L2C API

Generated by Doxygen 1.8.13

Contents

Chapter 1

Module Documentation

1.1 Logical Link Control and Adaptation Protocol (L2CAP)

Modules

- L2CAP API

1.1.1 Detailed Description

1.1.2 Introduction

The L2C subsystem implements the LE L2CAP protocol. It is a substantially scaled-down version of regular Bluetooth L2CAP.

L2C interfaces to HCI to send and receive ACL packets. The ATT and SMP protocol layers interface to L2C to send and receive L2CAP packets. L2C also interfaces to DM to perform the L2CAP connection update procedure.

In the TX data path, the main function of L2C is building L2CAP packets and sending them to HCI. L2C also implements flow control for the TX data path. L2C may in the future implement admission control in the TX path, allowing more optimal buffer sharing between multiple simultaneous links.

In the RX data path, its main function is receiving packets from HCI and routing them to either SMP or ATT. L2C also implements the connection parameter update procedure.

For full API, see L2CAP API

1.1.3 Subsystem Architecture

Module l2c_api contains the API. Module l2c_main contains the main API function implementation, main event handler, and functions for processing packets. Module l2c_master contains API functions and other functions used only when operating as an LE master. Module l2c_slave contains API functions and other functions used only when operating as an LE slave. Module lcc_coc contains functions for L2CAP Connection Oriented Channels.

1.1.4 Usage Scenarios

This section describes example scenarios for initialization and connection.

1.1.4.1 Initialization

Figure 2 shows the initialization process. In this example, the system supports operation as both a master and a slave so L2cMasterInit() and L2cSlaveInit() are called. Then function L2cSlaveHandlerInit() is called after L2cSlaveHandler() is set up in the WSF OS implementation.

1.1.4.2 Data Path

Figure 3 shows the operation of the data path with ATT shown as an example L2C client. ATT calls L2cDataReq() to send a packet to L2C. Then L2C calls HciSendAclData() to send the packet to HCI. In the receive direction, HCI calls HciAclDataCback() to send a packet to L2C. L2C calls ATT callback function attDataCback() to send the packet to ATT.

1.1.4.3 Connection Parameter Update

Figure 4 shows a connection parameter update procedure with the stack operating as a slave. DM calls L2cDmConnUpdateReq() to initiate the process. L2C builds and sends an L2CAP Connection Parameter Update Request. The peer device receives the request and initiates a connection update procedure. When the procedure completes, an HCI LE Connection Update Complete Event is sent from HCI to DM. Then the L2CAP Connection Parameter Update Response is received from the peer and L2C calls DmL2cConnUpdateCnf().

1.2 L2CAP API

Data Structures

- struct l2cCocReg_t

  Connection oriented channel registration structure.
- struct l2cCocConnectInd_t

  Connection oriented channel connect indication structure.
- struct l2cCocDisconnectInd_t

  Connection oriented channel disconnect indication structure.
- struct l2cCocDataInd_t

  Connection oriented channel data indication structure.
- struct l2cCocDataCnf_t

  Connection oriented channel disconnect indication structure.
- struct l2cCocEnConnectInd_t

  Enhanced connection oriented channel connect indication structure.
- struct l2cCocEnReconfigInd_t

  Enhanced connection oriented channel reconfiguration indication structure.
- union l2cCocEvt_t

  Connection oriented channel event structure.
- struct l2cCfg_t

  Configurable parameters.

Macros

- #define L2C_COC_REG_ID_NONE 0

  Invalid channel registration ID for connection oriented channels.
- #define L2C_COC_CID_NONE 0

  Invalid channel ID for connection oriented channels.
- #define L2C_SIGNAL_ID_INVALID 0

  Invalid signal identifier.
- #define L2C_MAX_EN_CHAN 5

  Max number of channels per enhanced connection request.
- #define L2C_PAYLOAD_START (HCI_ACL_HDR_LEN + L2C_HDR_LEN)

  Start of L2CAP payload in an HCI ACL packet buffer.
- #define L2C_SIG_PKT_BASE_LEN (HCI_ACL_HDR_LEN + L2C_HDR_LEN + L2C_SIG_HDR_LEN)

  L2CAP signaling packet base length, including HCI header.
- #define L2C_LE_SDU_PKT_BASE_LEN (HCI_ACL_HDR_LEN + L2C_HDR_LEN + L2C_LE_SDU_HDR↩ _LEN)

  L2CAP LE SDU packet base length, including HCI header.
- #define L2C_SIG_RSP_FLAG 0x01

  Signaling response code flag.

Typedefs

- typedef uint16_t l2cCocRegId_t

  Connection oriented channel registration ID.
- typedef void(∗ l2cDataCback_t) (uint16_t handle, uint16_t len, uint8_t ∗pPacket)

  This callback function sends a received L2CAP packet to the client.
- typedef void(∗ l2cCtrlCback_t) (wsfMsgHdr_t ∗pMsg)

  This callback function sends control messages to the client.
- typedef void(∗ l2cCocCback_t) (l2cCocEvt_t ∗pMsg)

  This callback function sends data and other events to connection oriented channels clients.
- typedef uint8_t(∗ l2cCocAcceptCb_t) (dmConnId_t connId, uint8_t numChans)

  This callback function asks clients of connection oriented channels if a given number of channels can be created on the PSM.
- typedef uint16_t(∗ l2cCocAuthorCback_t) (dmConnId_t connId, l2cCocRegId_t regId, uint16_t psm)

  This callback function is used for authoriztion of connection oriented channels.

L2CAP Control Callback Events

Control callback message events

- #define L2C_CTRL_FLOW_ENABLE_IND 0

  Data flow enabled.
- #define L2C_CTRL_FLOW_DISABLE_IND 1

  Data flow disabled.

L2CAP COC Channel Roles

Connection oriented channel initiator/acceptor role

- #define L2C_COC_ROLE_NONE 0x00

  No role (unallocated)
- #define L2C_COC_ROLE_INITIATOR 0x01

  Channel initiator.
- #define L2C_COC_ROLE_ACCEPTOR 0x02

  Channel acceptor.

L2CAP COC Data Confirm Codes

Connection oriented channel data confirm status values

- #define L2C_COC_DATA_SUCCESS 0

  Data request successful.
- #define L2C_COC_DATA_ERR_MEMORY 1

  Out of memory.
- #define L2C_COC_DATA_ERR_OVERFLOW 2

  Transaction overflow.

L2CAP COC Callback Events

Connection oriented channel callback events.

- enum {
  L2C_COC_CONNECT_IND = L2C_COC_CBACK_START,
  L2C_COC_DISCONNECT_IND,
  L2C_COC_EN_CONNECT_IND,
  L2C_COC_EN_RECONFIG_IND,
  L2C_COC_DATA_IND,
  L2C_COC_DATA_CNF }

  COC callback events.
- #define L2C_COC_CBACK_START 0x70

  L2C callback event starting value.
- #define L2C_COC_CBACK_CBACK_END L2C_COC_DATA_CNF

  L2C callback event ending value.

L2CAP Initialization

Initialization and registration functions

- void L2cInit (void)

  Initialize L2C subsystem.
- void L2cMasterInit (void)

  Initialize L2C for operation as a Bluetooth LE master.
- void L2cSlaveInit (void)

  Initialize L2C for operation as a Bluetooth LE slave.

L2CAP CID Functions

Register and send data over a CID

- void L2cRegister (uint16_t cid, l2cDataCback_t dataCback, l2cCtrlCback_t ctrlCback)

  called by the L2C client, such as ATT or SMP, to register for the given CID.
- void L2cDataReq (uint16_t cid, uint16_t handle, uint16_t len, uint8_t *pL2cPacket)

  Send an L2CAP data packet on the given CID.
- void L2cDmSigReq (uint16_t handle, uint8_t code, uint16_t len, uint8_t *pParam)

  Build and send a signaling packet.

L2CAP COC Functions

Connection Oriented Channels Functions

- void L2cCocInit (void)

  Initialize L2C connection oriented channel subsystem.
- l2cCocRegId_t L2cCocRegister (l2cCocCback_t cback, l2cCocReg_t *pReg)

  Register to use a connection oriented channel, as either a channel acceptor, initiator, or both. If registering as channel acceptor then the PSM is specified. After registering a connection can be established by the client using this registration instance.
- void L2cCocDeregister (l2cCocRegId_t regId)

  Deregister and deallocate a connection oriented channel registration instance. This function should only be called if there are no active channels using this registration instance.
- void L2cCocSetAcceptCback (l2cCocRegId_t regId, l2cCocAcceptCb_t cback)

  Set the channel accept callback.
- uint16_t L2cCocConnectReq (dmConnId_t connId, l2cCocRegId_t regId, uint16_t psm)

  Initiate a connection to the given peer PSM.
- void L2cCocDisconnectReq (uint16_t cid)

  Disconnect the channel for the given CID.
- void L2cCocDataReq (uint16_t cid, uint16_t len, uint8_t *pPayload)

  Send an L2CAP data packet on the given connection oriented CID.
- bool_t L2cCocEnhancedConnectReq (dmConnId_t connId, l2cCocRegId_t regId, uint16_t psm, uint16_t credits, uint8_t numChan)

  Send a request to open enhanced credit based channels.
- bool_t L2cCocEnhancedReconfigReq (dmConnId_t connId, uint16_t mtu, uint16_t mps, uint8_t numChan, uint16_t *pChanList)

  Send a request to reconfigure enhanced credit based channels.
- void L2cCocErrorTest (uint16_t result)

  For testing purposes only.
- void L2cCocCreditSendTest (uint16_t cid, uint16_t credits)

  For testing purposes only.

L2CAP Connection Parameter Update Functions

- void L2cDmConnUpdateReq (uint16_t handle, hciConnSpec_t *pConnSpec)

  For internal use only. This function is called by DM to send an L2CAP connection update request.
- void L2cDmConnUpdateRsp (uint8_t identifier, uint16_t handle, uint16_t result)

  For internal use only. This function is called by DM to send an L2CAP connection update response.

L2CAP Packet Constants

- #define L2C_HDR_LEN 4

  L2CAP packet header length.
- #define L2C_MIN_MTU 23

  Minimum packet payload MTU for LE.
- #define L2C_SIG_HDR_LEN 4

  L2CAP signaling command header length.
- #define L2C_LE_SDU_HDR_LEN 2

  L2CAP LE SDU data header length.

L2CAP Parameter Lengths

Signaling packet parameter lengths

- #define L2C_SIG_CONN_UPDATE_REQ_LEN 8

  Connection update request length.
- #define L2C_SIG_CONN_UPDATE_RSP_LEN 2

  Connection update response length.
- #define L2C_SIG_CMD_REJ_LEN 2

  Command reject length.
- #define L2C_SIG_DISCONN_REQ_LEN 4

  Disconnection request length.
- #define L2C_SIG_DISCONN_RSP_LEN 4

  Disconnection response length.
- #define L2C_SIG_LE_CONN_REQ_LEN 10

  LE connection request length.
- #define L2C_SIG_LE_CONN_RSP_LEN 10

  LE connection response length.
- #define L2C_SIG_FLOW_CTRL_CREDIT_LEN 4

  Flow control credit lenghth.
- #define L2C_SIG_EN_CONNECT_REQ_LEN 8

  Enhanced credit based connection request.
- #define L2C_SIG_EN_CONNECT_RSP_LEN 8

  Enhanced credit based connection response.
- #define L2C_SIG_EN_RECONFIG_REQ_LEN 4

  Enhanced credit based reconfiguration request.
- #define L2C_SIG_EN_RECONFIG_RSP_LEN 2

  Enhanced credit based reconfiguration response.

L2CAP Connection Identifiers

BLE Defined Connection Identifiers (CID)

- #define L2C_CID_ATT 0x0004

  CID for attribute protocol.
- #define L2C_CID_LE_SIGNALING 0x0005

  CID for LE signaling.
- #define L2C_CID_SMP 0x0006

  CID for security manager protocol.

L2CAP Signaling Codes

- #define L2C_SIG_CMD_REJ 0x01
  Comand reject.
- #define L2C_SIG_DISCONNECT_REQ 0x06
  Disconnect request.
- #define L2C_SIG_DISCONNECT_RSP 0x07
  Disconnect response.
- #define L2C_SIG_CONN_UPDATE_REQ 0x12
  Connection parameter update request.
- #define L2C_SIG_CONN_UPDATE_RSP 0x13
  Connection parameter update response.
- #define L2C_SIG_LE_CONNECT_REQ 0x14
  LE credit based connection request.
- #define L2C_SIG_LE_CONNECT_RSP 0x15
  LE credit based connection response.
- #define L2C_SIG_FLOW_CTRL_CREDIT 0x16
  LE flow control credit.
- #define L2C_SIG_EN_CONNECT_REQ 0x17
  Enhanced credit based connection request.
- #define L2C_SIG_EN_CONNECT_RSP 0x18
  Enhanced credit based connection response.
- #define L2C_SIG_EN_RECONFIG_REQ 0x19
  Enhanced credit based reconfiguration request.
- #define L2C_SIG_EN_RECONFIG_RSP 0x1A
  Enhanced credit based reconfiguration response.

L2CAP Command Rejection Codes

BLE defined Command rejection reason codes

- #define L2C_REJ_NOT_UNDERSTOOD 0x0000
  Command not understood.
- #define L2C_REJ_MTU_EXCEEDED 0x0001
  Signaling MTU exceeded.
- #define L2C_REJ_INVALID_CID 0x0002
  Invalid CID in request.

L2CAP Connection Parameter Update Result Codes

BLE defined result codes

- #define L2C_CONN_PARAM_ACCEPTED 0x0000
  Connection parameters accepted.
- #define L2C_CONN_PARAM_REJECTED 0x0001
  Connection parameters rejected.

L2CAP Connection Result Codes

BLE defined result codes

- #define L2C_CONN_SUCCESS 0x0000
  Connection successful.
- #define L2C_CONN_NONE 0x0001
  No connection result value available.
- #define L2C_CONN_FAIL_PSM 0x0002
  Connection refused LE_PSM not supported.
- #define L2C_CONN_FAIL_RES 0x0004
  Connection refused no resources available.
- #define L2C_CONN_FAIL_AUTH 0x0005
  Connection refused insufficient authentication.
- #define L2C_CONN_FAIL_AUTHORIZ 0x0006
  Connection refused insufficient authorization.
- #define L2C_CONN_FAIL_KEY_SIZE 0x0007
  Connection refused insufficient encryption key size.
- #define L2C_CONN_FAIL_ENC 0x0008
  Connection Refused insufficient encryption.
- #define L2C_CONN_FAIL_INVALID_SCID 0x0009
  Connection refused invalid source CID.
- #define L2C_CONN_FAIL_ALLOCATED_SCID 0x000A
  Connection refused source CID already allocated.
- #define L2C_CONN_FAIL_UNACCEPT_PARAM 0x000B
  Connection refused unacceptable parameters.
- #define L2C_CONN_FAIL_INVALID_PARAM 0x000C
  Connection refused invalid parameters.

L2CAP Interal Connection Result Codes

Proprietary codes not sent in any L2CAP packet.

- #define L2C_CONN_FAIL_TIMEOUT 0xF000
  Request timeout.

L2CAP Signaling Parameter Value Ranges

- #define L2C_PSM_MIN 0x0001
  PSM minimum.
- #define L2C_PSM_MAX 0x00FF
  PSM maximum.
- #define L2C_CID_DYN_MIN 0x0040
  CID dynamic minimum.
- #define L2C_CID_DYN_MAX 0x007F
  CID dynamic maximum.
- #define L2C_MTU_MIN 0x0017
  MTU minimum.
- #define L2C_MPS_MIN 0x0017
  MPS minimum.
- #define L2C_MPS_MAX 0xFFFD
  MPS maximum.
- #define L2C_CREDITS_MAX 0xFFFF
  Credits maximum.

L2CAP Enhanced Connection Reconfigure Result Codes

- #define L2C_RECONFIG_FAIL_MTU 0x0001

  Enhanced Reconfiguration refuded - cannot reduce MTU.
- #define L2C_RECONFIG_FAIL_MPS 0x0002

  Enhanced Reconfiguration refuded - cannot reduce MPS on more than one channel.
- #define L2C_RECONFIG_FAIL_CID 0x0003

  Enhanced Reconfiguration refuded - invalid CID.
- #define L2C_RECONFIG_FAIL_PARAM 0x0004

  Enhanced Reconfiguration refuded - unacceptable parameters.

1.2.1 Detailed Description

1.2.2 Typedef Documentation

1.2.2.1 l2cDataCback_t

```
typedef void(* l2cDataCback_t) (uint16_t handle, uint16_t len, uint8_t *pPacket)
```

This callback function sends a received L2CAP packet to the client.

Parameters

handle	The connection handle.
len	The length of the L2CAP payload data in pPacket.
pPacket	A buffer containing the packet.

Returns

None.

Definition at line 233 of file l2c_api.h.

1.2.2.2 l2cCtrlCback_t

```
typedef void(* l2cCtrlCback_t) (wsfMsgHdr_t *pMsg)
```

This callback function sends control messages to the client.

Parameters

pMsg	Pointer to message structure.

Returns

None.

Definition at line 244 of file l2c_api.h.

1.2.2.3 l2cCocCback_t

```
typedef void(* l2cCocCback_t) (l2cCocEvt_t *pMsg)
```

This callback function sends data and other events to connection oriented channels clients.

Parameters

pMsg	Pointer to message structure.

Returns

None.

Definition at line 256 of file l2c_api.h.

1.2.2.4 l2cCocAcceptCb_t

```
typedef uint8_t(* l2cCocAcceptCb_t) (dmConnId_t connId, uint8_t numChans)
```

This callback function asks clients of connection oriented channels if a given number of channels can be created on the PSM.

Parameters

connId	DM connection ID.
numChans	number of channels requested.

Returns

number of channels permitted by client.

Definition at line 269 of file l2c_api.h.

1.2.2.5 l2cCocAuthorCback_t

```
typedef uint16_t(* l2cCocAuthorCback_t) (dmConnId_t connId, l2cCocRegId_t regId, uint16_t psm)
```

This callback function is used for authoriztion of connection oriented channels.

Parameters

connId	DM connection ID.
regId	The registration instance requiring authorization.
psm	The PSM of the registration instance.

Returns

L2C_CONN_SUCCESS if authorization is successful, any other value for failure.

Definition at line 282 of file l2c_api.h.

1.2.3 Enumeration Type Documentation

1.2.3.1 anonymous enum

```
anonymous enum
```

COC callback events.

Enumerator

L2C_COC_CONNECT_IND	Channel connect indication.
L2C_COC_DISCONNECT_IND	Channel disconnect indication.
L2C_COC_EN_CONNECT_IND	Received enhanced connection indication.
L2C_COC_EN_RECONFIG_IND	Received enhanced reconfiguration indication.
L2C_COC_DATA_IND	Received data indication.
L2C_COC_DATA_CNF	Transmit data confirm.

Definition at line 82 of file l2c_api.h.

```
83 {
84   L2C_COC_CONNECT_IND = L2C_COC_CBACK_START,        /*!< \brief
       Channel connect indication */
85   L2C_COC_DISCONNECT_IND,                          /*!< \brief Channel disconnect
       indication */
86   L2C_COC_EN_CONNECT_IND,                          /*!< \brief Received enhanced
       connection indication */
87   L2C_COC_EN_RECONFIG_IND,                         /*!< \brief Received enhanced
       reconfiguration indication */
88   L2C_COC_DATA_IND,                                /*!< \brief Received data indication */
89   L2C_COC_DATA_CNF                                 /*!< \brief Transmit data confirm */
90 };
```

1.2.4 Function Documentation

1.2.4.1 L2cInit()

```
void L2cInit (
             void  )
```

Initialize L2C subsystem.

Returns

None.

1.2.4.2 L2cMasterInit()

```
void L2cMasterInit (
             void  )
```

Initialize L2C for operation as a Bluetooth LE master.

Returns

None.

1.2.4.3 L2cSlaveInit()

```
void L2cSlaveInit (
             void  )
```

Initialize L2C for operation as a Bluetooth LE slave.

Returns

None.

1.2.4.4 L2cRegister()

```
void L2cRegister (
             uint16_t cid,
             l2cDataCback_t dataCback,
             l2cCtrlCback_t ctrlCback )
```

called by the L2C client, such as ATT or SMP, to register for the given CID.

Parameters

cid	channel identifier.
dataCback	Callback function for L2CAP data received for this CID.
ctrlCback	Callback function for control events for this CID.

Returns

None.

1.2.4.5 L2cDataReq()

```
void L2cDataReq (
            uint16_t cid,
            uint16_t handle,
            uint16_t len,
            uint8_t * pL2cPacket )
```

Send an L2CAP data packet on the given CID.

Parameters

cid	The channel identifier.
handle	The connection handle. The client receives this handle from DM.
len	The length of the payload data in pPacket.
pL2cPacket	A buffer containing the packet.

Returns

None.

1.2.4.6 L2cDmSigReq()

```
void L2cDmSigReq (
            uint16_t handle,
            uint8_t code,
            uint16_t len,
            uint8_t * pParam )
```

Build and send a signaling packet.

Parameters

handle	The connection handle.
code	Type of command.
len	Length of the parameter.
pParam	parameters of command to send.

Returns

None.

1.2.4.7 L2cCocInit()

```
void L2cCocInit (
            void  )
```

Initialize L2C connection oriented channel subsystem.

Returns

None.

1.2.4.8 L2cCocRegister()

```
l2cCocRegId_t L2cCocRegister (
            l2cCocCback_t cback,
            l2cCocReg_t * pReg )
```

Register to use a connection oriented channel, as either a channel acceptor, initiator, or both. If registering as channel acceptor then the PSM is specified. After registering a connection can be established by the client using this registration instance.

Parameters

cback	Client callback function.
pReg	Registration parameter structure.

Returns

Registration instance ID or L2C_COC_REG_ID_NONE if registration failed.

1.2.4.9 L2cCocDeregister()

```
void L2cCocDeregister (
            l2cCocRegId_t regId )
```

Deregister and deallocate a connection oriented channel registration instance. This function should only be called if there are no active channels using this registration instance.

Parameters

regId	Registration instance ID.

Returns

None.

1.2.4.10 L2cCocSetAcceptCback()

```
void L2cCocSetAcceptCback (
            l2cCocRegId_t regId,
            l2cCocAcceptCb_t cback )
```

Set the channel accept callback.

Parameters

regId	Registration instance ID.
cback	Client callback function.

Returns

None.

1.2.4.11 L2cCocConnectReq()

```
uint16_t L2cCocConnectReq (
            dmConnId_t connId,
            l2cCocRegId_t regId,
            uint16_t psm )
```

Initiate a connection to the given peer PSM.

Parameters

connId	DM connection ID.
regId	The associated registration instance.
psm	Peer PSM.

Returns

Local CID or L2C_COC_CID_NONE none if failure.

1.2.4.12 L2cCocDisconnectReq()

```
void L2cCocDisconnectReq (
            uint16_t cid )
```

Disconnect the channel for the given CID.

Parameters

cid	Channel ID.

Returns

None.

1.2.4.13 L2cCocDataReq()

```
void L2cCocDataReq (
            uint16_t cid,
            uint16_t len,
            uint8_t * pPayload )
```

Send an L2CAP data packet on the given connection oriented CID.

Parameters

cid	The local channel identifier.
len	The length of the payload data in pPacket.
pPayload	Packet payload data.

Returns

None.

1.2.4.14 L2cCocEnhancedConnectReq()

```
bool_t L2cCocEnhancedConnectReq (
            dmConnId_t connId,
```

```
            l2cCocRegId_t regId,
            uint16_t psm,
            uint16_t credits,
            uint8_t numChan )
```

Send a request to open enhanced credit based channels.

Parameters

connId	DM connection ID.
regId	The associated registration instance.
psm	The protocol slave multiplexer.
credits	The initial number of credits for each CID channel.
numChan	The number of channels to create - L2C_MAX_EN_CHAN max.

Returns

FALSE if unable make request, else TRUE.

1.2.4.15 L2cCocEnhancedReconfigReq()

```
bool_t L2cCocEnhancedReconfigReq (
            dmConnId_t connId,
            uint16_t mtu,
            uint16_t mps,
            uint8_t numChan,
            uint16_t * pChanList )
```

Send a request to reconfigure enhanced credit based channels.

Parameters

connId	DM connection ID.
mtu	The maximum transmission unit of each source CID channel.
mps	The maximum payload size on each source CID channel.
numChan	The number of channels to create (1 to L2C_MAX_EN_CHAN).
pChanList	A list of local CID to reconfigure (L2C_MAX_EN_CHAN channels, set unused to 0).

Returns

FALSE if unable make request, else TRUE.

1.2.4.16 L2cCocErrorTest()

```
void L2cCocErrorTest (
            uint16_t result )
```

For testing purposes only.

Parameters

result	Result code

Returns

None.

1.2.4.17 L2cCocCreditSendTest()

```
void L2cCocCreditSendTest (
            uint16_t cid,
            uint16_t credits )
```

For testing purposes only.

Parameters

cid	The local channel identifier.
credits	Credits to send.

Returns

None.

1.2.4.18 L2cDmConnUpdateReq()

```
void L2cDmConnUpdateReq (
            uint16_t handle,
            hciConnSpec_t * pConnSpec )
```

For internal use only. This function is called by DM to send an L2CAP connection update request.

Parameters

handle	The connection handle.
pConnSpec	Pointer to the connection specification structure.

Returns

None.

1.2.4.19 L2cDmConnUpdateRsp()

```
void L2cDmConnUpdateRsp (
             uint8_t identifier,
             uint16_t handle,
             uint16_t result )
```

For internal use only. This function is called by DM to send an L2CAP connection update response.

Parameters

identifier	Identifier value previously passed from L2C to DM.
handle	The connection handle.
result	Connection update response result.

Returns

None.

1.3 STACK_INIT

L2CAP Configuration Structure

Pointer to structure containing initialization details of the L2CAP Subsystem. To be configured by Application.

- l2cCfg_t * pL2cCfg
  Configuration pointer.

1.3.1 Detailed Description

1.4 STACK_EVENT

L2CAP Event Handling

Message passing interface to L2CAP from other tasks through WSF.

- void L2cSlaveHandlerInit (wsfHandlerId_t handlerId)

  Event handler initialization function for L2C when operating as a slave.
- void L2cSlaveHandler (wsfEventMask_t event, wsfMsgHdr_t *pMsg)

  The WSF event handler for L2C when operating as a slave.
- void L2cCocHandlerInit (wsfHandlerId_t handlerId)

  Event handler initialization function for L2C with connection oriented channels.
- void L2cCocHandler (wsfEventMask_t event, wsfMsgHdr_t *pMsg)

  The WSF event handler for L2C with connection oriented channels.

1.4.1 Detailed Description

1.4.2 Function Documentation

1.4.2.1 L2cSlaveHandlerInit()

```
void L2cSlaveHandlerInit (
            wsfHandlerId_t handlerId )
```

Event handler initialization function for L2C when operating as a slave.

Parameters

handlerId	ID for this event handler.

Returns

None.

1.4.2.2 L2cSlaveHandler()

```
void L2cSlaveHandler (
            wsfEventMask_t event,
            wsfMsgHdr_t * pMsg )
```

The WSF event handler for L2C when operating as a slave.

Parameters

event	Event mask.
pMsg	Pointer to message.

Returns

None.

1.4.2.3 L2cCocHandlerInit()

```
void L2cCocHandlerInit (
             wsfHandlerId_t handlerId )
```

Event handler initialization function for L2C with connection oriented channels.

Parameters

handler↩ Id	ID for this event handler.

Returns

None.

1.4.2.4 L2cCocHandler()

```
void L2cCocHandler (
             wsfEventMask_t event,
             wsfMsgHdr_t * pMsg )
```

The WSF event handler for L2C with connection oriented channels.

Parameters

event	Event mask.
pMsg	Pointer to message.

Returns

None.

1.5 WSF_TYPES

Integer Data Types

- #define **bool_t** uint8_t
- #define **FALSE** 0
- #define **TRUE** (!FALSE)
- #define **UINT64_C**(x) x##ULL
- #define **UINT32_C**(x) x##UL
- #define **UINT8_C**(x) (x)

1.5.1 Detailed Description

Chapter 2

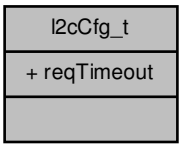
Data Structure Documentation

2.1 l2cCfg_t Struct Reference

Configurable parameters.

`#include <l2c_api.h>`

Collaboration diagram for l2cCfg_t:



Data Fields

- uint16_t reqTimeout
  Request timeout in seconds.

2.1.1 Detailed Description

Configurable parameters.

Definition at line 190 of file l2c_api.h.

The documentation for this struct was generated from the following file:

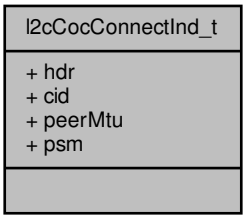
- /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/ble-host/include/l2c_api.h

2.2 l2cCocConnectInd_t Struct Reference

Connection oriented channel connect indication structure.

```
#include <l2c_api.h>
```

Collaboration diagram for l2cCocConnectInd_t:



Data Fields

- wsfMsgHdr_t hdr

  Header structure.
- uint16_t cid

  Local channel ID.
- uint16_t peerMtu

  Data packet MTU peer can receive.
- uint16_t psm

  Connected PSM.

2.2.1 Detailed Description

Connection oriented channel connect indication structure.

Definition at line 115 of file l2c_api.h.

The documentation for this struct was generated from the following file:

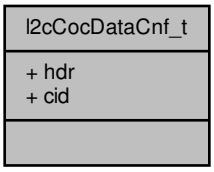
- /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/ble-host/include/l2c_api.h

2.3 l2cCocDataCnf_t Struct Reference

Connection oriented channel disconnect indication structure.

```
#include <l2c_api.h>
```

Collaboration diagram for l2cCocDataCnf_t:



Data Fields

- wsfMsgHdr_t hdr

  Header structure.
- uint16_t cid

  Local channel ID.

2.3.1 Detailed Description

Connection oriented channel disconnect indication structure.

Definition at line 141 of file l2c_api.h.

The documentation for this struct was generated from the following file:

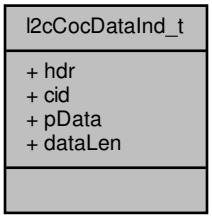
- /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/ble-host/include/l2c_api.h

2.4 l2cCocDataInd_t Struct Reference

Connection oriented channel data indication structure.

```
#include <l2c_api.h>
```

Collaboration diagram for l2cCocDataInd_t:



Data Fields

- wsfMsgHdr_t hdr

  Header structure.
- uint16_t cid

  Local channel ID.
- uint8_t * pData

  Pointer to packet data.
- uint16_t dataLen

  packet data length

2.4.1 Detailed Description

Connection oriented channel data indication structure.

Definition at line 132 of file l2c_api.h.

The documentation for this struct was generated from the following file:

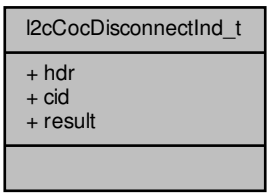
- /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/ble-host/include/l2c_api.h

2.5 l2cCocDisconnectInd_t Struct Reference

Connection oriented channel disconnect indication structure.

```
#include <l2c_api.h>
```

Collaboration diagram for l2cCocDisconnectInd_t:



Data Fields

- wsfMsgHdr_t hdr

  Header structure.
- uint16_t cid

  Local channel ID.
- uint16_t result

  Connection failure result code.

2.5.1 Detailed Description

Connection oriented channel disconnect indication structure.

Definition at line 124 of file l2c_api.h.

The documentation for this struct was generated from the following file:

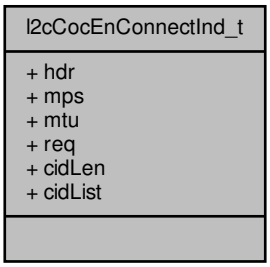
- /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/ble-host/include/l2c_api.h

2.6 l2cCocEnConnectInd_t Struct Reference

Enhanced connection oriented channel connect indication structure.

```
#include <l2c_api.h>
```

Collaboration diagram for l2cCocEnConnectInd_t:



Data Fields

- wsfMsgHdr_t hdr

  Header structure.
- uint16_t mps

  Data packet MPS peer can receive.
- uint16_t mtu

  Data packet MTU peer can receive.
- bool_t req

  TRUE if indicating a request, else a response.
- uint8_t cidLen

  Number of channels in cidList.
- uint16_t cidList [L2C_MAX_EN_CHAN]

  Local channel ID list.

2.6.1 Detailed Description

Enhanced connection oriented channel connect indication structure.

Definition at line 148 of file l2c_api.h.

The documentation for this struct was generated from the following file:

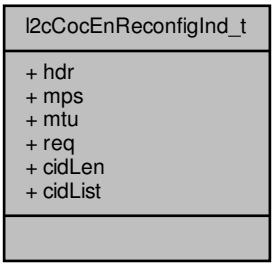
- /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/ble-host/include/l2c_api.h

2.7 l2cCocEnReconfigInd_t Struct Reference

Enhanced connection oriented channel reconfiguration indication structure.

```
#include <l2c_api.h>
```

Collaboration diagram for l2cCocEnReconfigInd_t:



Data Fields

- wsfMsgHdr_t hdr
  Header structure.
- uint16_t mps
  Data packet MPS.
- uint16_t mtu
  Data packet MTU.
- bool_t req
  TRUE if indicating a request, else a response.
- uint8_t cidLen
  Number of channels in cidList.
- uint16_t cidList [L2C_MAX_EN_CHAN]
  Local channel ID list.

2.7.1 Detailed Description

Enhanced connection oriented channel reconfiguration indication structure.

Definition at line 159 of file l2c_api.h.

The documentation for this struct was generated from the following file:

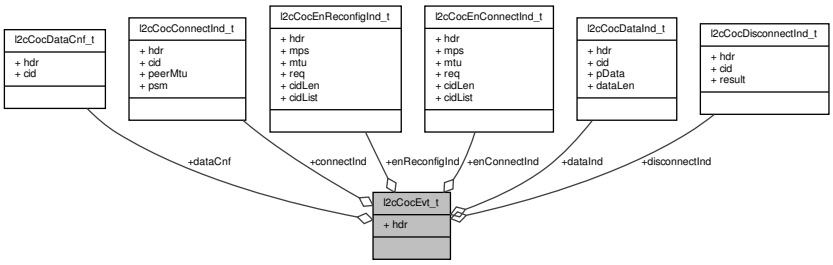
- /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/ble-host/include/l2c_api.h

2.8 l2cCocEvt_t Union Reference

Connection oriented channel event structure.

```
#include <l2c_api.h>
```

Collaboration diagram for l2cCocEvt_t:



Data Fields

- wsfMsgHdr_t hdr

  Header structure.
- l2cCocConnectInd_t connectInd

  Channel connect indication.
- l2cCocDisconnectInd_t disconnectInd

  Channel disconnect indication.
- l2cCocDataInd_t dataInd

  Received data indication.
- l2cCocDataCnf_t dataCnf

  Transmit data confirm.
- l2cCocEnConnectInd_t enConnectInd

  Enhanced channel connect indication.
- l2cCocEnReconfigInd_t enReconfigInd

  Enhanced channel reconfigure indication.

2.8.1 Detailed Description

Connection oriented channel event structure.

Connection oriented channel callback header parameters:

Parameters

hdr.event	Callback event
hdr.param	DM connection ID
hdr.status	Event status (L2C_COC_DATA_CNF only)

Definition at line 178 of file l2c_api.h.

The documentation for this union was generated from the following file:

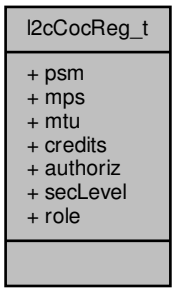
- /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/ble-host/include/l2c_api.h

2.9 l2cCocReg_t Struct Reference

Connection oriented channel registration structure.

`#include <l2c_api.h>`

Collaboration diagram for l2cCocReg_t:



Data Fields

- uint16_t psm

  Protocol service multiplexer.
- uint16_t mps

  Maximum receive PDU fragment size.
- uint16_t mtu

  Maximum receive data packet size.
- uint16_t credits

  Data packet receive credits for this channel.
- bool_t authoriz

  TRUE if authorization is required.
- uint8_t secLevel

  Channel minimum security level requirements.
- uint8_t role

  Channel initiator/acceptor role.

2.9.1 Detailed Description

Connection oriented channel registration structure.

Definition at line 103 of file l2c_api.h.

The documentation for this struct was generated from the following file:

- /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/ble-host/include/l2c_api.h

Chapter 3

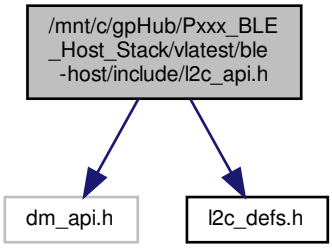
File Documentation

3.1 /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/ble-host/include/l2c_api.h File Reference

L2CAP subsystem API.

```
#include "dm_api.h"
#include "l2c_defs.h"
```

Include dependency graph for l2c_api.h:



Data Structures

- struct l2cCocReg_t

  Connection oriented channel registration structure.
- struct l2cCocConnectInd_t

  Connection oriented channel connect indication structure.
- struct l2cCocDisconnectInd_t

  Connection oriented channel disconnect indication structure.
- struct l2cCocDataInd_t

  Connection oriented channel data indication structure.
- struct l2cCocDataCnf_t

Connection oriented channel disconnect indication structure.

- struct l2cCocEnConnectInd_t

  Enhanced connection oriented channel connect indication structure.

- struct l2cCocEnReconfigInd_t

  Enhanced connection oriented channel reconfiguration indication structure.

- union l2cCocEvt_t

  Connection oriented channel event structure.

- struct l2cCfg_t

  Configurable parameters.

Macros

- #define L2C_COC_REG_ID_NONE 0

  Invalid channel registration ID for connection oriented channels.

- #define L2C_COC_CID_NONE 0

  Invalid channel ID for connection oriented channels.

- #define L2C_SIGNAL_ID_INVALID 0

  Invalid signal identifier.

L2CAP Control Callback Events

Control callback message events

- #define L2C_CTRL_FLOW_ENABLE_IND 0

  Data flow enabled.

- #define L2C_CTRL_FLOW_DISABLE_IND 1

  Data flow disabled.

L2CAP COC Channel Roles

Connection oriented channel initiator/acceptor role

- #define L2C_COC_ROLE_NONE 0x00

  No role (unallocated)

- #define L2C_COC_ROLE_INITIATOR 0x01

  Channel initiator.

- #define L2C_COC_ROLE_ACCEPTOR 0x02

  Channel acceptor.

L2CAP COC Data Confirm Codes

Connection oriented channel data confirm status values

- #define L2C_COC_DATA_SUCCESS 0

  Data request successful.

- #define L2C_COC_DATA_ERR_MEMORY 1

  Out of memory.

- #define L2C_COC_DATA_ERR_OVERFLOW 2

  Transaction overflow.

Typedefs

- typedef uint16_t l2cCocRegId_t

  Connection oriented channel registration ID.
- typedef void(∗ l2cDataCback_t) (uint16_t handle, uint16_t len, uint8_t ∗pPacket)

  This callback function sends a received L2CAP packet to the client.
- typedef void(∗ l2cCtrlCback_t) (wsfMsgHdr_t ∗pMsg)

  This callback function sends control messages to the client.
- typedef void(∗ l2cCocCback_t) (l2cCocEvt_t ∗pMsg)

  This callback function sends data and other events to connection oriented channels clients.
- typedef uint8_t(∗ l2cCocAcceptCb_t) (dmConnId_t connId, uint8_t numChans)

  This callback function asks clients of connection oriented channels if a given number of channels can be created on the PSM.
- typedef uint16_t(∗ l2cCocAuthorCback_t) (dmConnId_t connId, l2cCocRegId_t regId, uint16_t psm)

  This callback function is used for authoriztion of connection oriented channels.

Functions

L2CAP Initialization

Initialization and registration functions

- void L2cInit (void)

  Initialize L2C subsystem.
- void L2cMasterInit (void)

  Initialize L2C for operation as a Bluetooth LE master.
- void L2cSlaveInit (void)

  Initialize L2C for operation as a Bluetooth LE slave.

L2CAP CID Functions

Register and send data over a CID

- void L2cRegister (uint16_t cid, l2cDataCback_t dataCback, l2cCtrlCback_t ctrlCback)

  called by the L2C client, such as ATT or SMP, to register for the given CID.
- void L2cDataReq (uint16_t cid, uint16_t handle, uint16_t len, uint8_t ∗pL2cPacket)

  Send an L2CAP data packet on the given CID.
- void L2cDmSigReq (uint16_t handle, uint8_t code, uint16_t len, uint8_t ∗pParam)

  Build and send a signaling packet.

L2CAP COC Functions

Connection Oriented Channels Functions

- void L2cCocInit (void)

  Initialize L2C connection oriented channel subsystem.
- l2cCocRegId_t L2cCocRegister (l2cCocCback_t cback, l2cCocReg_t ∗pReg)

  Register to use a connection oriented channel, as either a channel acceptor, initiator, or both. If registering as channel acceptor then the PSM is specified. After registering a connection can be established by the client using this registration instance.
- void L2cCocDeregister (l2cCocRegId_t regId)

  Deregister and deallocate a connection oriented channel registration instance. This function should only be called if there are no active channels using this registration instance.
- void L2cCocSetAcceptCback (l2cCocRegId_t regId, l2cCocAcceptCb_t cback)

  Set the channel accept callback.
- uint16_t L2cCocConnectReq (dmConnId_t connId, l2cCocRegId_t regId, uint16_t psm)

  Initiate a connection to the given peer PSM.

- void L2cCocDisconnectReq (uint16_t cid)

  Disconnect the channel for the given CID.
- void L2cCocDataReq (uint16_t cid, uint16_t len, uint8_t *pPayload)

  Send an L2CAP data packet on the given connection oriented CID.
- bool_t L2cCocEnhancedConnectReq (dmConnId_t connId, l2cCocRegId_t regId, uint16_t psm, uint16_t credits, uint8_t numChan)

  Send a request to open enhanced credit based channels.
- bool_t L2cCocEnhancedReconfigReq (dmConnId_t connId, uint16_t mtu, uint16_t mps, uint8_t numChan, uint16_t *pChanList)

  Send a request to reconfigure enhanced credit based channels.
- void L2cCocErrorTest (uint16_t result)

  For testing purposes only.
- void L2cCocCreditSendTest (uint16_t cid, uint16_t credits)

  For testing purposes only.

L2CAP Connection Parameter Update Functions

- void L2cDmConnUpdateReq (uint16_t handle, hciConnSpec_t *pConnSpec)

  For internal use only. This function is called by DM to send an L2CAP connection update request.
- void L2cDmConnUpdateRsp (uint8_t identifier, uint16_t handle, uint16_t result)

  For internal use only. This function is called by DM to send an L2CAP connection update response.

Variables

L2CAP Configuration Structure

Pointer to structure containing initialization details of the L2CAP Subsystem. To be configured by Application.

- l2cCfg_t * pL2cCfg

  Configuration pointer.

L2CAP COC Callback Events

Connection oriented channel callback events.

- #define L2C_COC_CBACK_START 0x70

  L2C callback event starting value.
- #define L2C_COC_CBACK_CBACK_END L2C_COC_DATA_CNF

  L2C callback event ending value.
- enum {
  L2C_COC_CONNECT_IND = L2C_COC_CBACK_START,
  L2C_COC_DISCONNECT_IND,
  L2C_COC_EN_CONNECT_IND,
  L2C_COC_EN_RECONFIG_IND,
  L2C_COC_DATA_IND,
  L2C_COC_DATA_CNF }

  COC callback events.

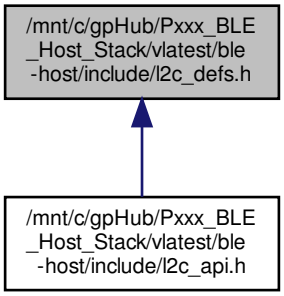
3.1.1 Detailed Description

L2CAP subsystem API.

3.2 /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/ble-host/include/l2c_defs.h File Reference

L2CAP constants and definitions from the Bluetooth specification.

This graph shows which files directly or indirectly include this file:



Macros

- #define L2C_MAX_EN_CHAN 5

  Max number of channels per enhanced connection request.
- #define L2C_PAYLOAD_START (HCI_ACL_HDR_LEN + L2C_HDR_LEN)

  Start of L2CAP payload in an HCI ACL packet buffer.
- #define L2C_SIG_PKT_BASE_LEN (HCI_ACL_HDR_LEN + L2C_HDR_LEN + L2C_SIG_HDR_LEN)

  L2CAP signaling packet base length, including HCI header.
- #define L2C_LE_SDU_PKT_BASE_LEN (HCI_ACL_HDR_LEN + L2C_HDR_LEN + L2C_LE_SDU_HDR↩ _LEN)

L2CAP LE SDU packet base length, including HCI header.

- #define L2C_SIG_RSP_FLAG 0x01

  Signaling response code flag.

L2CAP Packet Constants

- #define L2C_HDR_LEN 4

  L2CAP packet header length.
- #define L2C_MIN_MTU 23

  Minimum packet payload MTU for LE.
- #define L2C_SIG_HDR_LEN 4

  L2CAP signaling command header length.
- #define L2C_LE_SDU_HDR_LEN 2

  L2CAP LE SDU data header length.

L2CAP Parameter Lengths

Signaling packet parameter lengths

- #define L2C_SIG_CONN_UPDATE_REQ_LEN 8

  Connection update request length.
- #define L2C_SIG_CONN_UPDATE_RSP_LEN 2

  Connection update response length.
- #define L2C_SIG_CMD_REJ_LEN 2

  Command reject length.
- #define L2C_SIG_DISCONN_REQ_LEN 4

  Disconnection request length.
- #define L2C_SIG_DISCONN_RSP_LEN 4

  Disconnection response length.
- #define L2C_SIG_LE_CONN_REQ_LEN 10

  LE connection request length.
- #define L2C_SIG_LE_CONN_RSP_LEN 10

  LE connection response length.
- #define L2C_SIG_FLOW_CTRL_CREDIT_LEN 4

  Flow control credit lenghth.
- #define L2C_SIG_EN_CONNECT_REQ_LEN 8

  Enhanced credit based connection request.
- #define L2C_SIG_EN_CONNECT_RSP_LEN 8

  Enhanced credit based connection response.
- #define L2C_SIG_EN_RECONFIG_REQ_LEN 4

  Enhanced credit based reconfiguration request.
- #define L2C_SIG_EN_RECONFIG_RSP_LEN 2

  Enhanced credit based reconfiguration response.

L2CAP Connection Identifiers

BLE Defined Connection Identifiers (CID)

- #define L2C_CID_ATT 0x0004

  CID for attribute protocol.
- #define L2C_CID_LE_SIGNALING 0x0005

  CID for LE signaling.
- #define L2C_CID_SMP 0x0006

  CID for security manager protocol.

L2CAP Signaling Codes

- #define L2C_SIG_CMD_REJ 0x01

Comand reject.

- #define L2C_SIG_DISCONNECT_REQ 0x06
  Disconnect request.
- #define L2C_SIG_DISCONNECT_RSP 0x07
  Disconnect response.
- #define L2C_SIG_CONN_UPDATE_REQ 0x12
  Connection parameter update request.
- #define L2C_SIG_CONN_UPDATE_RSP 0x13
  Connection parameter update response.
- #define L2C_SIG_LE_CONNECT_REQ 0x14
  LE credit based connection request.
- #define L2C_SIG_LE_CONNECT_RSP 0x15
  LE credit based connection response.
- #define L2C_SIG_FLOW_CTRL_CREDIT 0x16
  LE flow control credit.
- #define L2C_SIG_EN_CONNECT_REQ 0x17
  Enhanced credit based connection request.
- #define L2C_SIG_EN_CONNECT_RSP 0x18
  Enhanced credit based connection response.
- #define L2C_SIG_EN_RECONFIG_REQ 0x19
  Enhanced credit based reconfiguration request.
- #define L2C_SIG_EN_RECONFIG_RSP 0x1A
  Enhanced credit based reconfiguration response.

L2CAP Command Rejection Codes

BLE defined Command rejection reason codes

- #define L2C_REJ_NOT_UNDERSTOOD 0x0000
  Command not understood.
- #define L2C_REJ_MTU_EXCEEDED 0x0001
  Signaling MTU exceeded.
- #define L2C_REJ_INVALID_CID 0x0002
  Invalid CID in request.

L2CAP Connection Parameter Update Result Codes

BLE defined result codes

- #define L2C_CONN_PARAM_ACCEPTED 0x0000
  Connection parameters accepted.
- #define L2C_CONN_PARAM_REJECTED 0x0001
  Connection parameters rejected.

L2CAP Connection Result Codes

BLE defined result codes

- #define L2C_CONN_SUCCESS 0x0000
  Connection successful.
- #define L2C_CONN_NONE 0x0001
  No connection result value available.
- #define L2C_CONN_FAIL_PSM 0x0002
  Connection refused LE_PSM not supported.
- #define L2C_CONN_FAIL_RES 0x0004
  Connection refused no resources available.
- #define L2C_CONN_FAIL_AUTH 0x0005
  Connection refused insufficient authentication.
- #define L2C_CONN_FAIL_AUTHORIZ 0x0006
  Connection refused insufficient authorization.

- #define L2C_CONN_FAIL_KEY_SIZE 0x0007
  Connection refused insufficient encryption key size.
- #define L2C_CONN_FAIL_ENC 0x0008
  Connection Refused insufficient encryption.
- #define L2C_CONN_FAIL_INVALID_SCID 0x0009
  Connection refused invalid source CID.
- #define L2C_CONN_FAIL_ALLOCATED_SCID 0x000A
  Connection refused source CID already allocated.
- #define L2C_CONN_FAIL_UNACCEPT_PARAM 0x000B
  Connection refused unacceptable parameters.
- #define L2C_CONN_FAIL_INVALID_PARAM 0x000C
  Connection refused invalid parameters.

L2CAP Interal Connection Result Codes

Proprietary codes not sent in any L2CAP packet.

- #define L2C_CONN_FAIL_TIMEOUT 0xF000
  Request timeout.

L2CAP Signaling Parameter Value Ranges

- #define L2C_PSM_MIN 0x0001
  PSM minimum.
- #define L2C_PSM_MAX 0x00FF
  PSM maximum.
- #define L2C_CID_DYN_MIN 0x0040
  CID dynamic minimum.
- #define L2C_CID_DYN_MAX 0x007F
  CID dynamic maximum.
- #define L2C_MTU_MIN 0x0017
  MTU minimum.
- #define L2C_MPS_MIN 0x0017
  MPS minimum.
- #define L2C_MPS_MAX 0xFFFD
  MPS maximum.
- #define L2C_CREDITS_MAX 0xFFFF
  Credits maximum.

L2CAP Enhanced Connection Reconfigure Result Codes

- #define L2C_RECONFIG_FAIL_MTU 0x0001
  Enhanced Reconfiguration refuded - cannot reduce MTU.
- #define L2C_RECONFIG_FAIL_MPS 0x0002
  Enhanced Reconfiguration refuded - cannot reduce MPS on more than one channel.
- #define L2C_RECONFIG_FAIL_CID 0x0003
  Enhanced Reconfiguration refuded - invalid CID.
- #define L2C_RECONFIG_FAIL_PARAM 0x0004
  Enhanced Reconfiguration refuded - unacceptable parameters.

3.2.1 Detailed Description

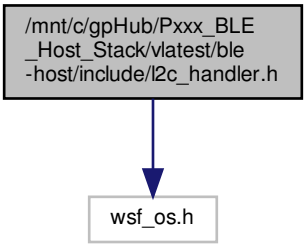
L2CAP constants and definitions from the Bluetooth specification.

3.3 /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/ble-host/include/l2c_handler.h File Reference

L2CAP handler interface.

```
#include "wsf_os.h"
```

Include dependency graph for l2c_handler.h:



Functions

L2CAP Event Handling

Message passing interface to L2CAP from other tasks through WSF.

- void L2cSlaveHandlerInit (wsfHandlerId_t handlerId)
  Event handler initialization function for L2C when operating as a slave.
- void L2cSlaveHandler (wsfEventMask_t event, wsfMsgHdr_t *pMsg)
  The WSF event handler for L2C when operating as a slave.
- void L2cCocHandlerInit (wsfHandlerId_t handlerId)
  Event handler initialization function for L2C with connection oriented channels.
- void L2cCocHandler (wsfEventMask_t event, wsfMsgHdr_t *pMsg)
  The WSF event handler for L2C with connection oriented channels.

3.3.1 Detailed Description

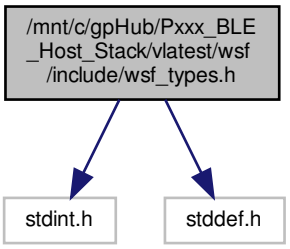
L2CAP handler interface.

3.4 /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/wsf_types.h File Reference

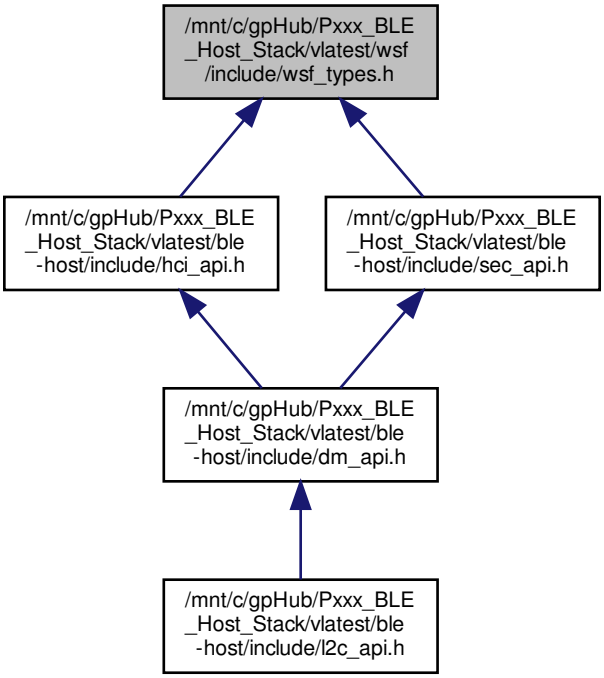
Platform-independent data types.

```
#include <stdint.h>
#include <stddef.h>
```

Include dependency graph for wsf_types.h:



This graph shows which files directly or indirectly include this file:



Macros

Integer Data Types

- #define **bool_t** uint8_t
- #define **FALSE** 0
- #define **TRUE** (!FALSE)
- #define **UINT64_C**(x) x##ULL
- #define **UINT32_C**(x) x##UL
- #define **UINT8_C**(x) (x)

3.4.1 Detailed Description

Platform-independent data types.

Index