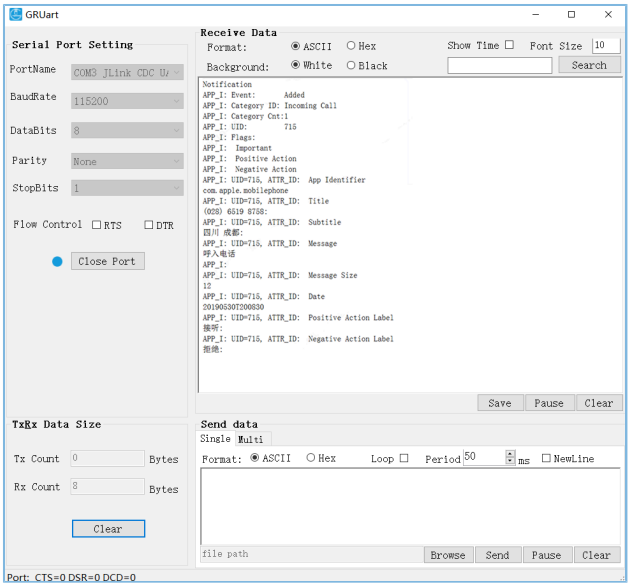

Figure 3-5 Printed information on GRUart

To answer the phone call, press **RIGHT** on the board, then the call is connected. To decline the phone call, press **LEFT**.

Note:

For more information about buttons on an SK Board, see *GR551x Starter Kit User Guide*.

4 Application Details

This chapter introduces the running procedures and major code of the ANCS Client example.

4.1 Running Procedures

After proper running, the ANCS Client example successively performs advertising, pairing and bonding, ANCS discovery, Client Characteristic Configuration Descriptor (CCCD) enablement, notification handling, as well as interactive command operations. Using ANCS Client as an example, this section elaborates on the interaction process between an ANCS Server and an ANCS Client, as illustrated in the figure below.

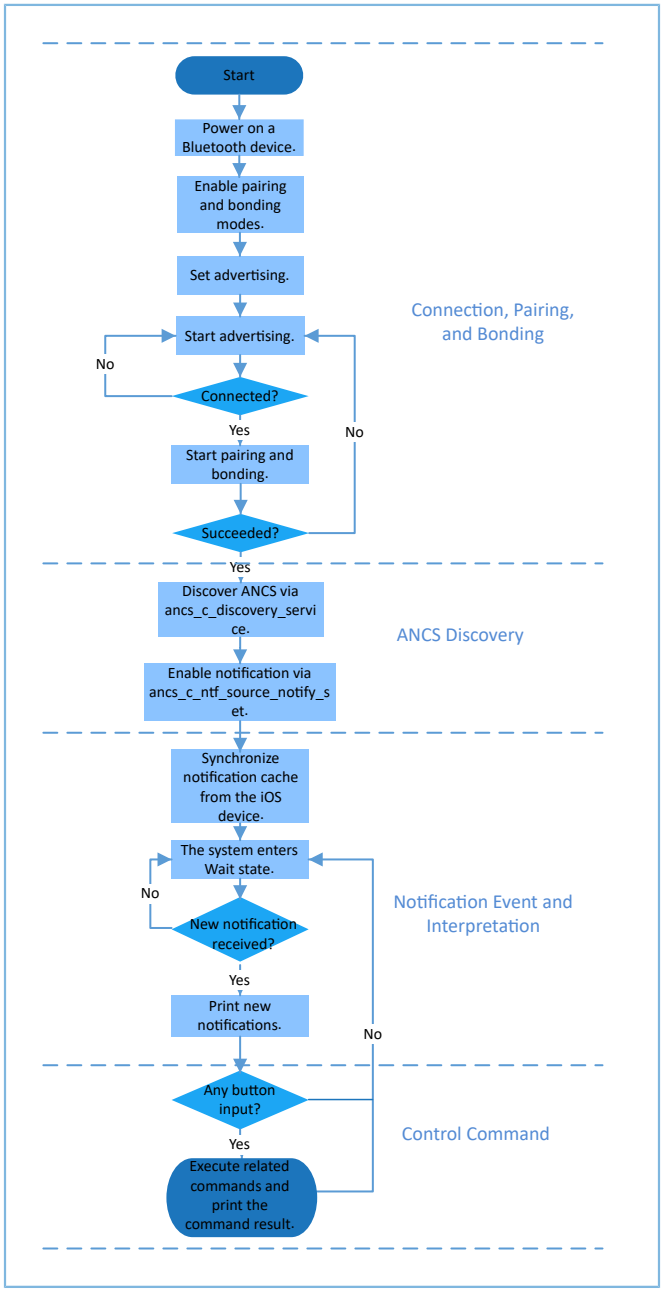

Figure 4-1 ANCS Client-Server interaction process

4.2 Major Code

Write the control information to the Control Point of ANCS Server via ANCS Client. Details about a particular notification can be retrieved from the Data Source returned from the iOS device. In following subsections, relevant control commands are described in detail.

4.2.1 Accessing Notification Attributes

Path: `gr_profiles\ancs_protocol.h` under the project directory

Name: *ancs_protocol.h*

The macro listed below helps to access alternative values of the notification attributes.

```
typedef enum
{
    ANCS_NOTIF_ATTR_ID_APP_IDENTIFIER = 0,     /**< Identifies that the
    attribute data is of an "App Identifier" type. */
    ANCS_NOTIF_ATTR_ID_TITLE,                  /**< Identifies that the
    attribute data is a "Title". */
    ANCS_NOTIF_ATTR_ID_SUBTITLE,               /**< Identifies that the
    attribute data is a "Subtitle". */
    ANCS_NOTIF_ATTR_ID_MESSAGE,                /**< Identifies that the
    attribute data is a "Message". */
    ANCS_NOTIF_ATTR_ID_MESSAGE_SIZE,           /**< Identifies that the
    attribute data is a "Message Size". */
    ANCS_NOTIF_ATTR_ID_DATE,                   /**< Identifies that the
    attribute data is a "Date". */
    ANCS_NOTIF_ATTR_ID_POSITIVE_ACTION_LABEL,  /**< The notification has a
    "Positive action" that can be executed associated with it. */
    ANCS_NOTIF_ATTR_ID_NEGATIVE_ACTION_LABEL,  /**< The notification has a
    "Negative action" that can be executed associated with it. */
} ancs_notification_attr_t;
```

Path: `gr_profiles\ancs_protocol.c` under the project directory

Name: ancs_notify_attr_get();

This function helps to access the corresponding notification attribute according to its UID. Take an e-mail sent from the iOS device for example. The ANCS Client example is capable of inquiring the detailed contents, receiving time, and the sender of the e-mail via this function.

```
void ancs_notify_attr_get(int uid, char noti_attr)
{
    int len = 0;
    uint8_t buf[8];
    buf[0] = CTRL_POINT_GET_NTF_ATTRIBUTE;
    memcpy(&buf[1], &uid, 4);
    buf[5] = noti_attr;
    len = CFG_ANCS_ATTRIBUTE_MAXLEN;
    buf[6] = (len & 0xff);
    buf[7] = (len>>8) & 0xff;
    ancs_c_write_control_point(0, buf, 8);
}
```

4.2.2 Execution

Path: `gr_profiles\ancs_protocol.h` under the project directory

Name: *ancs_protocol.h*

Two alternatives are available for users on the operation for each notification: 0 indicates agreement, and 1 indicates rejection.

```
typedef enum
{
    ACTION_ID_POSITIVE = 0,                  /**< Positive action. */
    ACTION_ID_NEGATIVE                       /**< Negative action. */
} ancs_c_action_id_t;
```

Path: `gr_profiles\ancs_protocol.c` under the project directory

Name: ancs_action_perform();

This function is used to process notifications.

```
void ancs_action_perform(int uid, int action)
{
    uint8_t buf[6];
    buf[0] = CTRL_POINT_PERFORM_NTF_ACTION;
    memcpy(&buf[1], &uid, 4);
    buf[5] = action;
    ancs_c_write_control_point(0, buf, 6);
}
```

4.2.3 Interaction

To help users perform an interaction test on ANCS Client, this example implements the button-based commands, enabling users to operate on Control Point by pressing specific buttons.

Path: `user_app\user_gui.c` under the project directory

Name: app_key_evt_handler();

The following functions provide the process in which the response is triggered by using buttons on the SK Board. When users press specific buttons, the ANCS Client example generates corresponding interaction commands.

Functionalities of each button are presented below:

- **OK**: Obtains and prints attribute values in various types. Such attribute values include details about text messages and e-mails, and the sending time.
- **RIGHT**: Represents Yes or agree. When there is an incoming call, Yes means answering the call.
- **LEFT**: Represents No or decline. When there is an incoming call, No means declining the call.

For detailed test methods for commands, refer to *Apple Notification Center Service (ANCS) Specification*.

```
void app_key_evt_handler(uint8_t key_id, app_key_click_type_t key_click_type)
{
    uint16_t uid;
    if (key_click_type == APP_KEY_SINGLE_CLICK)
    {
        if (BSP_KEY_OK_ID == key_id)
        {
            pwr_mgmt_mode_set(PMR_MGMT_IDLE_MODE);
            uid = ancs_get_uid();
            if (uid > 0)
            {
```

```
                ancs_notify_attr_get(uid, ANCS_NOTIF_ATTR_ID_TITLE);
                ancs_notify_attr_get(uid, ANCS_NOTIF_ATTR_ID_MESSAGE);
            }
        }
        else if (BSP_KEY_LEFT_ID == key_id)
        {
            APP_LOG_INFO("pressed key left");
            uid = ancs_get_uid();
            if (uid > 0)
            {
                ancs_action_perform(uid, ACTION_ID_NEGATIVE);
            }
        }
        else if (BSP_KEY_RIGHT_ID == key_id)
        {
            APP_LOG_INFO("pressed key right");
            uid = ancs_get_uid();
            if (uid > 0)
            {
                ancs_action_perform(uid, ACTION_ID_POSITIVE);
            }
        }
    }
}
```

5 FAQ

This chapter describes problems, reasons, and solutions when verifying and using the ANCS Client example.

5.1 Why Is There No Output Information from GRUart?

- Description

  No printed information displays on GRUart, or GRUart encounters garbled printing.

- Analysis

  The *ble_app_ancs_c.bin* firmware is not programmed on the board correctly, or the **BaudRate** on the GRUart is incorrect, resulting in GRUart's failure to print information.

- Solution

  Confirm on GRUart, the **BaudRate** is 115200 with **DataBits** of 8, **StopBits** of 1, None **Parity**, and no **Flow Control**. Confirm the serial ports have been connected correctly.

  If there is nothing wrong with the serial port connection, redo the firmware programming, and ensure no modification has been done on the project, then directly download the firmware to the Bluetooth device using GProgrammer.

5.2 Why Does an iOS Device Fail to Scan Any Bluetooth Advertising from Goodix_ANCS_C?

- Description

  An iOS device with Bluetooth enabled fails to find advertising from **Goodix_ANCS_C**.

- Analysis

  Exceptions occur in the Bluetooth antenna connection or firmware.

- Solution

  1. Check whether the Bluetooth function on the iOS device is enabled. If the Bluetooth function is enabled, check whether the antenna of the GR551x platform is connected successfully.
  2. If the Bluetooth functions properly and the connection is successful, check the hardware problem by downloading the factory default test firmware.

5.3 Why Does an iOS Device Fail to Access Notification after Connection?

- Description

  After being connected to a Bluetooth device, the iOS device cannot receive any notification.

- Analysis

  The Bluetooth function on the iOS device may be turned off. The Bluetooth device may have been connected to the phone previously. Or the notification function on the iOS device is disabled.

- Solution

1. In **Settings** of the iOS device, check whether the Bluetooth device has been connected to the mobile phone before. If connection has been established previously, tap **Goodix_ANCS_C** to **Forget This Device**, and redo the scanning, pairing, and bonding.
2. Ensure the notification function of the iOS device is enabled.