Part C

LINK MANAGER PROTOCOL

This specification describes the Link Manager Protocol (LMP) which is used for link set-up and control. The signals are interpreted and filtered out by the Link Manager on the receiving side and are not propagated to higher layers.

CONTENTS

1 GENERAL

LMP messages are used for link set-up, security and control. They are transferred in the payload instead of L2CAP and are distinguished by a reserved value in the L_CH field of the payload header. The messages are filtered out and interpreted by LM on the receiving side and are not propagated to higher layers.

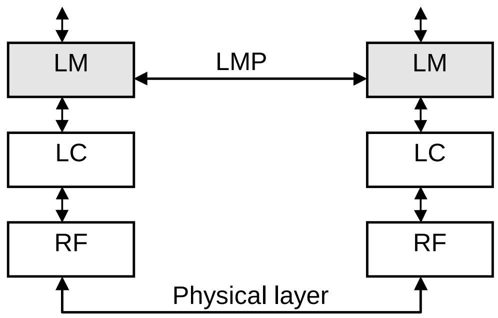


Figure 1.1: Link Manager's place on the global scene.

Link Manager messages have higher priority than user data. This means that if the Link Manager needs to send a message, it shall not be delayed by the L2CAP traffic, although it can be delayed by many retransmissions of individual baseband packets.

We do not need to explicitly acknowledge the messages in LMP since LC, see Baseband Specification Section 5, on page 67, provides us with a reliable link.

The time between receiving a baseband packet carrying an LMP PDU and sending a baseband packet carrying a valid response PDU, according to the procedure rules in Section 3 on page 193, must be less than 30 seconds.

2 FORMAT OF LMP

LM PDUs are always sent as single-slot packets and the payload header is therefore one byte. The two least significant bits in the payload header determine the logical channel. For LM PDUs these bits are set..

L_CH code	Logical Channel	Information
00	na	undefined
01	UA/I	Continuing L2CAP message
10	UA/I	Start L2CAP message
11	LM	LMP message

Table 2.1: Logical channel L_CH field contents.

The FLOW bit in the payload header is always zero and is ignored on the receiving side. Each PDU is assigned a 7bit OpCode used to uniquely identify different types of PDUs, see Table 5.1 on page 227. The OpCode and a one-bit transaction ID are positioned in the first byte of the payload body. The transaction ID is positioned in the LSB. It is 0 if the PDU belongs to a transaction initiated by the master and 1 if the PDU belongs to a transaction initiated by the slave. If the PDU contains one or more parameters these are placed in the payload starting at the second byte of the payload body. The number of bytes used depends on the length of the parameters. If an SCO link is present using HV1 packets and length of *content* is less than 9 bytes the PDUs can be transmitted in DV packets. Otherwise DM1 packets must be used. All parameters have little endian format, i.e. the least significant byte is transmitted first.

The source/destination of the PDUs is determined by the AM_ADDR in the packet header.

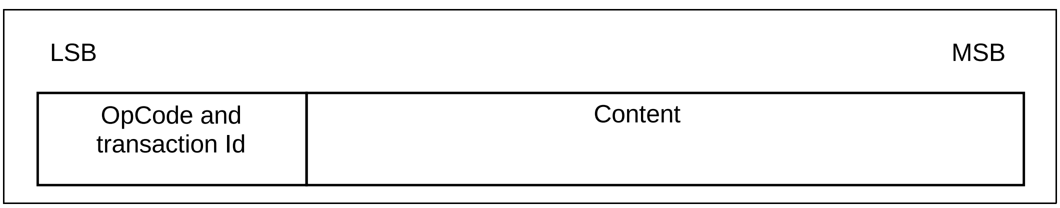


Figure 2.1: Payload body when LM PDUs are sent.

Each PDU is either mandatory or optional. The M/O field in the tables of Section 3 indicates this. The LM does not need to be able to transmit a PDU that is optional. The LM must recognise all optional PDUs that it receives and, if a response is required, send a valid response according to the procedure rules in Section 3. The reason that should be used in this case is *unsupported LMP feature*. If the optional PDU that is received does not require a response, no response is sent. Which of the optional PDUs a device supports can be requested, see Section 3.11 on page 206.

3 THE PROCEDURE RULES AND PDUS

Each procedure is described and depicted with a sequence diagram. The following symbols are used in the sequence diagrams:

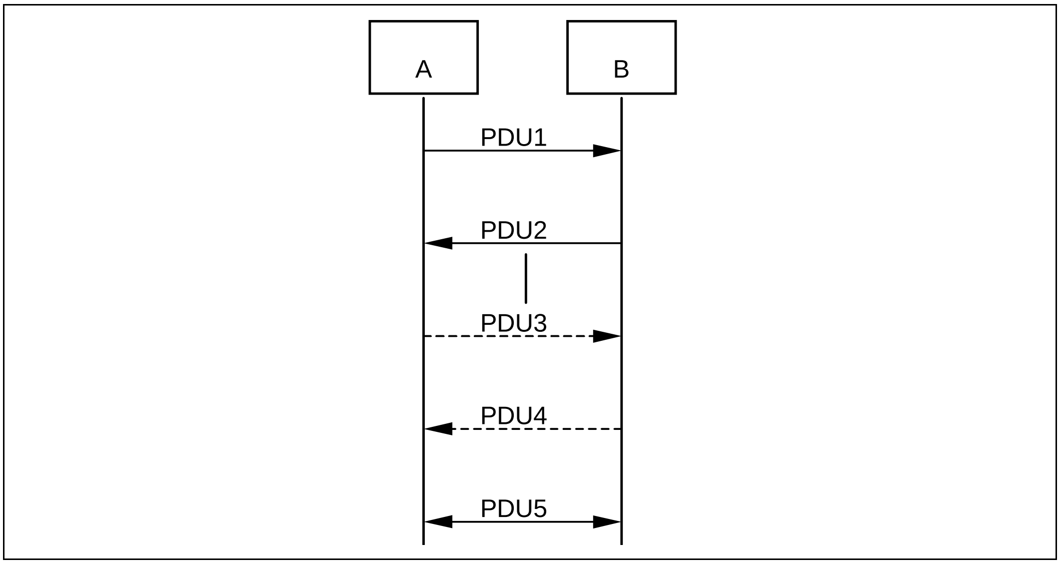


Figure 3.1: Symbols used in sequence diagrams.

PDU1 is a PDU sent from A to B. PDU2 is a PDU sent from B to A. PDU3 is a PDU that is optionally sent from A to B. PDU4 is a PDU that is optionally sent from B to A. PDU5 is a PDU sent from either A or B. A vertical line indicates that more PDUs can optionally be sent.

3.1 GENERAL RESPONSE MESSAGES

The PDUs LMP_accepted and LMP_not_accepted are used as response messages to other PDUs in a number of different procedures. The PDU LMP_accepted includes the opCode of the message that is accepted. The PDU LMP_not_accepted includes the opCode of the message that is not accepted and the reason why it is not accepted.

M/O	PDU	Contents
M	LMP_accepted	op code
M	LMP_not_accepted	op code reason

Table 3.1: General response messages.

3.2 AUTHENTICATION

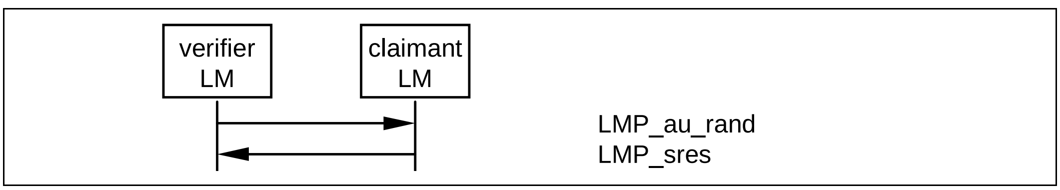
The authentication procedure is based on a challenge-response scheme as described in Baseband Specification Section 14.4, on page 169. The verifier sends an LMP_au_rand PDU which contains a random number (the challenge) to the claimant. The claimant calculates a response, which is a function of the challenge, the claimant's BD_ADDR and a secret key. The response is sent back to the verifier, which checks if the response was correct or not. How the response should be calculated is described in Baseband Specification Section 14.5.1, on page 171. A successful calculation of the authentication response requires that two devices share a secret key. How this key is created is described in Section 3.3 on page 196. Both the master and the slave can be verifiers. The following PDUs are used in the authentication procedure:

M/O	PDU	Contents
M	LMP_au_rand	random number
M	LMP_sres	authentication response

Table 3.2: PDUs used for authentication.

3.2.1 Claimant has link key

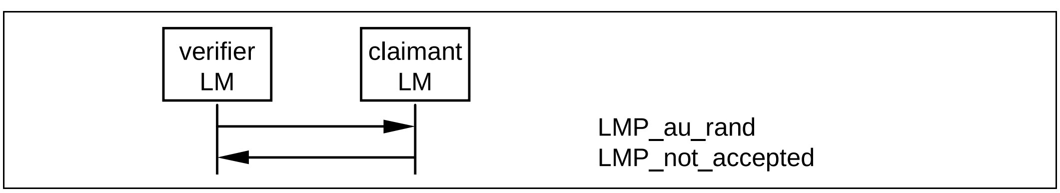
If the claimant has a link key associated with the verifier, it calculates the response and sends it to the verifier with LMP_sres. The verifier checks the response. If the response is not correct, the verifier can end the connection by sending LMP_detach with the reason code *authentication failure,* see Section 3.14 on page 208.



Sequence 1: Authentication. Claimant has link key.

3.2.2 Claimant has no link key

If the claimant does not have a link key associated with the verifier it sends LMP_not_accepted with the reason code *key missing* after receiving LMP_au_rand.



Sequence 2: Authentication fails. Claimant has no link key.

3.2.3 Repeated attempts

If the claimant sends wrong authentication response, a certain waiting interval must pass before the next authentication attempt. The waiting interval is made twice as long after each subsequent failure (up to a maximum value). This prevents an intruder from trying a large number of different keys in a relatively short time. The waiting interval can be set separately for each device so that one intruder does not block all other units from connecting to a particular device (denial of service attacks).

3.3 PAIRING

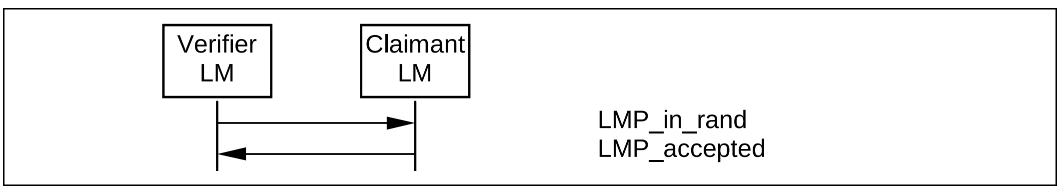
When two devices do not have a common link key an initialisation key (K_{init}) is created based on a PIN and a random number. The K_{init} is created when the verifier sends LMP_in_rand to the claimant. How the K_{init} is calculated is described in Baseband Specification Section 14.5.3, on page 175. Authentication is then performed, but the calculation of the authentication response is based on K_{init} instead of the link key. After a successful authentication, the link key is created. The PDUs used in the pairing procedure are:

M/O	PDU	Contents
M	LMP_in_rand	random number
M	LMP_au_rand	random number
M	LMP_sres	authentication response
M	LMP_comb_key	random number
M	LMP_unit_key	key

Table 3.3: PDUs used for pairing.

3.3.1 Claimant accepts pairing

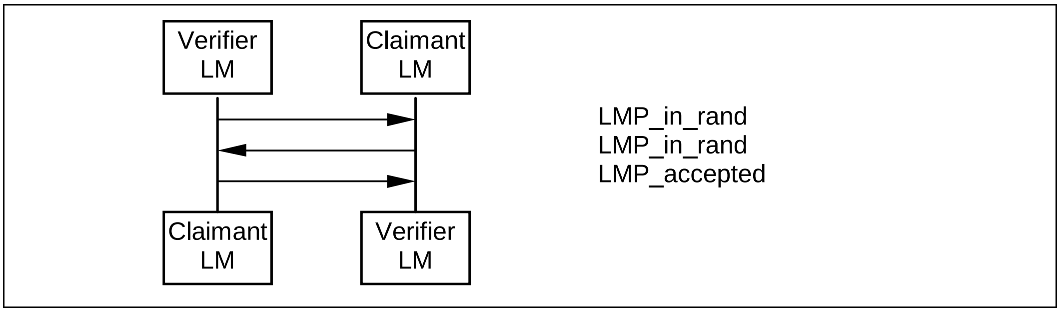
The verifier sends LMP_in_rand and the claimant replies with LMP_accepted. Both devices calculate K_{init} and an authentication, see Sequence 1, is performed based on this key. The verifier checks the authentication response and if correct, the link key is created, see Section 3.3.4 on page 197. If the authentication response is not correct the verifier can end the connection by sending LMP_detach with the reason code *authentication failure*.



Sequence 3: Claimant accepts pairing.

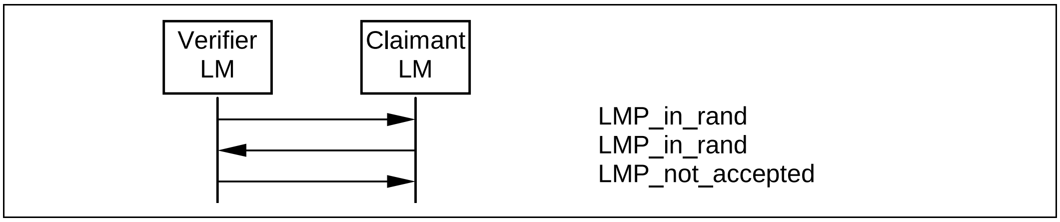
3.3.2 Claimant requests to become verifier

If the claimant has a fixed PIN it may request a switch of the claimant-verifier role in the pairing procedure by generating a new random number and send it back in LMP_in_rand. If the device that started the pairing procedure has a variable PIN it must accept this and respond with LMP_accepted. The roles are then successfully switched and the pairing procedure continues as described in Section 3.3.1 on page 196.



Sequence 4: Claimant accepts pairing but requests to be verifier.

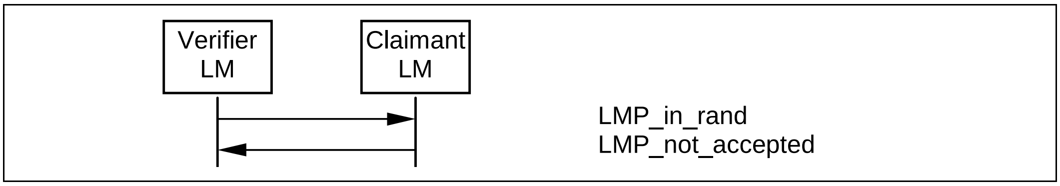
If the device that started the pairing procedure has a fixed PIN and the other device requests to switch roles, the switch is rejected by sending LMP_not_accepted with the reason *pairing not allowed* and the pairing procedure is unsuccessfully ended.



Sequence 5: Unsuccessful switch of claimant-verifier role.

3.3.3 Claimant rejects pairing

If the claimant rejects pairing it sends LMP_not_accepted with the reason code *pairing not allowed* after receiving LMP_in_rand.



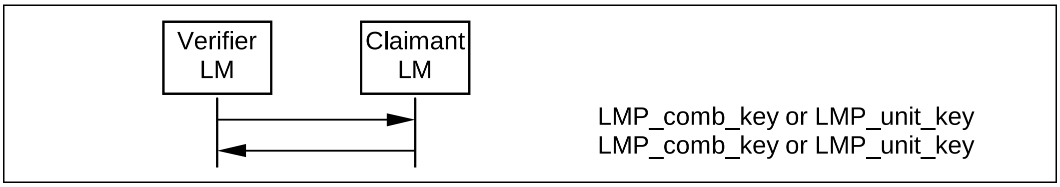
Sequence 6: Claimant rejects pairing.

3.3.4 Creation of the link key

When the authentication is finished the link key must be created. This link key will be used in the authentication between the two units for all subsequent connections. However, a new link key can be created that will be used for the rest of the connection and for all subsequent connections, see Section 3.4. A temporary link key can also be created that will be used as link only for the rest of the connection, see Section 3.5. The link key created in the pairing procedure will either be a combination key or one of the unit's unit keys. The following rules apply to the selection of the link key:

- if one unit sends LMP_unit_key and the other unit sends LMP_comb_key, the unit key will be the link key
- if both units send LMP_unit_key, the master's unit key will be the link key
- if both units send LMP_comb_key, the link key is calculated as described in Baseband Specification Section 14.2.2, on page 153.

The content of LMP_unit_key is the unit key bitwise XORed with K_{init}. The content of LMP_comb_key is LK_RAND bitwise XORed with K_{init}. Any device configured to use a combination key will store the link key in non-volatile memory.



Sequence 7: Creation of the link key.

3.3.5 Repeated attempts

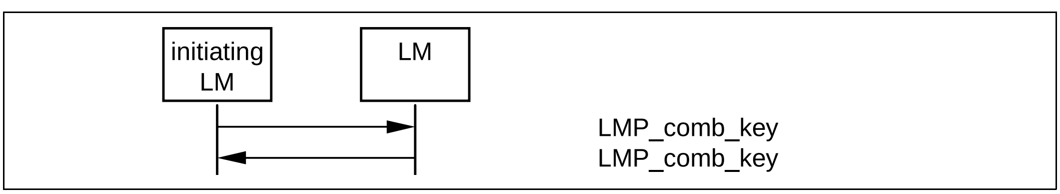
The same scheme as in Section 3.2.3 on page 195 is applied. This prevents an intruder from trying a large number of different PINs in a relatively short time.

3.4 CHANGE LINK KEY

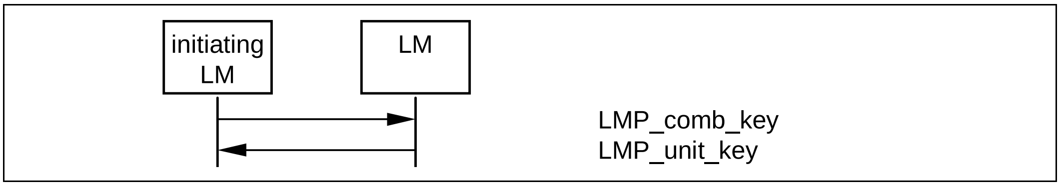
If two devices are paired and the link key is derived from combination keys, the link key can be changed. If the link key is derived from a unit key, the units must go through the pairing procedure in order to change the link key. The procedure for changing link key is the same as in Section 3.3.4 on page 197 except that the current link key instead of K_{init} protects the transferred information.

M/O	PDU	Contents
M	LMP_comb_key	random number
M	LMP_unit_key	key

Table 3.4: PDUs used for change of link key.



Sequence 8: Successful change of the link key.



Sequence 9: Change of the link key not possible since the other unit uses a unit key.

If the change of link key is successful the new link key is stored in non-volatile memory instead of the old one, which is discarded. The new link key will be used as link key for all the following connections between the two devices until the link key is changed again. The new link key also becomes the current link key. It will remain the current link key until the link key is changed again or until a temporary link key is created, see Section 3.5 on page 199.

3.5 CHANGE THE CURRENT LINK KEY

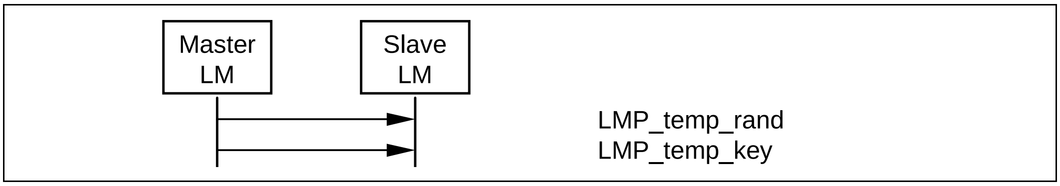
The current link key can be a semi-permanent link key or a temporary link key key. It can be changed temporarily, but the change is only valid for the session, see Baseband Specification Section 14.2.1, on page 151. Changing to a temporary link key is necessary if the piconet shall support encrypted broadcast.

M/O	PDU	Contents
M	LMP_temp_rand	random number
M	LMP_temp_key	key
M	LMP_use_semi_permanent_key	-

Table 3.5: PDUs used to change the current link key.

3.5.1 Change to a temporary link key

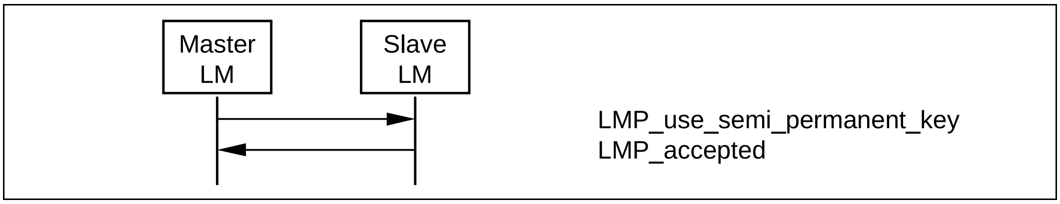
In the following we use the same terms as in Baseband Specification Section 14.2.2.8, on page 158. The master starts by creating the master key K_{master} as described in Baseband Specification (EQ 23), on page 158. Then the master issues a random number RAND and sends it to the slave in LMP_temp_rand. Both sides can then calculate an overlay denoted OVL as OVL= E_{22}(current link key, RAND, 16). Then the master sends K_{master} protected by a modulo-2 addition with OVL to the slave in LMP_temp_key. The slave, who knows OVL, calculates K_{master}. After this, K_{master} becomes the current link key. It will be the current link key until a new temporary key is created or until the link key is changed, see Section 3.4 on page 198.



Sequence 10: Change to a temporary link key.

3.5.2 Make the semi-permanent link key the current link key

After the current link key has been changed to K_{master}, this change can be undone and the semi-permanent link key becomes the current link key again. If encryption is used on the link, the procedure of going back to the semi-permanent link key must be immediately followed by a stop of the encryption by invoking the procedure in Section 3.6.4 on page 203. Encryption can then be started again. This is to assure that encryption with encryption parameters known by other devices in the piconet is not used when the semi-permanent link key is the current link key.



Sequence 11: Link key changed to the semi-permanent link key.

3.6 ENCRYPTION

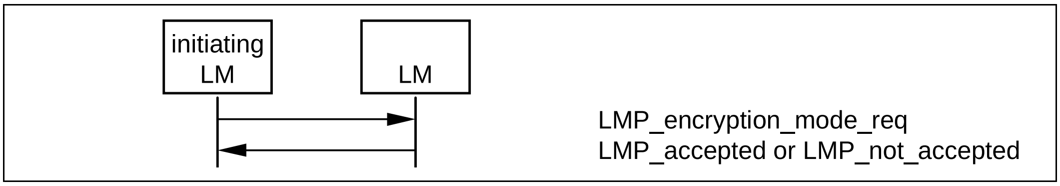
If at least one authentication has been performed encryption may be used. If the master wants all slaves in the piconet to use the same encryption parameters it must issue a temporary key (K_{master}) and make this key the current link key for all slaves in the piconet before encryption is started, see Section 3.5 on page 199. This is necessary if broadcast packets should be encrypted.

M/O	PDU	Contents
O	LMP_encryption_mode_req	encryption mode
O	LMP_encryption_key_size_req	key size
O	LMP_start_encryption_req	random number
O	LMP_stop_encryption_req	-

Table 3.6: PDUs used for handling encryption.

3.6.1 Encryption mode

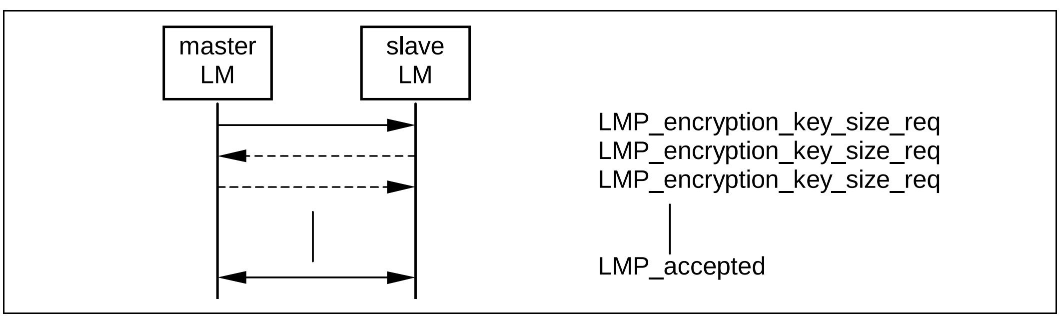
First of all the master and the slave must agree upon whether to use encryption or not and if encryption shall only apply to point to point packets or if encryption shall apply to both point to point packets and broadcast packets. If master and slave agree on the encryption mode, the master continues to give more detailed information about the encryption.



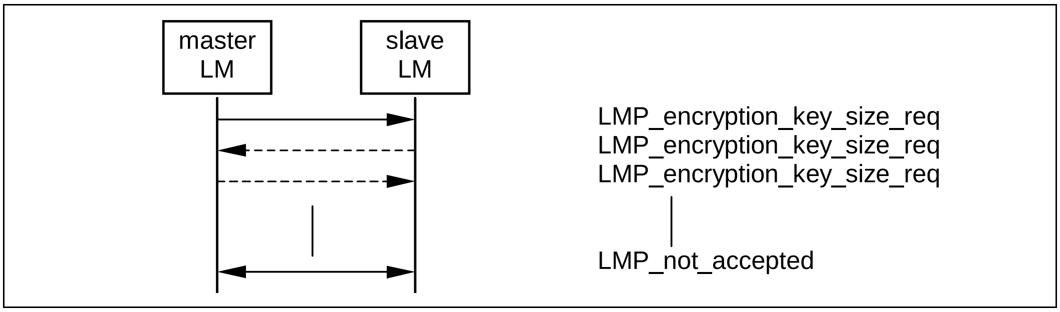
Sequence 12: Negotiation for encryption mode.

3.6.2 Encryption key size

The next step is to determine the size of the encryption key. In the following we use the same terms as in Baseband Specification Section 14.3.1, on page 160. The master sends LMP_encryption_key_size_req including the suggested key size Lsug, m, which is initially equal to $L_{max, m}$. If $L_{min, s} \leq L_{sug, m}$ and the slave supports $L_{sug, m}$ it responds with LMP_accepted and $L_{sug, m}$ will be used as the key size. If both conditions are not fulfilled the slave sends back LMP_encryption_key_size_req including the slave's suggested key size L_{sug}, s. This value is the slave's largest supported key size that is less than $L_{sug, m}$. Then the master performs the corresponding test on the slave suggestion. This procedure is repeated until a key size agreement is reached or if becomes clear that no such agreeement can be reached. If an agreement is reached a unit sends LMP_accepted and the key size in the last LMP_encryption_key_size_req will be used. After this the encryption is started, see Section 3.6.3 on page 202. If an agreement is not reached a unit sends LMP_not_accepted and the units are not allowed to communicate using Bluetooth link encryption.



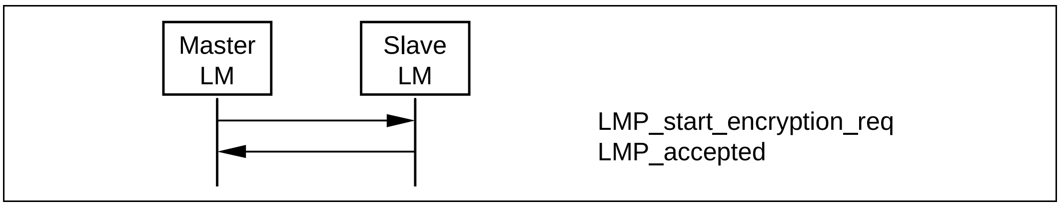
Sequence 13: Encryption key size negotiation successful.



Sequence 14: Encryption key size negotiation failed.

3.6.3 Start encryption

Finally encryption is started. The master issues the random number EN_RAND and calculates the encryption key as K_c=E_3(current link key, EN_RAND, COF). See Baseband Specification Section 14.2.2.5, on page 156 and 14.2.2.2 for the definition of the COF. The random number must be the same for all slaves if the piconet should support encrypted broadcast. Then the master sends LMP_start_encryption_req, which includes EN_RAND. The slave calculates K_c when this message is received and acknowledges with LMP_accepted. On both sides, Kc and EN_RAND are used as input to the encryption algorithm E_0.



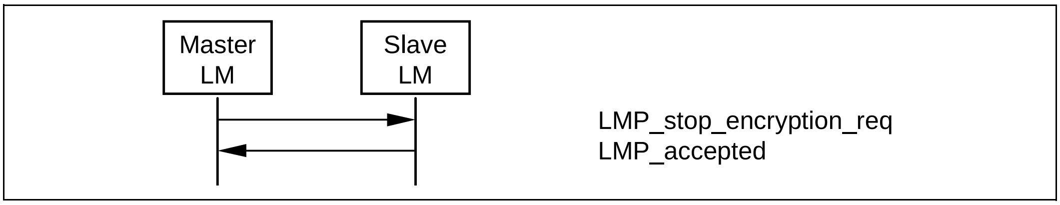
Sequence 15: Start of encryption.

Before starting encryption, higher layer data traffic must be temporarily stopped to prevent reception of corrupt data. The start of encryption will be done in three steps:

1. Master is configured to transmit unencrypted packets, but to receive encrypted packets.
2. Slave is configured to transmit and receive encrypted packets.
3. Master is configured to transmit and receive encrypted packets.

Between step1 and step2 master to slave transmission is possible. This is when LMP_start_encryption_req is transmitted. Step2 is triggered when the slave receives this message. Between step2 and step3 slave to master transmission is possible. This is when LMP_accepted is transmitted. Step3 is triggered when the master receives this message.

3.6.4 Stop encryption



Sequence 16: Stop of encryption.

Before stopping encryption, higher layer data traffic must be temporarily stopped to prevent reception of corrupt data. Stopping of encryption is then done in three steps, similar to the procedure for starting encryption.

1. Master is configured to transmit encrypted packets, but to receive unencrypted packets.
2. Slave is configured to transmit and receive unencrypted packets.
3. Master is configured to transmit and receive unencrypted packets.

Between step1 and step2 master to slave transmission is possible. This is when LMP_stop_encryption_req is transmitted. Step2 is triggered when the slave receives this message. Between step2 and step3 slave to master transmission is possible. This is when LMP_accepted is transmitted. Step3 is triggered when the master receives this message

3.6.5 Change encryption mode, key or random number

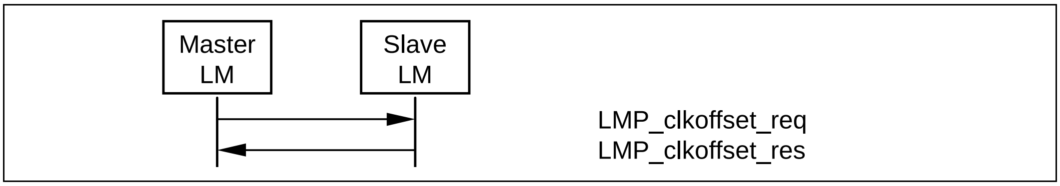
If the encryption mode, encryption key or encryption random number need to be changed, encryption must first be stopped and then re-started with the new parameters.

3.7 CLOCK OFFSET REQUEST

When a slave receives the FHS packet the difference between its own clock and the master's clock included in the payload of the FHS packet is computed. The clock offset is also updated each time a packet is received from the master. The master can request this clock offset anytime during the connection. By saving this clock offset the master knows when and on what channel the slave wakes up to PAGE SCAN after it has left the piconet. This can be used to speed up the paging time the next time the same device is paged.

M/O	PDU	Contents
M	LMP_clkoffset_req	-
M	LMP_clkoffset_res	clock offset

Table 3.7: PDUs used for clock offset request.



Sequence 17: Clock offset requested.

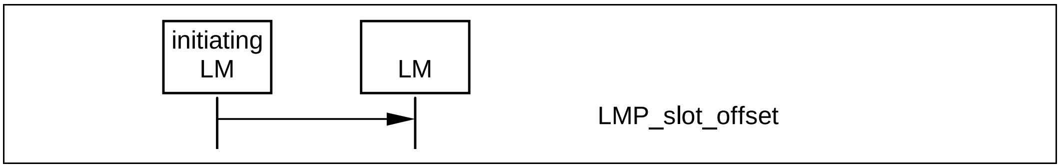
3.8 SLOT OFFSET INFORMATION

With LMP_slot_offset the information about the difference between the slot boundaries in different piconets is transmitted. This PDU carries the parameters slot offset and BD_ADDR. The slot offset is the time in μs between the start of the master's TX slot in the piconet where the PDU is transmitted and the start of the master's TX slot in the piconet where the BD_ADDR device is master.

Before doing a master-slave switch, see Section 3.12 on page 207, this PDU should be transmitted from the device that becomes master in the switch procedure. The PDU can also be useful in inter-piconet communications.

M/O	PDU	Contents
O	LMP_slot_offset	slot offset BD_ADDR

Table 3.8: PDU used for slot offset information.



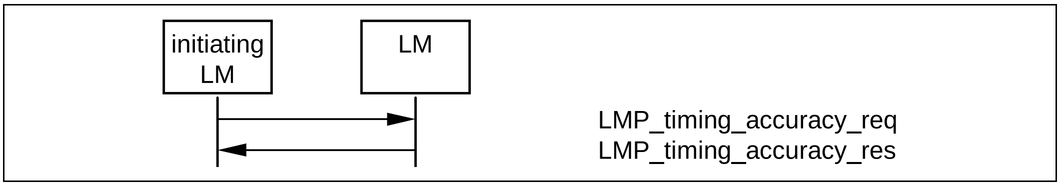
Sequence 18: Slot offset information is sent.

3.9 TIMING ACCURACY INFORMATION REQUEST

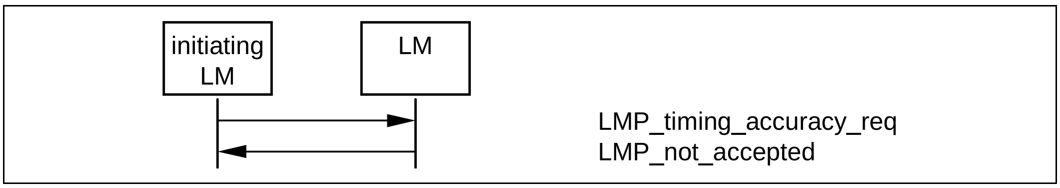
LMP supports requests for the timing accuracy. This information can be used to minimise the scan window for a given hold time when returning from hold and to extend the maximum hold time. It can also be used to minimise the scan window when scanning for park mode beacon packets. The timing accuracy parameters returned are the long term drift measured in ppm and the long term jitter measured in μs of the clock used during hold and park mode. These parameters are fixed for a certain device and must be identical when requested several times. If a device does not support the timing accuracy information it sends LMP_not_accepted with the reason code *unsupported LMP feature* when the request is received. The requesting device must in this case assume worst case values (drift=250ppm and jitter=10μs) .

M/O	PDU	Contents
O	LMP_timing_accuracy_req	-
O	LMP_timing_accuracy_res	drift jitter

Table 3.9: PDUs used for requesting timing accuracy information.



Sequence 19: The requested device supports timing accuracy information.



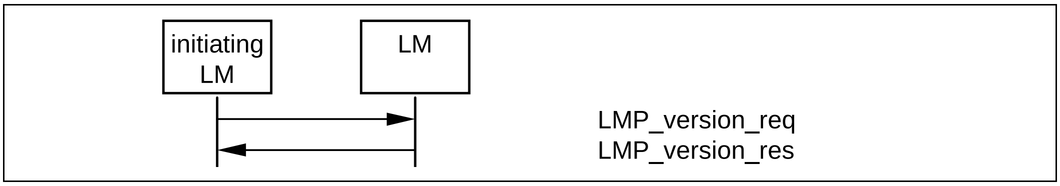
Sequence 20: The requested device does not support timing accuracy information.

3.10 LMP VERSION

LMP has support for requesting the version of the LM protocol. The requested device will send a response with three parameters: VersNr, CompId and Sub-VersNr. VersNr specifies the version of the Bluetooth LMP specification that the device supports. CompId is used to track possible problems with the lower Bluetooth layers. All companies that create a unique implementation of the Link Manager shall have their own CompId. The same company is also responsible for the administration and maintenance of the SubVersNr. It is recommended that each company has a unique SubVersNr for each RF/BB/LM implementation. For a given VersNr and CompId, the values of the SubVersNr must increase each time a new implementaion is released. For both CompId and SubVersNr the value 0xFFFF means that no valid number applies. There is no ability to negotiate the version of the LMP. The sequence below is only used to exchange the parameters.

M/O	PDU	Contents
M	LMP_version_req	VersNr CompId SubVersNr
M	LMP_version_res	VersNr CompId SubVersNr

Table 3.10: PDUs used for LMP version request.



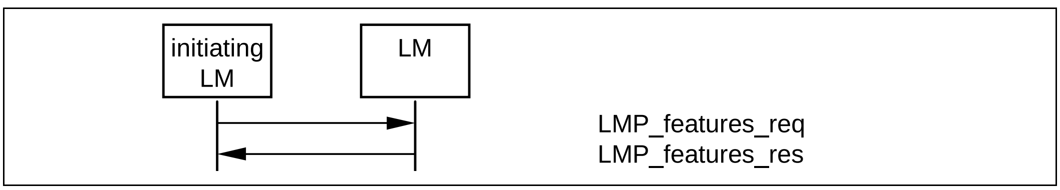
Sequence 21: Request for LMP version.

3.11 SUPPORTED FEATURES

The Bluetooth radio and link controller may support only a subset of the packet types and features described in Baseband Specification and Radio Specification. The PDU LMP_features_req and LMP_features_res are used to exchange this information. A device may not send any packets other than ID, FHS, NULL, POLL, DM1 or DH1 before it is aware of the supported features of the other device. After the features request has been carried out, the intersection of the supported packet types for both sides may also be transmitted. Whenever a request is issued, it must be compatible with the supported features of the other device. For instance when establishing an SCO link, the initiator may not propose to use HV3 packets if that packet type is not supported by the other device. An exception to this rule is LMP_switch_req, which can be sent as the first LMP message when two Bluetooth devices have been connected and thus before the requesting side is aware of the other side's features (switch is an optional feature).

M/O	PDU	Contents
M	LMP_features_req	features
M	LMP_features_res	features

Table 3.11: PDUs used for features request.



Sequence 22: Request for supported features.

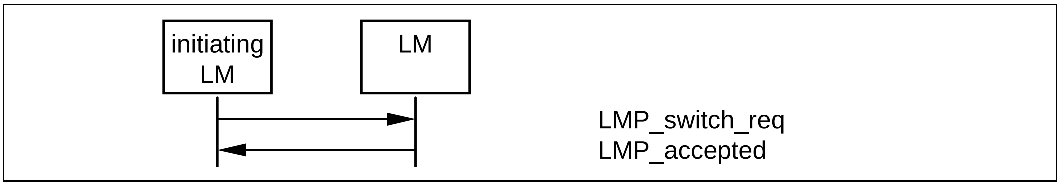
3.12 SWITCH OF MASTER SLAVE ROLE

Since the paging device always becomes the master of the piconet, a switch of the master-slave role is sometimes needed, see Baseband Specification Section 10.9.3, on page 123. Suppose device A is slave and device B is master. The device that initiates the switch sends LMP_switch_req. The other device responds with LMP_accepted if it accepts to perform the switch. After this, both devices do the TDD switch and A is the master of the piconet. Device A then sends the FHS packet to B and after FHS acknowledgement both A and B switch to the channel parameters as indicated by the FHS packet.

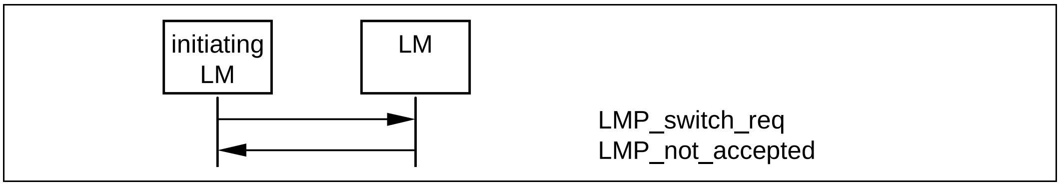
If a device that receives LMP_switch_req does not accept to switch master-slave role, it responds with LMP_not_accepted and no switch is performed.

M/O	PDU	Contents
O	LMP_switch_req	-

Table 3.12: PDU used for master slave switch.



Sequence 23: Master-slave switch accepted.



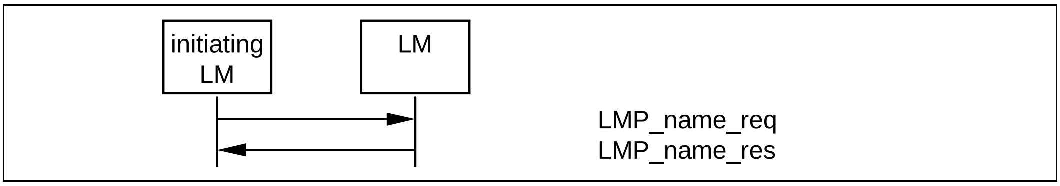
Sequence 24: Master-slave switch not accepted.

3.13 NAME REQUEST

LMP supports name request to another Bluetooth device. The name is a user-friendly name associated with the Bluetooth device and consists of a maximum of 248 bytes coded according to the UTF-8 standard. The name is fragmented over one or more DM1 packets. When the LMP_name_req is sent a name offset indicates which fragment is expected. The corresponding LMP_name_res carries the same name offset, the name length indicating the total number of bytes in the name of the Bluetooth device and the name fragment, where name fragment (N) = name (N + name offset) if N + name offset < name length and name fragment (N) = 0 otherwise. Here 0<=N<=13. In the first sent LMP_name_req name offset=0. Sequence 25 is repeated until the initiator has collected all fragments of the name.

M/O	PDU	Contents
M	LMP_name_req	name offset
M	LMP_name_res	name offset name length name fragment

Table 3.13: PDUs used for name request.



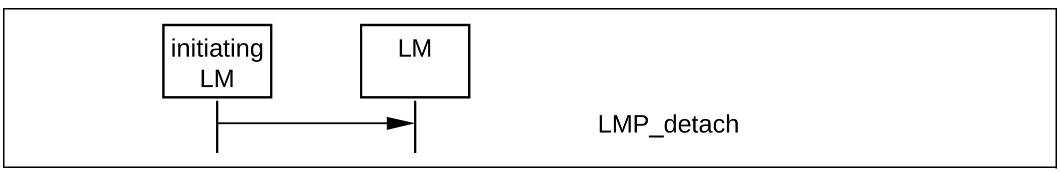
Sequence 25: Device's name requested and it responses.

3.14 DETACH

The connection between two Bluetooth devices can be closed anytime by the master or the slave. A reason parameter is included in the message to inform the other party of why the connection is closed.

M/O	PDU	Contents
M	LMP_detach	reason

Table 3.14: PDU used for detach.



Sequence 26: Connection closed by sending LMP_detach.

3.15 HOLD MODE

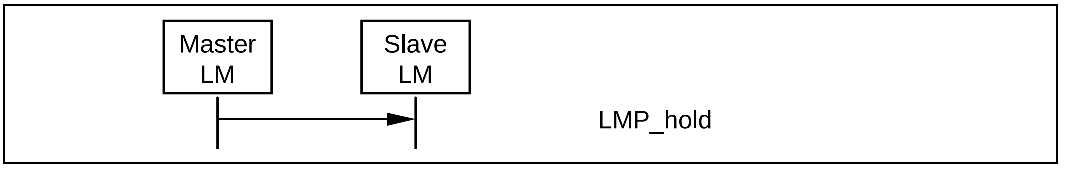
The ACL link of a connection between two Bluetooth devices can be placed in hold mode for a specified hold time. During this time no ACL packets will be transmitted from the master. The hold mode is typically entered when there is no need to send data for a relatively long time. The transceiver can then be turned off in order to save power. But the hold mode can also be used if a device wants to discover or be discovered by other Bluetooth devices, or wants to join other piconets. What a device actually does during the hold time is not controlled by the hold message, but it is up to each device to decide.

M/O	PDU	Contents
O	LMP_hold	hold time
O	LMP_hold_req	hold time

Table 3.15: PDUs used for hold mode.

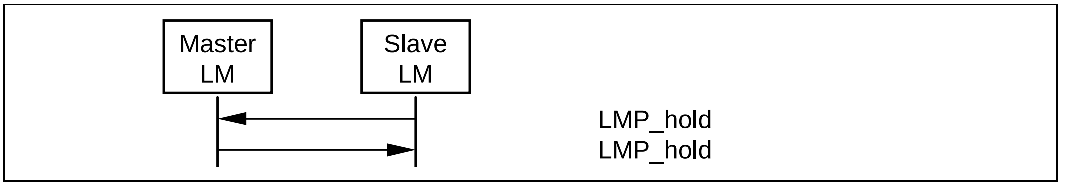
3.15.1 Master forces hold mode

The master can force hold mode if there has previously been a request for hold mode that has been accepted. The hold time included in the PDU when the master forces hold mode cannot be longer than any hold time the slave has previously accepted when there was a request for hold mode.



Sequence 27: Master forces slave into hold mode.

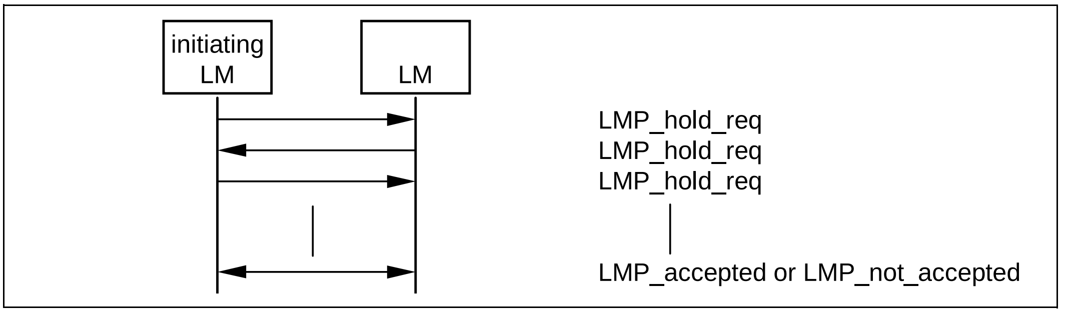
3.15.2 Slave forces hold mode



Sequence 28: Slave forces master into hold mode.

3.15.3 Master or slave requests hold mode

The master or the slave can request to enter hold mode. Upon receipt of the request, the same request with modified parameters can be returned or the negotiation can be terminated. If an agreement is seen LMP_accepted terminates the negotiation and the ACL link is placed in hold mode. If no agreement is seen, LMP_not_accepted terminates the negotiation and hold mode is not entered.



Sequence 29: Negotiation for hold mode.

3.16 SNIFF MODE

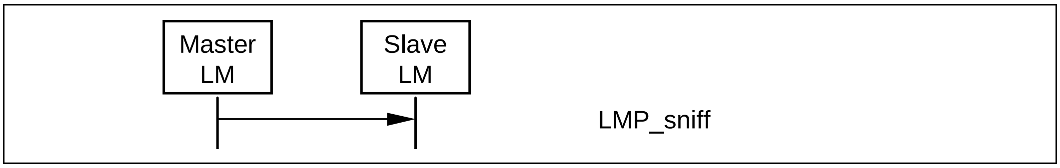
To enter sniff mode, master and slave negotiate a sniff interval T_{sniff} and a sniff offset, D_{sniff}, which specifies the timing of the sniff slots. The offset determines the time of the first sniff slot; after that the sniff slots follows periodically with the sniff interval T_{sniff}. To avoid problems with a clock wrap around during the initialisation one of two options is chosen for the calculation of the first sniff slot. A timing control flag in the message from the master indicates this. Note: Only bit1 of this field is valid.

When the link is in sniff mode the master can only start a transmission in the sniff slot. Two parameters control the listening activity in the slave. The sniff attempt parameter determines for how many slots the slave must listen, with beginning at the sniff slot. The sniff timeout parameter determine for how many additional slots the slave must listen as long as it receives packets with its own AM address.

M/O	PDU	Contents
O	LMP_sniff	timing control flags D_{sniff} T_{sniff} sniff attempt sniff timeout
O	LMP_sniff_req	timing control flags D_{sniff} T_{sniff} sniff attempt sniff timeout
O	LMP_unsniff_req	-

Table 3.16: PDUs used for sniff mode.

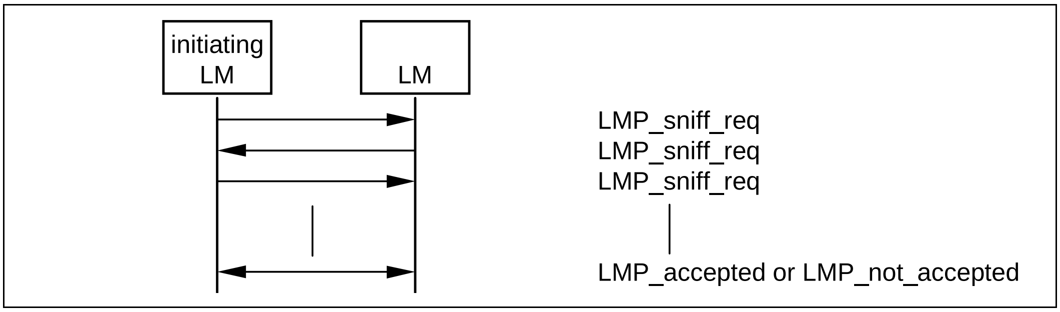
3.16.1 Master forces a slave into sniff mode



Sequence 30: Master forces slave into sniff mode.

3.16.2 Master or slave requests sniff mode

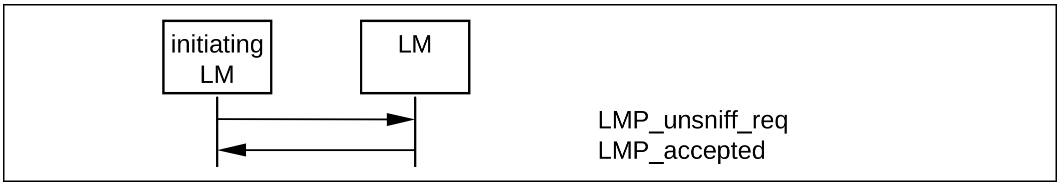
The master or the slave can request to enter sniff mode. Upon receipt of the request, the same request with modified parameters can be returned or the negotiation can be terminated. If an agreement is seen LMP_accepted terminates the negotiation and the ACL link is placed in sniff mode. If no agreement is seen, LMP_not_accepted terminates the negotiation and sniff mode is not entered.



Sequence 31: Negotiation for sniff mode. The double-arrowed line means that either the initiating LM or the other LM sends the message.

3.16.3 Moving a slave from sniff mode to active mode

Sniff mode is ended by sending the PDU LMP_unsniff_req. The requested device must reply with LMP_accepted. If the slave requests it will enter active mode after receiving LMP_accepted. If the master requests, the slave will enter active mode after receiving LMP_unsniff_req.



Sequence 32: Slave moved from sniff mode to active mode.

3.17 PARK MODE

If a slave does not need to participate in the channel, but still should be FH synchronised it can be placed in park mode. In this mode the device gives up its AM_ADDR but still re-synchronises to the channel by waking up at the beacon instants separated by the beacon interval. The beacon interval, a beacon offset and a flag indicating how the first beacon instant is calculated determine the first beacon instant. After this the beacon instants follow periodically with the beacon interval. At the beacon instant the parked slave can be activated again by the master, the master can change the park mode parameters, transmit broadcast information or let the parked slaves request access to the channel.

All messages sent from the master to the parked slaves are broadcast and to increase reliability for broadcast, the packets are made as short as possible. Therefore the format for these LMP messages are somewhat different. The parameters are not always byte-aligned and the length of the messages is variable.

The messages for controlling the park mode include many parameters, which are all defined in Baseband Specification Section 10.8.4, on page 115. When a slave is placed in park mode it is assigned a unique PM_ADDR, which can be used by the master to unpark that slave. The all-zero PM_ADDR has a special meaning; it is not a valid PM_ADDR. If a device is assigned this PM_ADDR, it must be identified with its BD_ADDR when it is unparked by the master.

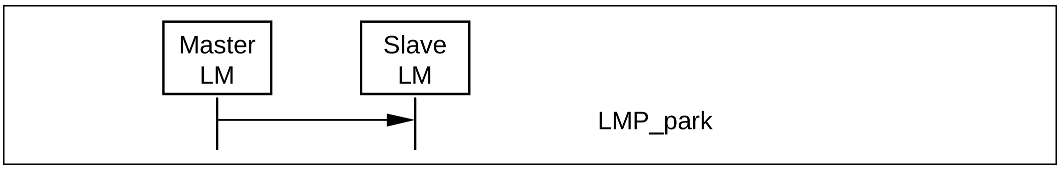
M/O	PDU	Contents
O	LMP_park_req	-
O	LMP_park	timing control flags D_B T_B N_B Δ_B PM_ADDR AR_ADDR N_{Bsleep} D_{Bsleep} D_{access} T_{access} $N_{acc\text{-}slots}$ N_{poll} M_{access} access scheme
O	LMP_set_broadcast_scan_window	timing control flags D_B (optional) broadcast scan window
O	LMP_modify_beacon	timing control flags D_B (optional) T_B N_B Δ_B D_{access} T_{access} $N_{acc\text{-}slots}$ N_{poll} M_{access} access scheme
O	LMP_unpark_PM_ADDR_req	timing control flags D_B (optional) AM_ADDR PM_ADDR AM_ADDR (optional) PM_ADDR (optional) (totally 1-7 pairs of AM_ADDR, PM_ADDR)

Table 3.17: PDUs used for park mode.

M/O	PDU	Contents
O	LMP_unpark_BD_ADDR _req	timing control flags D_B (optional) AM_ADDR BD_ADDR AM_ADDR (optional) BD_ADDR (optional)

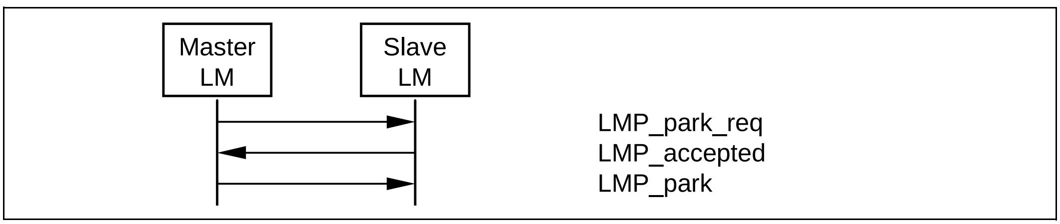
Table 3.17: PDUs used for park mode.

3.17.1 Master forces a slave into park mode



Sequence 33: Slave forced into park mode.

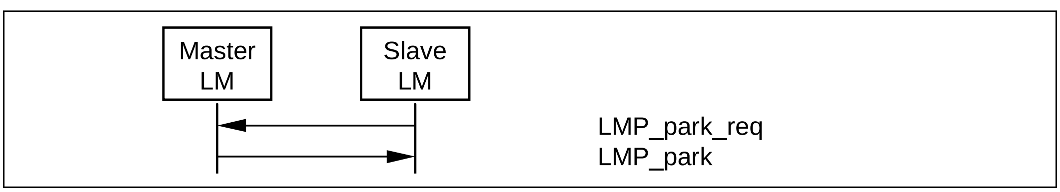
3.17.2 Master requests slave to enter park mode



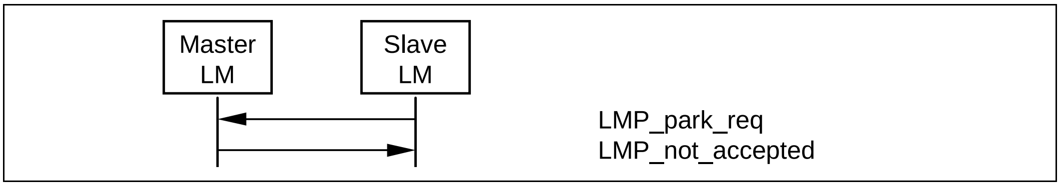
Sequence 34: Slave accepts to be placed in park mode.

Sequence 35: Slave rejects to be placed in park mode.

3.17.3 Slave requests to be placed in park mode



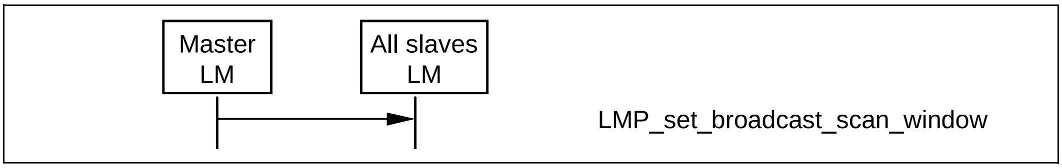
Sequence 36: Master accepts and places slave in park mode.



Sequence 37: Master rejects to place slave in park mode.

3.17.4 Master sets up broadcast scan window

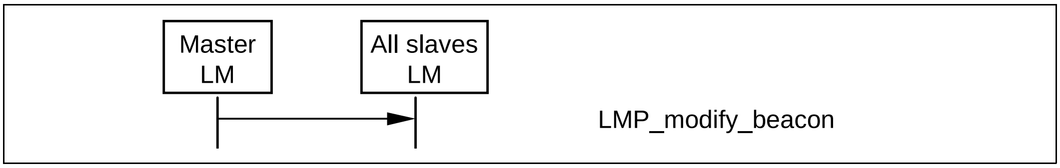
If more broadcast capacity is needed than the beacon train, the master can indicate to the slaves that more broadcast information will follow the beacon train by sending LMP_set_broadcast_scan_window. This message is always sent in a broadcast packet at the beacon slot(s). The scan window starts in the beacon instant and is only valid for the current beacon.



Sequence 38: Master notifies all slaves of increase in broadcast capacity.

3.17.5 Master modifies beacon parameters

When the beacon parameters change the master notifies the parked slaves of this by sending LMP_modify_beacon. This message is always sent in a broadcast packet at the beacon slot(s).

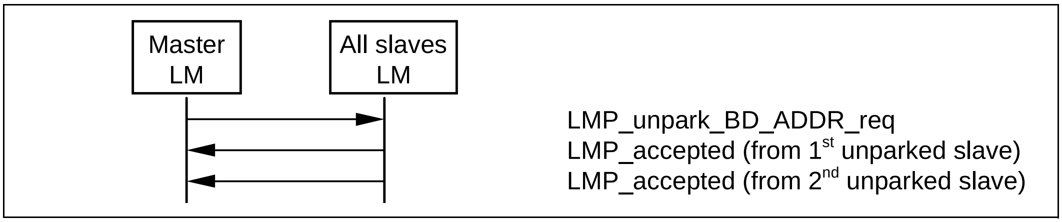


Sequence 39: Master modifies beacon parameters.

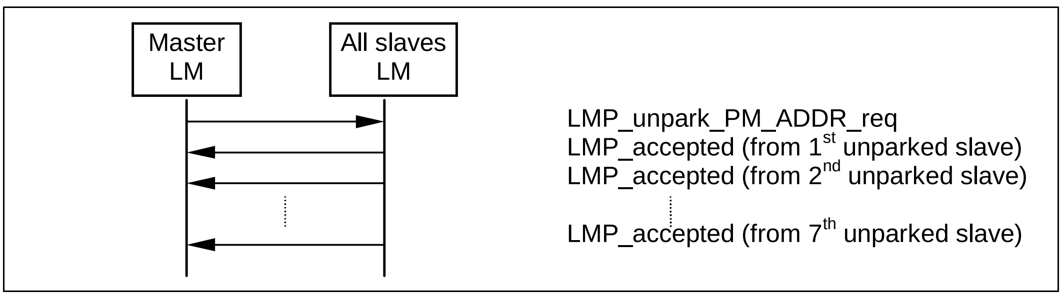
3.17.6 Unparking slaves

The master can unpark one or many slaves by sending a broadcast LMP message including the PM_ADDR or the BD_ADDR of the device(s) it wishes to unpark at the beacon slot(s). This message also includes the AM_ADDR that the master assigns to the slave(s). After sending this message, the master must check the success of the unpark by polling each unparked slave, i.e. sending POLL or NULL packets, so that the slave is granted access to the channel. The unparked slave must then send a response with LMP_accepted. If this message is not received from the slave within a certain time after the master sent the unpark message, the unpark failed and the master must consider the slave as still being in park mode.

One message is used where the parked device is identified with the PM_ADDR, and another message is used where it is identified with the BD_ADDR. Both messages have variable length depending on the number of slaves the master unparks. For each slave the master wishes to unpark an AM_ADDR followed by the PM/BD_ADDR of the device that is assigned this AM_ADDR is included in the payload. If the slaves are identified with the PM_ADDR a maximum of 7 slaves can be unparked with the same message. If they are identified with the BD_ADDR a maximum of 2 slaves can be unparked with the same message.



Sequence 40: Master unparks slaves addressed with their BD_ADDR.



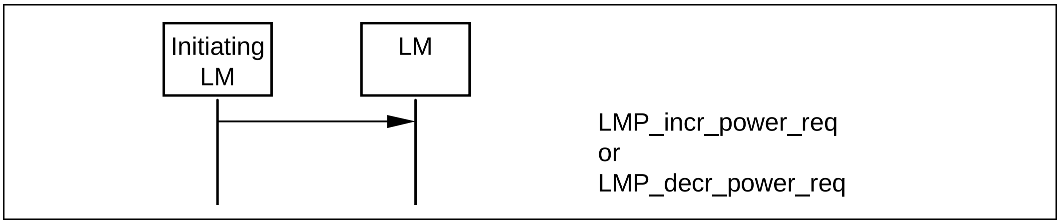
Sequence 41: Master unparks slaves addressed with their PM_ADDR.

3.18 POWER CONTROL

If the RSSI value differs too much from the preferred value of a Bluetooth device, it can request an increase or a decrease of the other device's TX power. Upon receipt of this message, the output power is increased or decreased one step, see Radio Specification Section 3.1, on page 21. At the master side the TX power is completely independent for different slaves; a request from one slave can only effect the master's TX power for that same slave.

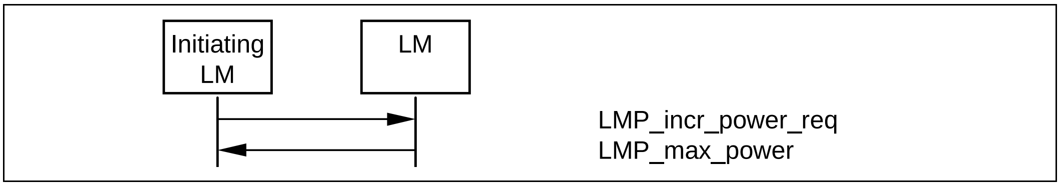
M/O	PDU	Contents
O	LMP_incr_power_req	TBD (1 Byte)
O	LMP_decr_power_req	TBD (1 Byte)
O	LMP_max_power	-
O	LMP_min_power	-

Table 3.18: PDUs used for power control.

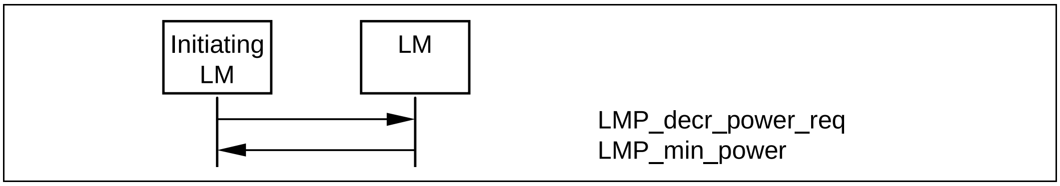


Sequence 42: A device requests a change of the other device's TX power.

If the receiver of LMP_incr_power_req already transmits at maximum power LMP_max_power is returned. The device may then only request for an increase again after having requested for a decrease at least once. Similarly, if the receiver of LMP_decr_power_req already transmits at minimum power LMP_min_power is returned and the device may only request for a decrease again after having requested for an increase at least once.



Sequence 43: The TX power cannot be increased.



Sequence 44: The TX power cannot be decreased.

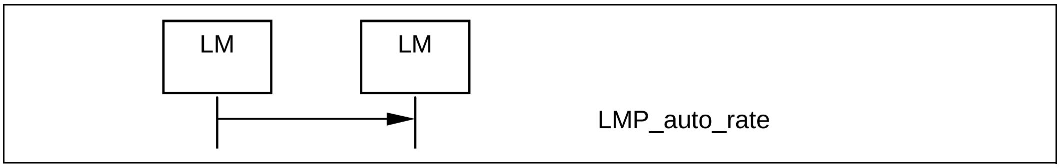
One byte is reserved in LMP_incr/decr_power_req for future use. It could be the mismatch between preferred and measured RSSI. The receiver of LMP_incr/decr_power_req can perhaps use this value to adjust to the correct power at once, instead of only changing it one step for each request. The parameter value must be 0x00 for all versions of LMP where this parameter is not yet defined.

3.19 CHANNEL QUALITY DRIVEN CHANGE BETWEEN DM AND DH

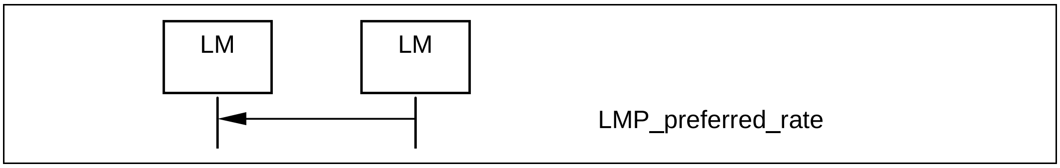
A device is configured to always use DM packets or to always use DH packets or to automatically adjust its packet type according to the quality of the channel. Nevertheless, all devices are capable of transmitting either DM or DH packets. The difference between DM and DH is that the payload in a DM packet is protected with a 2/3 FEC code, whereas the payload of a DH is not protected with any FEC. If a device wants to automatically adjust between DM and DH it sends LMP_auto_rate to the other device. Based upon quality measures in LC, the device determines if throughput will be increased by a change of packet type. If so, LMP_preferred_rate is sent to the other device. The PDUs used for this are:

M/O	PDU	Contents
O	LMP_auto_rate	-
O	LMP_preferred_rate	data rate

Table 3.19: PDUs used for quality driven change of the data rate.



Sequence 45: The left-hand unit is configured to automatically change between DM and DH.



Sequence 46: The right-hand device orders the left-hand device to change data rate.

3.20 QUALITY OF SERVICE (QOS)

The Link Manager provides Quality of Service capabilities that support bandwidth allocation and latency control. The Link Manager, of the device that is the master of the piconet, uses a polling list to determine when slaves are queried and the time interval between consecutive queries to each slave.

Two PDUs provide the means to modify the polling list of the master. The Link Manager of the master of the piconet uses the maximum poll interval for a connection to adjust the polling list. The master and the slave for a connection determine the maximum poll interval. By adjusting the polling list the master can control the bandwidth allocated for each ACL connection and the latency between consecutive queries to the slave. The maximum poll interval is the maximum tolerable time interval between consecutive master polls that a slave accepts for the connection. When the maximum poll interval is determined, the master guaranties that the slave will be polled every *N* baseband slots.

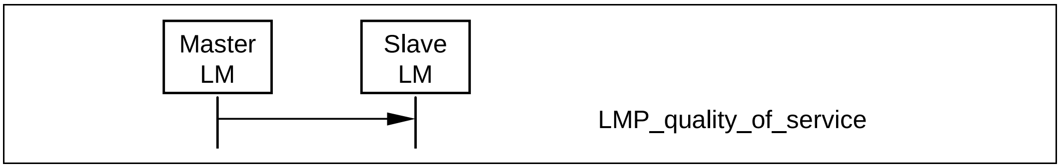
In addition, the master notifies the slave of the number of repetitions for broadcast packets (N_{BC}), see Baseband Specification Section 5.3, on page 68.

M/O	PDU	Contents
M	LMP_quality_of_service	poll interval N_{BC}
M	LMP_quality_of_service_req	poll interval N_{BC}

Table 3.20:   PDUs used for quality of service.

3.20.1 Master notifies slave of the quality of service

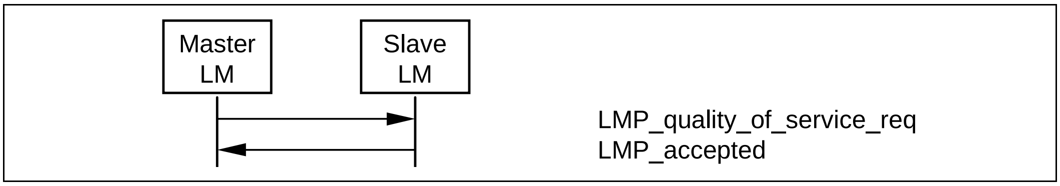
In this case the master notifies the slave of the new maximum poll interval. The slave cannot reject the notification.



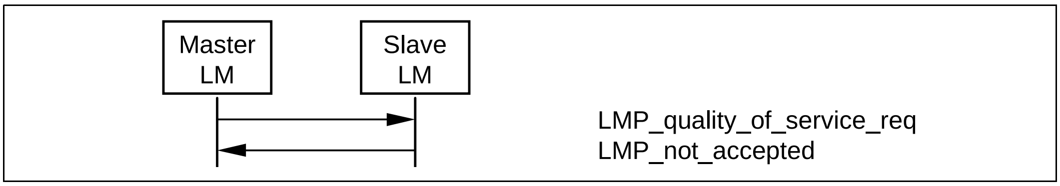
Sequence 47: Master notifies slave of new maximum poll interval.

3.20.2 Master requests slave for a new quality of service

In this case the master requests a new maximum poll interval for a particular slave. The slave can either accept or reject the requested maximum poll interval. This will allow the master and slave to dynamically renegotiate the maximum poll interval as needed.



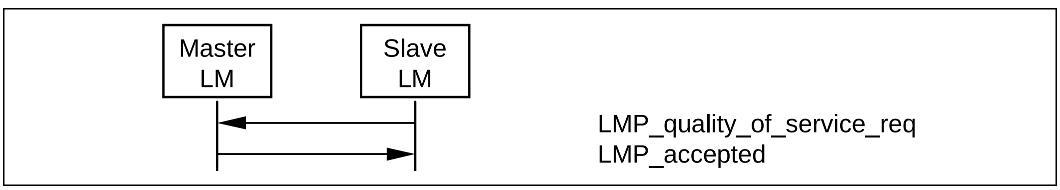
Sequence 48: Slave accepts new maximum poll interval.



Sequence 49: Slave rejects new maximum poll interval.

3.20.3 Slave requests master for a new maximum poll interval

In this case the slave requests the master for a new maximum poll interval. The master can either accept or reject the request maximum poll interval. This will allow the master and slave to dynamically renegotiate the maximum poll interval as needed.



Sequence 50: Master accepts new maximum poll interval.

Sequence 51: Master rejects new maximum poll interval.

3.21 SCO LINKS

When a connection has been established between two Bluetooth devices the connection consists of an ACL link. One or more SCO links can then be established. The SCO link reserves slots separated by the SCO interval, T_{sco}. The first slot reserved for the SCO link is defined by T_{sco} and the SCO delay, D_{sco}. After that the SCO slots follows periodically with the SCO interval. To avoid problems with a wrap around of the clock during initialisation of the SCO link, a flag indicating how the first SCO slot should be calculated is included in a message from the master. Note: Only bit0 and bit1 of this field is valid. Each SCO link is distinguished from all other SCO links by an SCO handle. The SCO handle zero is never used.

M/O	PDU	Contents
O	LMP_SCO_link_req	SCO handle timing control flags D_{sco} T_{sco} SCO packet air mode
O	LMP_remove_SCO_link_req	SCO handle reason

Table 3.21: PDUs used for managing the SCO links.

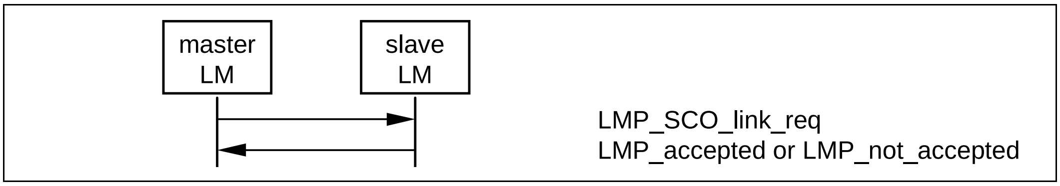
3.21.1 Master initiates an SCO link

When establishing an SCO link the master sends a request with parameters that specify the timing, packet type and coding that will be used on the SCO link. For each of the SCO packets Bluetooth supports three different voice coding formats on the air-interface: μ-law log PCM, A-law log PCM and CVSD.

The slots used for the SCO links are determined by three parameters controlled by the master: T_{sco}, D_{sco} and a flag indicating how the first SCO slot should be calculated. After the first slot, the SCO slots follows periodically with the T_{sco}.

If the slave does not accept the SCO link, it can indicate what it does not accept in the error reason field of LMP_not_accepted. The master then has the possibility to issue a new request with modified parameters.

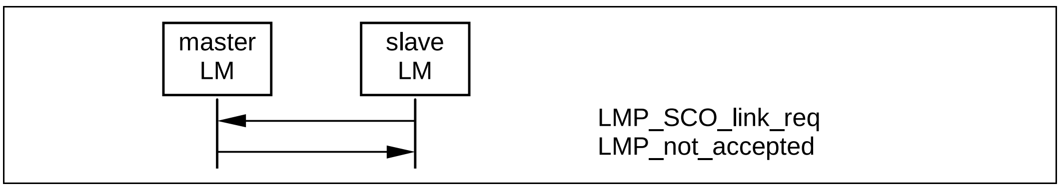
The SCO handle in the message must be different from any already existing SCO link(s).



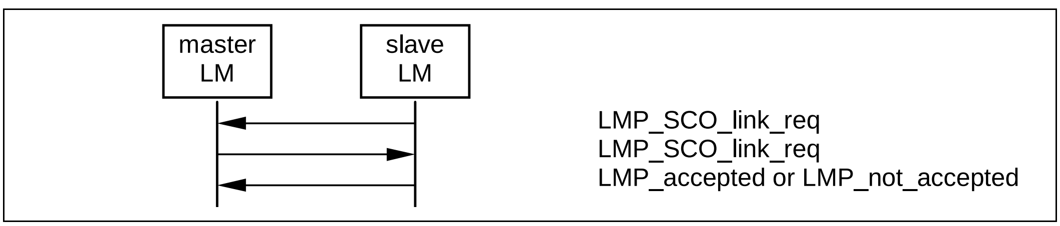
Sequence 52: Master requests an SCO link.

3.21.2 Slave initiates an SCO link

The slave can also initiate the establishment of an SCO link. The slave sends LMP_SCO_link_req, but the parameters: timing control flags and D_{sco} are invalid as well as the SCO handle, which must be zero. If the master is not capable of establishing an SCO link it replies with LMP_not_accepted. Otherwise it sends back LMP_SCO_link_req. This message includes the assigned SCO handle, D_{sco} and the timing control flags. For the other parameters the master should try to use the same parameters as in the slave request, but if the master cannot meet that request it is allowed to use other values. The slave must then reply with LMP_accepted or LMP_not_accepted.



Sequence 53: Master rejects slave's request for an SCO link.



Sequence 54: Master accepts slave's request for an SCO link.

3.21.3 Master requests change of SCO parameters

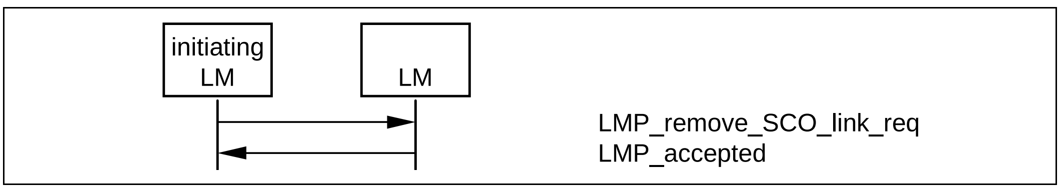
The master sends LMP_SCO_link_req, where the SCO handle is the handle of the SCO link the master wishes to change parameters for. If the slave accepts the new parameters it replies with LMP_accepted and the SCO link will change to the new parameters. If the slave does not accept the new parameters it replies with LMP_not_accepted and the SCO link is left unchanged. When the slave replies with LMP_not_accepted it shall indicate in the error reason parameter what it does not accept. The master can then try to change the SCO link again with modified parameters. The sequence is the same as in Section 3.21.1 on page 221.

3.21.4 Slave requests change of SCO parameters

The slave sends LMP_SCO_link_req, where the SCO handle is the handle of the SCO link the slave wishes to change parameters for. The parameters timing control flags and D_{sco} are not valid in this message. If the master does not accept the new parameters it replies with LMP_not_accepted and the SCO link is left unchanged. If the master accepts the new parameters it replies with LMP_SCO_link_req where it must use the same parameters as in the slave request. When receiving this message the slave replies with LMP_not_accepted if it does not accept the new parameters. The SCO link is then left unchanged. If the slave accepts the new parameters it replies with LMP_accepted and the SCO link will change to the new parameters. The sequence is the same as in Section 3.21.2 on page 222.

3.21.5 Remove an SCO link

Master or slave can remove the SCO link by sending a request including the SCO handle of the SCO link to be removed and a reason indicating why the SCO link is removed. The receiving party must respond with LMP_accepted.



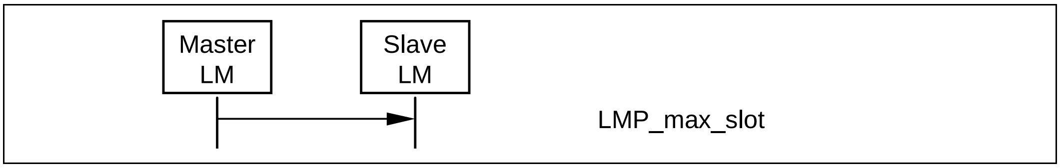
Sequence 55: SCO link removed.

3.22 CONTROL OF MULTI-SLOT PACKETS

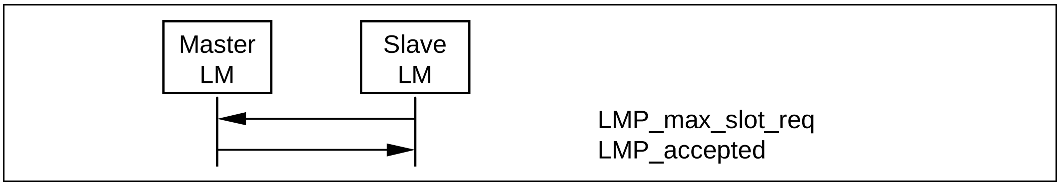
The number of slots used by a slave in its return packet can be limited. The master can force each slave to use a maximal number of slots by sending the PDU LMP_max_slots providing max slots as parameter. Each slave can request to use a maximal number of slots by sending the PDU LMP_max_slot_req providing max slots as parameter. The default value is 1 slot, i.e. if the slave has not been informed about the number of slots, it may only use 1-slot packets. Two PDUs are used for the control of multi-slot packets.

M/O	PDU	Contents
M	LMP_max_slot	max slots
M	LMP_max_slot_req	max slots

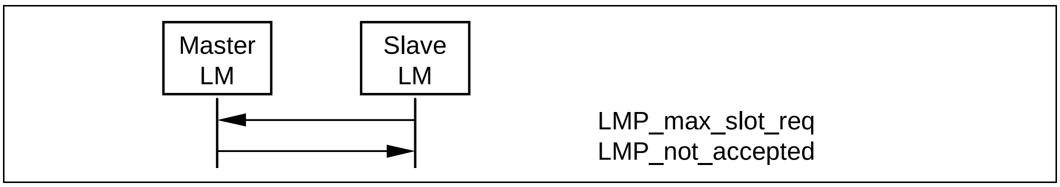
Table 3.22: PDUs used to control the use of multi-slot packets.



Sequence 56: Master allows slave to use a maximal number of slots.



Sequence 57: Slave requests to use a maximal number of slots. Master accepts.



Sequence 58: Slave requests to use a maximal number of slots. Master rejects.

3.23 PAGING SCHEME

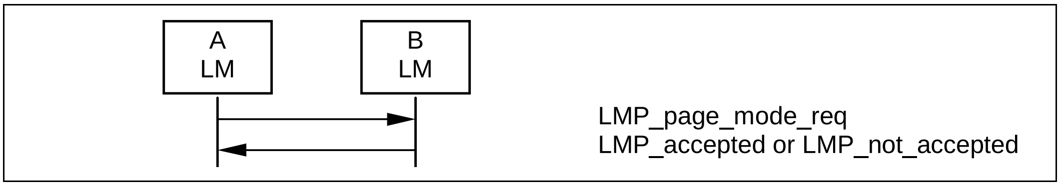
In addition to the mandatory paging scheme Bluetooth defines optional paging schemes, see Appendix VII, on page 983. LMP provides a means to negotiate the paging scheme to be used the next time a unit is paged.

M/O	PDU	Contents
O	LMP_page_mode_req	paging scheme paging scheme settings
O	LMP_page_scan_mode_req	paging scheme paging scheme settings

Table 3.23: PDUs used to request paging scheme.

3.23.1 Page mode

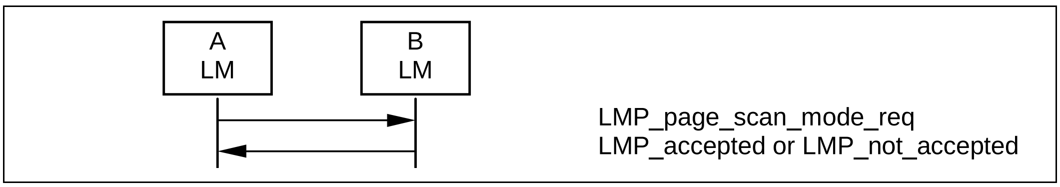
This procedure is initiated from device A and negotiates the paging scheme used when device A pages device B. Device A proposes a paging scheme including the parameters for this scheme and device B can accept or reject. On rejection the old setting is not changed. A request to switch back to the mandatory scheme may be rejected.



Sequence 59: Negotiation for page mode.

3.23.2 Page scan mode

This procedure is initiated from device A and negotiates the paging scheme used when device B pages device A. Device A proposes a paging scheme including the parameters for this scheme and device B can accept or reject. On rejection the old setting is not changed. A request to switch to the mandatory scheme must be accepted.



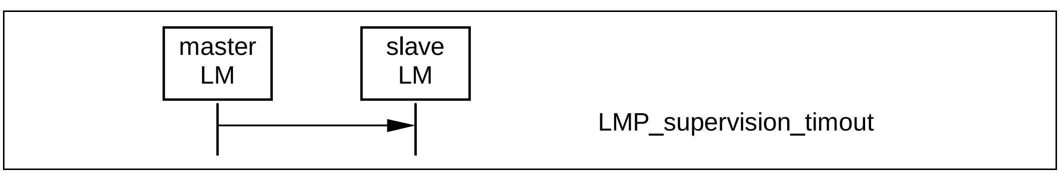
Sequence 60: Negotiation for page scan mode

3.24 LINK SUPERVISION

Each Bluetooth link has a timer that is used for link supervision. This timer is used to detect link loss caused by devices moving out of range, power down of a device or other similar failure cases. The scheme for link supervision is described in Baseband Specification Section 10.11, on page 126. An LMP procedure is used to set the value of the supervision timeout.

M/O	PDU	Contents
M	LMP_supervision_timeout	supervision timeout

Table 3.24: PDU used to set the supervision timeout.



Sequence 61: Setting the link supervision timeout.

4 CONNECTION ESTABLISHMENT

After the paging procedure, the master must poll the slave by sending POLL or NULL packets, with a max poll interval as defined in Table 5.5 on page 237. LMP procedures that do not require any interactions between the LM and the host at the paged unit's side can then be carried out.

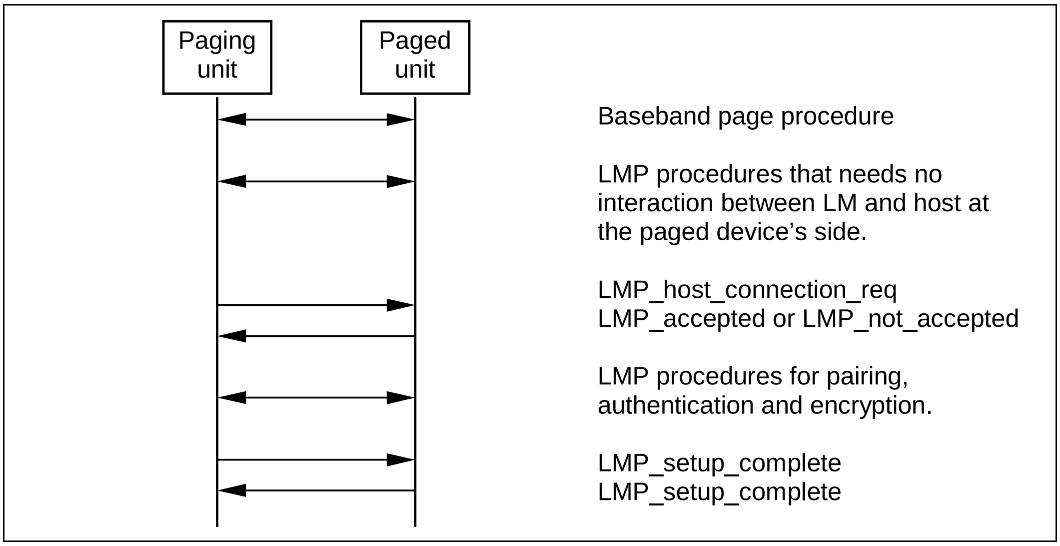


Figure 4.1: Connection establishment.

When the paging device wishes to create a connection involving layers above LM, it sends LMP_host_connection_req. When the other side receives this message the host is informed about the incoming connection. The remote device can accept or reject the connection request by sending LMP_accepted or LMP_not_accepted.

When a device does not require any further link set-up procedures it will send LMP_setup_complete. The device will still respond to requests from the other device. When also that one is ready with link set-up it will send LMP_setup_complete. After this the first packet on a logical channel different from LMP can be transmitted.

M/O	PDU	Contents
M	LMP_host_connection_req	-
M	LMP_setup_complete	-

Table 4.1: PDUs used for connection establishment.

5 SUMMARY OF PDUS

LMP PDU	Length (bytes)	op code	Packet type	Possible direction	Contents	Position in payload
LMP_accepted	2	3	DM1/ DV	m ↔ s	op code	2
LMP_au_rand	17	11	DM1	m ↔ s	random number	2-17
LMP_auto_rate	1	35	DM1/ DV	m ↔ s	-	
LMP_clkoffset_req	1	5	DM1/ DV	m → s	-	
LMP_clkoffset_res	3	6	DM1/ DV	m ← s	clock offset	2–3
LMP_comb_key	17	9	DM1	m ↔ s	random number	2-17
LMP_decr_power_req	2	32	DM1/ DV	m ↔ s	TBD	2
LMP_detach	2	7	DM1/ DV	m ↔ s	reason	2
LMP_encryption_key_size _req	2	16	DM1/ DV	m ↔ s	key size	2
LMP_encryption_mode_ req	2	15	DM1/ DV	m ↔ s	encryption mode	2
LMP_features_req	9	39	DM1/ DV	m ↔ s	features	2-9
LMP_features_res	9	40	DM1/ DV	m ↔ s	features	2-9
LMP_host_connection_req	1	51	DM1/ DV	m ↔ s	-	
LMP_hold	3	20	DM1/ DV	m ↔ s	hold time	2-3
LMP_hold_req	3	21	DM1/ DV	m ↔ s	hold time	2-3
LMP_incr_power_req	2	31	DM1/ DV	m ↔ s	TBD	2
LMP_in_rand	17	8	DM1	m ↔ s	random number	2-17
LMP_max_power	1	33	DM1/ DV	m ↔ s	-	

Table 5.1: Coding of the different LM PDUs.

LMP PDU	**Length (bytes)**	**op code**	**Packet type**	**Possible direction**	**Contents**	**Position in payload**
LMP_max_slot	2	45	DM1/ DV	m $\rightarrow$ s	max slots	2
LMP_max_slot_req	2	46	DM1/ DV	m $\leftarrow$ s	max slots	2
LMP_min_power	1	34	DM1/ DV	m $\leftrightarrow$ s	-	
LMP_modify_beacon	11 or 13	28	DM1	m $\rightarrow$ s	timing control flags	2
					D_B	3-4
					T_B	5-6
					N_B	7
					Δ_B	8
					D_{access}	9
					T_{access}	10
					$N_{acc\text{-}slots}$	11
					N_{poll}	12
					M_{access}	13:0-3
					access scheme	13:4-7
LMP_name_req	2	1	DM1/ DV	m $\leftrightarrow$ s	name offset	2
LMP_name_res	17	2	DM1	m $\leftrightarrow$ s	name offset	2
					name length	3
					name fragment	4-17
LMP_not_accepted	3	4	DM1/ DV	m $\leftrightarrow$ s	op code	2
					reason	3
LMP_page_mode_req	3	53	DM1/ DV	m $\leftrightarrow$ s	paging scheme	2
					paging scheme settings	3
LMP_page_scan_mode_req	3	54	DM1/ DV	m $\leftrightarrow$ s	paging scheme	2
					paging scheme settings	3

Table 5.1: Coding of the different LM PDUs.

LMP PDU	Length (bytes)	op code	Packet type	Possible direction	Contents	Position in payload
LMP_park	17	26	DM	$m \rightarrow s$	timing control flags	2
					D_B	3-4
					T_B	5-6
					N_B	7
					Δ_B	8
					PM_ADDR	9
					AR_ADDR	10
					N_{Bsleep}	11
					D_{Bsleep}	12
					D_{access}	13
					T_{access}	14
					$N_{acc-slots}$	15
					N_{poll}	16
					M_{access}	17:0-3
					access scheme	17:4-7
LMP_park_req	1	25	DM1/ DV	$m \leftrightarrow s$	-	
LMP_preferred_rate	2	36	DM1/ DV	$m \leftrightarrow s$	data rate	2
LMP_quality_of_service	4	41	DM1/ DV	$m \rightarrow s$	poll interval	2-3
					N_{BC}	4
LMP_quality_of_service_req	4	42	DM1/ DV	$m \leftrightarrow s$	poll interval	2-3
					N_{BC}	4
LMP_remove_SCO_link_req	3	44	DM1/ DV	$m \leftrightarrow s$	SCO handle	2
					reason	3

Table 5.1: Coding of the different LM PDUs.

LMP PDU	**Length (bytes)**	**op code**	**Packet type**	**Possible direction**	**Contents**	**Position in payload**
LMP_SCO_link_req	7	43	DM1/ DV	m ↔ s	SCO handle	2
					timing control flags	3
					D_{sco}	4
					T_{sco}	5
					SCO packet	6
					air mode	7
LMP_set_broadcast_ scan_window	4 or 6	27	DM1	m → s	timing control flags	2
					D_B	3-4
					broadcast scan window	5-6
LMP_setup_complete	1	49	DM1/ DV	m ↔ s	-	
LMP_slot_offset	9	52	DM1/ DV	m ↔ s	slot offset	2-3
					BD_ADDR	4-9
LMP_sniff	10	22	DM1	m → s	timing control flags	2
					D_{sniff}	3-4
					T_{sniff}	5-6
					sniff attempt	7-8
					sniff timeout	9-10
LMP_sniff_req	10	23	DM1	m ↔ s	timing control flags	2
					D_{sniff}	3-4
					T_{sniff}	5-6
					sniff attempt	7-8
					sniff timeout	9-10
LMP_sres	5	12	DM1/ DV	m ↔ s	authentication response	2-5
LMP_start_encryption_req	17	17	DM1	m → s	random number	2-17
LMP_stop_encryption_req	1	18	DM1/ DV	m → s	-	
LMP_supervision_timout	3	55	DM1/ DV	m ↔ s	supervision timout	2-3
LMP_switch_req	1	19	DM1/ DV	m ↔ s	-	

Table 5.1: Coding of the different LM PDUs.

LMP PDU	Length (bytes)	op code	Packet type	Possible direction	Contents	Position in payload
LMP_temp_rand	17	13	DM1	m → s	random number	2-17
LMP_temp_key	17	14	DM1	m → s	key	2-17
LMP_timing_accuracy_req	1	47	DM1/ DV	m ↔ s	-	
LMP_timing_accuracy_res	3	48	DM1/ DV	m ↔ s	drift	2
					jitter	3
LMP_unit_key	17	10	DM1	m ↔ s	key	2-17
LMP_unpark_BD_ADDR _req	variable	29	DM1	m → s	timing control flags	2
					D_B	3-4
					AM_ADDR 1st unpark	5:0-3
					AM_ADDR 2nd unpark	5:4-7
					BD_ADDR 1st unpark	6-11
					BD_ADDR 2nd unpark	12-17
LMP_unpark_PM_ADDR _req	variable	30	DM1	m → s	timing control flags	2
					D_B	3-4
					AM_ADDR 1st unpark	5:0-3
					AM_ADDR 2nd unpark	5:4-7
					PM_ADDR 1st unpark	6
					PM_ADDR 2nd unpark	7
LMP_unsniff_req	1	24	DM1/ DV	m ↔ s	-	
LMP_use_semi_ permanent_key	1	50	DM1/ DV	m → s	-	
LMP_version_req	6	37	DM1/ DV	m ↔ s	VersNr	2
					CompId	3-4
					SubVersNr	5-6
LMP_version_res	6	38	DM1/ DV	m ↔ s	VersNr	2
					CompId	3-4
					SubVersNr	5-6

Table 5.1: Coding of the different LM PDUs.

Note1: For LMP_set_broadcast_scan_window, LMP_modify_beacon, LMP_unpark_BD_ADDR_req and LMP_unpark_PM_ADDR_req the parameter

D_B is optional. This parameter is only present if bit0 of *timing control flags* is 0. If the parameter is not included, the position in payload for all parameters following D_B are decreased by 2.

Note2: For LMP_unpark_BD_ADDR the AM_ADDR and the BD_ADDR of the 2^{nd} unparked slave are optional. If only one slave is unparked *AM_ADDR* 2^{nd} *unpark* should be zero and *BD_ADDR* 2^{nd} *unpark* is left out.

Note3: For LMP_unpark_PM_ADDR the AM_ADDR and the PM_ADDR of the 2^{nd} – 7^{th} unparked slaves are optional. If N slaves are unparked, the fields up to and including the N^{th} unparked slave are present. If N is odd, the *AM_ADDR* $(N+1)^{th}$ *unpark* must be zero. The length of the message is x + 3N/2 if N is even and x + 3(N+1)/2 -1 if N is odd, where x = 2 or 4 depending on if the D_B is included or not (see Note1).

5.1 DESCRIPTION OF PARAMETERS

Name	Length (bytes)	Type	Unit	Detailed
access scheme	1	u_int4		0: polling technique 1-15: Reserved
air mode	1	u_int8		0: µ-law log 1: A-law log 2: CVSD 3-255: Reserved
AM_ADDR	1	u_int4		
AR_ADDR	1	u_int8		
authentication response	4	multiple bytes		
BD_ADDR	6	multiple bytes		
broadcast scan window	2	u_int16	slots	
clock offset	2	u_int16	1.25ms	($CLKN_{16-2}$ slave - $CLKN_{16-2}$ master) mod 2^{15} MSbit of second byte not used.
CompId	2	u_int16		see BT Assigned Numbers Section 2.1 on page 1002
D_{access}	1	u_int8	slots	
D_B	2	u_int16	slots	

Table 5.2: Parameters in LM PDUs.

Name	Length (bytes)	Type	Unit	Detailed
D_{Bsleep}	1	u_int8	slots	
data rate	1	u_int8		0: medium rate 1: high rate 2-255: Reserved
drift	1	u_int8	ppm	
D_{sco}	1	u_int8	slots	
D_{sniff}	2	u_int16	slots	
encryption mode	1	u_int8		0: no encryption 1: point to point encryption 2: point to point and broadcast encryption 3 -255: Reserved
features	8	multiple bytes		See Table 5.3 on page 235
hold time	2	u_int16	slots	
jitter	1	u_int8	µs	
key	16	multiple bytes		
key size	1	u_int8	byte	
M_{access}	1	u_int4	slots	
max slots	1	u_int8	slots	
$N_{acc-slots}$	1	u_int8	slots	
name fragment	14	multiple bytes		UTF-8 characters.
name length	1	u_int8	bytes	
name offset	1	u_int8	bytes	
N_B	1	u_int8		
N_{BC}	1	u_int8		
N_{Bsleep}	1	u_int8	slots	
N_{poll}	1	u_int8	slots	
op code	1	u_int8		
paging scheme	1	u_int8		0: mandatory scheme 1: optional scheme 1 2-255: Reserved

Table 5.2: Parameters in LM PDUs.

Name	Length (bytes)	Type	Unit	Detailed
paging scheme settings	1	u_int8		For mandatory scheme: 0: R0 1: R1 2: R2 3-255: Reserved For optional scheme 1: 0: Reserved 1: R1 2: R2 3-255: Reserved
PM_ADDR	1	u_int8		
poll interval	2	u_int16	slots	
random number	16	multiple bytes		
reason	1	u_int8		See Table 5.4 on page 236.
SCO handle	1	u_int8		
SCO packet	1	u_int8		0: HV1 1: HV2 2: HV3 3: DV 4-255: Reserved
slot offset	2	u_int16	µs	$0 \leq$ slot offset < 1250
sniff attempt	2	u_int16	slots	
sniff timeout	2	u_int16	slots	
SubVersNr	2	u_int16		Defined by each company
supervision time-out	2	u_int16	slots	
T_{access}	1	u_int8	slots	
T_B	2	u_int16	slots	

Table 5.2: Parameters in LM PDUs.

Name	Length (bytes)	Type	Unit	Detailed
timing control flags	1	u_int8		bit0 = 0: no timing change bit0 = 1: timing change bit1 = 0: use initialisation 1 bit1 = 1: use initialisation 2 bit2 = 0: access window bit2 = 1: no access window bit3-7: Reserved
T_{sco}	1	u_int8	slots	
T_{sniff}	2	u_int16	slots	
VersNr	1	u_int8		0: Bluetooth LMP 1.0 1-255: Reserved
Δ_B	1	u_int8	slots	

Table 5.2: Parameters in LM PDUs.

5.1.1 Coding of features

This parameter is a bitmap with information about which Bluetooth radio, baseband and LMP features a device supports. The bit is set if the feature is supported. The bits of the feature parameter that are not defined in Table 5.3 must be zero.

Byte	Bit	Supported feature
0	0	3-slot packets
	1	5-slot packets
	2	encryption
	3	slot offset
	4	timing accuracy
	5	switch
	6	hold mode
	7	sniff mode

Table 5.3: Coding of the parameter features.

1	0	park mode
	1	RSSI
	2	channel quality driven data rate
	3	SCO link
	4	HV2 packets
	5	HV3 packets
	6	u-law log
	7	A-law log
2	0	CVSD
	1	paging scheme
	2	power control

Table 5.3: Coding of the parameter features.

5.1.2 List of error reasons

The following table contains the codes of the different error reasons used in LMP.

Reason	Description
0x05	Authentication Failure
0x06	Key Missing
0x0A	Max Number Of SCO Connections To A Device (The maximum number of SCO connections to a particle device has been reached. All allowed SCO connection handles to that device are used.)
0x0D	Host Rejected due to limited resources (The host at the remote side has rejected the connection because the remote host did not have enough additional resources to accept the connection.)
0x0E	Host Rejected due to security reasons (The host at the remote side has rejected the connection because the remote host determined that the local host did not meet its security criteria.)
0x0F	Host Rejected due to remote device is only a personal device (The host at the remote side has rejected the connection because the remote host is a personal device and will only accept the connection from one particle remote host.)
0x10	Host Timeout (Used at connection accept timeout, the host did not respond to an incoming connection attempt before the connection accept timer expired.)
0x13	Other End Terminated Connection: User Ended Connection
0x14	Other End Terminated Connection: Low Resources
0x15	Other End Terminated Connection: About to Power Off

Table 5.4: List of error reasons.

Reason	Description
0x16	Connection Terminated by Local Host
0x17	Repeated Attempts (An authentication or pairing attempt is made too soon after a previously failed authentication or pairing attempt.)
0x18	Pairing Not Allowed
0x19	Unknown LMP PDU
0x1A	Unsupported LMP Feature
0x1B	SCO Offset Rejected
0x1C	SCO Interval Rejected
0x1D	SCO Air Mode Rejected
0x1E	Invalid LMP Parameters
0x1F	Unspecified Error
0x20	Unsupported parameter value

Table 5.4: List of error reasons.

5.2 DEFAULT VALUES

The Bluetooth device must use these values before anything else has been negotiated:

Parameter	Value
drift	250
jitter	10
max slots	1
poll interval	40

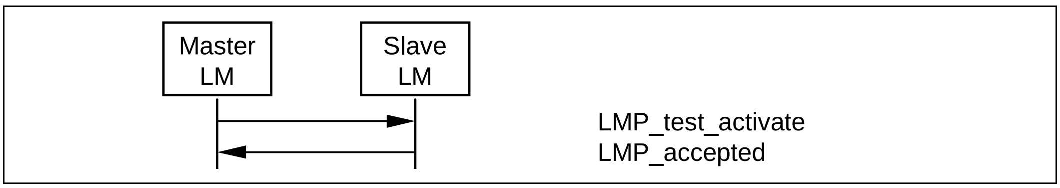
Table 5.5: Default values.

6 TEST MODES

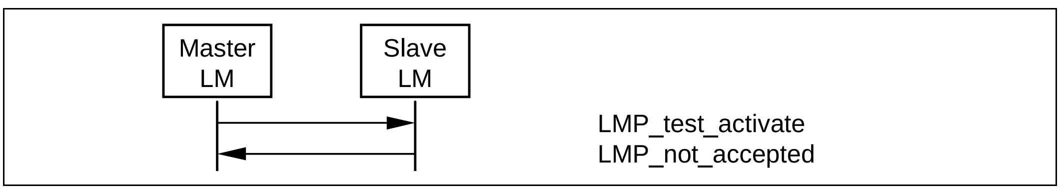
LMP has PDUs to support different Bluetooth test modes, which are used for certification and compliance testing of the Bluetooth radio and baseband. See Bluetooth Test Mode, on page 789 for a detailed description of these test modes.

6.1 ACTIVATION AND DEACTIVATION OF TEST MODE

The test mode is activated by sending LMP_test_activate to the device under test (DUT). The DUT is always the slave. The link manager must be able to receive this message anytime. If entering test mode is locally enabled in the DUT it responds with LMP_accepted and test mode is entered. Otherwise the DUT responds with LMP_not_accepted and the DUT remains in normal operation.



Sequence 62: Activation of test mode successful.

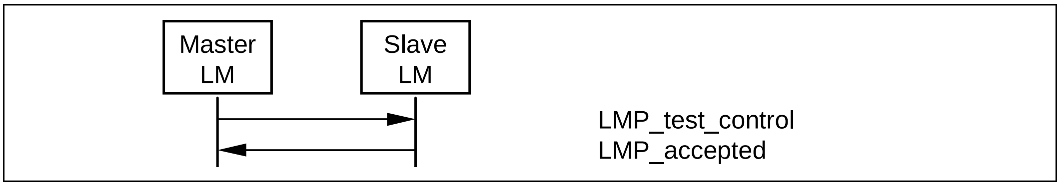


Sequence 63: Activation of test mode fails. Slave is not allowed to enter test mode.

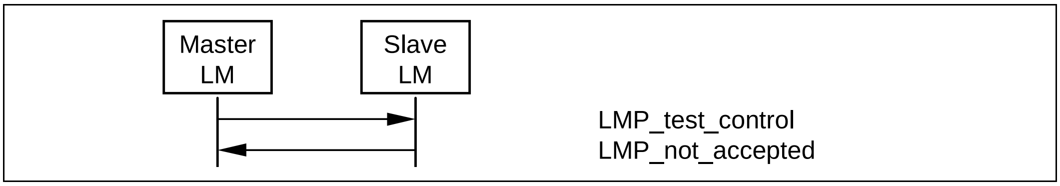
The test mode can be deactivated in two ways. Sending LMP_test_control with the test scenarion set to "exit test mode" exits the test mode and the slave returns to normal operation still connected to the master. Sending LMP_detach to the DUT ends the test mode and the connection.

6.2 CONTROL OF TEST MODE

When the DUT has entered test mode, the PDU LMP_test_control can be sent to the DUT to start a specific test. This PDU is acknowledged with LMP_accepted. If a device that is not in test mode receives LMP_test_control it responds with LMP_not_accepted.



Sequence 64: Control of test mode successful.



Sequence 65: Control of test mode rejected since slave is not in test mode.

6.3 SUMMARY OF TEST MODE PDUS

The PDUs used for test purposes are summarized in the following table. For a detailed description of the parameters, see Bluetooth Test Mode Table 3.2 on page 803.

LMP PDU	Length	op code	Packet type	Possible direction	Contents	Position in payload
LMP_test_activate	1	56	DM1/ DV	$m \rightarrow s$	-	
LMP_test_control	10	57	DM1	$m \rightarrow s$	test scenario	2
					hopping mode	3
					TX frequency	4
					RX frequency	5
					power control mode	6
					poll period	7
					packet type	8
					length of test data	9-10

Table 6.1: Test mode PDUs.

7 ERROR HANDLING

If the Link Manager receives a PDU with unrecognised OpCode, it responds with LMP_not_accepted with the reason code *unknown LMP PDU.* The op code parameter that is echoed back is the unrecognised OpCode.

If the Link Manager receives a PDU with invalid parameters, it responds with LMP_not_accepted with the reason code *invalid LMP parameters*.

If the maximum response time, see Section 1 on page 191, is exceeded or if a link loss is detected, see Baseband Specification Section 10.11, on page 126, the party that waits for the response shall conclude that the procedure has terminated unsuccessfully.

Erroneous LMP messages can be caused by errors on the channel or systematic errors at the transmit side. To detect the latter case the LM should monitor the number of erroneous messages and disconnect if it exceeds a threshold.

8 LIST OF FIGURES

9 LIST OF TABLES