

# **LTP Interface**

Specification

Release r09

---

**Table of contents**

1	Introduction .....	7
2	Glossary .....	7
3	LTP Protocol Description .....	10
3.1	Introduction .....	10
3.2	Packet Syntax Description .....	11
3.3	Syntax Example → ExampleMessage .....	12
3.4	Rules for LTP Message Handling .....	13
3.4.1	Handling of unknown optional parameters: .....	13
3.4.2	Handling of unknown messages .....	13
3.4.3	Definition of optional fields .....	14
3.4.4	MDL creation and negotiation of QoS configuration .....	14
3.4.5	Security Procedures .....	15
3.4.6	Handling of malformed messages .....	15
3.4.7	System Startup .....	15
3.4.8	Flow control .....	15
3.4.9	Data Handling .....	17
3.4.10	Message CRC checking .....	18
3.5	MDC Usage and Procedures .....	19
3.5.1	System Startup .....	19
3.5.2	Incoming new MDL connections .....	19
3.5.3	Outgoing new MDL connections .....	20
3.5.4	APDU data exchange .....	20
3.5.5	disconnection .....	21
3.6	Message Field Descriptions .....	22
3.6.1	QoS parameter for MDL negotiation .....	22
3.6.2	Causes .....	22
3.6.3	SSP Event type .....	23
3.6.4	Link key Type .....	23
3.7	Application Management and Configuration Access .....	24

---

3.7.1	Exit LTP Request → ExitReq .....	24
3.7.2	Exit LTP Response → ExitRsp .....	25
3.7.3	Exit LTP Information → ExitInfo .....	26
3.7.4	Reset Request → ResetReq.....	27
3.7.5	Reset Response → ResetRsp .....	28
3.7.6	Configuration Tunnel Request → ConfigTunnelReq .....	29
3.7.7	Configuration Tunnel Information → ConfigTunnelInfo.....	29
3.7.8	Configuration Tunnel Response → ConfigTunnelRsp.....	30
3.8	System Activation, Configuration and Status.....	31
3.8.1	LTP Active information → ActInfo .....	31
3.8.2	Radio Mode Set Request → RadioModeSetReq.....	32
3.8.3	Radio Mode Set Response → RadioModeSetRsp .....	33
3.8.4	Internal Event Info → InternalEventInfo .....	34
3.9	Connection Management.....	37
3.9.1	Channel create request → ConnectMDLReq .....	37
3.9.2	Channel create response → ConnectMDLRsp.....	39
3.9.3	Channel reconnect request → ReconnectMDLReq.....	40
3.9.4	Channel reconnect response → ReconnectMDLRsp .....	41
3.9.5	Channel reconnect indication → ReconnectMDLInd .....	42
3.9.6	Channel reconnect confirmation → ReconnectMDLCnf .....	43
3.9.7	channel disconnect request → DisconnectMDLReq .....	44
3.9.8	Channel disconnect response → DisconnectMDLRsp .....	45
3.9.9	channel disconnect indication → DisconnectMDLInd.....	46
3.9.10	Channel disconnect confirmation → DisconnectMDLCnf .....	47
3.10	MDL / MCL / ACL Management .....	48
3.10.1	new channel create indication → CreateMDLInd.....	48
3.10.2	new channel create confirmation → CreateMDLCnf.....	49
3.10.3	Channel connect information → ConnectMDLInfo.....	50
3.10.4	channel delete information → DeleteMDLInfo .....	51
3.10.5	MCL Status Info → MCLStatusInfo .....	52

---

3.10.6	ACL Status Info → ACLStatusInfo .....	53
3.11	Security Management .....	54
3.11.1	Pairable Mode set Request → PairableModeSetReq .....	54
3.11.2	Pairable Mode set Response → PairableModeSetRsp .....	56
3.11.3	Authentication Request → AuthReq.....	57
3.11.4	Authentication Response → AuthRsp .....	57
3.11.5	Authentication Request Indication → AuthRequestInd .....	58
3.11.6	Authentication Request Confirmation → AuthRequestCnf.....	59
3.11.7	User Confirmation Request Indication → UserConfRequestInd .....	60
3.11.8	User Confirmation Request Confirmation → UserConfRequestCnf.	60
3.11.9	Passkey Request Indication → PasskeyRequestInd .....	61
3.11.10	Passkey Request Confirmation → PasskeyRequestCnf.....	61
3.11.11	Passkey Request reply Request → PasskeyReqReplyReq .....	62
3.11.12	Passkey Request reply Response → PasskeyReqReplyRsp.....	63
3.11.13	Passkey Notification Information → PasskeyNotificationInfo.....	64
3.11.14	Keypress Notification Request → KeypressNotificationReq.....	64
3.11.15	Keypress Notification Response → KeypressNotificationRsp.....	65
3.11.16	Keypress Notification Information → KeypressNotificationInfo .....	65
3.11.17	Remote OOB-Data Request Indication → RemoteOOBReqInd .....	66
3.11.18	Remote OOB-Data Request confirmation → RemoteOOBReqCnf..	67
3.11.19	Local OOB-Data Request → LocalOOBReq .....	67
3.11.20	Local OOB-Data Response → LocalOOBRsp.....	68
3.11.21	Authentication result Indication → AuthResultInd.....	69
3.11.22	Authentication result Confirmation → AuthResultCnf .....	70
3.11.23	Authentication result Request Ind → AuthResultRequestInd.....	71
3.11.24	Authentication result Request Conf → AuthResultRequestCnf.....	72
3.11.25	Authentication delete Request → AuthDeleteReq.....	72
3.11.26	Authentication delete Response → AuthDeleteRsp .....	73
3.11.27	Authentication list Request → AuthListReq.....	74
3.11.28	Authentication list Information → AuthListInfo .....	75

---

3.11.29	Authentication list Response → AuthListRsp .....	76
3.11.30	Authorization request Indication → AuthorizationReqInd.....	77
3.11.31	Authorization request Confirmation → AuthorizationReqCnf .....	78
3.12	Data exchange.....	79
3.12.1	Unsegmented Application Data Packet → DataUnsegmented .....	79
3.12.2	Start of Segmented Application Data Packet → DataStartSegment	81
3.12.3	End of Segmented Application Data Packet → DataEndSegment ..	83
3.12.4	Continuation of Segmented Data Packet → DataContinueSegment	85
3.13	SDP Record / MDEP management .....	87
3.13.1	Register HDP MDEP Request → RegisterHDPMDEPReq.....	87
3.13.2	Register HDP MDEP Response → RegisterHDPMDEPRsp .....	88
3.13.3	Register SPP MDEP Request → RegisterSPPMDEPReq.....	88
3.13.4	Register SPP MDEP Response → RegisterSPPMDEPRsp .....	90
3.13.5	Release MDEP Request → ReleaseMDEPReq .....	91
3.13.6	Release MDEP Response → ReleaseMDEPRsp.....	92
3.14	Inquiry, DID information and name discovery management .....	93
3.14.1	Inquiry Request → InquiryReq .....	93
3.14.2	Inquiry Response → InquiryRsp.....	94
3.14.3	Inquiry Device Info → InquiryDeviceInfo .....	95
3.14.4	Device name Request → DeviceNameReq.....	96
3.14.5	Device name Response → DeviceNameRsp .....	96
3.14.6	DID Device Info → DIDDeviceInfo .....	97
3.15	HDP service discovery management .....	99
3.15.1	HDP Discovery Request → HDPDiscoveryReq.....	99
3.15.2	HDP Discovery Response → HDPDiscoveryRsp .....	100
3.15.3	HDP Service Info → HDPServiceInfo.....	101
3.15.4	HDP Endpoint Info → HDPEndpointInfo .....	102
3.16	SPP service discovery management.....	103
3.16.1	SPP Discovery Request → SPPDiscoveryReq .....	103
3.16.2	SPP Discovery Response → SPPDiscoveryRsp.....	104

---

3.16.3	SPP Endpoint Info → SPPEndpointInfo .....	105
4	Message flow examples .....	106
4.1	Inquiry .....	106
4.2	HDP Service Discovery .....	107
4.3	Outgoing new HDP channel connection establishment .....	108
4.4	Incoming new HDP channel connection establishment .....	109
4.5	Outgoing permanent disconnection of HDP data channel .....	110
4.6	Incoming permanent disconnection of HDP data channel .....	111
4.7	Outgoing temporary disconnection of HDP data channel .....	112
4.8	Incoming temporary disconnection of HDP data channel .....	113
4.9	Outgoing HDP session with data exchange .....	114
4.10	Incoming HDP session with data exchange .....	115
4.11	SPP Service Discovery .....	116
4.12	Outgoing new SPP channel connection establishment .....	117
4.13	Incoming new SPP channel connection establishment .....	118
4.14	Outgoing disconnection of SPP data channel .....	119
4.15	Incoming disconnection of SPP data channel .....	120
4.16	Successful active Bonding .....	121
4.17	Successful passive Bonding .....	122
4.18	Pairing procedure: Legacy (PIN entry) .....	123
4.19	Pairing procedure: Just Works .....	124
4.20	Pairing Procedure: Numeric Comparison (Local "DisplayYesNo") .....	125
4.21	Pairing Procedure: Passkey Entry (Local Display) .....	126
4.22	Pairing Procedure: Passkey Entry (Local KeyboardOnly) .....	127
5	References .....	128
6	History .....	129
7	Appendix A: All LTP Messages sorted by Opcode .....	130
8	Appendix B: All LTP Messages sorted by Name .....	132

---

## 1 Introduction

This document describes the interface between the vendor host system (also called MDH, Medical Device Host) and the LTP capable Bluetooth Module (also called MDC, Medical Device Controller). The interface is called Local Transport Protocol (LTP). This document defines LTP Version 1.3.

The physical interface of the MDH towards the MDC is a serial UART interface. The UART interface is used to transport the LTP, the protocol enables the MDH and MDC to communicate with each other, establish and terminate connections and exchange data frames while a Bluetooth connection is established. Additionally the UART interface can be used to configure various settings via the “configurator command set” as described in [X1].

## 2 Glossary

### Configurator

The configurator command set is a Stollmann proprietary control for all sorts of module configuration related functions. For further information about this command set please refer to [X1].

### IEEE 11073

The IEEE 11073 protocol refers to the Data Exchange Protocol IEEE 11073-20601 in conjunction with the Device Data Specializations IEEE 11073-10xxx. These facilitate interoperability between sources and recipients of device data minimizing the need for proprietary drivers on either side to format and interpret the data.

The IEEE 11073 “Data Exchange Protocol” refers to the following specifications:

[X3]/[X4]/[X5]

Relative to the Data Exchange Protocol specifications defined above, device data specializations refer to, but are not limited to, the following specifications:

[X6]/[X7]/[X8] / [X9] / [X10] / [X11]

The documents listed above are for reference only. Please refer to the Bluetooth Assigned Numbers list [X12] for a complete listing of applicable specifications.

### LTP

The Local Transport Protocol (LTP) is a proprietary protocol and is designed to carry data frames, or segments of data frames via an asynchronous serial interface. It also includes mechanisms to ensure data integrity on this interface.

---

## MCAP

The Multi-Channel Adaptation Protocol (MCAP) has been designed to support the requirements of the Health Device Profile but is expected to be useful for a number of other use models. For further information about MCAP please refer to [X2].

## MDC

The Medical Device Controller (MDC) of a medical device is a Stollmann LTP capable module and fulfills all “transport specific” functionality. This includes the handling of the MCAP [X2] protocol and all Bluetooth related functionality including service exposure and discovery, connection handling, security, ensuring data integrity and flow-control.

## MDH

The Medical Device Host (MDH) of a medical device is vendor specific and fulfills all “medical” related functionality like measurement of medical parameters and calculations.

## MDEP

A MCAP Data End Point (MDEP) represents the logical function of a device and includes such information as MDEP ID, MDEP Role (*source/sink*), data type (device data specialization) and description. If a given implementation wants to accept MDL connections it must register at least one MDEP. Multiple MDLs to a single MDEP are allowed.

## MDEP\_ID

A MDEP\_ID is an identifier that addresses a specific MDEP of a remote device.

## MDL

An MCAP Data Link (MDL) is a connection to an MDEP, and is explicitly created by one of the two participating devices. An MDL can be seen as a data channel that addresses two specific endpoints of a connection. Multiple MDLs to one or multiple endpoints to or from a single remote device are allowed.

---

## MDL\_ID

A MDL\_ID is an identifier that addresses a specific MDL. The LTP MDL\_ID differs from the MCAP MDL\_ID by the fact that it is system unique not only device unique for a given remote device. That means that an LTP MDL\_ID addresses a specific MDEP and(!) a specific remote device while an MCAP MDL\_ID addresses a MDEP of(!) a remote device. To use an MCAP MDL\_ID, the remote device must be provided as additional information while the LTP MDL\_ID already defines this information. Due to that, the LTP MDL\_ID is always referenced as *loc\_MDL\_ID* (local MDL\_ID) to point out that it is rather a locally unique representation of the MCAP MDL\_ID. This difference is completely transparent for the MDH, but has to be kept in mind if it comes to testing with other HDP parties since the “over the air” MCAP MDL\_ID might differ from the “over the UART” LTP MDL\_ID since it becomes translated MDC internal.

## HDP

The Health Device Profile (HDP) is an application profile that defines the requirements for qualified Bluetooth healthcare and fitness (referred to as ‘health’) device implementations. This profile is used for connecting application data *source* devices such as blood pressure monitors, weight scales, glucose meters, thermometers, and pulse ox meters to application data *sink* devices such as mobile phones, laptops, desktop computers, and health appliances without the need for cables. This profile makes use of the Multi-Channel Adaptation Protocol (MCAP).

## FSM

Finite State Machine.

## MITM

Man in the middle protection.

---

## 3 LTP Protocol Description

### 3.1 Introduction

The Health Device Profile (HDP) requires that packet boundaries of application data frames are maintained when they are transported via Bluetooth. This is due to the fact that medical applications expect that only complete protocol packets are carried via a transport. Since the HDP allows application data frames up to a size of 64kBytes but most embedded “transport” solutions (e.g. Bluetooth modules) can not store-and-forward this amount of data due to memory restrictions, the HDP mandates the use of the SAR (segmentation and re-assembly) functionality of the lower Bluetooth layers. To maintain packet boundaries while the packets are “locally” transported from the MDH to the MDC (and vice versa), the LTP protocol is used. As a result application data frames can be segmented and re-assembled by the MDH. The LTP grants and expects that once a LTP frame is started by the sender it will be completed before any other action (e.g. signaling or command) is initiated.

#### Example:

An HDP connection is established. Due to negotiation on Bluetooth level, lets assume the maximum Bluetooth PDU size for this connection is 300 byte for both directions.

Now, the vendor application of the remote device wants to send an application data frame of 400 byte. Since the Bluetooth PDU size for the connection is limited to 300 byte, the application data frame becomes segmented into two Bluetooth packets, lets assume one with 300 byte, and one with 100 byte payload.

The local MDC receives these two Bluetooth PDUs, wraps them into the LTP and forwards them via the local UART to the local MDH that carries the vendor application. The local vendor application can now re-assemble the original 400 byte application data frame by using the information contained in the LTP framing.

Beside the fact that the LTP allows the segmentation and re-assembly of application data frames up to a size of 64 kilobyte, it also simplifies the connection handling for the vendor application since it grants that all connection related signaling is performed on LTP packet level / occurs on LTP packet boundaries.

In the case that the Bluetooth connection is terminated while the second application data frame segment is transferred, the signaling of this event will happen after the first segment is forwarded via LTP to the MDH, no “incomplete” application data frame segment will / can be reported, once an LTP frame is started it will / has to be completed before any other action (e.g. indicating a disconnect) is performed.

---

### 3.2 Packet Syntax Description

**<parameter>** The “< >” brackets are used to define a mandatory field of an LTP packet. If the parameter represents a multibyte value, it is formatted most significant byte first (Motorola byte order)

**[parameter]** The “[ ]” brackets are used to define an optional field of an LTP packet. If the parameter represents a multibyte value, it is formatted most significant byte first (Motorola byte order)

**cmd** Defines a command opcode that specifies the command included in a specific LTP packet. Each LTP packet includes exactly one command opcode.

**copmsk** Stands for “cmd optional parameter mask” and defines a field that is used to identify which optional parameters for the command are included in this LTP packet.

For each optional 8bit parameter one bit is reserved in the copmsk of a command, so multiple optional parameters can be indicated by combining the corresponding bits. Each bit in this mask represents an optional one byte (8bit) parameter.

To represent an optional parameter with a size of more than one byte, a multiple amount of bits, representing the size of that parameter, are reserved. Due to this definition, the size of unknown optional parameters can be determined by “counting the bits” in the copmsk, so the unknown optional parameters can be ignored.

The order of optional parameters of a command within an LTP message is the same as the order of the corresponding bits in the copmsk with the least significant bit in the copmsk representing the first byte of the optional parameters.

If an optional parameter that is defined for a command is not included in a given LTP message, the corresponding bit(s) in the copmsk is (are) not set (set to 0) and no space is reserved in the LTP message (see example below).

**lp** Stands for “length packet” and defines a value that indicates the length of this LTP packet including all included headers, optional parameters and payload

### 3.3 Syntax Example → ExampleMessage

**Syntax example message:**

<cmd><copmsk><lp>[P1][P2][P3]<payload>

**Mandatory fields of this example message:**

Name	Size	Description
cmd	1	0xXX → ExampleMessage
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of LTP packet, up to 64kB
payload	N/A	Application data APDU contained in this LTP packet

**Optional fields of this example message:**

Name	msk	Description
P1	0x01	8 bit parameter
P2	0x06	16 bit parameter
P3	0x08	8 bit parameter

**Hexadecimal LTP example message dump:**

0xXX 0x0E 0x00 0x09 0x11 0x22 0x33 0x44 0x55

**Decoded LTP example message:**

cmd opcode = 0xXX  
 copmsk = 0x0E (3 bits set, so 3 byte optional parameter)  
 lp = 0x0009 (4 byte header + 3 byte optional parameter + 2 byte payload)  
 P1 = not included in this example message  
 P2 = 0x1122 (16Bit)  
 P3 = 0x33 (8Bit)  
 payload = 0x44 0x55 ( 2 byte)

---

### 3.4 Rules for LTP Message Handling

LTP is designed to be easily extended without violating some basic rules that allow legacy application to stay compatible to newer LTP implementations.

To allow this, LTP messages are structured in a way that allows a protocol peer to ignore unknown messages or unknown optional parameters.

#### 3.4.1 Handling of unknown optional parameters:

The optional parameters of a given LTP message are referenced by the “copmsk” field in the LTP message header. Basically each bit in the mask references to one byte of optional parameters. The meaning of this bit (and consequently the optional parameter it refers to) is defined in the section of this document describing the LTP message addressed by the “opcode” field of the message.

If the meaning of a optional parameter is not known to a given LTP implementation, the implementation can still determine the size and position of that optional parameter and using this “copmsk” information to ignore it.

#### 3.4.2 Handling of unknown messages

If a given LTP implementation receives a message with an <opcode> that can't be decoded by that implementation, that unknown opcode value still includes some information how to react to this message and the <lp> field of that unknown message enables the given LTP implementation to determine the overall size of the message, so the message data can be identified and handled or ignored in a controlled way.

If a LTP message with an unknown opcode is received and the most significant bit is set in that opcode, then a response to that message is strictly required, otherwise the message shall be ignored by the receiver.

If a response is required, than this response shall have the following structure:

- The <opcode> of the response shall be identical to the received unknown opcode with the exception that the most significant bits shall be set to zero
- The <copmsk> field shall be set to zero so no optional parameters shall be included in that response
- The <cause> field of the response shall be set to “LTP\_NotSupported”
- All other parameters of the received unknown message shall be ignored and truncated, so a response to an unknown LTP message has always a fixed length of 5 bytes in the <lp> field.

---

### 3.4.3 Definition of optional fields

This protocol description distinguishes between “mandatory” and “optional” parameter fields in messages. This is due to the fact that different use-cases for LTP require different capabilities from the MDC and MDH. If the handling of an “optional” field is mandated depends on the MDC implementation and context addressed. Please refer to the MDC documentation to gather information about mandated / supported “optional” fields.

### 3.4.4 MDL creation and negotiation of QoS configuration

To exchange data with a remote device an MDL must be used.

This can be done by creating an MDL (see command ConnectMDLReq or CreateMDLInd) or reconnecting an MDL (see command ReconnectMDLReq or ReconnectMDLInd) that was created previously and then temporary (see DisconnectMDLReq or DisconnectMDLInd) or unintentional disconnected (e.g. remote Peer became out of Range).

Part of the creation of an MDL is the negotiation of a given QoS configuration of that MDL (for valid configurations refer to chapter 3.6.1 QoS parameter for MDL negotiation).

For a new MDL the MDC automatically assigns a system unique local\_MDL\_ID in the range of 0x01 to 0xFE. Due to this, the local\_MDL\_ID is sufficient to address a specific MDL even if multiple MDLs and/or MCL are established in parallel.

This differs from the MDL ID concept of MCAP [X2] where an MDL ID is only unique within its MCL context.

An MDL is always connected to an MDEP “functional endpoint” (e.g. Thermometer) that is identified by an MDEP\_ID. Every endpoint includes a “functional role” (e.g. “source” or “sink”) that defines if it is the origin or the destination of the measurement data.

LTP (respectively the underlying MCAP) defines that the MDEP “source” of an MDL to be created is in charge of defining the QoS configuration of that MDL, regardless if it is the initiator or the acceptor of that MDL connection.

Due to that role an MDH must implement the appropriate handling of QoS negotiation:

If an MDH wants to connect to a MDEP of a remote peer, it depends on the role of that MDEP if the MDH shall define the QoS configuration for the MDL to create or indicate no preference.

If an MDH wants to accept an MDL from a remote peer it depends on the role of the local MDEP if the MDH is mandated to define the QoS configuration for the MDL to create or accept the configuration mandated by the remote peer.

---

### 3.4.5 Security Procedures

The HDP mandates that all data exchanges have to happen on secure, authenticated and encrypted Bluetooth ACL connections. To handle the necessary procedures, LTP defines messages for security handling (see chapter 3.11 Security Management). The security procedures might be initiated explicitly by the MDH or automatically while connection setup from the MDC or the remote device. Basically, the MDH must expect security procedures (e.g. UserAuthRequestInd, UserConfirmationRequestInd, UserAuthorizationReqInd, UserPasskeyReqInd or RemoteOOBDataReqInd messages) at any time, for an incoming connection it might be the first transaction initiated and even while an already established connection a (re-) authentication is possible.

If an authentication is started some MDH interaction can be required. This interaction might include displaying of decimal digits and forwarding keyboard input or returning a PIN code to the MDC. The exact requirements depend on the capabilities of the local (see PairableModeSetReq message) and the remote device. For further information please see [X14] and/or contact Stollmann.

### 3.4.6 Handling of malformed messages

If a given MDC implementation receives a malformed message (e.g. a Message that has an additional unknown , or misses a mandatory parameter) than this MDC will ignore that message and send an internalEventInfo message with event type "LTP\_MalformedMsgReceived" with cause "LTP\_CauseInvalidParameter".

### 3.4.7 System Startup

Once the MDC performed a complete startup after a power up, power cycle or reset, it will indicate the result of this startup with an ActInfo message (see 3.7.4).

**The MDH shall wait for this ActInfo message before initiating any other LTP transaction than a ResetReq message.**

### 3.4.8 Flow control

Since LTP allows communication via multiple MDL channels, it supports MDL channel based flow control that allows to provide back pressure to each MDL channel individually (e.g. on one MDL the data transfer is stopped by flow control while on an other MDL data can be still exchanged).

MDL connections support "credit based" flow control in downstream direction (from MDH to the MDC) and in upstream direction (from MDC to MDH).

**Please be aware that credit based flow control should be used by a MDH to perform channel based flow control . UART hardware flow control might**

---

**interfere with the LTP protocol and should consequently not be used for channel based flow control**

**An MDH shall support UART hardware flow control (RTS/CTS handshake).**

To enable credit based flow control for a given MDL, the MDH should assign one or more maxTPDUusCredits to the channel as part of the CreateMDLCnf message. If no maxTPDUusCredits are indicated in this message credit based flow control is disabled for that specific MDL.

If credit based flow control is enabled by the MDH for a given MDL, the ConnectMDLInfo message will include a maxTPDUusCredits and maxTPDUUsCredits parameter that indicates how many data segment messages (DataUnsegmented/DataStartSegment/DataEndSegment/DataContinueSegment) are allowed to be sent in any direction before the MDH/MDC has to wait for return credits from its peer MDC/MDH.

The MDC and MDH can return one or more credits as an optional parameter of data messages. The transfer of data messages with or without payload data including one or multiple return credits shall be used for this purpose. Data messages without payload do not require a transfer credit and hence can be used to return credits to the peer independent of flow control conditions in the forward direction.

Return credits are cumulative, that is, are added by the receiving peer.

It is not allowed to exceed the initial credit configuration on a link by returning an excessive number of return credits.

If an HDP based MDL is reconnected, the ConnectMDLInfo will include the same values as indicated for the initial creation of that MDL.

The maximum number of open credits granted shall not exceed the initial numbers given in the ConnectMDLInfo message.

This credit based flow control mechanism only applies to LTP data messages, LTP signaling messages are not affected and can be exchanged any time.

If the MDH violates the credit based flow control mechanism by sending data messages without permission or exceeding the initial number of granted/returned credits, the MDC will ignore the data message and reply with an internalEventInfo message with event type LTP\_InvalidDataReceived and cause LTP\_CauseFlowcontrolViolation.

---

**Example:**

- The MDC indicates an incoming connection by a CreateMDLInd message
- The MDH confirms this message with a CreateMDLCnf message including two maxTPDUUsCredits as optional parameter
- The MDC indicates that the MDL is established by an ConnectMDLInfo message including two maxTPDUUsCredits and two maxTPDUdsCredits.
- Now the MDH is allowed to send two data packets, each consumes one downstream credit. As soon as all downstream credits are consumed it has to stop sending additional data due to flow control condition.
- If the MDC has transmitted a data packet to a remote device it will return one returnCredit as optional parameter of its next data packet. If no data is to be send to the MDH, a data packet with no data payload will be used for this purpose.
- As soon as the MDH receives such return credits, it is allowed to send data packets as long as granted credits are not consumed completely again.

The same mechanism applies vice versa.

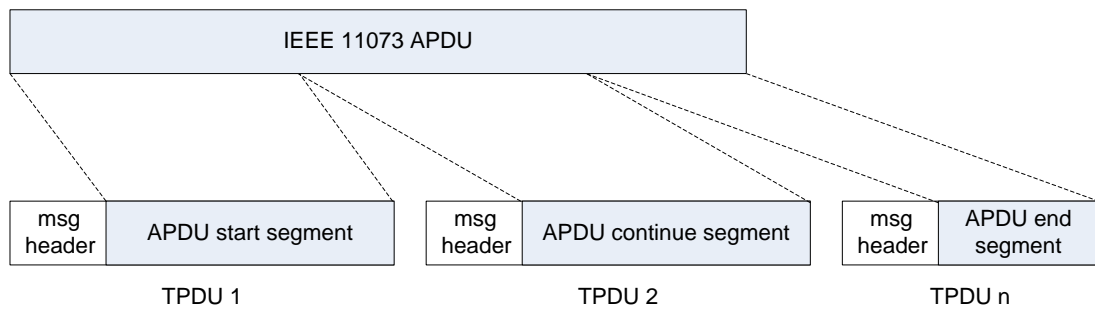
For additional information refer to the descriptions of the involved messages.

### **3.4.9 Data Handling**

To exchange IEEE 11073 APDUs via a given HDP based MDL connection that are larger than the signaled TPDU size of that MDL connection a segmentation and re-assembly mechanism has to be used/implemented.

To send such large 11073 APDU, that APDU has to be split into multiple data exchange messages that indicate if they provide the first data of a new APDU (start segment) continuation data of an already started APDU (continuation segment) or the last data of an already started or continued APDU (end segment).

Received large APDUs will be exchanged in segments with the same mechanisms.



If multiple MDL connections are active simultaneously segments of APDU can/will be signaled interleaved in both directions.

Since SPP does not define an APDU segmentation and reassembly mechanism, SPP based MDL connection support only unsegmented APDU exchange so the maximum APDU size for a given MDL is limited to the TPDU size of that MDL.

### 3.4.10 Message CRC checking

Each LTP message includes an optional CRC8 value that will be used by the MDC to check the LTP message header for correctness. If a CRC8 is part of an LTP message directed to the MDC and the message header is detected to be invalid by calculation of its CRC, the MDC assumes that communication with the MDH is corrupted and is out of sync. MDC then initiates an internalEventInfo message with event type LTP\_CommunicationOutOfSync and cause LTP\_CauseConnectionLost.

The CRC/FCS routines are based on ETSI TS 101 369 V6.3.0 (1999-03) GSM 07.10 version 6.3.0 Release 1997, B.3.5 Reversed CRC table.

Generator polynomial:  $x^{**8} + x^{**2} + x + 1$

---

## 3.5 MDC Usage and Procedures

### 3.5.1 System Startup

Once the MDC is powered up or performed a reset, it will run its internal initialization. Once this process is completed it will indicate that it is ready to communication via the “ActInfo” message. The MDH shall not send any message to the MDC until after this “ActInfo” message is send by the MDC.

### 3.5.2 Incoming new MDL connections

To allow a remote devices to discover and/or connect to the local device, the local device must be made “visible” and/or “connectable” on the air interface. In this context “visible” means that the local device can be discovered by an inquiry procedure of the remote device and “connectable” means that the local device at least allows a remote device to establish an ACL level point-to-point connection and responds to SDP,HDP or SPP connection attempts.

The “visibility” and “connectability” of the local device can be controlled with the command “RadioModeSetReq”.

To allow a remote device to connect on HDP or SPP profile level, a corresponding MDEP that provides all information necessary to access these profiles has to be registered. This service registration procedure can be performed with the RegHDPMDEPReq or RegSPPMDEPReq messages.

The first indication for an incoming HDP or SPP connection is, besides security related procedures (see chapter [3.4.5 Security Procedures]) and the target specific informal ACLStatusInfo and MCLStatusInfo messages, the CreateMDLInd message that has to be confirmed with a CreateMDLConf message. For the generation of this CreateMDLConf messages the information provided in chapter [3.6.1 QoS parameter for MDL negotiation] has to be obeyed.

If the MDL creation is completed successfully, a ConnectMDLInfo message will be generated by the MDC that includes all necessary informations to exchange data in that new MDL.

If the MDL creation was not successfully completed due to reasons of the local or remote device, the newly created MDL will be indicated to be deleted with a DeleteMDLInfo message.

For a graphical representation of this procedures see chapter [4 Message flow examples]

---

### 3.5.3 Outgoing new MDL connections

To connect to a remote device, that remote device must be discovered first. This discovery procedure can be initiated with the InquiryReq message.

Once a remote device is discovered or known otherwise, a service discovery has to be performed to gather all information necessary to perform a profile level connection to that remote device. This service discovery procedure can be initiated with the HDPDiscoveryReq and SPPDiscoveryReq messages.

To initiate a HDP or SPP profile level connection for data exchange, the gathered information can be used to initiate a ConnectMDLReq message. To generate this message the informations provided in chapter [3.6.1 QoS parameter for MDL negotiation] has to be obeyed.

Once a HDP or SPP profile MDL creation setup is initiated, the basic procedures to setup that new MDL connection are equal to the ones described in chapter [3.5.2 Incoming new MDL connections] with the only difference that at one point in time a ConnectMDLRsp message is generated by the MDC as a response to the initiating ConnectMDLReq message.

Depending on the result of the profile level connection attempt this ConnectMDLRsp message will be either generated as a direct response to the initiating ConnectMDLReq message (e.g. remote device is out of range) or following the CreateMDLInd/Rsp procedure (e.g. connection success, QoS negotiation process is started) but before the terminating ConnectMDLInfo or DeleteMDLInfo messages (see chapter [3.5.2 Incoming new MDL connections]).

For a graphical representation of this procedures see chapter [4 Message flow examples]

### 3.5.4 APDU data exchange

Once a ConnectMDLInfo message is received, the signaled MDL connection is ready to exchange data but expects compliance to the MDL parameters indicated with the ConnectMDLInfo message especially regarding APDU size, TPDU size and flow control. For more information see chapters [3.4.9 Data Handling] and [3.4.8 Flow control]

For a graphical representation of this procedures see chapter [4 Message flow examples]

---

### 3.5.5 disconnection

An established MDL connection can be terminated due to an intended disconnection from the local or remote device or an unintentional disconnect (e.g. remote device moved out of range).

In case the disconnection is intentionally initiated from the local device, the DisconnectMDLReq message can be used.

In all cases a DisconnectMDLInd will be the message of the MDC that indicates such a disconnection event. Once this message is received by the MDH, no new data exchange message shall be send by the MDH for that terminated MDL connection and a DisconnectMDLConf message shall be generated to complete MDL termination. Depending on the cause for the MDL disconnection and the capabilities of the used MDC implementation, a DeleteMDLInfo message can follow the described disconnect procedure.

For a graphical representation of this procedures see chapter [4 Message flow examples]

## 3.6 Message Field Descriptions

### 3.6.1 QoS parameter for MDL negotiation

This section describes the link configuration that are valid for this version of LTP.

For roles and procedures how to negotiate a LinkConfigType for a given MDL refer to chapter 3.4.4 MDL creation and negotiation of QoS configuration.

Type	Name	Description
0x00	LTP_LinkConfigDontCare	identifies “no preference” from the peer that initiates the transaction. This config type is not applicable for SPP/RFCOMM MDLs.
0x01	LTP_LinkConfigReliable	Identifies that a reliable “lossless” MDL is mandated. A MDL that is created with this channel type attribute shows a very high reliability but weak latency behavior This config type is not applicable for SPP/RFCOMM MDLs.
0x02	LTP_LinkConfigStreaming	Identifies that a streaming “lossy” MDL is mandated. A MDL that is created with this channel type attribute shows a well defined latency behavior but cannot guaranty reliability. This config type is not applicable for SPP/RFCOMM MDLs.
0x03	LTP_LinkConfigBasic	Identifies that a SPP/RFCOMM based MDL is mandated. An MDL that is created with this channel type attribute shows a basic reliability but weak latency behavior. This config type is not applicable for HDP/MCAP MDLs.

### 3.6.2 Causes

This section describes the causes that might be signaled while LTP operations in general. Please refer to the individual cause description for each message.

Cause	Name	Description
0x00	LTP_CauseSuccess	Indicates that an operation is completed successfully.
0x01	LTP_CauseAccept	Indicates that an operation is accepted.
0x02	LTP_CauseReject	Indicates that an operation is rejected.
0x03	LTP_CauseResourceError	Indicated that a operation can't be completed due to not enough resources
0x04	LTP_CauseInvalidParameter	Indicates that at least one parameter of a request is invalid, so the request is rejected
0x05	LTP_CauseInvalidState	Indicates that a requested operation can't be processed due to an MDC internal state transition. This shall never occur in normal operation
0x06	LTP_CauseConnectionDisconnect	Indicates that a connection is terminated intentionally.
0x07	LTP_CauseConnectionPaused	Indicates that a connection is terminated temporarily.
0x08	LTP_CauseConnectionLost	Indicates that a connection is terminated due to RF link loss.

0x09	LTP_CauseAuthenticationFailed	Indicates that an authentication attempt failed.
0x0A	LTP_CauseFlowcontrolViolation	Indicates that the flow control mechanisms were violated for the given MDL so the data was ignored
0x0B	LTP_CauseInitTimeout	Indicates that the initialization of the MDC or the Bluetooth protocol stack could not be completed within a given time limit (e.g. due to HCI controller communication timeout). This cause indicates that the MDC is not operational.
0x0C	LTP_CauseInitOutOfSync	Indicates that the Bluetooth protocol stack detected a communication failure with the HCI controller. This cause indicates that the MDC is not operational.
0x0D	LTP_CauseInitHardwarefailure	Indicates that the MDC or the Bluetooth protocol stack detected a hardware failure during startup. This cause indicates that the MDC is not operational.
0xFD	LTP_CauseUnspecified	Indicates that other causes as listed here apply.
0xFE	LTP_CauseNotSupported	Indicates that the operation requested is defined for LTP but not supported by MDC implementation
0xFF	Reserved	n.a.

### 3.6.3 SSP Event type

Value	Name	Description
0x00	LTP_SSPEntryStarted	the passkey key entry protocol procedure is started
0x01	LTP_SSPDigitEntered	a digit is entered by the remote user
0x02	LTP_SSPDigitErased	a digit is erased by the remote user
0x03	LTP_SSPCleared	the display is cleared by the remote user
0x04	LTP_SSPentryComplete	the passkey key entry protocol procedure is completed

### 3.6.4 Link key Type

Value	Name	Description
0x00	LTP_LinkKeyTypeCombination	BT2.0 link key
0x01	LTP_LinkKeyTypeLocalUnit	BT2.0 fixed local unit key
0x02	LTP_LinkKeyTypeRemoteUnit	BT2.0 fixed remote unit key
0x03	reserved	n.a.
0x04	LTP_LinkKeyTypeUnauthenticated	SSP generated link key without MITM protection
0x05	LTP_LinkKeyTypeAuthenticated	SSP generated link key with MITM protection
0x06	LTP_LinkKeyTypeChanged	Link key is updated, key type stays the same
0xFF	LTP_LinkKeyTypeDeleted	Link key is no longer valid and deleted

### 3.7 Application Management and Configuration Access

#### 3.7.1 Exit LTP Request → ExitReq

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<status>

This command shall be used by the MDH to request a termination of the LTP protocol handler of the MDC.

Once an ExitReq is sent from the MDH to the MDC, no other LTP messages shall be sent by the MDH until the ExitRsp message is received.

Once an ExitReq is received by the MDC, all pending or received application data APDUs or APDU segments will be dumped.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x8C → ExitReq
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
status	1	defining status of LTP handler on MDC after termination <u>Possible Values are:</u> <u>0x00:</u> final termination, all MDLs will be closed <u>0x01:</u> temporary termination, all MDLs will stay open

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

### 3.7.2 Exit LTP Response → ExitRsp

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<cause>

This command will be used by the MDC to response to a ExitReq of the MDH.

Once an ExitReq is sent from the MDH to the MDC, no other LTP messages shall be sent by the MDH until this ExitRsp message is received.

If this ExitRsp message indicates success, no new LTP transactions shall be started by the MDH, the only messages that are allowed in this state for the MDH are responses / confirmations for transactions started by the MDC.

If established or pending MDLs have to be disconnected or terminated as a result of this exit transaction, the MDC will automatically initiate these operations and use the standard LTP mechanisms to acknowledge these operations with the MDH.

The final message that is used by the MDC to inform about the LTP termination is the ExitInfo message.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x0C → ExitRsp
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
cause	1	indicates the result of the exit transaction. For a description of defined causes please see the table "Result causes" in this chapter.

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

**Result causes:**

Name	Description
LTP_CauseSuccess	Indicates that an requested operation is completed successfully

### 3.7.3 Exit LTP Information → ExitInfo

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<cause>

This command will be used by the MDC to indicate the termination of the LTP protocol handler.

This message can be either a final result of a termination requested by the MDH with a ExitReq message, or caused by a detected internal error condition of the MDC implementation.

In the case that the MDH receives this ExitInfo message, all MDLs have to be expected being closed or terminated and no further LTP messages shall be sent or expected by the MDH.

Depending on the cause for this ExitInfo message, the MDH might choose to use an other protocol (like “Configurator” commands) to communicate with the MDC, or hardware reset the MDC to bring it back to a defined state.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x0D → ExitInfo
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
cause	1	indicates the result of the exit transaction. For a description of defined causes please see the table “Result causes” in this chapter.

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

**Result causes:**

Name	Description
LTP_CauseSuccess	Indicates that an requested operation is completed successfully

### 3.7.4 Reset Request → ResetReq

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]

This command shall be used by the MDH to reset the MDC. The MDC will respond with a ResetRsp.

Once a ResetReq is send from the MDH to the MDC, no other LTP messages shall be send by the MDH until and after the ResetRsp message is received.

Once ExitReq is received by the MDC, all pending or received application data APDUs or APDU segments will be dumped.

This message can be send by the MDH at anytime and will be detected regardless of internal state of the MDC when send with a precede and consecutive idle time of at least one second. Idle time means that no other data is send by the MDH for this interval. This “off sync” functionality can be used to re-synchronize MDH and MDC.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x93 → ResetReq
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

### 3.7.5 Reset Response → ResetRsp

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<cause>

This command will be used by the MDC to response to a ResetReq of the MDH.

If the ExitRsp send by the MDC indicates success, all pending transactions will be dumped without notification to the MDH and any context will be lost. The MDC will perform a complete hardware reset, so the next valid MDC message to be expected is the ActInfo message.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x13 → ResetRsp
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
cause	1	indicates the result of the transaction. For a description of defined causes please see the table “Result causes” in this chapter.

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

**Result causes:**

Name	Description
LTP_CauseSuccess	Indicates that an requested operation is completed successfully

### 3.7.6 Configuration Tunnel Request → ConfigTunnelReq

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<configCmd>

This command shall be used by the MDH to access the configuration database of the MDC by initiation of configurator commands. For details please contact Stollmann.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0xA2 → ConfigTunnelReq
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
configCmd	n.a.	ASCII coded configuration command

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

### 3.7.7 Configuration Tunnel Information → ConfigTunnelInfo

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<configInfo>

With this command the MDC will forward responses to a configurator command initiated by a ConfigTunnelReq of the MDH.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x23 → ConfigTunnelInfo
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
configInfo	n.a.	ASCII coded configuration information

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

### 3.7.8 Configuration Tunnel Response → ConfigTunnelRsp

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<cause>

With this command the MDC will indicate that a configurator command initiated by a ConfigTunnelReq of the MDH is completed.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x22 → ConfigTunnelRsp
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
cause	1	indicates the result of the transaction. For a description of defined causes please see the table “Result causes” in this chapter.

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

**Result causes:**

Name	Description
LTP_CauseSuccess	Indicates that an requested operation is completed successfully

## 3.8 System Activation, Configuration and Status

### 3.8.1 LTP Active information → ActInfo

**Syntax:** <cmd><copmsk><lp>[max\_Rx\_LTP\_Size][Header\_CRC8]<cause><LTP version><loc\_BD><FW version>

This command will be used by the MDC to indicate the activation of the LTP protocol handler.

This message can be either a result of a startup of the MDC after power on sequence, a performed hardware reset of the MDC, or a protocol switch from an other protocol to LTP initiated by the MDH.

This message can be send by the MDC at any time and shall be detected regardless of the internal state of the MDH when send with a precede idle time of at least one second. Idle time means that no other data is send by the MDC for this interval. This “off sync” functionality can be used to re-synchronize MDH and MDC.

#### Mandatory fields:

Name	Size	Description
cmd opcode	1	0x0E → ActInfo
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
cause	1	indicates the result of the activation For a description of defined causes please see the table “Result causes” in this chapter.
LTP version	1	version of LTP supported by the MDC. The coding of the LTP Version is octet based. The 4 MSBs are the LTP main version, the 4 LSBs are the LTP subversion (e.g. LTP V1.3 is coded as 0x13)
loc_BD	6	Bluetooth device address of local device
FW version	n.a.	UTF-8 formatted, zero terminated human readable version string of this Firmware.

#### Optional fields:

Name	msk	Description
max_Rx_LTP_Size	0x03	The maximum LTP message size the MDC can receive. If an MDH send a LTP message larger then the size indicated here, the MDC will assume the communication is out of sync and send a internalEvent info with cause LTP_CauseInvalidParameter and event type LTP_CommunicationOutOfSync. <b>(Default=117)</b>
max_Tx_LTP_Size	0x0C	The maximum LTP message size the MDC will generate. The MDH can use this information to optimize memory usage. <b>(Default=131)</b>
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

**Result causes:**

Name	Description
LTP_CauseSuccess	Indicates that an requested operation is completed successfully
LTP_CauseInitTimeout	Indicates that the initialization of the MDC or the Bluetooth protocol stack could not be completed within a given time limit (e.g. due to HCI controller communication timeout). This cause indicates that the MDC is not operational.
LTP_CauseInitOutOfSync	Indicates that the Bluetooth protocol stack detected a communication failure with the HCI controller. This cause indicates that the MDC is not operational.
LTP_CauseInitHardwarefailure	Indicates that the MDC or the Bluetooth protocol stack detected a hardware failure during startup. This cause indicates that the MDC is not operational.

**3.8.2 Radio Mode Set Request → RadioModeSetReq**

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<radioMode>

This command shall be used by the MDH to control the functionality of the local device on Bluetooth radio level.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0xA4 → RadioModeSetReq
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
radioMode	1	Mode to set the radio to. For possible Values see Radio Mode Table in this chapter

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

**Radio Modes:**

Value	Name	Description
0x01	LTP_RadioVisibleConnectable	The device is visible for inquiring devices and scans for incoming connections (inquiry scan and page scan enabled on local device)
0x02	LTP_RadioVisible	The device is visible for inquiring devices (inquiry scan enabled, no page scan)
0x03	LTP_RadioConnectable	The device scans for incoming connections (page scan enabled, no inquiry scan)
0x04	LTP_RadioNonDiscoverable	The device is not visible for inquiring devices and does not scan for incoming connections (no inquiry scan and page scan disabled) but is able to initiate connections
0x05	LTP_RadioDeepSleep	The Bluetooth Radio is in a non-operational mode but retains its local configuration and can be restored to normal operation without reconfiguration. The implementation of this mode is platform specific and is not supported for all Platforms.
0x06	LTP_RadioOff	The Bluetooth Radio is switched off and has to be reconfigured for normal operation. The implementation of this mode is platform specific and is not supported for all platforms. If this mode is supported, the MDC recognizes the nonfunctional Bluetooth Radio as "out of sync" and generates a corresponding "ActInfo" message.

**3.8.3 Radio Mode Set Response → RadioModeSetRsp**

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<cause>

This command will be used by the MDC to respond to a RadioModeSetReq initiated from the MDH.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x24 → RadioModeSetRsp
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
cause	1	indicates the result of the transaction. For a description of defined causes please see the table "Result causes" in this chapter.

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

**Result causes:**

Name	Description
LTP_CauseSuccess	Indicates that an requested operation is completed successfully
LTP_CauseReject	The requested operation could not be completed due to a conflict with the MDC status (e.g. deep sleep request during an active connection)
LTP_causeNotSupported	The requested mode is not supported.

### 3.8.4 Internal Event Info → InternalEventInfo

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<cause><eventType><eventInfo>

This command will be used by the MDC to inform the MDH about an illegal message sent by the MDH or regarding illegal events on the air interface.

This message will only be initiated if the MDC has no other means to signal that the message or event was illegal. An example would be a confirmation message from the MDH with an invalid parameter. Since this protocol does not define a reaction for a confirmation message from the MDH, the InternalEventInfo message will be used by the MDC to notify the MDH that an invalid confirmation message was received and will be ignored.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x1D → InternalEventInfo
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
cause	1	indicates the result of the transaction. For a description of defined causes please see the table “Result causes” in this chapter.
eventType	1	Indicates the reason for this indication For possible values see table Event Types in this chapter
eventInfo	4	Additional information for internal reproduction and tracing purpose

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

**Event Types:**

Value	Name	Description
0x01	LTP_InvalidCreateMDLConfReceived	The MDC received a CreateMDLConf message from the MDH that could not be handled and that will be ignored
0x02	LTP_InvalidReconnectMDLConfReceived	The MDC received a ReconnectMDLConf message from the MDH that could not be handled and that will be ignored
0x03	LTP_InvalidDisconnectMDLConfReceived	The MDC received a DisconnectConf message from the MDH that could not be handled and that will be ignored
0x04	LTP_InvalidDIDDeviceConfReceived	The MDC received a DIDDeviceConf message from the MDH that could not be handled and that will be ignored
0x05	LTP_InvalidDataCreditsReceived	The MDC received a Data message from the MDH that included invalid credit information that will be ignored
0x06	LTP_InvalidHDPSERVICEConfReceived	The MDC received a HDPSERVICEConf message from the MDH that could not be handled and that will be ignored
0x07	LTP_InvalidHDPEndpointConfReceived	The MDC received a HDPEndpointConf message from the MDH that could not be handled and that will be ignored
0x08	LTP_InvalidSPPEndpointConfReceived	The MDC received a SPPEndpointConf message from the MDH that could not be handled and that will be ignored
0x09	LTP_InvalidSecurityConfReceived	The MDC received a security related confirmation message from the MDH that could not be handled and that will be ignored
0x0A	LTP_InvalidRemoteEventReceived	The MDC received an invalid event or discovered an invalid behavior from / of the remote device that can affect the functionality of the connection. The reaction of the MDC on the air interface will be according to the respective specifications ([X1], [X2], [X3], [X4],[X5])
0x0B	LTP_CommunicationTimeout	The MDC implements several timeouts that prevent the system to be driven into a deadlock situation by a non functional MDH or remote device. This timeout should never occur during normal operation
0x40	LTP_CommunicationOutOfSync	THE MDC received a message with invalid Header_CRC8
0x41	LTP_MalformedMsgReceived	The MDC received a message that was detected as malformed (e.g. misses a mandatory parameter)
0x42	LTP_InvalidDataReceived	The MDC received a data message from the MDH that could not be handled and that will be ignored

---

**Result causes:**

<b>Name</b>	<b>Description</b>
LTP_CauseInvalidParameter	Indicates that at least one parameter of a message is invalid
LTP_CauseResourceError	Indicated that an internal operation can't be performed due to resource constrains
LTP_InvalidState	indicates that a received message was not expected, so the message is ignored.
LTP_CauseFlowcontrolViolation	Indicates that flow control mechanisms where violated for the given MDL so the data was ignored
LTP_CauseUnspecified	other causes

### 3.9 Connection Management

#### 3.9.1 Channel create request → ConnectMDLReq

**Syntax:** <cmd><copmsk><lp>[[linkConfigType]][loc\_MDEP\_ID]  
[Header\_CRC8]<rem\_BD><rem\_MDEP\_ID><rem\_C\_PSM><rem\_D\_PSM>

This command shall be used by the MDH to request creation of a new MDL connection to a specific MDEP\_ID of a remote device by the MDC.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x85 → ConnectMDLReq
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
rem_BD	6	Bluetooth device address of remote device to connect to.
rem_MDEP_ID	1	MDEP_ID of remote device to connect an MDL to. Depending on the protocol (MCAP/RFCOMM) that shall be used for this MDL connection, either the MDEP-ID (MCAP) or the server channel (RFCOMM) of the remote device shall be referenced here. If this parameter is set to zero, the MCAP Echo Channel on the remote device will be opened. The MCAP echo channel is defined to be only used for testing purposes. For details be referred to X2.
rem_C_PSM	2	control channel PSM of remote device. If this parameter is set to 3, a SPP/RFCOMM Channel will be opened, otherwise a HDP/MCAP control Channel will be opened to the PSM defined by this parameter. For information how to gather this parameter please refer to chapter [3.15 HDP service discovery management and 3.16 SPP service discovery management].
rem_D_PSM	2	data channel PSM of remote device. If a SPP/RFCOMM channel is intended, this Parameter shall be set to zero. If a HDP/MCAP connection is intended and this Parameter is set to zero, only the MCAP control channel will be opened, otherwise a MCAP data channel will be opened to the PSM identified by this parameter For information how to gather this parameter please refer to chapter [3.15 HDP service discovery management and 3.16 SPP service discovery management].

---

**Optional fields:**

Name	msk	Description
LinkConfigType	0x01	requested QoS-Configuration for this link . see chapter 3.6.1 QoS parameter for MDL negotiation for defined settings <b>(Default=LTP_LinkConfigDontCare)</b>
loc_MDEP_ID	0x02	local MDEP_ID as reference for this MDL <b>(Default=0x01)</b>
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

### 3.9.2 Channel create response → ConnectMDLRsp

**Syntax:** <cmd><copmsk><lp>[loc\_MDL\_ID][loc\_MDEP\_ID]  
[Header\_CRC8]<cause><rem\_BD><rem\_MDEP\_ID>

This command will be used by the MDC to response to a connectMDLReq initiated from the MDH.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x05 → ConnectMDLRsp
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
cause	1	indicates the result of the connectMDL transaction. For a description of defined causes please see the table "Result causes" in this chapter.
rem_BD	6	Bluetooth device address of remote device to connect to (same as in the corresponding Request)
rem_MDEP_ID	1	MDEP_ID of remote device to connect a MDL to (same as in the corresponding Request)

**Optional fields:**

Name	msk	Description
Loc_MDL_ID	0x01	local mediated data link ID for this LTP packet ( <b>Default=0x01</b> )
loc_MDEP_ID	0x02	local MDEP_ID as reference for this MDL ( <b>Default=0x01</b> )
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

**Result causes:**

Name	Description
LTP_CauseSuccess	Requested operation is completed successfully
LTP_CauseReject	The remote device was in range and accepted a connection but rejected the requested operation (e.g. Invalid MDEP_ID used for connection attempt)
LTP_CauseResourceError	The requested operation can't be completed due to lack of resources.
LTP_CauseInvalidParameter	Indicates that at least one parameter of a request is invalid, so the request is rejected
LTP_CauseConnectionDisconnect	The remote device was in range but rejected a connection attempt (e.g. invalid PSM used for connection attempt)
LTP_CauseConnectionLost	The remote device did not respond to a connection request or was out of range.
LTP_CauseAuthenticationFailed	The connection could not established due to security procedures failed

### 3.9.3 Channel reconnect request → ReconnectMDLReq

**Syntax:** <cmd><copmsk><lp>[rem\_C\_PSM][rem\_D\_PSM]  
[Header\_CRC8]<loc\_MDL\_ID>

This command shall be used by the MDH to request a HDP reconnection of a disconnected but formerly created MDL by the MDC.

Only the Device that initially created the MDL (by use of the ConnectMDLReq command) can request a reconnect of that MDL.

If the transaction succeeds, the complete configuration and setup that was used for the creation of that MDL to be reconnected will be reestablished.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x8A → ReconnectMDLReq
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
Loc_MDL_ID	1	local mediated data link ID to be reconnected

**Optional fields:**

Name	msk	Description
rem_C_PSM	0x03	control channel PSM of remote device. If this Parameter is set to 0x0000, the same rem_C_PSM value will be used as it was negotiated during the creation of the loc_MDL_ID. <b>(Default=0x0000)</b> <b>Please be aware that this default behavior is not HDP compliant since the PSMs are defined to be dynamic and can change any time. To gather actual PSM values a new HDPDiscoveryReq shall be initiated.</b>
rem_D_PSM	0x0C	data channel PSM of remote device. If this parameter is set to 0x0000, the same rem_C_PSM value will be used as it was negotiated during the creation of the loc_MDL <b>(Default=0x0000)</b> <b>Please be aware that this default behavior is not HDP compliant since the PSMs are defined to be dynamic and can change any time. To gather actual PSM values a new HDPDiscoveryReq shall be initiated.</b>
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

### 3.9.4 Channel reconnect response → ReconnectMDLRsp

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<cause><loc\_MDL\_ID>

This command will be used by the MDC to respond to a ReconnectMDLReq of the MDH.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x0A → ReconnectMDLRsp
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
cause	1	indicates the result of the transaction. For a description of defined causes please see the table “Result causes” in this chapter.
Loc_MDL_ID	1	local mediated data link ID to be reconnected

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

**Result causes:**

Name	Description
LTP_CauseSuccess	Indicates that an requested operation is completed successfully
LTP_CauseReject	Indicates that re remote device rejected the reconnect
LTP_CauseConnectionDisconnect	Indicates that the remote device did not accept the connection
LTP_CauseConnectionLost	Indicates that the remote device did not respond to the connection attempt (e.g. device is out of range)
LTP_CauseResourceError	The requested operation can't be completed due due to lack of resources.
LTP_CauseInvalidState	The initiator of the ReconnectReq was not the initial creator of the MDL so the reconnect operation can not performed
LTP_CauseAuthenticationFailed	The remote device was in range but authentication failed
LTP_CauseNotSupported	this feature is not supported (e.g. not implemented)

### 3.9.5 Channel reconnect indication → ReconnectMDLInd

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<loc\_MDL\_ID>

This command will be used by the MDC to indicate a pending reconnection to a specific local mediated data link ID to the MDH.

This message is triggered by a pending reconnection that is either initiated from the local or from the remote side. The message is triggered on both ends of the communication link, in order to keep the FSM on app layer structural simple.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x8B → ReconnectMDLInd
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
Loc_MDL_ID	1	local mediated data link ID to be reconnected

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

### 3.9.6 Channel reconnect confirmation → ReconnectMDLCnf

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<cause><loc\_MDL\_ID>

This command shall be used by the MDH to respond to a ReconnectMDLInd of the MDC.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x0B → ReconnectMDLCnf
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
cause	1	indicates the result of the transaction. For a description of defined causes please see the table “Result causes” in this chapter.
Loc_MDL_ID	1	local mediated data link ID to be reconnected

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

**Result causes:**

Name	Description
LTP_CauseAccept	Indicates that the MDL_ID is known to the MDH and the MDL establishment is accepted
LTP_CauseReject	Indicates the MDL_ID is known by the MDH but the MDL establishment is rejected
LTP_CauseInvalidParameter	Indicates that MDL_ID is not known to the MDH

### 3.9.7 channel disconnect request → DisconnectMDLReq

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<cause><loc\_MDL\_ID>

This command shall be used by the MDH to request a disconnection of an established MDL or the release of a formally created but then temporarily or unintentional disconnected MDL\_ID by the MDC.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x88 → DisconnectMDLReq
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
cause	1	indicates the cause for the disconnect. For a description of defined causes please see the table “Result causes” in this chapter.
Loc_MDL_ID	1	local mediated data link ID to be disconnected or terminated

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

**Result causes:**

Name	Description
LTP_CauseConnectionDisconnect	Indicates that a MDL shall be disconnected permanently and released. In case that an idle MDL is addressed, that MDL shall be released by the MDC. This is the only valid cause to terminate a MCAP echo channel connection. For MCAP echo channel details be referred to X2.
LTP_CauseConnectionPaused	Indicates that a MDL shall be disconnected temporary but might be reconnected later. This cause shall only used for a HDP based MDL

### 3.9.8 Channel disconnect response → DisconnectMDLRsp

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<cause><loc\_MDL\_ID>

This command will be used by the MDC to respond to a DisconnectMDLReq of the MDH.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x08 → DisconnectMDLRsp
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
cause	1	indicates the result of the DisconnectMDL transaction. For a description of defined causes please see the table "Result causes" in this chapter.
Loc_MDL_ID	1	local mediated data link ID to be disconnected or terminated

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

**Result causes:**

Name	Description
LTP_CauseSuccess	Indicates that an requested operation is completed successfully
LTP_CauseInvalidParameter	Indicates that at least one parameter of the corresponding message was invalid, so the data is ignored

### 3.9.9 channel disconnect indication → DisconnectMDLInd

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<cause><loc\_MDL\_ID>

This command will be used by the MDC to indicate a disconnection or termination (i.e. permanent disconnection) of an established MDL to the MDH.

This message is triggered by a disconnection or termination that is either initiated from the local, or from the remote side.

In case a MCAP echo channel MDL is disconnected, this message will only be signaled if the local MDC was the initiator of that MCAP echo channel MDL. For MCAP echo channel details be referred to X2.

The MDH shall respond with a DisconnectMDLCnf to this DisconnectMDLInd message.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x89 → DisconnectMDLInd
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
cause	1	indicates the cause for the disconnect. For a description of defined causes please see the table "Result causes" in this chapter.
Loc_MDL_ID	1	local mediated data link ID to be disconnected or terminated

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

**Result causes:**

Name	Description
LTP_CauseConnectionDisconnect	Indicates that a MDL is disconnected permanently, that is "terminated". This is the only valid cause to terminate a MCAP echo channel connection. For MCAP echo channel details be referred to X2
LTP_CauseConnectionPaused	Indicates that a MDL is disconnected temporary but might be reconnected later
LTP_CauseConnectionLost	Indicates that a connection is disconnected unintentionally (e.g. device moved out of range) but might be reconnected later

---

### 3.9.10 Channel disconnect confirmation → DisconnectMDLCnf

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<loc\_MDL\_ID>

This command will be used by the MDH to confirm a DisconnectMDLInd of the MDC.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x09 → DisconnectMDLCnf
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
Loc_MDL_ID	1	local mediated data link ID to be disconnected or terminated

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

### 3.10 MDL / MCL / ACL Management

#### 3.10.1 new channel create indication → CreateMDLInd

**Syntax:** <cmd><copmsk><lp>[LinkConfigType][loc\_MDEP\_ID][rem\_MDEP\_ID]  
[Header\_CRC8]<rem\_BD><loc\_MDL\_ID>

This command will be used by the MDC to indicate a creation of a new MDL to the MDH.

This message is triggered by a connection that is either initiated from the local, or from the remote side.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x86 → CreateMDLInd
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
rem_BD	6	Bluetooth device address of remote device
Loc_MDL_ID	1	local mediated data link ID to be created. For a new MDL the MDC automatically assigns a system unique local_MDL_ID in the range of 0x01 to 0xFE. Due to this, the local_MDL_ID is sufficient to address a specific MDL even if multiple MDLs and/or MCL are established in parallel. This differs from the MDL ID concept of MCAP [X2] where an MDL ID is only unique within its MCL context.

**Optional fields:**

Name	msk	Description
LinkConfigType	0x01	requested QoS-Configuration for this link. If this message is initiated due to a local action (e.g. ConnectMDLReq) this parameter reflects the corresponding parameter of that action, otherwise it indicates the configuration that is requested by the remote device. See chapter 3.6.1 QoS parameter for MDL negotiation for defined settings.
loc_MDEP_ID	0x02	MDEP_ID of local device to connect a MDL to. If this message is initiated due to a local action (e.g. ConnectMDLReq) this parameter reflects the corresponding parameter of that action, otherwise it indicates the local MDEP the remote device requests a connection to. <b>(Default=0x01)</b>
rem_MDEP_ID	0x04	MDEP_ID of remote device to connect a MDL to. If this message is initiated due to a local action (e.g. ConnectMDLReq) this parameter reflects the corresponding parameter of that action, otherwise it is set to zero.
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

### 3.10.2 new channel create confirmation → CreateMDLCnf

**Syntax:** <cmd><copmsk><lp>[LinkConfigType][maxTPDUusCredits]  
[Header\_CRC8]<cause><loc\_MDL\_ID>

This command will be used by the MDH to response to a to CreateMDLInd from the MDC.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x06 → CreateMDLCnf
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
cause	1	indicates the result of the createMDL transaction. For a description of defined causes please see the table “Result causes” in this chapter.
Loc_MDL_ID	1	local mediated data link ID to be created

**Optional fields:**

Name	msk	Description
LinkConfigType	0x01	requested QoS-configuration for this link. See chapter 3.6.1 QoS parameter for MDL negotiation for defined settings <b>(default=LTP_LinkConfigReliable)</b>
maxTPDUusCredits	0x02	Maximum number of pending usCredits from the MDH that are allowed for this MDL. If this parameter is set to Zero, no credit based flow control is supported by the MDH <b>(default 0x00)</b> . <b>Please be aware that credit based flow control should be supported by a MDH due to the fact that the UART hardware flowcontrol might interfere with the LTP protocol otherwise.</b>
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

**Result causes:**

Name	Description
LTP_CauseAccept	The requested operation is accepted by the MDH
LTP_CauseReject	The requested operation is rejected by the MDH

### 3.10.3 Channel connect information → ConnectMDLInfo

**Syntax:**

```
<cmd><copmsk><lp>[LinkConfigType][maxTPDUusCredits][maxTPDUdsCredits]
[Header_CRC8]<loc_MDL_ID>< max_LTP_size><max_APDU_size>
```

The purpose is to inform the MDH of the parameters of the new MDL connection.

After this command is received by the MDH, the MDL is ready to exchange application data APDUs.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x04 → ConnectMDLInfo
copmsk	1	0x00 if no optional parameters applied otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
loc_MDL_ID	1	local mediated data link ID for this MDL
max_LTP_size	2	the maximum LTP message size the MDH is allowed to send on this MDL. If an LTP message to transport an application data packet would be larger than the maximum packet size indicated here, the application data packet has to be segmented and transported via multiple LTP packets instead
max_APDU_size	2	maximum application data packet size for this MDL

**Optional fields:**

Name	msk	Description
LinkConfigType	0x01	requested QoS-Configuration for this MDL see chapter 3.6.1 QoS parameter for MDL negotiation for defined settings <b>(Default=LTP_LinkConfigTypeReliable)</b>
maxTPDUusCredits	0x02	Maximum number of pending upstream usCredits that will be used by the MDC for this MDL. This value will only be indicated if the MDC supports MDL channel based flow control and the MDH granted maxTPDUusCredits in the CreateMDLCnf for this MDL. The indicated value is equal or less that the maxTPDUusCredits granted in the CreateMDLCnf for this MDL by the MDH. <b>(Default=0x00)</b>
maxTPDUdsCredits	0x04	This value will only be indicated if the MDH granted maxTPDUusCredits in the CreateMDLCnf for this MDL by the MDH Maximum number of pending downstream dsCredits that can be used by the MDH for this MDL. <b>(Default=0x00)</b>
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

---

### 3.10.4 channel delete information → DeleteMDLInfo

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<loc\_MDL\_ID>

This command will be used by the MDC to inform the MDH that a local mediated data link ID is no longer valid.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x07 → DeleteMDLInfo
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
Loc_MDL_ID	1	local mediated data link ID deleted

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

### 3.10.5 MCL Status Info → MCLStatusInfo

**Syntax:**

<cmd><copmsk><lp>[Header\_CRC8]<rem\_BD><local\_MCL\_ID><local\_MCL\_Status>

The purpose of this message is to inform the MDH about the status of a given MCL connection. Depending on the given Platform, only a subset of status changes will be indicated.

The event indicates always the latest transition. This fact is to be considered for multiple data channels on a single MCL.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x25 → MCLStatusInfo
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
rem_BD	6	Bluetooth device address of remote device
local_MCL_ID	1	Identifier to referenced MCL
local_MCL_Status	1	status of referenced MCL, for possible settings see MCL Status Table in this chapter

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

**MCL Status:**

Value	Name	Description
0x01	idle	MCL is allocated
0x02	controlConnecting	MCL has a control channel setup pending
0x03	controlConnected	MCL has a connected control channel but no data channels
0x04	controlDisconnecting	MCL has a pending control channel disconnect
0x05	controlListening	MCL waits for incoming control channel
0x06	dataConnecting	MCL has a data channel setup pending
0x07	dataConnected	MCL has at least one data channel connected
0x08	dataDisconnecting	MCL has a data channel disconnect pending
0x09	dataListening	MCL waits for incoming data channel
0x0A	controlWaitForResponse	MCL waits for MCAP response
0x0B	comWaitForResponse	MCL waits for application response
0x0C	released	MCL is not allocated

### 3.10.6 ACL Status Info → ACLStatusInfo

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<rem\_BD><local\_ACL\_Status>

The purpose of this message is to inform the MDH about the status of a given ACL connection. Depending on the given Platform, only a subset of status changes will be indicated.

The event indicates always the latest transition. This fact is to be considered for multiple MCL and/or MDL connections on a single ACL connection.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x0F → ACLStatusInfo
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
rem_BD	6	Bluetooth device address of remote device
local_ACL_Status	1	status of referenced ACL connection, for possible settings see ACL Status Table in this chapter

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

**ACL Status:**

Value	Name	Description
0x01	connectedActive	ACL connection is established and in active (not in sniff) mode
0x02	connectedSniff	ACL connection is in sniff mode
0x03	authenticationStarted	Authentication for this ACL connection has started
0x04	authenticationSuccess	ACL connection is successfully authenticated
0x05	AuthenticationFailure	Authentication for this ACL connection failed
0x06	connectionEncrypted	ACL connection data traffic is encrypted
0x07	connectionDisconnected	ACL connection is no longer connected
0x08	connectionNotEncrypted	ACL connection encryption is disabled

### 3.11 Security Management

#### 3.11.1 Pairable Mode set Request → PairableModeSetReq

**Syntax:**

```
<cmd><copmsk><lp>[Header_CRC8]<enablePairableMode><BluetoothMode><AuthRequirements><IOCapabilities><remoteOOBDataPresent>
```

This command shall be used by the MDH to set the pairable mode of the local device.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0xA6 → PairableModeSetReq
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
enablePairableMode	1	If this parameter is set to TRUE, pairable mode is enabled, otherwise pairable mode is disabled
BluetoothMode	1	This parameter defines the Bluetooth mode the device supports while the pairable Mode is enabled. For possible settings see BluetoothMode table below.
AuthRequirements	1	This parameter defines the authentication requirements for authentication while the local device is in pairable mode. For possible settings see table AuthRequirements below.
IOCapabilities	1	defines the input/output capabilities that can be used for authentication. For possible settings see table IOCapabilities below.
remoteOOBDataPresent	1	If this parameter is set to TRUE, remote OOB data is present and can be used for authentication, otherwise OOB functionality is disabled

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

BluetoothMode:

Value	Name	Description
0x00	Bluetooth2_1SupportDisabled	disable all Bluetooth 2.1 functionality <b>Note: this mode is only for Testing, it shall not be used for Product level configuration</b>
0x01	Bluetooth2_1SupportEnabled	enable Bluetooth 2.1 functionality
0x02	Bluetooth2_1DebugSupportEnabled	enable Bluetooth 2.1 functionality with SSP debug support (Note: this mode shall be used only for debugging purpose since it disables over-the-air data encryption)

IOCapabilities:

Value	Name	Description
0x00	displayOnly	only a Display present, no Keyboard or Yes/No Keys
0x01	displayYesNo	Display and Yes/No Keys present
0x02	keyboardOnly	only a Keyboard present, no Display
0x03	noIOCapabilities	no input/output capabilities

AuthRequirements:

Value	Name	Description
0x00	noMITMRequiredNoStore	MITM protection not required, no bonding
0x01	MITMRequiredNoStore	MITM protection required, use IO capabilities, no bonding
0x02	noMITMRequiredDedicatedBonding	This setting is deprecated and should not be used. <b>Note: if used, it will be handled equal to noMITMRequiredGeneralBonding</b>
0x03	MITMRequiredDedicatedBonding	This setting is deprecated and should not be used. <b>Note: if used, it will be handled equal to MITMRequiredGeneralBonding</b>
0x04	noMITMRequiredGeneralBonding	MITM protection not required, perform general Bonding
0x05	MITMRequiredGeneralBonding	MITM protection not required, use IO capabilities, perform general Bonding

### 3.11.2 Pairable Mode set Response → PairableModeSetRsp

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<cause>

This command will be used by the MDC to respond to a PairableModeSetReq of the MDH.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x26 → PairableModeSetRsp
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
cause	1	indicates the result. For a description of defined causes please see the table "Result causes" in this chapter.

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

**Result causes:**

Name	Description
LTP_btmedCauseSuccess	Operation completed successfully
LTP_btmedCauseInvalidState	Operation can not be performed due to invalid radio state
LTP_btmedCauseInvalidParameter	Indicates that the authentication code was detected as invalid by the remote device

### 3.11.3 Authentication Request → AuthReq

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<rem\_BD>

This command is used by the MDH to initiate a dedicated authentication with a given remote device.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x9A → AuthReq
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
rem_BD	6	Bluetooth device address of remote device to authenticate

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

### 3.11.4 Authentication Response → AuthRsp

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<cause><rem\_BD>

This command will be used by the MDC to respond to a AuthReq of the MDH.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x1A → AuthRsp
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
cause	1	indicates the result of the requested action. For a description of defined causes please see the table “Result causes” in this chapter.
Rem_BD	6	Bluetooth device address of remote device to authenticate to (same as in the corresponding request)

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

**Result causes:**

Name	Description
LTP_CauseSuccess	Operation completed successfully
LTP_CauseReject	Indicates that the remote or local device rejects the authentication attempt
LTP_CauseInvalidParameter	Indicates that the authentication code was detected as invalid by the remote device
LTP_CauseConnectionLost	Indicates that the connection attempt failed (e.g. device was out of range)

**3.11.5 Authentication Request Indication → AuthRequestInd**

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<rem\_BD>

This command is used by MDC to indicate the requirement for a non SSP user level authentication via PIN entry (legacy pairing).

This authentication might be either triggered by the local MDC or the remote device.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x90 → AuthRequestInd
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
rem_BD	6	Bluetooth device address of remote device to authenticate to

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

### 3.11.6 Authentication Request Confirmation → AuthRequestCnf

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<cause><rem\_BD><authCode>

This command is used by the MDH to respond to an AuthRequestInd of the MDC. The response includes all information needed to perform a user level authentication. This might be a PIN code entered by a user or a result of other out-of-band operations.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x10 → AuthRequestCnf
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
cause	1	indicates the result of the authentication. For a description of defined causes please see the table “Result causes” in this chapter.
Rem_BD	6	Bluetooth device address of remote device to authenticate to
authCode	N/A	Binary authentication code provided by user interaction

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

**Result causes:**

Name	Description
LTP_CauseAccept	Indicates that the MDH accepts the authentication attempt and the AuthCode in this message can be used for authentication purposes
LTP_CauseReject	Indicates that the MDH rejects the authentication attempt

If an authentication is rejected or fails for other reasons, the connection will be disconnected.

### 3.11.7 User Confirmation Request Indication → UserConfRequestInd

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8] <rem\_BD><displayValue>

If a SSP procedure is started that requires “Display Yes/No” functionality the MDC will send this message to the MDH to request a value to be displayed as a 6 digit decimal value to the User and wait for a User interaction.

For further information about SSP and its procedures refer to [X16].

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x9E → UserConfRequestInd
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
rem_BD	6	Bluetooth device address of remote device to authenticate to
displayValue	4	value to be displayed as a six digit decimal number

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

### 3.11.8 User Confirmation Request Confirmation → UserConfRequestCnf

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<cause><rem\_BD>

With this message the MDH shall respond to a SSP driven UserConfirmationReqInd as soon as User interaction is performed.

For further information about SSP and its procedures refer to [X16].

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x1E → UserConfRequestCnf
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
cause	1	indicates the result of the authentication. For a description of defined causes please see the table “Result causes” in this chapter.
Rem_BD	6	Bluetooth device address of remote device to authenticate to

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

**Result causes:**

Name	Description
LTP_CauseAccept	Indicates that the User accepts the authentication
LTP_CauseReject	Indicates that the User rejects the authentication

### 3.11.9 Passkey Request Indication → PasskeyRequestInd

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<rem\_BD>

If a SSP procedure is started that requires Keyboard functionality from the remote peer, and display functionality from the local device the MDC will send this message to the MDH to request User interaction.

For further information about SSP and its procedures refer to [X16].

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x9F → PasskeyRequestInd
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
rem_BD	6	Bluetooth device address of remote device to authenticate to

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

### 3.11.10 Passkey Request Confirmation → PasskeyRequestCnf

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<cause><rem\_BD>

With this message the MDH shall respond to a SSP driven UserPasskeyReqInd as soon as User interaction is performed.

For further information about SSP and its procedures be referred to [X16].

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x1F → PasskeyRequestCnf
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
cause	1	indicates the result of the authentication. For a description of defined causes please see the table “Result causes” in this chapter.
Rem_BD	6	Bluetooth device address of remote device to authenticate to

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

**Result causes:**

Name	Description
LTP_CauseAccept	Indicates that the User accepts the authentication
LTP_CauseReject	Indicates that the User rejects the authentication

### 3.11.11 Passkey Request reply Request → PasskeyReqReplyReq

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<cause><rem\_BD>< nicode>

If a SSP procedure is completed that requires Keyboard functionality from the local, and display functionality from the remote device the MDH shall send this message to the MDC to transfer the result of the Keyboard input and request completion of the SSP procedures.

For further information about SSP and its procedures be refer [X16].

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0xA7 → PasskeyReqReplyReq
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
cause	1	indicates the result of the authentication. For a description of defined causes please see the table “Result causes” in this chapter.
Rem_BD	6	Bluetooth device address of remote device to authenticate to
passKey	4	result of Keyboard input

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

**Result causes:**

Name	Description
LTP_CauseAccept	Indicates that the User accepts the authentication
LTP_CauseReject	Indicates that the User rejects the authentication

### 3.11.12 Passkey Request reply Response → PasskeyReqReplyRsp

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<cause>

With this message the MDC will respond to a SSP driven PasskeyReqReplyReq message of the MDH.

For further information about SSP and its procedures refer to [X16].

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x27 → PasskeyReplyRsp
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
cause	1	indicates the result of the transaction. For a description of defined causes please see the table “Result causes” in this chapter.

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

**Result causes:**

Name	Description
LTP_CauseSuccess	Indicates that the operation was performed successfully
LTP_CauseInvalidState	Operation can not be performed due to invalid radio state (see TBTMedRadioMode definition)

### 3.11.13 Passkey Notification Information → PasskeyNotificationInfo

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<rem\_BD><displayValue>

If a SSP procedure is completed that requires Keyboard functionality from the remote, and display functionality from the local device the MDC will send this message to notify the MDH about the initial state of the display content.

For further information about SSP and its procedures refer to [X16].

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x28 → PasskeyNotificationInfo
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
rem_BD	6	Bluetooth device address of remote device to authenticate to
displayValue	4	Value to be displayed as a 6 digit decimal number

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

### 3.11.14 Keypress Notification Request → KeypressNotificationReq

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<rem\_BD><eventType>

If a SSP procedure is started that requires Keyboard functionality from the local, and display functionality from the remote device the MDH shall send this message to the MDC to indicate User interaction.

For further information about SSP and its procedures be referred to [X16].

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0xA9 → KeypressNotificationReq
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
rem_BD	6	Bluetooth device address of remote device to authenticate to
eventType	1	event to be communicated. See chapter 3.6.3 SSP Event type for Details

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

**3.11.15 Keypress Notification Response → KeypressNotificationRsp**

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<cause>

With this message the MDC will respond to a SSP driven KeypressNotificationReq message of the MDH.

For further information about SSP and its procedures refer to [X16].

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x29 → KeypressNotificationRsp
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
cause	1	indicates the result of the transaction. For a description of defined causes please see the table “Result causes” in this chapter.

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

**Result causes:**

Name	Description
LTP_CauseSuccess	Indicates that the User accepts the authentication
LTP_CauseInvalidState	Operation can not be performed due to invalid state

**3.11.16 Keypress Notification Information → KeypressNotificationInfo**

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<rem\_BD><eventType>

If a SSP procedure is in progress that requires Keyboard functionality from the remote, and display functionality from the local device the MDC will send this message to notify the MDH about User interaction.

For further information about SSP and its procedures refer to [X16].

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x2A → KeypressNotificationInfo
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
rem_BD	6	Bluetooth device address of remote device to authenticate to
eventType	1	event to be communicated. See chapter 3.6.3 SSP Event type for Details

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

**3.11.17 Remote OOB-Data Request Indication → RemoteOOBReqInd**

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<rem\_BD>

If a SSP procedure is started that requires OOB functionality from the local and remote device the MDC will send this message to the MDH to request OOB data from the remote device via OOB communication.

For further information about SSP and its procedures refer to [X17].

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0xA0 → RemoteOOBRequestInd
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
rem_BD	6	Bluetooth device address of remote device to authenticate to

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

### 3.11.18 Remote OOB-Data Request confirmation → RemoteOOBReqCnf

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<cause><rem\_BD><C><R>

With this message the MDH shall respond to a SSP driven RemoteOOBDataReqInd message of the MDC.

For further information about SSP and its procedures refer to [X16].

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x20 → RemoteOOBReqCnf
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
cause	1	indicates the result of the transaction. For a description of defined causes please see the table "Result causes" in this chapter.
Rem_BD	6	Bluetooth device address of remote device to authenticate to
C	16	Hash C value
R	16	Randomizer R value

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

**Result causes:**

Name	Description
LTP_CauseAccept	Indicates that the User accepts the authentication
LTP_CauseReject	Indicates that the User rejects the authentication

### 3.11.19 Local OOB-Data Request → LocalOOBReq

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]

If a SSP procedure is started that requires OOB functionality from the local and remote device the MDH can send this message to the MDC to request OOB data from the local device to be transferred via OOB communication to the remote device for authentication.

For further information about SSP and its procedures refer to [X16].

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0xAB → LocalOOBReq
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

### 3.11.20 Local OOB-Data Response → LocalOOBRsp

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<cause><C><R>

With this message the MDC will respond to a SSP driven LocalOOBDataReq message of the MDH.

For further information about SSP and its procedures be referred to [X16].

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x2B → LocalOOBRsp
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
cause	1	indicates the result of the transaction. For a description of defined causes please see the table "Result causes" in this chapter.
C	16	Hash C value
R	16	Randomizer R value

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

**Result causes:**

Name	Description
LTP_CauseSuccess	Indicates that the operation was performed successfully
LTP_CauseFeatureNotSupported	Indicates that the OOB data can not be supplied

### 3.11.21 Authentication result Indication → AuthResultInd

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<cause><rem\_BD><LinkKey><KeyType><AppData>

This command is used by the MDC to indicate the resulting information of a performed bonding attempt.

Please note that this AuthResultInd Message will NOT be generated if the MDC is configured with AuthRequirements set to noMITMRequiredNoStore or noMITMRequiredNoStore. For details please refer to [3.11.1 Pairable Mode set Request → PairableModeSetReq]

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0xA1 → AuthResultInd
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
cause	1	indicates the result of the transaction. For a description of defined causes please see the table “Result causes” in this chapter.
Rem_BD	6	Bluetooth device address of remote device authenticated to
LinkKey	16	128 bit link key
KeyType	1	Key Type of link key, See chapter 3.6.4 Link key Type for details.
AppData	4	if a bond table entry was known before the authentication was performed successfully, this field communicates the content of the AppData field of that known bond table entry. Otherwise it will be set to zero.

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

**Result causes:**

Name	Description
LTP_CauseSuccess	Authentication is completed

### 3.11.22 Authentication result Confirmation → AuthResultCnf

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<cause><rem\_BD><AppData>

This command is used by the MDH to respond to an AuthResultInd of the MDC.

In case the authentication was indicated to be successfully and a link key was generated or changed, the response can be used to assign MDH specific information to a given bond table entry.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x21 → AuthResultCnf
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
cause	1	indicates the result of the transaction. For a description of defined causes please see the table "Result causes" in this chapter.
Rem_BD	6	Bluetooth device address of remote device authenticated to
AppData	4	This parameter can be used to assign MDH specific data to a given bond table entry

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

**Result causes:**

Name	Description
LTP_CauseSuccess	Operation is completed successfully

### 3.11.23 Authentication result Request Ind → AuthResultRequestInd

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<rem\_BD>

The MDC will send this message to the MDH to request information of a prior performed successful bonding procedure that was indicated by an AuthResultInd message. This message will only be used by the MDC if the MDC is configured to store bonding results externally.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x9B → AuthResultRequestInd
Copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
Lp	2	length of this LTP packet
rem_BD	6	Bluetooth device address of bonded remote device

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

### 3.11.24 Authentication result Request Conf → AuthResultRequestCnf

**Syntax:**

<cmd><copmsk><lp>[Header\_CRC8]<cause><rem\_BD><LinkKey><KeyType>

This command shall be used by the MDH to respond to an AuthResultRequestInd of the MDC.

In case the requested bonding information can be provided by the MDH it shall be returned with this message.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x1B → AuthResultRequestCnf
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
cause	1	Indicates the result of the transaction. For a description of defined causes please see the table “Result causes” in this chapter.
Rem_BD	6	Bluetooth device address of remote device authenticated to
LinkKey	16	128 bit link key
KeyType	1	Key Type of link key, See chapter 3.6.4 Link key Type for details.

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

**Result causes:**

Name	Description
LTP_CauseAccept	Requested Bonding information is provided
LTP_CauseReject	Requested Bonding information is not provided

### 3.11.25 Authentication delete Request → AuthDeleteReq

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<rem\_BD>

This command is used by the MDH to delete a formally established trusted relation to a remote device (bond table entry).

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x9C → AuthDeleteReq
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
rem_BD	6	Bluetooth device address of remote device authenticated to

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

### 3.11.26 Authentication delete Response → AuthDeleteRsp

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<cause><rem\_BD>

This command will be used by the MDC to response to a DeleteAuthReq of the MDH.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x1C → AuthDeleteRsp
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
cause	1	indicates the result of the transaction. For a description of defined causes please see the table “Result causes” in this chapter.
Rem_BD	6	Bluetooth device address of remote device authenticated to

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

**Result causes:**

Name	Description
LTP_CauseSuccess	Operation completed successfully
LTP_CauseInvalidParameter	Indicates that no trusted relation can be identified for the specified device.

---

### 3.11.27 Authentication list Request → AuthListReq

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<rem\_BD>

This command is used by the MDH to list all formally established trusted relations to remote devices (bond table entries).

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0xAC → AuthListReq
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
rem_BD	6	Bluetooth device address of remote device. If this parameter is set to "all zero", all trusted relations will be listed.

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

### 3.11.28 Authentication list Information → AuthListInfo

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<rem\_BD><reliable><AppData><Remote\_DeviceName>

This command is used by the MDC to indicate a single entry of the MDC internal list of authenticated devices.

Please be aware that multiple of this AuthListInfo messages might be generated as a result of a AuthListReq.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x2D → AuthListInfo
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
rem_BD	6	Bluetooth device address of remote device. If this parameter is set to "all zero", all trusted relations will be listed.
KeyType	1	Key Type of link key, See chapter 3.6.4 Link key Type for details.
AppData	4	This parameter indicates MDH assigned data of the given bond table entry.
Remote_DeviceName	N/A	UTF-8 coded, zero terminated string of octets including the user friendly name of the remote device. Note that this field may be empty

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

### 3.11.29 Authentication list Response → AuthListRsp

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<cause><rem\_BD>

With this message the MDC responds to an AuthListReq message.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x2C → AuthListRsp
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
cause	1	indicates the result of the operation. For a description of defined causes please see the table „Result causes“ in this chapter.
Rem_BD	6	Bluetooth device address of remote device authenticated to

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

**Result causes:**

Name	Description
LTP_CauseSuccess	Operation completed successfully

---

**3.11.30 Authorization request Indication → AuthorizationReqInd**

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<rem\_BD>

This command is used by the MDC to indicate the request for a User level authorization for a connection to remote device

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0xB2 → AuthorizationReqInd
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
rem_BD	6	Bluetooth device address of remote device authenticated to

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

### 3.11.31 Authorization request Confirmation → AuthorizationReqCnf

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<cause><rem\_BD>

This command is used by the MDH to respond to an AuthorizationReqInd of the MDC.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x32 → AuthorizationReqCnf
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
cause	1	indicates the result of the operation. For a description of defined causes please see the table „Result causes“ in this chapter.
Rem_BD	6	Bluetooth device address of remote device authenticated to

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

**Result causes:**

Name	Description
LTP_CauseAccept	Indicates that the User authorizes the remote device
LTP_CauseReject	Indicates that the User rejects the remote device

### 3.12 Data exchange

#### 3.12.1 Unsegmented Application Data Packet → DataUnsegmented

**Syntax:** <cmd><copmsk><lp>[loc\_MDL\_ID][returnCredits]  
[Header\_CRC8]<payload>

This command can be used to transfer an unsegmented application data packet APDU via LTP, that means that the whole APDU is included within this single LTP packet.

The maximum APDU size that can be send on a given MDL with this command results out of the parameters *max\_LTP\_size* and *max\_APDU\_size* of the ConnectMDLInfo message that indicated the opening of the MDL used.

If the loc\_MDL\_ID is not known by the MDC or the *max\_LTP\_size* / *max\_APDU\_size* values are exceeded by a message of the MDH, the MDC will ignore the data message and generate an internalEventInfo message with event type LTP\_InvalidDataReceived and cause LTP\_CauseInvalidParameter.

If credit based flow control is negotiated, this message shall only be send by the MDH if at least one dsCredit is still granted by the MDC. This message will only be send by the MDC if at least one usCredit is still granted by the MDH.

The sending of this message consumes one granted credits (reduces the number of granted credits by one).

For a description of the credit based flow control please refer to chapter [3.4.8 Flow control].

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x40 → DataUnsegmented
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
payload	N/A	application data APDU contained in this LTP packet. Please be aware that this parameter might have length zero.

**Optional fields:**

Name	msk	Description
Loc_MDL_ID	0x01	local mediated data link ID for this LTP packet ( <b>Default=0x01</b> )
returnCredits	0x02	<p>With this parameter one peer (MDH or MDC) can return upstream/downstream credits to the remote peer (e.g. the MDC return dsCredits to the MDH) (<b>Default=0x00</b>)</p> <p>This value will/shall only be indicated if the MDH granted maxTPDUusCredits in the CreateMDLCnf for this MDL.</p> <p>The overall credits granted to the MDC shall not exceed the maxTPDUusCredits indicated in the ConnectMDLInfo for this MDL.</p> <p>The overall credits granted to the MDH will not exceed the maxTPDUusCredits indicated in the ConnectMDLInfo for this MDL.</p>
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

### 3.12.2 Start of Segmented Application Data Packet → DataStartSegment

**Syntax:** <cmd><copmsk><lp>[loc\_MDL\_ID][returnCredits]  
[Header\_CRC8]<length\_APDU><payload>

This command can be used to start the transfer of a segmented application data packet APDU via LTP, that means that a single application data packet is segmented into multiple LTP packets and this LTP packet is used to transfer the first segment of the APDU.

The maximum APDU\_size that can be send on a given MDL with this command is limited to the size indicated with the parameter *max\_APDU\_size* of the ConnectMDLInfo message that indicated the opening of the MDL used.

The maximum segment size that can be send with this message results out of the parameter *max\_LTP\_size* of the ConnectMDLInfo message that indicated the opening of the MDL used.

If the loc\_MDL\_ID is not known by the MDC or the *max\_LTP\_size* / *max\_APDU\_size* values are exceeded by a message of the MDH, the MDC will ignore the data message and generate an internalEventInfo message with event type LTP\_InvalidDataReceived and cause LTP\_CauseInvalidParameter.

If credit based flow control is negotiated, this message shall only be send by the MDH if at least one dsCredit is still granted by the MDC. This message will only be send by the MDC is at least one usCredit is still granted by the MDH.

The sending of this message consumes on granted credits (reduces the number of granted credits by one).

For a description of the credit based flow control please refer to chapter [3.4.8 Flow control].

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x41 → DataStartSegment
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
length_APDU	2	whole length of unsegmented application data packet APDU
payload	N/A	application data APDU contained in this LTP packet. Please be aware that this parameter might have a length of zero.

**Optional fields:**

Name	msk	Description
Loc_MDL_ID	0x01	local mediated data link ID for this LTP packet ( <b>Default=0x01</b> )
returnCredits	0x02	<p>With this parameter one peer (MDH or MDC) can return upstream/downstream credits to the remote peer (e.g. the MDC return dsCredits to the MDH) (<b>Default=0x00</b>)</p> <p>This value will/shall only be indicated if the MDH granted maxTPDUusCredits in the CreateMDLCnf for this MDL.</p> <p>The overall credits granted to the MDC shall not exceed the maxTPDUusCredits indicated in the ConnectMDLInfo for this MDL.</p> <p>The overall credits granted to the MDH will not exceed the maxTPDUusCredits indicated in the ConnectMDLInfo for this MDL.</p>
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

### 3.12.3 End of Segmented Application Data Packet → DataEndSegment

**Syntax:** <cmd><copmsk><lp>[local MDL][returnCredits] [Header\_CRC8]<payload>

This command can be used to end the transfer of a segmented application data packet APDU via LTP, that means that a single application data packet is segmented into multiple LTP packets and this LTP packet is used to transfer the last segment of the APDU.

The maximum segment size that can be send with this message results out of the parameter *max\_LTP\_size* of the ConnectMDLInfo message that indicated the opening of the MDL used.

If the *loc\_MDL\_ID* is not known by the MDC or the *max\_LTP\_size* / *max\_APDU\_size* values are exceeded by a message of the MDH, the MDC will ignore the data message and generate an internalEventInfo message with event type *LTP\_InvalidDataReceived* and cause *LTP\_CauseInvalidParameter*.

If credit based flow control is negotiated, this message shall only be send by the MDH if at least one *dsCredit* is still granted by the MDC. This message will only be send by the MDC is at least one *usCredit* is still granted by the MDH.

The sending of this message consumes on granted credits (reduces the number of granted credits by one).

For a description of the credit based flow control please refer to chapter chapter [3.4.8 Flow control].

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x42 → DataEndSegment
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
payload	N/A	application data APDU contained in this LTP packet. Please be aware that this parameter might have length zero.

**Optional fields:**

Name	msk	Description
Loc_MDL_ID	0x01	local mediated data link ID for this LTP packet ( <b>Default=0x01</b> )
returnCredits	0x02	<p>With this parameter one peer (MDH or MDC) can return upstream/downstream credits to the remote peer (e.g. the MDC return dsCredits to the MDH) (<b>Default=0x00</b>)</p> <p>This value will/shall only be indicated if the MDH granted maxTPDUusCredits in the CreateMDLCnf for this MDL.</p> <p>The overall credits granted to the MDC shall not exceed the maxTPDUusCredits indicated in the ConnectMDLInfo for this MDL.</p> <p>The overall credits granted to the MDH will not exceed the maxTPDUusCredits indicated in the ConnectMDLInfo for this MDL.</p>
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

### 3.12.4 Continuation of Segmented Data Packet → DataContinueSegment

**Syntax:** <cmd><copmsk><lp>[loc\_MDL\_ID][returnCredits]  
[Header\_CRC8]<payload>

This command can be used to continue the transfer a segmented application data packet APDU via LTP, that means that a single application data packet is segmented into multiple LTP packets and this LTP packet is used to transfer intermediate segment of the APDU.

The maximum segment size that can be send with this message results out of the parameter *max\_LTP\_size* of the ConnectMDLInfo message that indicated the opening of the MDL used.

If the *loc\_MDL\_ID* is not known by the MDC or the *max\_LTP\_size* / *max\_APDU\_size* values are exceeded by a message of the MDH, the MDC will ignore the data message and generate an internalEventInfo message with event type *LTP\_InvalidDataReceived* and cause *LTP\_CauseInvalidParameter*.

If credit based flow control is negotiated, this message shall only be send by the MDH if at least one *dsCredit* is still granted by the MDC. This message will only be send by the MDC is at least one *usCredit* is still granted by the MDH.

The sending of this message consumes on granted credits (reduces the number of granted credits by one).

For a description of the credit based flow control please refer to chapter chapter [3.4.8 Flow control].

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x43 → DataContinueSegment
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
payload	N/A	application data APDU contained in this LTP packet. Please be aware that this parameter might have length zero.

**Optional fields:**

Name	msk	Description
Loc_MDL_ID	0x01	local mediated data link ID for this LTP packet ( <b>Default=0x01</b> )
returnCredits	0x02	<p>With this parameter one peer (MDH or MDC) can return upstream/downstream credits to the remote peer (e.g. the MDC return dsCredits to the MDH) (<b>Default=0x00</b>)</p> <p>This value will/shall only be indicated if the MDH granted maxTPDUUsCredits in the CreateMDLCnf for this MDL.</p> <p>The overall credits granted to the MDC shall not exceed the maxTPDUUsCredits indicated in the ConnectMDLInfo for this MDL.</p> <p>The overall credits granted to the MDH will not exceed the maxTPDUUsCredits indicated in the ConnectMDLInfo for this MDL.</p>
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

### 3.13 SDP Record / MDEP management

#### 3.13.1 Register HDP MDEP Request → RegisterHDPMDEPReq

**Syntax:**

<cmd><copmsk><lp>[Header\_CRC8]<locMDEP\_ID><DataType><Role><Name>

This command is used by the MDH to request a HDP MDEP entry in the SDP record of the MDC to be created. The MDC will respond with a RegisterHDPMDEPReq message specifying if the operation could be completed.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x91 → RegisterHDPMDEPReq
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
locMDEP_ID	1	identifier that will be used to indicate an incoming connection to this MDEP. Valid range is 0x00 to 0x7F. It is allowed to register multiple MDEPs with the same locMDEP_ID as long as they share the same role.
DataType	2	reference to a IEEE 11073 “device specialization” see [X15] for details
Role	1	defines if this MDEP refers to a MCAP / IEEE 11073 role <u>Possible Values are:</u> 0x00: Source / Agent 0x01: Sink / Manager 0x02: MCAP (this setting is <b>not</b> HDP compliant)
Name	N/A	UTF-8 formatted, zero terminated user friendly name of this MDEP. The maximum length of this parameter is target specific and will be truncated if the targets maximum is exceeded.

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

### 3.13.2 Register HDP MDEP Response → RegisterHDPMDEPRsp

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<cause><MDEP\_Handle>

This command is used by MDC to respond to a RegisterHDPMDEPReq of the MDH.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x11 → RegisterHDPMDEPRsp
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
cause	1	indicates the result of the transaction. For a description of defined causes please see the table “Result causes” in this chapter.
MDEP_Handle	1	reference to new MDEP if operation was successful

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

**Result causes:**

Name	Description
LTP_CauseSuccess	Indicates that a requested operation is completed successfully
LTP_CauseResourceError	The requested operation can't be completed due to lack of resources.
LTP_CauseInvalidParameter	Indicates that at least one parameter of the corresponding message was invalid, so the message is ignored

### 3.13.3 Register SPP MDEP Request → RegisterSPPMDEPReq

**Syntax:**

<cmd><copmsk><lp>[Header\_CRC8]<cause><locMDEP\_ID><DataType><reqAuth enticationIn><reqAuthorizationIn><reqMITMIn>

This command is used by the MDH to request a SPP MDEP entry in the SDP record of the MDC to be created. The MDC will respond with a RegisterSPPMDEPRsp message specifying if the operation could be completed.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0xB1 → RegisterSPPMDEPReq
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
locMDEP_ID	1	identifier that will be used to indicate an incoming connection to this MDEP. Valid range is 1 to 31. If multiple MDEPs are registered, each MDEP shall be registered with a unique locMDEP_ID.
DataType	2	This parameter defines a 16 bit UUID that refers to a RFCOMM based profile
reqAuthenticationIn	1	This parameter defines requirement for authentication to access this MDEP from a remote device.
reqAuthorizationIn	1	This parameter defines requirement for authorization to access this MDEP from a remote device.
reqMITMIn	1	This parameter defines requirement for MITM protection to access this MDEP from a remote device. Only valid in combination with RequireAuthenticationIn.
reqEncryptionIn	1	This parameter defines requirement for encryption to access this MDEP from a remote device. Only valid in combination with RequireAuthenticationIn.
Name	N/A	UTF-8 formatted, zero terminated user friendly name of this MDEP. The maximum length of this parameter is target specific and will be truncated if the targets maximum is exceeded.

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

### 3.13.4 Register SPP MDEP Response → RegisterSPPMDEPRsp

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<cause><MDEP\_Handle>

This command will be used by the MDC to response to a RegisterSPPMDEPReq of the MDH.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x31 → RegisterSPPMDEPRsp
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
cause	1	indicates the result of the transaction. For a description of defined causes please see the table “Result causes” in this chapter.
MDEP_Handle	1	reference to new MDEP if operation was successful

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

**Result causes:**

Name	Description
LTP_CauseSuccess	Indicates that a requested operation is completed successfully
LTP_CauseResourceError	The requested operation can't be completed due to lack of resources.
LTP_CauseInvalidParameter	Indicates that at least one parameter of the corresponding message was invalid, so the message is ignored

### 3.13.5 Release MDEP Request → ReleaseMDEPReq

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<MDEP\_Handle>

This command shall be used by the MDH to delete a SPP or HDP MDEP entry from the SDP record of the MDC. The MDC will respond with a ReleaseMDEPRsp indicating if the operation could be completed successfully.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x92 → ReleaseMDEPReq
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
MDEP_Handle	1	reference to MDEP to delete

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

### 3.13.6 Release MDEP Response → ReleaseMDEPRsp

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<cause>

This command will be used by the MDC to response to a ReleaseMDEPReq of the MDH.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x12 → ReleaseMDEPRsp
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
cause	1	indicates the result of the transaction. For a description of defined causes please see the table “Result causes” in this chapter.

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

**Result causes:**

Name	Description
LTP_CauseSuccess	Indicates that a requested operation is completed successfully
LTP_CauseInvalidParameter	Indicates that at least one parameter of the corresponding message was invalid, so the message is ignored

### 3.14 Inquiry, DID information and name discovery management

#### 3.14.1 Inquiry Request → InquiryReq

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]

When an InquiryReq is received by the MDC, the MDC will perform an inquiry, indicate remote devices found with InquiryDeviceInfo messages and indicate the finalization of the inquiry process with an InquiryRsp.

During an active InquiryReq, no other transactions shall be started by the MDH until the InquiryRsp message is received.

Please be aware that negotiated QoS requirements for existing channels are suspended for the duration of the inquiry process. Due to this fact, channels might be disconnected by the remote device or the local Bluetooth Implementation.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x94 → InquiryReq
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

### 3.14.2 Inquiry Response → InquiryRsp

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<cause>

This command is used by the MDC to respond to an InquiryReq of the MDH.

If the InquiryRsp sent by the MDC indicates success, a search for remote Bluetooth devices in the operation range was performed and devices found were indicated by InquiryDeviceInfo messages.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x14 → InquiryRsp
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
cause	1	indicates the result of the transaction. For a description of defined causes please see the table “Result causes” in this chapter.

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

**Result causes:**

Name	Description
LTP_CauseSuccess	Indicates that a requested operation is completed successfully
LTP_CauseInvalidState	Operation can not be performed due to invalid state

### 3.14.3 Inquiry Device Info → InquiryDeviceInfo

**Syntax:** <cmd><copmsk><lp>[rem\_DevClass][rem\_RSSI]  
[Header\_CRC8]<rem\_BD><rem\_DevName>

This command is used by the MDC to indicate a remote device found during an inquiry process to the MDH.

Please be aware that multiple of this InquiryDeviceInfo messages might be generated during an inquiry process.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x15 → InquiryDeviceInfo
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
rem_BD	6	Bluetooth device address of remote device
rem_DevName	NA	UTF-8 coded, zero terminated string of octets including the user friendly name of the remote device. Note that this field may be empty

**Optional fields:**

Name	msk	Description
rem_DevClass	0x07	Bluetooth Class of Device information. For further information refer to [X15]
rem_RSSI	0x08	Signed byte value indicating the absolute receive signal strength of the remote device. The range of indicated values will vary, as indication the range of +10 to -100 might be expected. Be aware that Bluetooth supports signal strength control mechanisms so this indicator can NOT be seen as a absolute reference to the distance of a given remote device.
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

### 3.14.4 Device name Request → DeviceNameReq

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<rem\_BD>

This command shall be used by the MDH to request the user friendly name of a remote device.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0xAE → DeviceNameReq
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
rem_BD	6	Bluetooth device address of remote device

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

### 3.14.5 Device name Response → DeviceNameRsp

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<cause><rem\_BD>< rem\_DevName>

This command is used by the MDC to respond to a DeviceNameReq of the MDH.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x2E → DeviceNameRsp
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
cause	1	indicates the result of the transaction. For a description of defined causes please see the table “Result causes” in this chapter.
Rem_BD	6	Bluetooth device address of remote device
rem_DevName	NA	UTF-8 coded, zero terminated string of octets including the user friendly name of the remote device. Note that this field may be empty

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

**3.14.6 DID Device Info → DIDDeviceInfo**

**Syntax:** <cmd><copmsk><lp>[VendorIDSource]  
[Header\_CRC8]<rem\_BD><rem\_VendorID><rem\_ProductID><rem\_Version><rem\_DevName>

This command is used by the MDC to indicate that the remote device specified in an HDPDiscoveryReq supports HDP and includes all device specific information for that remote device.

In case this message results out of a HDPDiscoveryReq, the next message to be expected after reception of a DIDDeviceInfo message is at least one HDPServiceInfo message.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x17 → DIDDeviceInfo
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
Lp	2	length of this LTP packet
rem_BD	6	Bluetooth device address of remote device
rem_VendorID	2	Uniquely identifies the vendor of the device within the realm of remote_VendorID_Source
rem_ProductID	2	Uniquely identifies different products made by the vendor. These Ids are managed by the vendors themselves
rem_Version	2	A numeric expression identifying the device release number in binary-coded decimal notation (version 2.1.3 is value 0x0213)
rem_DevName	NA	UTF-8 coded, zero terminated string of octets including the user friendly name of the remote device. Be aware that this field might be empty or truncated.

---

**Optional fields:**

Name	msk	Description
VendorIDSource	0x03	This attribute designates which organization assigned the rem_VendorID attribute ( <b>Default=0x0002</b> ) <u>Possible Values are:</u> 0x0001: Bluetooth SIG assigned Device ID Vendor ID 0x0002: USB Implementer's Forum assigned Vendor ID
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

### 3.15 HDP service discovery management

#### 3.15.1 HDP Discovery Request → HDPDiscoveryReq

**Syntax:** <cmd><copmsk><lp>[rem\_MDEP\_Type][rem\_MDEP\_Role][Header\_CRC8]  
<rem\_BD>

**Command description:**

This command is used by the MDH to request HDP/MCAP relevant information from a remote Bluetooth device in range.

Once a HDPDiscoveryReq is received by the MDC, the MDC will perform a service discovery on the remote device identified by the <rem\_BD> field.

If the remote device supports HDP, all PnP specific information of that remote device will be indicated to the MDH using a DIDDeviceInfo message, all IEEE 11073-20601 relevant information of the device will be indicated with one or multiple HDPServiceInfo messages, and all data-endpoint relevant information of the device will be indicated with one or multiple HDPEndpointInfo messages.

Please be aware that negotiated QoS requirements for existing channels are suspended for the duration of the service discovery process. Due to this fact, channels might be disconnected by the remote device or the local Bluetooth Implementation.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x96 → HDPDiscoveryReq
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
rem_BD	6	Bluetooth device address of remote device

**Optional fields:**

Name	msk	Description
rem_MDEP_Type	0x03	MDEP Data Type / IEEE Device specialization identifier to be searched for. <b>(Default = 0 all MDEP types)</b>
rem_MDEP_Role	0x04	Identifies if the search shall address Source/Agent or a Sink/manager endpoints. <b>(Default = 0x02)</b>  <u>Possible values are:</u> 0x00: Source/Agent 0x01: Sink/manager 0x02: Source or Sink
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

### 3.15.2 HDP Discovery Response → HDPDiscoveryRsp

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<cause>

This command is used by the MDC to respond to a HDPDiscoveryReq.

If the HDPDiscoveryRsp sent by the MDC indicates success, a service discovery to the remote device was performed and HDP relevant information found was indicated to the MDH using DIDDeviceInfo, HDPServiceInfo, and HDPEndpointInfo messages.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x16 → HDPDiscoveryRsp
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
cause	1	indicates the result of the transaction. For a description of defined causes please see the table “Result causes” in this chapter.

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

**Result causes:**

Name	Description
LTP_CauseSuccess	Indicates that a requested operation is completed successfully
LTP_CauseConnectionLost	Indicates that the remote device did not respond (e.g. device is out of range)
LTP_CauseResourceError	Indicates that the operation failed due to lack of resources.

### 3.15.3 HDP Service Info → HDPServiceInfo

**Syntax:** <cmd><copmsk><lp>[DataFormat][MCAPFeatures]  
[Header\_CRC8]<rem\_C\_PSM><rem\_D\_PSM><rem\_ServName>

This command is used by MDC to indicate that the remote device specified in a HDPDiscoveryReq supports HDP and includes all service specific information for a service of that remote device.

Please be aware that multiple HDPServiceInfo messages might be generated for a given remote device.

The next message to be expected after reception of a HDPServiceInfo message is at least one HDPEndpointInfo message.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x18 → HDPServiceInfo
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
rem_C_PSM	2	control channel PSM of remote device
rem_D_PSM	2	data channel PSM of remote device
rem_ServName	NA	UTF-8 coded, zero terminated string of octets including the user friendly name of the remote service. Be aware that this field might be empty or truncated

**Optional fields:**

Name	msk	Description
DataFormat	0x01	This attribute designates Data Exchange Specification is supported <b>(Default=0x01)</b> <u>Possible Values are:</u> 0x01 ISO/IEEE 11073-20601
MCAP_Features	0x02	This attribute is a one byte bit-mask that indicates the MCAP procedures that are supported by this HDP service <b>(Default=0x00)</b> <u>Possible Bitmasks fields are:</u> 0x02: Supports Reconnect Initiation 0x04: Supports Reconnect Acceptance 0x08: Supports Clock Synchronization Protocol 0x10: Supports Sync-Master Role
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

### 3.15.4 HDP Endpoint Info → HDPEndpointInfo

**Syntax:**

```
<cmd><copmsk><lp>[Header_CRC8]<rem_MDEP_ID><rem_MDEP_Role><rem_MDEP_Type><rem_MDEPName>
```

This command will be used by the MDC to indicate that a remote device that was targeted by a HDPDiscoveryReq supports the HDP and includes all endpoint specific information for one endpoint supported by the service that was indicated via the last HDPServiceInfo message generated by the MDC.

To address this endpoint via a ConnectMDLReq message, some additional information included in the preceding DIDDeviceInfo and HDPServiceInfo messages are needed.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x19 → HDPEndpointInfo
copmsk	1	0x00 if no optional parameters applied otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
rem_MDEP_ID	1	MDEP_ID of remote endpoint to connect a MDL to
rem_MDEP_Role	1	indicates whether this MDEP is a Source/Agent or a Sink/Manager <u>Possible Values are:</u> 0x00: Source/Agent 0x01: Sink/Manager
rem_MDEP_Type	2	MDEP Data Type / IEEE Device specialization identifier to be addressed by this endpoint
rem_MDEPName	NA	UTF-8 formatted, zero terminated unicode string of octets including the user friendly name of the remote endpoint. Be aware that this field might be empty or truncated

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

### 3.16 SPP service discovery management

#### 3.16.1 SPP Discovery Request → SPPDiscoveryReq

**Syntax:** <cmd><copmsk><lp>[remote\_DID\_Discovery][Header\_CRC8]<rem\_BD><rem\_MDEP\_Type>

This command is used by the MDH to request SPP/RFCOMM relevant information from a remote Bluetooth device in the operation range.

Once a SPPDiscoveryReq is received by the MDC, the MDC will perform a service discovery on the remote device identified by the <remote\_BD> field.

If the remote device supports SPP/RFCOMM, all SPP/RFCOMM and data-endpoint relevant information of the device will be indicated with one or multiple SPPServiceEndpointInfo messages.

Please be aware that negotiated QoS requirements for existing channels are suspended for the duration of the service discovery process. Due to this fact, channels might be disconnected by the remote device or the local Bluetooth Implementation.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0xAF → SPPDiscoveryReq
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
rem_BD	6	Bluetooth device address of remote device
rem_MDEP_Type	2	16 bit UUID that refers to a RFCOMM based profile to search for (please be referred to [X15] for details).

**Optional fields:**

Name	msk	Description
remote_DID_Discovery	0x01	Identifies if the search shall also perform a DID discovery. If this parameter is set to 0x01, a DID (Device ID) discovery of the remoted device will be performed in addition to a SPP service discovery. If the Remote device supports DID, the fetched DID information will be indicated by a DIDDeviceInfo message prior to a possible SPPEndpointInfo message ( <b>Default = 0x00</b> ) <u>Possible values are:</u> 0x00: Do not perform DID discovery 0x01: Perform a DID discovery
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

### 3.16.2 SPP Discovery Response → SPPDiscoveryRsp

**Syntax:** <cmd><copmsk><lp>[Header\_CRC8]<cause>

This command is used by the MDC to respond to a SPPDiscoveryReq.

If the SPPDiscoveryRsp sent by the MDC indicates success, a service discovery to the remote device was performed and SPP relevant information found was indicated to the MDH using SPPEndpointInfo messages.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x2F → SPPDiscoveryRsp
copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
lp	2	length of this LTP packet
cause	1	indicates the result of the transaction. For a description of defined causes please see the table “Result causes” in this chapter.

**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

**Result causes:**

Name	Description
LTP_CauseSuccess	Indicates that a requested operation is completed successfully
LTP_CauseConnectionLost	Indicates that the remote device did not respond (e.g. device is out of range)
LTP_CauseResourceError	Indicates that the operation failed due to lack of resources.

### 3.16.3 SPP Endpoint Info → SPPEndpointInfo

**Syntax:**

<cmd><copmsk><lp>[Header\_CRC8]<cause><rem\_BD><rem\_MDEP\_Type><rem\_MDEP\_ID><rem\_MDEPName>

This command will be used by the MDC to indicate that a remote device that was targeted by a SPPDiscoveryReq supports RFCOMM based profiles and includes all endpoint specific information for one endpoint supported.

**Mandatory fields:**

Name	Size	Description
cmd opcode	1	0x30 → SPPEndpointInfo
Copmsk	1	0x00 if no optional parameters applied, otherwise see optional packets fields for this message below
Lp	2	length of this LTP packet
rem_BD	6	Bluetooth device address of remote device
rem_MDEP_Type	2	16 bit UUID that refers to a RFCOMM based profile to search for (please be referred to [X15] for details).
Rem_MDEP_ID	1	MDEP_ID of remote endpoint to connect a MDL to
rem_MDEPName	NA	UTF-8 formatted, zero terminated unicode string of octets including the user friendly name of the remote endpoint. Be aware that this field might be empty or truncated

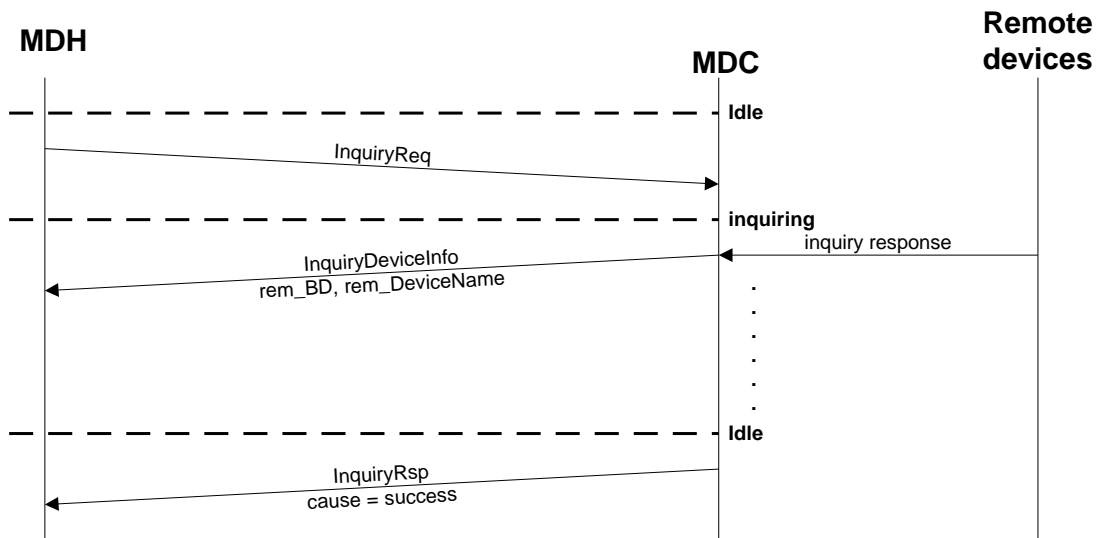
**Optional fields:**

Name	msk	Description
Header_CRC8	0x80	CRC checksum for the first 4 bytes of this packet (opcode, copmsk, lp)

## 4 Message flow examples

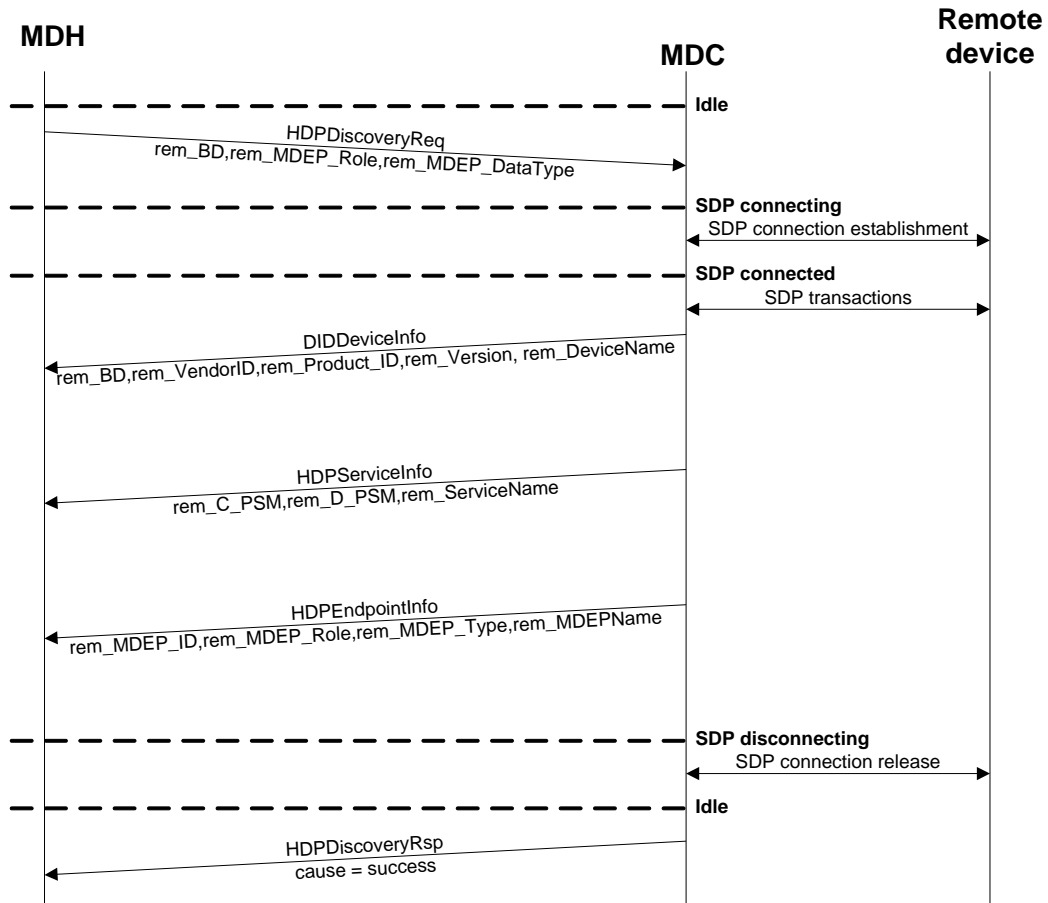
### 4.1 Inquiry

This section describes the message flow for the case that an MDH successfully initiates an inquiry and one or multiple devices are found.



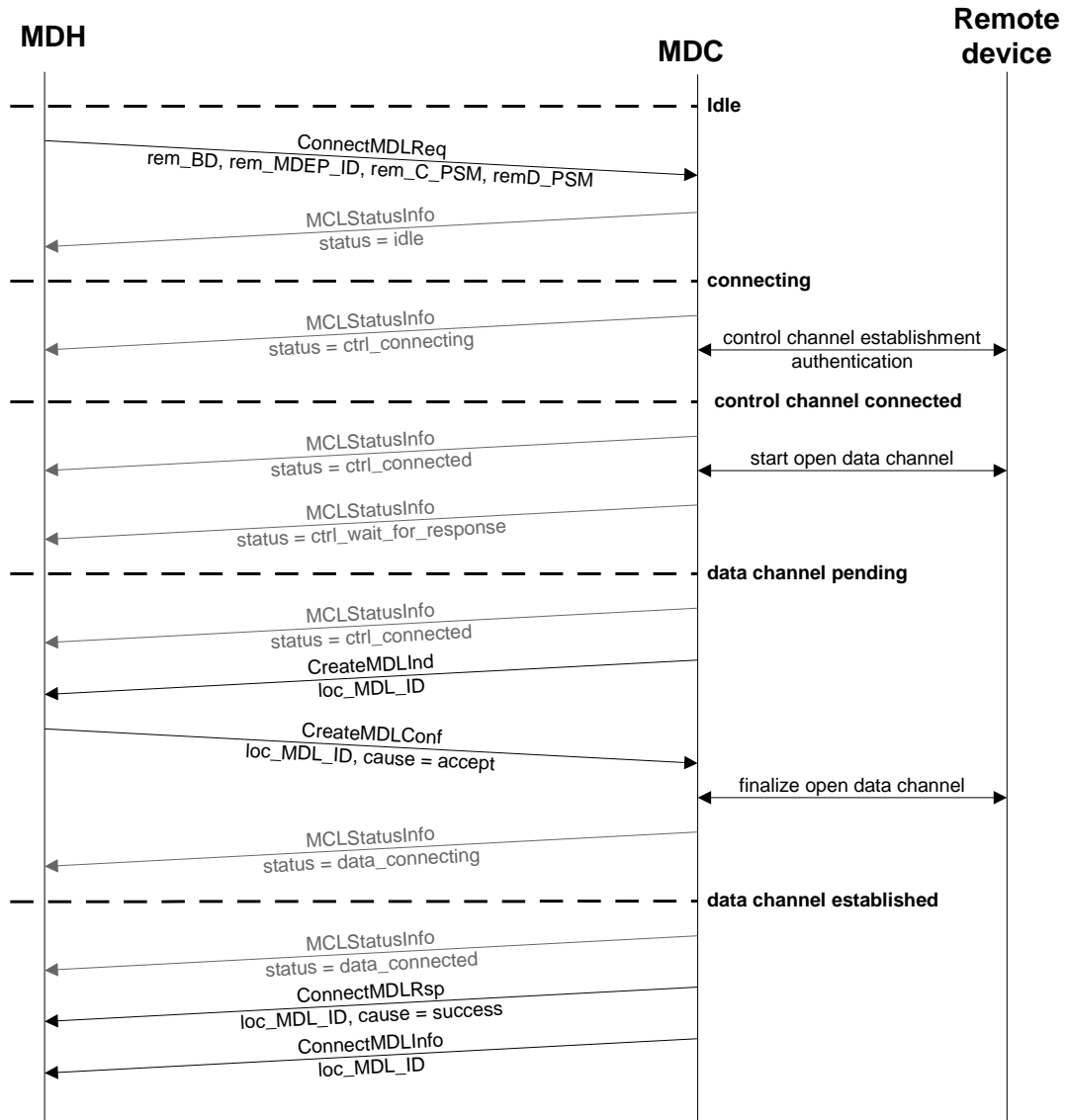
## 4.2 HDP Service Discovery

This section describes the message flow for the case that the MDH successfully initiates a HDP discovery to a remote device and one or multiple HDP-Services with one or more endpoints are found.



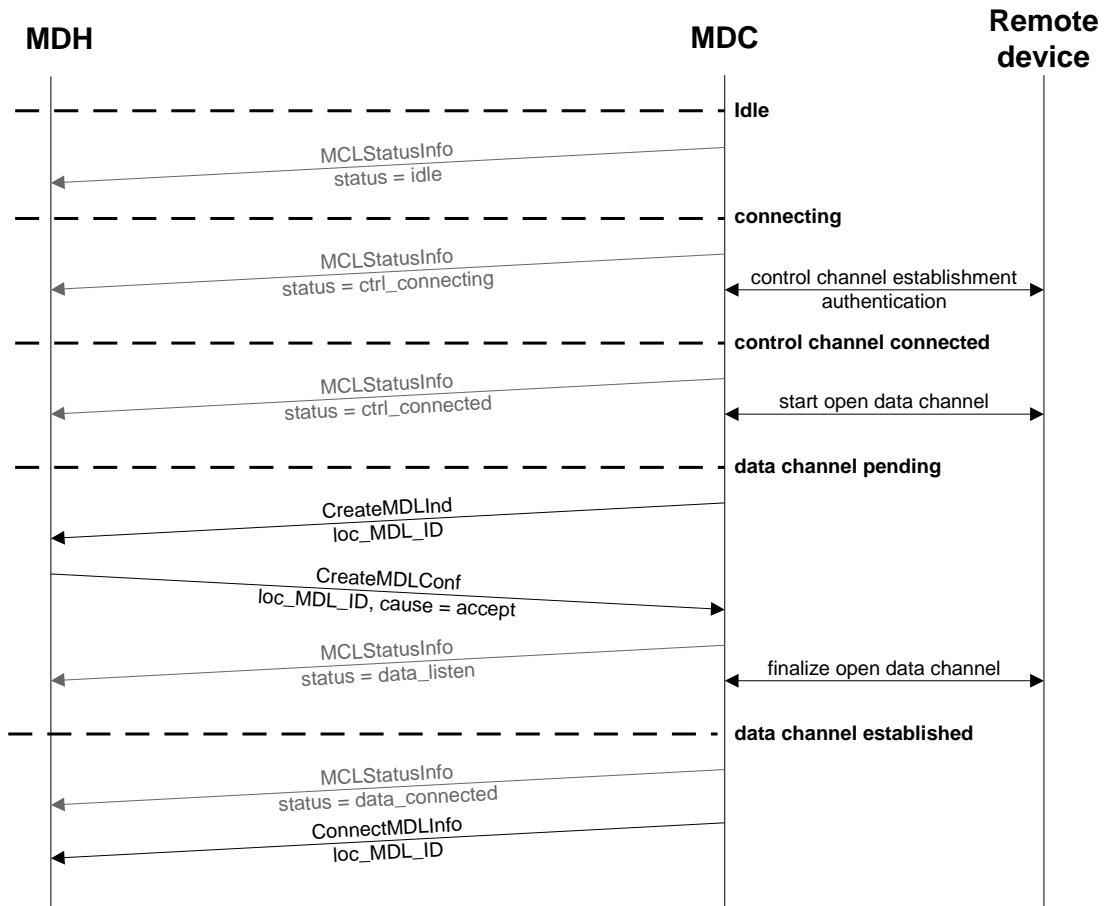
### 4.3 Outgoing new HDP channel connection establishment

This section describes the message flow for the case that a MDH successfully requests a new MDL connection to a remote device.



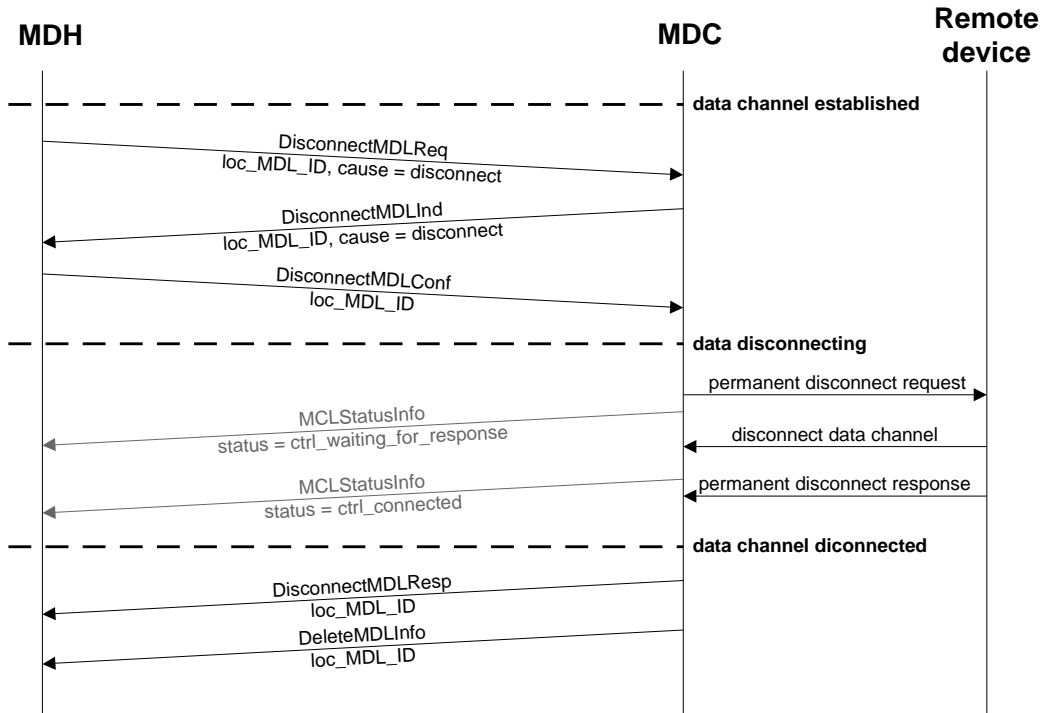
#### 4.4 Incoming new HDP channel connection establishment

This section describes the message flow for the case that the MDC indicates a new MDL connection from a remote device that is accepted by the MDH.



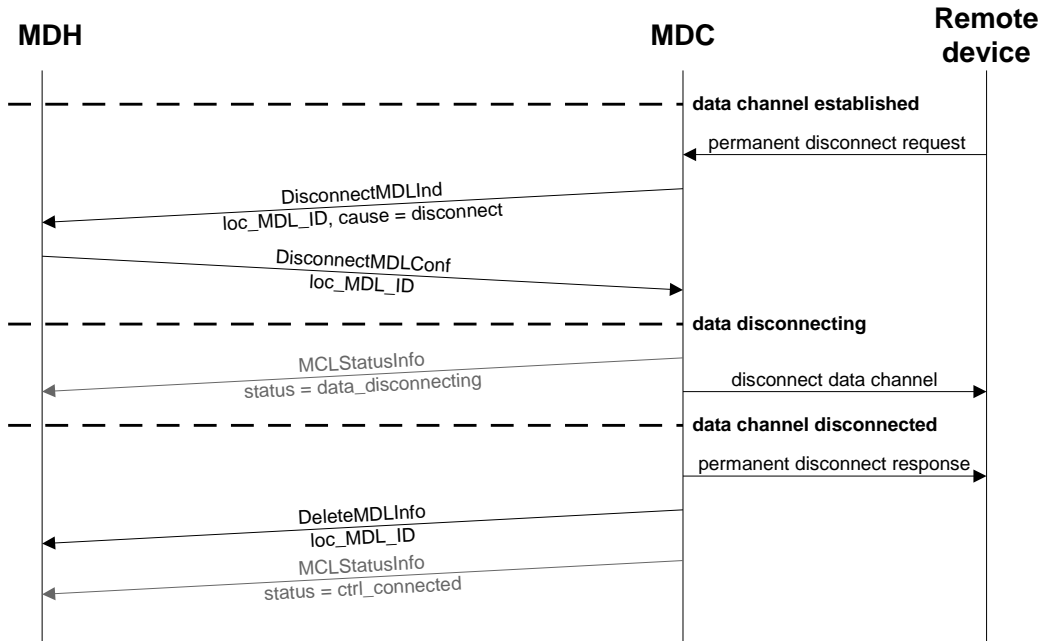
#### 4.5 Outgoing permanent disconnection of HDP data channel

This section describes the message flow for the case that a MDH requests a permanent disconnection of an established HDP data channel.



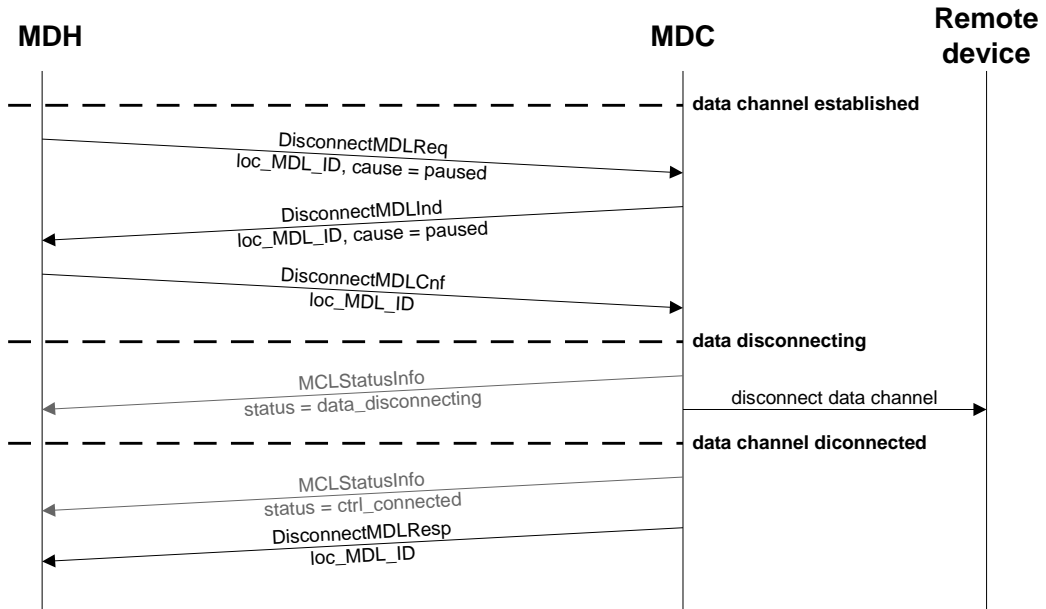
#### 4.6 Incoming permanent disconnection of HDP data channel

This section describes the message flow for the case that the MDC indicates a permanent disconnection from the remote device.



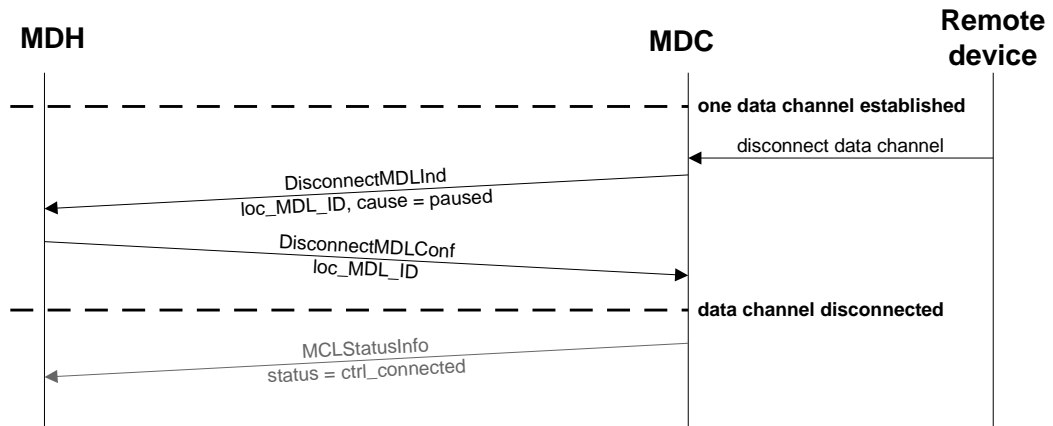
#### 4.7 Outgoing temporary disconnection of HDP data channel

This section describes the message flow for the case that the MDH requests a temporary disconnection of an established data channel.



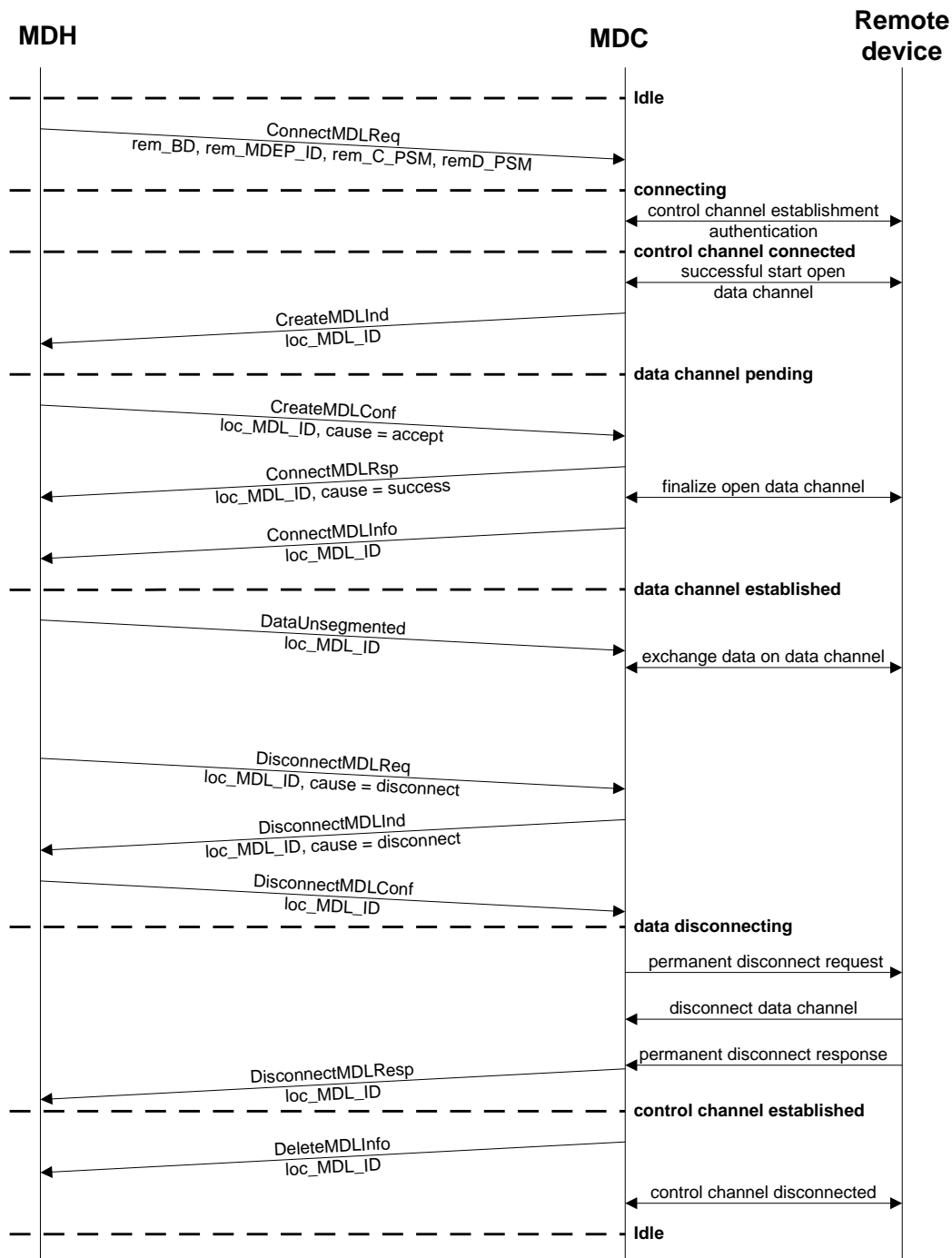
#### 4.8 Incoming temporary disconnection of HDP data channel

This section describes the message flow for the case that the MDC indicates a temporary disconnection from the remote device.



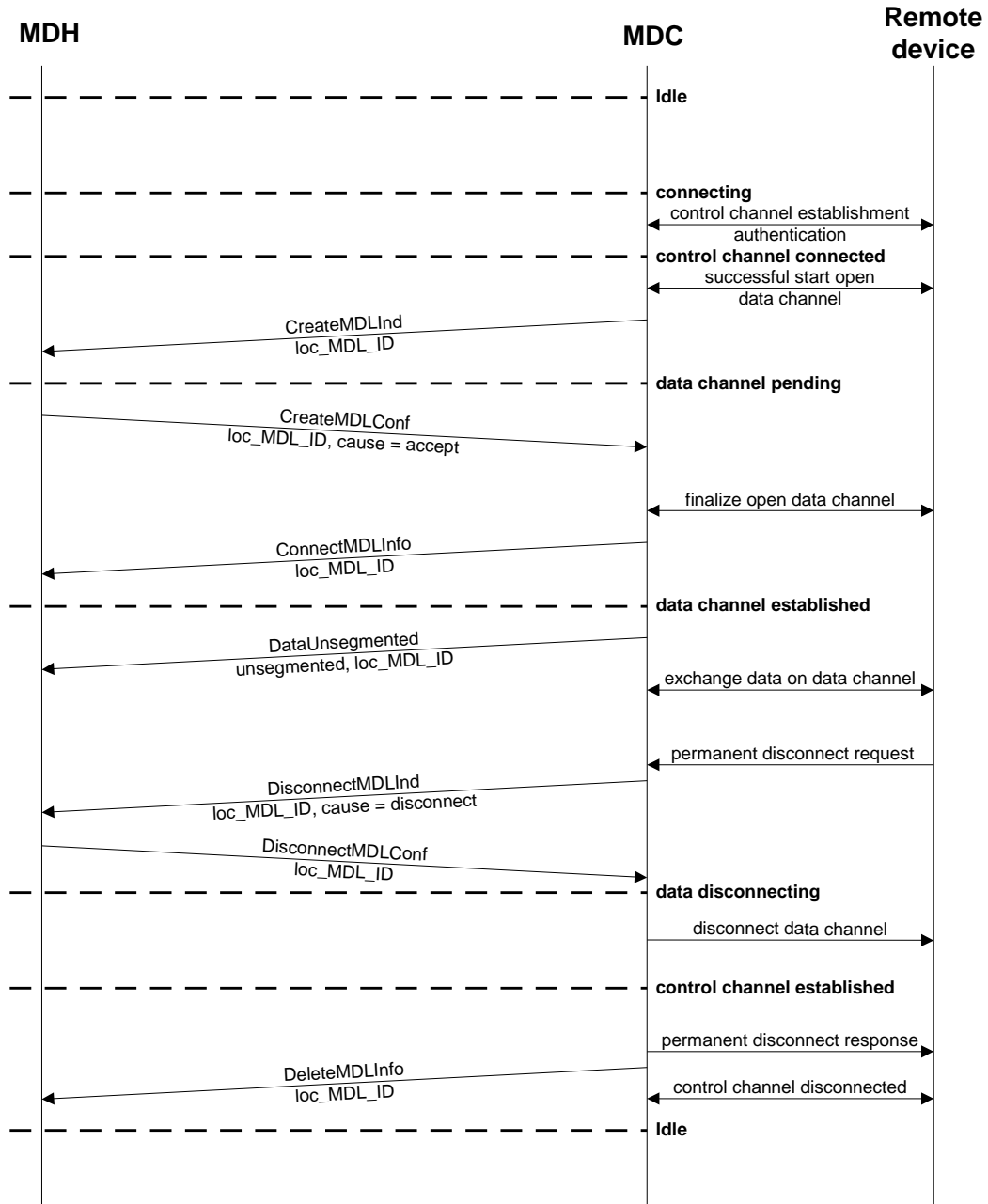
#### 4.9 Outgoing HDP session with data exchange

This section describes the message flow for the case that an outgoing connection is established, data is exchanged and connection becomes disconnected by local device (MCLStatusInfo messages are not shown for lucidity).



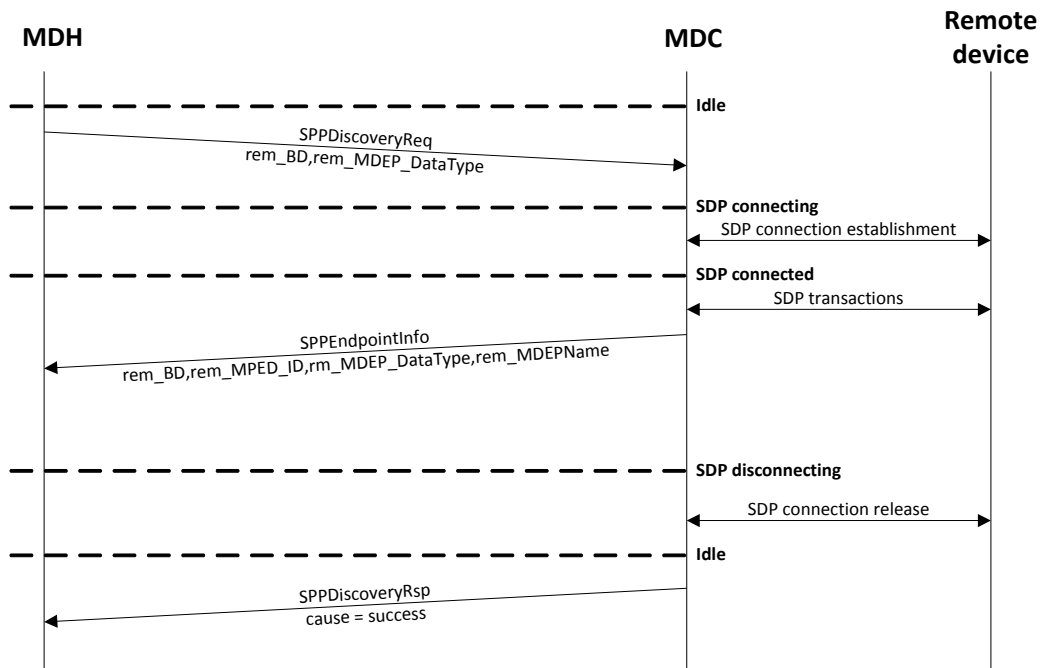
#### 4.10 Incoming HDP session with data exchange

This section describes the message flow for the case that an incoming connection is accepted, data is exchanged and the connection becomes disconnected by remote device (MCLStatusInfo messages are not shown for lucidity).



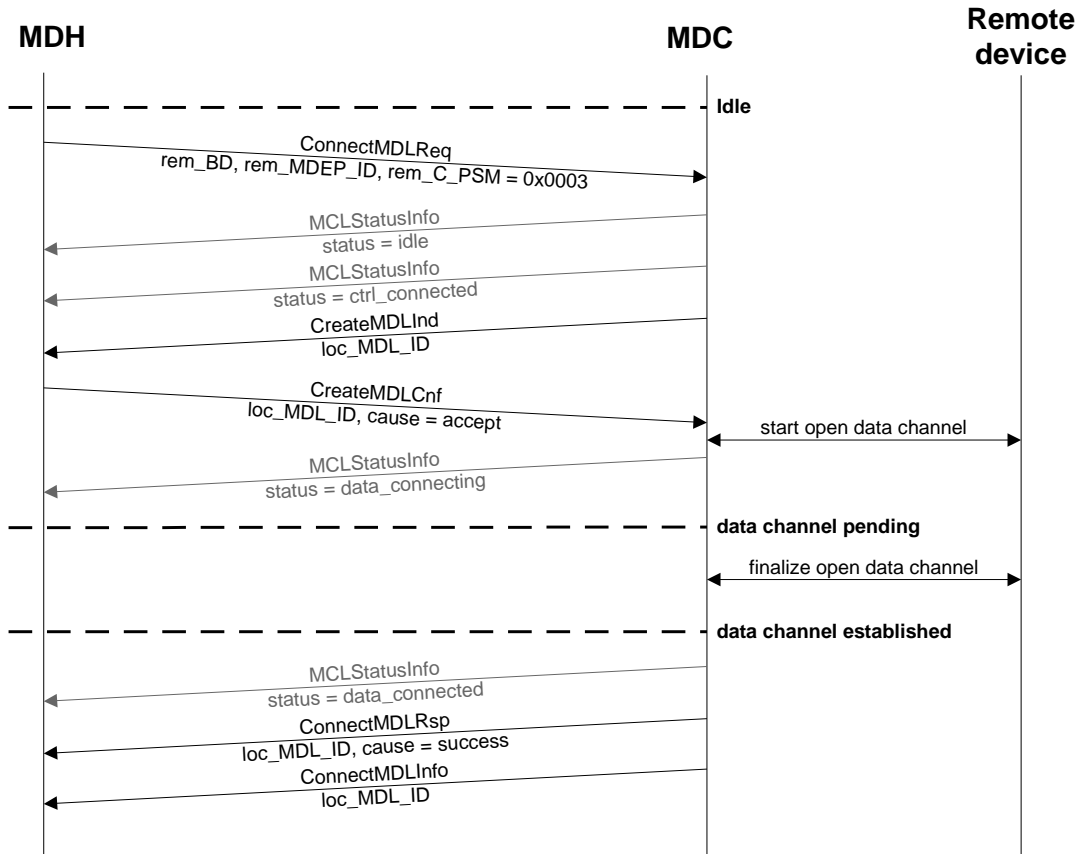
### 4.11 SPP Service Discovery

This section describes the message flow for the case that the MDH successfully initiates an SPP discovery to a remote device and one or multiple SPP endpoints are found.



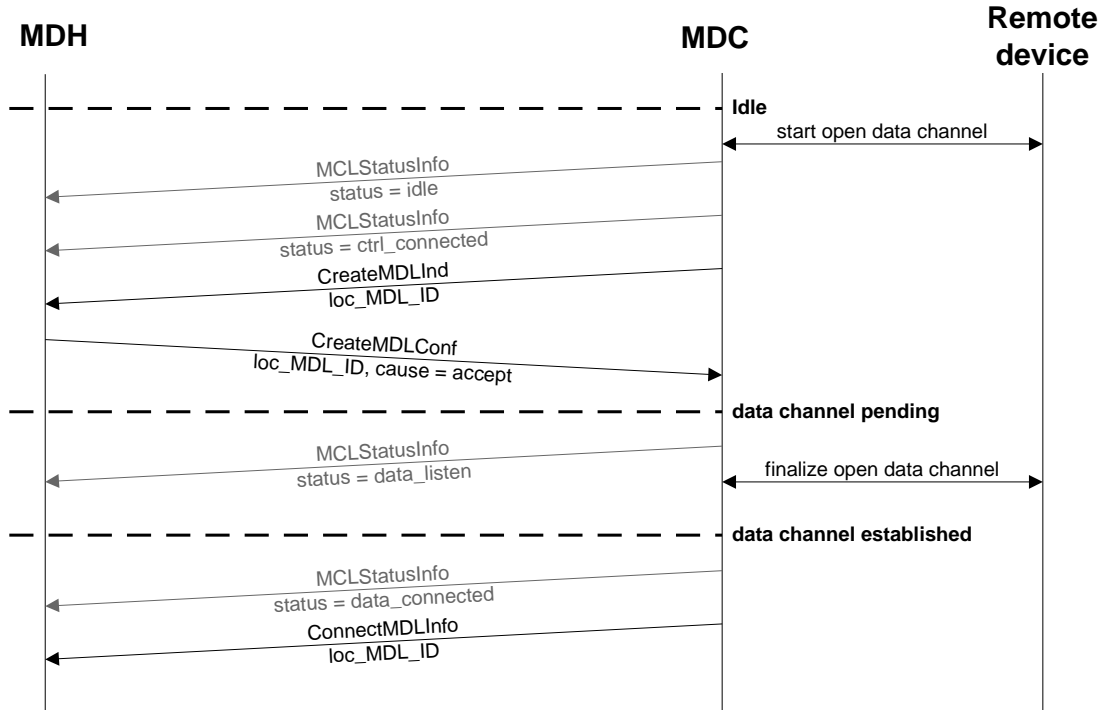
#### 4.12 Outgoing new SPP channel connection establishment

This section describes the message flow for the case that an MDH successfully requests a new MDL connection to a remote device.



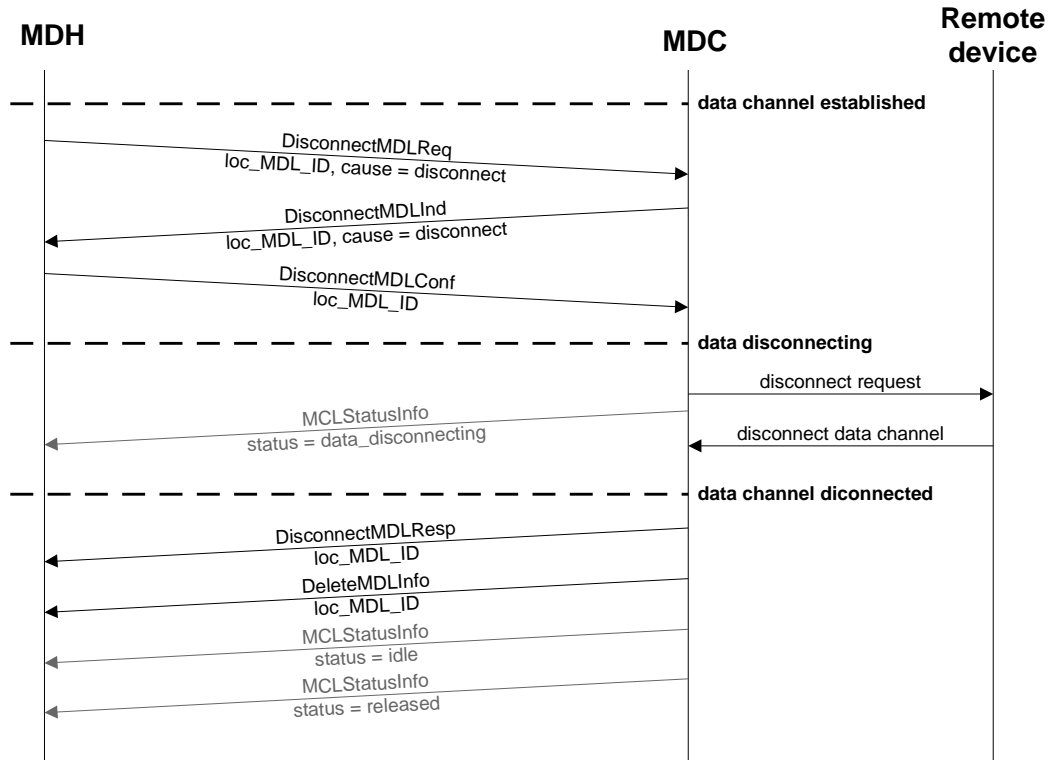
### 4.13 Incoming new SPP channel connection establishment

This section describes the message flow for the case that the MDC indicates a new MDL connection from a remote device that is accepted by the MDH.



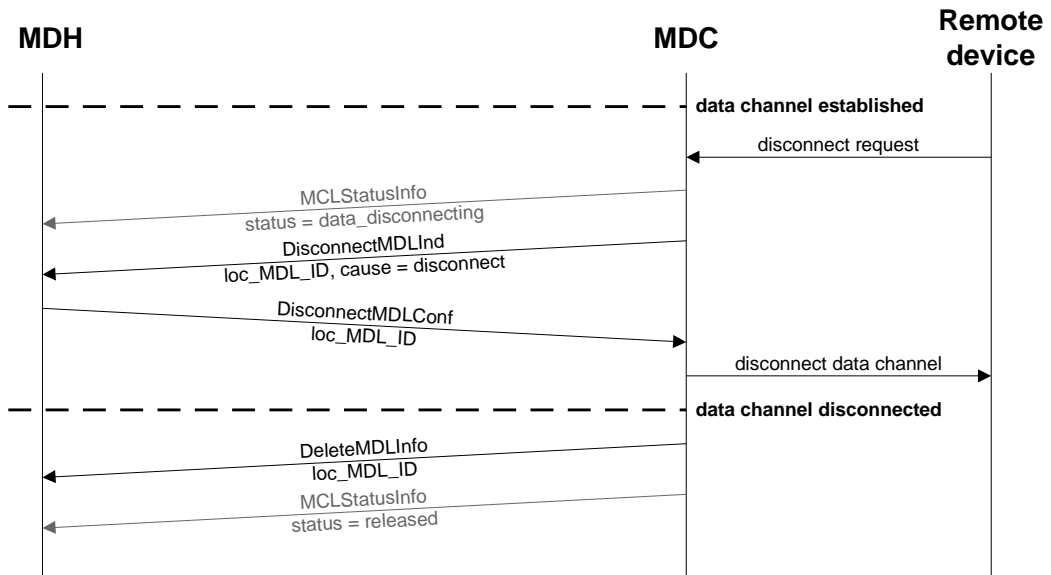
#### 4.14 Outgoing disconnection of SPP data channel

This section describes the message flow for the case that an MDH requests a disconnection of an established SPP data channel.



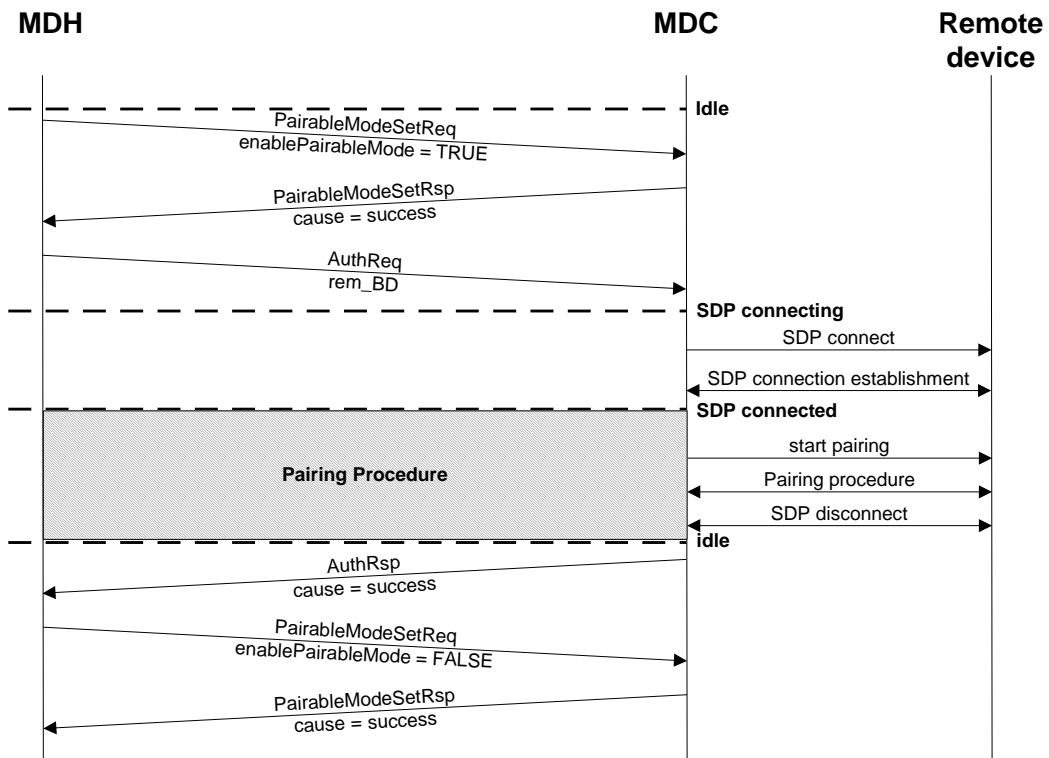
#### 4.15 Incoming disconnection of SPP data channel

This section describes the message flow for the case that the MDC indicates a disconnection from the remote device.



#### 4.16 Successful active Bonding

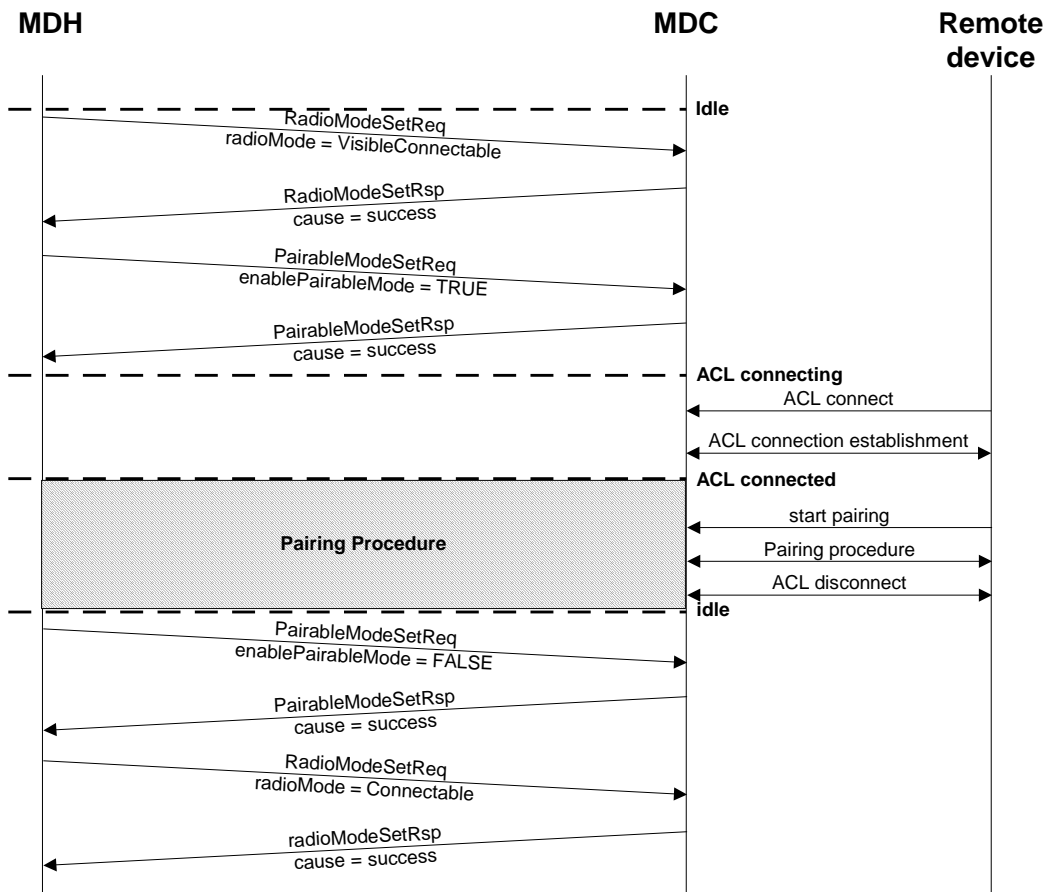
This section describes a message flow example for the case that the MDH initiates a successful bonding to a remote device.



Note: To perform a bonding, an ACL connection is necessary. The SDP connection is established for this purpose, no SDP activity is performed.

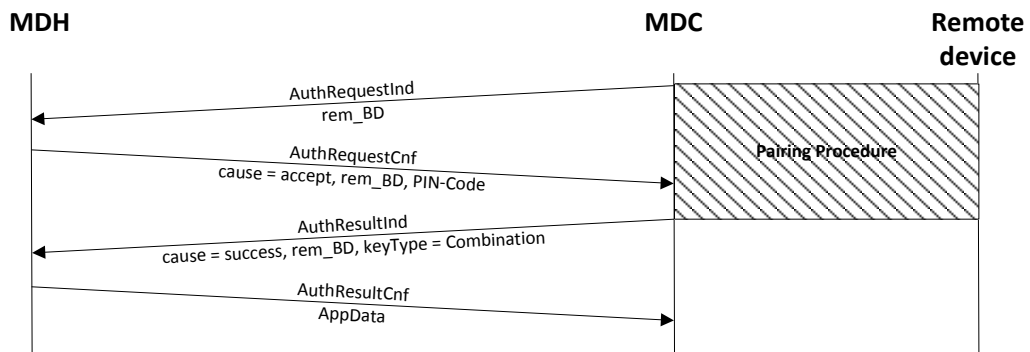
### 4.17 Successful passive Bonding

This section describes a message flow example for the case that the MDH accepts a successful bonding of a remote device.



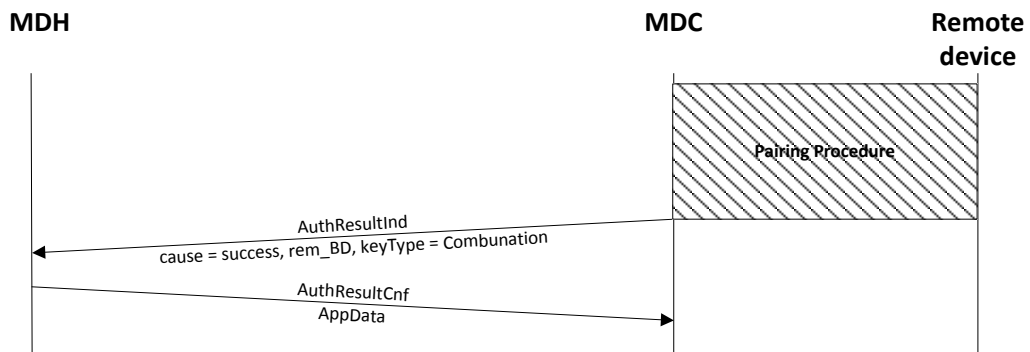
#### 4.18 Pairing procedure: Legacy (PIN entry)

This section describes the message flow for a successful pairing procedure with PIN entry



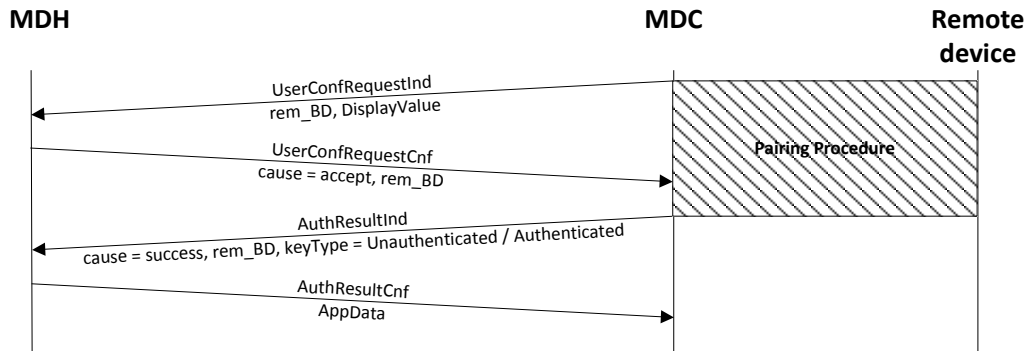
#### 4.19 Pairing procedure: Just Works

This section describes the message flow for a successful pairing procedure with SSP “Just Works”



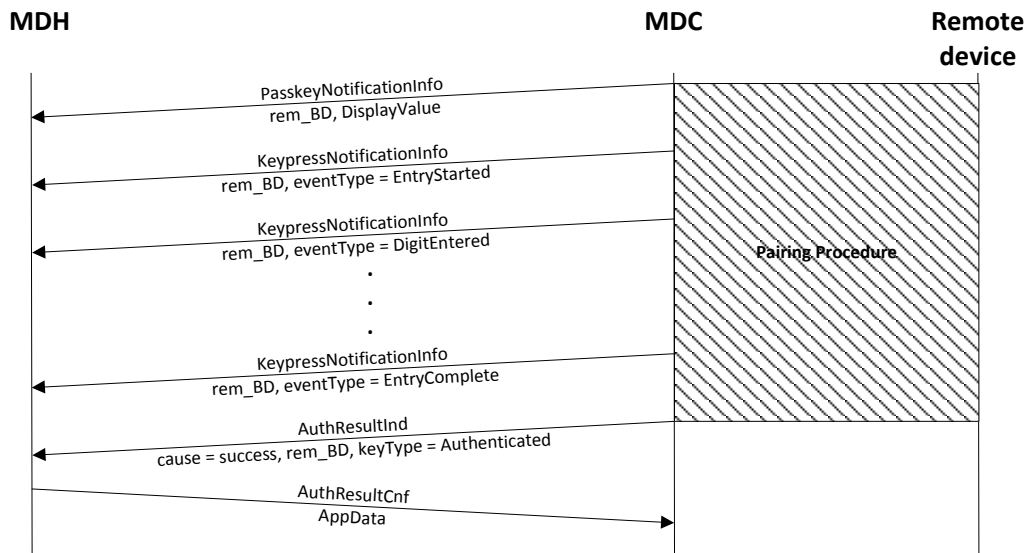
#### 4.20 Pairing Procedure: Numeric Comparison (Local “DisplayYesNo”)

This section describes the message flow for a successful pairing procedure with SSP “Numeric Comparison” and local “DisplayYesNo” IO Capabilities.



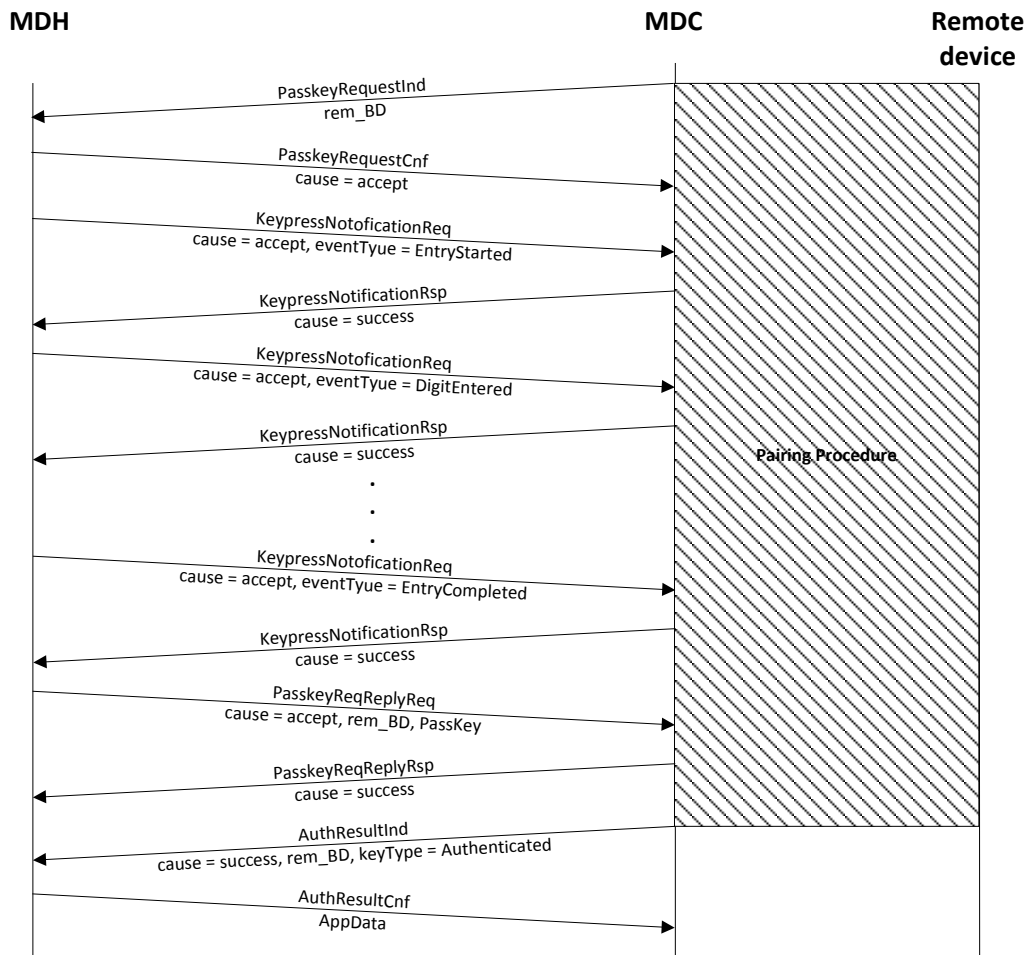
#### 4.21 Pairing Procedure: Passkey Entry (Local Display)

This section describes the message flow for a successful pairing procedure with SSP "Passkey Entry" and local "DisplayYesNo"/"DisplayOnly" IO Capabilities.



## 4.22 Pairing Procedure: Passkey Entry (Local KeyboardOnly)

This section describes the message flow for a successful pairing procedure with SSP “Passkey Entry” and local “KeyboardOnly” IO Capabilities.



---

## 5 References

- X1 BlueMod+HDP\_Command\_Reference
- X2 MCAP (Multi-Channel Adaptation Protocol) Specification 1.0
- X3 IEEE 11073-10101 – Point-of-care medical device communication: Nomenclature
- X4 IEEE 11073-10400 Health informatics – Personal health device communication – Device specialization – Common framework
- X5 IEEE 11073-20601 Health informatics – Personal health device communication – Application profile – Optimized exchange protocol
- X6 IEEE 11073-10404 Health informatics – Personal health device communication – Device specialization – Pulse oximeter
- X7 IEEE 11073-10406 Health informatics – Personal health device communication – Device specialization – Heart rate monitor
- X8 IEEE 11073-10407 Health informatics – Personal health device communication – Device specialization – Blood pressure monitor
- X9 IEEE 11073-10408 Health informatics – Personal health device communication – Device specialization – Thermometer
- X10 IEEE 11073-10415 Health informatics – Personal health device communication – Device specialization – Weighing scale
- X11 IEEE 11073-10417 Health informatics – Personal health device communication – Device specialization – Glucose meter
- X12 Bluetooth SIG, Bluetooth Assigned Numbers, <http://www.bluetooth.org/assigned-numbers.htm>
- X13 Health Device Profile Specification
- X14 Bluetooth SIG, Bluetooth 2.1 (or later) Core specification
- X15 Bluetooth assigned numbers
- X16 Bluetooth SSP whitepapers

## 6 History

Version	Release Date	By	Change description
r01	02.04.08	fh	first release
r02	06.11.08	ka	first "public" version
r03	19.03.09	fh	3rd release
r04	26.02.10	ka	first "full featured" version
r05	24.08.10	ka	NEW: Chapter 3.4.7
r06	26.08.10	ka	Added SSP support, SPP/RFCOMM support, configurator access via LTP tunnel, channel based flow control for data messages, MCL and ACL link status infos and LTP message header CRC.  Renamed "HDPDeviceInfo" message in "DIDDeviceInfo" Added Device and Firmware information to "ActInfo" message
r07	20.10.10	ka	Added chapter "MDC Usage and Procedures" Added description for range of "MDEP_Entry Handle" Added additional information for MCAP echo channel handling Added additional information for HDP APDU segmentation
r08	05.04.11	ka	Fixed "User Confirmation Request Indication" definition: (removed <cause> out of Syntax description) Fixed "Authentication result Indication" definition: (renamed <reliable> with <KeyType> in Syntax description) Fixed "Authentication list Information" definition: (renamed <reliable> with <KeyType> in Syntax description) Fixed "SPP Discovery Response" definition: (removed <BD> out of Syntax description) Fixed "SPP Endpoint Info" definition: (removed <cause> out of Syntax description) renamed "reliable" with "KeyType" in various security related flow charts Clarified requirements of "Pairable Mode set Request" definition. Clarified limitations of "Authentication Result Indication" definition. Added "AuthResultRequestInd" and "AuthResultRequestCnf" message definitions
r09	08.04.11	ka	Defined default for parameter LinkConfigType in message CreateMDLCnf  add optional parameter max_Tx_LTP_Size to ActInfo message to allow MDH to optimize memory usage

## 7 Appendix A: All LTP Messages sorted by Opcode

Opcode	message name	Flow	Page
0x04	ConnectMDLInfo	MDH ← MDC	50
0x05	ConnectMDLRsp	MDH ← MDC	39
0x06	CreateMDLCnf	MDH → MDC	49
0x07	DeleteMDLInfo	MDH ← MDC	51
0x08	DisconnectMDLRsp	MDH ← MDC	45
0x09	DisconnectMDLCnf	MDH → MDC	47
0x0A	ReconnectMDLRsp	MDH ← MDC	41
0x0B	ReconnectMDLCnf	MDH → MDC	43
0x0C	ExitRsp	MDH ← MDC	25
0x0D	ExitInfo	MDH ← MDC	26
0x0E	ActInfo	MDH ← MDC	31
0x0F	ACLStatusInfo	MDH ← MDC	53
0x10	AuthRequestCnf	MDH → MDC	59
0x11	RegisterHDPMDPRsp	MDH ← MDC	88
0x12	ReleaseMDPRsp	MDH ← MDC	92
0x13	ResetRsp	MDH ← MDC	28
0x14	InquiryRsp	MDH ← MDC	94
0x15	InquiryDeviceInfo	MDH ← MDC	95
0x16	HDPDiscoveryRsp	MDH ← MDC	100
0x17	DIDDeviceInfo	MDH ← MDC	101
0x18	HDPServiceInfo	MDH ← MDC	101
0x19	HDPEndpointInfo	MDH ← MDC	102
0x1A	AuthRsp	MDH ← MDC	57
0x1B	AuthResultRequestCnf	MDH → MDC	72
0x1C	AuthDeleteRsp	MDH ← MDC	73
0x1D	InternalEventInfo	MDH ← MDC	34
0x1E	UserConfRequestCnf	MDH → MDC	60
0x1F	PasskeyRequestCnf	MDH → MDC	61
0x20	RemoteOOBReqCnf	MDH → MDC	67
0x21	AuthResultCnf	MDH → MDC	70
0x22	ConfigTunnelRsp	MDH ← MDC	30
0x23	ConfigTunnelInfo	MDH ← MDC	29
0x24	RadioModeSetRsp	MDH ← MDC	33
0x25	MCLStatusInfo	MDH ← MDC	52
0x26	PairableModeSetRsp	MDH ← MDC	56
0x27	PasskeyReqReplyRsp	MDH ← MDC	63
0x28	PasskeyNotificationInfo	MDH ← MDC	64
0x29	KeypressNotificationRsp	MDH ← MDC	65
0x2A	KeypressNotificationInfo	MDH ← MDC	65
0x2B	LocalOOBRsp	MDH ← MDC	68

0x2C	AuthListRsp	MDH ← MDC	76
0x2D	AuthListInfo	MDH ← MDC	75
0x2E	DeviceNameRsp	MDH ← MDC	96
0x2F	SPPDiscoveryRsp	MDH ← MDC	104
0x30	SPPEndpointInfo	MDH ← MDC	105
0x31	RegisterSPPMDEPRsp	MDH ← MDC	90
0x32	AuthorizationReqCnf	MDH → MDC	78
0x40	DataUnsegmented	MDH ↔ MDC	79
0x41	DataStartSegment	MDH ↔ MDC	81
0x42	DataEndSegment	MDH ↔ MDC	83
0x43	DataContinueSegment	MDH ↔ MDC	85
0x85	ConnectMDLReq	MDH → MDC	37
0x86	CreateMDLInd	MDH ← MDC	48
0x88	DisconnectMDLReq	MDH → MDC	44
0x89	DisconnectMDLInd	MDH ← MDC	46
0x8A	ReconnectMDLReq	MDH → MDC	40
0x8B	ReconnectMDLInd	MDH ← MDC	42
0x8C	ExitReq	MDH → MDC	24
0x90	AuthRequestInd	MDH ← MDC	58
0x91	RegisterHDPMDPEPReq	MDH → MDC	87
0x92	ReleaseMDEPReq	MDH → MDC	91
0x93	ResetReq	MDH → MDC	27
0x94	InquiryReq	MDH → MDC	93
0x96	HDPDiscoveryReq	MDH → MDC	99
0x9A	AuthReq	MDH → MDC	57
0x9B	AuthResultRequestInd	MDH ← MDC	71
0x9C	AuthDeleteReq	MDH → MDC	72
0x9E	UserConfRequestInd	MDH ← MDC	60
0x9F	PasskeyRequestInd	MDH ← MDC	61
0xA0	RemoteOOBReqInd	MDH ← MDC	66
0xA1	AuthResultInd	MDH ← MDC	69
0xA2	ConfigTunnelReq	MDH → MDC	29
0xA4	RadioModeSetReq	MDH → MDC	32
0xA6	PairableModeSetReq	MDH → MDC	54
0xA7	PasskeyReqReplyReq	MDH → MDC	62
0xA9	KeypressNotificationReq	MDH → MDC	64
0xAB	LocalOOBReq	MDH → MDC	67
0xAC	AuthListReq	MDH → MDC	74
0xAE	DeviceNameReq	MDH → MDC	96
0xAF	SPPDiscoveryReq	MDH → MDC	103
0xB1	RegisterSPPMDEPReq	MDH → MDC	88
0xB2	AuthorizationReqInd	MDH ← MDC	77

## 8 Appendix B: All LTP Messages sorted by Name

Opcode	message name	Flow	Page
0x0F	ACLStatusInfo	MDH ← MDC	53
0x0E	ActInfo	MDH ← MDC	31
0x9C	AuthDeleteReq	MDH → MDC	72
0x1C	AuthDeleteRsp	MDH ← MDC	73
0x2D	AuthListInfo	MDH ← MDC	75
0xAC	AuthListReq	MDH → MDC	74
0x2C	AuthListRsp	MDH ← MDC	76
0x32	AuthorizationReqCnf	MDH → MDC	78
0xB2	AuthorizationReqInd	MDH ← MDC	77
0x9A	AuthReq	MDH → MDC	57
0x10	AuthRequestCnf	MDH → MDC	59
0x90	AuthRequestInd	MDH ← MDC	58
0x21	AuthResultCnf	MDH → MDC	70
0xA1	AuthResultInd	MDH ← MDC	69
0x9B	AuthResultRequestCnf	MDH → MDC	72
0x1B	AuthResultRequestInd	MDH ← MDC	71
0x1A	AuthRsp	MDH ← MDC	57
0x23	ConfigTunnelInfo	MDH ← MDC	29
0xA2	ConfigTunnelReq	MDH → MDC	29
0x22	ConfigTunnelRsp	MDH ← MDC	30
0x04	ConnectMDLInfo	MDH ← MDC	50
0x85	ConnectMDLReq	MDH → MDC	37
0x05	ConnectMDLRsp	MDH ← MDC	39
0x06	CreateMDLCnf	MDH → MDC	49
0x86	CreateMDLInd	MDH ← MDC	48
0x43	DataContinueSegment	MDH ↔ MDC	85
0x42	DataEndSegment	MDH ↔ MDC	83
0x41	DataStartSegment	MDH ↔ MDC	81
0x40	DataUnsegmented	MDH ↔ MDC	79
0x07	DeleteMDLInfo	MDH ← MDC	51
0xAE	DeviceNameReq	MDH → MDC	96
0x2E	DeviceNameRsp	MDH ← MDC	96
0x17	DIDDeviceInfo	MDH ← MDC	101
0x09	DisconnectMDLCnf	MDH → MDC	47
0x89	DisconnectMDLInd	MDH ← MDC	46
0x88	DisconnectMDLReq	MDH → MDC	44
0x08	DisconnectMDLRsp	MDH ← MDC	45
0x0D	ExitInfo	MDH ← MDC	26
0x8C	ExitReq	MDH → MDC	24
0x0C	ExitRsp	MDH ← MDC	25

0x96	HDPDiscoveryReq	MDH → MDC	99
0x16	HDPDiscoveryRsp	MDH ← MDC	100
0x19	HDPEndpointInfo	MDH ← MDC	102
0x18	HDPServiceInfo	MDH ← MDC	101
0x15	InquiryDeviceInfo	MDH ← MDC	95
0x94	InquiryReq	MDH → MDC	93
0x14	InquiryRsp	MDH ← MDC	94
0x1D	InternalEventInfo	MDH ← MDC	34
0x2A	KeypressNotificationInfo	MDH ← MDC	65
0xA9	KeypressNotificationReq	MDH → MDC	64
0x29	KeypressNotificationRsp	MDH ← MDC	65
0xAB	LocalOOBReq	MDH → MDC	67
0x2B	LocalOOBRsp	MDH ← MDC	68
0x25	MCLStatusInfo	MDH ← MDC	52
0xA6	PairableModeSetReq	MDH → MDC	54
0x26	PairableModeSetRsp	MDH ← MDC	56
0x28	PasskeyNotificationInfo	MDH ← MDC	64
0xA7	PasskeyReqReplyReq	MDH → MDC	62
0x27	PasskeyReqReplyRsp	MDH ← MDC	63
0x1F	PasskeyRequestCnf	MDH → MDC	61
0x9F	PasskeyRequestInd	MDH ← MDC	61
0xA4	RadioModeSetReq	MDH → MDC	32
0x24	RadioModeSetRsp	MDH ← MDC	33
0x0B	ReconnectMDLCnf	MDH → MDC	43
0x8B	ReconnectMDLInd	MDH ← MDC	42
0x8A	ReconnectMDLReq	MDH → MDC	40
0x0A	ReconnectMDLRsp	MDH ← MDC	41
0x91	RegisterHDPMDPEPRReq	MDH → MDC	87
0x11	RegisterHDPMDPEPRsp	MDH ← MDC	88
0xB1	RegisterSPPMDPEPRReq	MDH → MDC	88
0x31	RegisterSPPMDPEPRsp	MDH ← MDC	90
0x92	ReleaseMDEPRReq	MDH → MDC	91
0x12	ReleaseMDEPRsp	MDH ← MDC	92
0x20	RemoteOOBReqCnf	MDH → MDC	67
0xA0	RemoteOOBReqInd	MDH ← MDC	66
0x93	ResetReq	MDH → MDC	27
0x13	ResetRsp	MDH ← MDC	28
0xAF	SPPDiscoveryReq	MDH → MDC	103
0x2F	SPPDiscoveryRsp	MDH ← MDC	104
0x30	SPPEndpointInfo	MDH ← MDC	105
0x1E	UserConfRequestCnf	MDH → MDC	60
0x9E	UserConfRequestInd	MDH ← MDC	60

---

Stollmann Entwicklungs- und Vertriebs-GmbH  
Mendelssohnstraße 15 D  
22761 Hamburg  
Germany

Phone: +49 (0)40 890 88-0  
Fax: +49 (0)40 890 88-444  
E-mail: [info@stollmann.de](mailto:info@stollmann.de)  
[www.stollmann.de](http://www.stollmann.de)