GR551x AMS 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 AMS Client 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 engineer
- Hobbyist developer
- Technical writer

Release Notes

This document is the seventh release of *GR551x AMS 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	Modified the name of Track-Artist entity in “Test and Verification”.
1.5	2020-05-30	Modified the code format in “Application Details”.
1.6	2020-06-30	Updated the document version based on SDK changes.
1.7	2021-04-20	Optimized descriptions in “Initial Operation” and “Application Details”.
1.8	2021-08-09	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

The Apple Media Service (AMS) 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 AMS-related control commands to iOS devices.

This document introduces the approaches to implementing AMS Client based on a GR551x System-on-Chip (SoC) with details shown in running procedures and major code.

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

Table 1-1 Reference documents

Name	Description
Apple Media Service Reference	Offers Apple Media Service specification. Available at Apple Media Service(AMS) 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 https://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 www.keil.com/support/man/docs/uv4/.

2 Profile Overview

The AMS Profile defines two device roles:

- Server: iOS devices serve as the Central, providing services and data sources.
- 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 illustrated in Figure 2-1.

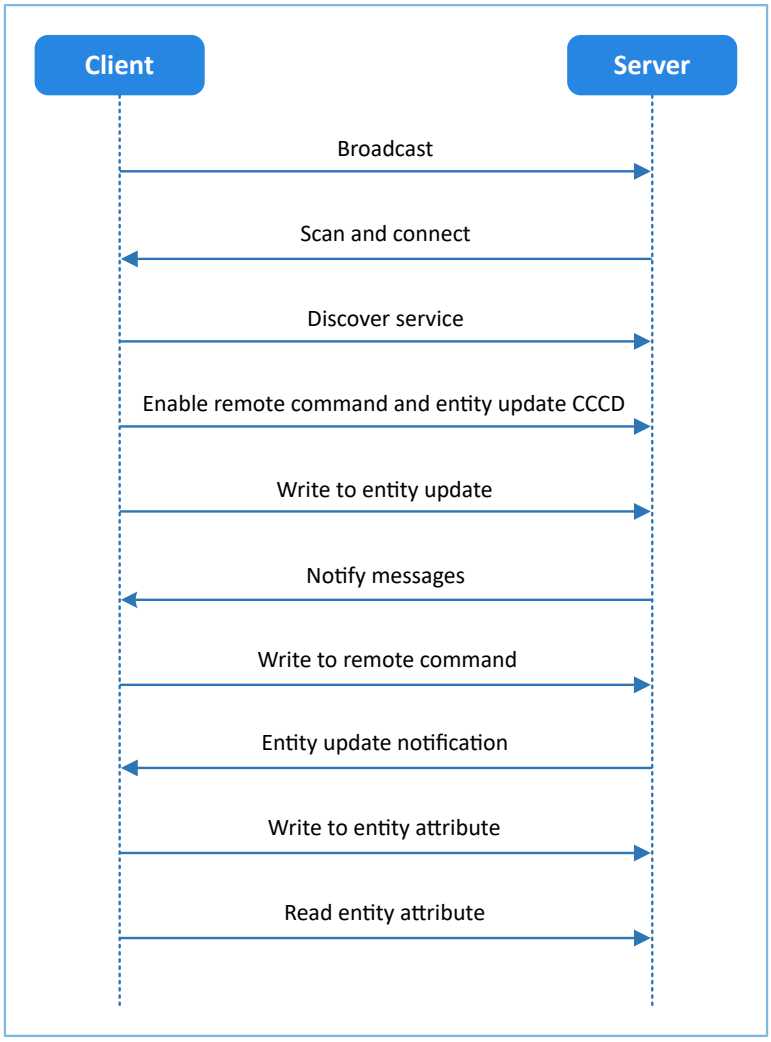

Figure 2-1 Server-Client interaction process

AMS characteristics include Remote Command, Entity Update, and Entity Attribute, as listed in Table 2-1.

Table 2-1 AMS characteristics

Characteristic	UUID	Type	Support	Security	Property
Remote Command	9B3C81D8-57B1-4A8A-B8DF0E56F7CA51C2	128 bits	Mandatory	None	Write and Notify

Characteristic	UUID	Type	Support	Security	Property
Entity Update	2F7CABCE-808D-411F-9A0C-BB92BA96C102	128 bits	Mandatory	None	Write and Notify
Entity Attribute	C6B2F38C-23AB-46D8-A6AB-A3A870BBD5D7	128 bits	Mandatory	None	Write and Read

The role of each AMS characteristic:

- Remote Command: Used to send remote commands and receive updates of available remote commands. The Write property means a Bluetooth device sends a remote command to an iOS device to implement remote control. The Notify property means when the available remote command of the iOS device changes, the iOS device notifies the Bluetooth device of the updated available remote command.
- Entity Update: Used to set a focused entity and receive updates from the focused entity. The Write property is used to set a focused entity at the Bluetooth device. The Notify property is used when the focused entity of an iOS device changes, the iOS device notifies a Bluetooth device of the updated value of the focused entity (the value may be incomplete due to maximum transmission unit/MTU limit).
- Entity Attribute: Used to set and read display entity. The Write property is used to set a display entity on a Bluetooth device, and the Read property is used to read the complete value of the display entity from an iOS device.

3 Initial Operation

This chapter introduces how to run an AMS Client example for the first time by using GR551x Starter Kit Board (SK Board) as an AMS Client and an iOS device as an AMS Server.

Note:

SDK_Folder is the root directory of the GR551x SDK in use.

3.1 Preparation

Perform the following tasks before running the AMS example.

- **Hardware preparation**

Table 3-1 Hardware preparation

Name	Description
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 AMS Client example is in SDK_Folder\projects\ble\ble_peripheral\ble_app_ams_c.

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

Note:

- The *ble_app_ams_c_fw.bin* is in SDK_Folder\projects\ble\ble_peripheral\ble_app_ams_c\build\.

3.3 Test and Verification

Perform AMS communications test on the iOS device and SK Board when all the previous preparations are ready. This document aims to test AMS from two aspects:

- SK Board accesses media notifications from the iOS device.
- SK Board sends AMS control commands to the iOS device.

Users can verify the AMS based on serial port printing information on GRUart. (For more information about AMS, see *Apple Media Service Reference*.)

Follow the steps below to test the AMS 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_AMS_C**, as shown in Figure 3-1.

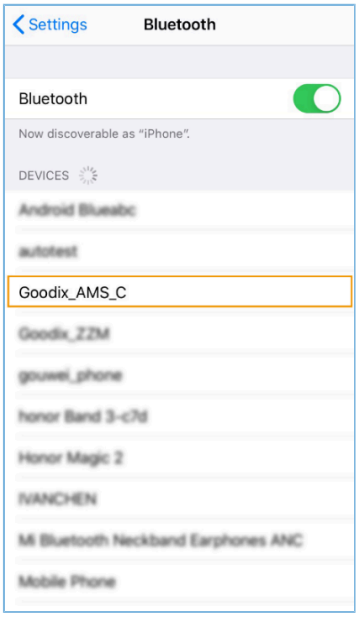

Figure 3-1 Discovering **Goodix_AMS_C**.

Note:

This document is based on tests on an iPhone 6s running on iOS 12.4.1. The interface can be different depending on the device and operating system in use.

   Tap **Goodix_AMS_C** to connect the device to the SK Board. A pairing request dialog pops up, as shown in Figure 3-2. Enter **123456** in the dialog, and tap **Pair** (For methods in setting a pairing password, see app_sec_rcv_enc_req_cb() function descriptions in “Major Code” in *GR551x HID Mouse Example Application*).

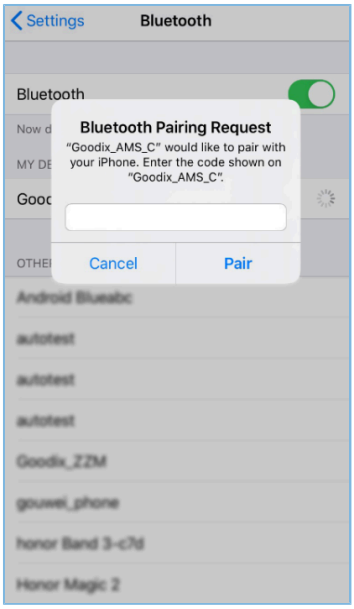

Figure 3-2 Entering pairing password

After pairing, **Goodix_AMS_C** displays as **Connected** under **MY DEVICES**, as shown in Figure 3-3.

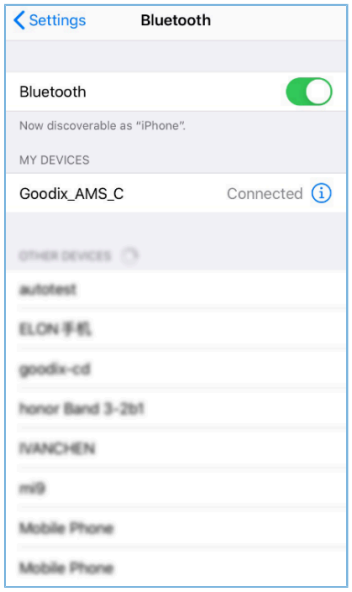

Figure 3-3 Successful pairing

After the iOS device is successfully connected to the SK Board via Bluetooth, the connection information displays on GRUart, as shown in Figure 3-4.

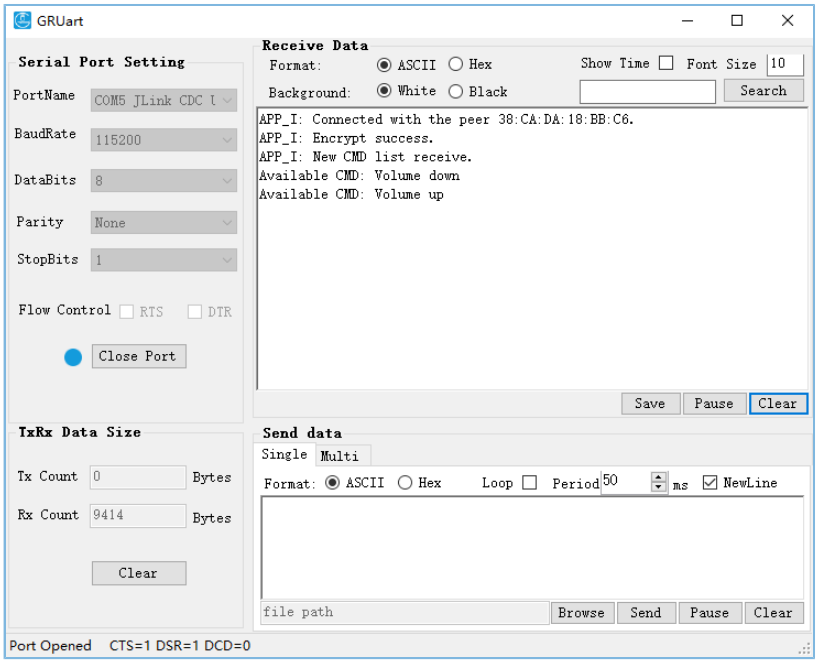

Figure 3-4 Serial port printing information on GRUart

2. Access media notifications from an iOS device.

   Follow the steps below to test how an SK Board accesses media notifications from an iOS device:

   (1). Launch a music player App on the iOS device.

   (2). Play a track on the App. In the test, The Sound Of Silence (By Pat Metheny) is played with the App interface shown in Figure 3-5.

Figure 3-5 Playing music on the iOS device

(3). View the printed information of serial port on GRUart on the PC.

The printed information (shown in Figure 3-6) comprises:

- List of remote commands supported by the music player App
- Update information of focused entities; for example, the SK Board focuses on the media Track-Title and Track-Artist information of the iOS device, so any update on these entities may incur notifications. (For more details about focused entities, refer to *Apple Media Service Reference*.)

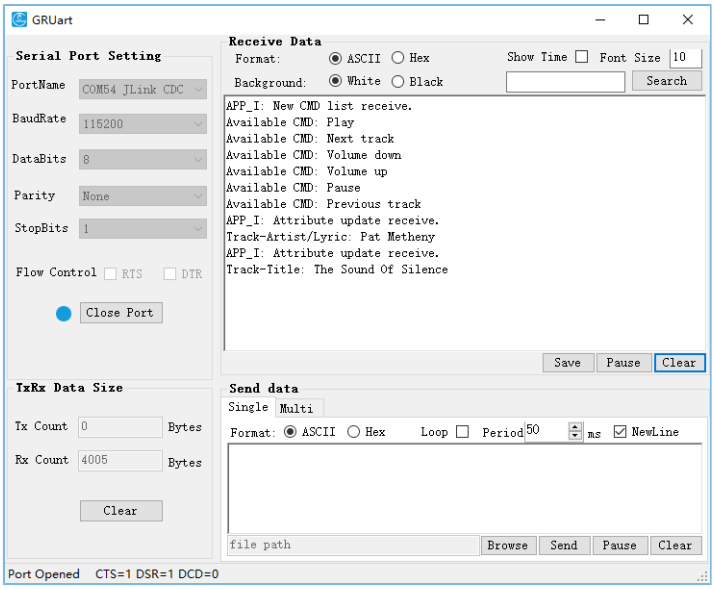

Figure 3-6 Printed information on GRUart when accessing media notifications from an iOS device

Description of printed information on GRUart is as follows:

Table 3-3 Description of printing information on GRUart

Name	Description
APP_I: New CMD list receive	Receive a new list of available remote commands.
Available CMD: Next track	Indicates the `Next track` command is available currently, and ready to send a `Next track` command.
APP_I: Attribute update receive	Receive the new value of the focused entity when the entity changes.
Track-Artist/Lyric: Pat Metheny	The Track-Artist entity changes with a new value of "Pat Metheny".

Note:

According to *Apple Media Service Reference*, the ID of the Track-Artist entity corresponds to the artist of the track currently being played. However, most music player Apps display the lyrics through the ID. Therefore, GRUart displays the ID of the Track-Artist entity as Track-Artist/Lyric.

According to Figure 3-6, the AMS functions properly between the SK Board and the iOS device.

3. Send AMS control commands to an iOS device.

   Follow the steps below to test how the SK Board sends AMS control commands to the iOS device:

   (1). Launch a music player App on the iOS device.

   (2). Play a track on the App.

   (3). Press **RIGHT** on the SK Board (to send a `Next track` command). In this test, Deep in A Dream (By Jim Hall) is used as the next track.

   (4). Check whether the music player App switches to the next track and the printed information on the GRUart.

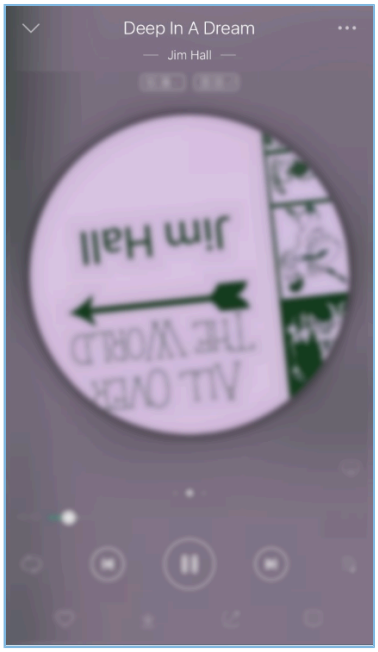

Figure 3-7 Playing music on the iOS device

The printed information (shown in Figure 3-8) comprises:

- Message indicating the remote commands have been successfully delivered
- Up-to-date track name and artist information (indicating the remote command has been executed)

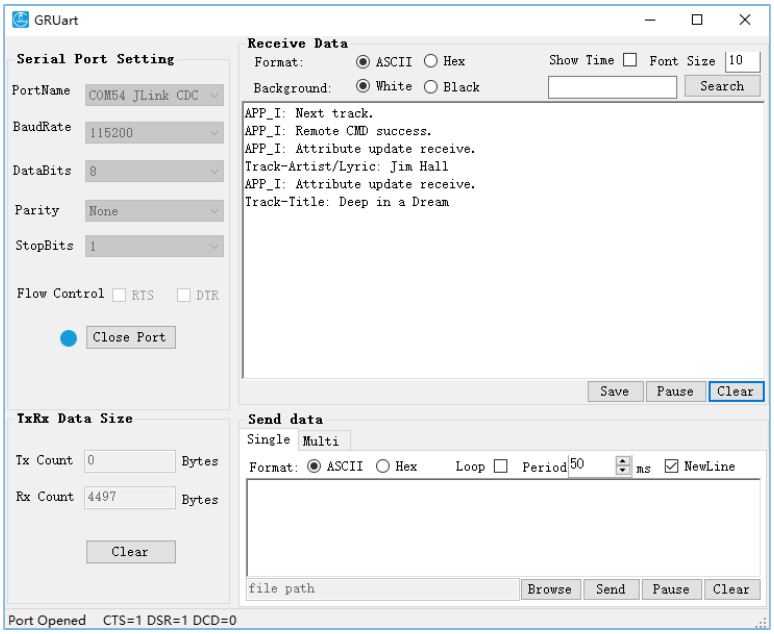

Figure 3-8 Printed information on GRUart

According to Figure 3-8, the AMS functions properly between the SK Board and the iOS device.

Note:

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

Table 3-4 Button-command relations of SK Board

Button	Command
OK (a short press)	Play
OK (two short presses)	Pause
OK (a long press)	Toggle play/pause
RIGHT	Next track
LEFT	Previous track
UP	Volume up
DOWN	Volume down

4 Application Details

This chapter elaborates on the running procedures and major code of the GR551x AMS Client example.

4.1 Running Procedures

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

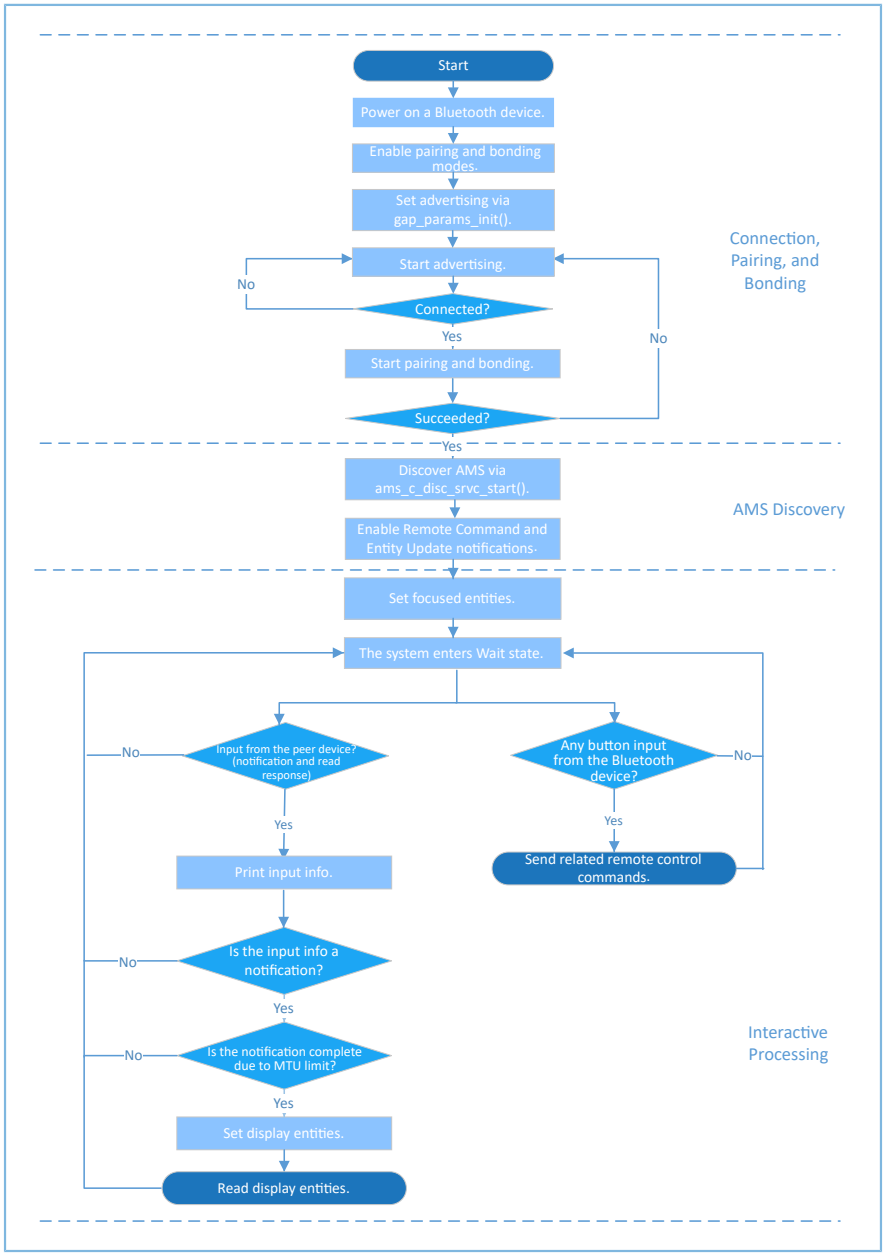

Figure 4-1 AMS Server-Client interaction process

4.2 Major Code

The Bluetooth device (or SK Board in this document) can scan, pair with, and bond to an iOS device; then it discovers AMS and enables the related CCCD. The Bluetooth device can interact remotely with the iOS device by writing information to and reading data from the AMS characteristics of the iOS device. This section introduces major code of read and write interactions supported by AMS.

4.2.1 Performing Remote Command

When performing button-related operations on Bluetooth devices, users can implement remote control of an iOS device by sending commands to the AMS Remote Command characteristic of the iOS device through ams_c_cmd_send(). The command ID indicates the command content. The ams_c_cmd_id_t structure defines the enumeration values of all the remote command IDs.

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

Name: ams_c_cmd_send();

```
sdk_err_t ams_c_cmd_send(uint8_t conn_idx, uint8_t cmd_id)
{
    ...
    gattc_write_attr_value_t write_attr_value;
    write_attr_value.handle = s_ams_c_env.handles.ams_cmd_handle;
    write_attr_value.offset = 0;
    write_attr_value.length = 1;
    write_attr_value.p_value = (uint8_t *)&cmd_id;

    return ble_gattc_prf_write(s_ams_c_env.prf_id, conn_idx, &write_attr_value);
}
```

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

Name: ams_c_cmd_id_t;

```
typedef enum
{
    AMS_CMD_ID_PLAY,                            /**< Command index of play. */
    AMS_CMD_ID_PAUSE,                           /**< Command index of pause. */
    AMS_CMD_ID_TOGGLE_PLAY_PAUSE,               /**< Command index of toggle. */
    AMS_CMD_ID_NEXT_TRACK,                      /**< Command index of next track. */
    AMS_CMD_ID_PREVIOUS_TRACK,                  /**< Command index of previous track. */
    ...
} ams_c_cm
```

4.2.2 Setting a Focused Entity

The Bluetooth device can set focused entities by writing entity IDs to Entity Update (an Entity Update command comprises Entity ID and Attribute ID; for details, see *Apple Media Service Reference*).

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

Name: attr_focus_set();

```
static void attr_focus_set(uint8_t conn_idx)
{
    sdk_err_t   error_code;
    ams_c_ett_attr_id_t track_attr_id =
```

```
    {
        .ett_id     = AMS_TRACK_ID,
        .attr_id    = {AMS_TRACK_ARTIST_ID, AMS_TRACK_TITTLE_ID},
        .attr_count = 2
    };
    error_code = ams_c_attr_focus_set(conn_idx, &track_attr_id);
    APP_ERROR_CHECK(error_code);
}
```

4.2.3 Setting a Display Entity

The Bluetooth device can set a display entity by writing entity IDs to Entity Attribute through ams_c_attr_display_set(). After successful writing, complete entity data can be accessed by reading the Entity Attribute value.

Path: gr_profiles\ams_c.c under the project directory

Name: ams_c_attr_display_set();

```
sdk_err_t ams_c_attr_display_set(uint8_t conn_idx, const ams_c_attr_info_t *p_attr_info)
{
    ...
    gattc_write_attr_value_t write_attr_value;
    write_attr_value.handle = s_ams_c_env.handles.ams_attr_display_handle;
    write_attr_value.offset = 0;
    write_attr_value.length = 2;
    write_attr_value.p_value = (uint8_t *)&(p_attr_info->ett_id);
    return ble_gattc_prf_write(s_ams_c_env.prf_id, conn_idx, &write_attr_value);
}
```

4.2.4 Reading a Display Entity

Limited by MTU, the entity data notified by Entity Update may be truncated. Entity Attribute should be used to obtain complete entity data. After successful writing, complete entity data can be accessed by reading the Entity Attribute value through ams_c_cplt_attr_read().

Path: gr_profiles\ams_c.c under the project directory

Name: ams_c_cplt_attr_read()

```
sdk_err_t ams_c_cplt_attr_read(uint8_t conn_idx)
{
    if (BLE_ATT_INVALID_HDL == s_ams_c_env.handles.ams_attr_display_handle)
    {
        return BLE_ATT_ERR_INVALID_HANDLE;
    }
    return ble_gattc_prf_read(s_ams_c_env.prf_id, conn_idx,
                              s_ams_c_env.handles.ams_attr_display_handle, 0);
}
```

5 FAQ

This chapter describes possible problems, reasons, and solutions during verification and application of the AMS 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_ams_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

  1. Confirm on GRUart, the **BaudRate** is 115200 with **DataBits** of 8, **StopBits** of 1, None **Parity**, and no **Flow Control**. Confirm the serial port cable has been correctly connected.
  2. If there is nothing wrong with the serial port connection, redo the firmware programming, and ensure no code 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_AMS_C?

- Description

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

- Analysis

  Exceptions occur in the Bluetooth antenna connection or firmware.

- Solution

  1. Check whether the device is truly powered by iOS operating system and the Bluetooth function has been turned on. If the iOS device still cannot find Goodix_AMS_C when the Bluetooth function is turned on, check the antennae of the GR551x Bluetooth device.
  2. If both the Bluetooth function on the iOS device and the antennae of the GR551x Bluetooth device operate properly, check hardware problems by downloading a template firmware, *ble_app_template_fw.bin*, to the GR551x Bluetooth device. The *ble_app_template_fw.bin* is in SDK_`Folder\projects\ble\ble_peripheral\ble_app_template\build\`. After downloading, run the firmware. If the hardware functions properly, the iOS device can scan the advertising from Goodix_Tem; otherwise, exceptions occur in the Bluetooth antenna connection or firmware.

5.3 Why Does an iOS Device Fail to Connect to Goodix_AMS_C Bluetooth Requests?

- Description

  An iOS device cannot connect to Goodix_AMS_C even though it displays under **MY DEVICES**.

- Analysis

  The iOS device has been connected and bonded to GR551x Bluetooth devices, but the data on the Bluetooth device has been erased or overwritten, leading to bonding failure due to loss of bonding information.

- Solution

  1. From **MY DEVICES**, tap **Goodix_AMS_C** to **Forget This Device**.
  2. Redo the Bluetooth scanning, pairing, and bonding on the iOS device.