Bluetooth Accessory Design Guidelines for Apple Products

Release R8

Developer

Contents

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

Figures and Tables

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

1. Introduction

This specification presents design guidelines for hardware accessories that use the Bluetooth transport to communicate with Apple products including Mac, iPhone, iPad, and iPod touch models.

To be compatible with Apple products, both current and future, Bluetooth accessories should follow the guidelines in this specification. An Apple product may make feature availability contingent on the Bluetooth accessory following these specifications.

1.1 Requirements, Recommendations, and Permissions

This specification contains statements that are incorporated by reference into legal agreements between Apple and its licensees. The use of the words *must*, *must not*, *required*, *shall*, *shall not*, *should*, *should not*, *recommended*, *not recommended*, *may*, *optional*, and *deprecated* in a statement have the following meanings:

- *must*, *shall*, or *required* means the statement is an absolute requirement.
- *must not*, *shall not* or *prohibited* means the statement is an absolute prohibition.
- *should* or *recommended* means the full implications must be understood before choosing a different course.
- *should not* or *not recommended* means the full implications must be understood before choosing this course.
- *may* or *optional* means the statement is truly optional, and its presence or absence cannot be assumed.
- *deprecated* means the statement is provided for historical purposes only and is equivalent to 'must not'.

The absence of requirements, recommendations, or permissions for a specific accessory design in this specification must not be interpreted as implied approval of that design. Developers are strongly encouraged to ask Apple for feedback on accessory designs that are not explicitly mentioned in this specification.

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

1.2 Terminology

1.2.1 Accessory, Device, and Product

Throughout this specification:

- The term *device* is used to refer to:
  - An Apple iPhone, iPad, or iPod (typically running iOS, Apple's mobile operating system).
  - An Apple Watch (typically running watchOS, Apple's watch operating system).
  - An Apple TV (typically running tvOS, Apple's television operating system).
- The term *accessory* is used to refer to any product intended to interface with a *device* via the means described in this specification.
- The term Apple *product* is used to refer generically to either a Mac (Apple computers that run macOS or OS X) or to an aforementioned *device*.

Statements that explicitly mention iOS, watchOS, tvOS, or macOS / OS X apply only to products running those operating systems.

1.3 Organization of This Specification

The specifications in this specification are presented as follows:

- Bluetooth (page 9) relates the design of hardware accessories to the general Bluetooth specification.
- Bluetooth Low Energy (page 21) relates the design of hardware accessories to the general Bluetooth Low Energy specification.
- Bluetooth Accessory Identification (page 32) describes enabling Apple-specific Bluetooth commands.
- Bluetooth Headset Battery Level Indication (page 34) describes providing headset battery level information to an Apple device.
- Siri (page 35) presents design guidelines for Bluetooth accessories that will use the Siri functionality of Apple devices.
- iPod Accessory Protocol (page 51) references an Apple protocol that can extend accessory capabilities beyond those supported by standard Bluetooth profiles.

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

1.4 Apple Bluetooth Development Mailing List

Questions or comments regarding accessory development for Apple products can be posted to the Apple mailing list for Bluetooth development, mailto:bluetooth-dev@lists.apple.com. See http://lists.apple.com/mailman/listinfo/bluetooth-dev to join the mailing list.

Engineers on the Apple Bluetooth development team monitor this mailing list and will try to answer your questions. Apple recommends searching the archives to see if a topic has already been discussed before starting a new thread.

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

2. Bluetooth

Accessories that integrate Bluetooth technology must comply with the requirements stated in this chapter.

2.1 Conformity With Bluetooth Specifications

Every accessory that is compatible with an Apple product must support the *Bluetooth Core Specification* Version 2.1 + EDR or higher. This specification introduced the important security feature Secure Simple Pairing as well as Extended Inquiry Response.

2.1.1 Enhanced Data Rate

The Enhanced Data Rate (EDR) feature introduced in the *Bluetooth 2.0* specification enables accessories to communicate more efficiently. Every accessory must use EDR for the following reasons:

- It provides higher data rates compared to Basic Data Rate (BDR).
- It communicates more efficiently, transferring more data bits per unit of time.
- It reduces the power consumption used per bit transferred.
- It improves coexistence with Wi-Fi and other connected Bluetooth devices because it frees up more air time.
- It improves performance in multipoint configurations.

2.1.2 Adaptive Frequency Hopping

The Adaptive Frequency Hopping (AFH) feature introduced in the *Bluetooth 1.2* specification improves coexistence with Wi-Fi and other connected Bluetooth devices. Every accessory must use AFH.

2.1.3 Sniff Mode for Low Power Consumption

Minimizing power consumption is critical for all mobile devices. Therefore, every accessory that is compatible with an Apple product:

- Must support and should request Bluetooth sniff mode.
- Must accept requests for sniff mode and support all valid parameters listed in the Bluetooth specification.

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

- Must support a sniff interval of 15 ms.
- Should use the following recommended sniff mode values:
  - Max Interval: 15 ms
  - Min Interval: 15 ms
  - Sniff Attempt: 1
  - Sniff Timeout: 0
- Must not renegotiate sniff after being established.
- Must support sniff subrating.

Accessories that are compatible with Apple products should also use sniff mode as much as possible, especially when there is little or no data being transmitted over the Bluetooth link. Besides its power consumption advantages, sniff mode enables better antenna sharing with Wi-Fi.

The sniff mode parameters are specific to the usage model and Bluetooth profile. The Apple product expects the accessory to request sniff mode with appropriate parameters for a specific usage. If the accessory does not send such a request, the Apple product may send a sniff mode request. When the Apple product sends a request for sniff mode, the remote device must accept the request and its parameters without negotiation.

If the accessory sets the sniff mode parameters, the accessory must set the sniff interval to less than a third of the Bluetooth baseband Link Supervision Timeout, see Link Supervision Timeout (page 12). This makes the Bluetooth link less susceptible to interference. To improve link robustness, the accessory should use a shorter sniff interval instead of multiple sniff attempts.

Links with a sniff interval of 1 second or more make the slave device open up a large correlation window, which has to be taken into account when calculating the number of sniff attempts. With sniff intervals shorter than 1 second, multiple sniff attempts can improve link robustness but will increase power consumption.

2.1.4 Role and Topology Management

Every accessory that is compatible with an Apple product must:

- Accept a request for Role Switch from an Apple product.
- Continue with the connection when the Apple product rejects a request for Role Switch.

In a Bluetooth connection, one device is the master and the other the slave. The master can have multiple slaves, thus forming a piconet. The master can also be a slave to another master, creating a scatternet.

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

Such a scenario creates complications since the device has to alternate between the two piconets and thus wastes valuable bandwidth. Managing the topology of the network is therefore important for maximum performance. The Apple product may request a Role Switch, depending on its current topology, and the remote device must accept the request. The Apple product may also reject a request for a Role Switch because of topology concerns. Having a suboptimal topology may degrade the audio quality and the user's experience.

The accessory should avoid requesting to be the master as the Apple product will need to be the master in more frequently occurring scenarios. Accessories that always insist on being the master may impact the overall user experience.

Accessories that connect to multiple Apple products simultaneously must support creating a scatternet.

2.1.5 Extended Inquiry Response

Every accessory that is compatible with an Apple product must provide the following information in its Extended Inquiry Response packet:

- The Local Name of the accessory (Complete or Shortened).
- The TX Power Level.

During the Bluetooth discovery process, the Apple product prefers to display the Friendly Name of discovered accessories. Before the 2.1 version of the Bluetooth specification the Apple product would have to set up a connection to the accessory and do a Remote Name Request, which takes power, antenna time, and user's time. The Extended Inquiry Response feature, introduced in Bluetooth 2.1, lets an accessory send its Local Name and other information as part of the Inquiry Response and thereby increase the speed and efficiency of the discovery process.

The Local Name should match the accessory's markings and packaging and not contain ':' or ';'.

2.1.6 Secure Simple Pairing

Every accessory that is compatible with an Apple product must:

- Use Secure Simple Pairing.
- Use the Numerical Comparison method if it has a display and input device supporting it.

Secure Simple Pairing greatly increases security and is a mandatory security feature introduced in the Bluetooth 2.1 specification. To protect against a man-in-the-middle attack, the Numerical Comparison association model should be used whenever feasible. See Volume 1, Section 5.4 in the *Bluetooth Core Specification* , Version 2.1 + EDR.

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

2.1.7 Pairing Button

If the accessory has a dedicated pairing button and it is labeled, it should use official Bluetooth branding. See https://www.bluetooth.com/marketing-branding/brand-best-practices-guidelines.

2.1.8 Class of Device (CoD)

Apple products use the accessory's Class of Device for UI purposes or to configure specific features.

Every accessory that is compatible with an Apple product must accurately set its Class of Device using the Bluetooth SIG defined Major Device Class and Minor Device Class. See Volume 3, Part C, Section 3.2.4 in the *Bluetooth Core Specification* , Version 5.0.

For example, an audio/video accessory intended to operate in a vehicle should set Major Device Class to *audio/video* and Minor Device Class to *car-audio* .

2.1.9 Link Supervision Timeout

The link supervision timeout is used to detect link loss between the accessory and the Apple device.

The accessory must set the link supervision timeout to 2 seconds or greater when it is the master to account for the unpredictable nature of RF signals as well as the Apple device's need to service other concurrent wireless systems.

2.1.10 Delay Reporting

Apple devices (as of iOS 8.2) support the Delay Reporting commands as specified in the Bluetooth *Audio/Video Distribution Transport Protocol* , Version 1.3. Accessories should provide this information as it is used to improve audio/video synchronization for video playback. Accessories should not report a delay of more than 1000 ms and should not update the delay more than 1 time per second.

2.2 Profiles

The Apple knowledge base article https://support.apple.com/kb/ht3647 provides a complete list of the Bluetooth profiles that certain Apple devices support. The Bluetooth specifications are the starting point for designing accessories that are compatible with these products. The following sections add information and requirements for some profiles, which can help accessory developers achieve superior results.

2.2.1 Device ID Profile (DID)

Every accessory that is compatible with an Apple product must:

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

- Support the Bluetooth Device ID Profile, version 1.3 or higher.
- Use the company identifier from the Assigned Numbers specification assigned by the Bluetooth SIG as its Vendor ID value (VID). See http://www.bluetooth.org/Technical/AssignedNumbers/identifiers.htm (requires login). Bluetooth HID Profile accessories may use a VID assigned by the USB Implementers Forum (USB-IF at http://www.usb.org) if the manufacturer does not have a Bluetooth SIG company identifier.
- Use its VID value for the end product manufacturer.
- Not use the Company ID assigned to Apple by the Bluetooth SIG or the Vendor ID assigned to Apple by the USB Implementers Forum.
- Use the Vendor ID Source field to identify which organization assigned the value used in Vendor ID field value. See Section 5.6 of the *Bluetooth Device ID Profile Specification* .
- Use a ProductID value that uniquely identities the product.
- Use a Version value that uniquely identifies the software version.

The Device ID record lets the Apple product identify the implementation of the remote accessory. This is valuable information and can be used to bridge alternate interpretations of the Bluetooth specification when communicating with a remote accessory. Therefore it is important that the information in the Device ID record uniquely identify the implementation.

In the case of Bluetooth car kit devices, for instance, the same car kit might go into two different car models. Ideally the two car kits should have different ProductIDs. However, it is acceptable for them to have the same ProductID as long as they have identical hardware, software, and features. If the implementations differ at all, they should have different ProductIDs. The accessory can also use a secondary Device ID record to uniquely identify the product ID or model number.

2.2.2 Service Discovery Protocol (SDP)

To facilitate caching of Service Discovery Protocol (SDP) service records, every accessory that is compatible with an Apple product must:

- Support the ServiceDiscoveryServer Service Class.
- Support the ServiceDatabaseState attribute.
  - The attribute's value must change whenever any SDP service record or attributes within a record are added, removed, or modified.
  - The attribute's value must not change based on RFCOMM channel protocol parameters since Apple devices query these values separately at connection time.

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

2.2.3 Hands-Free Profile (HFP)

Every accessory that is compatible with an Apple product and supports the Handsfree Profile should meet the requirements of the *Bluetooth Hands-Free Profile Specification* , Version 1.5 or higher. Additional Apple requirements are specified in this section.

Remote accessories can use the Bluetooth *Hands-Free Profile* for phone communications. To achieve the best user experience, the remote accessory should support the following features, which are optional in the Bluetooth specification.

2.2.3.1 Remote Audio Volume Control

Every accessory that is compatible with an Apple product and supports HFP should:

- Support Remote Audio Volume Control so the speaker volume on the Hands-Free accessory can be controlled from the Apple product as described in Section 4.28 in the *Bluetooth Hands-Free Profile Specification* version 1.5.
- Set the Remote volume control bit in the Supported Features bitmap sent with the `AT+BRSF=` command.

In some situations it is easier for the user to control the output volume through the Apple product instead of directly on the remote accessory. For example, a passenger (or-if the car is parked-the driver) in a car could use the volume slider on the phone to control the audio volume. Volume control synchronization is outlined in Section 4.48.2 in the *Bluetooth Hands-Free Profile Specification* version 1.5.

2.2.3.2 Indicator Event Reporting

Every accessory that is compatible with an Apple product and supports HFP should use indicator events reporting and not perform repetitive polling of status.

Apple products support all mandatory and optional indicators specified in HFP version 1.5 (service, call, callsetup, callheld, signal, roam, battchg). To minimize unnecessary polling of status using the `AT+CIND?` command, the remote accessory should enable indicator events reporting by sending an `AT+CMER` command. The Apple product will then send a `+CIEV` event when there is a change in status of an indicator. The remote accessory should request the initial status using the `AT+CIND=?` and `AT+CIND?` commands, according to the HFP specification.

2.2.3.3 Voice Recognition Activation

Every accessory that is compatible with an Apple product and supports HFP must:

- Support Voice Recognition Activation, both AG and HF initiated as described in Section 4.25 in the *Bluetooth Hands-Free Profile Specification* version 1.5.

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

- Set the Voice Recognition Activation bit in the "SupportedFeatures" bitmap sent with the AT+BRSF= command.

Apple products support voice recognition initiated by remote (Hands-Free) accessories and iOS (Audio Gateway) accessories.

2.2.3.4 Echo Cancellation and Noise Reduction

When echo cancellation and noise reduction are performed locally on a Hands-Free accessory, it should turn off echo cancellation and noise reduction on the Apple product by sending an AT+NREC command, as described in Section 4.24 in the *Bluetooth Hands-Free Profile Specification* version 1.5.

Apple products support echo cancellation and noise reduction; these features are active by default. If a Hands-Free accessory also does echo cancellation and noise reduction it needs to turn these features off on the Apple product (the Audio Gateway). This avoids unnecessary degradation of audio quality due to double audio processing.

2.2.3.5 In-Band Ringing

Every accessory that is compatible with an Apple product and supports HFP should also support In-Band Ringing as specified in Section 4.13.1 in the *Bluetooth Hands-Free Profile Specification* version 1.5. If the user sets a ring tone on the Apple product, the same ring tone should sound on the hands-free accessory.

2.2.3.6 Synchronous Connection

Every accessory that is compatible with an Apple product and supports HFP must:

- Support eSCO parameter set S2 and S3 and accept requests for these settings. See Section 5.6 of the *Bluetooth Hands-Free Profile Specification* version 1.5.
- Request eSCO parameter set S2 or S3 when setting up a Synchronous Connection. Note that eSCO parameter set S1 should not be requested.
- Render audio within 40 ms after the SCO/eSCO connection has been set up.

The eSCO packet types offers retransmission of packets; traditional SCO packets are not retransmitted. This improves audio quality and the user's experience. The eSCO packet types 2-EV3 and 3-EV3 offer a greater time interval between packets, which can improve Wi-Fi performance and allow time for other concurrent Bluetooth connections to send data. Apple strongly recommends the use of 2-EV3 and 3-EV3 packets for SCO connections. Using HV3 packets is highly discouraged. HV3 packets require more link time and does not allow for retransmission of audio packets which impacts the audio performance in presence of RF interference.

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

2.2.3.7 Wide Band Speech

Every accessory that is compatible with an Apple product and supports HFP should support Wide Band Speech as described in Section 5.7.4 of the Bluetooth *Hands-Free Profile* specification version 1.6. If Wide Band Speech is supported, it should support the T2 link parameter settings.

All Apple devices running iOS 5 or later support Wide Band Speech. If both the Apple device and the accessory support Wide Band Speech then Wide Band Speech link will be used for eSCO connection for use cases like cellular calls, FaceTime and Siri.

2.2.4 Message Access Profile (MAP)

Every accessory that is compatible with an Apple product and supports MAP must:

- Support Message Notification as described in Section 4.1 of the *Bluetooth Message Access Profile Specification*, version 1.0.
- Register for notifications immediately after the connection is established, as described in Section 4.5 in the *Message Access Profile Specification*, version 1.0.
- Not expect the TEL property to be present in the originator VCARD (the properties N and FN will be included). See Section 3.1.3 in the *Message Access Profile Specification*, version 1.0.
- Not provide a user interface for sending messages. Apple devices do not support sending messages using MAP.

All Apple devices running iOS 6.0 or later support MAP.

2.2.5 Audio/Video Remote Control Profile (AVRCP)

Every accessory that is compatible with an Apple product and supports the Audio/Video Remote Control Profile should meet the requirements of the *Bluetooth Audio/Video Remote Control Profile Specification*, Version 1.4. Additional Apple requirements are specified in this section.

2.2.5.1 Supported Operations

Apple products support the following operation_IDs in Pass Through commands:

- Play
- Stop
- Pause
- Fast Forward
- Rewind

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

- Forward
- Backward

2.2.5.2 Repeat and Shuffle Modes

Every Apple device supports Repeat and Shuffle modes in the role of an AVRCP target. An AVRCP controller may use `SetPlayerApplicationSettingValue` to set a value on the Apple device and `GetPlayerApplicationSettingValue` to read a value, as described in Sections 6.5.4 and 6.4.3 of the *Bluetooth Audio/Video Remote Control Profile Specification* version 1.4.

2.2.5.3 Notifications

Every accessory that is compatible with an Apple product and supports AVRCP should register for notifications and not perform repetitive polling to determine the status of the Apple product.

Every Apple device supports registering for notifications in the role of an AVRCP Target, as described in Section 6.7 of the *Bluetooth Audio/Video Remote Control Profile Specification* version 1.4. The commands `RegisterNotification` and `GetPlayStatus` are supported for these notifications:

- EVENT_PLAYBACK_STATUS_CHANGED
- EVENT_TRACK_CHANGED
- EVENT_NOW_PLAYING_CONTENT_CHANGED
- EVENT_AVAILABLE_PLAYERS_CHANGED
- EVENT_ADDRESSED_PLAYER_CHANGED
- EVENT_VOLUME_CHANGED

2.2.5.4 Play/Pause Button

All accessories that support AVRCP and implement a Play/Pause button must confirm the playback status of the Apple device via AVRCP notifications (see Notifications (page 17) before sending a Play or Pause command (see Supported Operations (page 16)). Specifically:

- If the Apple device has notified the accessory that it is paused, pressing the accessory's Play/Pause button should send a Play command.
- If the Apple device has notified the accessory that it is playing, pressing the accessory's Play/Pause button should send a Pause command.
- The accessory should not infer Apple device playback status based on the number of times the Play/Pause button has been pressed.

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

2.2.5.5 Volume Handling

Every accessory that is compatible with an Apple product and supports AVRCP should support Absolute Volume, as described in Section 6.13 of the *Bluetooth Audio/Video Remote Control Profile Specification* version 1.4.

Every Apple device supports volume handling in the role of AVRCP Controller.

2.2.5.6 Browsing

Every accessory that is compatible with an Apple product and supports Browsing (in controller role) as part of AVRCP must:

- Not try to index or cache the entire library upon connection. The Apple product may contain tens of thousands of media items, each present multiple times in the hierarchy.
- When browsing a specific folder, do not fetch all its items. Only fetch those that are displayed to the user. It may prefetch a few items to improve the responsiveness of the user interface.
- Not reorder items (e.g. alphabetically).
- Not assume UIDs to be statically defined, especially in the root folder. The ordering and UIDs of folders and items may change at any point in future releases.
- Send the `SetBrowsedPlayer` command after receiving an EVENT_UIDS_CHANGED notification.
- Not assume that the UID passed to the PlayItem command will result in the media player playing that UID.

Currently only the built-in Music app supports browsing. When switching between players, an EVENT_AVAILABLE_PLAYERS_CHANGED notification and an EVENT_ADDRESSED_PLAYER_CHANGED notification will be generated. The UI then needs to look at the feature bit mask of the listed player to determine whether browsing is currently available.

All Apple devices running iOS 6.0 or later support AVRCP Browsing.

2.2.5.7 iOS App-Provided Metadata

An audio app running on an Apple device may use the iOS Media Player Framework to provide metadata about the current audio stream. The Apple device supplies this metadata to the accessory using AVRCP. For more information, see the `MPNowPlayingInfoCenter` class in Apple Media Player Framework documentation.

2.2.6 Advanced Audio Distribution Profile (A2DP)

See Advanced Audio Distribution Profile (A2DP) (page 28).

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

2.3 Audio Routing

This section describes how an accessory can differentiate between various audio contents coming from an Apple device and use this information to decide playback behavior.

An accessory can receive audio data from the Apple device via either of two Bluetooth profiles:

- HFP using eSCO channel
- A2DP using ACL channel

The Apple device picks which channel to use depending on how the audio content is used. An audio path created for two way communication (such as phone calls or FaceTime) always uses the HFP (eSCO) route for sending audio data. Music and similar content uses the A2DP route. In the absence of a defined route, audio playback will default to the Apple device.

2.3.1 Audio Data Received via HFP Profile

Most of the audio content sent via HFP (eSCO) routes requires two way communication. Cases where HFP (eSCO) is used include (but are not limited to) cellular calls, FaceTime, and voice mail.

For any audio content that is being received via the HFP (eSCO) route, it is expected that both the speaker and the microphone of the accessory are dedicated to the Bluetooth link and should not handle any other audio content.

2.3.2 Audio Data Received via A2DP Profile

Audio content transferred via A2DP profiles can be broadly classified into two categories:

- Audio content from music, video, or game-like applications.
- System-generated sound for alerts and notifications.

2.3.2.1 Differentiating Audio Content from System Sounds

Music-like content can be differentiated from system sound by adding support for Audio/Video Remote Control Profile (AVRCP) version 1.3 or later. The AVRCP profile allows an accessory to be aware of the audio playback state in the Apple device, using notifications. See Audio/Video Remote Control Profile (AVRCP) (page 16).

When an Apple device initiates audio playback over an A2DP channel for playing music content, an AVRCP notification EVENT_PLAYBACK_STATUS_CHANGED is sent to indicate that playback status has changed to play state. See Section 6.7.2 of the *Audio/Video Remote Control Profile* specification, version 1.4. This indicates that audio data via the A2DP profile contains music. When an Apple device initiates audio playback over an A2DP channel for playing system sound, no AVRCP notification is sent.

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

Figure 2-1 (page 20) and Figure 2-2 (page 20) show the difference between the notifications for music playback and for system sounds.

Figure 2-1 Initiate Audio Playback (e.g. music)

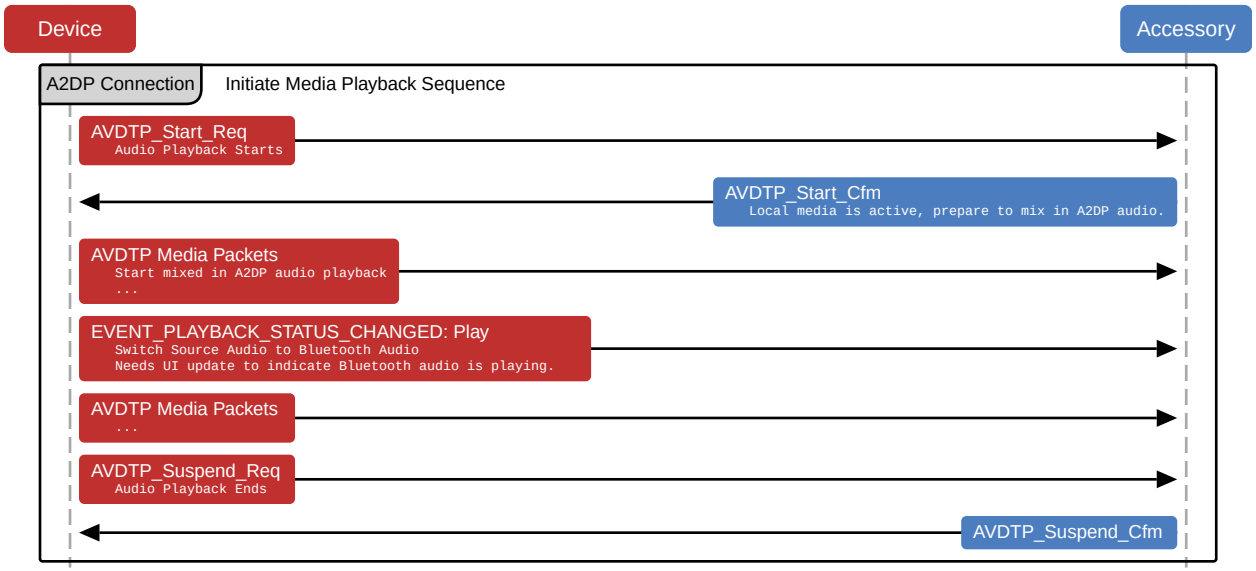
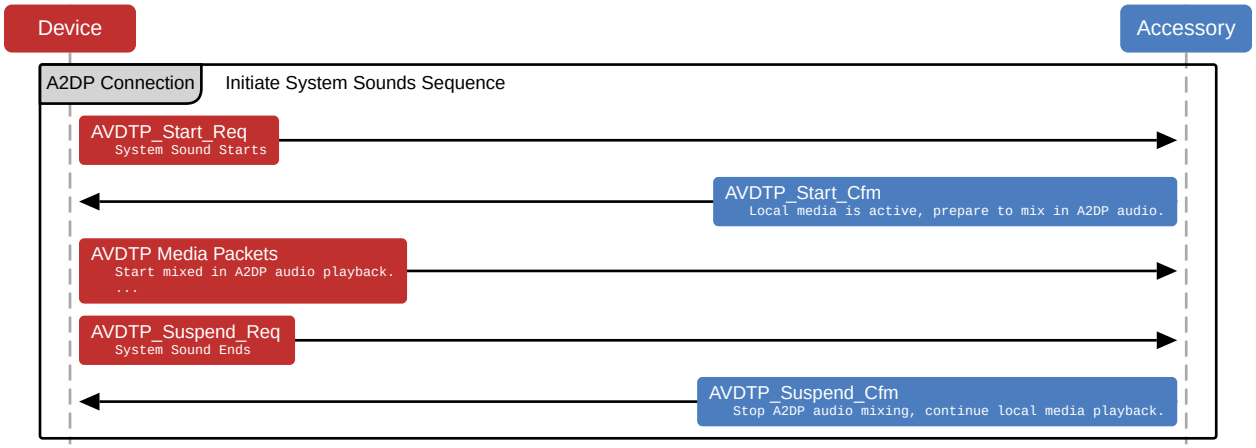


Figure 2-2 Initiate System Sound (e.g. turn-by-turn directions)



2.3.2.2 Expected Audio Routing Behavior for A2DP

The accessory should tune its audio routing behavior based on audio content over the A2DP channel.

If audio data contains music, then it is expected that the accessory speakers are dedicated to audio data coming via the Bluetooth link and any other audio playback is paused. If audio data contains system sound, then it is expected that the accessory can render audio as desired. If the accessory is playing audio from a different source, then system sound data can be mixed with the existing track for playback; it is not necessary to pause existing audio playback on the device.

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

3. Bluetooth Low Energy

The *Bluetooth 4.0* specification introduces Bluetooth Low Energy, a new wireless technology targeted for accessories with limited battery resources. If Bluetooth Low Energy is supported, the accessory should follow the guidelines in this section.

3.1 Role

The accessory should implement either the Peripheral role as defined in the *Bluetooth 4.0* specification, Volume 3, Part C, Section 2.2.2.3 or the Broadcaster role, as defined in Section 2.2.2.1.

3.2 Advertising Channels

The accessory should advertise on all three advertising channels (37, 38, and 39) at each advertising event. See the *Bluetooth 4.0* specification, Volume 6, Part B, Section 4.4.2.1.

3.3 Advertising PDU

The accessory should use one of the following advertising PDUs:

- ADV_IND
- ADV_NOCONN_IND
- ADV_SCAN_IND

ADV_DIRECT_IND should not be used. See the *Bluetooth 4.0* specification, Volume 6, Part B, Section 2.3.1.

3.4 Advertising Data

The advertising data sent by the accessory should contain at least the following information as described in the *Bluetooth Core Specification Supplement*, Part A:

- Flags

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

- TX Power Level
- Local Name
- Services

The Local Name should match the accessory's markings and packaging and not contain ':' or ';'.

The accessory may put the Local Name and the TX Power Level data in the SCAN_RSP PDU if, for example, it needs to reduce power consumption or not all of the advertising data fit into the advertising PDU. Note that, depending on its state, the Apple product may not always perform active scanning.

The primary services should always be advertised in the advertising PDU. Secondary services should not be advertised. Services not significant to the primary use case of the accessory may be omitted if space is limited in the Advertising PDU.

The advertising data and the scan response data in the SCAN_RSP PDU should comply with the formatting guidelines in the *Bluetooth 4.0* specification, Volume 3, Part C, Section 18: it starts with a length field, followed by AD Type and AD Data.

3.5 Advertising Interval 

The advertising interval of the accessory should be carefully considered, because it affects the time to discovery and connect performance. For a battery-powered accessory, its battery resources should also be considered.

To be discovered by the Apple product, the accessory should first use the recommended advertising interval of 20 ms for at least 30 seconds. If it is not discovered within the initial 30 seconds, Apple recommends using one of the following longer intervals to increase chances of discovery by the Apple product:

- 152.5 ms
- 211.25 ms
- 318.75 ms
- 417.5 ms
- 546.25 ms
- 760 ms
- 852.5 ms
- 1022.5 ms
- 1285 ms

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

Note: Longer advertising intervals usually result in longer discovery and connect times.

3.6 Connection Parameters

The accessory is responsible for the connection parameters used for the Low Energy connection. The accessory should request connection parameters appropriate for its use case by sending an L2CAP Connection Parameter Update Request at the appropriate time. See the *Bluetooth 4.0* specification, Volume 3, Part A, Section 4.20 for details.

The connection parameter request may be rejected if it does not comply with all of these rules:

- Slave Latency ≤ 30
- 2 seconds ≤ connSupervisionTimeout ≤ 6 seconds
- Interval Min modulo 15 ms == 0
- Interval Min ≥ 15 ms
- One of the following:
  - Interval Min + 15 ms ≤ Interval Max
  - Interval Min == Interval Max == 15 ms
- Interval Max * (Slave Latency + 1) ≤ 2 seconds
- Interval Max * (Slave Latency + 1) * 3 < connSupervisionTimeout

Note that if an accessory requests Interval Min == Interval Max == 15 ms, some Apple devices will scale the interval to 30 ms to balance power and performance constraints.

If Bluetooth Low Energy HID is one of the connected services of an accessory, connection interval down to 11.25 ms may be accepted by the Apple product.

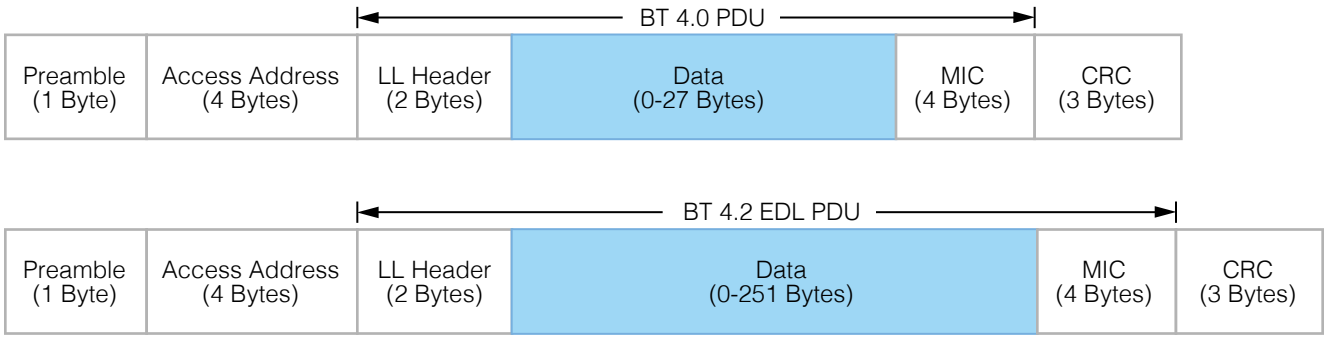
The Apple product will not read or use the parameters in the Peripheral Preferred Connection Parameters characteristic. See the *Bluetooth 4.0* specification, Volume 3, Part C, Section 12.5.

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

3.7 Data Packet Length Extension

Data Packet Length Extension is an enhancement introduced in the *Bluetooth 4.2* specification which increases the maximum data length from 27 to 251. Using a longer per-packet data length improves radio efficiency, greatly increases application data rates, and boosts battery life. See the *Bluetooth 5.0* specification, Volume 6, Part B, Section 4.6.6 for details.

Figure 3-1 Data Packet Length Extension



Accessories should support Data Packet Length Extension for best performance with Apple devices.

Apple products operating as master will negotiate optimal data packet lengths based on various factors such as connection event length, system topology, and protocol.

3.8 Privacy

The accessory should be able to resolve a Resolvable Private Address in all situations. Due to privacy concerns, the Apple product will use a Random Device Address as defined in the *Bluetooth 4.0* specification, Volume 3, Part C, Section 10.8.

3.9 Permissions

The accessory should not require special permissions, such as pairing, authentication, or encryption to discover services and characteristics. It may require special permissions only for access to a characteristic value or a descriptor value. See the *Bluetooth 4.0* specification, Volume 3, Part G, Section 8.1, fifth paragraph.

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

3.10 Pairing

The accessory should not request pairing until an ATT request is rejected using the Insufficient Authentication error code. See the *Bluetooth 4.0* specification, Volume 3, Part F, Section 4 for details.

If, for security reasons, the accessory requires a bonded relationship with the Central, the Peripheral should reject the ATT request using the Insufficient Authentication error code, as appropriate. As a result, the Apple product may proceed with the necessary security procedures.

Similarly, if the Apple device acts as a Central and a GATT server, it may reject an ATT request using the Insufficient Authentication error code. The accessory should initiate the security procedure for pairing in response.

Pairing may require user authorization depending on Apple product. Once an accessory is paired with an Apple product, it must retain the distributed keys of both central and peripheral for future use. If the pairing is no longer required, the accessory must delete both sets of keys.

3.11 MTU Size

An accessory that supports packet length extension must perform the packet length update procedure before performing the Exchange MTU Request handshake, see Data Packet Length Extension (page 24).

Apple devices will support and request an MTU size larger than the default during the Exchange MTU Request handshake. See the *Bluetooth 4.0* specification, Volume 3, Part F, Section 3.2.8.

When operating as ATT client, the Apple device will request the optimal MTU size based on factors such as the Bluetooth topology, connection event length, maximum data length, and protocol (GATT or connection-oriented L2CAP).

An accessory operating as ATT server should select an MTU that is equal to or greater than the Apple device's MTU request.

3.12 Services

3.12.1 Generic Access Profile Service

The accessory should implement the Device Name characteristic per the *Bluetooth 4.0* specification, Volume 3, Part C, Section 12.1. The Device Name characteristic should be writeable.

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

3.12.2 Generic Attribute Profile Service

The accessory must implement the Service Changed characteristic only if the accessory has the ability to change its services during its lifetime.

The Apple product may use the Service Changed characteristic to determine if it can rely on previously read (cached) information from the device. See the *Bluetooth 4.0* specification, Volume 3, Part G, Section 7.1.

3.12.3 Device Information Service

The accessory must implement the Device Information Service. The service UUID for this service should not be advertised in the Advertising Data. The following characteristics should be supported:

- Manufacturer Name String
- Model Number String
- Firmware Revision String
- Software Revision String

3.12.4 Available Services

With iOS 7.0, any Apple device makes Battery Service, Current Time Service and Apple Notification Center Service (ANCS) available to an accessory. The Current Time Service supports the current time and local time information characteristics. The service does not provide an "Adjust Reason" when the current time changes. ANCS uses 7905F431-B5CE-4E99-A40F-4B1E122D00D0 as its UUID.

These services are not guaranteed to be available immediately after connection and the accessory must support Characteristic Value Indication of the Service Changed characteristic (see *Bluetooth 4.0* specification, Volume 3, Part G, Section 7.1) to be notified when the services become available. The Apple device will maintain a connection to an accessory as long as it is paired and uses one of the available services.

3.13 GATT Server

With iOS 6.0, applications may contribute services and characteristics to the GATT server that the Apple device makes available to the accessory. The recommendations in this section apply to the accessory in this case.

The following services are implemented internally by iOS and must not be published by third party iOS applications:

- Generic Attribute Profile Service
- Generic Access Profile Service

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

- Bluetooth Low Energy HID Service
- Battery Service
- Current Time Service
- Apple Notification Center Service

The Apple device implements the GAP Service Changed characteristic, because the database contents can change at any time. The accessory should therefore support the Characteristic Value Indication of this characteristic and, upon receiving indications, invalidate its database cache accordingly. See the *Bluetooth 4.0* specification, Volume 3, Part G, Section 7.1.

The accessory should minimize the use of ATT/GATT requests and commands and only send what is necessary. For example, do not use GATT Discover All Services when the accessory is looking for specific services. Use Discover Primary Service By Service UUID instead. Less airtime equals less power consumption and better performance for both the accessory and the Apple device.

When third party iOS applications discover services on the accessory, the following services are used internally by iOS and are filtered out from the list of discovered services:

- Generic Attribute Profile Service
- Generic Access Profile Service
- Bluetooth Low Energy HID Service
- Apple Notification Center Service

The accessory should be robust enough to handle any error gracefully. Pairing and Characteristic Value reads/writes may fail if the application that owns the service is not in the foreground and is not entitled to run in the background.

If an ATT Prepare Write Request is used, all queued attributes are contained within the same GATT Service.

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

4. Advanced Audio Distribution Profile (A2DP)

Accessories may implement the Advanced Audio Distribution Profile (A2DP) over Bluetooth (see Bluetooth (page 9)) to receive audio from Apple products.

The audio content from the Apple product can be broadly classified into two categories:

- Audio content from music, video, or gaming applications.
- System-generated sounds for alerts and notifications.

A2DP is often implemented in speakers and headsets.

Accessories that implement A2DP must satisfy all requirements stated in Bluetooth (page 9).

4.1 Bluetooth A2DP Specification

Every accessory that implements the Advanced Audio Distribution Profile must meet the requirements of the Bluetooth *Advanced Audio Distribution Profile* specification, Version 1.2.

4.1.1 AVDTP Transactions

Accessories must respond to Audio/Video Distribution Transport Protocol (AVDTP) signaling transactions before the Apple device's 5 second RTX_SIG_TIMER expires or the Apple device will terminate the signaling channel. See Section 6.2 "Transaction Model" and section 6.4 "Signal Command Set" of the Bluetooth *Audio/Video Distribution Transport Protocol*, Version 1.3.

4.2 SubBand Codec (SBC)

The SBC Codec Specific Information Elements, defined in Section 4.3.2 of the A2DP specification, that are applicable to Apple products are listed in Table 4-1 (page 28).

Table 4-1 SubBand Codec Information Elements for Apple products

Element	Value
Sampling Frequency	44,100 Hz

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

Element	Value
Channel Mode	Stereo
Block Length	16
Subbands	8
Allocation Method	Loudness
Bitpool range	2 to 53. Accessories for Apple products should support 53.

4.3 MPEG 2/4 AAC Codecs

Apple devices support the non-mandatory codec MPEG-2/4 AAC, as defined in Section 4.5 of the *Advanced Audio Distribution Profile* specification, Version 1.2. Accessories should use the AAC codec in addition to SBC, because it provides higher audio quality for a given bit rate.

> **Note:** The following specifications provide details of Apple's implementation of the MPEG-2/4 AAC codec. In case of conflicts, the A2DP specification governs.

The MPEG 2/4 AAC Codec Specific Information Elements, defined in Section 4.5 of the A2DP specification, that are applicable to Apple devices are listed in Table 4-2 (page 29).

Table 4-2 MPEG-2/4 AAC Codec Information Elements for Apple devices

Element	Value
Object Type	MPEG-2 AAC LC
Sampling Frequency	44,100 Hz
Channels	2
Bit rate	264,630 bps
VBR	0

AAC audio stream packets in Apple devices have the structure shown in Table 4-3 (page 30).

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

Table 4-3 AAC audio packet for Apple devices

L2CAP	AVDTP	MPEG-4 LATM	MPEG-4 AAC
Header	Header	AudioMuxElement	Audio Payload

The AAC Media Payload Format, as defined in Section 4.5.4 of the A2DP specification, is formatted using LATM, as defined in Section 4 of *IETF RFC 3016* . The following notes apply to the packet fields shown in Table 4-3 (page 30).

- The recommended L2CAP MTU value for each Apple device's AAC streaming channel is 885 bytes.
- The AVDTP Header is shown as the RTP header in Figure 4 of RFC 3016, and is the header defined in Section 7.2.1 of the Bluetooth *Audio/Video Distribution Transport Protocol* , Version 1.2.
- The `AudioMuxElement` is the same as the RTP payload in RFC 3016. It is defined in Section 1.7.3, Table 1.41 in ISO/IEC 14496-3:2009, subpart 1. The `muxConfigPresent` argument to the `AudioMuxElement` is set to 1 (in-band mode), as recommended in Section 4.1 of RFC 3016. As recommended in Section 4.3 of RFC 3016, only one `AudioMuxElement` is put into each AVDTP packet.
- The audio payload is encoded using MPEG-4, as recommended in Section 4.5.4 of the A2DP specification.
- For AAC-LC support, the accessory should support VBR capability. The Apple device will be varying AAC bit rate depending on the content and the accessory should be able to handle the variation without causing gap in the audio.

4.4 Test Procedures

4.4.1 Audio Quality

Assess the quality of the audio signal in each of the following scenarios:

1. Stream music from Music app.
2. Stream music using Apple Music Radio.
3. Stream audio using Podcast app.
4. Stream audio using Beats Music app.
5. Stream audio using 3rd party apps (Pandora, Spotify, etc).

4.4.2 Audio Switching

1. During A2DP streaming, switch audio back to Apple device and switch back to accessory.

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

2. Audio should be routed to the intended source. Audio quality should be good switching back to BT.

4.4.3 HFP Interaction

1. Make incoming / outgoing call during A2DP.
2. Audio should be suspended during the call and resume after the call.

4.4.4 Siri

1. Trigger Siri during A2DP.
2. Audio should be resumed after the Siri session.

4.4.5 Video Playback

1. Stream A2DP while watching a video.
2. Audio / video synchronization and quality should be good.

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

5. Bluetooth Accessory Identification

This chapter describes Apple-specific Bluetooth commands that extend accessory capabilities beyond those supported by standard Bluetooth profiles.

To enable Apple-specific features, the accessory must support HFP Command AT+XAPL (page 32), which provides accurate information about the accessory's supported features. The Apple device will use the information sent by this command to enable and disable custom commands.

The accessory must send the following AT+XAPL command after making a successful HFP Service Level Connection (SLC) to the Apple device. The accessory should send an AT+XAPL command first, before sending any of the additional Apple-specific commands described below.

5.1 HFP Command AT+XAPL

Description: Enables custom AT commands from an accessory.

Initiator: Bluetooth accessory

Format: AT+XAPL=*vendorID*-*productID*-*version*,*features*

Parameters:

- *vendorID*: A string representation of the hex value of the vendor ID from the manufacturer, without the `0x` prefix.
- *productID*: A string representation of the hex value of the product ID from the manufacturer, without the `0x` prefix.
- *version*: The revision of the software.
- *features*: A base-10 representation of a bit field. Available features are:
  - Bit 0 = reserved
  - Bit 1 = The accessory supports battery reporting (reserved only for battery operated accessories).
  - Bit 2 = The accessory is docked or powered (reserved only for battery operated accessories).
  - Bit 3 = The accessory supports Siri status reporting.
  - Bit 4 = the accessory supports noise reduction (NR) status reporting.

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

- All other values are reserved.

Example: AT+XAPL=ABCD-1234-0100,10 (Supports battery reporting and Siri status)

Response: +XAPL=iPhone,*features*

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

6. Bluetooth Headset Battery Level Indication

Any Hands-Free Bluetooth headset accessory can show its battery level to the user as an indicator icon in the Apple device status bar. This feature is supported on all Apple devices that support the Hands-Free Profile, including iPhone, iPod touch, and iPad.

Headset battery indication is implemented by two Apple-specific Bluetooth HFP AT commands, HFP Command AT+XAPL (page 32) and HFP Command AT+IPHONEACCEV (page 34)

6.1 HFP Command AT+IPHONEACCEV

Description: Reports a headset state change.

Initiator: Headset accessory

Format: `AT+IPHONEACCEV=`*Number of key/value pairs* `,`*key1* `,`*val1* `,`*key2* `,`*val2* `,...`

Parameters:

- *Number of key/value pairs*: The number of parameters coming next.
- *key*: the type of change being reported:
  - 1 = Battery Level
  - 2 = Dock State
- *val*: the value of the change:
  - Battery Level: string value between '`0`' and '`9`'
  - Dock State: `0` = undocked, `1` = docked

Example: `AT+IPHONEACCEV=1,1,3`

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

7. Siri

Siri enables a user to have rich interactions with an Apple device by primarily using their voice.

Accessories supporting Siri must not use an icon that resembles the Siri microphone icon.

The rest of this chapter is applicable to accessories that support Siri over Bluetooth.

7.1 Enabling Custom Siri Commands

Every accessory that supports Siri over Bluetooth must support HFP Command AT+XAPL (page 32). The Apple device will use the information sent by this command to enable and disable custom commands related to Siri.

To receive Siri status events, the accessory must send the `AT+XAPL` command after making a successful HFP Service Level Connection (SLC) to the Apple device. The accessory should send an `AT+XAPL` command first, before sending any of the additional Siri-specific commands described below.

7.2 Obtaining Siri Availability Information

After establishing an HFP profile connection, an accessory can determine if Siri is available and enabled on an Apple device. It can also receive notifications of changes in Siri status. If Siri is disabled, Voice Control will be activated instead.

7.2.1 Obtaining Status Information at Connection

The accessory should send the following command after making a successful HFP profile (SLC) connection and sending an `AT+XAPL` command.

7.2.1.1 HFP Command AT+APLSIRI?

Description: AT command to retrieve Siri status information.

Initiator: accessory

Format: `AT+APLSIRI?`

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

Response: `+APLSIRI:`*value*

Defined Values:

- 0 = Siri is not available on this platform.
- 1 = Siri is available and enabled.
- 2 = Siri is available but not enabled.

Example: `+APLSIRI:1` (Siri is available and enabled)

7.2.2 Receiving Siri Availability Updates from the Apple Device

After initialization has been completed, the Apple device will send the accessory the following notification if there is a change in Siri status. This notification will be provided only if the accessory has requested Siri status (by sending `AT+APLSIRI?`) at least once after connection, and if the Apple device has reported that Siri is available and enabled.

7.2.2.1 HFP Command +APLSIRI

Description: Unsolicited event indicating a change in Siri status.

Initiator: Apple device

Format: `+APLSIRI:`*value*

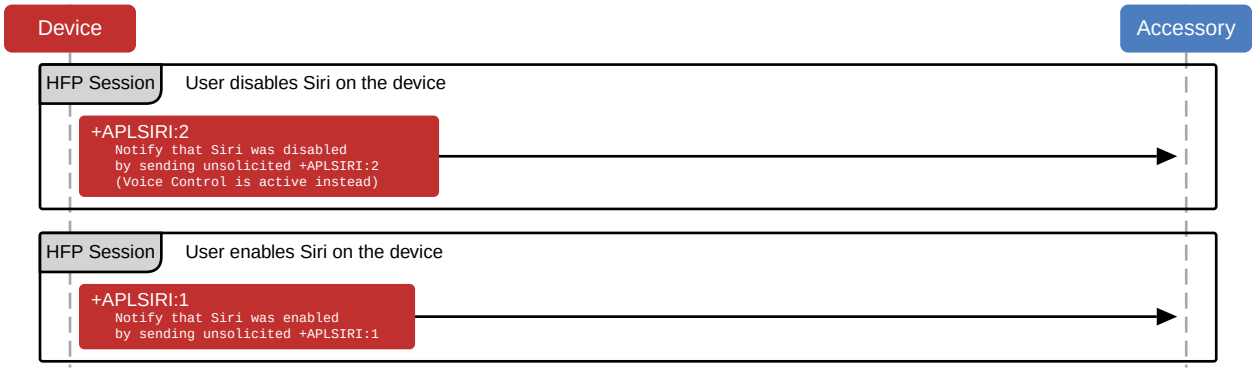
Defined Values:

- 1 = Siri is available and enabled.
- 2 = Siri is available but not enabled.

Example: `+APLSIRI:2` (Siri is available but not enabled)

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

Figure 7-1 Siri is Disabled/Enabled from the Apple Device's Settings



7.3 Initiating a Siri Session

Once support for Siri is established on both the accessory and the Apple device, a Siri session can be started from either one.

7.3.1 Initiating a Session from the Accessory

To initiate a Siri session, the accessory must use the voice recognition command `AT+BVRA` defined in the Bluetooth *Hands-Free Profile* specification. For further details, see the Bluetooth *Hands-Free Profile* 1.6 profile specification, section 4.25. The HFP profile must be connected and SLC must exist.

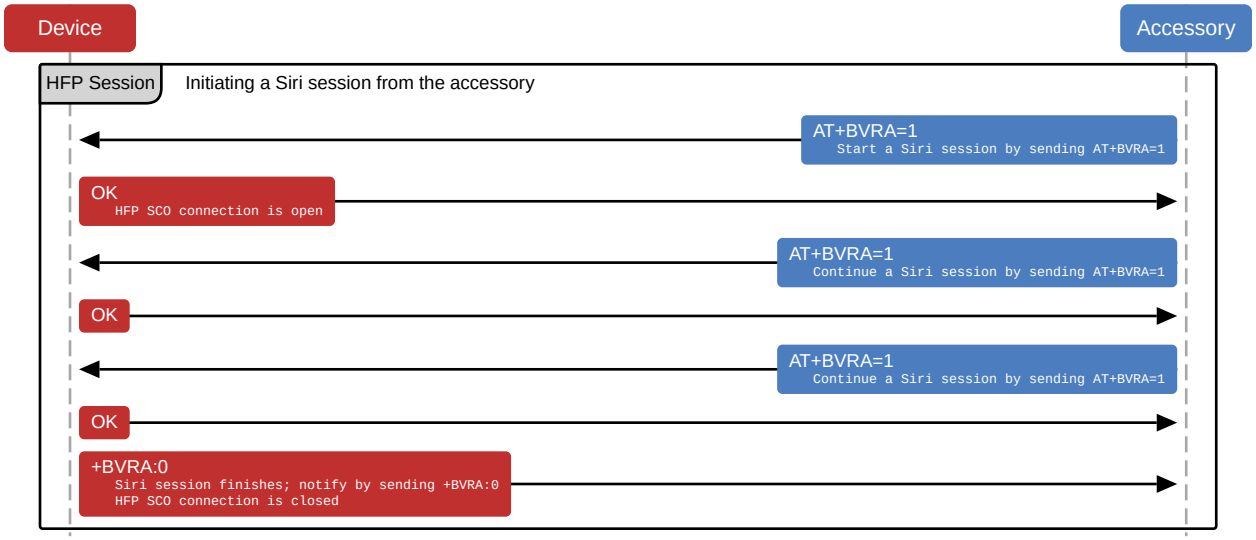
The accessory should use the following command sequence:

- The accessory sends an `AT+BVRA=1` command to the Apple device.
- The Apple device sends an `OK` response.
- The Apple device launches a Siri session and creates a Synchronous Connection (SCO) for the audio.
- If the Siri session is not finished, the accessory must send `AT+BVRA=1` to continue the conversation. This may need to happen multiple times.
- When the Siri session is finished, the Apple device sends a `+BVRA:0` result code to the accessory.
- The Apple device disconnects the SCO connection.

While a Siri session is active, the accessory must let the user continue the conversation and ask follow up questions within the current context. In order to do so, the accessory must be able to send an `AT+BVRA=1` command to the Apple device even after Siri has been already activated and before `+BVRA:0` is received. Figure 7-2 (page 38) shows an overview of the interaction when Siri is triggered from the accessory, the running session was continued twice and once Siri was finished, the device dismissed the session.

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

Figure 7-2 Initiating a Siri Session from the Accessory



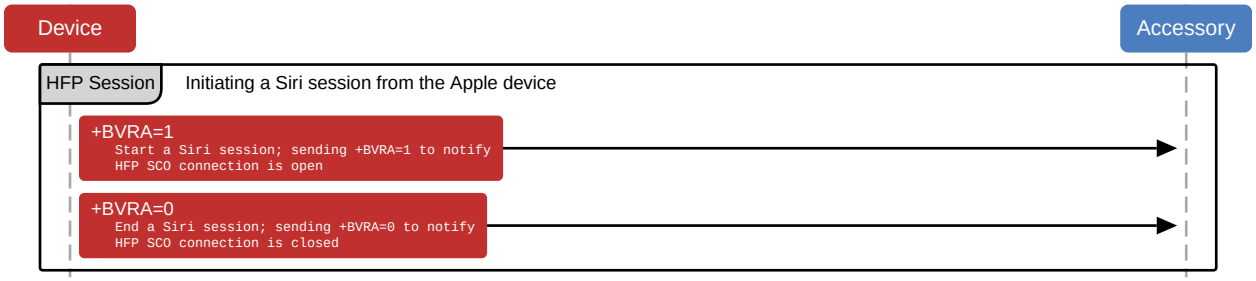
7.3.2 Initiating a Session from the Apple Device

If the accessory supports voice recognition commands, the Apple device sends a `+BVRA` event to indicate the start of a Siri session. The accessory must enable support for voice recognition and indicate it in its feature response as described in the Bluetooth *Hands-Free Profile* 1.6 specification, section 4.34.1, "Bluetooth Defined AT Capabilities." Specifically, the HFP profile must be connected, SLC must exist, and voice recognition activation (bit 3) must be enabled in the `AT+BRSF` command. The Apple device will not use virtual call functionality for the Siri session if voice recognition activation is supported by the accessory.

The accessory should expect the following command sequence:

- The Apple device sends a `+BVRA:1` event to the accessory.
- The Apple device launches a Siri session and creates a SCO connection for the audio.
- When the Siri session is finished, the Apple device sends a `+BVRA:0` result code to the accessory.
- The Apple device disconnects the SCO connection.

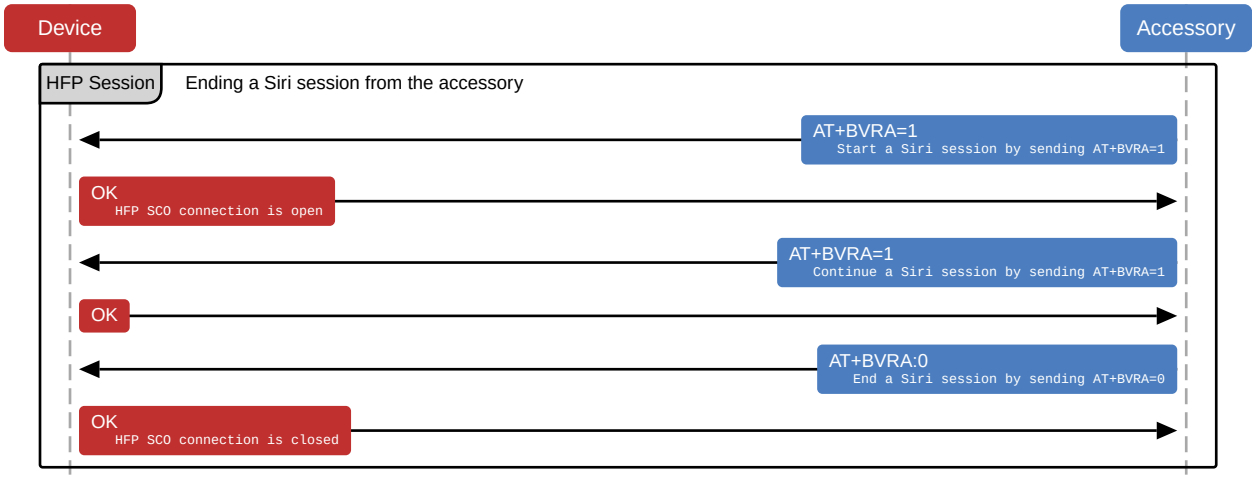
Figure 7-3 Initiating a Siri Session from the Apple Device



7.3.3 Ending a Session from the Accessory

Once a Siri session is running the accessory must be capable of ending the session by sending an `AT+BVRA=0` command to the Apple device. Figure 7-4 (page 39) shows an example of ending a running Siri session from the accessory. The accessory should only end an active session as a direct result of a user action.

Figure 7-4 Ending a Siri Session from the Accessory



7.4 Siri Eyes Free Mode

Siri Eyes Free mode is a feature to control Siri responses that include display information and can be enabled or disabled as needed. In Eyes Free mode, the user experience is tailored towards a driving scenario and interactions with Siri are done primarily via voice to minimize the need for the user to look at a screen. Siri Eyes Free mode is supported only for Bluetooth-enabled vehicle entertainment systems and should not be used by any other accessories. Siri Eyes Free should not be triggered via a voice command.

The Apple device will listen for the HFP AT command `AT+APLEFM` to enable or disable Eyes Free mode.

This command is used by the Apple device to modify Siri responses that contain visual information or require user interaction. Suitable audio feedback and voice commands will be available to the user based on the Siri use case that was initiated.

Eyes Free mode is disabled by default. Once the accessory has enabled Eyes Free mode, it remains enabled for all subsequent Siri sessions initiated from the accessory until the accessory disables it or the Bluetooth connection is disconnected.

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

7.4.1 HFP Command AT+APLEFM

Description: An accessory sends this command to notify an Apple device of the preferred state of Eyes Free mode.

Initiator: accessory

Format: `AT+APLEFM=`*value*

Response: `OK`

Defined Values:

- 0x00 = Disable Eyes Free mode.
- 0x01 = Enable Eyes Free mode.
- 0x02-0xFF = reserved

Example: `AT+APLEFM=1`

7.5 Improving Voice Recognition

The microphone audio that the accessory sends to the Apple device during a Siri session should be suitable for voice recognition. Audio requirements for optimal voice recognition may differ from requirements for optimal human perception (such as during a cellular phone call).

Filtering of the audio signal to remove echoes or feedback noise is acceptable.

To provide the best possible audio quality as Siri input, the accessory must observe the following recommendations:

- **Echo cancellation and noise suppression (EC/NR)**: Directional microphones and linear beamforming with microphone arrays giving improved SNR are recommended. Linear echo cancellation for reducing unwanted audio sources (such as audio output from the system) without having any other effect on the speech signal are also recommended. However, single channel noise reduction methods (such as spectrum subtraction) must not be applied, as they will be detrimental to the speech recognition accuracy. Similarly, automatic gain control, residual echo suppression and attempts to blank out non-speech periods in the waveform must not be applied.
- **Signal gain**: When adjusting signal levels, the accessory must avoid artifacts, dropouts, and clipping in all circumstances. Automatic Gain Control is not recommended. If the accessory adjusts signal gain, the gain should be held constant across each spoken utterance. The nominal level measured at the uplink output

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

of the accessory should be A-weighted -30 dB ±2 dB root-mean-square (RMS), expressed in units relative to full-scale (dBFS(A)). Alternatively, the nominal level may be 13 dB ±2 dB SLR if using the ITU measurement procedure.

- **Signal-to-noise ratio (SNR)**: An average SNR greater than 20 dB is recommended. Below 20 dB, recognition rates will be impacted.
- **Reverberation**: Maintaining RT60 time at less than 200 ms is recommended.

7.5.1 Wide Band Speech Support

An accessory using Siri should support 16 kHz wide band speech audio for better audio quality and voice recognition performance. See the Bluetooth *Hands-Free Profile* 1.6 specification for details about wide band speech audio. Narrow band audio signal (8 kHz) is supported but not recommended.

7.6 Optimizing the Siri Experience

For best results in using Siri, the accessory design should follow these guidelines:

- The start of a Siri session should not be accompanied by local beeps or verbal indications (such as an announcement of "...voice dialing...") from the accessory. When a Siri session becomes active, the Apple device sends two beeps indicating that Siri is ready to receive instructions. Adding extra audible notifications only inserts delays in the system.
- The accessory should wait for the Apple device to end each Siri session. The accessory should not send an `AT+BVRA=0` command unless it is prompted to do so by user interaction.
- The Apple device expects that the accessory is capable of rendering audio as soon as the SCO connection is active. This is necessary to make sure that the user always hears the Siri introductory beeps with minimum delay. The delay should be within 200 ms.

7.7 Common Siri Applications

Siri can send messages, find points of interests, place phone calls, and much more. As Siri capabilities are constantly growing, additional use cases may become available after the initial integration. In Eyes Free mode, some of these use cases may not be accessible as the user experience is tailored towards a driving scenario.

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

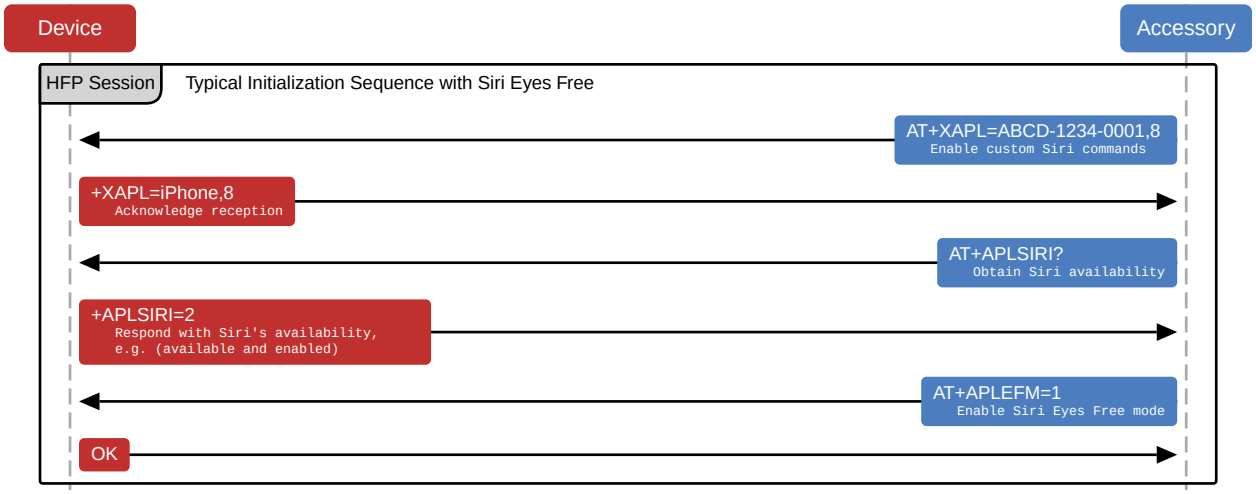
7.7.1 Initialization Procedure After Connection is Established

Figure 7-5 (page 42) outlines the sequence the accessory has to trigger to be able to use Siri on an Apple device. After establishing an HFP profile connection, the accessory must first enable the custom Siri commands by sending `AT+XAPL` and provide the features it supports. After a confirmation is received from the device, the accessory should determine Siri's availability with `AT+APLSIRI?`.

Vehicles with Bluetooth-enabled infotainment systems can also enable Siri Eyes Free Mode during initialization. This is detailed in Figure 7-6 (page 42).

Figure 7-5 Siri Initialization Procedure

Figure 7-6 Siri Initialization Procedure with Siri Eyes Free



7.7.2 Phone Dialing Using Siri

Upon the user's request Siri can initiate an outgoing phone call. The Apple device will initiate HFP call signaling to establish a phone call as described in Bluetooth (page 9). The accessory must be able to transition to Hands-Free dialing at any time during or after a Siri session when signaled by the Apple device.

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

7.7.3 Audio Routing and Media Playback Using Siri

Siri can control the media playback on an Apple device, and if Siri determines that the user wants to play or pause music it will either start, pause or resume media playback. The Apple device will send a notification to the accessory indicating a change in playback state and any associated track information. The accessory must respond to those notifications, start or stop the music playback as requested, as well as update the correct playback state (e.g. shuffle, repeat).

The accessory must not force a change in the playback state after a Siri session is ended. If music was playing before Siri was started, it must continue playing, if it was paused, it must remain paused.

After Siri starts music playback the accessory must set its current audio route to match the audio source, depending on how audio is being received from the Apple device (via Bluetooth or by a wired connection).

The available media playback notifications depend on the audio route being used:

- For a Bluetooth audio route, use the approach described in Notifications (page 17) and Audio Data Received via A2DP Profile (page 19).
- For a wired audio route, use the iPod Accessory Protocol.

7.7.4 Turn-By-Turn Directions Using Siri

Siri can initiate active route guidance that will play turn-by-turn directions. In case the Apple device is the active source and is already playing music, turn-by-turn directions will be mixed in as part of the audio stream. In case the Apple device is not playing music, the accessory should be able to mix in turn-by-turn directions with the active audio source.

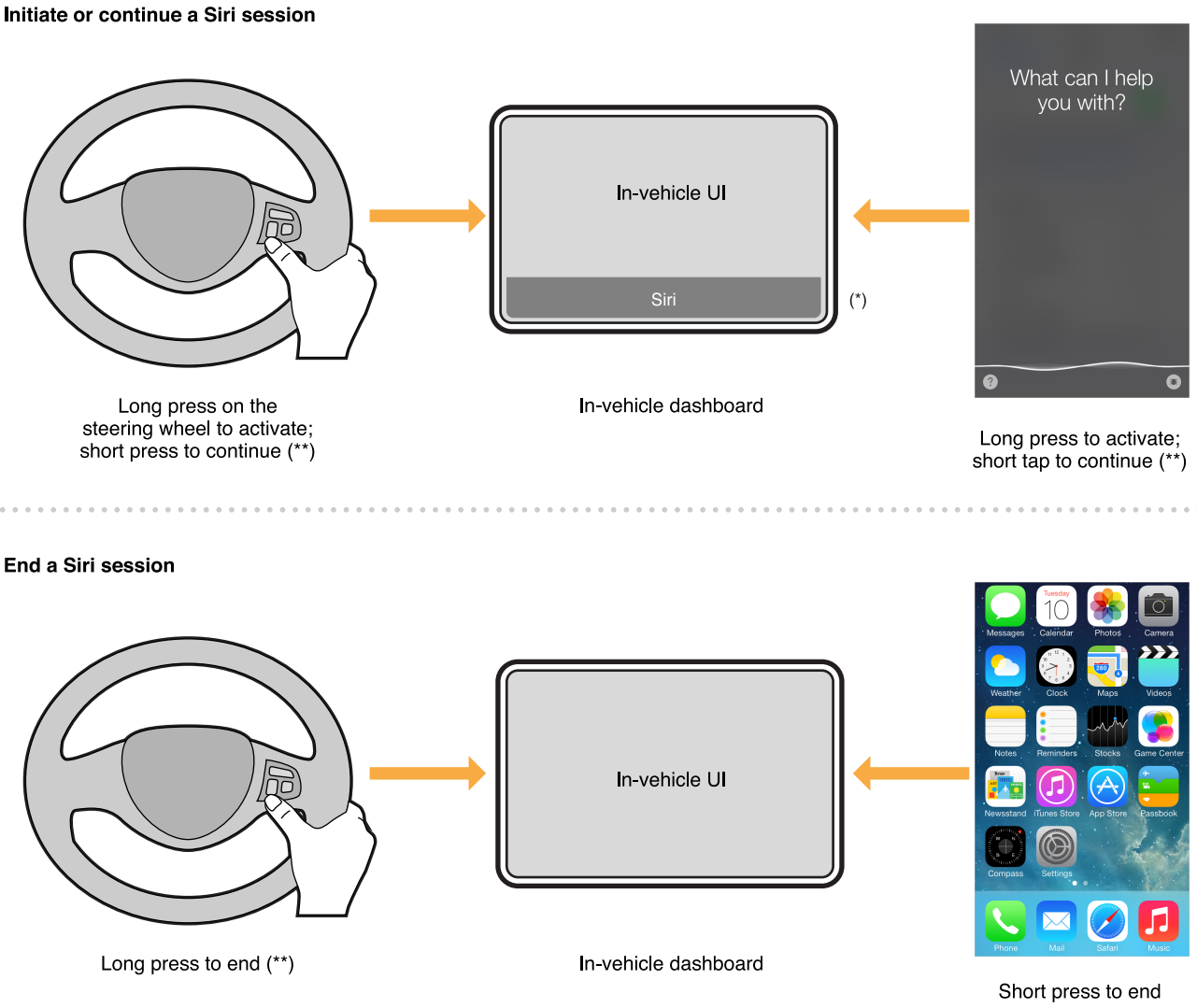
The Apple device will notify the accessory to play turn-by-turn directions only over Bluetooth. Detailed information on how to distinguish between music playback and turn-by-turn notifications is available in Notifications (page 17).

7.8 User Interaction with Siri Eyes Free in a Vehicle

A vehicle that uses Siri Eyes Free mode must integrate the Siri experience with the existing in-vehicle entertainment system and controls. The vehicle should provide a convenient interface to initiate, continue, and end a Siri session. Once a Siri session is running, the vehicle must display a visual cue that voice recognition is in process. Figure 7-7 (page 44) outlines how a Siri interaction should be designed.

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

Figure 7-7 Siri Eyes Free User Interaction



(*) If the accessory wishes to indicate that Siri is active, it must do one of the following:

- Display the word 'Siri' (as capitalized) with no additional text or icon.
- Use generic text or icon that does not resemble the Siri microphone icon.

(**) If the vehicle is equipped with steering wheel controls, a dedicated button or a long-press on a button on the vehicle's steering wheel is required to start, continue and end a Siri session. The button long-press must be 600 ms or less. If no steering wheel controls are available, a soft button must be available within the in-vehicle user interface to start, continue or end a Siri session.

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

When a vehicle enables Siri Eyes Free mode, the Apple device will not display any onscreen Siri content. If the device was locked at the time the Siri session was activated from the vehicle, it will remain locked and the screen will not turn on. If the user unlocks or manually activates the device while in an Eyes Free Session there will be a notification that the device is in an active Siri session but there will be no visual Siri content displayed.

7.9 Enabling/Disabling Siri from the Apple Device

The user has the ability to disable or enable Siri from the Settings menu on the Apple device. When Siri is disabled, Voice Control becomes the recognition engine on the device and will be triggered by default. The accessory may choose to either launch Voice Control with no further changes, in the same way Siri is launched (shown in Figure 7-8 (page 45)) or display a warning message and not send an activation command to the Apple device (Figure 7-9 (page 45)).

Figure 7-8 Siri is Deactivated - Launching Voice Control

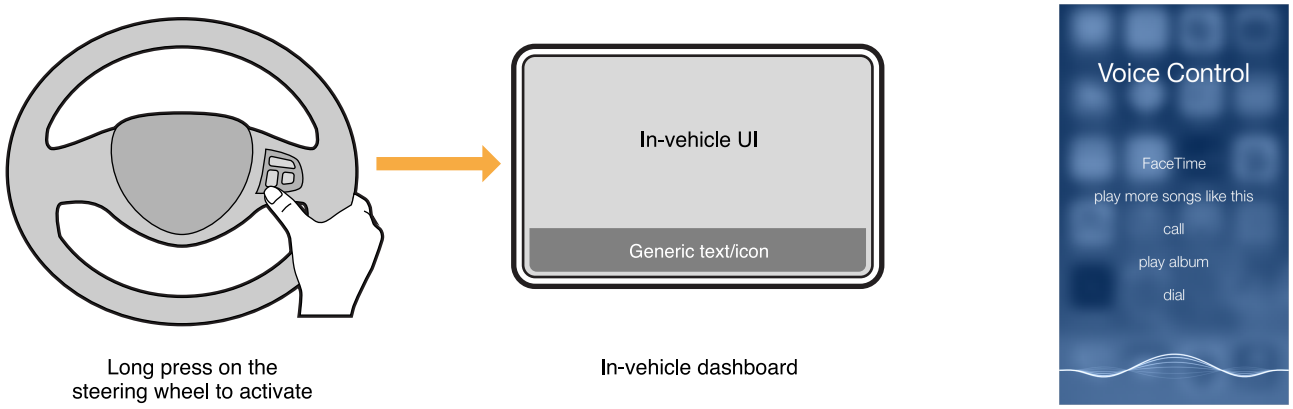
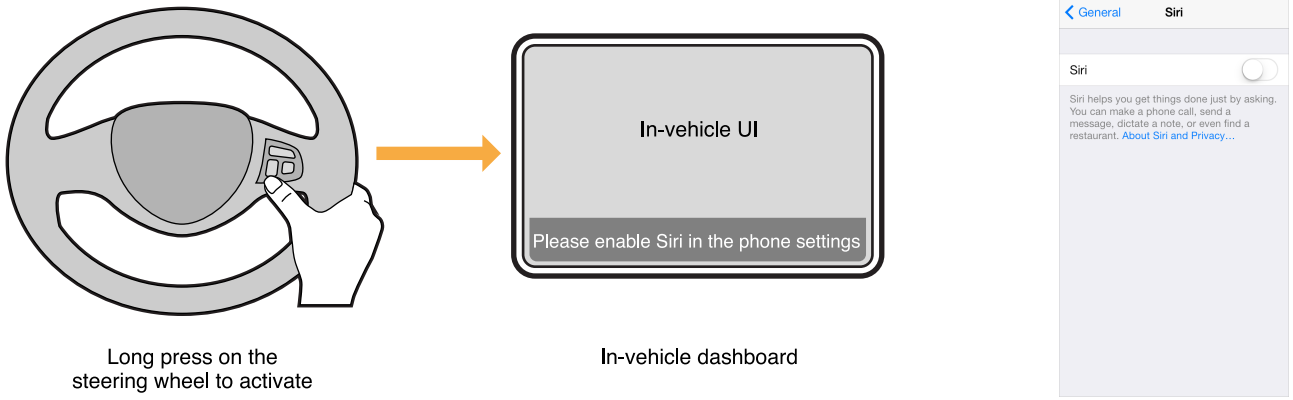


Figure 7-9 Siri is Deactivated - Displaying a Warning Message



7.10 Test Procedures

7.10.1 Siri Eyes Free

The following test procedures are applicable to accessories that interact with Siri Eyes Free.

For the following spoken tests, the speaker should ideally be a native speaker of North American English. If the tester's native language is not English, set Siri to your native language and translate the phrases to be spoken below into your native language.

7.10.1.1 General

1. Pair and establish a Bluetooth Handsfree Profile (HFP) connection between the iPhone and the head unit. Activate Siri from the vehicle steering wheel button (e.g. by press hold):
    a. Observe that the iPhone screen remains inactive after a Siri session has started (a purple status bar will be visible on the Apple device if the screen is activated manually).
    b. Ensure that Siri opening chime is heard completely through the vehicle speakers.
    c. Observe a visual notification in the in-car User Interface (UI) that a Siri session is active (textual notification, on-screen UI, etc.).
2. Activate Siri from the vehicle steering wheel button and say "Send a message to Peter. How are you?". While still saying the message, press the vehicle steering wheel button to cancel Siri (e.g. by press):
    a. Ensure the iPhone screen remains inactive (if manually activated, the purple bar on the phone will disappear).
    b. Verify that the in-car UI for Siri interaction dismisses and the head unit resumes the state before Siri's interaction.
3. Activate Siri from the vehicle steering wheel button and say "How is the weather in San Francisco?". Wait for Siri to respond with the weather forecast. Once the weather forecast is complete, resume Siri from the vehicle steering wheel button and say "What about New York?":
    a. Observe that the purple bar is still active on the phone.
    b. Listen for the Siri opening chime.
    c. The vehicle UI should be displaying an on-going Siri session.
    d. Verify that Siri responds with the weather forecast for New York.
4. In case the vehicle UI offers on-screen controls to activate/cancel/resume Siri, repeat steps (1) to (3) for all on-screen controls.
5. Activate Siri from the steering wheel button and say "What's the time". Listen to the current time and do not interact with Siri or the iPhone. After 5 seconds have expired:

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

a. Observe that the purple bar on the phone is no longer visible.

b. The in-car UI for Siri interaction should be dismissed.

c. The head unit should resume the state before Siri's interaction.

6. Listen to FM radio from the car speakers (no A2DP streaming active). Press and hold phone's home button to activate Siri from the phone:

   a. Observe a visual notification in the in-car UI that a Siri session is active (textual notification, on-screen UI, etc.).

   b. Observe Siri's interaction on the phone's screen and ask "What's the time?"

   c. After Siri has responded, lock the phone again to dismiss the Siri session by pressing the phone's sleep/wake button.

7. On the phone go to Settings and turn off Siri. Activate Siri from the head unit. Observe one of the following depending on the actual implementation (a) Voice Control starts instead of Siri (b) The head unit displays a warning that Siri Eyes Free is not available.

8. On the phone go to Settings and turn Siri back on. Verify that Siri can be activated/cancelled from the head unit and from the Home button on the phone.

9. Turn Bluetooth off from the Settings on the phone. Verify that Siri cannot be started.

10. Turn Bluetooth back on from the Settings on the phone. Verify that Bluetooth HFP profile reconnects and that Siri can be activated/cancelled from the head unit and from the Home button on the phone.

11. Make sure there is no accessory battery status level indicator icon displayed on the phone's status bar.

7.10.1.2 Siri Dialog

1. Activate Siri from the vehicle's steering wheel button and say "Send a text message to **insert contact name**". When Siri prompts for "what would you like it to say", dictate a short message. After Siri has read back your dictated message, say "Review it". After Siri has read back the message again, say "Review it" again. Repeat this cycle ~5 times to ensure that the head unit is able to handle a long interaction with Siri. At the end say "Send it" and verify that the message is sent. Verify that the opening chime is audible and the message is sent. After the Siri session is closed, the audio playback should go back to the state it was before Siri was started, i.e. if it was paused remains paused, if it was playing remains playing.

2. Start Siri from the vehicle's steering wheel button and ask for directions. Follow up through the dialog until the navigation is started. Verify that the Siri session is closed and that the audio playback goes back to the state it was before Siri was started, e.g. if it was paused remains paused, if it was playing remains playing.

3. Start Siri from the vehicle's steering wheel button and say "Search the web for polar bears". Verify that Eyes Free mode is on and that this use case is blocked by Siri. Note: In some implementations the vehicle has to be in motion before Eyes Free is activated by the car kit.

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

4. Start Siri from the vehicle's steering wheel button and say "What is the current time in Munich?". After Siri has answered but before ~5 seconds have elapsed, resume Siri (e.g. by a short press on the steering wheel button) and verify that Siri is initiated again. Say "What about San Francisco?". Repeat (with a different city) and verify that this can continue indefinitely as long as you short press on the steering wheel button within 5 seconds of the last response.

7.10.1.3 Bluetooth HFP A2DP Music

1. Establish a Bluetooth A2DP connection and switch to Bluetooth audio source on the head unit. Activate Siri and ask "Next track". Verify that track advances and that audio is playing through vehicle speakers. Verify that the Siri in-car UI is dismissed and the head unit goes back to its initial state.
2. Activate Siri and say "Pause the music". Verify that audio remains paused after Siri has been dismissed. Verify that the Siri in-car UI is dismissed and the head unit goes back to its initial state.
3. Pause music playback on the head unit (via AVRCP command). Activate Siri and ask "What time is it?". Verify that the music playback remains paused after the Siri session has been dismissed. Verify that the Siri in-car UI is dismissed and the head unit goes back to its initial state.
4. Switch to FM radio on the head unit. Activate Siri and ask "Play me a song". Verify that head unit is able to automatically switch to BT audio and iPhone music starts playing. Verify that the beginning of the selected track is heard, e.g. there is no skipping of audio packets. Verify that the Siri in-car UI is dismissed and the head unit goes back to its initial state.
5. Activate Siri and ask "Shuffle all songs". Verify that head unit correctly updates NowPlaying track information. Verify that the Siri in-car UI is dismissed and the head unit goes back to its initial state.
6. Activate Siri and ask to play a specific artist or title. Verify that the Siri session is dismissed after the music starts. Make sure the correct metadata is displayed on the screen. Verify that the Siri in-car UI is dismissed and the head unit goes back to its initial state.

7.10.1.4 Call

1. Activate Siri and call a contact with more than one phone number (home and mobile). Wait for Siri's response to ask for which phone number to call. Answer with "home". Verify that call transition is handled correctly by the head unit and any Siri UI displayed on the vehicle screen is dismissed.
2. While iPhone music is playing, activate Siri and say "Call **insert contact name to call**". Verify that call transition is handled correctly by the head unit. Verify that iPhone music playback resumes after the call has been answered and terminated on the far end. Verify that the Siri in-car UI is dismissed and the head unit goes back to its initial state.

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

3. While iPhone music is playing, start Siri and say "Call **insert contact name to call**". Verify that call transition is handled correctly by the head unit. Verify that iPhone music playback resumes after the call has been answered and terminated on the near end (i.e. on the head unit). Verify that the Siri in-car UI is dismissed and the head unit goes back to its initial state.
4. While in a Siri session, receive an incoming call on the head unit. Verify that head unit handles call-signaling correctly and transitions to phone UI once the call has been accepted. Verify that the Siri in-car UI is dismissed and the head unit goes back to its initial state.

7.10.1.5 Bluetooth + Wired iAP

1. Connect Apple device to head unit via 30-pin or Lightning connector (iPhone 4s / iPhone 5 respectively). Switch to iPod music and verify that audio is playing. Activate Siri and say "Next track". Verify that track advances and head unit displays track metadata correctly. Verify that the Siri in-car UI is dismissed and the head unit goes back to its initial state.
2. From the head unit UI, select a playlist with a single song and start playing it. Start Siri from the vehicle steering wheel and say "Play ..... **make sure to select a song to play that is (a) not in the same album as the single-track playlist and (b) not song track index 0 of its album**". Verify that the new song starts playing and that the head unit displays the track metadata for this new song correctly. Verify that the Siri in-car UI is dismissed and the head unit goes back to its initial state.
3. Turn Shuffle off on the head unit UI. Then start Siri and say "Shuffle all songs". Verify that the shuffle indicator on the head unit UI is updated and the correct track metadata for the new now playing song is displayed correctly Verify that the Siri in-car UI is dismissed and the head unit goes back to its initial state.
4. Switch to FM radio on the head unit. Activate Siri and say "Play me a song". Verify that head unit is able to automatically switch to iPOD audio source and that audio starts playing through the speakers. Verify that there is no skipping of audio at the beginning of the selected track. Verify that the Siri in-car UI is dismissed and the head unit goes back to its initial state.
5. Pause music playback on the head unit (via iAP commands). Activate Siri and ask "What time is it?". Verify that music playback remains paused after Siri session has been dismissed. Verify that the Siri in-car UI is dismissed and the head unit goes back to its initial state.
6. While iPhone music is playing, start Siri and say "Call **insert contact name to call**". Verify that call transition is handled correctly by the head unit. Verify that iPhone music playback resumes after the call has been answered and terminated on the far end. Verify that the Siri in-car UI is dismissed and the head unit goes back to its initial state.
7. While iPhone music is playing, start Siri and say "Call **insert contact name to call**". Verify that call transition is handled correctly by the head unit. Verify that iPhone music playback resumes after the call has been answered and terminated on the near end (i.e. on the head unit). Verify that the Siri in-car UI is dismissed and the head unit goes back to its initial state.

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

8. Pause music playback on the head unit (via iAP commands). Start Siri and say "Call **insert contact name to call**". Verify that call transition is handled correctly by the head unit. Verify that iPhone music playback remains paused after the call has been answered and terminated on the far end. Verify that the Siri in-car UI is dismissed and the head unit goes back to its initial state.

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

8. iPod Accessory Protocol

Third-party accessories can use the iPod Accessory Protocol (iAP) to access advanced features of Apple devices. One such feature is the ability to communicate securely with third-party iOS applications via the iOS External Accessory Framework. For information about the External Accessory Framework, see http://developer.apple.com/library/ios/#featuredarticles/ExternalAccessoryPT/Introduction/Introduction.html.

To incorporate iAP into an accessory design, the accessory developer must be a member of the Apple MFi licensing program and integrate specific MFi hardware into the accessory. For further information about MFi, see https://developer.apple.com/programs/mfi.

2017-06-16 | Copyright © 2017 Apple Inc. All Rights Reserved.

Apple Inc.
Copyright © 2017 Apple Inc.
All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, mechanical, electronic, photocopying, recording, or otherwise, without prior written permission of Apple Inc., with the following exceptions: Any person is hereby authorized to store documentation on a single computer for personal use only and to print copies of documentation for personal use provided that the documentation contains Apple's copyright notice.

No licenses, express or implied, are granted with respect to any of the technology described in this document. Apple retains all intellectual property rights associated with the technology described in this document. This document is intended to be used in the development of solutions for Apple-branded products.

Apple Inc.
1 Infinite Loop
Cupertino, CA 95014
408-996-1010

Apple, the Apple logo, and iPhone are trademarks of Apple Inc., registered in the U.S. and other countries.

IOS is a trademark or registered trademark of Cisco in the U.S. and other countries and is used under license.

Even though Apple has reviewed this document, APPLE MAKES NO WARRANTY OR REPRESENTATION, EITHER EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT, ITS QUALITY, ACCURACY, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. AS A RESULT, THIS DOCUMENT IS PROVIDED "AS IS," AND YOU, THE READER, ARE ASSUMING THE ENTIRE RISK AS TO ITS QUALITY AND ACCURACY.

IN NO EVENT WILL APPLE BE LIABLE FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES RESULTING FROM ANY DEFECT OR INACCURACY IN THIS DOCUMENT, even if advised of the possibility of such damages.

THE WARRANTY AND REMEDIES SET FORTH ABOVE ARE EXCLUSIVE AND IN LIEU OF ALL OTHERS, ORAL OR WRITTEN, EXPRESS OR IMPLIED. No Apple dealer, agent, or employee is authorized to make any modification, extension, or addition to this warranty.

Some states do not allow the exclusion or limitation of implied warranties or liability for incidental or consequential damages, so the above limitation or exclusion may not apply to you. This warranty gives you specific legal rights, and you may also have other rights which vary from state to state.