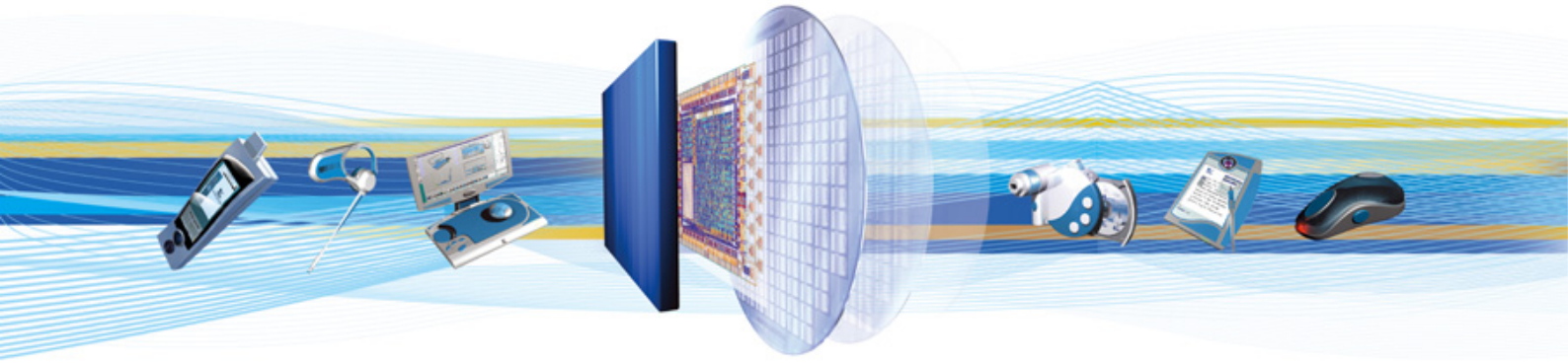
CSR Synergy Bluetooth 18.2.0

BNEP Service Layer (BSL)

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

Contents

List of Figures

List of Tables

1 Introduction

1.1 Overview

This document describes the functionality and message interface provided by the BNEP Service Layer (BSL) profile manager in CSR Synergy Bluetooth (BlueCore® Host Software). BNEP is the communication protocol used by the PAN profile.

The PAN profile is described in the Bluetooth profile specification 'Personal Area Networking Profile' [PAN] and describes how Bluetooth can be used for providing networking capabilities by encapsulating layer three network protocols. Layer three encapsulation is provided by utilising the Bluetooth Network Encapsulation Protocol, [BNEP].

Basically there are two elements to implementing an application complying with the PAN profile:

- a BNEP service, which is responsible for dealing with BNEP connections
- a protocol service, which is responsible for dealing with higher layer protocol exchange. For PAN services this is typically IP based services but can be virtually any higher layer protocol

The CSR Synergy Bluetooth profile manager for PAN services implements the BNEP service, thus leaving the implementation and handling of any network service to the application layer above CSR Synergy Bluetooth, see also section 2.2.

The PAN profile defines three scenarios:

1. Network Access Points (NAP)
2. Group ad-hoc Networking (GN)
3. Pan User to Pan User connection (PANU)

In the network access point scenario (1) a network access point acts as a bridge between a Bluetooth network and some other technology. In the Group ad-hoc networking scenario (2) a number of Bluetooth devices can create ad-hoc wireless networks where one device operates as the piconet master and a number of other devices can be connected to the master to form a wireless network without any additional network infrastructure. The pan user to pan user scenario (3) is a point to point connection between two pan users; this is typically used for simulating a cross-over cable network connection.

Please note that the PAN profile requires at least one of the two directly connected devices to be a PANU.

NOTICE: Currently, the PAN profile cannot be qualified with the CSR Synergy Bluetooth RFCOMM solution due to memory limitations. The PAN profile works with standard TCP/IP protocols when the IP datagram size is limited to the recommended 576 bytes (RFC 791), but for qualification purposes the profile must support 1500 bytes packets (BNEP MTU size).

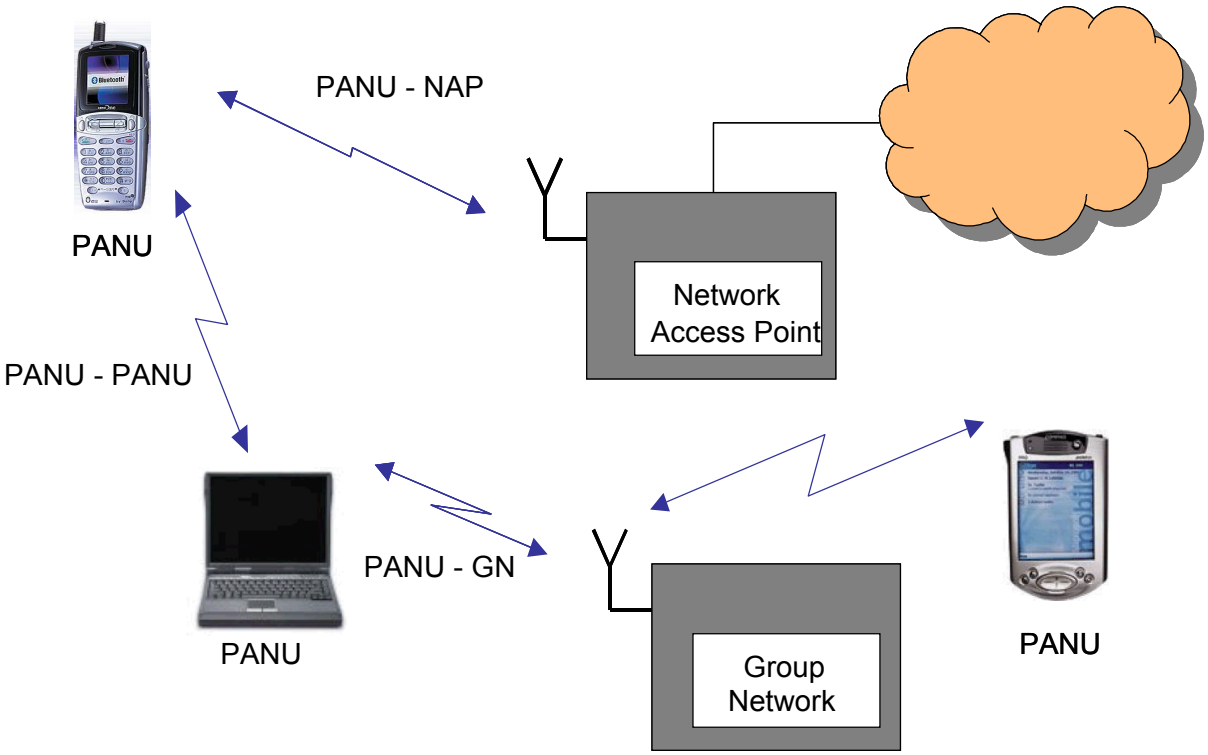


Figure 1: PAN profile roles and system configuration

The physical devices in the different scenarios may differ. Typical devices in the GN scenario are a number of notebooks forming an ad-hoc network, for instance to exchange files during a meeting. A network access point can be a dedicated access point connected to some kind of fixed network technology, e.g. an Ethernet but could also be a mobile phone providing network access through GSM or GPRS.

2 Description

2.1 Introduction

The PAN profile defines several scenarios and roles for devices using the PAN profile, see section 1.1. In CSR Synergy Bluetooth the application layer interface for PAN services is through the BNEP Service Layer (BSL) profile manager. The BSL profile manager is able to support the BNEP related services of all roles and scenarios defined in [PAN]. For further details about the BNEP Protocol, see [BNEP].

In short using the BSL API the following functionality is supported:

- Establishment and disconnection of a BNEP connection between a pair of PAN devices complying with one of the roles defined in the PAN profile (PANU, GN or NAP)
- Forwarding and transmission of Ethernet data packets
- Filtering and routing of packet types to run time defined tasks
- Automatic and configurable use of low power mode sniff

The BSL layer handles proper registration/initialisation of the following:

- enable/disable inquiry and page scan
- registration/deregistration of service discovery record in the service discovery database
- enable/disable class of device for peer devices doing discovery to discover a device with the network bit set in the class of device field
- registration of security information in the security manager

2.2 Reference Model

A typical application layer must interface to the BSL, Security Controller (SC) and Connection Manager (CM) though it is not a strict requirement to use the SC and CM. The Security Controller has an interface that is used for bonding and pairing - please refer to the relevant CSR Synergy Bluetooth API documents for more information.

The BSL profile manager provides flexibility to the layer utilizing the services; thus it is possible to use legacy protocol stacks if such exist. Depending on the lower interface of the protocol stack it may be necessary to add a small driver layer in between the protocol layer and the BSL layer.

In the current version of CSR Synergy Bluetooth only the BSL layer is supplied. This implies that the network protocol layers must be provided as part of the application, e.g. as a legacy from existing network applications.

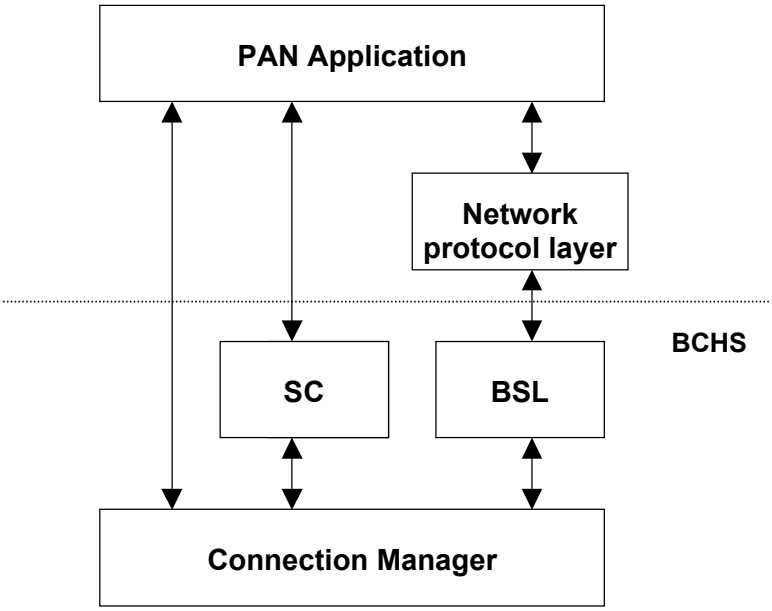


Figure 2: CSR Synergy Bluetooth reference model

In a typical scenario the Connection Manager and Security Controller functions are carried out before any other transaction. E.g. bonding is completed before setting up any transactions using the BSL interface as described below.

3 Interface Description

3.1 Registration

Before the BSL and PAN services can be used, a CSR_BT_BSL_REGISTER_REQ signal must be sent to the BSL layer; no other signal must be sent before the register signal. The register signal must include two handles:

- one used for all connection related signalling, e.g. connect indications etc.
- a handle where all data is sent to

where a handle defines a scheduler instance/component. This approach allows the network protocol part to be divided from the management part; the handles can be identical, but must be specified as two separate handles in the signal.

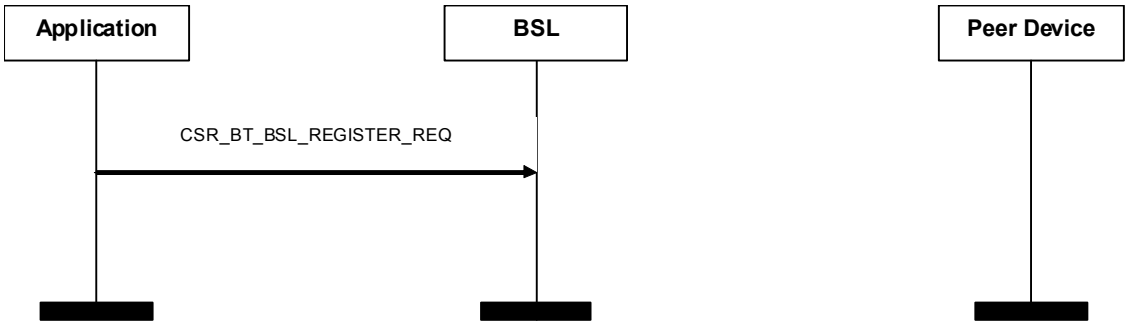


Figure 3: Registration sequence

All addresses included in the signals for the management part are specified as Bluetooth addresses (see the define in file csr_bt_bluetooth.h). All addresses in data related signals are specified as Ethernet addresses, i.e. as an array of 3 uint 16. Bluetooth addresses are valid Ethernet addresses. The Bluetooth address 0x00025b123456 corresponds to the Ethernet address 00-02-5b-12-34-56 and is represented as the Ethernet address 0x0002, 0x5b12, 0x3456.

The register signal will take care of internal initialisation and security registration in the CSR Synergy Bluetooth stack. The register signal is not confirmed, but any signal received in BSL before registration is completed, is saved and is restored upon completion of the registration process.

3.2 Service Activation (Incoming Connections)

An application can make itself connectable for other devices by issuing the activation primitive.

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.

CSR_BT_BSL_ACTIVATE_REQ. The CSR_BT_BSL_ACTIVATE_REQ defines the roles that are allowed; see section 1.1 for an introduction of defined roles. The activate signal must include at least one local and one remote role. Further, at least one of the remote or local roles must be PANU. The BSL layer will automatically disallow any incoming connections with invalid role combinations; possible roles are defined in the file csr_bt_bsl_prim.h.

The allowed roles can at any time be changed by issuing another CSR_BT_BSL_ACTIVATE_REQ with the wanted role combinations. **Please note** that due to race conditions the application must be prepared to receive a connect indication for a previous role combination as long as the activate request is not confirmed.

The application must specify whether one or more remote users are allowed simultaneously; this is done using the single user flag in the CSR_BT_BSL_ACTIVATE_REQ signal. In case multiple users are allowed, this is only

relevant for the GN and NAP roles, the BSL layer will automatically, and in accordance with the PAN profile, perform a master slave role switch in order for the GN or NAP to become the master of the connection.

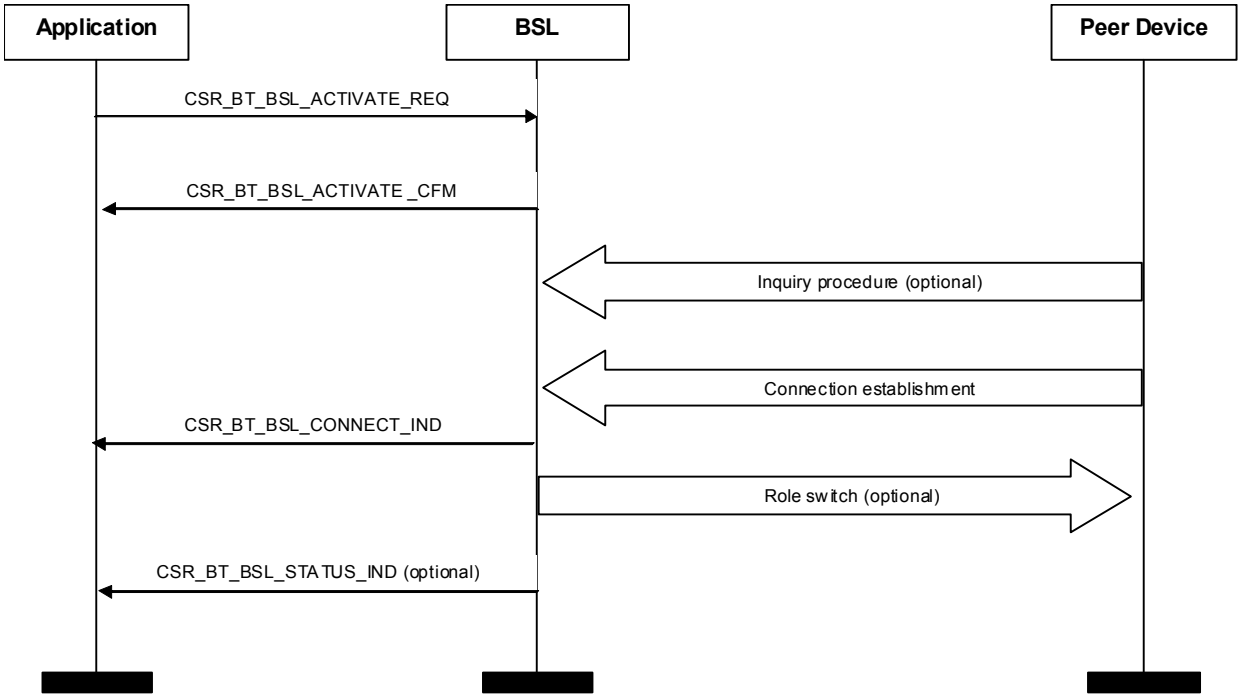


Figure 4: Activate sequence

The activation is confirmed once accepted by BSL. The confirmation only indicates that the local device is now activated (connectable) and ready to receive a connection from a peer device. When the connection is established this is indicated through a CSR_BT_BSL_CONNECT_IND. The CSR_BT_BSL_CONNECT_IND includes the local and remote role for this connection.

To deactivate the service completely simply un-define all roles and issue a CSR_BT_BSL_ACTIVATE_REQ. When confirmed the PAN profile will no longer be connectable from remote devices and the local device will exit from connectable mode unless other profiles in CSR Synergy Bluetooth are activated. Already existing connections are not affected by the de-activation of the service and each connection must be disconnected following the procedure outlined in section 3.6.

If a CSR_BT_BSL_CONNECT_REQ is sent after the service is opened for incoming connections another CSR_BT_BSL_ACTIVATE_REQ must sent to reactivate the service for incoming connections again.

3.3 Remote Device Connection (Outgoing Connection)

In order to setup a connection towards a peer device the CSR_BT_BSL_CONNECT_REQ must be used. This primitive must specify the remote device address and the required roles, both local and remote; exactly one role must be defined for both.

The connection is confirmed with the CSR_BT_BSL_CONNECT_IND. If for any reason the connection fails the result field indicates the reason of failure; in case of failure only the result field is valid and the remaining parameters must be ignored.

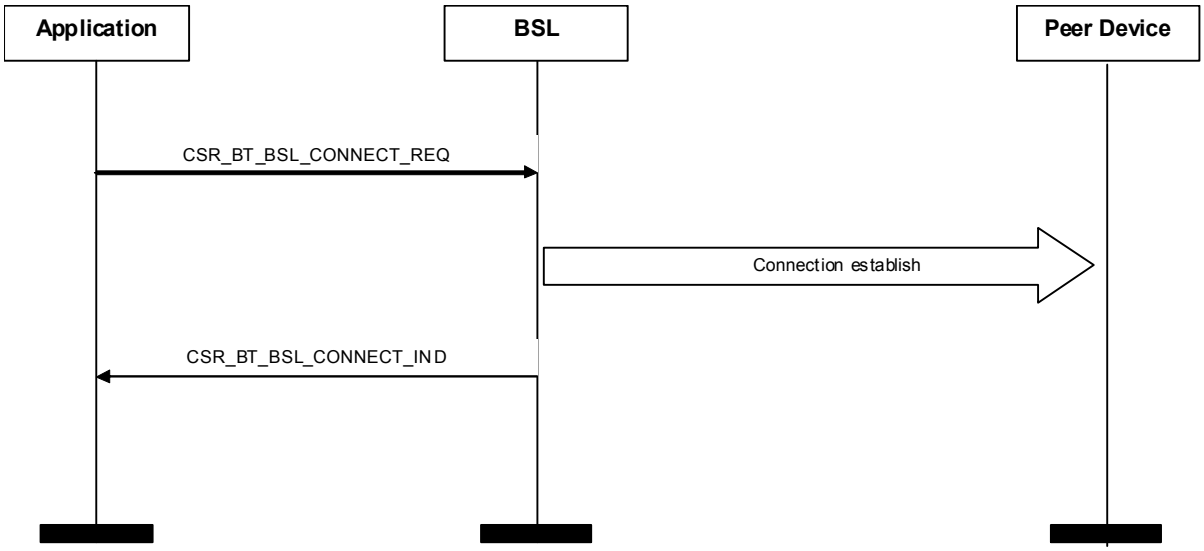


Figure 5: Connect sequence

The BSL layer is able to receive more than one outgoing connect request simultaneously. However, internally only one request is handled at a time. This implies that subsequent requests are queued and restored on completion of the previous request.

Please note that it is only possible to establish one physical connection between a pair of Bluetooth devices and thus a CSR_BT_BSL_CONNECT_REQ must be issued only if no other connection exists.

3.4 Cancel Outgoing Connect

It is possible to cancel an on-going connect attempt in the outgoing direction. For this to happen, the CSR_BT_BSL_CANCEL_CONNECT_REQ signal must be used.

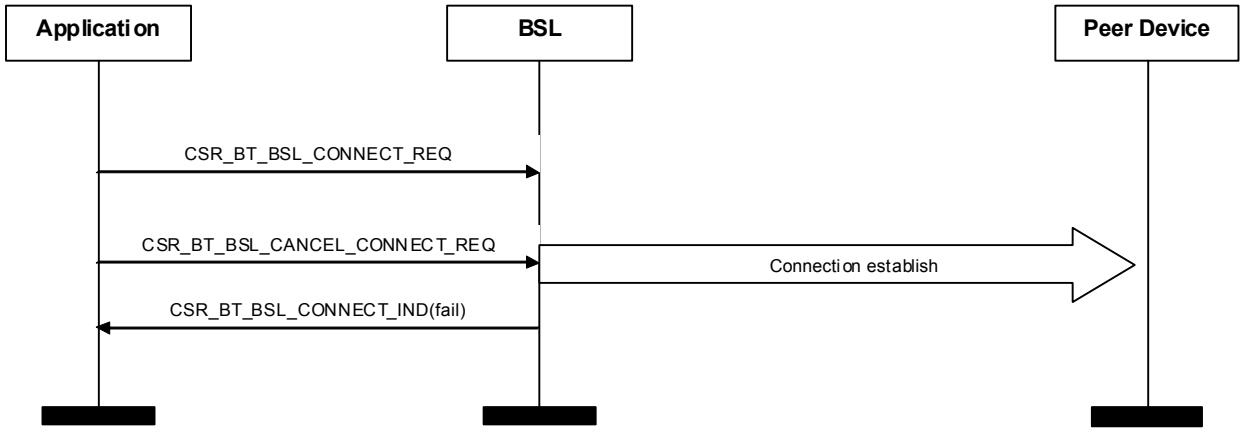


Figure 6: Connect sequence cancelled

It is possible for the cancel connect signal to arrive in the BSL layer after a successful connection has been established. In this case, the signal is treated as a disconnect, so a disconnect indication will follow the cancel signal in that case

3.5 Data Exchange

3.5.1 Point to Point Data Exchange

Once a connection is established data can be exchanged between the connected devices; this is done using the CSR_BT_BSL_DATA_REQ and CSR_BT_BSL_DATA_IND pair. The address field in these messages are in Ethernet format. Functions for converting Bluetooth addresses to Ethernet addresses are located in the csr_bt_util.c file.

The data primitives include the packet type as well as the source and destination address.

Data is passed on by a reference to the payload. When downstream data is submitted by the application to the BSL layer, the BSL layer takes over responsibility of the data reference. When data is sent from the BSL layer to the application layer, it is the responsibility of the receiving application to free the received data when found appropriate.

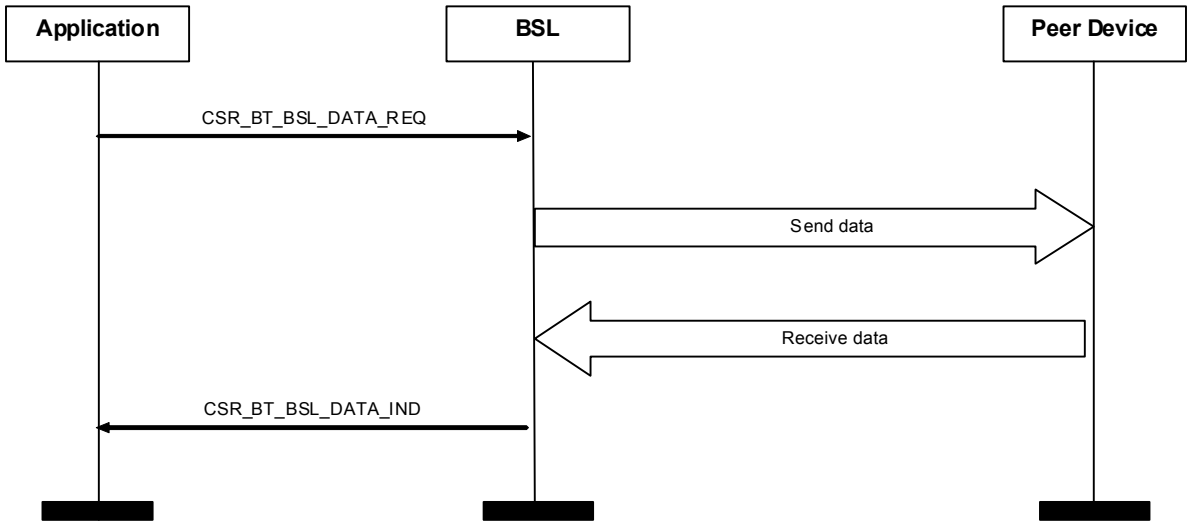


Figure 7: Data exchange

When data is transmitted the id field must be used. The id field can be used in one of two modes:

1. The id received in the CSR_BT_BSL_CONNECT_IND can be used.
2. The CSR_BT_BSL_IS_DEST_ADDR define can be used.

Using option 1 the BSL layer will ensure that the data is always and irrespective of the destination address forwarded to the device represented by the id received in the connect indication. If the CSR_BT_BSL_ID_DEST_ADDR id is used the BSL layer will use the destination address and internally relate the destination address to the right peer device; The CSR_BT_BSL_ID_DEST_ADDR is defined in the file csr_bt_bsl_prim.h.

3.5.2 Multipoint Data Exchange

Multicast or broadcast data can be transmitted using the CSR_BT_BSL_MULTICAST_DATA_REQ primitive. The multicast request must also include an id. Depending on if the message should be sent to its own stack or not the following id defines can be used:

- CSR_BT_BSL_ID_MULTI_TO_LOCAL: include the message to own stack
- CSR_BT_BSL_ID_MULTI_NO_LOCAL: do not send the broadcast message to own stack

The CSR Synergy Bluetooth stack will take care of copying the message to every established link.

3.5.3 Data Buffer Configuration

The maximum amount of data that can be sent in one signal is fixed and cannot be changed. The maximum packet size is 1500 bytes, which corresponds to the maximum Ethernet packet payload size. It is the responsibility of the application not to exceed this packet size.

If data for some reason is held up in the local device, e.g. due to a bad radio connection and multiple retransmissions, data is buffered until the radio subsystem is ready for transmission. The amount of data packets that can be stored in the buffer is defined by the CSR_BT_MAX_BSL_QUEUE_LENGTH defined in the csr_bt_usr_config.h file. The buffer implements a very simple queue structure without any timer supervision, i.e. if more tha CSR_BT_MAX_BSL_QUEUE_LENGTH packets are waiting for transmission any new data packet will be discarded. Please note that a queue exists for every established connection and may consequently consume memory.

For multicast data the define to be used is CSR_BT_MAX_ BSL_LOCAL_MULTICAST_QUEUE_LENGTH.

3.6 Disconnection

When a connection is no longer needed the application layer must disconnect the link; this is done by means of the CSR_BT_BSL_DISCONNECT_REQ signal. The id field indicates the connection being disconnected and must be used.

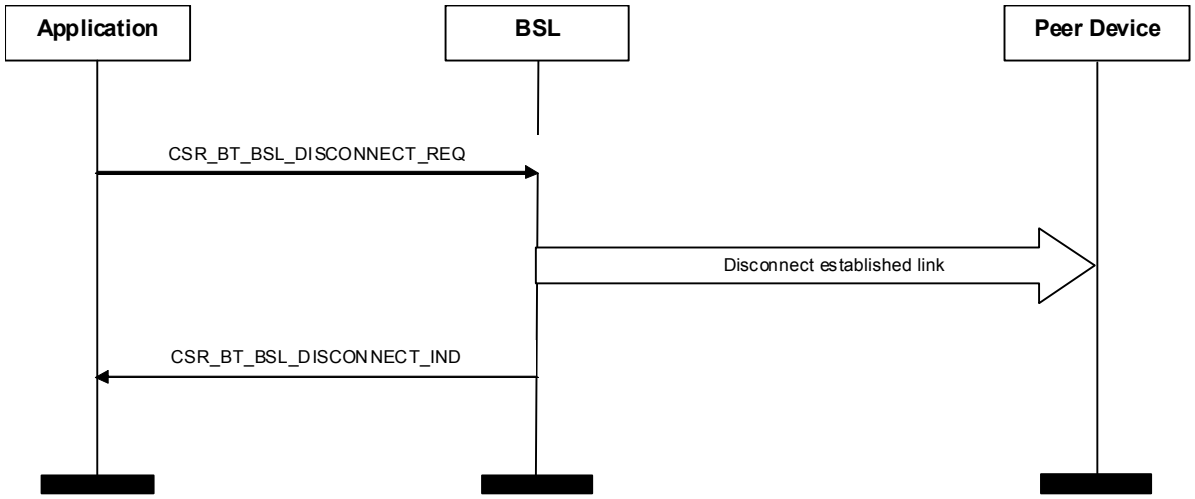


Figure 8: Disconnect sequence

The disconnect is confirmed with the CSR_BT_BSL_DISCONNECT_IND.

Disconnection initiated from peer side is indicated to the application layer via the CSR_BT_BSL_DISCONNECT_IND signal.

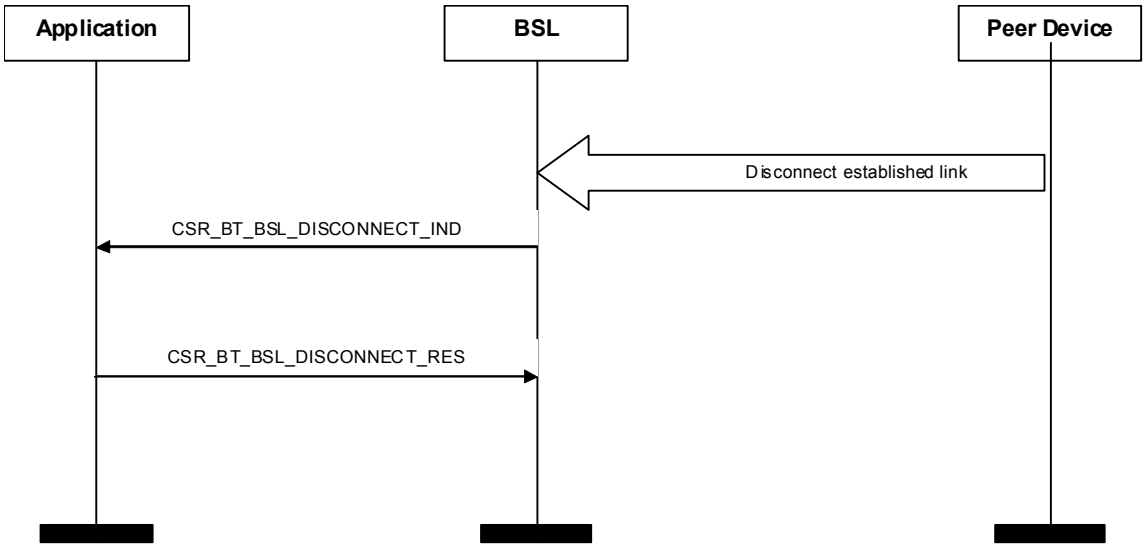


Figure 9: Disconnect sequence

The peer initiated disconnect must be confirmed by the application layer using the CSR_BT_BSL_DISCONNECT_RES signal.

3.7 Service Discovery

3.7.1 Remote Device Service Discovery

When a possible remote device has been located using the discovery procedure, it is possible to search that remote device for accessible features defined in the service discovery record. A search is initiated using the CSR_BT_BSL_SERVICE_SEARCH_REQ primitive; this primitive must include the remote device Bluetooth address and the PAN role (e.g. PANU) for which the service attributes is requested.

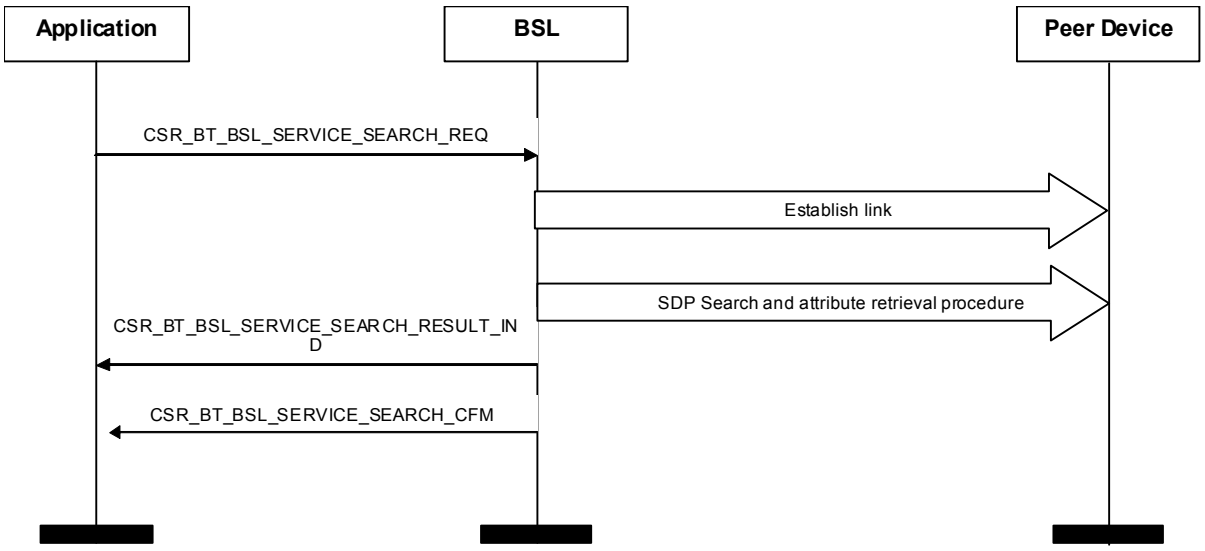


Figure 10, Service search

Results of the service search are reported to the application layer in a CSR_BT_BSL_SERVICE_SEARCH_RESULT_IND signal. The result signal includes all the remote attributes available in the service record. If more than one record is defined on the remote device more result indications

may be sent to the application layer. Each role defines a separate structure that must be used; the three structures are defined in the csr_bt_bsl_prim.h file.

Completion of search is indicated with a CSR_BT_BSL_SERVICE_SEARCH_CFM; this signal is also used if no record matching the search criteria is found.

3.7.2 Defining Service Discovery Records

On the local side service records must be defined. BSL will automatically enable and disable these for retrieval by remote devices depending on the enabled roles (see 3.2). The service records are defined in the csr_bt_bsl_util.c file; one record for each role is defined. These records can and should be adjusted for each specific application and porting; especially the service name and supported network packet types should be considered.

4 BSL Primitives

This section gives an overview of the primitives and parameters in the interface. Detailed information can be found in the corresponding csr_bt_bsl_prim.h file.

The library file csr_bt_bsl_lib.h defines the valid information elements, which may be included when sending a signal.

4.1 List of All Primitives

Primitives	Reference
CSR_BT_BSL_REGISTER_REQ	See reference 4.2
CSR_BT_BSL_ACTIVATE_REQ	See reference 4.3
CSR_BT_BSL_ACTIVATE_CFM	See reference 4.3
CSR_BT_BSL_CONNECT_REQ	See reference 4.4
CSR_BT_BSL_CONNECT_IND	See reference 4.4
CSR_BT_BSL_DISCONNECT_REQ	See reference 4.5
CSR_BT_BSL_DISCONNECT_RES	See reference 4.5
CSR_BT_BSL_DISCONNECT_IND	See reference 4.5
CSR_BT_BSL_DATA_REQ	See reference 4.6
CSR_BT_BSL_DATA_IND	See reference 4.6
CSR_BT_BSL_MULTICAST_DATA_REQ	See reference 4.7
CSR_BT_BSL_STATUS_IND	See reference 4.8
CSR_BT_BSL_SERVICE_SEARCH_REQ	See reference 4.9
CSR_BT_BSL_SERVICE_SEARCH_CFM	See reference 4.9
CSR_BT_BSL_SERVICE_SEARCH_RESULT_IND	See reference 4.10
CSR_BT_BSL_SECURITY_IN_REQ	See reference 4.11
CSR_BT_BSL_ SECURITY_IN_CFM	See reference 4.11
CSR_BT_BSL_SECURITY_OUT_REQ	See reference 4.11
CSR_BT_BSL_SECURITY_OUT_CFM	See reference 4.11
CSR_BT_BSL_CANCEL_CONNECT_REQ	See reference 4.12

Table 1: List of BSL primitives

4.2 CSR_BT_BSL_REGISTER

Primitives \ Parameters	type	connPhandle	dataPhandle
CSR_BT_BSL_REGISTER_REQ	✓	✓	✓

Table 2: CSR_BT_BSL_REGISTER Primitive

Description

This signal is used for registering the application handles in the BSL layer. This is the first signal that must be sent before any other.

Parameters

type	The signal identity, CSR_BT_BSL_REGISTER_REQ.
connPhandle	Handle of the layer responsible for all connection setup and disconnect
dataPhandle	Handle of the layer responsible for all data handling

The connPhandle and dataPhandle can be identical.

4.3 CSR_BT_BSL_ACTIVATE

Primitives \ Parameters	type	singleUser	localRole	remoteRole	resultCode	resultSupplier
CSR_BT_BSL_ACTIVATE_REQ	✓	✓	✓	✓		
CSR_BT_BSL_ACTIVATE_CFM	✓				✓	✓

Table 3: CSR_BT_BSL_ACTIVATE Primitive

Description

This signal is used for activating the local device for incoming connections. After the active confirm signal is received remote devices can initiate connections and the application layer must be prepared to receive the CSR_BT_BSL_CONNECT_IND signal.

Parameters

type — The signal identity, CSR_BT_BSL_ACTIVATE_REQ/CFM.

singleUser — TRUE is only one connection is allowed; in this case the BSL layer will not allow more than one connection and the application layer will not receive more than one simultaneous connect indication.

localRole — The allowed/supported local roles according to the PAN profile. Valid roles are:

CSR_BT_BSL_PANU_ROLE
CSR_BT_BSL_NAP_ROLE
CSR_BT_BSL_GN_ROLE

remoteRole — The allowed/supported roles of the remote device connecting to the local device. The roles are defined according to the PAN profile. Valid roles are:
CSR_BT_BSL_PANU_ROLE
CSR_BT_BSL_NAP_ROLE
CSR_BT_BSL_GN_ROLE

resultCode — The result code of the operation. Possible values depend on the value of resultSupplier. If e.g. the resultSupplier == CSR_BT_SUPPLIER_CM then the possible result codes can be found in csr_bt_cm_prim.h. All values which are currently not specified in the respective prim.h file are regarded as reserved and the application should consider them as errors.

resultSupplier — This parameter specifies the supplier of the result given in resultCode. Possible values can be found in csr_bt_result.h

4.4 CSR_BT_BSL_CONNECT

Primitives \ Parameters	type	bdAddr	remoteRole	localRole	resultCode	rsultSupplier	resultSupplier	id
CSR_BT_BSL_CONNECT_REQ	✓	✓	✓	✓				
CSR_BT_BSL_CONNECT_IND	✓	✓	✓	✓	✓	✓	✓	✓

Table 4: CSR_BT_BSL_CONNECT Primitive

Description

This signal is used for initiating connection towards a remote device defined the Bluetooth address. The connect request must define exactly one local and remote role.

Parameters

type — The signal identity, CSR_BT_BSL_CONNECT_REQ/IND.

bdAddr — The Bluetooth® device address to which a connection must be established.

localRole — The allowed/supported local roles according to the PAN profile. Valid roles are:

CSR_BT_BSL_PANU_ROLE
CSR_BT_BSL_NAP_ROLE
CSR_BT_BSL_GN_ROLE

remoteRole — The allowed/supported roles of the remote device connecting to the local device. The roles are defined according to the PAN profile. Valid roles are:
CSR_BT_BSL_PANU_ROLE
CSR_BT_BSL_NAP_ROLE
CSR_BT_BSL_GN_ROLE

resultCode — The result code of the operation. Possible values depend on the value of resultSupplier. If e.g. the resultSupplier == CSR_BT_SUPPLIER_CM then the possible result codes can be found in csr_bt_cm_prim.h. All values which are currently not specified in the respective prim.h file are regarded as reserved and the application should consider them as errors.

resultSupplier — This parameter specifies the supplier of the result given in resultCode. Possible values can be found in csr_bt_result.h

id — Unique identifier of this connection. Must be used in subsequent disconnect requests and also used for identifying all subsequent data indication signals.

4.5 CSR_BT_BSL_DISCONNECT

Primitives \ Parameters	type	id	resultCode	resultSupplier
CSR_BT_BSL_DISCONNECT_REQ	✓	✓		
CSR_BT_BSL_DISCONNECT_RES	✓	✓		
CSR_BT_BSL_DISCONNECT_IND	✓	✓	✓	✓

Table 5: CSR_BT_BSL_DISCONNECT Primitive

Description

This signal is used for disconnecting an already established connection. The identity received in the connect indication signal must be used.

Parameters

type	The signal identity, CSR_BT_BSL_DISCONNECT_REQ/RES/IND.
id	Unique identifier of this connection. Received in the connect indication signal.
resultCode	The result code of the operation. Possible values depend on the value of resultSupplier. If e.g. the resultSupplier == CSR_BT_SUPPLIER_CM then the possible result codes can be found in csr_bt_cm_prim.h. All values which are currently not specified in the respective prim.h file are regarded as reserved and the application should consider them as errors.
resultSupplier	This parameter specifies the supplier of the result given in resultCode. Possible values can be found in csr_bt_result.h

4.6 CSR_BT_BSL_DATA

Primitives \ Parameters	type	id	etherType	dstAddr	srcAddr	length	*payload
CSR_BT_BSL_DATA_REQ	✓	✓	✓	✓	✓	✓	✓
CSR_BT_BSL_DATA_IND	✓	✓	✓	✓	✓	✓	✓

Table 6: CSR_BT_BSL_DATA Primitive

Description

This signal is used for sending data to a remote device for which a connection is already established.

Parameters

type	The signal identity, CSR_BT_BSL_DATA_REQ/IND.
id	Unique identifier of this connection. Received in the connect indication signal.
etherType	Type of Ethernet packet.
dstAddr	Ethernet address of destination for this data packet.
srcAddr	Ethernet address of source for this data packet
length	Length in bytes for the data payload
*payload	Pointer reference to the payload data. The BSL layer takes control of this reference when received; i.e. the BSL will free this data when submitted to the remote side.

4.7 CSR_BT_BSL_MULTICAST_DATA

Primitives \ Parameters	type	idNot	etherType	dstAddr	srcAddr	length	*payload
CSR_BT_BSL_MULTICAST_DATA_REQ	✓	✓	✓	✓	✓	✓	✓

Table 7: CSR_BT_BSL_ MULTICAST_DATA Primitive

Description

This signal is used for sending multicast data. I.e. send payload data to all connected remote devices and possibly own higher layer stack.

Parameters

type	The signal identity, CSR_BT_BSL_ MULTICAST_DATA_REQ.
idNot	Defines whether the packet should be sent to local stack or not. Valid values are: CSR_BT_BSL_ID_MULTI_TO_LOCAL CSR_BT_BSL_ID_MULTI_NO_LOCAL
etherType	Type of Ethernet packet.
dstAddr	Ethernet address of destination for this data packet.
srcAddr	Ethernet address of source for this data packet
length	Length in bytes for the data payload
*payload	Pointer reference to the payload data. The BSL layer takes control of this reference when received; i.e. the BSL will free this data when submitted to the remote side.

4.8 CSR_BT_BSL_STATUS

Primitives \ Parameters	type	id	event	status
CSR_BT_BSL_STATUS_IND	✓	✓	✓	✓

Table 8: CSR_BT_BSL_STATUS Primitive

Description

This signal is used for indicating link status information to the application layer.

Parameters

type	The signal identity, CSR_BT_BSL_STATUS_IND.
id	Unique identifier of this connection. Received in the connect indication signal.
event	The event. Possible events are: CSR_BT_BSL_LINK_STATE_EVENT CSR_BT_BSL_SWITCH_ROLE_EVENT
status	The status value for the defined event. Valid values are defined in the csr_bt_profiles.h file.

4.9 CSR_BT_BSL_SERVICE_SEARCH

Primitives \ Parameters	type	phandle	bd_addr	searchRole	resultCode	resultSupplier
CSR_BT_BSL_SERVICE_SEARCH_REQ	✓	✓	✓	✓		
CSR_BT_BSL_SERVICE_SEARCH_CFM	✓		✓		✓	✓

Table 9: CSR_BT_BSL_SERVICE_SEARCH Primitive

Description

This signal is used for starting a service search on a remote device.

Parameters

type	The signal identity, CSR_BT_BSL_SERVICE_SEARCH_REQ/CFM.
phandle	Handle of instance which should receive the responses
bd_addr	The Bluetooth® device address to which a connection must be established.
searchRole	The role for which the service record should be received. Valid roles are: CSR_BT_BSL_PANU_ROLE CSR_BT_BSL_NAP_ROLE CSR_BT_BSL_GN_ROLE
resultCode	The result code of the operation. Possible values depend on the value of resultSupplier. If e.g. the resultSupplier == CSR_BT_SUPPLIER_CM then the possible result codes can be found in csr_bt_cm_prim.h. All values which are currently not specified in the respective prim.h file are regarded as reserved and the application should consider them as errors.
resultSupplier	This parameter specifies the supplier of the result given in resultCode. Possible values can be found in csr_bt_result.h

4.10 CSR_BT_BSL_SERVICE_SEARCH_RESULT

Parameters / Primitives	type	bd_addr	moreResults	searchRole	searchResult
CSR_BT_BSL_SERVICE_SEARCH_RESULT_IND	✓	✓	✓	✓	✓

Table 10: CSR_BT_BSL_SERVICE_SEARCH_RESULT Primitive

Description

This signal is used for indicating result of service search.

Parameters

type	The signal identity, CSR_BT_BSL_SERVICE_SEARCH_RESULT_IND.
bd_addr	The Bluetooth® device address to which a connection must be established.
moreResults	TRUE if more records can be sent, i.e. the remote device has more than this record matching the search pattern.
searchRole	The role for which the service record should be received. Valid roles are: CSR_BT_BSL_PANU_ROLE CSR_BT_BSL_NAP_ROLE CSR_BT_BSL_GN_ROLE
searchResult	Found remote search record parameters, see below. The result is constructed of the common sds record and for the NAP the nap sds record respectively GN.

4.11 CSR_BT_BSL_SECURITY_IN / _OUT

Primitives \ Parameters	type	appHandle	secLevel	resultCode	resultSupplier
CSR_BT_BSL_SECURITY_IN_REQ	✓	✓	✓		
CSR_BT_BSL_ SECURITY_IN_CFM	✓			✓	✓
CSR_BT_BSL_SECURITY_OUT_REQ	✓	✓	✓		
CSR_BT_BSL_SECURITY_OUT_CFM	✓			✓	✓

Table 11: CSR_BT_BSL_SECURITY_IN and CSR_BT_BSL_ SECURITY_OUT 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.

The *CSR_BT_SECURITY_OUT_REQ* signal sets up the security level for new outgoing connections. Already established and pending connections are not altered. Note that *authorization* should not be used for outgoing connections as that may be confusing for the user – there is really no point in requesting an outgoing connection and afterwards having to authorize as they are both locally-only decided procedures.

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_BSL_SECURITY_IN/_OUT_REQ/CFM

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)

resultCode	The result code of the operation. Possible values depend on the value of resultSupplier. If e.g. the resultSupplier == CSR_BT_SUPPLIER_CM then the possible result codes can be found in csr_bt_cm_prim.h. All values which are currently not specified in the respective prim.h file are regarded as reserved and the application should consider them as errors.
resultSupplier	This parameter specifies the supplier of the result given in resultCode. Possible values can be found in csr_bt_result.h

4.12 CSR_BT_BSL_CANCEL_CONNECT

Primitives \ Parameters	type	bdAddr
CSR_BT_BSL_CANCEL_CONNECT_REQ	✓	✓

Table 12: CSR_BT_BSL_CANCEL_CONNECT Primitive

Description

This signal is used for cancelling a previously initiated connection attempt. The signal is confirmed by a CSR_BT_BSL_CONNECT_IND with the result code CSR_BT_CANCELLED_CONNECT_ATTEMPT. If the signal arrives after the connection has been established successfully, it will be treated as a disconnect and causing a CSR_BT_BSL_DISCONNECT_IND to be sent to the application layer.

Parameters

type	The signal identity, CSR_BT_BSL_CANCEL_CONNECT_REQ.
bdAddr	The Bluetooth® device address to which a connection must be established.

5 Document References

Document	Reference
The Bluetooth® Specification, the Personal Area Networking Profile.	[PAN]
Bluetooth Network Encapsulation Protocol Specification	[BNEP]
CSR Synergy Bluetooth, SC – Security Controller API Description, Document no. api-0102-sc	[SC]
CSR Synergy Bluetooth. CM – Connection Manager API Description, doc. no. api-0101-cm	[CM]
Scheduler API, api-0004-coal	[SCHED_API]

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
CSR	Cambridge Silicon Radio
UniFi™	Group term for CSR's range of chips designed to meet IEEE 802.11 standards

Document History

Revision	Date	History
1	26 SEP 11	Ready for release 18.2.0

TradeMarks, Patents and Licences

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.