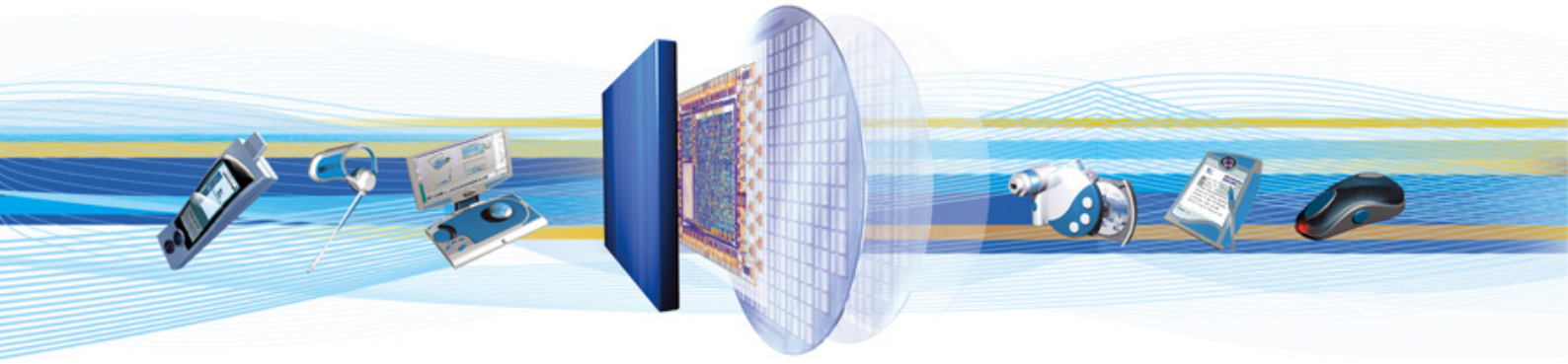
CSR Synergy Bluetooth 18.2.0

OBEX Basic Imaging Profile Server

API Description

November 2011



Cambridge Silicon Radio Limited

Churchill House
Cambridge Business Park
Cowley Road
Cambridge CB4 0WZ
United Kingdom

Registered in England and Wales 3665875

Tel: +44 (0)1223 692000
Fax: +44 (0)1223 692001
www.csr.com

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

Contents

1 Introduction 5

1.1 Introduction and Scope 5

1.2 Assumptions 5

2 Description 6

2.1 Introduction 6

2.2 Reference Model 6

2.3 Sequence Overview 7

3 Interface Description 9

3.1 Common Interfaces 9

3.1.1 Relations between Application and BIPS Profile 9
3.1.2 Activation 9
3.1.3 Connect without Authentication 10
3.1.4 Connect with Automatic Archive 10
3.1.5 Connect with Authentication 11
3.1.6 Abort Handling 12
3.1.7 Disconnect 12
3.1.8 Disconnect with Automatic Archive 13
3.1.9 Deactivation 14
3.1.10 Get Instances Identifier 15

3.2 Image Push Interfaces 15

3.2.1 Get Capabilities 15
3.2.2 Put Image 16
3.2.3 Put Linked Thumbnail 16
3.2.4 Put Linked Attachment 17

3.3 Remote Camera Interfaces 17

3.3.1 Get Monitoring Image 18
3.3.2 Get Image Properties 18
3.3.3 Get Image 19
3.3.4 Get Linked Thumbnail 19

3.4 Automatic Archive Interfaces 20

3.4.1 Get Image List 20
3.4.2 Get Capabilities 21
3.4.3 Get Image Properties 21
3.4.4 Get Image 22
3.4.5 Get Linked Attachment 23
3.4.6 Get Linked Thumbnail 23
3.4.7 Delete Image 24
3.4.8 Abort 24

3.5 Payload Encapsulated Data 25

3.5.1 Using Offsets 25
3.5.2 Payload Memory 25

4 OBEX Basic Imaging Profile Responder Primitives 26

4.1 List of All Primitives 26

4.2 Common Primitives 28

4.2.1 CSR_BT_BIPS_ACTIVATE 28
4.2.2 CSR_BT_BIPS_DEACTIVATE 30
4.2.3 CSR_BT_BIPS_CONNECT 31
4.2.4 CSR_BT_BIPS_AUTHENTICATE 33
4.2.5 CSR_BT_BIPS_ABORT 35
4.2.6 CSR_BT_BIPS_DISCONNECT 36
4.2.7 CSR_BT_BIPS_GET_INSTANCES_QID 37
4.2.8 CSR_BT_BIPS_SECURITY_IN 38
4.2.9 CSR_BT_BIPS_CHALLENGE 40

4.3 Image Push Primitives 42

4.3.1 CSR_BT_BIPS_PUSH_GET_CAPABILITIES_HEADER 42

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

4.3.2 CSR_BT_BIPS_PUSH_GET_CAPABILITIES_OBJECT 44
4.3.3 CSR_BT_BIPS_PUSH_PUT_IMAGE_HEADER 46
4.3.4 CSR_BT_BIPS_PUSH_PUT_IMAGE_FILE 48
4.3.5 CSR_BT_BIPS_PUSH_PUT_LINKED_THUMBNAIL_HEADER 50
4.3.6 CSR_BT_BIPS_PUSH_PUT_LINKED_THUMBNAIL_FILE 52
4.3.7 CSR_BT_BIPS_PUSH_PUT_LINKED_ATTACHMENT_HEADER 54
4.3.8 CSR_BT_BIPS_PUSH_PUT_LINKED_ATTACHMENT_FILE 56
4.4 Remote Camera Primitives 58
4.4.1 CSR_BT_BIPS_RC_GET_MONITORING_IMAGE_HEADER 58
4.4.2 CSR_BT_BIPS_RC_GET_MONITORING_IMAGE_OBJECT 60
4.4.3 CSR_BT_BIPS_RC_GET_IMAGE_PROPERTIES_HEADER 62
4.4.4 CSR_BT_BIPS_RC_GET_IMAGE_PROPERTIES_OBJECT 64
4.4.5 CSR_BT_BIPS_RC_GET_IMAGE_HEADER 66
4.4.6 CSR_BT_BIPS_RC_GET_IMAGE_OBJECT 68
4.4.7 CSR_BT_BIPS_RC_GET_LINKED_THUMBNAIL_HEADER 70
4.4.8 CSR_BT_BIPS_RC_GET_LINKED_THUMBNAIL_OBJECT 72
4.5 Automatic Archive Primitives 74
4.5.1 CSR_BT_BIPS_AA_GET_IMAGE_LIST & CSR_BT_BIPS_AA_GET_IMAGE_LIST_HEADER 74
4.5.2 CSR_BT_BIPS_AA_GET_CAPABILITIES 77
4.5.3 CSR_BT_BIPS_AA_GET_IMAGE_PROPERTIES 79
4.5.4 CSR_BT_BIPS_AA_GET_IMAGE 81
4.5.5 CSR_BT_BIPS_AA_GET_LINKED_ATTACHMENT 83
4.5.6 CSR_BT_BIPS_AA_GET_LINKED_THUMBNAIL 85
4.5.7 CSR_BT_BIPS_AA_DELETE_IMAGE 87
4.5.8 CSR_BT_BIPS_AA_ABORT 88
5 Document References 89

List of Figures

Figure 1: Reference model 6
Figure 2: BIPS state diagram – Image Push and Remote Camera 7
Figure 3: BIPS state diagram – Automatic Archive 8
Figure 4: A single application handling multiple BIPS instances 9
Figure 5: Multiple applications handling multiple BIPS instances 9
Figure 6: Activation handling 10
Figure 7: Connection handling without authentication 10
Figure 8: System MSC of the OBEX connections setup needed for Automatic Archive 11
Figure 9: Connection handling with authentication 12
Figure 10: Abort handling 12
Figure 11: Normal disconnect handling for Image Push and Remote Camera, abnormal for Automatic archive 12
Figure 12: Normal disconnect for Automatic Archive 13
Figure 13: System MSC of the normal disconnect scenario showing OBEX disconnections 13
Figure 14: System MSC of the abnormal disconnect scenario showing OBEX disconnections 14
Figure 15: Deactivation request handling 14
Figure 16: Deactivation indication handling 14
Figure 17: Get Instances QID handling 15
Figure 18: Image Push Get Capabilities handling 15
Figure 19: Image Push Put Image handling 16
Figure 20: Image Push Put Linked Thumbnail handling 17
Figure 21: Image Push Put Linked Attachment handling 17
Figure 22: Remote Camera Get Monitoring Image handling 18
Figure 23: Remote Camera Get Image Properties handling 19
Figure 24: Remote Camera Get Image handling 19
Figure 25: Remote Camera Get Linked Thumbnail handling 20

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

Figure 26: Automatic Archive Get Image List handling....21
Figure 27: Automatic Archive Get Capabilities handling....21
Figure 28: Automatic Archive Get Image Properties handling....22
Figure 29: Automatic Archive Get Image handling....22
Figure 30: Automatic Archive Get Linked Attachment handling....23
Figure 31: Automatic Archive Get Linked Thumbnail handling....24
Figure 32: Automatic Archive Delete Image handling....24
Figure 33: Automatic Archive Abort handling....25

List of Tables

Table 1: List of all primitives....27
Table 2: CSR_BT_BIPS_ACTIVATE Primitives....28
Table 3: CSR_BT_BIPS_DEACTIVATE Primitives....30
Table 4: CSR_BT_BIPS_CONNECT Primitives....31
Table 5: CSR_BT_BIPS_AUTHENTICATE Primitives....33
Table 6: CSR_BT_BIPS_ABORT Primitives....35
Table 7: CSR_BT_BIPS_DISCONNECT Primitives....36
Table 8: CSR_BT_BIPS_REGISTER_QID Primitives....37
Table 9: CSR_BT_BIPS_SECURITY_IN Primitives....38
Table 10: CSR_BT_BIPS_CHALLENGE Primitives....40
Table 11: CSR_BT_BIPS_PUSH_GET_CAPABILITIES_HEADER Primitives....42
Table 12: CSR_BT_BIPS_PUSH_GET_CAPABILITIES_OBJECT Primitives....44
Table 13: CSR_BT_BIPS_PUSH_PUT_IMAGE_HEADER Primitives....46
Table 14: CSR_BT_BIPS_PUSH_PUT_IMAGE_FILE Primitives....48
Table 15: CSR_BT_BIPS_PUSH_PUT_LINKED_THUMBNAIL_HEADER Primitives....50
Table 16: CSR_BT_BIPS_PUSH_PUT_LINKED_THUMBNAIL_FILE Primitives....52
Table 17: CSR_BT_BIPS_PUSH_PUT_LINKED_ATTACHMENT_HEADER Primitives....54
Table 18: CSR_BT_BIPS_PUSH_PUT_LINKED_ATTACHMENT_FILE Primitives....56
Table 19: CSR_BT_BIPS_RC_GET_MONITORING_IMAGE_HEADER Primitives....58
Table 20: CSR_BT_BIPS_RC_GET_MONITORING_IMAGE_OBJECT Primitives....60
Table 21: CSR_BT_BIPS_RC_GET_IMAGE_PROPERTIES_HEADER Primitives....62
Table 22: CSR_BT_BIPS_RC_GET_IMAGE_PROPERTIES_OBJECT Primitives....64
Table 23: CSR_BT_BIPS_RC_GET_IMAGE_HEADER Primitives....66
Table 24: CSR_BT_BIPS_RC_GET_IMAGE_OBJECT Primitives....68
Table 25: CSR_BT_BIPS_RC_GET_LINKED_THUMBNAIL_HEADER Primitives....70
Table 26: CSR_BT_BIPS_RC_GET_LINKED_THUMBNAIL_OBJECT Primitives....72
Table 27: CSR_BT_BIPS_AA_GET_IMAGE_LIST & CSR_BT_BIPS_AA_GET_IMAGE_LIST_HEADER Primitives....74
Table 28: CSR_BT_BIPS_AA_GET_CAPABILITIES Primitives....77
Table 29: CSR_BT_BIPS_AA_GET_IMAGE_PROPERTIES Primitives....79
Table 30: CSR_BT_BIPS_AA_GET_IMAGE Primitives....81
Table 31: CSR_BT_BIPS_AA_GET_LINKED_ATTACHMENT Primitives....83
Table 32: CSR_BT_BIPS_AA_GET_LINKED_THUMBNAIL Primitives....85
Table 33: CSR_BT_BIPS_AA_DELETE_IMAGE Primitives....87
Table 34: CSR_BT_BIPS_AA_ABORT Primitives....88

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

1 Introduction

1.1 Introduction and Scope

This document describes the message interface provided by the OBEX Basic Imaging Profile (BIPS). BIPS conforms to the responder side of the Image Push Feature, Remote Camera Feature and the Automatic Archive Feature, ref. [BIP].

1.2 Assumptions

The following assumptions and preconditions are made in the following:

- There is a secure and reliable transport between the profile part, i.e. BIPS and the application
- The BIPS shall only handle one request at a time
- The client only authenticates BIPS doing a connect session
- Bonding (pairing) is NOT handled by the BIPS

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

2 Description

2.1 Introduction

The scenarios covered by this profile are the following:

- Usage of a Bluetooth® device e.g. a camera to send one or more images to another Bluetooth® device e.g. a mobile phone
- Usage of a Bluetooth® device e.g. a mobile phone to monitor and capture images on a Bluetooth camera device
- Usage of a Bluetooth® device e.g. a mobile phone to automatically archive images on another Bluetooth® device e.g. a computer

The OBEX BIP server must be activated by the application. When it is activated it is able to provide the application with incoming images and request monitoring images from the application. Furthermore, if an Automatic Archive connection is made the application is able to retrieve images from the client using the server. The scheme used for selecting which images to retrieve from the client is to be determined by the application.

The BIPS provides Service Discovery handling.

The BIPS is handling the interpretation of the OBEX packet.

The application is responsible for handling the indications from the BIPS and sending the correct responses. The response codes used are described in the IrOBEX Specification [OBEX]. The BIPS does not check and verify the data in the responses. Thus, it is the responsibility of the application to make sure that data follows the appropriate standards and formats. For further details on this subject please consult ref. [BIP] and [OBEX].

2.2 Reference Model

The BIPS interfaces to the Connection Manager (CM).

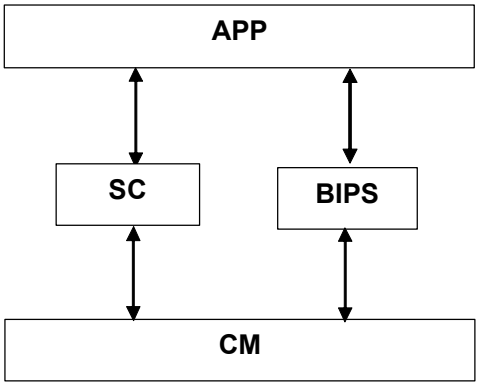


Figure 1: Reference model

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

2.3 Sequence Overview

The BIPS is capable of performing three different roles that are quite separated. For this reason the sequence overview figure is divided into two. Figure 2 illustrates the Image Push and the Remote Camera features and Figure 3 illustrates the Automatic Archive. No matter which feature is going to be used the BIPS starts up being in IDLE state. When the application activates BIPS, the server enters ACTIVE state and is ready to handle incoming requests. The incoming connections are for one specific function, i.e. Image Push, Remote Camera or Automatic Archive. When a connection for Remote Camera is made, only Remote Camera features are available. If Image Push features are needed the profile must be disconnected and an Image Push connection must be established. The server re-enters ACTIVE state when a connection is disconnected, and re-enters IDLE when deactivated.

Automatic Archive in Figure 3 uses two connections to perform its operations. This should, however, be invisible to the application using the profile.

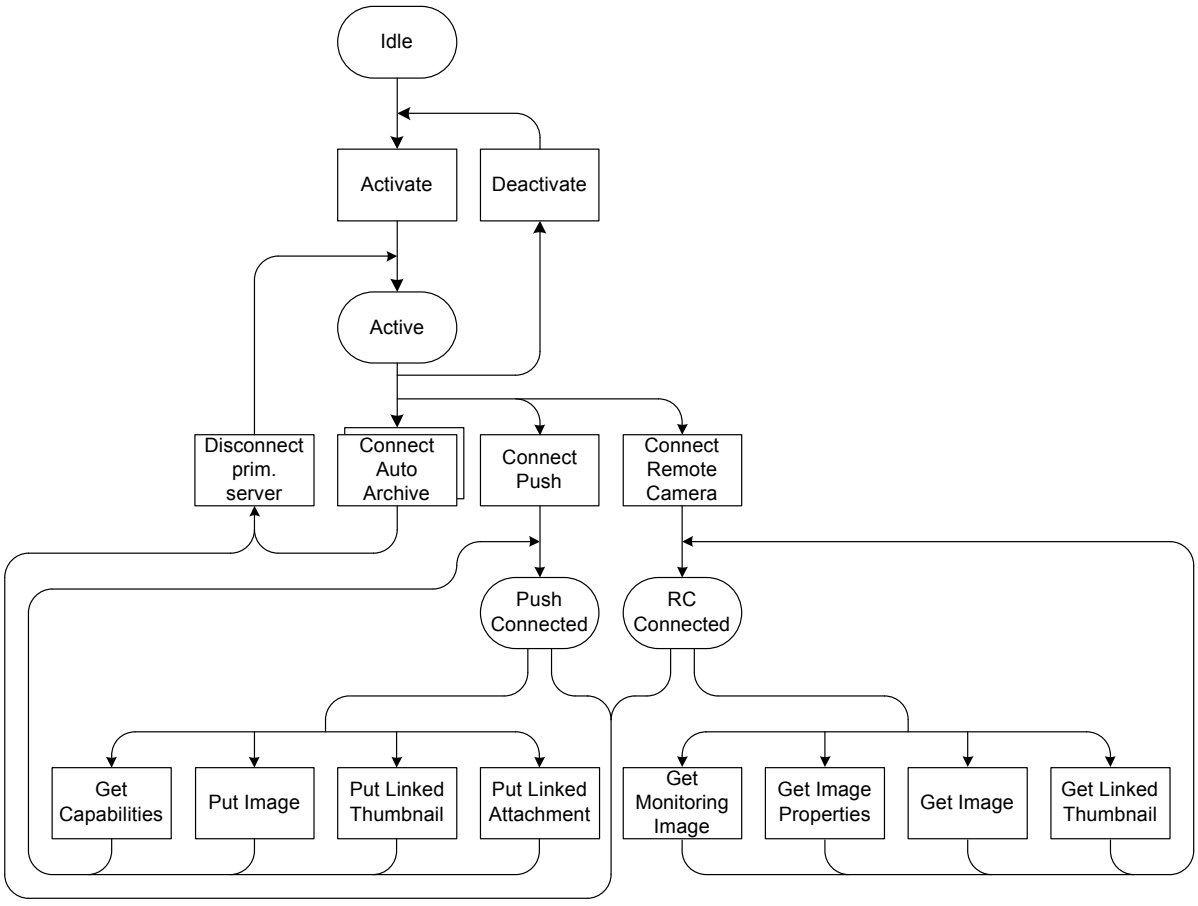


Figure 2: BIPS state diagram – Image Push and Remote Camera

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

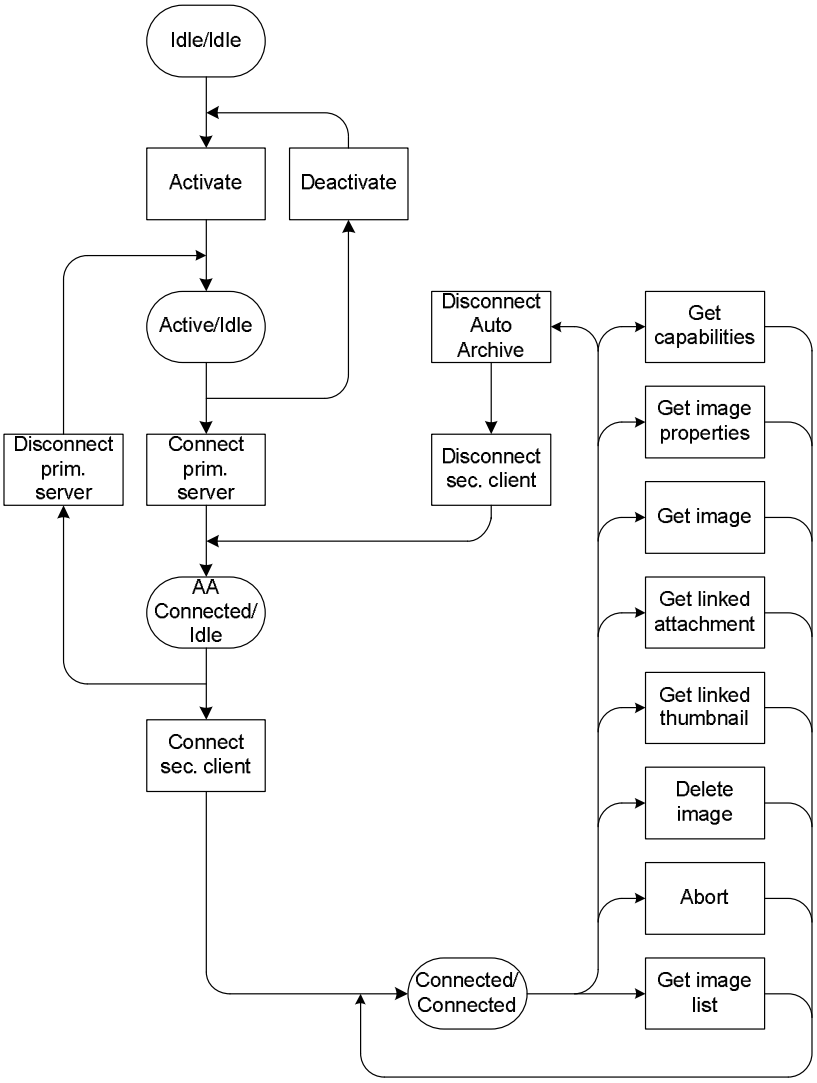


Figure 3: BIPS state diagram – Automatic Archive

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

3 Interface Description

The BIPS can perform three different operations depending on which type of connection is made. Therefore, this section is divided into four parts: a common part that describes the interfaces that are the same for all three session types; Image Push specific interfaces; Remote Camera specific interfaces; and finally Automatic Archive specific interfaces.

3.1 Common Interfaces

The sequences in this section are common to the profile and not dependant on which type of connection is being made.

3.1.1 Relations between Application and BIPS Profile

It is possible to run multiple instances of BIPS simultaneously. These instances can be handled by one or multiple application. This is illustrated in Figure 4 and Figure 5 respectively.

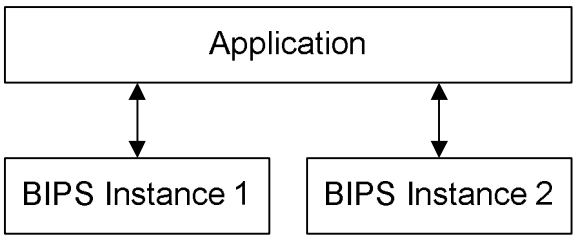


Figure 4: A single application handling multiple BIPS instances

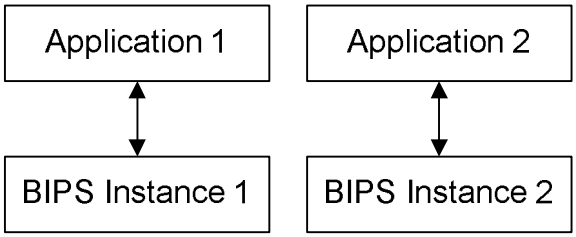


Figure 5: Multiple applications handling multiple BIPS instances

Each instance of the BIPS profile has its own queue id, e.g. if the application must send a message to BIPS Instance 2, it must be put onto the message queue of BIPS Instance 2. The message queues of the different instances may be retrieved by means of the `CSR_BT_BIPS_GET_INSTANCES_QID_REQ/CFM` messages.

BIPS instance 1 is special because it is the BIP-manager where all other BIPS instances register themselves.

This means that in order to get the BIPS message queue for the given instance the `CSR_BT_BIPS_GET_INSTANCES_QID_REQ` is to be sent to queue ID = CSR_BT_BIPS_IFACEQUEUE.

Csr_bt_bips_lib.h defines a BIPS access library, which provides functions for building and sending downstream BIPS primitives. These functions provide a parameter (pHandleInst) which is the queue ID of the BIPS instance that the message is to be sent to. It is the responsibility of the application to remember the instance QID and provide it in all messages.

3.1.2 Activation

Sending a CSR_BT_BIPS_ACTIVATE_REQ activates BIPS. The BIPS then registers a Service Record in the Service Discovery Server and make it connectable. The BIPS is now ready to handle incoming requests.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

Figure 6: Activation handling

Please note that whether or not the Bluetooth device will be discoverable, i.e. can be found by other Bluetooth devices, it must be controlled by the application. For more information, please refer to [CM]. After initialization of CSR Synergy Bluetooth the Bluetooth® device is set up to be discoverable.

3.1.3 Connect without Authentication

When the client is making an OBEX connect request against the server, the first message the application receives is CSR_BT_BIPS_CONNECT_IND, which the application must respond with a CSR_BT_BIPS_CONNECT_RES message with the appropriate result code. The CSR_BT_BIPS_CONNECT_IND contains information about which type of session is requested.

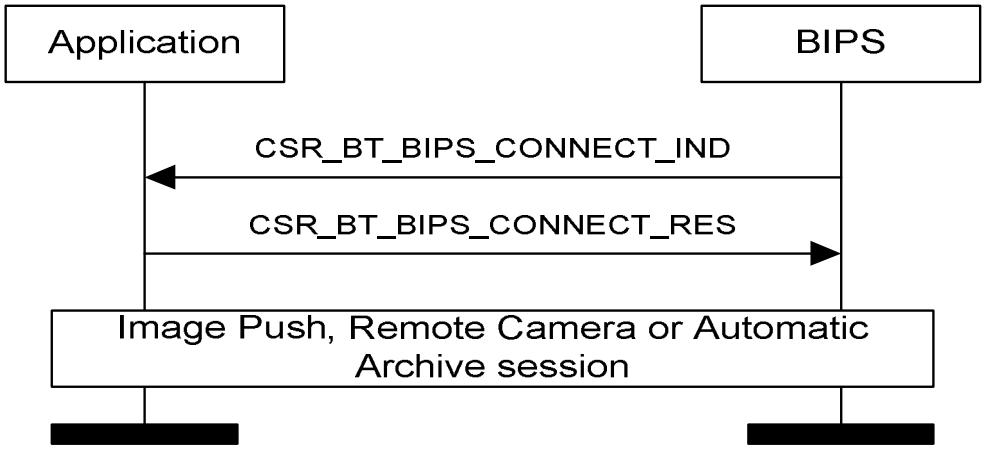


Figure 7: Connection handling without authentication

3.1.4 Connect with Automatic Archive

During the connection establishment of the automatic archive the BIPS profile takes full responsibility of establishing the needed primary (green) and secondary (blue) OBEX connection.

The automatic archive session is established under the control of the BIPS profile as follows: The BIP client peer initiates a primary (green) OBEX connection. Following the connect response BIPS awaits the OBEX request ‘start archive’. When the start archive request has been received and responded to, BIPS will take the role as client and request a new secondary (blue) OBEX connection. When this is granted the application is finally informed of the automatic archive connection.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR’s non-disclosure agreement.

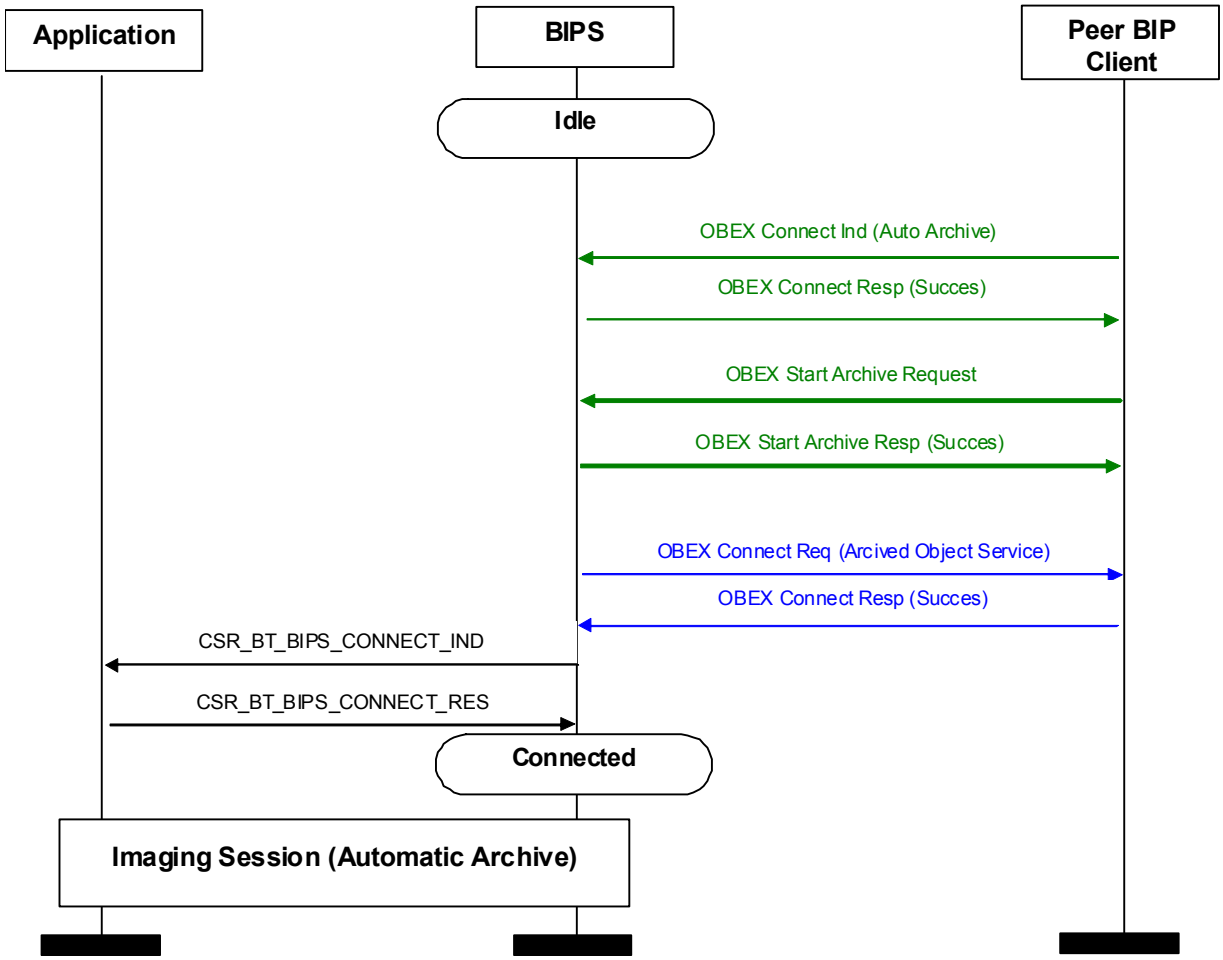


Figure 8: System MSC of the OBEX connections setup needed for Automatic Archive

Upon the establishment of the automatic archive imaging session, the application must take the role of client.

3.1.5 Connect with Authentication

If the client doing the OBEX connect request has authenticated the server, the application will receive a CSR_BT_BIPS_AUTHENTICATE_IND, which the application must respond with a CSR_BT_BIPS_AUTHENTICATE_RES message.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

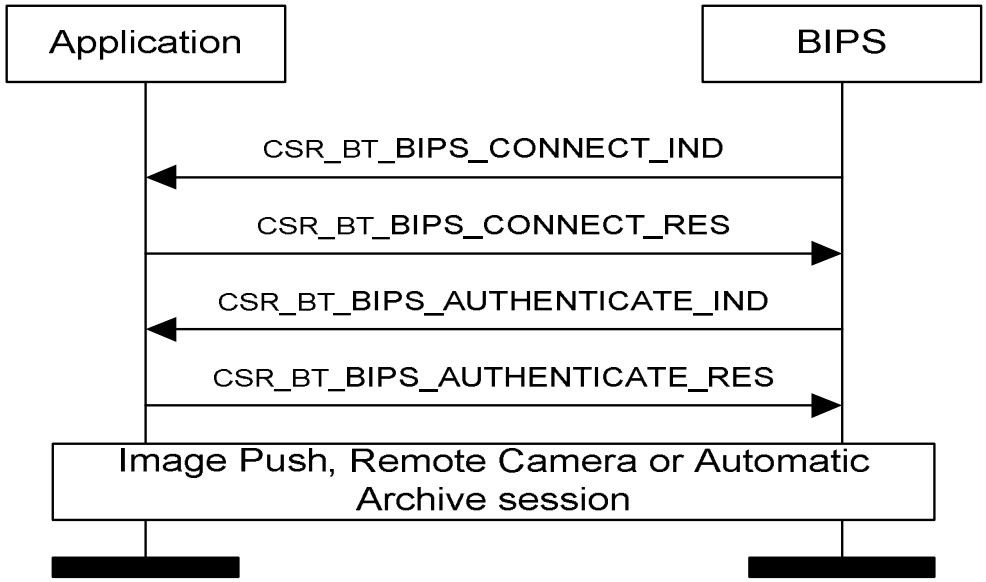


Figure 9: Connection handling with authentication

3.1.6 Abort Handling

The orderly sequence of request (from an OBEX client) followed by response (from an OBEX server) has one exception. An abort operation may come in the middle of a request/response sequence. It cancels the current operation. If the client makes an abort request in order to terminate a multi-packet operation before it normally ends, the BIPS will receive a CSR_BT_BIPS_ABORT_IND message.

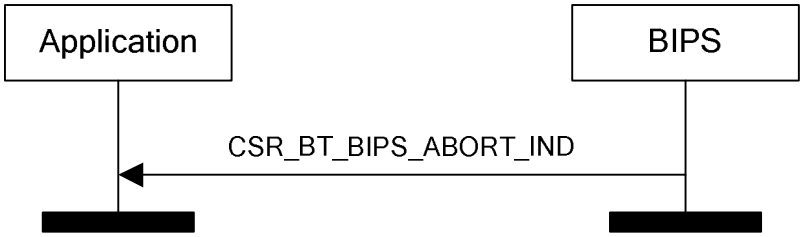


Figure 10: Abort handling

3.1.7 Disconnect

The CSR_BT_BIPS_DISCONNECT_IND message signals the termination of an OBEX session. This signal can come at any time.

Figure 11: Normal disconnect handling for Image Push and Remote Camera, abnormal for Automatic archive

CSR Synergy Bluetooth 18.2.0 OBEX Basic Imaging Profile Server API

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

3.1.8 Disconnect with Automatic Archive

Due to the role reversal during automatic archive the application is expected to disconnect the image session. For this the CSR_BT_BIPS_DISCONNECT_REQ signal is provided.

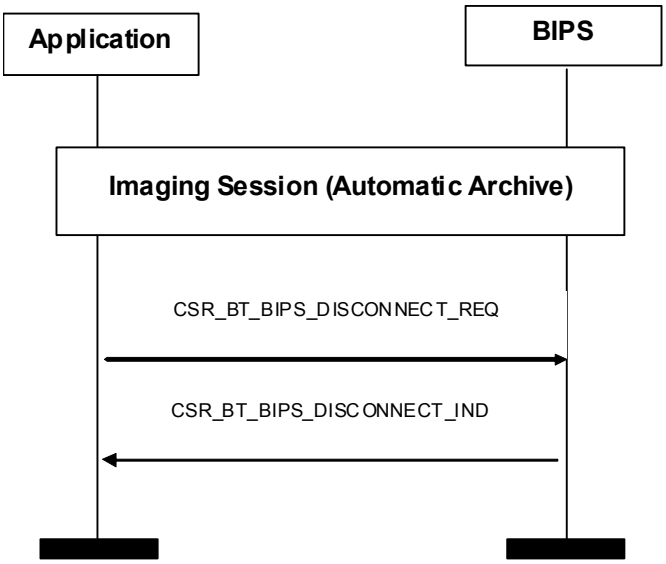


Figure 12: Normal disconnect for Automatic Archive

The system MSC below shows how the secondary (blue) OBEX connection is being disconnected on the initiative of the BIPS profile upon the receipt of the disconnect request from the application. The primary OBEX connection (green) is then expected to be taken down on initiative from the BIP client peer.

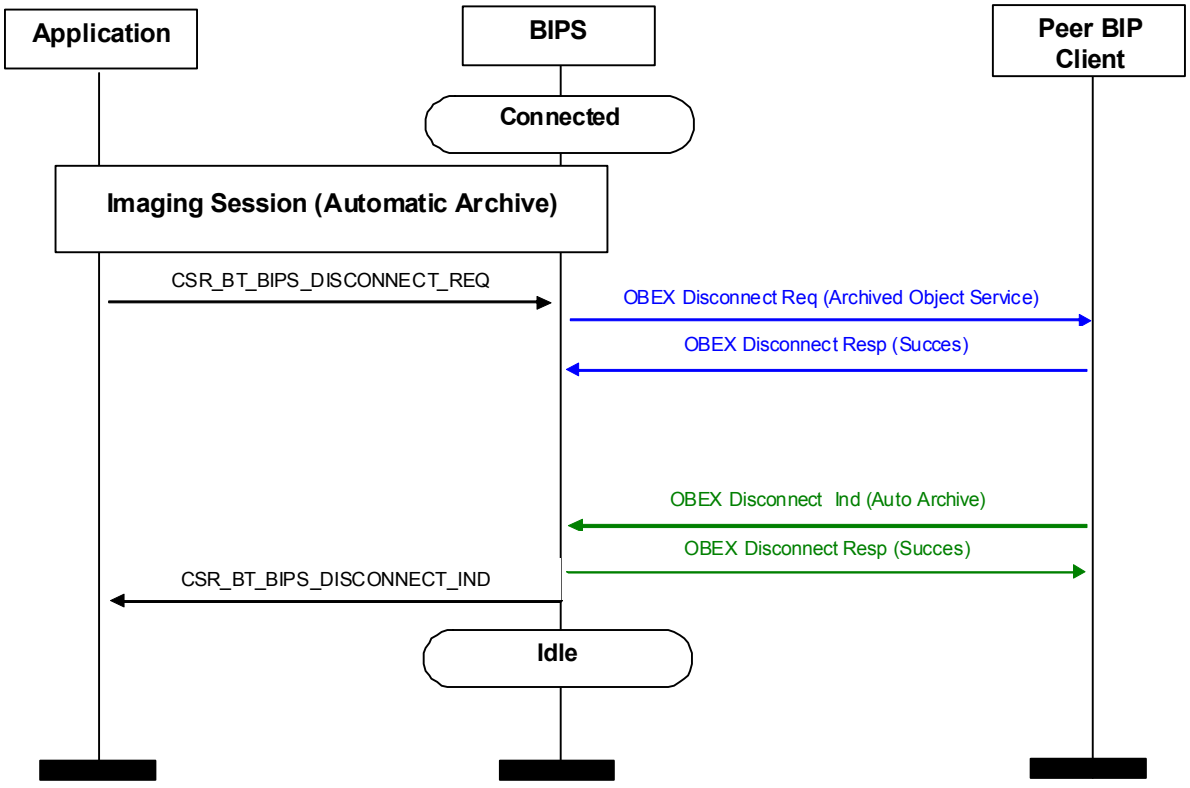


Figure 13: System MSC of the normal disconnect scenario showing OBEX disconnections

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

In case of an abnormal disconnect of the secondary (blue) OBEX connection, the application is informed immediately and BIPS takes responsibility for waiting for the primary OBEX (green) connection to be disconnected.

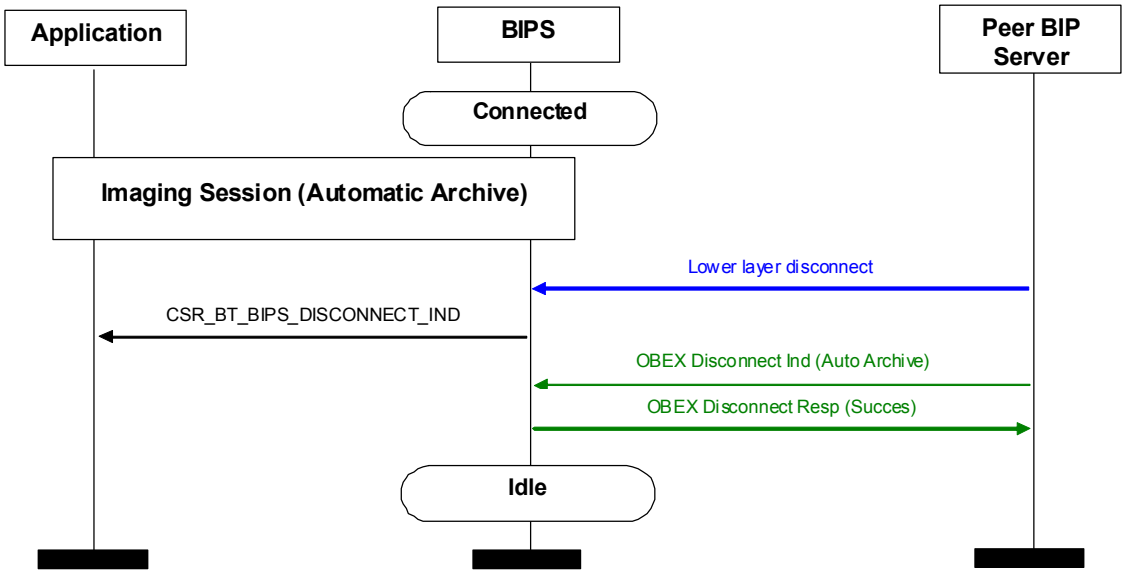


Figure 14: System MSC of the abnormal disconnect scenario showing OBEX disconnections

3.1.9 Deactivation

Sending a CSR_BT_BIPS_DEACTIVATION_REQ deactivates the BIPS. This procedure may take some time depending on the current BIPS activity. When deactivated, the BIPS confirms with a CSR_BT_BIPS_DEACTIVATE_IND message.

Any transaction in progress will be terminated immediately when this message is received by the BIPS.

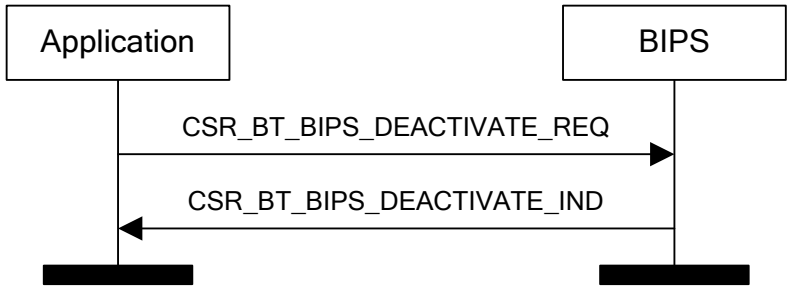


Figure 15: Deactivation request handling

Provided that the service discovery server cannot register the service record for BIPS, or provided that CSR Synergy Bluetooth does not allow any more simultaneous RFCOMM connections, the application receives a CSR_BT_BIPS_DEACTIVATE_IND message.

Figure 16: Deactivation indication handling

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

3.1.10 Get Instances Identifier

As mentioned earlier most signals require a queue handle, as BIPS is capable of running multiple instances. The application gets this queue handle from the profile by issuing a CSR_BT_BIPS_GET_INSTANCES_QID_REQ. This is responded with a CSR_BT_BIPS_GET_INSTANCES_QID_CFM containing the queue handles. The application is responsible for remembering this handles and providing it in all signals.

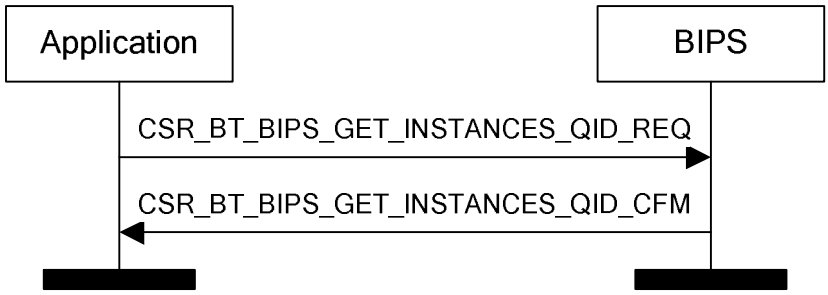


Figure 17: Get Instances QID handling

3.2 Image Push Interfaces

The sequences in this section can only occur when the BIPS is connected in an Image Push session. In the Image Push session images are pushed from the client to the server.

3.2.1 Get Capabilities

When an OBEX connection for Image Push has been made the get capabilities function is used for retrieving the imaging-capabilities object from BIPS.

When the client is making an OBEX get capabilities request against the server, the application receives a CSR_BT_BIPS_PUSH_GET_CAPABILITIES_HEADER_IND message, which the application must respond with CSR_BT_BIPS_PUSH_GET_CAPABILITIES_HEADER_RES with the appropriate result code. If the application accepts the request, it will receive a CSR_BT_BIPS_PUSH_GET_CAPABILITIES_OBJECT_IND message, which it must respond with a CSR_BT_BIPS_PUSH_GET_CAPABILITIES_OBJECT_RES with the appropriate result code and data regarding capabilities. In case the capabilities object is large enough to require several OBEX packets the CSR_BT_BIPS_PUSH_GET_CAPABILITIES_OBJECT_IND / CSR_BT_BIPS_PUSH_GET_CAPABILITIES_OBJECT_RES message sequence is repeated.

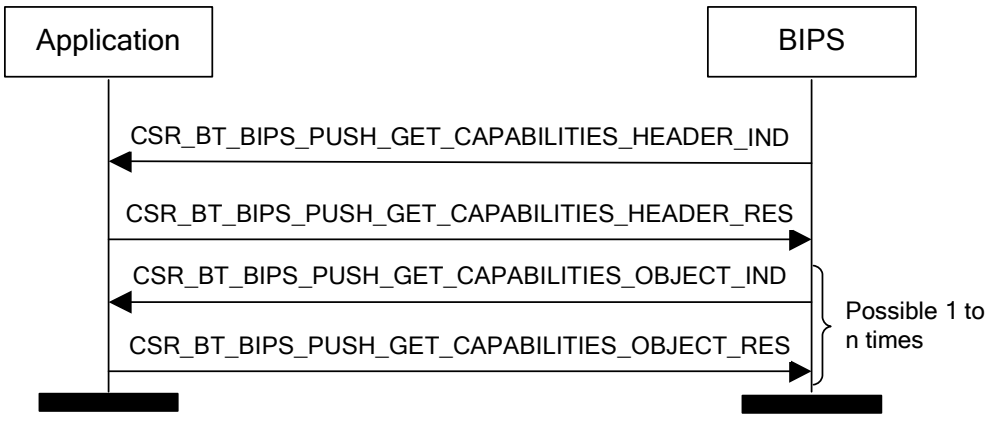


Figure 18: Image Push Get Capabilities handling

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

3.2.2 Put Image

When an OBEX connection for Image Push has been made the put image function is used for pushing an image to BIPS.

When the client is making an OBEX put image request against the server, the application receives a CSR_BT_BIPS_PUSH_PUT_IMAGE_HEADER_IND message, which the application must respond with CSR_BT_BIPS_PUSH_PUT_IMAGE_HEADER_RES with the appropriate result code. If the application accepts the request, it will receive a CSR_BT_BIPS_PUSH_PUT_IMAGE_FILE_IND message, which it must respond with a CSR_BT_BIPS_PUSH_PUT_IMAGE_FILE_RES with the appropriate result code.

In case the image being pushed is large enough to require several OBEX packets the CSR_BT_BIPS_PUSH_PUT_IMAGE_FILE_IND / CSR_BT_BIPS_PUSH_PUT_IMAGE_FILE_RES message sequence is repeated.

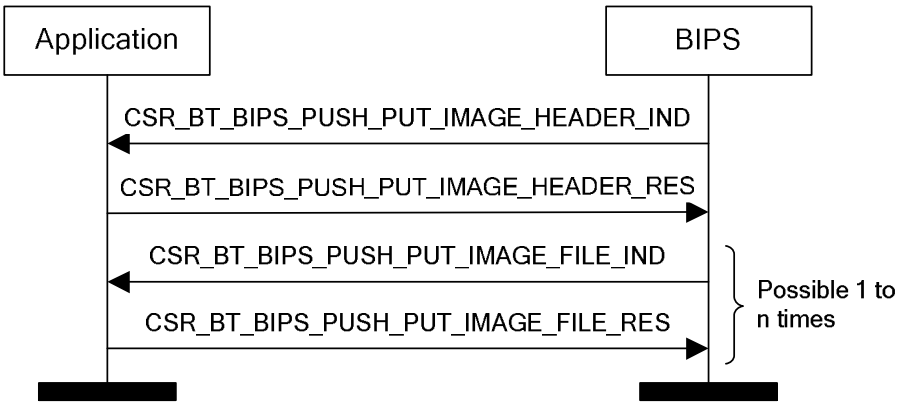


Figure 19: Image Push Put Image handling

BIPS may use the final CSR_BT_BIPS_PUSH_PUT_IMAGE_FILE_RES message to request that the client sends the thumbnail version of the image it just received, by replacing the success response code with the partial content response code. This capability is designed for applications that do not have the ability to convert images into the imaging thumbnail format.

3.2.3 Put Linked Thumbnail

If the response code of the final CSR_BT_BIPS_PUSH_PUT_IMAGE_FILE_RES message is partial content, the client will push the thumbnail version of the image just being sent to the BIPS. Then the application will receive a CSR_BT_BIPS_PUSH_PUT_LINKED_THUMBNAIL_HEADER_IND message, which it must respond with a CSR_BT_BIPS_PUSH_PUT_LINKED_THUMBNAIL_HEADER_RES with the appropriate result code. If the application accepts the request, it will receive a CSR_BT_BIPS_PUSH_PUT_LINKED_THUMBNAIL_FILE_IND message, which it must respond with a CSR_BT_BIPS_PUSH_PUT_LINKED_THUMBNAIL_FILE_RES with the appropriate result code.

In case the thumbnail version of the images is large enough to require several OBEX packets the CSR_BT_BIPS_PUSH_PUT_LINKED_THUMBNAIL_FILE_IND / CSR_BT_BIPS_PUSH_PUT_LINKED_THUMBNAIL_FILE_RES message sequence is repeated.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

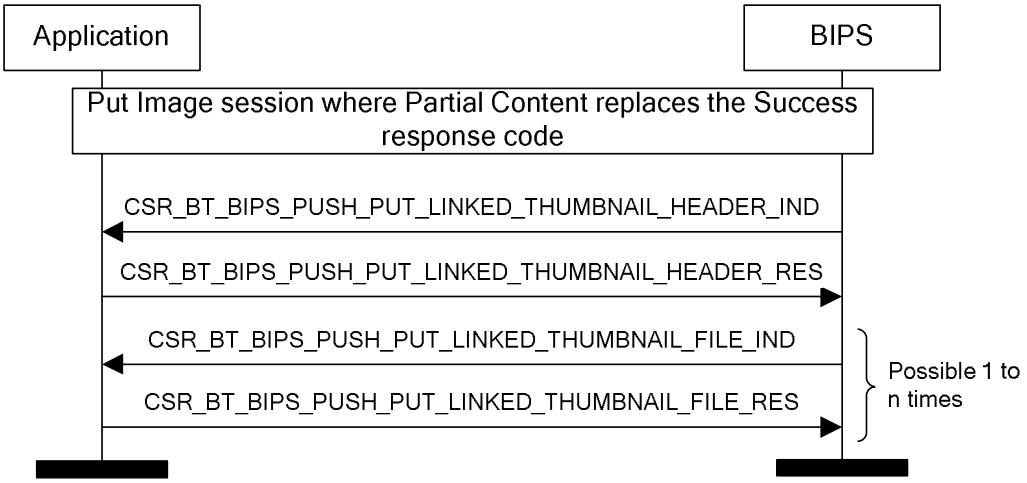


Figure 20: Image Push Put Linked Thumbnail handling

3.2.4 Put Linked Attachment

When an image has been pushed to BIPS, the client might use the Put Linked Attachment function to send the attachments being linked to the image. If the client makes this request the application receives a CSR_BT_BIPS_PUSH_PUT_LINKED_ATTACHMENT_HEADER_IND message, which it must respond with a CSR_BT_BIPS_PUSH_PUT_LINKED_ATTACHMENT_HEADER_RES message, with the appropriate result code. If the application accepts the request, it will receive a CSR_BT_BIPS_PUSH_PUT_LINKED_ATTACHMENT_FILE_IND message, which it must respond with a CSR_BT_BIPS_PUSH_PUT_LINKED_ATTACHMENT_FILE_RES with the appropriate result code.

In case the attachments being pushed are large enough to require several OBEX packets, the CSR_BT_BIPS_PUSH_PUT_LINKED_ATTACHMENT_FILE_IND / CSR_BT_BIPS_PUSH_PUT_LINKED_ATTACHMENT_FILE_RES message sequence is repeated.

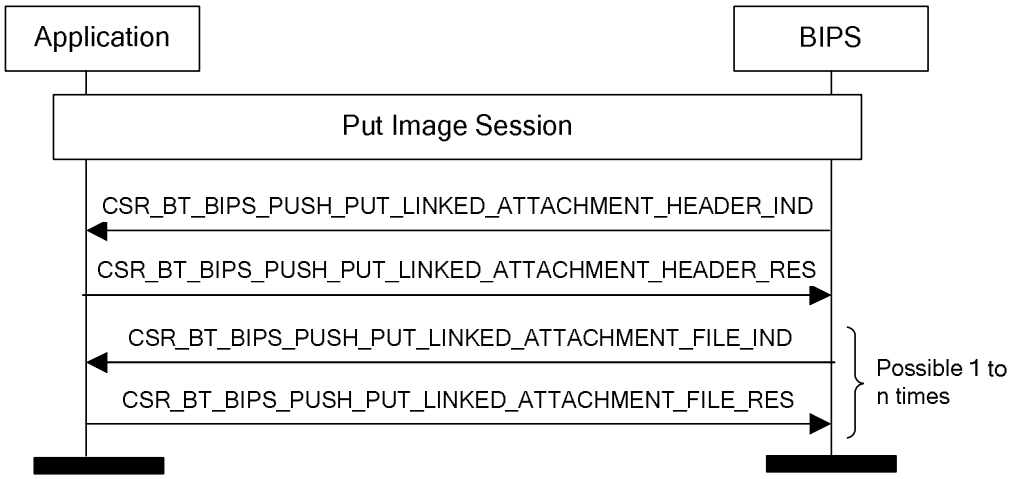


Figure 21: Image Push Put Linked Attachment handling

3.3 Remote Camera Interfaces

The sequences in this section can only occur when the BIPS is connected in a Remote Camera session. In the Remote Camera session the server is most often an image recording device that the client can pull monitoring images and recorded images from.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

3.3.1 Get Monitoring Image

When an OBEX connection for Remote Camera has been made the get monitoring image function is used for retrieving a monitoring image from BIPS.

When the client is making an OBEX get monitoring image request against the server, the application receives a CSR_BT_BIPS_RC_MONITORING_IMAGE_HEADER_IND message, which the application must respond with CSR_BT_BIPS_RC_MONITORING_IMAGE_HEADER_RES containing an image handle if available. Following this the application will receive a CSR_BT_BIPS_RC_MONITORING_IMAGE_OBJECT_IND message requesting the image data itself. This must be responded with a CSR_BT_BIPS_RC_MONITORING_IMAGE_OBJECT_RES message containing the requested data.

In case the image being sent is large enough to require several OBEX packets the CSR_BT_BIPS_RC_MONITORING_IMAGE_OBJECT_IND / BIPS CSR_BT_BIPS_RC_MONITORING_IMAGE_OBJECT_RES message sequence is repeated.

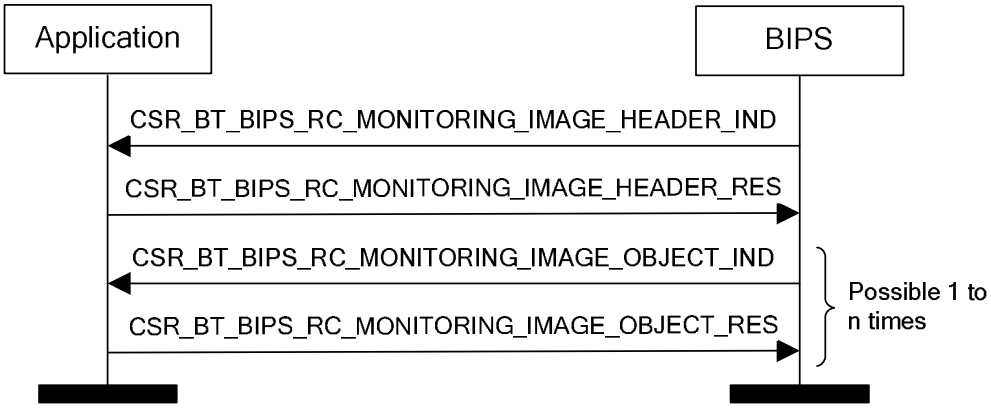


Figure 22: Remote Camera Get Monitoring Image handling

3.3.2 Get Image Properties

When an OBEX connection for Remote Camera has been made the get image properties function can be used for retrieving information about a specific image from BIPS.

When the client is making an OBEX get image properties request against the server, the application receives a CSR_BT_BIPS_RC_GET_IMAGE_PROPERTIES_HEADER_IND message containing an image identifier, which the application must respond with a CSR_BT_BIPS_RC_GET_IMAGE_PROPERTIES_HEADER_RES message containing the appropriate response code. Following this the application will receive a CSR_BT_BIPS_RC_GET_IMAGE_PROPERTIES_OBJECT_IND message requesting the image properties itself. This must be responded with a CSR_BT_BIPS_RC_GET_IMAGE_PROPERTIES_OBJECT_RES message containing the requested data.

In case the image properties being sent is large enough to require several OBEX packets the CSR_BT_BIPS_RC_GET_IMAGE_PROPERTIES_OBJECT_IND / CSR_BT_BIPS_RC_GET_IMAGE_PROPERTIES_OBJECT_RES message sequence is repeated.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

Figure 23: Remote Camera Get Image Properties handling

3.3.3 Get Image

When an OBEX connection for Remote Camera has been made the get image function can be used for retrieving a specific image from BIPS.

When the client is making an OBEX get image request against the server, the application receives a CSR_BT_BIPS_RC_GET_IMAGE_HEADER_IND message containing an image identifier, which the application must respond with a CSR_BT_BIPS_RC_GET_IMAGE_HEADER_RES message containing the appropriate response code. Following this the application will receive a CSR_BT_BIPS_RC_GET_IMAGE_OBJECT_IND message requesting the image itself. This must be responded with a CSR_BT_BIPS_RC_GET_IMAGE_OBJECT_RES message containing the requested data.

In case the image being sent is large enough to require several OBEX packets the CSR_BT_BIPS_RC_GET_IMAGE_OBJECT_IND / CSR_BT_BIPS_RC_GET_IMAGE_OBJECT_RES message sequence is repeated.

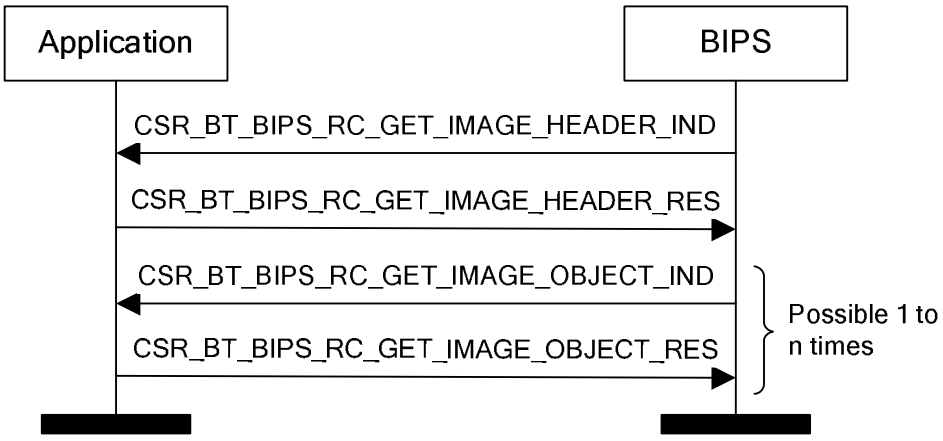


Figure 24: Remote Camera Get Image handling

3.3.4 Get Linked Thumbnail

When an OBEX connection for Remote Camera has been made the get linked thumbnail function can be used for retrieving the thumbnail of a specific image from BIPS.

When the client is making an OBEX get linked thumbnail request against the server, the application receives a CSR_BT_BIPS_RC_GET_LINKED_THUMBNAIL_HEADER_IND message containing an image identifier, which the application must respond with a CSR_BT_BIPS_RC_GET_LINKED_THUMBNAIL_HEADER_RES message containing the appropriate response code. Following this the application will receive a CSR_BT_BIPS_RC_GET_LINKED_THUMBNAIL_OBJECT_IND message requesting the thumbnail itself. This

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

must be responded with a CSR_BT_BIPS_RC_GET_LINKED_THUMBNAIL_OBJECT_RES message containing the requested data.

In case the thumbnail being sent is large enough to require several OBEX packets the CSR_BT_BIPS_RC_GET_LINKED_THUMBNAIL_OBJECT_IND / CSR_BT_BIPS_RC_GET_LINKED_THUMBNAIL_OBJECT_RES message sequence is repeated.

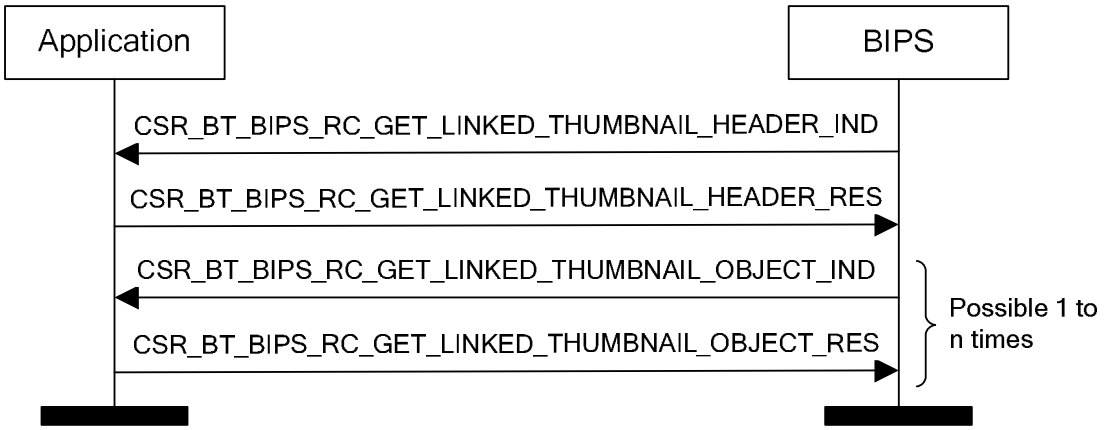


Figure 25: Remote Camera Get Linked Thumbnail handling

3.4 Automatic Archive Interfaces

The sequences in this section can only occur when the BIPS is connected in an Automatic Archive session. In the Automatic Archive session the client asks the server to perform Automatic Archive and the server then retrieves and stores images from the client. In fact the server will start acting as a client and the client will start acting as a server, until the Automatic Archive session is completed.

3.4.1 Get Image List

When an OBEX connection for Automatic Archive has been made the get image list function can be used for retrieving a list of images available on the client, using BIPS.

When the application is making an OBEX get image list request against the client, the application sends a CSR_BT_BIPS_AA_GET_IMAGE_LIST_REQ message. This request can specify criteria that the image list must obey, e.g. that it must only contain images of the JPEG format. As a result the application will receive a CSR_BT_BIPS_AA_GET_IMAGE_LIST_HEADER_IND message containing the criteria that apply to the image list. This message must be responded with a CSR_BT_BIPS_AA_GET_IMAGE_LIST_HEADER_RES message. Following this the application will receive a CSR_BT_BIPS_AA_GET_IMAGE_LIST_IND message with image list data, which must be responded with a CSR_BT_BIPS_AA_GET_IMAGE_LIST_RES message. When the whole image list is received the application will receive a CSR_BT_BIPS_AA_GET_IMAGE_LIST_CFM message.

In case the image list being sent is large enough to require several OBEX packets the CSR_BT_BIPS_AA_GET_IMAGE_LIST_IND / CSR_BT_BIPS_AA_GET_IMAGE_LIST_RES message sequence is repeated.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

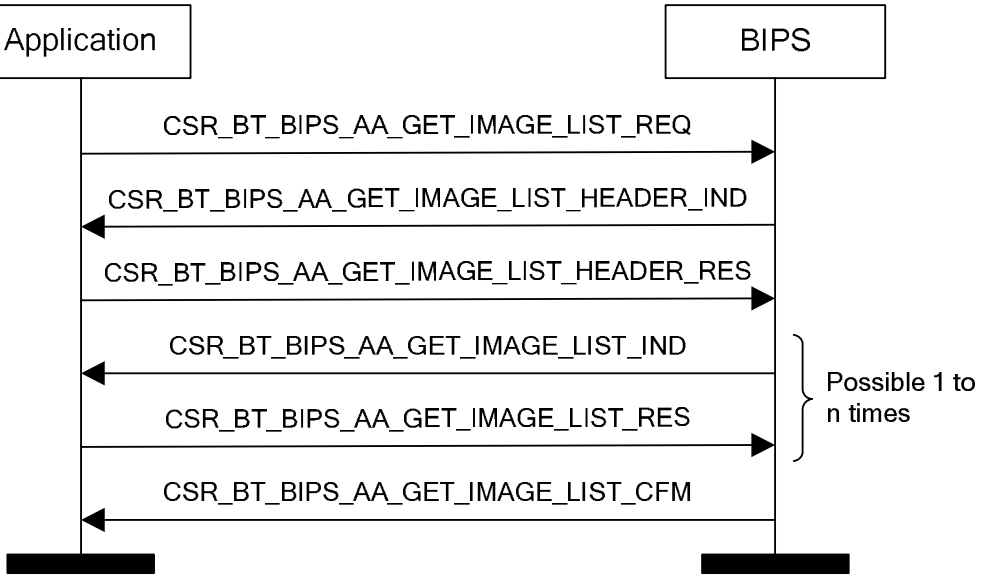


Figure 26: Automatic Archive Get Image List handling

3.4.2 Get Capabilities

When an OBEX connection for Automatic Archive has been made the get capabilities function can be used for retrieving capabilities of the client, using BIPS.

When the application is making an OBEX get capabilities request against the client the application sends a CSR_BT_BIPS_AA_GET_CAPABILITIES_REQ message. The application will then receive a CSR_BT_BIPS_AA_GET_CAPABILITIES_IND message containing the capabilities data, which must be responded with a CSR_BT_BIPS_AA_GET_CAPABILITIES_RES message. When all the capabilities have been received a CSR_BT_BIPS_AA_GET_CAPABILITIES_CFM message is received.

In case the capabilities object being sent is large enough to require several OBEX packets the CSR_BT_BIPS_AA_GET_CAPABILITIES_IND / CSR_BT_BIPS_AA_GET_CAPABILITIES_RES message sequence is repeated.

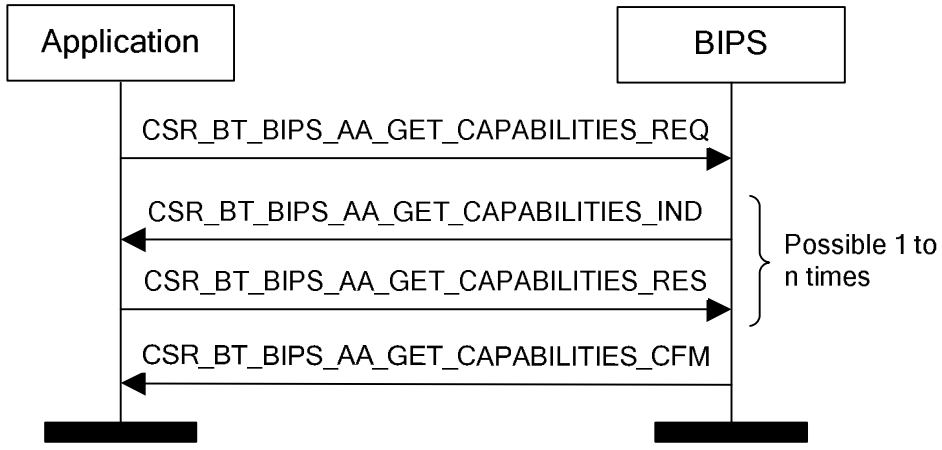


Figure 27: Automatic Archive Get Capabilities handling

3.4.3 Get Image Properties

When an OBEX connection for Automatic Archive has been made the get image properties function can be used for retrieving image properties on a specific image from the client, using BIPS.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

When the application is making an OBEX get image properties request against the client the application sends a CSR_BT_BIPS_AA_GET_IMAGE_PROPERTIES_REQ message. The application will then receive a CSR_BT_BIPS_AA_GET_IMAGE_PROPERTIES_IND message containing the properties data, which must be responded with a CSR_BT_BIPS_AA_GET_IMAGE_PROPERTIES_RES message. When all the properties have been received a CSR_BT_BIPS_AA_GET_IMAGE_PROPERTIES_CFM message is received.

In case the properties object being sent is large enough to require several OBEX packets the CSR_BT_BIPS_AA_GET_IMAGE_PROPERTIES_IND / CSR_BT_BIPS_AA_GET_IMAGE_PROPERTIES_RES message sequence is repeated.

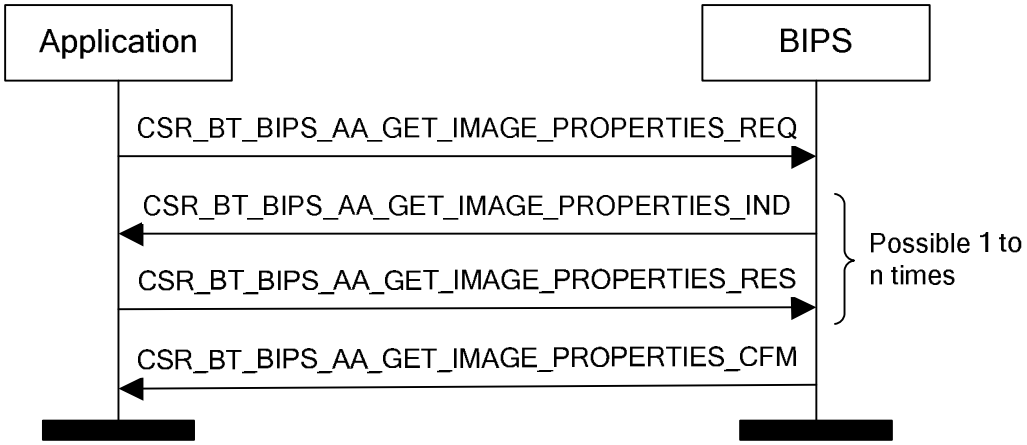


Figure 28: Automatic Archive Get Image Properties handling

3.4.4 Get Image

When an OBEX connection for Automatic Archive has been made the get image function can be used for retrieving an image from the client, using BIPS.

When the application is making an OBEX get image request against the client the application sends a CSR_BT_BIPS_AA_GET_IMAGE_REQ message. The application will then receive a CSR_BT_BIPS_AA_GET_IMAGE_IND message containing the image, which must be responded with a CSR_BT_BIPS_AA_GET_IMAGE_RES message. When the entire image has been received a CSR_BT_BIPS_AA_GET_IMAGE_CFM message is received.

In case the image being sent is large enough to require several OBEX packets the CSR_BT_BIPS_AA_GET_IMAGE_IND / CSR_BT_BIPS_AA_GET_IMAGE_RES message sequence is repeated.

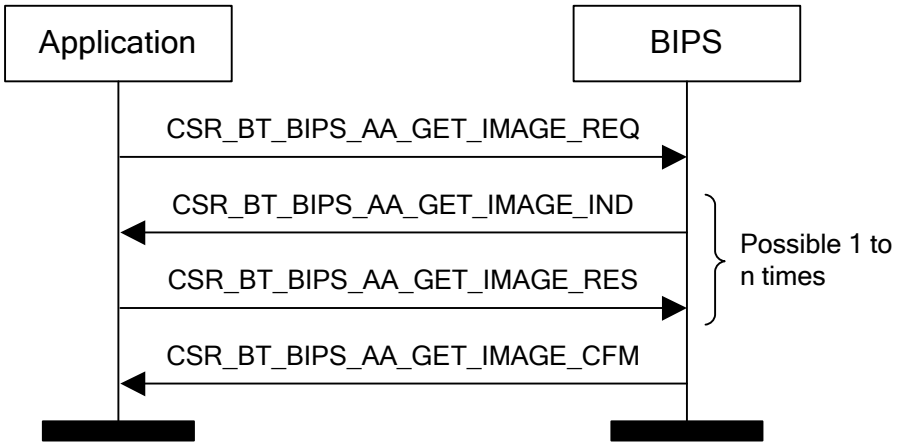


Figure 29: Automatic Archive Get Image handling

CSR Synergy Bluetooth 18.2.0 OBEX Basic Imaging Profile Server API

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

3.4.5 Get Linked Attachment

When an OBEX connection for Automatic Archive has been made the get linked attachment function can be used for retrieving an attachment associated with an image from the client, using BIPS.

When the application is making an OBEX get linked attachment request against the client the application sends a CSR_BT_BIPS_AA_GET_LINKED_ATTACHMENT_REQ message. The application will then receive a CSR_BT_BIPS_AA_GET_LINKED_ATTACHMENT_IND message containing the attachment, which must be responded with a CSR_BT_BIPS_AA_GET_LINKED_ATTACHMENT_RES message. When the entire attachment has been received a CSR_BT_BIPS_AA_GET_LINKED_ATTACHMENT_CFM message is received.

In case the attachment being sent is large enough to require several OBEX packets the CSR_BT_BIPS_AA_GET_LINKED_ATTACHMENT_IND / CSR_BT_BIPS_AA_GET_LINKED_ATTACHMENT_RES message sequence is repeated.

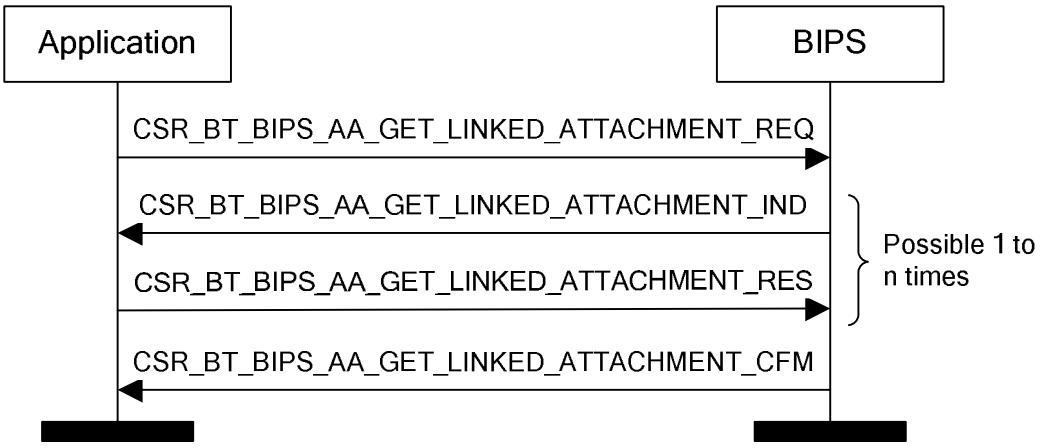


Figure 30: Automatic Archive Get Linked Attachment handling

3.4.6 Get Linked Thumbnail

When an OBEX connection for Automatic Archive has been made the get linked thumbnail function can be used for retrieving a thumbnail of an image from the client, using BIPS.

When the application is making an OBEX get linked thumbnail request against the client the application sends a CSR_BT_BIPS_AA_GET_LINKED_THUMBNAIL_REQ message. The application will then receive a CSR_BT_BIPS_AA_GET_LINKED_THUMBNAIL_IND message containing the thumbnail, which must be responded with a CSR_BT_BIPS_AA_GET_LINKED_THUMBNAIL_RES message. When the entire thumbnail has been received a CSR_BT_BIPS_AA_GET_LINKED_THUMBNAIL_CFM message is received.

In case the thumbnail being sent is large enough to require several OBEX packets the CSR_BT_BIPS_AA_GET_LINKED_THUMBNAIL_IND / CSR_BT_BIPS_AA_GET_LINKED_THUMBNAIL_RES message sequence is repeated.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

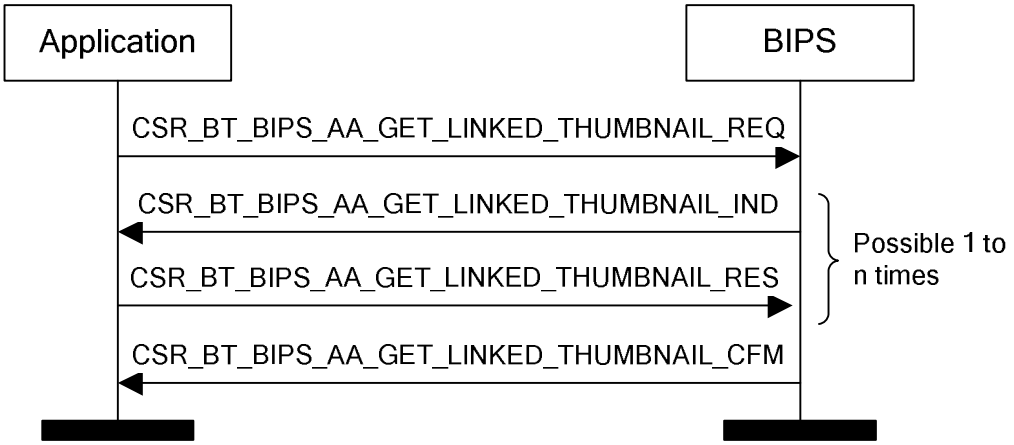


Figure 31: Automatic Archive Get Linked Thumbnail handling

3.4.7 Delete Image

When an OBEX connection for Automatic Archive has been made the delete image function can be used for deleting an image on the client, using BIPS.

When the application is making an OBEX delete image request against the client the application sends a CSR_BT_BIPS_AA_DELETE_IMAGE_REQ message. The application will then receive a CSR_BT_BIPS_AA_DELETE_IMAGE_CFM message containing a response code indicating the outcome of the attempt to delete the image on the client.

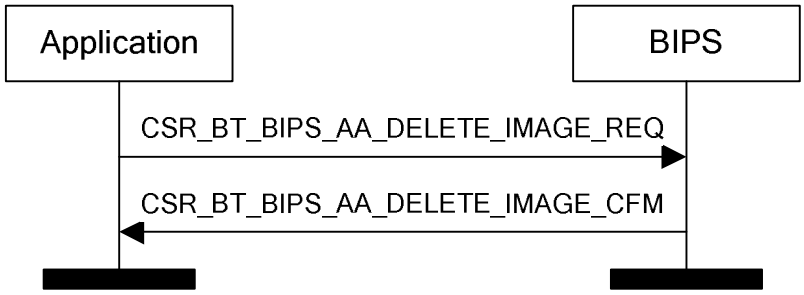
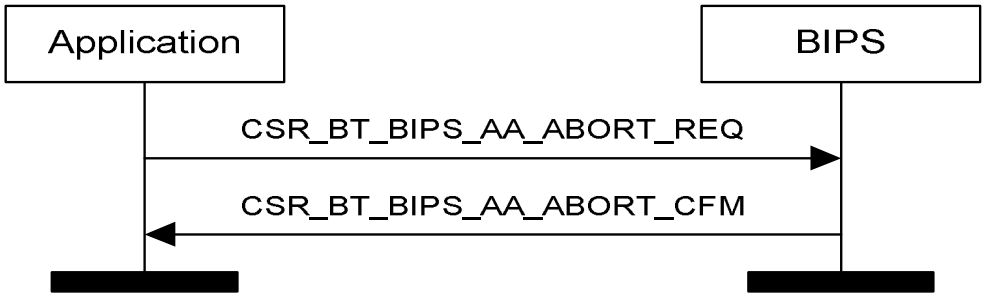


Figure 32: Automatic Archive Delete Image handling

3.4.8 Abort

When an OBEX connection for Automatic Archive has been made and one of the above sequences is ongoing the abort function can be used for aborting the sequence.

If the application wants to abort an ongoing operation the CSR_BT_BIPS_AA_ABORT_REQ message can be issued. This is responded with a CSR_BT_BIPS_AA_ABORT_CFM message when the operation is terminated. When the confirm is received the profile is ready to start a operation but is still OBEX connected for Automatic Archive.



© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

Figure 33: Automatic Archive Abort handling

3.5 Payload Encapsulated Data

3.5.1 Using Offsets

As many OBEX messages contain multiple parameters with variable length, some of the parameters are based on *offsets* instead of standard pointers to the data. Signals with offset-based data can easily be recognized as they have both a *payload* and a *payloadLength* parameter. The *payload* contains the actual data, on which the offset is based. For example, a typical signal may contain the following:

```
CsrBtCommonPrim    type;
CsrUint8       result;
CsrUint16  ucs2nameOffset;
CsrUint16  bodyOffset;
CsrUint16  bodyLength;
CsrUint16  payloadLength;
CsrUint8       *payload;
```

In this example, two offset parameters can be found, namely *ucs2nameOffset* and *bodyOffset*. To obtain the actual data, the offset value is added to the *payload* pointer, which yields a pointer to the data, i.e.:

```
CsrUint8  *ucs2name;
ucs2name = (CsrUint8*)(primitive->payload + primitive->ucs2nameOffset);
```

As can be seen, the offset contains the number of bytes within the *payload* where the information begins. Similarly, the body data can be retrieved using the following:

```
CsrUint8 *body;
body = (CsrUint8*)(primitive->payload + primitive->bodyOffset);
```

And to illustrate the usage of the *length* parameter, which is also a common parameter, to copy the body one would typically use:

```
CsrMemCpy( copyOfBody, body, primitive->bodyLength );
```

Offset parameters will always have an “Offset” suffix on the name, and offsets are *always* relative to the “payload” parameter.

If the `bodyOffset` or the `bodyLength` is 0 (zero), this means that the signal does not contain any body. The same holds when the `payloadLength` is 0 (zero), which means that there is not payload.

3.5.2 Payload Memory

When the application receives a signal which has a *payload* parameter, the application must always free the payload pointer to avoid memory leaks, for example

```
CsrPfree(primitive->payload);
CsrPfree(primitive);
```

will free both the payload data and the message itself. Note that when the payload has been freed, offsets can not be used anymore, as the actual data is contained within the payload.

Signals that do not use the *payload* parameter must still have each of their pointer-based parameters freed.

Likewise, the profile will free any pointers received as parameters in API signals or functions

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR’s non-disclosure agreement.

4 OBEX Basic Imaging Profile Responder Primitives

This section gives an overview of the primitives and parameters in the interface. Detailed information can be found in the corresponding csr_bt_bips_prim.h file and in some cases bip_common_prim.h. Like the previous section this section is divided into subsections according to if the primitives are common, Image Push related, Remote Camera related or Automatic Archive related.

4.1 List of All Primitives

Primitives:	Reference:
CSR_BT_BIPS_ACTIVATE_REQ	See Section 4.2.1
CSR_BT_BIPS_DEACTIVATE_REQ	See Section 4.2.1
CSR_BT_BIPS_DEACTIVATE_IND	See Section 4.2.1
CSR_BT_BIPS_CONNECT_IND	See Section 4.2.3
CSR_BT_BIPS_CONNECT_RES	See Section 4.2.3
CSR_BT_BIPS_AUTHENTICATE_IND	See Section 4.2.4
CSR_BT_BIPS_AUTHENTICATE_RES	See Section 4.2.4
CSR_BT_BIPS_ABORT_IND	See Section 4.2.5
CSR_BT_BIPS_DISCONNECT_IND	See Section 4.2.6
CSR_BT_BIPS_GET_INSTANCES_QID_REQ	See Section 4.2.7
CSR_BT_BIPS_GET_INSTANCES_QID_CFM	See Section 4.2.7
CSR_BT_BIPS_SECURITY_IN_REQ	See Section 4.2.8
CSR_BT_BIPS_SECURITY_IN_CFM	See Section 4.2.8
CSR_BT_BIPS_PUSH_GET_CAPABILITIES_HEADER_IND	See Section 4.3.1
CSR_BT_BIPS_PUSH_GET_CAPABILITIES_HEADER_RES	See Section 4.3.1
CSR_BT_BIPS_PUSH_GET_CAPABILITIES_OBJECT_IND	See Section 4.3.2
CSR_BT_BIPS_PUSH_GET_CAPABILITIES_OBJECT_RES	See Section 4.3.2
CSR_BT_BIPS_PUSH_PUT_IMAGE_HEADER_IND	See Section 4.3.3
CSR_BT_BIPS_PUSH_PUT_IMAGE_HEADER_RES	See Section 4.3.3
CSR_BT_BIPS_PUSH_PUT_IMAGE_FILE_IND	See Section 4.3.4
CSR_BT_BIPS_PUSH_PUT_IMAGE_FILE_RES	See Section 4.3.4
CSR_BT_BIPS_PUSH_PUT_LINKED_THUMBNAIL_HEADER_IND	See Section 4.3.5
CSR_BT_BIPS_PUSH_PUT_LINKED_THUMBNAIL_HEADER_RES	See Section 4.3.5
CSR_BT_BIPS_PUSH_PUT_LINKED_THUMBNAIL_FILE_IND	See Section 4.3.6
CSR_BT_BIPS_PUSH_PUT_LINKED_THUMBNAIL_FILE_RES	See Section 4.3.6
CSR_BT_BIPS_PUSH_PUT_LINKED_ATTACHMENT_HEADER_IND	See Section 4.3.7
CSR_BT_BIPS_PUSH_PUT_LINKED_ATTACHMENT_HEADER_RES	See Section 4.3.7
CSR_BT_BIPS_PUSH_PUT_LINKED_ATTACHMENT_FILE_IND	See Section 4.3.8
CSR_BT_BIPS_PUSH_PUT_LINKED_ATTACHMENT_FILE_RES	See Section 4.3.8
CSR_BT_BIPS_RC_MONITORING_IMAGE_HEADER_IND	See Section 4.4.1
CSR_BT_BIPS_RC_MONITORING_IMAGE_HEADER_RES	See Section 4.4.1
CSR_BT_BIPS_RC_MONITORING_IMAGE_OBJECT_IND	See Section 4.4.2
CSR_BT_BIPS_RC_MONITORING_IMAGE_OBJECT_RES	See Section 4.4.2
CSR_BT_BIPS_RC_GET_IMAGE_PROPERTIES_HEADER_IND	See Section 4.4.3
CSR_BT_BIPS_RC_GET_IMAGE_PROPERTIES_HEADER_RES	See Section 4.4.3
CSR_BT_BIPS_RC_GET_IMAGE_PROPERTIES_OBJECT_IND	See Section 4.4.4
CSR_BT_BIPS_RC_GET_IMAGE_PROPERTIES_OBJECT_RES	See Section 4.4.4

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

Primitives:	Reference:
CSR_BT_BIPS_RC_GET_IMAGE_HEADER_IND	See Section 4.4.5
CSR_BT_BIPS_RC_GET_IMAGE_HEADER_RES	See Section 4.4.5
CSR_BT_BIPS_RC_GET_IMAGE_OBJECT_IND	See Section 4.4.6
CSR_BT_BIPS_RC_GET_IMAGE_OBJECT_RES	See Section 4.4.6
CSR_BT_BIPS_RC_GET_LINKED_THUMBNAIL_HEADER_IND	See Section 4.4.7
CSR_BT_BIPS_RC_GET_LINKED_THUMBNAIL_HEADER_RES	See Section 4.4.7
CSR_BT_BIPS_RC_GET_LINKED_THUMBNAIL_OBJECT_IND	See Section 4.4.8
CSR_BT_BIPS_RC_GET_LINKED_THUMBNAIL_OBJECT_RES	See Section 4.4.8
CSR_BT_BIPS_AA_GET_GET_IMAGE_LIST_REQ	See Section 4.5.1
CSR_BT_BIPS_AA_GET_GET_IMAGE_LIST_HEADER_IND	See Section 4.5.1
CSR_BT_BIPS_AA_GET_GET_IMAGE_LIST_HEADER_RES	See Section 4.5.1
CSR_BT_BIPS_AA_GET_GET_IMAGE_LIST_IND	See Section 4.5.1
CSR_BT_BIPS_AA_GET_GET_IMAGE_LIST_RES	See Section 4.5.1
CSR_BT_BIPS_AA_GET_GET_IMAGE_LIST_CFM	See Section 4.5.1
CSR_BT_BIPS_AA_GET_CAPABILITIES_REQ	See Section 4.5.2
CSR_BT_BIPS_AA_GET_CAPABILITIES_IND	See Section 4.5.2
CSR_BT_BIPS_AA_GET_CAPABILITIES_RES	See Section 4.5.2
CSR_BT_BIPS_AA_GET_CAPABILITIES_CFM	See Section 4.5.2
CSR_BT_BIPS_AA_GET_IMAGE_PROPERTIES_REQ	See Section 4.5.3
CSR_BT_BIPS_AA_GET_IMAGE_PROPERTIES_IND	See Section 4.5.3
CSR_BT_BIPS_AA_GET_IMAGE_PROPERTIES_RES	See Section 4.5.3
CSR_BT_BIPS_AA_GET_IMAGE_PROPERTIES_CFM	See Section 4.5.3
CSR_BT_BIPS_AA_GET_IMAGE_REQ	See Section 4.5.4
CSR_BT_BIPS_AA_GET_IMAGE_IND	See Section 4.5.4
CSR_BT_BIPS_AA_GET_IMAGE_RES	See Section 4.5.4
CSR_BT_BIPS_AA_GET_IMAGE_CFM	See Section 4.5.4
CSR_BT_BIPS_AA_GET_LINKED_ATTACHMENT_REQ	See Section 4.5.5
CSR_BT_BIPS_AA_GET_LINKED_ATTACHMENT_IND	See Section 4.5.5
CSR_BT_BIPS_AA_GET_LINKED_ATTACHMENT_RES	See Section 4.5.5
CSR_BT_BIPS_AA_GET_LINKED_ATTACHMENT_CFM	See Section 4.5.5
CSR_BT_BIPS_AA_GET_LINKED_THUMBNAIL_REQ	See Section 4.5.6
CSR_BT_BIPS_AA_GET_LINKED_THUMBNAIL_IND	See Section 4.5.6
CSR_BT_BIPS_AA_GET_LINEKD_THUMBNAIL_RES	See Section4.5.6
CSR_BT_BIPS_AA_GET_LINKED_THUMBNAIL_CFM	See Section 4.5.6
CSR_BT_BIPS_AA_DELETE_IMAGE_REQ	See Section 4.5.7
CSR_BT_BIPS_AA_DELETE_IMAGE_CFM	See Section 4.5.7
CSR_BT_BIPS_AA_ABORT_REQ	See Section 4.5.8
CSR_BT_BIPS_AA_ABORT_CFM	See Section 4.5.8

Table 1: List of all primitives

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

4.2 Common Primitives

This section contains description of primitives that are common to Image Push, Remote Camera as well as Automatic Archive.

4.2.1 CSR_BT_BIPS_ACTIVATE

Parameters / Primitives	type	appHandle	upperDataCapacity	lowerDataCapacity	featureSelection	obexMaxPacketSize	digestChallenge	windowSize	srmEnable
CSR_BT_BIPS_ACTIVATE_REQ	✓	✓	✓	✓	✓	✓	✓	✓	✓

Table 2: CSR_BT_BIPS_ACTIVATE Primitives

Description

This signal is used for activating the BIPS and makes it accessible from a remote device. The process includes:

1. Register the OBEX Image Push Server, OBEX Remote Camera and/or OBEX Automatic Archive services in the service discovery database with the maximum memory available for image storage
2. Enabling page scan

The BIPS will remain activated until a CSR_BT_BIPS_DEACTIVATE_IND is received.

Parameters

type	Signal identity, CSR_BT_BIPS_ACTIVATE_REQ.
appHandle	The identity of the calling process. It is possible to initiate the procedure by any higher layer process as the response is returned to appHandle.
upperDataCapacity	The value of the maximum memory available for image storage in bytes. The value of this parameter represent bit 32 to bit 63.
lowerDataCapacity	The value of the maximum memory available for image storage in bytes. The value of this parameter represent bit 0 to bit 31.
featureSelection	Holds at least one of the following defines to identify which feature of the profile to activate: CSR_BT_BIPS_IMAGE_PUSH_FEATURE, CSR_BT_BIPS_AUTO_ARCHIVE_FEATURE, CSR_BT_BIPS_REMOTE_CAMERA_FEATURE. If multiple features are to be activated the above defines must be combined using bitwise or, e.g. (CSR_BT_BIPS_IMAGE_PUSH_FEATURE | CSR_BT_BIPS_REMOTE_CAMERA_FEATURE)
obexMaxPacketSize	The value of the maximum size of the OBEX packages to be received. Must be between 255 and 65535.
digestChallenge	Perform authentication challenge of clients when they connect.
windowSize	Controls how many packets the OBEX profile, and lower protocol layers, are allowed to cache on the data receive side. A value of zero (0) will cause the system to auto-detect this value.
srmEnable	TRUE enables local support for Single Response Mode (SRM).

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

If SRM is enabled FTS allows that PUT and GET commands, multiple OBEX request packets (PUT) or OBEX response packet (GET), can be send immediately, without waiting for the remote device.

Please note, SRM can only be enabled if both sides support it. For more information about SRM, please refer to [GOEP2.0].

Library Function

The library functions are provided by csr_bt_bips_lib.h, and should be used for building and sending the CSR_BT_BIPS_ACTIVATE_REQ signal:

```
void CsrBtBipsActivateReqSend(CsrSchedQid pHandleInst,
                              CsrSchedQid appHandle,
                              CsrUint32   upperDataCapacity,
                              CsrUint32   lowerDataCapacity,
                              CsrUint8    featureSelection,
                              CsrUint16   obexMaxPacketSize,
                              CsrBool     digestChallenge,
                              CsrUint16   windowSize,
                              CsrBool     srmEnable);
```

Note that if a device initiates OBEX authentication, interoperability cannot be guaranteed with devices that lack a user interface. Therefore it is recommended that OBEX authentication be turned off.

The `pHandleInst` parameter is used for indicating the destination BIPS instance of the signal. The remaining parameters are as described in the table above.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

4.2.2 CSR_BT_BIPS_DEACTIVATE

Primitives \ Parameters	type	pHandleInst
CSR_BT_BIPS_DEACTIVATE_REQ	✓	
CSR_BT_BIPS_DEACTIVATE_IND	✓	✓

Table 3: CSR_BT_BIPS_DEACTIVATE Primitives

Description

This signal deactivates the BIPS. The service cannot be re-activated until after the application has received a CSR_BT_BIPS_DEACTIVATE_IND.

The service will no longer be visible to inquire devices and the inquiry and page scan may be stopped (depending on the fact if other services are available or not). The OBEX Image Push, OBEX Remote Camera and OBEX Automatic Archive server services are removed from the service discovery database.

The signal will stop any ongoing transaction.

Parameters

type	Signal identity, CSR_BT_BIPS_DEACTIVATE_REQ/IND.
pHandleInst	This value is used for identifying which BIPS instance the signal is coming from.

Library Function

A library function is provided by csr_bt_bips_lib.h, and should be used for building and sending the CSR_BT_BIPS_DEACTIVATE_REQ signal:

```
void CsrBtBipsDeactivateReqSend(CsrSchedQid pHandleInst);
```

The parameter is as described in the table above.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

4.2.3 CSR_BT_BIPS_CONNECT

Primitives \ Parameters	type	pHandleInst	connectionId	deviceAddr	connectType	responseCode	supportedFunctions	length	count	btConnId
CSR_BT_BIPS_CONNECT_IND	✓	✓	✓	✓	✓			✓	✓	✓
CSR_BT_BIPS_CONNECT_RES	✓		✓			✓	✓			

Table 4: CSR_BT_BIPS_CONNECT Primitives

Description

This signal is indicating that the BIP client is starting a session. The type of session, Image Push, Remote Camera or Automatic Archive, is indicated in the parameters. The application can accept or deny the request and has to return the connectionId received in the indication.

Please note that the maximum size that BIPS can receive from the peer device is always set to CSR_BT_MAX_OBEX_SIGNAL_LENGTH is defined in csr_bt_usr_config.h.

Parameters

type — Signal identity, CSR_BT_BIPS_CONNECT_IND/RES.

pHandleInst — This value is used for identifying which BIPS instance the signal is coming from.

connectionId — The connection Id tells the recipient of the request which OBEX connection this request belongs to. The connection Id received in the indication must be returned in the response.

deviceAddr — The Bluetooth address of the device that is connecting

connectType — Indicates the type of session, where the following sessions are defined:

CSR_BT_BIPS_PUSH_CONNECT (Image push)
CSR_BT_BIPS_RC_CONNECT (Remote Camera)
CSR_BT_BIPS_AA_CONNECT (Automatic Archive)

These values are defined in csr_bt_bips_prim.h

responseCode — For accepting an OBEX connection the code is:
CSR_BT_OBEX_SUCCESS_RESPONSE_CODE.

The following response codes reject the OBEX connection request:
CSR_BT_OBEX_SERVICE_UNAVAILABLE_RESPONSE_CODE
CSR_BT_OBEX_BAD_REQUEST_RESPONSE_CODE
CSR_BT_OBEX_FORBIDDEN_RESPONSE_CODE

The responseCodes are defined in (csr_bt_obex.h) with the following type CsrBtObexResponseCode and can also be found in IrDA Object Exchange Protocol. The meaning of the response codes for the Basic Imaging Profile is described in [BIP].

supportedFunctions — An Imaging functions flags that Indicates the function the Imaging (secondary) Server supports. For a detailed description of the Imaging functions flags, please refer to

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

	[BIP]. Please note that this parameter is only valid if connectType equals CSR_BT_BIPS_AA_CONNECT.
length	The length parameter contains the length in bytes of the bodies of all the objects that the sender plans to send. Note this length cannot be guarantee correct, so while the value may be useful for status indicators and resource reservations, the application should not die if the length is not correct. If 0 this parameter were not included in the received OBEX Connect Request packet.
count	Count is use to indicate the number of objects that will be sent by the sender during this connection. If 0 this parameter were not included in the received OBEX Connect Request packet.
btConnId	Identifier which shall be used when using AMPM, for more information please refer to [AMPM].

Library Function

A library function is provided by csr_bt_bips_lib.h, and should be used for building and sending the CSR_BT_BIPS_CONNECT_RES signal:

```
void CsrBtBipsConnectResSend(CsrSchedQid           pHandleInst,
                             CsrUint32             connectionId,
                             CsrBtObexResponseCode responseCode);
```

The parameters are as described in the table above.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

4.2.4 CSR_BT_BIPS_AUTHENTICATE

Primitives \ Parameters	type	pHandleInst	options	realmLength	* realm	deviceAddr	passwordLength	*password	*userId
CSR_BT_BIPS_AUTHENTICATE_IND	✓	✓	✓	✓	✓	✓			
CSR_BT_BIPS_AUTHENTICATE_RES	✓						✓	✓	✓

Table 5: CSR_BT_BIPS_AUTHENTICATE Primitives

Description

The indication and response signal is used when the BIP client wants to OBEX authenticate the BIP server. The application has to response with a password or pin number in the password and a userId for client to identify the proper password.

Parameters

type — Signal identity, CSR_BT_BIPS_AUTHENTICATE_IND/RES.

pHandleInst — This value is used for identifying which BIPS instance the signal is coming from.

options — Challenge information of type CsrUint8.

Bit 0 controls the responding of a valid user Id.

If bit 0 is set it means that the application must response with a user Id in a CSR_BT_BIPS_AUTHENTICATE_RES message. If bit 0 is not set the application can just set the userId to NULL.

Bit 1 indicates the access mode being offered by the sender

If bit 1 is set the access mode is read only. If bit 1 is not set the sender gives full access, e.g. both read and write.

Bit 2 - 7 is reserved.

realmLength — Number of bytes in realm of type CsrUint16

Note in this release version the 'realmLength' parameter is always set to 0x0000

* realm — A displayable string indicating for the user which userid and/or password to use. The first byte of the string is the character set of the string. The table below shows the different values for character set.

Note that this pointer must be CsrPfree by the application, and that this pointer can be NULL because the realm field is optional to set by the peer device.

Note in this release version the 'realm' pointer is always set to NULL

Char set Code	Meaning
0	ASCII
1	ISO-8859-1

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

2	ISO-8859-2
3	ISO-8859-3
4	ISO-8859-4
5	ISO-8859-5
6	ISO-8859-6
7	ISO-8859-7
8	ISO-8859-8
9	ISO-8859-9
0xFF = 255	UNICODE

deviceAddr	The Bluetooth address of the device that has initiated the OBEX authentication procedure
passwordLength	The length of the response password.
*password	Containing the response password of the OBEX authentication. This is a pointer which shall be allocated by the application.
*userId	Pointer to a zero terminated string (ASCII) containing the userId for the authentication. This is a pointer which shall be allocated by the application.

Library Function

A library function is provided by csr_bt_bips_lib.h, and should be used for building and sending the CSR_BT_BIPS_AUTHENTICATE_RES signal:

```
void CsrBtBipsAuthenticateResSend(CsrSchedQid pHandleInst, CsrUint8 *password,
CsrUint16 passwordLength, CsrCharString *userId);
```

The parameters are as described in the table above.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

4.2.5 CSR_BT_BIPS_ABORT

Primitives \ Parameters	type	pHandleInst	connectionId
CSR_BT_BIPS_ABORT_IND	✓	✓	✓

Table 6: CSR_BT_BIPS_ABORT Primitives

Description

This signal is used when a client decides to terminate a multi-packet operation before it normally ends.

Please notice that the orderly sequence of request (from a client) followed by a response (from a server) has one exception. The ABORT operation may come in the middle of a request/response sequence.

Parameters

type	Signal identity, CSR_BT_BIPS_ABORT_IND.
pHandleInst	This value is used for identifying which BIPS instance the signal is coming from.
connectionId	The connection Id for this session, the BIP client must use this Id in the request.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

4.2.6 CSR_BT_BIPS_DISCONNECT

Primitives \ Parameters	type	pHandleInst	reasonCode	reasonSupplier	connectionId	normalDisconnect
CSR_BT_BIPS_DISCONNECT_IND	✓	✓	✓	✓	✓	
CSR_BT_BIPS_DISCONNECT_REQ	✓					✓

Table 7: CSR_BT_BIPS_DISCONNECT Primitives

Description

This signal is indicating that the OBEX image push session is finished, and is ready for a new one.

Parameters

type	Signal identity, CSR_BT_BIPS_DISCONNECT_IND.
reasonCode	The reason code of the operation. Possible values depend on the value of reasonSupplier. If e.g. the reasonSupplier == CSR_BT_SUPPLIER_CM then the possible reason codes can be found in csr_bt_cm_prim.h. If the reasonSupplier == CSR_BT_SUPPLIER_OBEX then the possible result codes can be found in csr_bt_obex.h. All values which are currently not specified in the respective prim.h files or csr_bt_obex.h are regarded as reserved and the application should consider them as errors.
reasonSupplier	This parameter specifies the supplier of the reason given in reasonCode. Possible values can be found in csr_bt_result.h
pHandleInst	This value is used for identifying which BIPS instance the signal is coming from.
connectionId	The connection Id for this session, the BIP client must use this Id in the request.
normalDisconnect	Boolean indication whether the disconnect is normal or abnormal.

Library Function

A library function is provided by csr_bt_bips_lib.h, and should be used for building and sending the CSR_BT_BIPS_DISCONNECT_REQ signal:

```
void CsrBtBipsDisconnectReqSend(CsrSchedQid pHandleInst, CsrBool normalDisconnect);
```

The parameters are as described in the table above.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

4.2.7 CSR_BT_BIPS_GET_INSTANCES_QID

Primitives \ Parameters	type	qId	phandlesListSize	phandlesList
CSR_BT_BIPS_GET_INSTANCE_QID_REQ	✓	✓		
CSR_BT_BIPS_GET_INSTANCE_QID_CFM	✓		✓	✓

Table 8: CSR_BT_BIPS_REGISTER_QID Primitives

Description

This request and confirm is used by the application to learn its own instance ID. This ID must be used in all signals as BIPS is capable of running multiple instances and it is thus necessary to know which instance the signal must be directed to.

Parameters

type	Signal identity, CSR_BT_BIPS_DISCONNECT_IND.
CsrSchedQid	This value is used for identifying which BIPS instance the signal is to be sent to. Must always be CSR_BT_BIPS_IFACEQUEUE.
phandlesListSize	Number of phandles that is returned in the CSR_BT_BIPS_GET_INSTANCES_QID_CFM.
phandlesList	Pointer to array of phandles for registered BIPS instances. These values must be used for distinguishing the registered BIPS instances when sending signals to, or receiving signals from them. They are carried in the pHandleInst or qId parameters of the signals and library functions.

Library Function

A library function is provided by csr_bt_bips_lib.h, and should be used for building and sending the CSR_BT_BIPS_GET_INSTANCE_QID_REQ signal:

```
void CsrBtBipsGetInstancesQidReqSend(CsrSchedQid appHandle);
```

The `appHandle` parameter is used for indicating the destination BIPS instance of the signal.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

4.2.8 CSR_BT_BIPS_SECURITY_IN

Primitives \ Parameters	type	pHandleInst	appHandle	secLevel	resultCode	resultSupplier
CSR_BT_BIPS_SECURITY_IN_REQ	✓	✓	✓	✓		
CSR_BT_BIPS_SECURITY_IN_CFM	✓	✓			✓	✓

Table 9: CSR_BT_BIPS_SECURITY_IN Primitives

Description

Applications that wish to change the enforcement to a specific profile security level, i.e. authentication, encryption and/or authorisation, can use this API to set up the security level for *new* connections. Note that this API is for the local device only and can be used from within any state.

The *CSR_BT_SECURITY_IN_REQ* signal sets up the security level for new incoming connections. Already established or pending connections are not altered.

Note, that any attempts to set security to a less secure level than the mandatory security level will be rejected. See csr_bt_profiles.h for mandatory security settings. The default settings used by CSR Synergy Bluetooth are set to require authentication and encryption.

Note that if MITM protection is requested and the remote device does not have the required IO capabilities, pairing/bonding will fail and connections to the remote device *cannot* be made. See [SC] for further details.

Parameters

type — Signal identity CSR_BT_BIPS_SECURITY_IN_REQ/CFM.

pHandleInst — This value is used for identifying which BIPS instance the signal is coming from.

appHandle — Application handle to which the confirm message is sent.

secLevel — The application must specify one of the following values:

- CSR_BT_SEC_DEFAULT : Use default security settings
- CSR_BT_SEC_MANDATORY : Use mandatory security settings
- CSR_BT_SEC_SPECIFY : Specify new security settings

If CSR_BT_SEC_SPECIFY is set the following values can be OR'ed additionally:

- CSR_BT_SEC_AUTHORISATION: Require authorisation
- CSR_BT_SEC_AUTHENTICATION: Require authentication
- CSR_BT_SEC_ SEC_ENCRYPTION: Require encryption (implies authentication)
- CSR_BT_SEC_MITM: Require MITM protection (implies encryption)

result — Standard CSR Synergy Bluetooth result code sent to the appHandle as a response to the security level request, please see csr_bt_profiles.h.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

Library Function

A library function is provided by csr_bt_bips_lib.h, and should be used for building and sending the CSR_BT_BIPS_GET_INSTANCE_QID_REQ signal:

```
void CsrBtBipsSecurityInReqSend(CsrSchedQid pHandleInst, CsrSchedQid appHandle,
CsrUint16 secLevel);
```

The parameters are as described in the table above.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

4.2.9 CSR_BT_BIPS_CHALLENGE

Primitives \ Parameters	type	pHandleInst	deviceAddr	connectType	realmLength	*realm	passwordLength	*password	*userId
CSR_BT_BIPS_CHALLENGE_IND	✓	✓	✓	✓					
CSR_BT_BIPS_CHALLENGE_RES	✓				✓	✓	✓	✓	✓

Table 10: CSR_BT_BIPS_CHALLENGE Primitives

Description

If the Server was activated by using the function CsrBtBipsActivateExtReqSend the application will received the CSR_BT_BIPS_CHALLENGE_IND before it receives a CSR_BT_BIPS_CONNECT_IND message. This will give the application the possibility of initiates OBEX authentication. The application shall always response a CSR_BT_BIPS_CHALLENGE_IND message by using the CsrBtBipsChallengeResSend. If the 'password' parameter in this function is different from NULL the OBEX authentication is initiated otherwise not. It also means that the application is guarantee that it is OBEX authenticated whenever the CSR_BT_BIPS_CONNECT_IND is receive. Note that the application must consider itself for disconnected until it receives a CSR_BT_BIPS_CONNECT_IND message and response it with success.

Note that if application initiates OBEX authentication, interoperability cannot be guaranteed with devices that lack a user interface. Therefore, it is strongly recommended that OBEX authentication be turned off.

Parameter

type — Signal identity, CSR_BT_BIPS_CHALLENGE_IND/RES.

pHandleInst — This value is used for identifying which BIPS instance the signal is coming from.

deviceAddr — The Bluetooth address of the device that is connecting

connectType — Indicates the type of session, where the following sessions are defined:

CSR_BT_BIPS_PUSH_CONNECT (Image push)
CSR_BT_BIPS_RC_CONNECT (Remote Camera)
CSR_BT_BIPS_AA_CONNECT (Automatic Archive)

These values are defined in csr_bt_bips_prim.h

challenged — Boolean indicating if the peer device has included a valid Authentication Challenge header in the incoming OBEX packet. If so, this parameter is set to TRUE, otherwise it is FALSE.

realmLength — Number of bytes in realm of type CsrUint16

*realm — A displayable string indicating for the peer device which userid and/or password to use. The first byte of the string is the character set of the string. The table below shows the different values for character set.

Note that this pointer shall be allocated by the application, and that this pointer can be NULL because the realm field is optional to use.

Char set Code	Meaning
0	ASCII
1	ISO-8859-1

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

2	ISO-8859-2
3	ISO-8859-3
4	ISO-8859-4
5	ISO-8859-5
6	ISO-8859-6
7	ISO-8859-7
8	ISO-8859-8
9	ISO-8859-9
0xFF = 255	UNICODE

passwordLength — The length of the challenge password of type CsrUint16

*password — Containing the challenge password of the OBEX authentication. This pointer shall be allocated by the application in order to initiates OBEX authentication. Note that if this pointer is NULL OBEX authentication is NOT initiates.

*userId — Zero terminated string (ASCII) containing the userId for the authentication. This is a pointer that shall be allocated by the application with a maximum of length of BT_OBEX_MAX_AUTH_USERID_LENGTH.

Note that this pointer can be NULL because the userId is optional to set by the peer device.

Library Function

A library function is provided by csr_bt_bips_lib.h, and should be used for building and sending the CSR_BT_BIPS_CHALLENGE_RES signal:

```
void CsrBtBipsChallengeResSend(CsrUint16  realmLength,
                               data_ptr_t *realm,
                               CsrUint16  passwordLength,
                               CsrCharString  *userId);
```

The parameters are as described in the table above.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

4.3 Image Push Primitives

This section contains description of primitives that are specific to Image Push.

4.3.1 CSR_BT_BIPS_PUSH_GET_CAPABILITIES_HEADER

Primitives \ Parameters	type	pHandleInst	connectionId	responseCode	srmpOn
CSR_BT_BIPS_PUSH_GET_CAPABILITIES_HEADER_IND	✓	✓	✓		
CSR_BT_BIPS_PUSH_GET_CAPABILITIES_HEADER_RES	✓			✓	✓

Table 11: CSR_BT_BIPS_PUSH_GET_CAPABILITIES_HEADER Primitives

Description

The indication and response signal is the first part of an operation where the client has requested to retrieve the imaging-capabilities object from the BIP server. If the BIP server accepts the request from the client the following messages will be CSR_BT_BIPS_PUSH_GET_CAPABILITIES_OBJECT indications/responses, see section 4.3.2.

Parameters

type — Signal identity, CSR_BT_BIPS_PUSH_GET_CAPABILITIES_HEADER_IND/RES.

pHandleInst — This value is used for identifying which BIPS instance the signal is coming from.

connectionId — The connection Id for this session, the BIP client must use this Id in the request.

responseCode — For accepting the GetCapabilities request the code is:
CSR_BT_OBEX_SUCCESS_RESPONSE_CODE.

The following response codes reject the GetCapabilities request:
CSR_BT_OBEX_FORBIDDEN_RESPONSE_CODE
CSR_BT_OBEX_BAD_REQUEST_RESPONSE_CODE
CSR_BT_OBEX_SERVICE_UNAVAILABLE_RESPONSE_CODE

If the connectionId is invalid it is recommended that the operation is being rejected with the response code:
CSR_BT_OBEX_SERVICE_UNAVAILABLE_RESPONSE_CODE.

The responseCodes are defined in (csr_bt_obex.h) with the following type CsrBtObexResponseCode and can also be found in IrDA Object Exchange Protocol. The meaning of the response codes for the Basic Imaging Profile is described in [BIP].

srmpOn — If Single Response Mode is enabled, see section 4.2.1, the BIP server can instruct the BIP Client to wait for the next response packet during a GET Operation by setting the srmpOn parameter TRUE.

If used, the srmpOn parameter shall be TRUE in the first GET response, and may be used in consecutive GET response packets to cause the Client to continue its wait; however, once the srmpOn parameter is FALSE in a GET response, the srmpOn parameter are consider to be FALSE for the duration of the operation.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

Library Function

A library function is provided by csr_bt_bips_lib.h, and should be used for building and sending the CSR_BT_BIPS_PUSH_GET_CAPABILITIES_HEADER_RES signal:

```
void CsrBtBipsPushGetCapabilitiesHeaderResSend(CsrSchedQid pHandleInst,
                                    CsrBtObexResponseCode   responseCode,
                                                  CsrBool     srmpOn);
```

The parameters are as described in the table above.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

4.3.2 CSR_BT_BIPS_PUSH_GET_CAPABILITIES_OBJECT

Primitives \ Parameters	type	pHandleInst	connectionId	responseCode	capabilitiesObjectLength	*capabilitiesObject	srmpOn
CSR_BT_BIPS_PUSH_GET_CAPABILITIES_OBJECT_IND	✓	✓	✓				
CSR_BT_BIPS_PUSH_GET_CAPABILITIES_OBJECT_RES	✓			✓	✓	✓	✓

Table 12: CSR_BT_BIPS_PUSH_GET_CAPABILITIES_OBJECT Primitives

Description

This signal is part of an operation where the client has requested to retrieve the imaging-capabilities object from the BIP server. The imaging-capabilities object is a mandatory object that describes in detail the various options, formats and attributes that are supported by the BIP server.

An example of an Imaging-capabilities object for the BIP Image Push client is illustrated below:

```
<imaging-capabilities version="1.0">
<preferred-format encoding="JPEG" pixel="1280*960" />
<image-formats encoding="JPEG" pixel="160*120" maxsize="5000" />
<image-formats encoding="JPEG" pixel="320*240" />
<image-formats encoding="JPEG" pixel="640*480" />
<image-formats encoding="JPEG" pixel="1280*960" />
<attachment-formats content-type="audio/basic" />
<filtering-parameters created="1" modified="1" />
</imaging-capabilities>
```

For a description of the definition for the imaging-capabilities object, together with the elements, and the attributes used in the imaging-capabilities object, please refer to [BIP].

Parameters

type	Signal identity, CSR_BT_BIPS_PUSH_GET_CAPABILITIES_OBJECT_IND/RES.
pHandleInst	This value is used for identifying which BIPS instance the signal is coming from.
connectionId	The connection Id for this session, the BIP client must use this Id in the request.
responseCode	A successful response for an object that fits in one response packet is CSR_BT_OBEX_SUCCESS_RESPONSE_CODE. If the response is large enough to require multiple CSR_BT_BIPS_GET_CAPABILITIES_OBJECT packets, only the last response must be CSR_BT_OBEX_SUCCESS_RESPONSE_CODE, and the other must be CSR_BT_OBEX_CONTINUE_RESPONSE_CODE. The following response codes reject the GetCapabilities operation: CSR_BT_OBEX_BAD_REQUEST_RESPONSE_CODE CSR_BT_OBEX_FORBIDDEN_RESPONSE_CODE CSR_BT_OBEX_SERVICE_UNAVAILABLE_RESPONSE_CODE If the connectionId is invalid it is recommended that the operation is being rejected with the response code: CSR_BT_OBEX_SERVICE_UNAVAILABLE_RESPONSE_CODE.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

	The responseCodes are defined in (csr_bt_obex.h) with the following type CsrBtObexResponseCode and can also be found in IrDA Object Exchange Protocol. The meaning of the response codes for the Basic Imaging Profile is described in [BIP].
capabilitiesObjectLength	The length of the capabilities object part being sent.
*capabilitiesObject	The imaging-capabilities object or part of it being sent.
srmpOn	If Single Response Mode is enabled, see section 4.2.1, the BIP server can instruct the BIP Client to wait for the next response packet during a GET Operation by setting the srmpOn parameter TRUE. If used, the srmpOn parameter shall be TRUE in the first GET response, and may be used in consecutive GET response packets to cause the Client to continue its wait; however, once the srmpOn parameter is FALSE in a GET response, the srmpOn parameter are consider to be FALSE for the duration of the operation.

Library Function

A library function is provided by csr_bt_bips_lib.h, and should be used for building and sending the CSR_BT_BIPS_PUSH_GET_CAPABILITIES_OBJECT_RES signal:

```
void CsrBtBipsPushGetCapabilitiesObjectResSend(CsrSchedQid pHandleInst,
                                               CsrUint16  capabilitiesObjectLength,
                                               CsrUint8   *capabilitiesObject,
                                    CsrBtObexResponseCode  responseCode,
                                               CsrBool    srmpOn);
```

parameters are as described in the table above.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

4.3.3 CSR_BT_BIPS_PUSH_PUT_IMAGE_HEADER

Primitives \ Parameters	type	pHandleInst	connectionId	imageDescriptorOffset	imageDescriptorLength	ucs2nameOffset	payloadLength	*payload	responseCode	imageHandle	srmpOn
CSR_BT_BIPS_PUSH_PUT_IMAGE_HEADER_IND	✓	✓	✓	✓	✓	✓	✓	✓			
CSR_BT_BIPS_PUSH_PUT_IMAGE_HEADER_RES	✓								✓	✓	✓

Table 13: CSR_BT_BIPS_PUSH_PUT_IMAGE_HEADER Primitives

Description

The indication and response signal is the first part of an operation where the client pushed an image to the BIP server. If the BIP server accepts the request from the client the following messages will be CSR_BT_BIPS_PUSH_PUT_IMAGE_FILE indications/responses, see section 4.3.4.

An example of an image descriptor object is illustrated below:

```
<image-descriptor version="1.0">
<image encoding="JPEG" pixel="1280*960" size="50000000" />
</image-descriptor>
```

Parameters

type	Signal identity, CSR_BT_BIPS_PUT_IMAGE_HEADER_IND/RES.
pHandleInst	This value is used for identifying which BIPS instance the signal is coming from.
connectionId	The connection Id for this session, the BIP client must use this Id in the request.
imageDescriptorOffset	The image descriptor describes the properties of the image being pushed. An example is seen above. For a description of the definition for the image descriptor, together with the elements, and the attributes used in the image descriptor, please refer to [BIP]. The value in this parameter indicates the offset in the payload data where the image descriptor data starts. I.e. the data is located at payload[imageDescriptorOffset].
imageDescriptorLength	The length of the imageDescriptor.
ucs2nameOffset	A null terminated 16 bit Unicode text string (UCS2) containing the name of the image being push to the BIP server. The function "CsrUcs2ByteString2Utf8" can be used for converting the name from UCS2 into a null terminated UTF8 text string. The value in this parameter indicates the offset in the payload data where the name starts. I.e. the data is located at payload[ucs2nameOffset].
payloadLength	The length of the received payload.
*payload	Pointer to the received payload.
responseCode	For accepting the PutImage request the code is CSR_BT_OBEX_SUCCESS_RESPONSE_CODE The following response codes reject the PutImage request: CSR_BT_OBEX_FORBIDDEN_RESPONSE_CODE CSR_BT_OBEX_BAD_REQUEST_RESPONSE_CODE

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

	CSR_BT_OBEX_NOT_ACCEPTABLE_RESPONSE_CODE CSR_BT_OBEX_SERVICE_UNAVAILABLE_RESPONSE_CODE CSR_BT_OBEX_PRECONDITION_FAILED_RESPONSE_CODE If the connectionId is invalid it is recommended that the operation is being rejected with the response code: CSR_BT_OBEX_SERVICE_UNAVAIABLE_RESPONSE_CODE. The responseCodes are defined in (csr_bt_obex.h) with the following type CsrBtObexResponseCode and can also be found in IrDA Object Exchange Protocol. The meaning of the response codes for the Basic Imaging Profile is described in [BIP].
imageHandle	The BIP server must assign an image handle to the image being pushed. This is done so the client can send the thumbnail and/or the attachments that might be linked to the image, see section 4.3.5 and 4.3.7. Image handles are 7 character long strings containing only the digits 0 to 9, and are only required to be unique on the source device. For a description of the imaging handles and an implementation guideline, please refer to [BIP].
srmpOn	If Single Response Mode is enabled, see section 4.2.1, the BIP server can instruct the BIP Client to wait for the next response packet during a PUT Operation by setting the srmpOn parameter TRUE. If used, the srmpOn parameter shall be TRUE in the first PUT response and may be used in consecutive PUT response packets to cause the Client to continue its wait; however, once the srmpOn parameter is FALSE in a PUT response, the srmpOn parameter are consider to be FALSE for the duration of the operation.

Library Function

A library function is provided by csr_bt_bips_lib.h, and should be used for building and sending the CSR_BT_BIPS_PUSH_PUT_IMAGE_HEADER_RES signal:

```
void CsrBtBipsPushPutImageHeaderResSend(CsrSchedQid           pHandleInst,
                                        CsrUint8              *imageHandle,
                                        CsrBtObexResponseCode responseCode,
                                        CsrBool               srmpOn);
```

The parameters are as described in the table above.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

4.3.4 CSR_BT_BIPS_PUSH_PUT_IMAGE_FILE

Parameters / Primitives	type	pHandleInst	connectionId	finalFlag	imageFileOffset	imageFileLength	PayloadLength	*payload	responseCode	srmpOn
CSR_BT_BIPS_PUSH_PUT_IMAGE_FILE_IND	✓	✓	✓	✓	✓	✓	✓	✓		
CSR_BT_BIPS_PUSH_PUT_IMAGE_FILE_RES	✓								✓	✓

Table 14: CSR_BT_BIPS_PUSH_PUT_IMAGE_FILE Primitives

Description

This signal is part of an operation where the client pushed an image to the BIP server. Please notice that the BIP server may use the response code to request the client to send the thumbnail version of the image it just received. This capability is designed for servers that do not have the ability to convert images into imaging thumbnail format.

Parameters

type	Signal identity, CSR_BT_BIPS_PUSH_PUT_IMAGE_FILE_IND/RES.
pHandleInst	This value is used for identifying which BIPS instance the signal is coming from.
connectionId	The connection Id for this session, the BIP client must use this Id in the request.
finalFlag	Indicates whether or not it is the last part of the image being sent. TRUE indicates that the last part has been sent.
imageFileOffset	The image data being received. The value in this parameter indicates the offset in the payload data where the image data starts. I.e. the data is located at payload[imageFileOffset].
imageFileLength	The length of the image part being received.
payloadLength	The length of the received payload.
*payload	Pointer to the received payload.
responseCode	If all of the image is received correctly the CSR_BT_OBEX_SUCCESS_RESPONSE_CODE or the CSR_BT_OBEX_PARTIAL_CONTENT_RESPONSE_CODE response codes must be returned. The CSR_BT_OBEX_PARTIAL_CONTENT_RESPONSE_CODE indicates that the BIP server requests the thumbnail version of the image just sent by the client. If an operation involves several request/response messages (i.e., the image being transferred does not fit in one OBEX request packet) the BIP server must respond with the CSR_BT_OBEX_PARTIAL_CONTENT_RESPONSE_CODE in the very last packet (where it replaces the CSR_BT_OBEX_SUCCESS_RESPONSE_CODE). The intermediate response packets must carry the CSR_BT_OBEX_CONTINUE_RESPONSE_CODE. The following response codes reject the PutImage request: CSR_BT_OBEX_FORBIDDEN_RESPONSE_CODE

CSR Synergy Bluetooth 18.2.0 OBEX Basic Imaging Profile Server API

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

CSR_BT_OBEX_BAD_REQUEST_RESPONSE_CODE
CSR_BT_OBEX_NOT_ACCEPTABLE_RESPONSE_CODE
CSR_BT_OBEX_SERVICE_UNAVAILABLE_RESPONSE_CODE
CSR_BT_OBEX_PRECONDITION_FAILED_RESPONSE_CODE

If the connectionId is invalid it is recommended that the operation is being rejected with the response code: CSR_BT_OBEX_SERVICE_UNAVAILABLE_RESPONSE_CODE.

The responseCodes are defined in (csr_bt_obex.h) with the following type CsrBtObexResponseCode and can also be found in IrDA Object Exchange Protocol. The meaning of the response codes for the Basic Imaging Profile is described in [BIP].

srmpOn	If Single Response Mode is enabled, see section 4.2.1, the BIP server can instruct the BIP Client to wait for the next response packet during a PUT Operation by setting the srmpOn parameter TRUE. If used, the srmpOn parameter shall be TRUE in the first PUT response and may be used in consecutive PUT response packets to cause the Client to continue its wait; however, once the srmpOn parameter is FALSE in a PUT response, the srmpOn parameter are consider to be FALSE for the duration of the operation.

Library Function

A library function is provided by csr_bt_bips_lib.h, and should be used for building and sending the CSR_BT_BIPS_PUSH_PUT_IMAGE_FILE_RES signal:

```
void CsrBtBipsPushPutImageFileResSend(CsrSchedQid           pHandleInst,
                                      CsrBtObexResponseCode responseCode,
                                      CsrBool               srmpOn);
```

The parameters are as described in the table above.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

4.3.5 CSR_BT_BIPS_PUSH_PUT_LINKED_THUMBNAIL_HEADER

Primitives \ Parameters	type	pHandleInst	connectionId	imageHandleOffset	payloadLength	*payload	responseCode	srmpOn
CSR_BT_BIPS_PUSH_PUT_LINKED_THUMBNAIL_HEADER_IND	✓	✓	✓	✓	✓	✓		
CSR_BT_BIPS_PUSH_PUT_LINKED_THUMBNAIL_HEADER_RES	✓						✓	✓

Table 15: CSR_BT_BIPS_PUSH_PUT_LINKED_THUMBNAIL_HEADER Primitives

Description

The Indication and response signal is the first part of an operation where the client pushes a thumbnail version of the image just being sent to the BIP server. If the BIP server accepts the request from the client the following messages will be CSR_BT_BIPS_PUSH_PUT_LINKED_THUMBNAIL_FILE indications/responses, see section 4.3.6.

Please notice that the client uses this operation only in response to a PutImage response indicating that the BIP server needs the thumbnail version of the image.

Parameters

type	Signal identity, CSR_BT_BIPS_PUT_LINKED_THUMBNAIL_HEADER_IND/RES.
pHandleInst	This value is used for identifying which BIPS instance the signal is coming from.
connectionId	The connection Id for this session, the BIP client must use this Id in the request
imageHandleOffset	The handle of the images the thumbnail is linked to. The value in this parameter indicates the offset in the payload data where the image handle starts. I.e. the data is located at payload[imageHandleOffset]. The length of this data is always 7 bytes as this is the length of the image handle.
payloadLength	The length of the received payload.
*payload	Pointer to the received payload.
responseCode	For accepting the PutILinkedThumbnail request the code is: CSR_BT_OBEX_SUCCESS_RESPONSE_CODE. The following response codes reject the PutILinkedThumbnail request: CSR_BT_OBEX_FORBIDDEN_RESPONSE_CODE CSR_BT_OBEX_BAD_REQUEST_RESPONSE_CODE CSR_BT_OBEX_SERVICE_UNAVAIABLE_RESPONSE_CODE CSR_BT_OBEX_PRECONDITION_FAILED_RESPONSE_CODE If the connectionId is invalid it is recommended that the operation is being rejected with the response code: CSR_BT_OBEX_SERVICE_UNAVAIABLE_RESPONSE_CODE. The responseCodes are defined in (csr_bt_obex.h) with the following type CsrBtObexResponseCode and can also be found in IrDA Object Exchange Protocol. The meaning of the response codes for the Basic Imaging Profile is described in [BIP].

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

srmpOn	If Single Response Mode is enabled, see section 4.2.1, the BIP server can instruct the BIP Client to wait for the next response packet during a PUT Operation by setting the srmpOn parameter TRUE. If used, the srmpOn parameter shall be TRUE in the first PUT response and may be used in consecutive PUT response packets to cause the Client to continue its wait; however, once the srmpOn parameter is FALSE in a PUT response, the srmpOn parameter are consider to be FALSE for the duration of the operation.

Library Function

A library function is provided by csr_bt_bips_lib.h, and should be used for building and sending the CSR_BT_BIPS_PUSH_PUT_LINKED_THUMBNAIL_HEADER_RES signal:

```
void CsrBtBipsPushPutLinkedThumbnailHeaderResSend(CsrSchedQid pHandleInst,
                                        CsrBtObexResponseCode responseCode,
                                                  CsrBool     srmpOn);
```

The parameters are as described in the table above.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

4.3.6 CSR_BT_BIPS_PUSH_PUT_LINKED_THUMBNAIL_FILE

Primitives \ Parameters	type	pHandleInst	connectionId	finalFlag	thumbnailFileOffset	thumbnailFileLength	payloadLength	*payload	responseCode	srmpOn
CSR_BT_BIPS_PUSH_PUT_LINKED_THUMBNAIL_FILE_IND	✓	✓	✓	✓	✓	✓	✓	✓		
CSR_BT_BIPS_PUSH_PUT_LINKED_THUMBNAIL_FILE_RES	✓								✓	✓

Table 16: CSR_BT_BIPS_PUSH_PUT_LINKED_THUMBNAIL_FILE Primitives

Description

This signal is a part of an operation where the client pushes a thumbnail version of an image to the BIP server.

Parameters

type — Signal identity, CSR_BT_BIPS_PUSH_PUT_LINKED_THUMBNAIL_FILE_IND/RES.

pHandleInst — This value is used for identifying which BIPS instance the signal is coming from.

connectionId — The connection Id for this session, the BIP client must use this Id in the request.

finalFlag — Indicates whether or not it is the last part of the thumbnail being sent. TRUE indicates that the last part has been sent.

thumbnailFileOffset — The part of the thumbnail being received.

The value in this parameter indicates the offset in the payload data where the image thumbnail data starts. I.e. the data is located at payload[thumbnailFileOffset].

thumbnailFileLength — The length of the thumbnail part being received.

payloadLength — The length of the received payload.

*payload — Pointer to the received payload.

responseCode — To accept the request the intermediate response packets must carry the CSR_BT_OBEX_CONTINUE_RESPONSE_CODE (i.e., when the finalFlag is FALSE). The CSR_BT_OBEX_SUCCESS_RESPONSE_CODE must be used in the very last packet (i.e., when the finalFlag is TRUE).

The following response codes reject the PutLinkedThumbnail request:
CSR_BT_OBEX_FORBIDDEN_RESPONSE_CODE
CSR_BT_OBEX_BAD_REQUEST_RESPONSE_CODE
CSR_BT_OBEX_NOT_ACCEPTABLE_RESPONSE_CODE
CSR_BT_OBEX_SERVICE_UNAVAILABLE_RESPONSE_CODE
CSR_BT_OBEX_PRECONDITION_FAILED_RESPONSE_CODE

If the connectionId is invalid it is recommended that the operation is being rejected with the response code:
CSR_BT_OBEX_SERVICE_UNAVAILABLE_RESPONSE_CODE.

The responseCodes are defined in (csr_bt_obex.h) with the following type CsrBtObexResponseCode and can also be found in IrDA Object Exchange Protocol.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

	The meaning of the response codes for the Basic Imaging Profile is described in [BIP].
srmpOn	If Single Response Mode is enabled, see section 4.2.1, the BIP server can instruct the BIP Client to wait for the next response packet during a PUT Operation by setting the srmpOn parameter TRUE. If used, the srmpOn parameter shall be TRUE in the first PUT response and may be used in consecutive PUT response packets to cause the Client to continue its wait; however, once the srmpOn parameter is FALSE in a PUT response, the srmpOn parameter are consider to be FALSE for the duration of the operation.

Library Function

A library function is provided by csr_bt_bips_lib.h, and should be used for building and sending the CSR_BT_BIPS_PUSH_PUT_LINKED_THUMBNAIL_FILE_RES signal:

```
void CsrBtBipsPushPutLinkedThumbnailFileResSend(CsrSchedQid           pHandleInst,
                                                CsrBtObexResponseCode responseCode,
                                                CsrBool               srmpOn);
```

The parameters are as described in the table above.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

4.3.7 CSR_BT_BIPS_PUSH_PUT_LINKED_ATTACHMENT_HEADER

Primitives \ Parameters	type	pHandleInst	connectionId	imageHandleOffset	attachmentDescriptorOffset	attachmentDescriptorLength	payloadLength	*payload	responseCode	srmpOn
CSR_BT_BIPS_PUSH_PUT_LINKED_ATTACHMENT_HEADER_IND	✓	✓	✓	✓	✓	✓	✓	✓		
CSR_BT_BIPS_PUSH_PUT_LINKED_ATTACHMENT_HEADER_RES	✓								✓	✓

Table 17: CSR_BT_BIPS_PUSH_PUT_LINKED_ATTACHMENT_HEADER Primitives

Description

The indication and response signal is the first part of an operation where the client pushes attachments associate with an image to the BIP server after the image has been sent to the server within the context of an OBEX session. If the BIP server accepts the request from the client the following messages will be CSR_BT_BIPS_PUSH_PUT_LINKED_ATTACHMENT_FILE indications/responses, see section 4.3.8.

An example of an attachment descriptor object is illustrated below:

```
<attachment-descriptor version="1.0">
<attachment name="DSCF0001.txt" content-type="text/plain"
size="5000" />
</attachment-descriptor>
```

Parameters

type	Signal identity, CSR_BT_BIPS_PUSH_PUT_LINKED_ATTACHMENT_HEADER_IND/RES.
pHandleInst	This value is used for identifying which BIPS instance the signal is coming from.
connectionId	The connection Id for this session, the BIP client must use this Id in the request.
imageHandleOffset	The handle of the images, which the attachment is associated with. The value in this parameter indicates the offset in the payload data where the image handle starts. I.e. the data is located at payload[imageHandleOffset]. The image handle is always of length 7 bytes.
attachmentDescriptorOffset	The attachmentDescriptor describes the properties of the attachment being pushed. An example is seen above. For a description of the definition for the attachment descriptor, together with the elements, and the attributes used in the attachment descriptor, please refer to [BIP]. The value in this parameter indicates the offset in the payload data where the descriptor data starts. I.e. the data is located at payload[attachmentDescriptorOffset].
attachmentDescriptorLength	The length of the attachmentDescriptor.
payloadLength	The length of the received payload.
*payload	Pointer to the received payload.

CSR Synergy Bluetooth 18.2.0 OBEX Basic Imaging Profile Server API

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

responseCode	For accepting the PutILinkedAttachment request the code is CSR_BT_OBEX_SUCCESS_RESPONSE_CODE. The following response codes reject the PutILinkedAttachment request: CSR_BT_OBEX_FORBIDDEN_RESPONSE_CODE CSR_BT_OBEX_BAD_REQUEST_RESPONSE_CODE CSR_BT_OBEX_NOT_ACCEPTABLE_RESPONSE_CODE CSR_BT_OBEX_SERVICE_UNAVAILABLE_RESPONSE_CODE CSR_BT_OBEX_PRECONDITION_FAILED_RESPONSE_CODE If the connectionId is invalid it is recommended that the operation is being rejected with the response code: CSR_BT_OBEX_SERVICE_UNAVAILABLE_RESPONSE_CODE. The responseCodes are defined in (csr_bt_obex.h) with the following type CsrBtObexResponseCode and can also be found in IrDA Object Exchange Protocol. The meaning of the response codes for the Basic Imaging Profile is described in [BIP].
srmpOn	If Single Response Mode is enabled, see section 4.2.1, the BIP server can instruct the BIP Client to wait for the next response packet during a PUT Operation by setting the srmpOn parameter TRUE. If used, the srmpOn parameter shall be TRUE in the first PUT response and may be used in consecutive PUT response packets to cause the Client to continue its wait; however, once the srmpOn parameter is FALSE in a PUT response, the srmpOn parameter are consider to be FALSE for the duration of the operation.

Library Function

A library function is provided by csr_bt_bips_lib.h, and should be used for building and sending the CSR_BT_BIPS_PUSH_PUT_LINKED_ATTACHMENT_HEADER_RES signal:

```
void CsrBtBipsPushPutLinkedAttachmentHeaderResSend(CsrSchedQid pHandleInst,
                                                   CsrBtObexResponseCode responseCode,
                                                   CsrBool     srmpOn);
```

The parameters are as described in the table above.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

4.3.8 CSR_BT_BIPS_PUSH_PUT_LINKED_ATTACHMENT_FILE

Primitives \ Parameters	type	pHandleInst	connectionId	finalFlag	attachmentFileOffset	attachmentFileLength	payloadLength	*payload	responseCode	srmpOn
CSR_BT_BIPS_PUSH_PUT_LINKED_ATTACHMENT_FILE_IND	✓	✓	✓	✓	✓	✓	✓	✓		
CSR_BT_BIPS_PUSH_PUT_LINKED_ATTACHMENT_FILE_RES	✓								✓	✓

Table 18: CSR_BT_BIPS_PUSH_PUT_LINKED_ATTACHMENT_FILE Primitives

Description

This signal is a part of an operation where the client pushes attachments associates with an image to the BIP server.

Parameters

type — Signal identity, CSR_BT_BIPS_PUSH_PUT_LINKED_ATTACHMENT_FILE_IND/RES.

pHandleInst — This value is used for identifying which BIPS instance the signal is coming from.

connectionId — The connection Id for this session, the BIP client must use this Id in the request.

finalFlag — Indicates whether or not it is the last part of the attachment being sent. TRUE indicates that the last part has been sent.

attachmentFileOffset — The part of the attachment being received.

The value in this parameter indicates the offset in the payload data where the attachment file data starts. I.e. the data is located at payload[attachmentFileOffset].

attachmentFileLength — The length of the attachment part being received.

payloadLength — The length of the received payload.

*payload — Pointer to the received payload.

responseCode — To accept the request the intermediate response packets must carry the CSR_BT_OBEX_CONTINUE_RESPONSE_CODE (i.e., when the finalFlag is FALSE). The CSR_BT_OBEX_SUCCESS_RESPONSE_CODE must be used in the very last packet (i.e., when the finalFlag is TRUE).

The following response codes reject the PutLinkedAttachment request:
CSR_BT_OBEX_FORBIDDEN_RESPONSE_CODE
CSR_BT_OBEX_BAD_REQUEST_RESPONSE_CODE
CSR_BT_OBEX_NOT_ACCEPTABLE_RESPONSE_CODE
CSR_BT_OBEX_SERVICE_UNAVAIABLE_RESPONSE_CODE
CSR_BT_OBEX_PRECONDITION_FAILED_RESPONSE_CODE

If the connectionId is invalid it is recommended that the operation is being rejected with the response code:
CSR_BT_OBEX_SERVICE_UNAVAIABLE_RESPONSE_CODE.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

	The responseCodes are defined in (csr_bt_obex.h) with the following type CsrBtObexResponseCode and can also be found in IrDA Object Exchange Protocol. The meaning of the response codes for the Basic Imaging Profile is described in [BIP].
srmpOn	If Single Response Mode is enabled, see section 4.2.1, the BIP server can instruct the BIP Client to wait for the next response packet during a PUT Operation by setting the srmpOn parameter TRUE. If used, the srmpOn parameter shall be TRUE in the first PUT response and may be used in consecutive PUT response packets to cause the Client to continue its wait; however, once the srmpOn parameter is FALSE in a PUT response, the srmpOn parameter are consider to be FALSE for the duration of the operation.

Library Function

A library function is provided by csr_bt_bips_lib.h, and should be used for building and sending the CSR_BT_BIPS_PUSH_PUT_LINKED_ATTACHMENT_FILE_RES signal:

```
void CsrBtBipsPushPutLinkedAttachmentFileResSend(CsrSchedQid pHandleInst,
                                                 CsrBtObexResponseCode responseCode,
                                                 CsrBool     srmpOn);
```

The parameters are as described in the table above.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

4.4 Remote Camera Primitives

This section contains description of primitives that are specific to Remote Camera.

4.4.1 CSR_BT_BIPS_RC_GET_MONITORING_IMAGE_HEADER

Primitives \ Parameters	type	pHandleInst	connectionId	storeFlag	responseCode	imageHandle	srmpOn
CSR_BT_BIPS_RC_GET_MONITORING_IMAGE_HEADER_IND	✓	✓	✓	✓			
CSR_BT_BIPS_RC_GET_MONITORING_IMAGE_HEADER_RES	✓				✓	✓	✓

Table 19: CSR_BT_BIPS_RC_GET_MONITORING_IMAGE_HEADER Primitives

Description

The indication and response signal is the first part of an operation where the client requests a monitoring image from the BIP server. In the signal the client can indicate if the image should be stored as a file on the server by setting the storeFlag. If the BIP server accepts the request from the client the following messages will be CSR_BT_BIPS_RC_GET_MONITORING_IMAGE_OBJECT indications/responses, see section 4.4.2.

Parameters

type	Signal identity, CSR_BT_BIPS_RC_GET_MONITORING_IMAGE_HEADER_IND/RES.
pHandleInst	This value is used for identifying which BIPS instance the signal is coming from.
connectionId	The connection Id for this session, the BIP client must use this Id in the request.
storeFlag	Indicates whether or not the camera should record the image when the monitoring image has been transmitted.
responseCode	For accepting the GetMonitoringImage request the code is CSR_BT_OBEX_SUCCESS_RESPONSE_CODE. The following response codes reject the GetMonitoringImage request: CSR_BT_OBEX_FORBIDDEN_RESPONSE_CODE CSR_BT_OBEX_BAD_REQUEST_RESPONSE_CODE CSR_BT_OBEX_NOT_ACCEPTABLE_RESPONSE_CODE CSR_BT_OBEX_SERVICE_UNAVAILABLE_RESPONSE_CODE CSR_BT_OBEX_PRECONDITION_FAILED_RESPONSE_CODE If the connectionId is invalid it is recommended that the operation is being rejected with the response code: CSR_BT_OBEX_SERVICE_UNAVAILABLE_RESPONSE_CODE. The responseCodes are defined in (csr_bt_obex.h) with the following type CsrBtObexResponseCode and can also be found in IrDA Object Exchange Protocol. The meaning of the response codes for the Basic Imaging Profile is described in [BIP].
imageHandle	If the storeFlag was set an image file is created and the handle for this file is returned to the client in this variable. If no image is stored this variable must be empty.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

srmpOn	If Single Response Mode is enabled, see section 4.2.1, the BIP server can instruct the BIP Client to wait for the next response packet during a GET Operation by setting the srmpOn parameter TRUE. If used, the srmpOn parameter shall be TRUE in the first GET response, and may be used in consecutive GET response packets to cause the Client to continue its wait; however, once the srmpOn parameter is FALSE in a GET response, the srmpOn parameter are consider to be FALSE for the duration of the operation.

Library Function

A library function is provided by csr_bt_bips_lib.h, and should be used for building and sending the CSR_BT_BIPS_RC_GET_MONITORING_IMAGE_HEADER_RES signal:

```
void CsrBtBipsRcGetMonitoringImageHeaderResSend(CsrSchedQid pHandleInst,
                                        CsrBtObexResponseCode  responseCode,
                                                    CsrUint8    *imgHandle,
                                                    CsrBool     srmpOn);
```

The parameters are as described in the table above.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

4.4.2 CSR_BT_BIPS_RC_GET_MONITORING_IMAGE_OBJECT

Parameters / Primitives	type	pHandleInst	connectionId	allowedImageLength	responseCode	monitoringObjectLength	*monitoringObject	srmpOn
CSR_BT_BIPS_RC_GET_MONITORING_IMAGE_OBJECT_IND	✓	✓	✓	✓				
CSR_BT_BIPS_RC_GET_MONITORING_IMAGE_OBJECT_RES	✓				✓	✓	✓	✓

Table 20: CSR_BT_BIPS_RC_GET_MONITORING_IMAGE_OBJECT Primitives

Description

This signal is a part of an operation where the client has requested a monitoring image from the BIP server. Part of or the entire monitoring image is returned.

Parameters

type — Signal identity, CSR_BT_BIPS_RC_GET_MONITORING_IMAGE_OBJECT_IND/RES.

pHandleInst — This value is used for identifying which BIPS instance the signal is coming from.

connectionId — The connection Id for this session, the BIP client must use this Id in the request.

allowedImageLength — Indicates how much data is maximal allowed to return in the response. This must be obeyed by the application. Please be aware that this number may change from indication to indication.

responseCode — If the image fits in one response or the final part is being transmitted the responseCode must be CSR_BT_OBEX_SUCCESS_RESPONSE_CODE. All intermediate responses must be CSR_BT_OBEX_CONTINUE_RESPONSE_CODE.

The following response codes reject the GetMonitoringImage request:
CSR_BT_OBEX_FORBIDDEN_RESPONSE_CODE
CSR_BT_OBEX_BAD_REQUEST_RESPONSE_CODE
CSR_BT_OBEX_NOT_ACCEPTABLE_RESPONSE_CODE
CSR_BT_OBEX_SERVICE_UNAVAILABLE_RESPONSE_CODE
CSR_BT_OBEX_PRECONDITION_FAILED_RESPONSE_CODE

If the connectionId is invalid it is recommended that the operation is being rejected with the response code:
CSR_BT_OBEX_SERVICE_UNAVAILABLE_RESPONSE_CODE.

The responseCodes are defined in (csr_bt_obex.h) with the following type CsrBtObexResponseCode and can also be found in IrDA Object Exchange Protocol. The meaning of the response codes for the Basic Imaging Profile is described in [BIP].

monitoringObjectLength — The length of the object pointed to by *monitoringObject. This must not be larger that the value given in allowedImageLength. If data is larger than this it must be fragmented and send in multiple signals.

*monitoringObject — A pointer to the monitoring image data itself. Must be fragmented if necessary.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

srmpOn	If Single Response Mode is enabled, see section 4.2.1, the BIP server can instruct the BIP Client to wait for the next response packet during a GET Operation by setting the srmpOn parameter TRUE. If used, the srmpOn parameter shall be TRUE in the first GET response, and may be used in consecutive GET response packets to cause the Client to continue its wait; however, once the srmpOn parameter is FALSE in a GET response, the srmpOn parameter are consider to be FALSE for the duration of the operation.

Library Function

A library function is provided by csr_bt_bips_lib.h, and should be used for building and sending the CSR_BT_BIPS_RC_GET_MONITORING_IMAGE_OBJECT_RES signal:

```
void CsrBtBipsRcGetMonitoringImageObjectResSend(CsrSchedQid pHandleInst,
                                                CsrUint16   monitoringObjectLength,
                                                CsrUint8    *monitoringObject,
                                          CsrBtObexResponseCode responseCode,
                                                CsrBool     srmpOn);
```

The parameters are as described in the table above.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

4.4.3 CSR_BT_BIPS_RC_GET_IMAGE_PROPERTIES_HEADER

Primitives \ Parameters	type	pHandleInst	connectionId	imageHandleOffset	payloadLength	*payload	responseCode	srmpOn
CSR_BT_BIPS_RC_GET_IMAGE_PROPERTIES_HEADER_IND	✓	✓	✓	✓	✓	✓		
CSR_BT_BIPS_RC_GET_IMAGE_PROPERTIES_HEADER_RES	✓						✓	✓

Table 21: CSR_BT_BIPS_RC_GET_IMAGE_PROPERTIES_HEADER Primitives

Description

The indication and response signal is the first part of an operation where the client requests image properties of an image on the BIP server. If the BIP server accepts the request from the client the following messages will be CSR_BT_BIPS_RC_GET_IMAGE_PROPERTIES_OBJECT indications/responses, see section 4.4.4.

Parameters

type	Signal identity, CSR_BT_BIPS_RC_GET_IMAGE_PROPERTIES_HEADER_IND/RES.
pHandleInst	This value is used for identifying which BIPS instance the signal is coming from.
connectionId	The connection Id for this session, the BIP client must use this Id in the request.
imageHandleOffset	The handle of the image the client is requesting properties on. The value in this parameter indicates the offset in the payload data where the image handle file data starts. I.e. the data is located at payload[imageHandleOffset]. The image handle is always 7 bytes in length.
payloadLength	The length of the received payload.
*payload	Pointer to the received payload.
responseCode	For accepting the GetImageProperties request the code is CSR_BT_OBEX_SUCCESS_RESPONSE_CODE. The following response codes reject the GetImageProperties request: CSR_BT_OBEX_FORBIDDEN_RESPONSE_CODE CSR_BT_OBEX_BAD_REQUEST_RESPONSE_CODE CSR_BT_OBEX_NOT_ACCEPTABLE_RESPONSE_CODE CSR_BT_OBEX_SERVICE_UNAVAILABLE_RESPONSE_CODE CSR_BT_OBEX_PRECONDITION_FAILED_RESPONSE_CODE If the connectionId is invalid it is recommended that the operation is being rejected with the response code: CSR_BT_OBEX_SERVICE_UNAVAILABLE_RESPONSE_CODE. The responseCodes are defined in (csr_bt_obex.h) with the following type CsrBtObexResponseCode and can also be found in IrDA Object Exchange Protocol. The meaning of the response codes for the Basic Imaging Profile is described in [BIP].

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

srmpOn	If Single Response Mode is enabled, see section 4.2.1, the BIP server can instruct the BIP Client to wait for the next response packet during a GET Operation by setting the srmpOn parameter TRUE. If used, the srmpOn parameter shall be TRUE in the first GET response, and may be used in consecutive GET response packets to cause the Client to continue its wait; however, once the srmpOn parameter is FALSE in a GET response, the srmpOn parameter are consider to be FALSE for the duration of the operation.

Library Function

A library function is provided by csr_bt_bips_lib.h, and should be used for building and sending the CSR_BT_BIPS_RC_GET_IMAGE_PROPERTIES_HEADER_RES signal:

```
void CsrBtBipsRcGetImagePropertiesHeaderResSend(CsrSchedQid           pHandleInst,
                                                CsrBtObexResponseCode responseCode,
                                                CsrBool               srmpOn);
```

The parameters are as described in the table above.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

4.4.4 CSR_BT_BIPS_RC_GET_IMAGE_PROPERTIES_OBJECT

Parameters / Primitives	type	pHandleInst	connectionId	allowedImageLength	responseCode	propertiesObjectLength	*propertiesObject	smpOn
CSR_BT_BIPS_RC_GET_IMAGE_PROPERTIES_OBJECT_IND	✓	✓	✓	✓				
CSR_BT_BIPS_RC_GET_IMAGE_PROPERTIES_OBJECT_RES	✓				✓	✓	✓	✓

Table 22: CSR_BT_BIPS_RC_GET_IMAGE_PROPERTIES_OBJECT Primitives

Description

This signal is a part of an operation where the client has requested image properties on an image from the BIP server. Part of or the entire image properties object is returned.

An example of an image properties object is illustrated below:

```
<image-properties version="1.0" handle="1000001">
<native encoding="JPEG" pixel="1280*1024" size="1048476" />
<variant encoding="JPEG" pixel="640*480" />
<variant encoding="JPEG" pixel="160*120" />
<variant encoding="GIF" pixel="80*60-640*480" />
<attachment content-type="text/plain" name="DSCF0001.txt"
size="5120" />
<attachment content-type="audio/basic" name="DSCF0001.wav"
size="102400" />
</image-properties>
```

Parameters

type	Signal identity, CSR_BT_BIPS_RC_GET_IMAGE_PROPERTIES_OBJECT_IND/RES.
pHandleInst	This value is used for identifying which BIPS instance the signal is coming from.
connectionId	The connection Id for this session, the BIP client must use this Id in the request.
allowedImageLength	Indicates how much data is maximal allowed to return in the response. This must be obeyed by the application. Please be aware that this number may change from indication to indication.
responseCode	If the image fits in one response or the final part is being transmitted the responseCode must be CSR_BT_OBEX_SUCCESS_RESPONSE_CODE. All intermediate responses must be CSR_BT_OBEX_CONTINUE_RESPONSE_CODE. The following response codes reject the GetImageProperties request: CSR_BT_OBEX_FORBIDDEN_RESPONSE_CODE CSR_BT_OBEX_BAD_REQUEST_RESPONSE_CODE CSR_BT_OBEX_NOT_ACCEPTABLE_RESPONSE_CODE CSR_BT_OBEX_SERVICE_UNAVAILABLE_RESPONSE_CODE CSR_BT_OBEX_PRECONDITION_FAILED_RESPONSE_CODE If the connectionId is invalid it is recommended that the operation is being rejected with the response code: CSR_BT_OBEX_SERVICE_UNAVAILABLE_RESPONSE_CODE.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

	The responseCodes are defined in (csr_bt_obex.h) with the following type CsrBtObexResponseCode and can also be found in IrDA Object Exchange Protocol. The meaning of the response codes for the Basic Imaging Profile is described in [BIP].
propertiesObjectLength	The length of the object pointed to by *propertiesObject. This must not be larger that the value given in allowedImageLength. If data is larger than this it must be fragmented and send in multiple signals.
*propertiesObject	A pointer to the properties data itself. Must be fragmented if necessary. An example of the properties object is seen above. For a description of the definition for the image properties, together with the elements, and the attributes used in the properties descriptor, please refer to [BIP].
srmpOn	If Single Response Mode is enabled, see section 4.2.1, the BIP server can instruct the BIP Client to wait for the next response packet during a GET Operation by setting the srmpOn parameter TRUE. If used, the srmpOn parameter shall be TRUE in the first GET response, and may be used in consecutive GET response packets to cause the Client to continue its wait; however, once the srmpOn parameter is FALSE in a GET response, the srmpOn parameter are consider to be FALSE for the duration of the operation.

Library Function

A library function is provided by csr_bt_bips_lib.h, and should be used for building and sending the CSR_BT_BIPS_RC_GET_IMAGE_PROPERTIES_OBJECT_RES signal:

```
void CsrBtBipsRcGetImagePropertiesObjectResSend(CsrSchedQid pHandleInst,
                                                CsrUint16   propertiesObjectLength,
                                                CsrUint8    *propertiesObject,
                                          CsrBtObexResponseCode responseCode,
                                                CsrBool     srmpOn);
```

The parameters are as described in the table above.

CSR Synergy Bluetooth 18.2.0 OBEX Basic Imaging Profile Server API

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

4.4.5 CSR_BT_BIPS_RC_GET_IMAGE_HEADER

Primitives \ Parameters	type	pHandleInst	connectionId	imageHandleOffset	descriptorLength	descriptorOffset	payloadLength	*payload	imageTotalLength	responseCode	srmpOn
CSR_BT_BIPS_RC_GET_IMAGE_HEADER_IND	✓	✓	✓	✓	✓	✓	✓	✓			
CSR_BT_BIPS_RC_GET_IMAGE_HEADER_RES	✓								✓	✓	✓

Table 23: CSR_BT_BIPS_RC_GET_IMAGE_HEADER Primitives

Description

The indication and response signal is the first part of an operation where the client requests an image file from the BIP server. If the BIP server accepts the request from the client the following messages will be CSR_BT_BIPS_RC_GET_IMAGE_OBJECT indications/responses, see section 4.4.6.

An example of the image descriptor object that can be received from the BIP Remote Camera client is illustrated below:

```
<image-descriptor version="1.0">
<image encoding="JPEG" pixel="1280*960" size="5000000" />
</image-descriptor>
```

Parameters

Type	Signal identity, CSR_BT_BIPS_RC_GET_IMAGE_HEADER_IND/RES.
pHandleInst	This value is used for identifying which BIPS instance the signal is coming from.
connectionId	The connection Id for this session, the BIP client must use this Id in the request.
imageHandleOffset	The handle of the image the client is requesting properties on. The value in this parameter indicates the offset in the payload data where the image handle file data starts. I.e. the data is located at payload[imageHandleOffset]. The image handle is always 7 bytes in length.
descriptorLength	The length of the image descriptor.
descriptorOffset	The client can supply an image descriptor in the indication. This descriptor specifies which format the client wants to receive the image in. The application is responsible for making sure that the image obeys these specifications prior to being transmitted. An empty descriptor is allowed in which case the image is transmitted in its native format. The descriptor is in XML and the specification of the format is found in [BIP]. Furthermore, see example above. The value in this parameter indicates the offset in the payload data where the image handle file data starts. I.e. the data is located at payload[descriptorOffset].
payloadLength	The length of the received payload.
*payload	Pointer to the received payload.

CSR Synergy Bluetooth 18.2.0 OBEX Basic Imaging Profile Server API

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

imageTotalLength	Holds the total length of the image being sent.
responseCode	For accepting the GetImage request the code is CSR_BT_OBEX_SUCCESS_RESPONSE_CODE. The following response codes reject the GetImage request: CSR_BT_OBEX_FORBIDDEN_RESPONSE_CODE CSR_BT_OBEX_BAD_REQUEST_RESPONSE_CODE CSR_BT_OBEX_NOT_ACCEPTABLE_RESPONSE_CODE CSR_BT_OBEX_SERVICE_UNAVAILABLE_RESPONSE_CODE CSR_BT_OBEX_PRECONDITION_FAILED_RESPONSE_CODE If the connectionId is invalid it is recommended that the operation is being rejected with the response code: CSR_BT_OBEX_SERVICE_UNAVAILABLE_RESPONSE_CODE. The responseCodes are defined in (csr_bt_obex.h) with the following type CsrBtObexResponseCode and can also be found in IrDA Object Exchange Protocol. The meaning of the response codes for the Basic Imaging Profile is described in [BIP].
srmpOn	If Single Response Mode is enabled, see section 4.2.1, the BIP server can instruct the BIP Client to wait for the next response packet during a GET Operation by setting the srmpOn parameter TRUE. If used, the srmpOn parameter shall be TRUE in the first GET response, and may be used in consecutive GET response packets to cause the Client to continue its wait; however, once the srmpOn parameter is FALSE in a GET response, the srmpOn parameter are consider to be FALSE for the duration of the operation.

Library Function

A library function is provided by csr_bt_bips_lib.h, and should be used for building and sending the CSR_BT_BIPS_RC_GET_IMAGE_HEADER_RES signal:

```
void CsrBtBipsRcGetImageHeaderResSend(CsrSchedQid           pHandleInst,
                                      CsrBtObexResponseCode responseCode,
                                      CsrUint32             imageTotalLength,
                                      CsrBool               srmpOn);
```

The parameters are as described in the table above.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

4.4.6 CSR_BT_BIPS_RC_GET_IMAGE_OBJECT

Primitives \ Parameters	type	pHandleInst	connectionId	allowedObjectLength	responseCode	imageObjectLength	*imageObject	srmpOn
CSR_BT_BIPS_RC_GET_IMAGE_OBJECT_IND	✓	✓	✓	✓				
CSR_BT_BIPS_RC_GET_IMAGE_OBJECT_RES	✓				✓	✓	✓	✓

Table 24: CSR_BT_BIPS_RC_GET_IMAGE_OBJECT Primitives

Description

This signal is a part of an operation where the client has requested an image from the BIP server. Part of or the entire image is returned.

Parameters

type — Signal identity, CSR_BT_BIPS_RC_GET_IMAGE_OBJECT_IND/RES.

pHandleInst — This value is used for identifying which BIPS instance the signal is coming from.

connectionId — The connection Id for this session, the BIP client must use this Id in the request.

allowedObjectLength — Indicates how much data is maximal allowed to return in the response. This must be obeyed by the application. Please be aware that this number may change from indication to indication.

responseCode — If the image fits in one response or the final part is being transmitted the responseCode must be CSR_BT_OBEX_SUCCESS_RESPONSE_CODE. All intermediate responses must be CSR_BT_OBEX_CONTINUE_RESPONSE_CODE.

The following response codes reject the GetImage request:
CSR_BT_OBEX_FORBIDDEN_RESPONSE_CODE
CSR_BT_OBEX_BAD_REQUEST_RESPONSE_CODE
CSR_BT_OBEX_NOT_ACCEPTABLE_RESPONSE_CODE
CSR_BT_OBEX_SERVICE_UNAVAILABLE_RESPONSE_CODE
CSR_BT_OBEX_PRECONDITION_FAILED_RESPONSE_CODE

If the connectionId is invalid it is recommended that the operation is being rejected with the response code:
CSR_BT_OBEX_SERVICE_UNAVAILABLE_RESPONSE_CODE.

The responseCodes are defined in (csr_bt_obex.h) with the following type CsrBtObexResponseCode and can also be found in IrDA Object Exchange Protocol. The meaning of the response codes for the Basic Imaging Profile is described in [BIP].

imageObjectLength — The length of the object pointed to by *imageObject. This must not be larger that the value given in allowedObjectLength. If data is larger than this it must be fragmented and send in multiple signals.

*imageObject — A pointer to the properties data itself. Must be fragmented if necessary.

srmpOn — If Single Response Mode is enabled, see section 4.2.1, the BIP server can instruct the

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

BIP Client to wait for the next response packet during a GET Operation by setting the srmpOn parameter TRUE.

If used, the srmpOn parameter shall be TRUE in the first GET response, and may be used in consecutive GET response packets to cause the Client to continue its wait; however, once the srmpOn parameter is FALSE in a GET response, the srmpOn parameter are consider to be FALSE for the duration of the operation.

Library Function

A library function is provided by csr_bt_bips_lib.h, and should be used for building and sending the CSR_BT_BIPS_RC_GET_IMAGE_OBJECT_RES signal:

```
void CsrBtBipsRcGetImageObjectResSend(CsrSchedQid           pHandleInst,
                                      CsrUint16             propertiesObjectLength,
                                      CsrUint8              *propertiesObject,
                                      CsrBtObexResponseCode responseCode,
                                      CsrBool               srmpOn);
```

The parameters are as described in the table above.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

4.4.7 CSR_BT_BIPS_RC_GET_LINKED_THUMBNAIL_HEADER

Parameters / Primitives	type	pHandleInst	connectionId	imageHandleOffset	payloadLength	*payload	responseCode	srmpOn
CSR_BT_BIPS_RC_GET_LINKED_THUMBNAIL_HEADER_IND	✓	✓	✓	✓	✓	✓		
CSR_BT_BIPS_RC_GET_LINKED_THUMBNAIL_HEADER_RES	✓						✓	✓

Table 25: CSR_BT_BIPS_RC_GET_LINKED_THUMBNAIL_HEADER Primitives

Description

The indication and response signal is the first part of an operation where the client requests an image file from the BIP server. If the BIP server accepts the request from the client the following messages will be CSR_BT_BIPS_RC_GET_LINKED_THUMBNAIL_OBJECT indications/responses, see section 4.4.8.

Parameters

type — Signal identity, CSR_BT_BIPS_RC_GET_LINKED_THUMBNAIL_HEADER_IND/RES.

pHandleInst — This value is used for identifying which BIPS instance the signal is coming from.

connectionId — The connection Id for this session, the BIP client must use this Id in the request.

imageHandleOffset — The handle of the image the client is requesting properties on.

The value in this parameter indicates the offset in the payload data where the image handle file data starts. I.e. the data is located at payload[imageHandleOffset]. The image handle is always 7 bytes in length.

payloadLength — The length of the received payload.

*payload — Pointer to the received payload.

responseCode — For accepting the GetLinkedThumbnail request the code is CSR_BT_OBEX_SUCCESS_RESPONSE_CODE.

The following response codes reject the GetLinkedThumbnail request:
CSR_BT_OBEX_FORBIDDEN_RESPONSE_CODE
CSR_BT_OBEX_BAD_REQUEST_RESPONSE_CODE
CSR_BT_OBEX_NOT_ACCEPTABLE_RESPONSE_CODE
CSR_BT_OBEX_SERVICE_UNAVAILABLE_RESPONSE_CODE
CSR_BT_OBEX_PRECONDITION_FAILED_RESPONSE_CODE

If the connectionId is invalid it is recommended that the operation is being rejected with the response code:
CSR_BT_OBEX_SERVICE_UNAVAILABLE_RESPONSE_CODE.

The responseCodes are defined in (csr_bt_obex.h) with the following type CsrBtObexResponseCode and can also be found in IrDA Object Exchange Protocol. The meaning of the response codes for the Basic Imaging Profile is described in [BIP].

srmpOn — If Single Response Mode is enabled, see section 4.2.1, the BIP server can instruct the BIP Client to wait for the next response packet during a GET Operation by setting the

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

srmpOn parameter TRUE.

If used, the srmpOn parameter shall be TRUE in the first GET response, and may be used in consecutive GET response packets to cause the Client to continue its wait; however, once the srmpOn parameter is FALSE in a GET response, the srmpOn parameter are consider to be FALSE for the duration of the operation.

Library Function

A library function is provided by csr_bt_bips_lib.h, and should be used for building and sending the CSR_BT_BIPS_RC_GET_LINKED_THUMBNAIL_HEADER_RES signal:

```
void CsrBtBipsRcGetLinkedThumbnailHeaderResSend(CsrSchedQid           pHandleInst,
                                                CsrBtObexResponseCode responseCode,
                                                CsrBool               srmpOn);
```

The parameters are as described in the table above.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

4.4.8 CSR_BT_BIPS_RC_GET_LINKED_THUMBNAIL_OBJECT

Parameters / Primitives	type	pHandleInst	connectionId	allowedObjectLength	responseCode	thumbnailObjectLength	*thumbnailObject	srmpOn
CSR_BT_BIPS_RC_GET_LINKED_THUMBNAIL_OBJECT_IND	✓	✓	✓	✓				
CSR_BT_BIPS_RC_GET_LINKED_THUMBNAIL_OBJECT_RES	✓				✓	✓	✓	✓

Table 26: CSR_BT_BIPS_RC_GET_LINKED_THUMBNAIL_OBJECT Primitives

Description

This signal is a part of an operation where the client has requested a linked thumbnail of an image from the BIP server. Part of or the entire thumbnail is returned.

Parameters

type — Signal identity, CSR_BT_BIPS_RC_GET_LINKED_THUMBNAIL_OBJECT_IND/RES.

pHandleInst — This value is used for identifying which BIPS instance the signal is coming from.

connectionId — The connection Id for this session, the BIP client must use this Id in the request.

allowedObjectLength — Indicates how much data is maximal allowed to return in the response. This must be obeyed by the application. Please be aware that this number may change from indication to indication.

responseCode — If the image fits in one response or the final part is being transmitted the responseCode must be CSR_BT_OBEX_SUCCESS_RESPONSE_CODE. All intermediate responses must be CSR_BT_OBEX_CONTINUE_RESPONSE_CODE.

The following response codes reject the GetLinkedThumbnail request:
CSR_BT_OBEX_FORBIDDEN_RESPONSE_CODE
CSR_BT_OBEX_BAD_REQUEST_RESPONSE_CODE
CSR_BT_OBEX_NOT_ACCEPTABLE_RESPONSE_CODE
CSR_BT_OBEX_SERVICE_UNAVAILABLE_RESPONSE_CODE
CSR_BT_OBEX_PRECONDITION_FAILED_RESPONSE_CODE

If the connectionId is invalid it is recommended that the operation is being rejected with the response code:
CSR_BT_OBEX_SERVICE_UNAVAILABLE_RESPONSE_CODE.

The responseCodes are defined in (csr_bt_obex.h) with the following type CsrBtObexResponseCode and can also be found in IrDA Object Exchange Protocol. The meaning of the response codes for the Basic Imaging Profile is described in [BIP].

thumbnailObjectLength — The length of the object pointed to by *thumbnailObject. This must not be larger that the value given in allowedObjectLength. If data is larger than this it must be fragmented and send in multiple signals.

*thumbnailObject — A pointer to the thumbnail data itself. Must be fragmented if necessary.

srmpOn — If Single Response Mode is enabled, see section 4.2.1, the BIP server can instruct the BIP Client to wait for the next response packet during a GET Operation by setting the

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

srmpOn parameter TRUE.

If used, the srmpOn parameter shall be TRUE in the first GET response, and may be used in consecutive GET response packets to cause the Client to continue its wait; however, once the srmpOn parameter is FALSE in a GET response, the srmpOn parameter are consider to be FALSE for the duration of the operation.

Library Function

A library function is provided by csr_bt_bips_lib.h, and should be used for building and sending the CSR_BT_BIPS_RC_GET_LINKED_THUMBNAIL_OBJECT_RES signal:

```
void CsrBtBipsRcGetLinkedThumbnailObjectResSend(CsrSchedQid pHandleInst,
                                                CsrUint16   thumbnailObjectLength,
                                                CsrUint8    *thumbnailObject,
                                          CsrBtObexResponseCode responseCode,
                                                CsrBool     srmpOn);
```

The parameters are as described in the table above.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

4.5 Automatic Archive Primitives

This section contains description of primitives that are specific to Automatic Archive. It is worth mentioning that the primitives in this section deviate from the primitives in the previous sections. The reason for this is that the BIP server when connected using the Automatic Archive feature is performing like a secondary client. The secondary client is the one to initiate the actions being performed.

4.5.1 CSR_BT_BIPS_AA_GET_IMAGE_LIST & CSR_BT_BIPS_AA_GET_IMAGE_LIST_HEADER

Primitives \ Parameters	type	pHandleInst	nbReturnedHandles	listStartOffset	latestCapturesImages	imageHandlesDescriptorLength	*imageHandlesDescriptor	imageListingObjectOffset	imageListingObjectLength	payloadLength	*payload	responseCode	srmpOn
CSR_BT_BIPS_AA_GET_IMAGE_LIST_REQ	✓		✓	✓	✓	✓	✓						✓
CSR_BT_BIPS_AA_GET_IMAGE_LIST_CFM	✓	✓						✓	✓	✓	✓	✓	
CSR_BT_BIPS_AA_GET_IMAGE_LIST_HEADER_IND	✓	✓	✓			✓	✓						
CSR_BT_BIPS_AA_GET_IMAGE_LIST_HEADER_RES	✓												✓
CSR_BT_BIPS_AA_GET_IMAGE_LIST_IND	✓	✓						✓	✓	✓	✓		
CSR_BT_BIPS_AA_GET_IMAGE_LIST_RES	✓												✓

Table 27: CSR_BT_BIPS_AA_GET_IMAGE_LIST & CSR_BT_BIPS_AA_GET_IMAGE_LIST_HEADER Primitives

Description

A typical first operation when performing Automatic Archive is to learn what images are on the BIP client. This is done with the CSR_BT_BIPS_AA_GET_IMAGE_LIST and CSR_BT_BIPS_AA_GET_IMAGE_LIST_HEADER. The operation initiated by the BIP server sending CSR_BT_BIPS_AA_GET_IMAGE_LIST_REQ and completed when CSR_BT_BIPS_AA_GET_IMAGE_LIST_CFM is received. In between these two signals the server will first receive a single CSR_BT_BIPS_AA_GET_IMAGE_LIST_HEADER_IND containing information about the image list about to be received. Following this comes one or multiple CSR_BT_BIPS_AA_GET_IMAGE_LIST_IND containing the image list itself.

An example of the image handles descriptor object that can be sent to and received from the BIP Automatic Archive client is illustrated below:

```
<image-handles-descriptor version="1.0">
<filtering-parameters created="20070101T000000Z-20070101T235959Z" />
</image-handles-descriptor>
```

An example of the image listing object that can be sent to and received from the BIP Automatic Archive client is illustrated below:

```
<images-listing version="1.0">
<image handle="1000001" created="20070801T060000Z" />
```


© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

```
<image handle=”1000003” created=”20070801T060115Z”
modified=”20070808T101500Z” />
<image handle=”1000004” created=”20070801T080137Z” />
</images-listing>
```

Parameters

type	Signal identity, CSR_BT_BIPS_AA_GET_IMAGE_LIST_REQ/CFM/IND/RES and CSR_BT_BIPS_AA_GET_IMAGE_LIST_HEADER_IND/RES.
pHandleInst	This value is used for identifying which BIPS instance the signal is coming from.
nbReturnedHandles	This value is in the request used for specifying how many items should be returned in the image list. In the header indication it is used for indicating how many items are actually in the image list.
listStartOffset	This value is used for specifying what offset in the image list on the client to start the image list to be returned to the server. E.g. a previous operation could have retrieved the first 10 image list elements. In the second operation this parameter would then be set to 11.
latestCapturesImages	This boolean is used for indicating to the client if the image list should be sorted descending in respect to capture date.
imageHandlesDescriptorLength	The length of the image handles descriptor pointed to by *imageHandlesDescriptor.
*imageHandlesDescriptor	A pointer to an image handles descriptor. In the request this descriptor represents the criteria the server wishes the client applies to the image list before sending it. The descriptor received in the indication is the criteria actually applied by the client. Ideally the two descriptors are identical. Please be aware that the length of this descriptor must not exceed the maximally allowed OBEX packet size. An example of an image handles descriptor is seen above. Detailed information is found in [BIP].
imageListingObjectOffset	The image list that describes the images on the BIP Automatic Archive server. The value in this parameter indicates the offset in the payload data where the image list object data starts. I.e. the data is located at payload[imageListingObjectOffset].
imageListingObjectLength	The length of the image list object.
payloadLength	The length of the received payload.
*payload	Pointer to the received payload.
responseCode	A successful response is CSR_BT_OBEX_SUCCESS_RESPONSE_CODE. Any other response code indicates a failure. The responseCodes are defined in (csr_bt_obex.h) with the following type CsrBtObexResponseCode and can also be found in IrDA Object Exchange Protocol. The meaning of the response codes for the Basic Imaging Profile is described in [BIP].
srmpOn	If Single Response Mode is enabled, see section 4.2.1, BIPS secondary client can instruct BIPC secondary server to wait for the next request packet during a GET Operation by setting the srmpOn parameter TRUE. If used, the srmpOn parameter shall be TRUE in the first GET request, and may be used in consecutive GET request packets to cause BIPC secondary server to continue its wait; however, once the srmpOn parameter is FALSE in a GET request, the srmpOn parameter are consider to be FALSE for the duration of the operation.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR’s non-disclosure agreement.

Library Function

Library functions are provided by csr_bt_bips_lib.h, and should be used for building and sending the CSR_BT_BIPS_AA_GET_IMAGE_LIST_REQ, CSR_BT_BIPS_AA_GET_IMAGE_LIST_HEADER_RES and CSR_BT_BIPS_AA_GET_IMAGE_LIST_RES signals:

```
void CsrBtBipsAaGetImageListReqSend(CsrSchedQid pHandleInst,
                                    CsrUint16   nbReturnedHandles,
                                    CsrUint16   listStartOffset,
                                    CsrBool     latestCapturedImages,
                                    CsrUint16   imageHandlesDescriptorLength,
                                    CsrUint8    *imageHandlesDescriptor,
                                    CsrBool     srmpOn);

void CsrBtBipsAaGetImageListHeaderResSend(CsrSchedQid pHandleInst,
                                          CsrBool     srmpOn);

void CsrBtBipsAaGetImageListResSend(CsrSchedQid pHandleInst,
                                    CsrBool     srmpOn);
```

The parameters are as described in the table above.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

4.5.2 CSR_BT_BIPS_AA_GET_CAPABILITIES

Parameters / Primitives	type	pHandleInst	responseCode	capabilitiesObjectLength	capabilitiesObjectOffset	payloadLength	*payload	srmpOn
CSR_BT_BIPS_AA_GET_CAPABILITIES_REQ	✓							✓
CSR_BT_BIPS_AA_GET_CAPABILITIES_RES	✓							✓
CSR_BT_BIPS_AA_GET_CAPABILITIES_IND	✓	✓		✓	✓	✓	✓	
CSR_BT_BIPS_AA_GET_CAPABILITIES_CFM	✓	✓	✓	✓	✓	✓	✓	

Table 28: CSR_BT_BIPS_AA_GET_CAPABILITIES Primitives

Description

The CSR_BT_BIPS_AA_GET_CAPABILITIES_REQ is used for retrieving the imaging-capabilities object from the Automatic Archive client.

Parameters

type	Signal identity, CSR_BT_BIPS_AA_GET_CAPABILITIES_REQ/RES/IND/CFM.
pHandleInst	This value is used for identifying which BIPS instance the signal is coming from.
responseCode	A successful response is CSR_BT_OBEX_SUCCESS_RESPONSE_CODE. Any other response code indicates a failure The responseCodes are defined in (csr_bt_obex.h) with the following type CsrBtObexResponseCode and can also be found in IrDA Object Exchange Protocol. The meaning of the response codes for the Basic Imaging Profile is described in [BIP].
capabilitiesObjectLength	The length of the Get Capabilities object.
capabilitiesObjectOffset	Offset (relative to payload) of the imaging-capabilities object or part of it (i.e. in case of a multi packet operation). An example of a capabilities object is found in section 4.3.2. Detailed information is found in [BIP]. The value in this parameter indicates the offset in the payload data where the image list object data starts. I.e. the data is located at payload[capabilitiesObjectOffset].
payloadLength	Number of bytes in the payload.
*payload	OBEX payload data. Offsets are relative to this pointer.
srmpOn	If Single Response Mode is enabled, see section 4.2.1, BIPS secondary client can instruct BIPC secondary server to wait for the next request packet during a GET Operation by setting the srmpOn parameter TRUE. If used, the srmpOn parameter shall be TRUE in the first GET request, and may be used in consecutive GET request packets to cause BIPC secondary server to

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

continue its wait; however, once the srmpOn parameter is FALSE in a GET request, the srmpOn parameter are consider to be FALSE for the duration of the operation.

Library Function

Library functions are provided by csr_bt_bips_lib.h, and should be used for building and sending the CSR_BT_BIPS_AA_GET_CAPABILITIES_REQ and CSR_BT_BIPS_AA_GET_CAPABILITIES_RES signals:

```
void CsrBtBipsAaGetCapabilitiesReqSend(CsrSchedQid pHandleInst,
                                       CsrBool     srmpOn);

void CsrBtBipsAaGetCapabilitiesResSend(CsrSchedQid pHandleInst,
                                       CsrBool     srmpOn);
```

The parameter is as described in the table above.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

4.5.3 CSR_BT_BIPS_AA_GET_IMAGE_PROPERTIES

Primitives \ Parameters	type	imageHandle	pHandleInst	propertiesObjectOffset	propertiesObjectLength	payloadLength	*payload	responseCode	srmpOn
CSR_BT_BIPS_AA_GET_IMAGE_PROPERTIES_REQ	✓	✓							✓
CSR_BT_BIPS_AA_GET_IMAGE_PROPERTIES_IND	✓		✓	✓	✓	✓	✓		
CSR_BT_BIPS_AA_GET_IMAGE_PROPERTIES_RES	✓								✓
CSR_BT_BIPS_AA_GET_IMAGE_PROPERTIES_CFM	✓		✓	✓	✓	✓	✓	✓	

Table 29: CSR_BT_BIPS_AA_GET_IMAGE_PROPERTIES Primitives

Description

The CSR_BT_BIPS_AA_GET_IMAGE_PROPERTIES_REQ is used for retrieving the image-properties object from the BIP Automatic Archive client.

Parameters

Type	Signal identity, CSR_BT_BIPS_AA_GET_IMAGE_PROPERTIES_REQ/IND/RES/CFM.
imageHandle	The imagehandle is used for identifying the relevant image on the client.
pHandleInst	This value is used for identifying which BIPS instance the signal is coming from.
propertiesObjectOffset	Offset (relative to payload) of the properties object or part of it (i.e., in case of a multi packet operation). An example of a capabilities object is found in section 4.4.4. Detailed information is found in [BIP]. The value in this parameter indicates the offset in the payload data where the image list object data starts. I.e. the data is located at payload[propertiesObjectOffset].
propertiesObjectLength	The length of the properties object.
payloadLength	Number of bytes in the payload.
*payload	OBEX payload data. Offsets are relative to this pointer.
responseCode	A successful response is CSR_BT_OBEX_SUCCESS_RESPONSE_CODE. Any other response code indicates a failure. The responseCodes are defined in (csr_bt_obex.h) with the following type CsrBtObexResponseCode and can also be found in IrDA Object Exchange Protocol. The meaning of the response codes for the Basic Imaging Profile is described in [BIP].
srmpOn	If Single Response Mode is enabled, see section 4.2.1, BIPS secondary client can instruct BIPC secondary server to wait for the next request packet during a

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

GET Operation by setting the srmpOn parameter TRUE.

If used, the srmpOn parameter shall be TRUE in the first GET request, and may be used in consecutive GET request packets to cause BIPC secondary server to continue its wait; however, once the srmpOn parameter is FALSE in a GET request, the srmpOn parameter are consider to be FALSE for the duration of the operation.

Library Function

Library functions are provided by csr_bt_bips_lib.h, and should be used for building and sending the CSR_BT_BIPS_AA_GET_IMAGE_PROPERTIES_REQ and CSR_BT_BIPS_AA_GET_IMAGE_PROPERTIES_RES signals:

```
void CsrBtBipsAaGetImagePropertiesReqSend(CsrSchedQid         pHandleInst,
                                          const CsrCharString *imageHandle,
                                          CsrBool             srmpOn);

void CsrBtBipsAaGetImagePropertiesResSend(CsrSchedQid pHandleInst);

void CsrBtBipsAaGetImagePropertiesExtResSend(CsrSchedQid pHandleInst,
                                             CsrBool       srmpOn);
```

The parameters are as described in the table above. Note the function CsrBtBipsAaGetImagePropertiesResSend always set srmpOn to FALSE.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

4.5.4 CSR_BT_BIPS_AA_GET_IMAGE

Parameters / Primitives	type	imageHandle	imageDestiptorLength	*imageDescriptor	pHandleInst	imageObjectOffset	imageObjectLength	payloadLength	*payload	responseCode	smpOn
CSR_BT_BIPS_AA_GET_IMAGE_REQ	✓	✓	✓	✓							✓
CSR_BT_BIPS_AA_GET_IMAGE_IND	✓				✓	✓	✓	✓	✓		
CSR_BT_BIPS_AA_GET_IMAGE_RES	✓										✓
CSR_BT_BIPS_AA_GET_IMAGE_CFM	✓				✓	✓	✓	✓	✓	✓	

Table 30: CSR_BT_BIPS_AA_GET_IMAGE Primitives

Description

The CSR_BT_BIPS_AA_GET_IMAGE_REQ is used for retrieving the image from the BIP Automatic Archive client. The format of the requested image is supplied in the image-descriptor object of the request.

Parameters

type	Signal identity, CSR_BT_BIPS_AA_GET_IMAGE_REQ/IND/RES/CFM
imageHandle	The imageHandle is used for identifying the relevant image on the client.
imageDescriptorLength	The length of the imageDescriptor. The function “returnImgDescriptionLength“ can be used for returning the imageDescriptorLength.
*imageDescriptor	The image descriptor describes the properties of the requested image. For a description of the definition for the image descriptor, together with the elements, and the attributes used in the image descriptor, please refer to [BIP]. The function “buildImgDescriptorHeader” can be used for building the imageDescriptor. An example of a capabilities object is found in section 4.4.5. Detailed information is found in [BIP].
pHandleInst	This value is used for identifying which BIPS instance the signal is coming from.
imageObjectOffset	The length of the image object. The value in this parameter indicates the offset in the payload data where the image list object data starts. I.e. the data is located at payload[imageObjectOffset].
imageObjectLength	Offset (relative to payload) of the image object or part of it (i.e., in case of a multi packet operation).
payloadLength	Number of bytes in the payload.
*payload	OBEX payload data. Offsets are relative to this pointer.
responseCode	A successful response is CSR_BT_OBEX_SUCCESS_RESPONSE_CODE. Any other response code indicates a failure.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR’s non-disclosure agreement.

	The responseCodes are defined in (csr_bt_obex.h) with the following type CsrBtObexResponseCode and can also be found in IrDA Object Exchange Protocol. The meaning of the response codes for the Basic Imaging Profile is described in [BIP].
srmpOn	If Single Response Mode is enabled, see section 4.2.1, BIPS secondary client can instruct BIPC secondary server to wait for the next request packet during a GET Operation by setting the srmpOn parameter TRUE. If used, the srmpOn parameter shall be TRUE in the first GET request, and may be used in consecutive GET request packets to cause BIPC secondary server to continue its wait; however, once the srmpOn parameter is FALSE in a GET request, the srmpOn parameter are consider to be FALSE for the duration of the operation.

Library Function

Library functions are provided by csr_bt_bips_lib.h, and should be used for building and sending the CSR_BT_BIPS_AA_GET_IMAGE_REQ and CSR_BT_BIPS_AA_GET_IMAGE_RES signals:

```
void CsrBtBipsAaGetImageReqSend(CsrSchedQid    pHandleInst,
                                const CsrUint8 *imageHandle,
                                CsrUint16      imageDescriptorLength,
                                CsrUint8       *imageDescriptor,
                                CsrBool        srmpOn);

void CsrBtBipsAaGetImageResSend(CsrSchedQid pHandleInst,
                                CsrBool     srmpOn);
```

The parameters are as described in the table above.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

4.5.5 CSR_BT_BIPS_AA_GET_LINKED_ATTACHMENT

Parameters / Primitives	type	imageHandle	pHandleInst	fileNameLength	*fileName	attachmentFileOffset	attachmentFileLength	payloadLength	*payload	responseCode	srmpOn
CSR_BT_BIPS_AA_GET_LINKED_ATTACHMENT_REQ	✓	✓		✓	✓						✓
CSR_BT_BIPS_AA_GET_LINKED_ATTACHMENT_IND	✓		✓			✓	✓	✓	✓		
CSR_BT_BIPS_AA_GET_LINKED_ATTACHMENT_RES	✓										✓
CSR_BT_BIPS_AA_GET_LINKED_ATTACHMENT_CFM	✓		✓			✓	✓	✓	✓	✓	

Table 31: CSR_BT_BIPS_AA_GET_LINKED_ATTACHMENT Primitives

Description

The CSR_BT_BIPS_AA_GET_LINKED_ATTACHMENT_REQ is used for retrieving a linked attachment associated with an image from the BIP Automatic Archive client.

Parameters

Type	Signal identity, CSR_BT_BIPS_AA_GET_LINKED_ATTACHMENT_REQ/IND/RES/CFM.
imageHandle	The imageHandle is used for identifying the relevant image on the client.
pHandleInst	This value is used for identifying which BIPS instance the signal is coming from.
fileNameLength	The length of the filename of the attachment
*fileName	Pointer to filename of the attachment
attachmentFileOffset	Offset (relative to payload) of the image object or part of it (i.e., in case of a multi packet operation). The value in this parameter indicates the offset in the payload data where the image list object data starts. I.e. the data is located at payload[thumbnailObjectOffset].
attachmentFileLength	The length of the linked attachment object.
payloadLength	Number of bytes in the payload.
*payload	OBEX payload data. Offsets are relative to this pointer.
responseCode	A successful response is CSR_BT_OBEX_SUCCESS_RESPONSE_CODE. Any other response code indicates a failure. The responseCodes are defined in (csr_bt_obex.h) with the following type CsrBtObexResponseCode and can also be found in IrDA Object Exchange Protocol. The meaning of the response codes for the Basic Imaging Profile is described in [BIP].
srmpOn	If Single Response Mode is enabled, see section 4.2.1, BIPS secondary client can

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

instruct BIPC secondary server to wait for the next request packet during a GET Operation by setting the srmpOn parameter TRUE.

If used, the srmpOn parameter shall be TRUE in the first GET request, and may be used in consecutive GET request packets to cause BIPC secondary server to continue its wait; however, once the srmpOn parameter is FALSE in a GET request, the srmpOn parameter are consider to be FALSE for the duration of the operation.

Library Function

Library functions are provided by csr_bt_bips_lib.h, and should be used for building and sending the CSR_BT_BIPS_AA_GET_LINKED_ATTACHMENT_REQ and CSR_BT_BIPS_AA_GET_LINKED_ATTACHMENT_RES signals:

```
void CsrBtBipsAaGetLinkedAttachmentReqSend(CsrSchedQid     pHandleInst,
                                           const CsrUint8  *imageHandle,
                                           CsrUint16       fileNameLength,
                                           CsrUint8        *filename,
                                           CsrBool         srmpOn);

void CsrBtBipsAaGetLinkedAttachmentResSend(CsrSchedQid pHandleInst,
                                           CsrBool     srmpOn);
```

The parameters are as described in the table above.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

4.5.6 CSR_BT_BIPS_AA_GET_LINKED_THUMBNAIL

Parameters / Primitives	type	imageHandle	pHandleInst	thumbnailObjectOffset	thumbnailObjectLength	payloadLength	*payload	responseCode	srmpOn
CSR_BT_BIPS_AA_GET_LINKED_THUMBNAIL_REQ	✓	✓							✓
CSR_BT_BIPS_AA_GET_LINKED_THUMBNAIL_IND	✓		✓	✓	✓	✓	✓		
CSR_BT_BIPS_AA_GET_LINKED_THUMBNAIL_RES	✓								✓
CSR_BT_BIPS_AA_GET_LINKED_THUMBNAIL_CFM	✓		✓	✓	✓	✓	✓	✓	

Table 32: CSR_BT_BIPS_AA_GET_LINKED_THUMBNAIL Primitives

Description

The CSR_BT_BIPS_AA_GET_LINKED_THUMBNAIL_REQ is used for retrieving the thumbnail of an image from the BIP Automatic Archive client.

Parameters

type	Signal identity, CSR_BT_BIPS_AA_GET_LINKED_THUMBNAIL_REQ/IND/RES/CFM
imageHandle	The imageHandle is used for identifying the relevant image on the client.
pHandleInst	This value is used for identifying which BIPS instance the signal is coming from.
thumbnailObjectLength	The length of the image object,
thumbnailObjectOffset	Offset (relative to payload) of the image object or part of it (i.e., in case of a multi packet operation). The value in this parameter indicates the offset in the payload data where the image list object data starts. I.e. the data is located at payload[thumbnailObjectOffset].
payloadLength	Number of bytes in the payload
*payload	OBEX payload data. Offsets are relative to this pointer.
responseCode	A successful response is CSR_BT_OBEX_SUCCESS_RESPONSE_CODE. Any other response code indicates a failure. The responseCodes are defined in (csr_bt_obex.h) with the following type CsrBtObexResponseCode and can also be found in IrDA Object Exchange Protocol. The meaning of the response codes for the Basic Imaging Profile is described in [BIP].
srmpOn	If Single Response Mode is enabled, see section 4.2.1, BIPS secondary client can instruct BIPC secondary server to wait for the next request packet during a GET Operation by setting the srmpOn parameter TRUE. If used, the srmpOn parameter shall be TRUE in the first GET request, and may be used in consecutive GET request packets to cause BIPC secondary server to

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

continue its wait; however, once the srmpOn parameter is FALSE in a GET request, the srmpOn parameter are consider to be FALSE for the duration of the operation.

Library Function

Library functions are provided by csr_bt_bips_lib.h, and should be used for building and sending the CSR_BT_BIPS_AA_GET_LINKED_THUMBNAIL_REQ and CSR_BT_BIPS_AA_GET_LINKED_THUMBNAIL_RES signals:

```
void CsrBtBipsAaGetLinkedThumbnailReqSend(CsrSchedQid     pHandleInst,
                                          const CsrUint8  *imageHandle,
                                          CsrBool         srmpOn);

void CsrBtBipsAaGetLinkedThumbnailResSend(CsrSchedQid pHandleInst,
                                          CsrBool     srmpOn);
```

The parameters are as described in the table above.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

4.5.7 CSR_BT_BIPS_AA_DELETE_IMAGE

Parameters / Primitives	type	imageHandle	pHandleInst	responseCode
CSR_BT_BIPS_AA_DELETE_IMAGE_REQ	✓	✓		
CSR_BT_BIPS_AA_DELETE_IMAGE_CFM	✓		✓	✓

Table 33: CSR_BT_BIPS_AA_DELETE_IMAGE Primitives

Description

The CSR_BT_BIPS_AA_DELETE_IMAGE_REQ is used for deleting an image on the BIP Automatic Archive client.

Parameters

type	Signal identity, CSR_BT_BIPS_AA_DELETE_IMAGE_REQ/CFM.
imageHandle	The imageHandle is used for identifying the relevant image on the client.
pHandleInst	This value is used for identifying which BIPS instance the signal is coming from.
responseCode	A successful response is CSR_BT_OBEX_SUCCESS_RESPONSE_CODE. Any other response code indicates a failure. The responseCodes are defined in (csr_bt_obex.h) with the following type CsrBtObexResponseCode and can also be found in IrDA Object Exchange Protocol. The meaning of the response codes for the Basic Imaging Profile is described in [BIP].

Library Function

A library function is provided by csr_bt_bips_lib.h, and should be used for building and sending the CSR_BT_BIPS_AA_DELETE_IMAGE_REQ signal:

```
void CsrBtBipsAaDeleteImageReqSend(CsrSchedQid    pHandleInst,
                                   const CsrUint8 *imageHandle);
```

The parameters are as described in the table above.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

4.5.8 CSR_BT_BIPS_AA_ABORT

Parameters / Primitives	type	pHandleInst
CSR_BT_BIPS_AA_ABORT_REQ	✓	
CSR_BT_BIPS_AA_ABORT_CFM	✓	✓

Table 34: CSR_BT_BIPS_AA_ABORT Primitives

Description

The CSR_BT_BIPS_AA_ABORT_REQ is used for aborting an ongoing operation with the BIP Automatic Archive client.

Parameters

type	Signal identity, CSR_BT_BIPS_AA_ABORT_REQ/CFM.
pHandleInst	This value is used for identifying which BIPS instance the signal is coming from.

Library Function

A library function is provided by csr_bt_bips_lib.h, and should be used for building and sending the CSR_BT_BIPS_AA_ABORT_REQ signal:

```
void CsrBtBipsAaAbortReqSend(CsrSchedQid pHandleInst);
```

The parameter is as described in the table above.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

5 Document References

Document	Reference
Basic Imaging Profile Revision V11r00 28 August 2010	[BIP]
GENERIC OBJECT EXCHANGE PROFILE Revision V20r00 26 August 2010	[GOEP2.0]
IrDA Object Exchange Protocol – IrOBEX Version 1.2 or Version 1.5	[OBEX]
CSR Synergy Bluetooth. CM – Connection Manager API Description, doc. no. api-0101-cm	[CM]
CSR Synergy Bluetooth, SC – Security Controller API Description, Document no. api-0102-sc	[SC]
CSR Synergy Bluetooth, AMPM – Alternate MAC and PHY Manager API Description, api-0148-ampm.pdf	[AMPM]

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

Terms and Definitions

BlueCore®	Group term for CSR's range of Bluetooth wireless technology chips
Bluetooth®	Set of technologies providing audio and data transfer over short-range radio connections
BIPS	OBEX Basic Imaging Profile (server)
CSR	Cambridge Silicon Radio
SDS	Service Discovery Server
SIG	Special Interest Group
UniFi™	Group term for CSR's range of chips designed to meet IEEE 802.11 standards
SRM	Single Response Mode
SRMP	Single Response Mode Parameters
GOEP	Generic Object Exchange Profile
AMPM	Alternate MAC and PHY Manager

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

Document History

Revision	Date	History
1	26 SEP 11	Ready for release 18.2.0

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.

TradeMarks, Patents and Licences

Unless otherwise stated, words and logos marked with ™ or ® are trademarks registered or owned by CSR plc or its affiliates. Bluetooth® and the Bluetooth logos are trademarks owned by Bluetooth SIG, Inc. and licensed to CSR. Other products, services and names used in this document may have been trademarked by their respective owners.

The publication of this information does not imply that any licence is granted under any patent or other rights owned by CSR plc.

CSR reserves the right to make technical changes to its products as part of its development programme.

While every care has been taken to ensure the accuracy of the contents of this document, CSR cannot accept responsibility for any errors.

Life Support Policy and Use in Safety-critical Compliance

CSR's products are not authorised for use in life-support or safety-critical applications. Use in such applications is done at the sole discretion of the customer. CSR will not warrant the use of its devices in such applications.

Performance and Conformance

Refer to www.csrsupport.com for compliance and conformance to standards information.

© Cambridge Silicon Radio Limited 2011
This material is subject to CSR's non-disclosure agreement.