Bluetooth Network Encapsulation Protocol (BNEP) Specification

Abstract:

The Bluetooth Network Encapsulation Protocol Specification describes the protocol to be used by Bluetooth profiles such as Personal Area Networking Profile. This document defines a packet format for Bluetooth network encapsulation used to transport common networking protocols over the Bluetooth media. Bluetooth network encapsulation supports the same networking protocols that are supported by IEEE 802.3/Ethernet encapsulation. Packets from the supported networking protocols are contained in Bluetooth network encapsulation packets, which are transported directly over the Bluetooth L2CAP protocol.

Disclaimer and Intellectual Property Notice

The copyright in these specifications is owned by the Promoter Members of Bluetooth SIG Inc. (“Bluetooth SIG”). Use of these specifications and any related intellectual property (collectively, the “Specification”) is governed by the Promoters Membership Agreement among the Promoter Members and Bluetooth SIG (the “Promoters Agreement”), certain membership agreements between Bluetooth SIG and its Adopter and Associate Members (the “Membership Agreements”), and the Bluetooth Specification Early Adopters Agreements (1.2 Early Adopters Agreements) among Early Adopter members of the unincorporated Bluetooth special interest group and the Promoter Members (the “Early Adopters Agreement”). Certain rights and obligations of the Promoter Members under the Early Adopters Agreements have been assigned to Bluetooth SIG by the Promoter Members.

Use of the Specification by anyone who is not a member of Bluetooth SIG or a party to an Early Adopters Agreement (each such person or party, a “Member”) is prohibited. The legal rights and obligations of each Member are governed by their applicable Membership Agreement, Early Adopters Agreement, or Promoters Agreement. No license, express or implied, by estoppel or otherwise, to any intellectual property rights are granted herein.

Any use of the Specification not in compliance with the terms of the applicable Membership Agreement, Early Adopters Agreement, or Promoters Agreement is prohibited and any such prohibited use may result in termination of the applicable Membership Agreement or Early Adopters Agreement and other liability permitted by the applicable agreement or by applicable law to Bluetooth SIG or any of its members for patent, copyright, and/or trademark infringement.

THE SPECIFICATION IS PROVIDED “AS IS” WITH NO WARRANTIES WHATSOEVER, INCLUDING ANY WARRANTY OF MERCHANTABILITY, NON-INFRINGEMENT, FITNESS FOR ANY PARTICULAR PURPOSE, SATISFACTORY QUALITY, OR REASONABLE SKILL OR CARE, OR ANY WARRANTY ARISING OUT OF ANY COURSE OF DEALING, USAGE, TRADE PRACTICE, PROPOSAL, SPECIFICATION, OR SAMPLE.

Each Member hereby acknowledges that products equipped with the Bluetooth™ technology ("Bluetooth™ Products") may be subject to various regulatory controls under the laws and regulations of various governments worldwide. Such laws and regulatory controls may govern, among other things, the combination, operation, use, implementation, and distribution of Bluetooth™ Products. Examples of such laws and regulatory controls include, but are not limited to, airline regulatory controls, telecommunications regulations, technology transfer controls, and health and safety regulations. Each Member is solely responsible for the compliance by their Bluetooth™ Products with any such laws and regulations and for obtaining any and all required authorizations, permits, or licenses for their Bluetooth™ Products related to such regulations within the applicable jurisdictions. Each Member acknowledges that nothing in the Specification provides any information or assistance in connection with securing such compliance, authorizations, or licenses. NOTHING IN THE SPECIFICATION CREATES ANY WARRANTIES, EITHER EXPRESS OR IMPLIED, REGARDING SUCH LAWS OR REGULATIONS.

ALL LIABILITY, INCLUDING LIABILITY FOR INFRINGEMENT OF ANY INTELLECTUAL PROPERTY RIGHTS OR FOR NON-COMPLIANCE WITH LAWS RELATING TO USE OF THE SPECIFICATION, IS EXPRESSLY DISCLAIMED. BY USE OF THE SPECIFICATION, EACH MEMBER EXPRESSLY WAIVES ANY CLAIM AGAINST BLUETOOTH SIG AND ITS PROMOTER MEMBERS RELATED TO USE OF THE SPECIFICATION.

Bluetooth SIG reserves the right to adopt any changes or alterations to the Specification as it deems necessary or appropriate and to adopt a process for adding new Bluetooth™ profiles after the release of the Specification.

Contents

Revision History

Revision	Date	Comments
0.0	April 4, 2000	Original Document Started
0.1	May 11, 2000	Added Ethernet encapsulation protocol proposal
0.45	July 24, 2000	Added Monte Carlo F2F feedback
0.46	Aug 16, 2000	Added Pittsburgh F2F feedback
0.5	September 26, 2000	Editorial changes Released to Adopters
0.6	October 12, 2000	Editorial changes Change to the protocol on the filters options
0.7	November 27, 2000	Editorial changes Add extension bit and extension header definitions
0.9	February 6, 2001	Minor Editorial changes
0.95	June 12, 2001	Editorial changes and changes based on BARB, BTI, BQRB, BTAB feedback. Support for 802.1p Filter Message clarification Filter Message for Multicast Connection Setup Network control message added
0.95a	June 26, 2001	Editorial changes and changes based on BARB & BTI adoption review.
0.96	September 9, 2001	Minor editorial and corrections changes.
0.96a	September 20, 2001	Editorial and correction changes.
0.96b	October 16, 2001	Editorial and correction changes.
0.96c	November 12, 2001	Editorial and clarification changes
0.96d	November 19, 2001	Editorial and clarification changes
0.96e	January 2, 2002	Moved note on section 3.3 to section 3.1 Editorial Changes
1.0	July 24, 2002	Errata and feedback incorporated. Interoperability testing completed
1.0RC	December 02, 2002	Updates to address issues raised from BTI, BQRB and individual comments

1.0RC2	December 06, 2002	PAN WG Meeting Edits
1.0RC3	December 17 2002	PAN WG Meeting editorial fixes
1.0	February 14, 2003	Specification Adopted

Contributors

Name	Company
David Moore	3COM Corporation
Tom Scribner	3COM Corporation
Barry Corlett	Agere Systems
Willy Sagefalk	Axis Communications
Dan Willey	Certicom Corporation
Horia Balog	Classwave Wireless Inc.
Conrad Maxwell	Conexant Systems
Mark Rison	CSR
Allan Bogeskov	Telefonaktiebolaget LM Ericsson
Theo Borst	Telefonaktiebolaget LM Ericsson
Per Johansson	Telefonaktiebolaget LM Ericsson
Tero Kauppinen	Telefonaktiebolaget LM Ericsson
Martin Kitchen	Telefonaktiebolaget LM Ericsson
Jesper Krogh	Telefonaktiebolaget LM Ericsson
Tony Larsson	Telefonaktiebolaget LM Ericsson
Johan Sorensen	Telefonaktiebolaget LM Ericsson
Dave Suvak	Extended Systems Inc.
Jean Tourrilhes	Hewlett Packard Corporation
Toru Aihara	International Business Machines Corporation
Chatschik Bisdikian	International Business Machines Corporation
Kris Fleming (Editor)	Intel Corporation
Robert Hunter	Intel Corporation
Jon Inouye	Intel Corporation
Diego Melpignano	Philips Inc.
Eiji Kato	Matsushita Electric Industrial

Mike Foley	Microsoft Corporation
Billy Brackenridge	Microsoft Corporation
Dale Farnsworth	Motorola Inc.
Brian Redding	Motorola Inc.
Carmen Kuhl	Nokia Corporation
Jaakko Lipasti	Nokia Corporation
James Scales	Nokia Corporation
Markus Schetelig	Nokia Corporation
Sander van Valkenburg	Nokia Corporation
Steven Kenny	Norwood Systems
Rebecca Ostergaard	Norwood Systems
Graeme Reid	Norwood Systems
Darrell Goff	Rappore
Daniel Shaw	Red-M Communications Ltd
Simon Harrison	Red-M Communications Ltd
Pravin Bhagwat	ReefEdge, Inc.
Daryl Hlasny	Sharp Laboratories of America Inc.
Leonard Ott	Socket Communications Inc.
Johannes Loebbert	Sony Corporation
Wilhelm Hagg	Sony Corporation
Takashi Sasai	Sony Corporation
Mike Blackstock	Synchropoint Wireless, Inc.
Yosuke Tajika	Toshiba Corporation
Tatuya Jinmei	Toshiba Corporation
Kazuo Nogami	Toshiba Corporation
Jim Hobza	Widcomm Inc.
Ravindranath Singamneni	Widcomm Inc.

1 Introduction

Bluetooth is a short-range wireless technology operating in the 2.4 GHz ISM band. Many devices such as notebook computers, phones, PDAs, Home Electric Appliances, and other computing devices will incorporate Bluetooth as a part of the device. Bluetooth enabled devices will have the ability to form networks and exchange information. For these devices to interoperate and exchange information, a common packet format needs to be defined to encapsulate layer 3 network protocols. This document defines the packet format used to transport common networking protocols over the Bluetooth media[5][6][7]. The packet format is based on EthernetII/DIX Framing as defined by IEEE 802.3[3][4].

Bluetooth Network Encapsulation Protocol (BNEP) encapsulates packets from various networking protocols, which are transported directly over the Bluetooth Logical Link Control and Adaptation Layer Protocol (L2CAP) [2]. L2CAP provides a data link layer for Bluetooth.

The Bluetooth Personal Area networking profile [1] describes how BNEP SHALL be used to provide networking capabilities for Bluetooth devices.

1.1 Bluetooth Networking Encapsulation Protocol (BNEP) Functional Requirements

The functional requirement for Bluetooth networking encapsulation protocol includes the following:

- Support for common networking protocols such as IPv4, IPv6, IPX, and other existing or emerging networking protocols as defined by the Network protocol types [3]. Many protocols are used for networking various computing devices together. Although IPv4 and IPv6 are perceived as the most important networking protocols, it is a requirement that Bluetooth Networking is able to supports other popular protocols
- Low Overhead -- The encapsulation format SHALL be bandwidth efficient.

1.2 Assumptions

1. This protocol is implemented using connection oriented L2CAP channels.

2. Bluetooth is considered to be a transmission media in the same OSI layer as Ethernet, Token Ring, ATM, etc.

3. L2CAP is considered to be the Bluetooth Data MAC (Media Access Control) Layer.

4. BNEP specifies a minimum L2CAP MTU of 1691 bytes[1].

5. The accepted rules of network connectivity and topology as defined for IEEE 802.3 (e.g. switching and routing) SHALL be applied to Bluetooth in a manner consistent with IEEE 802.3 media.

6. The Bluetooth BD_ADDR address space is administered by the IEEE, and is assigned from the Ethernet address space. This means that it is possible to build a Bluetooth network access point as a bridge between Bluetooth devices and an Ethernet network.

1.3 Scope

This document covers only the Bluetooth networking encapsulation packet format. The following items are beyond the scope of the Bluetooth Networking encapsulation document:

- Address Allocation
- Address Resolution
- Name Resolution
- Networking Security
- Routing
- Network Discovery
- Network Formation

The above issues are addressed in the Bluetooth Personal Area Networking profile document [1]. The Bluetooth Personal Area Networking profile [1] describes how the Bluetooth networking encapsulation is used to provide networking support.

1.4 Byte Order and Numeric Values

All values contained in the document are represented in hexadecimal notation. Multiple-byte fields are drawn with the more significant bytes toward the left and

[1] The minimum MTU of 1691 was selected based on the payload of a maximum Ethernet packet payload (1500 bytes, plus 4 bytes if an 802.1Q tag header is present) + BNEP header (15 bytes) + possible extension header. This minimum MTU is required to prevent violating any higher layer protocol assumptions about an "EthernetII/DIX Framing like" layer provided by BNEP. 1691 = 5*339(size of DH5) – 4 (L2CAP header).

the less significant bytes toward the right. The multiple-byte fields in the Bluetooth Networking encapsulation header are in standard network byte order (big endian), with more significant (byte 0 is the most significant byte) bytes being transferred before less-significant (low-order) bytes. Multiple-bit fields are drawn with the more significant bits toward the right and the less significant bits toward the left.

2 BNEP over L2CAP

2.1 Stack Overview

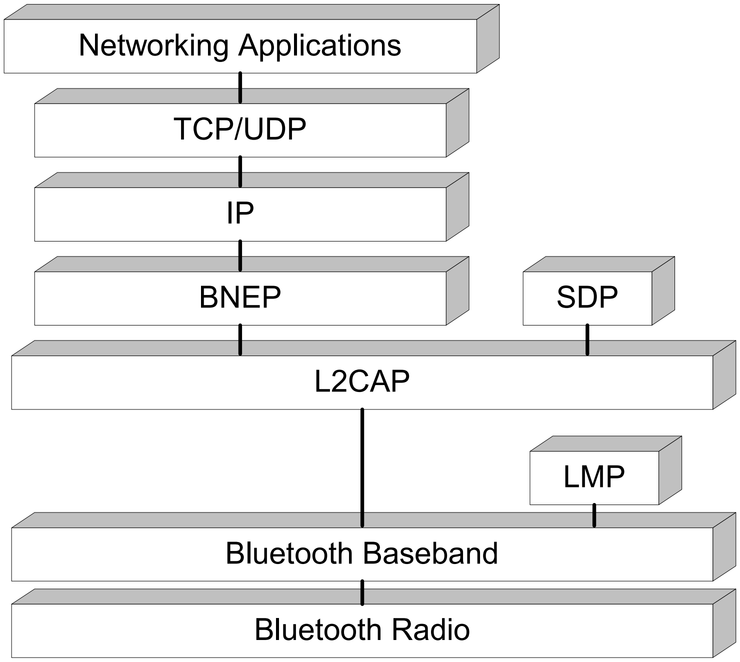


Figure 1: Stack Overview

2.2 Packet Encapsulation

The use of the BNEP for transporting an Ethernet packet is shown in Figure 2 on page 13. BNEP removes and replaces the Ethernet Header with the BNEP Header. Finally, both the BNEP Header and the Ethernet Payload is encapsulated by L2CAP and is sent over the Bluetooth media. The maximum payload that BNEP SHALL accept from the higher layer is equal to the negotiated L2CAP MTU (minimum value: 1691), minus 191 bytes (or 187 bytes if an IEEE 802.1Q tag header [9] is present) reserved for BNEP headers. This way it can be assured that enough frame buffer space is reserved to transmit all BNEP. The minimum payload that BNEP SHALL accept from the higher layer is zero; BNEP is not required to pad payloads to the Ethernet minimum size (46 bytes).

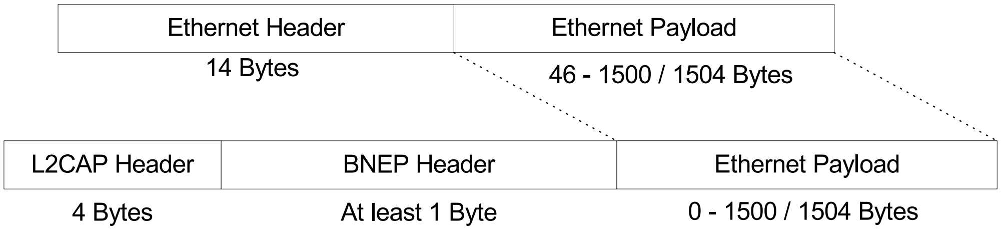


Figure 2: BNEP with an Ethernet Packet payload sent using L2CAP

2.3 BNEP Overview

BNEP is used for transporting both control and data packet over Bluetooth to provide networking capabilities for Bluetooth devices. BNEP provides capabilities that are similar to capabilities provided by Ethernet (EthernetII/DIX Framing /IEEE 802.3).

2.4 BNEP Header Formats

All BNEP Headers are in the following format as shown in Figure 3 on page 13. All devices supporting BNEP SHALL be able to interpret all defined BNEP packet types. BNEP capable devices MAY optionally transmit the BNEP compressed headers. Any packet containing a reserved BNEP header packet type SHALL be dropped. Processing of extension headers are defined in section 3 on page 39.

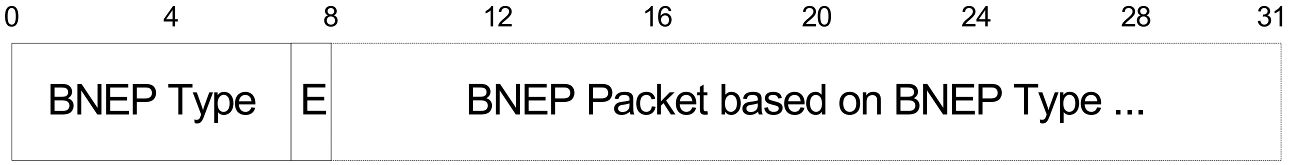


Figure 3 BNEP Header Format

BNEP Type: *Size: 7 Bits*

Value	Parameter Description
0x00 – 0x7F	Seven bit Bluetooth Network Encapsulation Protocol Type value identifies the type of BNEP header contained in this packet. Values are defined in Table 1 on page 14

Extension Flag (E): *Size: 1 Bit*

Value	Parameter Description
0x0 – 0x1	One bit extension flag that indicates if one or more extension headers follow the BNEP Header before the data payload if the data payload exists. Extension headers are defined in section 3 on page 39. If the extension flag is equal to 0x1 then one or more extension headers follows the BNEP header. If the extension flag is equal to 0x0 then the BNEP payload follows the BNEP header.

BNEP Packet: *Size: Based on BNEP Type*

Value	Parameter Description
0xXX	Based on the BNEP Type

2.4.1 BNEP Type Values

The Table 1 on page 14 defines the various BNEP packet formats

Value	BNEP Packet Type
0x00	BNEP_GENERAL_ETHERNET
0x01	BNEP_CONTROL
0x02	BNEP_COMPRESSED_ETHERNET
0x03	BNEP_COMPRESSED_ETHERNET_SOURCE_ONLY
0x04	BNEP_COMPRESSED_ETHERNET_DEST_ONLY
0x05 – 0x7E	Reserved for future use
0x7F	Reserved for 802.2 LLC Packets for IEEE 802.15.1 WG

Table 1: BNEP Types

2.5 BNEP_GENERAL_ETHERNET Packet Type Header Format

The BNEP_GENERAL_ETHERNET packet type header format is shown in Figure 4 on page 15. This packet type SHALL be used to carry Ethernet packets to and from Bluetooth networks.

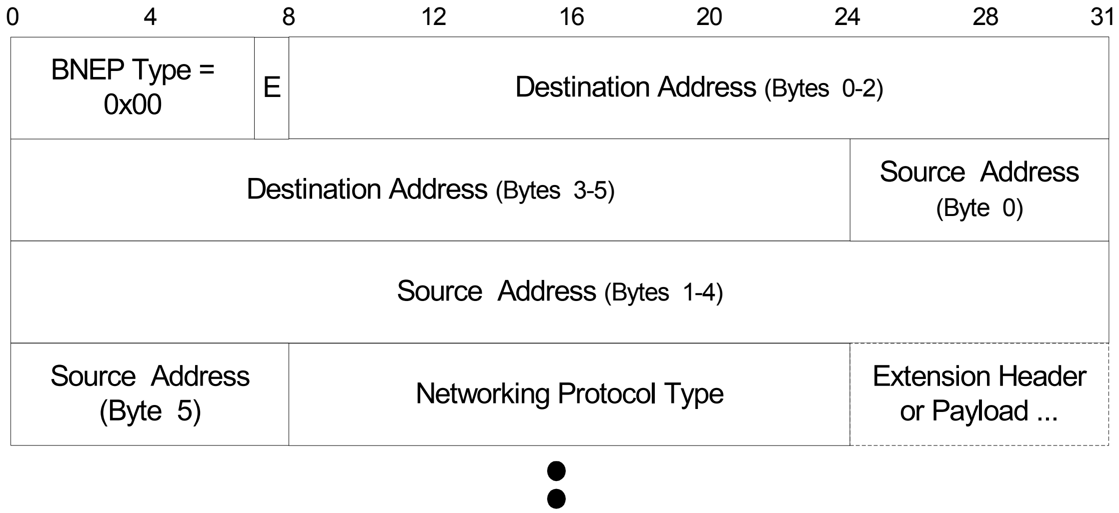

0
4
8
12
16
20
24
28
31
BNEP Type = 0x00
E
Destination Address (Bytes 0-2)
Destination Address (Bytes 3-5)
Source Address (Byte 0)
Source Address (Bytes 1-4)
Source Address (Byte 5)
Networking Protocol Type
Extension Header or Payload ...

Figure 4: BNEP_GENERAL_ETHERNET Packet Type Header

BNEP Type: *Size: 7 Bits*

Value	Parameter Description
0x00	Seven bit Bluetooth Network Encapsulation Protocol Type value identifies the type of BNEP header contained in this packet. SHALL be set to BNEP_GENERAL_ETHERNET

Extension Flag (E): *Size: 1 Bit*

Value	Parameter Description
0x0 – 0x1	One bit extension flag that indicates if one or more extension headers follow the BNEP Header before the data payload if the data payload exists. Extension headers are defined in section 3 on page 39. If the extension flag is equal to 0x1 then one or more extension headers follows the BNEP header. If the extension flag is equal to 0x0 then the BNEP payload follows the BNEP header.

Destination Address: *Size: 6 Bytes*

Value	Parameter Description
0xXXXXXXXXXXXX	48 bit Bluetooth device address/IEEE address of the destination of the BNEP packet/Ethernet frame

	contained in the payload.

Source Address: *Size: 6 Bytes*

Value	Parameter Description
0xXXXXXXXXXXXX	48 bit Bluetooth device address/IEEE address of the source of the BNEP packet/Ethernet frame contained in the payload.

Networking Protocol Type: *Size: 2 Bytes*

Value	Parameter Description
0xXXXX	16 bit type field identifies the type of networking protocol contained in the payload. The values for this field are the same as defined for Ethernet types in [3]

Either the destination or the source address MAY be an IEEE Ethernet address, if the actual destination/source is an IEEE device and not a Bluetooth device. BNEP SHALL use IEEE Ethernet broadcast and multicast addresses for the destination addresses for broadcast and multicast packets (IEEE Ethernet unicast addresses SHALL always be used for source addresses). Note: Networking Protocol Types as used in this specification SHALL be taken to include values in the range 0x0000-0x05dc, used to represent the IEEE802.3 length interpretation of the IEEE802.3 length/type field. It is not, however, mandatory to process packets with such Networking Protocol Types, and even if such packets are processed, it is not mandatory to support IEEE802.2 LLC (and so SNAP headers).

2.6 BNEP_ CONTROL Packet Type Header Format

The BNEP_ CONTROL packet type header format is shown in Figure 5 on page 17. This packet type is mandatory to recognize and respond to accordingly. The BNEP_CONTROL packet type is used to exchange control information. Note: In BNEP_CONTROL packets, the entire control packet is contained in the BNEP_CONTROL header and therefore no part of the BNEP_CONTROL packet is contained in the payload section of the BNEP packet.

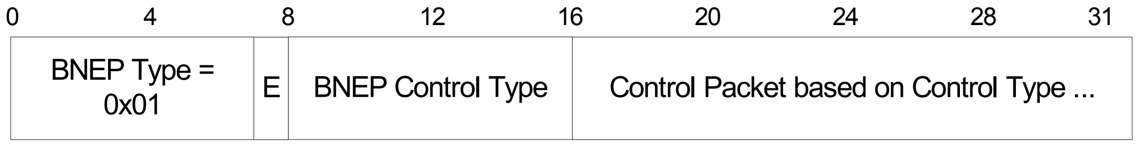


Figure 5: BNEP_ CONTROL Packet Type Header

BNEP Type: *Size: 7 Bits*

Value	Parameter Description
0x01	Seven bit Bluetooth Network Encapsulation Protocol Type value identifies the type of BNEP header contained in this packet. SHALL be set to BNEP _CONTROL

Extension Flag (E): *Size: 1 Bit*

Value	Parameter Description
0x0 – 0x1	One bit extension flag that indicates if one or more extension headers follow the BNEP Control header. Extension headers are defined in section 3 on page 39. If the extension flag is equal to 0x1 then one or more extension headers follows the BNEP control header.

BNEP Control Type: *Size: 1 Byte*

Value	Parameter Description
0x00 – 0xFF	Type of BNEP control message contained in the packet

2.6.1 BNEP Control Type Values

Table 2 on page 18 defines the various BNEP control and response message values to for the BNEP Control type field.

Value	BNEP Control Type
0x00	BNEP_CONTROL_COMMAND_NOT_UNDERSTOOD
0x01	BNEP_SETUP_CONNECTION_REQUEST_MSG
0x02	BNEP_SETUP_CONNECTION_RESPONSE_MSG

0x03	BNEP_FILTER_NET_TYPE_SET_MSG
0x04	BNEP_FILTER_NET_TYPE_RESPONSE_MSG
0x05	BNEP_FILTER_MULTI_ADDR_SET_MSG
0x06	BNEP_FILTER_MULTI_ADDR_RESPONSE_MSG
0x07 – 0xFF	Reserved for future use

Table 2: BNEP Control Types

2.6.2 BNEP_CONTROL_COMMAND_NOT_UNDERSTOOD Control Command Packet

This packet SHALL be used to reply to any control message received, which contains an unknown BNEP control type value. This allows devices to response to new control message that might be used in the future.

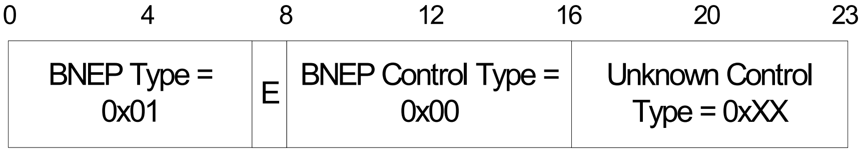


Figure 6: BNEP_CONTROL_COMMAND_NOT_UNDERSTOOD control message format

BNEP Type: *Size: 7 Bits*

Value	Parameter Description
0x01	Seven bit Bluetooth Network Encapsulation Protocol Type value identifies the type of BNEP header contained in this packet. SHALL be set to BNEP_CONTROL

Extension Flag (E): *Size: 1 Bit*

Value	Parameter Description
0x0 – 0x1	One bit extension flag that indicates if one or more extension headers follow the BNEP control header. Extension headers are defined in section 3 on page 39. If the extension flag is equal to 0x1 then one or more extension headers follows the BNEP header.

BNEP Control Type: *Size: 1 Byte*

Value	Parameter Description
0x00	Type of BNEP control message contained in the packet. SHALL be set to

	BNEP_CONTROL_COMMAND_NOT_UNDERSTOOD

Unknown Control Type: *Size: 1 Byte*

Value	Parameter Description
0xXX	Type of BNEP control message that was previously received and caused this message to be sent.

2.6.3 BNEP_SETUP_CONTROL Packets

This packet type SHALL contain control messages used to setup the initial connection information about the BNEP connection. All devices that support BNEP SHALL be able to recognize and respond accordingly to all BNEP_SETUP_CONTROL packets.

BNEP_SETUP_CONTROL packet types SHALL be processed in the order that they are received. For each connection, only one outstanding BNEP_SETUP_CONNECTION_REQUEST_MSG message is allowed. A BNEP_SETUP_CONNECTION_RESPONSE_MSG message SHALL be used to respond to each control message received. If a response message is not received, after T_{crt} time has elapsed, then the outstanding BNEP_SETUP_CONNECTION_REQUEST_MSG message can be assumed to be lost and the same BNEP_SETUP_CONNECTION_REQUEST_MSG message can be retransmitted. The range for T_{crt} is from 1 second to 30 seconds, with a suggested timeout value to be 10 seconds. This message SHALL only be retransmitted a limited number of times, at which time the intended receiver SHALL be determined unresponsive and the connection SHALL be disconnected. BNEP packets of type BNEP_SETUP_CONTROL are for the device with direct connection communication only, and SHALL NOT be forwarded.

2.6.3.1 BNEP_SETUP_CONNECTION_REQUEST_MSG setup control message format

The BNEP_SETUP_CONNECTION_REQUEST_MSG setup control message format is shown in Figure 7 on page 21. The purpose of this control message is to inform the peer entity of the destination and source SDP service UUIDs [8] which are being used for this BNEP connection.

This control message is also used to change the current BNEP roles for an established BNEP connection; if such a request is not successful the existing BNEP connection and BNEP roles are maintained.

This control message SHALL NOT be sent in an extension header.

The device which initiated the L2CAP connection for BNEP SHALL send this control message and the responding device SHALL send a successful response to it at the beginning of the BNEP session (the responding device SHALL NOT send this control message before then).

Specifically:

1. The "ignore/complain" rule is that if a control message has a reserved control type (as defined in Table 2 on page 18), then a BNEP_CONTROL_COMMAND_NOT_UNDERSTOOD message SHALL be sent (as defined in section 2.6.2 on page 18), else that control message SHALL be ignored.

2. The initiating device SHALL consider the BNEP connection established when (after sending a BNEP_SETUP_CONNECTION_REQUEST_MSG) it receives a BNEP_CONTROL packet whose control type is BNEP_SETUP_CONNECTION_RESPONSE_MSG and which indicates success.

   If, before the BNEP connection has been established, the initiating device receives a packet whose type is valid but is not BNEP_CONTROL or whose control type is not BNEP_SETUP_CONNECTION_RESPONSE_MSG, then it SHALL apply the ignore/complain rule to any control messages in that packet (whether in the BNEP header or in an extension header), and ignore any Ethernet payload.

   If, before the BNEP connection had been established, the initiating device receives a BNEP_SETUP_CONNECTION_RESPONSE_MSG which does not indicate success, then it SHALL apply the ignore/complain rule to any control messages in extension headers in that packet.

3. When the responding device receives a BNEP_CONTROL packet whose control type is BNEP_SETUP_CONNECTION_REQUEST_MSG it SHALL respond with a BNEP_SETUP_CONNECTION_RESPONSE_MSG. If the setup request is acceptable (i.e. the response indicates success), then it SHALL consider the BNEP connection established.

   If, before the BNEP connection has been established, the responding device receives a packet whose type is valid but is not BNEP_CONTROL or whose control type is not BNEP_SETUP_CONNECTION_REQUEST_MSG, then it SHALL apply the ignore/complain rule to any control messages in that packet (whether in the BNEP header or in an extension header), and ignore any Ethernet payload.

If, before the BNEP connection had been established, the responding device receives a BNEP_SETUP_CONNECTION_REQUEST_MSG which is not acceptable, then it SHALL apply the ignore/complain rule to any control messages in extension headers in that packet.

4. The initiating device's BNEP_SETUP_CONNECTION_REQUEST_MSG MAY contain extension headers. If the setup request is acceptable to the responding device then these headers SHALL be processed normally; any responses MAY be sent as extension headers in the BNEP_SETUP_CONNECTION_RESPONSE_MSG (if the responses are sent in separate packets then the BNEP_SETUP_CONNECTION_RESPONSE_MSG SHALL be sent first). If the setup request is not acceptable, then these headers SHALL only be processed and responded to in the normal way if the BNEP connection is already established; if the BNEP connection is not yet established then the ignore/complain rule SHALL be applied to any control messages in extension headers in that packet, as specified above.

   For example, the initiating device MAY, before BNEP connection establishment, send a BNEP_SETUP_CONNECTION_REQUEST_MSG with a BNEP_FILTER_NET_TYPE_SET_MSG extension and a reserved control type extension. If the setup request is acceptable the responding device SHALL respond to the setup request, the filtering request and the reserved control type extension, and MAY for example do so by sending a BNEP_SETUP_CONNECTION_RESPONSE_MSG with a BNEP_FILTER_NET_TYPE_RESPONSE_MSG extension and a BNEP_CONTROL_COMMAND_NOT_UNDERSTOOD extension. If, however, the setup request is not acceptable the responding device SHALL respond to the setup request, ignore the filtering request, and send a BNEP_CONTROL_COMMAND_NOT_UNDERSTOOD for the reserved control type extension.

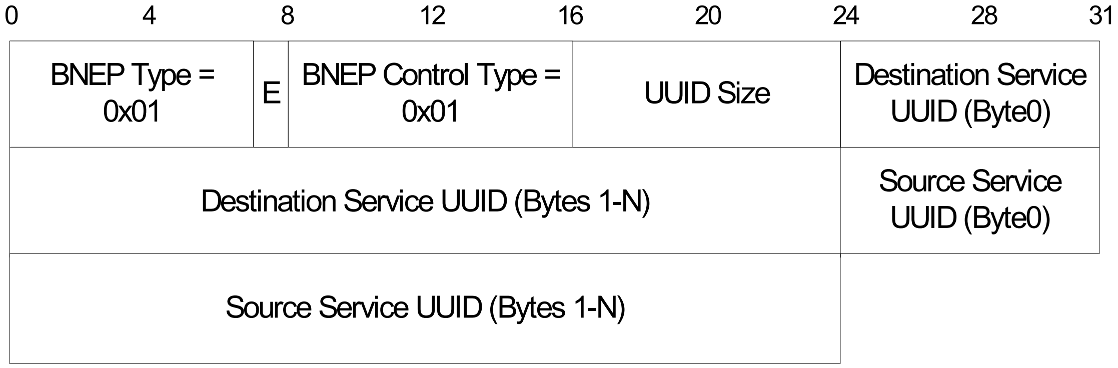

0
4
8
12
16
20
24
28
31
BNEP Type = 0x01
E
BNEP Control Type = 0x01
UUID Size
Destination Service UUID (Byte0)
Destination Service UUID (Bytes 1-N)
Source Service UUID (Byte0)
Source Service UUID (Bytes 1-N)

Figure 7: BNEP_SETUP_CONNECTION_REQUEST_MSG control message format

BNEP Type: *Size: 7 Bits*

Value	Parameter Description
0x01	Seven bit Bluetooth Network Encapsulation Protocol Type value identifies the type of BNEP header contained in this packet. SHALL be set to BNEP_CONTROL

Extension Flag (E): *Size: 1 Bit*

Value	Parameter Description
0x0 – 0x1	One bit extension flag that indicates if one or more extension headers follow the BNEP control header. Extension headers are defined in section 3 on page 39. If the extension flag is equal to 0x1 then one or more extension headers follows the BNEP header.

BNEP Control Type: *Size: 1 Byte*

Value	Parameter Description
0x01	Type of BNEP control message contained in the packet. SHALL be set to BNEP_SETUP_CONNECTION_REQUEST_MSG

UUID Size: *Size: 1 Byte*

Value	Parameter Description
0xXX	1 byte field identifies the length of one of the SDP service UUID [8], measured in bytes. Note: The size value is the length of one of the SDP service UUID and therefore the length of both the destination and source service UUIDs SHALL be the same.

Destination Service UUID: *Size: 2-16 Bytes*

Value	Parameter Description
0xXX	Depending on the UUID Size parameter, this is a 2-16 byte field containing the destination (service which the source device is connecting to) SDP service UUIDs [8]. Note: The size of both the destination and source service UUID SHALL be the same.

Source Service UUID: *Size: 2-16 Bytes*

Value	Parameter Description

0xXX	Depending on the UUID Size parameter, this is a 2-16 byte field containing the source (the service that the source device is using for the BNEP connection) SDP service UUIDs [8]. Note: The size of both the destination and source service UUID SHALL be the same.

2.6.3.2 BNEP_SETUP_ CONNECTION_RESPONSE_MSG response message format

The BNEP_SETUP_ CONNECTION_RESPONSE_MSG response message format is shown in Figure 8 on page 23. The response message SHALL be used to respond to each BNEP_SETUP_CONNECTION_REQUEST_MSG control message. Each of the received setup control messages SHALL be responded to by one response message.

The BNEP_SETUP_CONNECTION_RESPONSE_MSG SHALL NOT be sent in an extension header.

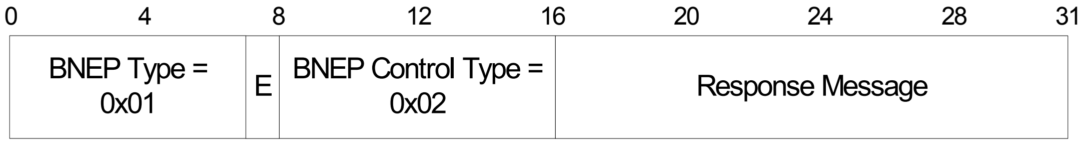


Figure 8: BNEP_SETUP_ CONNECTION_RESPONSE_MSG response message format

BNEP Type: *Size: 7 Bits*

Value	Parameter Description
0x01	Seven bit Bluetooth Network Encapsulation Protocol Type value identifies the type of BNEP header contained in this packet. SHALL be set to BNEP_CONTROL

Extension Flag (E): *Size: 1 Bit*

Value	Parameter Description
0x0 – 0x1	One bit extension flag that indicates if one or more extension headers follow the BNEP control header. Extension headers are defined in section 3 on page 39. If the extension flag is equal to 0x1 then one or more extension headers follows the BNEP header.

BNEP Control Type: *Size: 1 Byte*

Value	Parameter Description
0x02	Type of BNEP control message contained in the packet. SHALL be set to BNEP_SETUP_ CONNECTION_RESPONSE_MSG

Response Message: *Size: 2 Bytes*

Value	Parameter Description
0xXXXX	16 bit field identifies the response to the previous Setup Control Message. Valid responses are contained in Table 3

2.6.3.2.1 Response Messages

Table 3 on page 24 contains a list of the valid response messages to be used to respond to setup control messages.

Value	Response Messages
0x0000	Operation Successful
0x0001	Operation Failed: Invalid Destination Service UUID (the UUID is not a service UUID defined by the profile which is using BNEP)
0x0002	Operation Failed: Invalid Source Service UUID (the UUID is not a service UUID defined by the profile which is using BNEP)
0x0003	Operation Failed: Invalid Service UUID Size
0x0004	Operation Failed: Connection not allowed
0x0005 – 0xFFFF	Reserved for future use

Table 3: Setup Connection Response Messages

2.6.4 BNEP_FILTER_CONTROL Packets

This packet type contains control messages used to control which types of packets are to be transmitted over BNEP. Although all devices that support BNEP SHALL be able to recognize and respond accordingly to all BNEP_FILTER_CONTROL packets, the implied functionality to do filtering is optional and does not have to be supported by all devices.

BNEP_FILTER_CONTROL packet types SHALL be processed in the order that they are received. This allows Bluetooth devices to determine which types of packets are to be filtered to save networking bandwidth. For each connection, only one outstanding BNEP_FILTER_CONTROL_SET message for each filter type is allowed. Therefore there can be one BNEP_FILTER_NET_TYPE_SET and one BNEP_FILTER_MULTI_ADDR_SET outstanding message. A BNEP_FILTER_CONTROL_RESPONSE message SHALL be used to respond to each filter control message received. The response message accepts or rejects all of the filter message parameters. If the filter control message is rejected the filter settings remain unchanged. If a response message is not received after T_{frt} time has elapsed, then the outstanding BNEP_FILTER_CONTROL_SET message can be assumed to be lost and the same BNEP_FILTER_CONTROL_SET message SHALL be retransmitted at least once. The range for T_{frt} is from 1 second to 30 seconds, with a suggested timeout value of 10 seconds. This message SHALL only be retransmitted a limited number of times, at which time the intended receiver SHALL be determined unresponsive and the connection SHALL be disconnected. BNEP packets of type BNEP_FILTER_CONTROL are for the device with direct connection communication only, and SHALL NOT be forwarded.

Note that any new filter control message replaces the previous filter control settings. Note that ranges specified in filter control requests MAY overlap. Only packets not included in any filter range SHALL be filtered out.

If a filtered data packet contains one or more unknown extension headers (see section 3 on page 39) the Ethernet payload SHALL be removed and the unknown extension headers transmitted irrespective of the network protocol or multicast filter settings, and any local filtering policy. The network protocol type SHALL be set to 0 to reflect the new Ethernet payload length.

Note: BNEP implementations should be aware of the fact that in case of IEEE802.1Q, the actual 'network protocol type' field that is being used in determining packet filtering, has to be found after the TCI component of the IEEE802.1Q header and before the actual Ethernet payload. This is the field that SHALL be set to zero if the Ethernet payload has been filtered out. The IEEE802.1Q header SHALL be forwarded unchanged.

2.6.5 BNEP Filter Network Protocol Type

This packet type SHALL contain control messages used to control which Network Protocol types SHALL be filtered and not transmitted over BNEP. By default all Network Protocol types are not filtered, but network administrators MAY configure overriding default filters.

2.6.5.1 BNEP_FILTER_ NET_TYPE_SET_MSG filter control message format

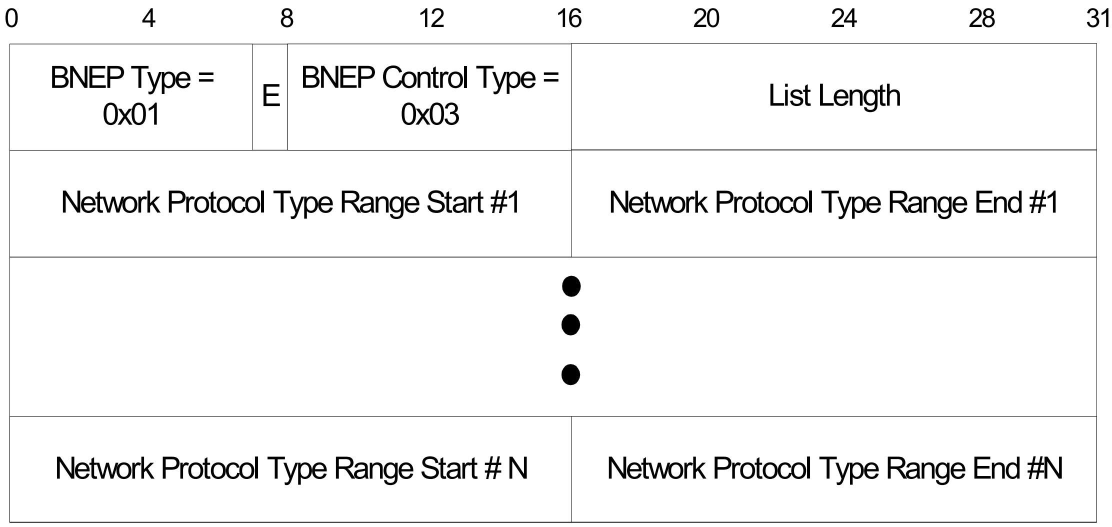
The BNEP_FILTER_ NET_TYPE_SET_MSG filter control message format is shown in Figure 9 on page 27. The purpose of this control message is to inform the peer entity of the set of Networking Protocol Types that the sender whishes to receive[2]. Note that the filter control message does not change the settings, unless its response message returns Operation Successful.

The length (in octets) of this message is 4+4*N, where N is the number of disjoint ranges of Networking protocol types that form the complete set. Note that N=0 (empty set) denotes a reset to default filters (if any) supported by the remote device.

When the filtering is enabled and supported, only packets containing networking protocol types that are within the indicated Networking Protocol Type range(s) are sent using BNEP. See section 5.5 for an example of how filters MAY be applied. Note: Some BNEP packets MAY contain IEEE 802.1Q tag header and all devices supporting BNEP SHALL be able to understand IEEE 802.1Q tag header, see section 4.1 on page 42. In addition, depending on the value of the network protocol type field, the actual network protocol type embedded in the payload MAY be contained at a known offset, which can be determined based on the network protocol type field contained in the BNEP header. Note: BNEP implementations SHALL be aware of the fact that in case of IEEE 802.1Q tag header, the actual 'network protocol type' field that is being used in determining packet filtering, has to be found inside the BNEP packet data payload after the remaining IEEE 802.1Q tag header and before the actual Ethernet payload. Note: If a device wishes to set a filter but receive packets with a Networking Protocol Type corresponding to the IEEE802.3 length interpretation, it SHALL include the range 0x0000-0x05dc in its Networking Protocol Type filter request.

Ethernet type/length values the range 0x0000 - 0x05dc (Length), 0x05dd - 0x05ff (Reserved in IEEE 802.3) and 0x0600-0xFFFF ((Ethernet) Type) SHALL be considered valid for a BNEP_FILTER_NET_TYPE_SET_MSG. Frames with these type/length values SHALL be sent over the BNEP link as filter settings and local device rules dictate. If any of its requested ranges has a start value greater than its end value, a BNEP_FILTER_NET_TYPE_SET_MSG SHALL be rejected with response message value 0x0002 (Operation Failed: Invalid Networking Protocol Type Range).

[2] In Network access points, the actual filters being applied may be further affected by policies set by a network administrator.



<table>
<tr><td>0</td><td>4</td><td>8</td><td>12</td><td>16</td><td>20</td><td>24</td><td>28</td><td>31</td></tr>
<tr><td colspan="2">BNEP Type = 0x01</td><td>E</td><td>BNEP Control Type = 0x03</td><td colspan="5">List Length</td></tr>
<tr><td colspan="4">Network Protocol Type Range Start #1</td><td colspan="5">Network Protocol Type Range End #1</td></tr>
<tr><td colspan="9">●
●
●</td></tr>
<tr><td colspan="4">Network Protocol Type Range Start # N</td><td colspan="5">Network Protocol Type Range End #N</td></tr>
</table>

Figure 9: BNEP_FILTER_ NET_TYPE_SET_MSG control message format

BNEP Type: *Size: 7 Bits*

Value	Parameter Description
0x01	Seven bit Bluetooth Network Encapsulation Protocol Type value identifies the type of BNEP header contained in this packet. SHALL be set to BNEP_ CONTROL

Extension Flag (E): *Size: 1 Bit*

Value	Parameter Description
0x0 – 0x1	One bit extension flag that indicates if one or more extension headers follow the BNEP control header. Extension headers are defined in section 3 on page 39. If the extension flag is equal to 0x1 then one or more extension headers follows the BNEP header.

BNEP Control Type: *Size: 1 Byte*

Value	Parameter Description
0x03	Type of BNEP control message contained in the packet. SHALL be set to BNEP_FILTER_ NET_TYPE_SET_MSG

List Length: *Size: 2 Byte*

Value	Parameter Description

0xXXXX	Length of the start and end range of network protocol types list. This field defines the number of bytes contained in the entire list.

The following two fields are repeated together N times. The range of N is from 0 to 421 if this command consumes the entire BNEP packet and 0 to 63 if this command is contained in an extension header and consumes the entire extension header payload.

Networking Protocol Type Start #N: *Size: 2 Bytes*

Value	Parameter Description
0xXXXX	16 bit type field identifies the start of a range of type of networking protocols to be enabled. The values for this field are the same as defined for Ethernet types in [3]. The value SHALL be less then or equal to the Networking Protocol Type End value.

Networking Protocol Type End #N: *Size: 2 Bytes*

Value	Parameter Description
0xXXXX	16 bit type field identifies the end of the range of type of networking protocols to be enabled. The values for this field are the same as defined for Ethernet types in [3]

2.6.5.2 BNEP_FILTER_ NET_TYPE_RESPONSE_MSG response message format

The BNEP_FILTER_ NET_TYPE_RESPONSE_MSG response message format is shown in Figure 10 on page 28. The response message SHALL be used to respond to each BNEP_FILTER_NET_TYPE_SET_MSG filter control message. Each of the received filter control messages SHALL be responded to by one response message. Note that the filter control message does not change the settings unless its response message returns Operation Successful.

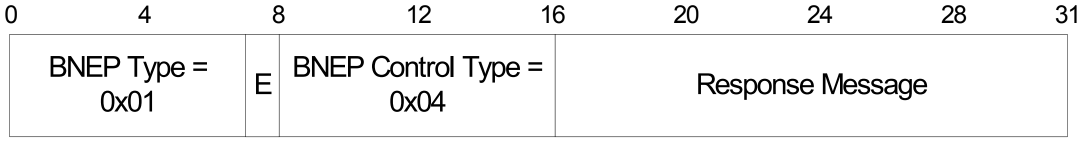


Figure 10: BNEP_FILTER_ NET_TYPE_RESPONSE_MSG response message format

BNEP Type: *Size: 7 Bits*

Value	Parameter Description

0x01	Seven bit Bluetooth Network Encapsulation Protocol Type value identifies the type of BNEP header contained in this packet. SHALL be set to BNEP_ CONTROL

Extension Flag (E): *Size: 1 Bit*

Value	Parameter Description
0x0 – 0x1	One bit extension flag that indicates if one or more extension headers follow the BNEP control header. Extension headers are defined in section 3 on page 39. If the extension flag is equal to 0x1 then one or more extension headers follows the BNEP header.

BNEP Control Type: *Size: 1 Byte*

Value	Parameter Description
0x04	Type of BNEP control message contained in the packet. SHALL be set to BNEP_FILTER_ NET_TYPE_RESPONSE_MSG

Response Message: *Size: 2 Bytes*

Value	Parameter Description
0xXXXX	16 bit type field identifies the response to the previous Filter Control Message. Valid responses are contained in Table 4 on page 30.

2.6.5.2.1 Response Messages

Table 4 on page 30 contains a list of the valid response messages to be used to respond to filter control messages.

Value	Response Messages
0x0000	Operation Successful
0x0001	Unsupported Request
0x0002	Operation Failed: Invalid Networking Protocol Type Range (A Networking Protocol Type Start value is greater than its corresponding Networking Protocol Type End value)
0x0003	Operation Failed: Too many filters
0x0004	Operation Failed: Unable to fulfill request due to security reasons.
0x0005 – 0xFFFF	Reserved for future use

Table 4: Network Protocol Type Filter Response Messages

2.6.6 BNEP Filter Multicast Address Type

This packet type contains control messages used to control which multicast destination addresses SHALL NOT be filtered and transmitted over BNEP. By default all multicast addresses are not filtered. Note: The broadcast address 0xFF:0xFF:0xFF:0xFF:0xFF:0xFF is a valid multicast address and SHALL be used accordingly.

2.6.6.1 BNEP_FILTER_ MULTI_ADDR _SET_MSG filter control message format

The BNEP_FILTER_ MULTI_ADDR _SET_MSG filter control message format is shown in Figure 11 on page 31. The purpose of this control message is to inform the peer entity of the set of multicast addresses that the sender wishes to receive and all other multicast addresses not contained in the set SHALL be filtered[3]. Note that the filter control message does not change the settings, unless its response message returns Operation Successful.

The length (in octets) of this message is 4+ (2*6)*N, where N is the number of multicast addresses that form the complete set. Note that N=0 (empty set) denotes a reset to default filters (if any) supported by the remote device.

When the filtering is enabled and supported, only packets with the Destination Address field contained in one of the multicast address ranges contained in BNEP_FILTER_MULTI_ADDR_SET_MSG are sent using BNEP. See section 5.6 for an example of how multicast address filters can be applied.

Filters SHALL only be set for those multicast and broadcast addresses included in a BNEP_FILER_MULTI_ADDR_SET_MSG. Unicast[4] addresses included in a range shall be ignored and not result in any unicast frames being dropped. For example, a device issuing a filter request which has a single range starting at 0x000000000000 and ends at 0x000000000000 is indicating it does not wish to receive any multicast or broadcast addressed frames, it SHALL still receive unicast traffic. A device issuing a filter request which starts at 0x000000000000 and ends at 0xFFFFFFFFFFFF is indicating it wishes to receive all multicast and broadcast addressed frames in addition to unicast traffic (this is the same as not issuing a filter request).

[3] In Network access points, the actual filters being applied may be further affected by policies set by a network administrator.

[4] Unicast: A unicast address is one which has the least significant bit of the first octet of the address set to 0.

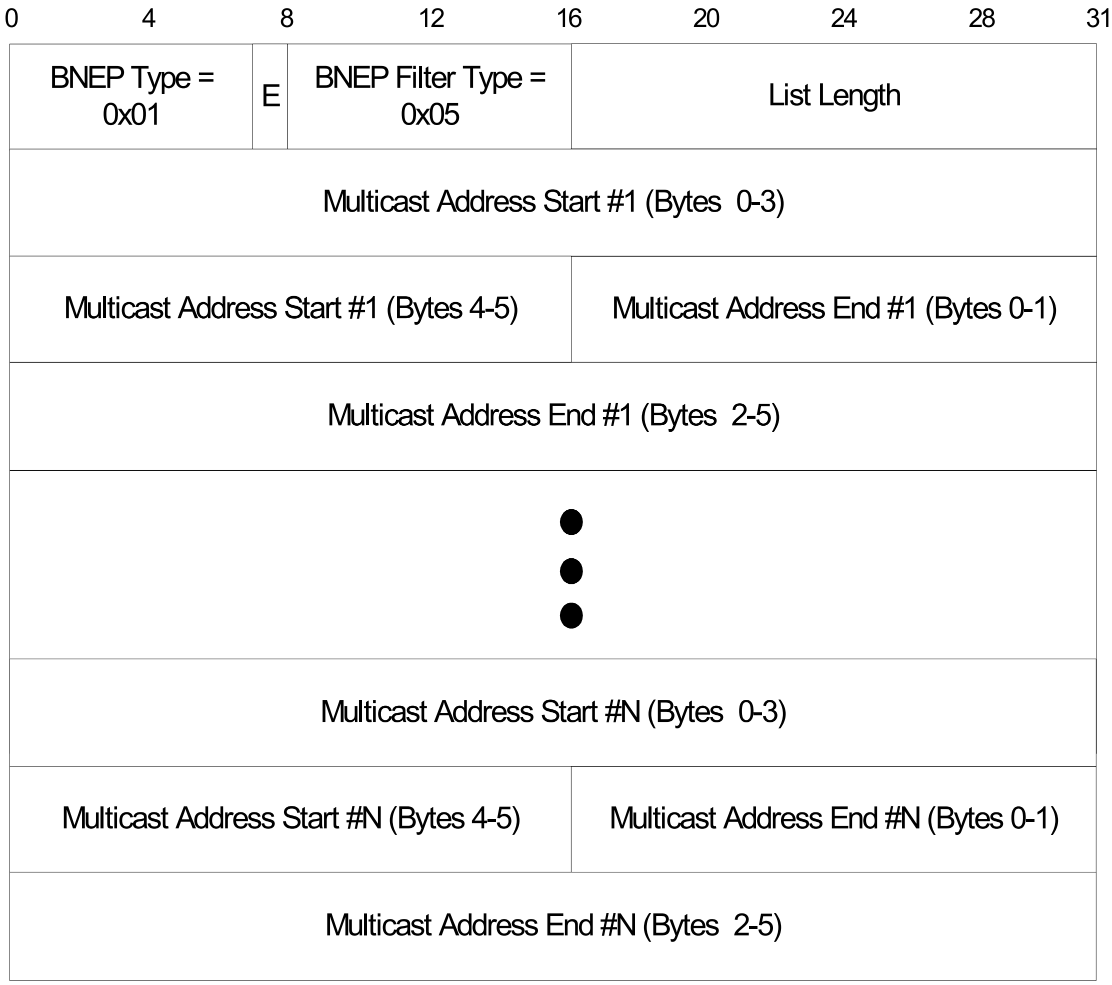

0
4
8
12
16
20
24
28
31
BNEP Type = 0x01
E
BNEP Filter Type = 0x05
List Length
Multicast Address Start #1 (Bytes 0-3)
Multicast Address Start #1 (Bytes 4-5)
Multicast Address End #1 (Bytes 0-1)
Multicast Address End #1 (Bytes 2-5)
Multicast Address Start #N (Bytes 0-3)
Multicast Address Start #N (Bytes 4-5)
Multicast Address End #N (Bytes 0-1)
Multicast Address End #N (Bytes 2-5)

Figure 11: BNEP_FILTER_MULTI_ADDR_SET_MSG control message format

BNEP Type: *Size: 7 Bits*

Value	Parameter Description
0x01	Seven bit Bluetooth Network Encapsulation Protocol Type value identifies the type of BNEP header contained in this packet. SHALL be set to BNEP_ CONTROL

Extension Flag (E): *Size: 1 Bit*

Value	Parameter Description
0x0 – 0x1	One bit extension flag that indicates if one or more extension headers follow the BNEP control header. Extension headers are defined in section 3 on page 39. If the extension flag is equal to 0x1 then one or more

	extension headers follows the BNEP header.

BNEP Control Type: *Size: 1 Byte*

Value	Parameter Description
0x05	Type of BNEP control message contained in the packet. SHALL be set to BNEP_FILTER_ MULTI_ADDR_SET_MSG

List Length: *Size: 2 Byte*

Value	Parameter Description
0xXXXX	Length of the start and end range of multicast IEEE address range list. This field defines the number of bytes contained in the entire list.

The following two fields are repeated together N times. The range of N is from 0 to 140 if this command consumes the entire BNEP packet and 0 to 21 if this command is contained in an extension header and consumes the entire extension header payload.

Multicast Address Start #N: *Size: 6 Bytes*

Value	Parameter Description
0xXXXXXXXXXXXX	Start of the range of 48 bit multicast IEEE address not to be filtered. The value SHALL be less then or equal to the Multicast Address End value.

Multicast Address End #N: *Size: 6 Bytes*

Value	Parameter Description
0xXXXXXXXXXXXX	End of the range of 48 bit multicast IEEE address not to be filtered.

2.6.6.2 BNEP_FILTER_MULTI_ADDR_RESPONSE_MSG response message format

The BNEP_FILTER_MULTI_ADDR_RESPONSE_MSG response message format is shown in Figure 12 on page 33. The response message SHALL be used to respond to each BNEP_FILTER_MULTI_ADDR_SET_MSG filter control message. Each of the received filter control messages SHALL be responded to by one response message. Note that the filter control message does not change the settings unless its response message returns Operation Successful.

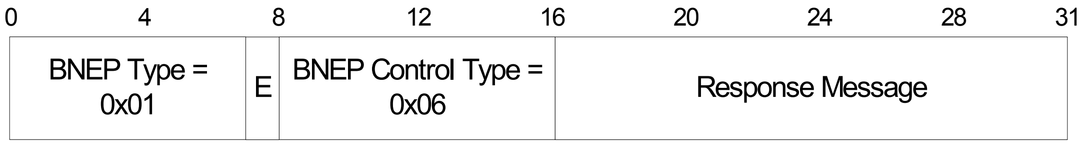


Figure 12: BNEP_FILTER_MULTI_ADDR_RESPONSE_MSG response message format

BNEP Type: *Size: 7 Bits*

Value	Parameter Description
0x01	Seven bit Bluetooth Network Encapsulation Protocol Type value identifies the type of BNEP header contained in this packet. SHALL be set to BNEP_ CONTROL

Extension Flag (E): *Size: 1 Bit*

Value	Parameter Description
0x0 – 0x1	One bit extension flag that indicates if one or more extension headers follow the BNEP control header. Extension headers are defined in section 3 on page 39. If the extension flag is equal to 0x1 then one or more extension headers follows the BNEP header.

BNEP Control Type: *Size: 1 Byte*

Value	Parameter Description
0x06	Type of BNEP control message contained in the packet. SHALL be set to BNEP_FILTER_MULTI_ADDR_RESPONSE_MSG

Response Message: *Size: 2 Bytes*

Value	Parameter Description
0xXXXX	16 bit type field identifies the response to the previous Multicast Address Filter Control Message. Valid responses are contained in Table 5 on page 34.

2.6.6.2.1 Response Messages

Table 5 on page 34 contains a list of the valid response messages to be used to respond to Multicast Address filter control messages.

Value	Response Messages
0x0000	Operation Successful
0x0001	Unsupported Request
0x0002	Operation Failed: Invalid multicast address (A Multicast Address Start is greater than the corresponding Multicast Address End)
0x0003	Operation Failed: Too many filters.
0x0004	Operation Failed: Unable to fulfill request due to security reasons.
0x0005 – 0xFFFF	Reserved for future use

Table 5: Multicast Address Filter Response Messages

2.7 BNEP_COMPRESSED_ETHERNET Packet Type Header Format

The BNEP_COMPRESSED_ETHERNET packet type header format is shown in Figure 13 on page 34. The header format is based on one of the compressed versions of the Ethernet header supported by BNEP. This packet type SHALL be used to carry Ethernet packets to and from devices that are directly connected at L2CAP level (have a valid L2CAP channel for BNEP) using BNEP. This compressed header MAY be used when two Bluetooth devices are exchanging packets, in which the source address is set to the local device's address which is the source device sending the packet and destination address is set to the other device's address which is the final destination for the packet. Devices do not need to include the source or destination addresses in the packet because the destination address is always the device's address that received the packet and the source address is always the device's address that sent the packet. Note: multicast/broadcast destination addresses SHALL NOT be compressed.

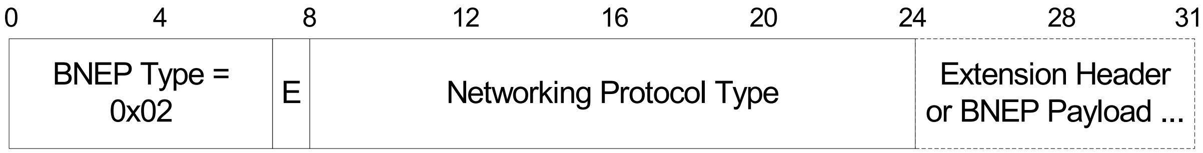


Figure 13: BNEP_COMPRESSED_ETHERNET Packet Type Header

BNEP Type: *Size: 7 Bit*

Value	Parameter Description
0x02	Seven bit Bluetooth Network Encapsulation Protocol

	Type value identifies the type of BNEP header contained in this packet. SHALL be set to BNEP_COMPRESSED_ETHERNET

Extension Flag (E): *Size: 1 Bit*

Value	Parameter Description
0x0 – 0x1	One bit extension flag that indicates if one or more extension headers follow the BNEP Header before the data payload if the data payload exists. Extension headers are defined in section 3 on page 39. If the extension flag is equal to 0x1 then one or more extension headers follows the BNEP header. If the extension flag is equal to 0x0 then the BNEP payload follows the BNEP header.

Networking Protocol Type: *Size: 2 Bytes*

Value	Parameter Description
0xXXXX	16 bit type field identifies the type of networking protocol contained in the payload. The values for this field are the same as defined for Ethernet types in [3]

2.8 BNEP_COMPRESSED_ETHERNET_SOURCE_ONLY Packet Type Header Format

The BNEP_COMPRESSED_ETHERNET_SOURCE_ONLY packet type header format is shown in Figure 14 on page 36. The header format is based on one of the compressed versions of the Ethernet header supported by BNEP. This packet type MAY be used to carry Ethernet packets to a device using BNEP, which is the final destination for that packet. Devices do not need to include the destination address in the packet, because the destination address of the BNEP packet is the same as the address corresponding to the L2CAP channel over which the packet is sent. Note: multicast/broadcast destination addresses SHALL NOT be compressed.

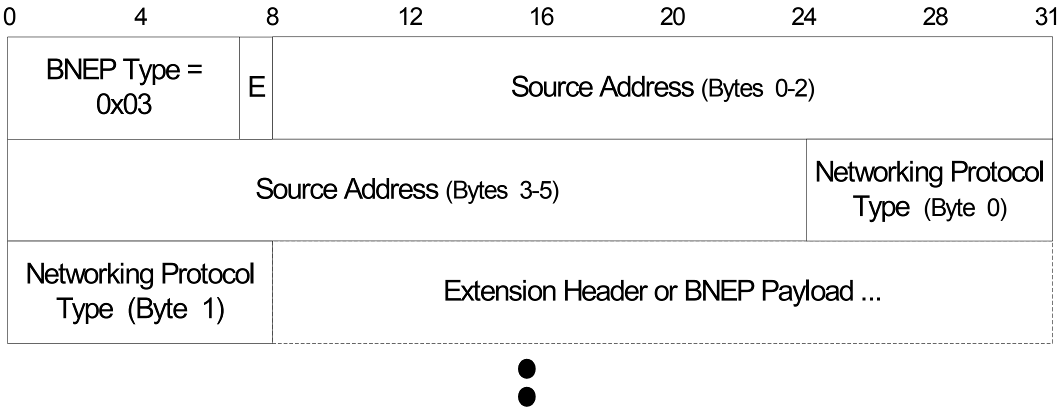

0
4
8
12
16
20
24
28
31
BNEP Type = 0x03
E
Source Address (Bytes 0-2)
Source Address (Bytes 3-5)
Networking Protocol Type (Byte 0)
Networking Protocol Type (Byte 1)
Extension Header or BNEP Payload ...

Figure 14: BNEP_COMPRESSED_ETHERNET_SOURCE_ONLY Packet Type Header

BNEP Type: *Size: 7 Bits*

Value	Parameter Description
0x03	Seven Bit Bluetooth Network Encapsulation Protocol Type value identifies the type of BNEP header contained in this packet. SHALL be set to BNEP_COMPRESSED_ETHERNET_SOURCE_ONLY

Extension Flag (E): *Size: 1 Bit*

Value	Parameter Description
0x0 – 0x1	One bit extension flag that indicates if one or more extension headers follow the BNEP Header before the data payload if the data payload exists. Extension headers are defined in section 3 on page 39. If the extension flag is equal to 0x1 then one or more extension headers follows the BNEP header. If the extension flag is equal to 0x0 then the BNEP payload follows the BNEP header.

Source Address: *Size: 6 Bytes*

Value	Parameter Description
0xXXXXXXXXXXXX	48 bit Bluetooth device address/IEEE address of the source of the BNEP packet/Ethernet frame contained in the payload.

Networking Protocol Type: *Size: 2 Bytes*

Value	Parameter Description

0xXXXX	16 bit type field identifies the type of networking protocol contained in the payload. The values for this field are the same as defined for Ethernet types in [3]

The source address MAY be an IEEE Ethernet address, if the actual source is an IEEE device and not a Bluetooth device.

2.9 BNEP_COMPRESSED_ETHERNET_DEST_ONLY Packet Type Header Format

The BNEP_COMPRESSED_ETHERNET_DEST_ONLY packet type header format is shown in Figure 15 on page 37. The header format is based on one of the compressed versions of the Ethernet header supported by BNEP. This packet type SHALL be used to carry Ethernet packets from a device using BNEP, which is the originator of that packet. Devices do not need to include the source address in the packet, because the source address can be determined from the L2CAP connection and which device sent the packet.

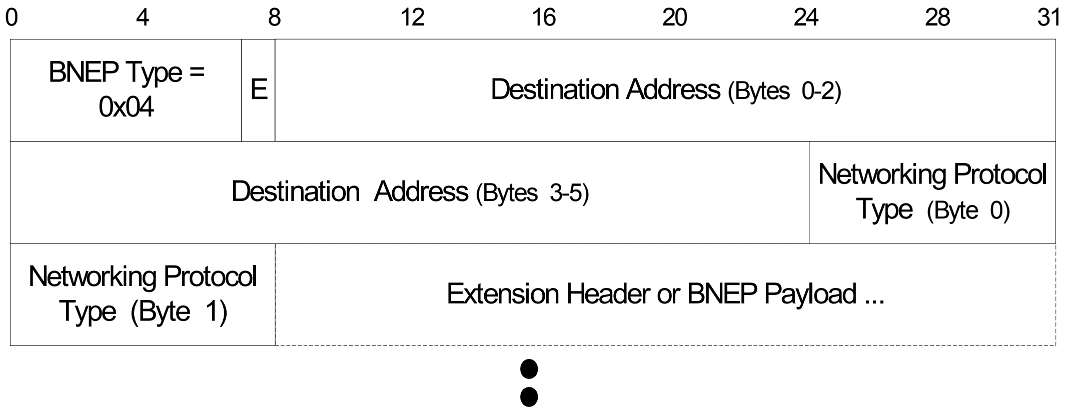

0
4
8
12
16
20
24
28
31
BNEP Type = 0x04
E
Destination Address (Bytes 0-2)
Destination Address (Bytes 3-5)
Networking Protocol Type (Byte 0)
Networking Protocol Type (Byte 1)
Extension Header or BNEP Payload ...

Figure 15: BNEP_COMPRESSED_ETHERNET_DEST_ONLY Packet Type Header

BNEP Type: *Size: 7 Bits*

Value	Parameter Description
0x04	Seven bit Bluetooth Network Encapsulation Protocol Type value identifies the type of BNEP header contained in this packet. SHALL be set to BNEP_COMPRESSED_ETHERNET_DEST_ONLY

Extension Flag (E): *Size: 1 Bit*

Value	Parameter Description
0x0 –	One bit extension flag that indicates if one or more

0x1	extension headers follow the BNEP Header before the data payload if the data payload exists. Extension headers are defined in section 3 on page 39. If the extension flag is equal to 0x1 then one or more extension headers follows the BNEP header. If the extension flag is equal to 0x0 then the BNEP payload follows the BNEP header.

Destination Address: *Size: 6 Bytes*

Value	Parameter Description
0xXXXXXXXXXXXX	48 bit Bluetooth device address/IEEE address of the destination of the BNEP packet/Ethernet frame contained in the payload.

Networking Protocol Type: *Size: 2 Bytes*

Value	Parameter Description
0xXXXX	16 bit type field identifies the type of networking protocol contained in the payload. The values for this field are the same as defined for Ethernet types in [3]

The destination MAY be an IEEE Ethernet address, if the actual destination is an IEEE device and not a Bluetooth device.

3 Extension Header

3.1 Extension Header Overview

Extension headers are used as optional headers in addition to the BNEP header. One or more extension headers MAY be included after the BNEP header and before BNEP payload and indicated by setting the BNEP Extension Flag to 1. If one or more extension headers are contained in the BNEP packet then, the Extension Flag in the BNEP SHALL be used to indicate that extension header follows the BNEP header. If more than one extension headers are contained in a BNEP packet then the extensions SHALL be ordered in ascending order by extension type. If one or more extension headers are sent out of ascending order then any extension headers after the out of order extension header MAY not be processed by the receiver. If an additional extension header follows the current extension header, then the extension flag in the extension header SHALL be used to indicate that an additional extension header follows. Extension headers SHALL be processed in the order they are present in the BNEP packet, after the BNEP header itself has been processed. If the Extension Type is not understood, then that extension header SHALL be skipped and remain as an extension header when the packet is forwarded. Unknown extension headers in data packets SHALL be forwarded irrespective of any network protocol or multicast filter settings and any local filtering policy (see section 2.6.4 on page 24). Note: The total size of all of the extensions and payload cannot exceed 1676 bytes[5].

0 4 8 12 16 20 24 28 31

Extension Type	E	Extension Length	Extension Payload ...

Figure 16 BNEP Extension Header Format

Extension Type: *Size: 7 Bits*

Value	Parameter Description
0x00 – 0x7F	One byte Bluetooth Network Encapsulation Protocol Extension Header Type value identifies the type of

[5] The size of 1676 is to preventing a packet from exceeding the minimum MTU (1691) for BNEP. This value is based on the minimum MTU - BNEP header (15 bytes). This maximum size is required to prevent exceeding the minimum MTU in the case where a BNEP packet containing a compressed header is forwarded and larger uncompressed header is needed to forward the packet to the correct destination.

	extension header contained in this packet. Values are defined in Table 6 on page 40.

Extension Flag (E): *Size: 1 Bit*

Value	Parameter Description
0x0 – 0x1	One bit extension flag that indicates if one or more extension headers follow the current extension header before the data payload if the data payload exists. Extension headers are defined in section 3 on page 39. If the extension flag is equal to 0x1 then one or more extension headers follow the current extension header. If the extension flag is equal to 0x0 then the BNEP payload follows the current extension header.

Extension Length: *Size: 1 Byte*

Value	Parameter Description
0x00 – 0xFF	One byte extension length that defines the number of bytes contain in the extension payload. This byte count does not include bytes used for the extension type or the extension length.

Extension Payload: *Size: Based on Extension Type*

Value	Parameter Description
0xXX	Based on the Extension Type

3.2 Extension Type Values

The Table 6 on page 40 defines the various extension type formats

Value	Extension Packet Type
0x00	BNEP_EXTENSION_CONTROL
0x01 – 0x7F	Reserved for future use

Table 6: Extension Types

3.3 BNEP_EXTENSION_CONTROL Packet Type Header Format

The BNEP_EXTENSION_CONTROL packet type header format is shown in Figure 17 on page 41. This packet type is mandatory to recognize and respond to accordingly, whereas the implied functionality to do command is optional and does not have to be supported by all devices. The BNEP Control Type and the Control Packet parameters are defined in section 2.6 on page 16. BNEP extension headers of type BNEP_EXTENSION_CONTROL are for the device with the direct connection only, and SHALL NOT be forwarded. BNEP_EXTENSION_CONTROL packet type headers in extension headers and BNEP_ CONTROL packets can be used interchangeably. Note: The size of each extension header is limited in length in terms of the maximum length of 255 bytes extension payload as well as the size of the entire BNEP packet, which is limited to the maximum MTU for the L2CAP connection. In addition, the total size of all of the extensions and payload cannot exceed 1676 bytes.

0 4 8 12 16 20 24 28 31

Extension Type = 0x00	E	Extension Length	BNEP Control Type	Control Packet based on Control Type ...

Figure 17: BNEP_EXTENSION_CONTROL Extension Header

4 Interpreting the IEEE 802.1Q Tag Header

4.1 IEEE 802.1Q Tag Header Support

The IEEE 802.1Q (also known as IEEE 802.1p) specification defines standardized frame prioritization tagging. In order to correctly determine the network protocol type, devices that implement the BNEP specification, SHALL be able to interpret the IEEE 802.1Q tag header and determine the actual network protocol type which follows it.

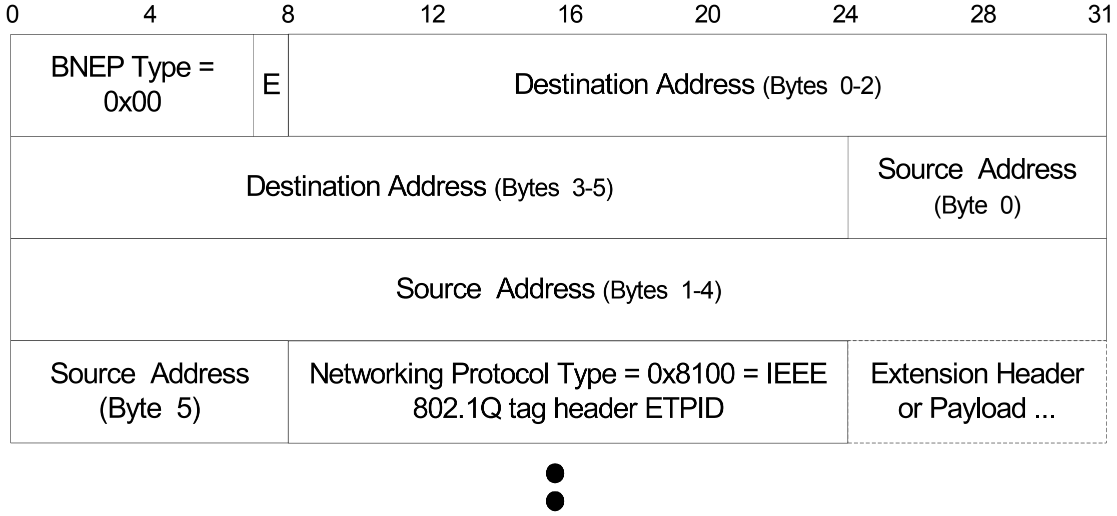

0
4
8
12
16
20
24
28
31
BNEP Type = 0x00
E
Destination Address (Bytes 0-2)
Destination Address (Bytes 3-5)
Source Address (Byte 0)
Source Address (Bytes 1-4)
Source Address (Byte 5)
Networking Protocol Type = 0x8100 = IEEE 802.1Q tag header ETPID
Extension Header or Payload ...

Figure 18: Example of a packet with an 802.1Q tag header protocol type contain in a BNEP header

The IEEE 802.1Q tag header, which has a size of 4 bytes, occupies the 2 byte Networking protocol type field (ETPID) in the BNEP header as well as additional two bytes (TCI) in the BNEP payload. The actual networking protocol type for this packet is located in the BNEP payload after the remaining part of the 802.1Q tag header, as shown in the Figure 19 on page 43 below, which is used for prioritization tagging. Devices MAY use the IEEE 802.1Q tag header to determine prioritization of the packets.

IEEE 802.1Q Tag Header TCI	Networking Protocol Type for Payload
Payload ...	

●
●

Figure 19: Ethernet payload for IEEE 802.1Q packet

5 Examples

5.1 Example Overview

The following examples are used to illustrate some of the possible ways to use BNEP.

5.2 Setting up a BNEP connection example

The following is a simple example for setting up a PAN connection from a PANU to a NAP using 128-bit UUIDs.

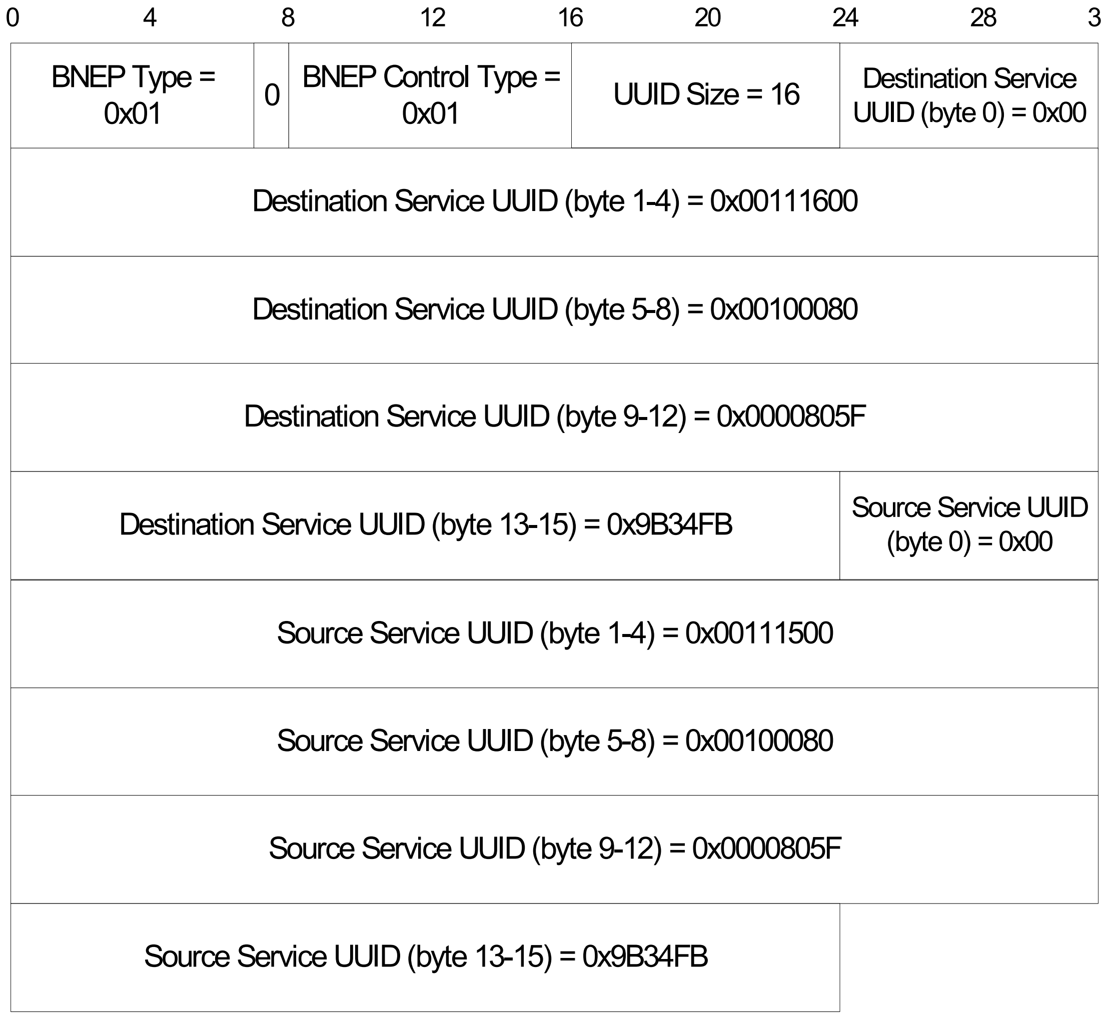

0
4
8
12
16
20
24
28
31
BNEP Type = 0x01
0
BNEP Control Type = 0x01
UUID Size = 16
Destination Service UUID (byte 0) = 0x00
Destination Service UUID (byte 1-4) = 0x00111600
Destination Service UUID (byte 5-8) = 0x00100080
Destination Service UUID (byte 9-12) = 0x0000805F
Destination Service UUID (byte 13-15) = 0x9B34FB
Source Service UUID (byte 0) = 0x00
Source Service UUID (byte 1-4) = 0x00111500
Source Service UUID (byte 5-8) = 0x00100080
Source Service UUID (byte 9-12) = 0x0000805F
Source Service UUID (byte 13-15) = 0x9B34FB

Figure 20: Sending an IP Packet Example

5.3 Sending an IP Packet Example

The following is a simple example in which an IP packet is sent using BNEP. The example illustrates an IPv4 packet sent from a device with 48 bit IEEE address of 00:AA:00:55:44:33 to a 48 bit Bluetooth address of 00:30:B7:45:67:89.

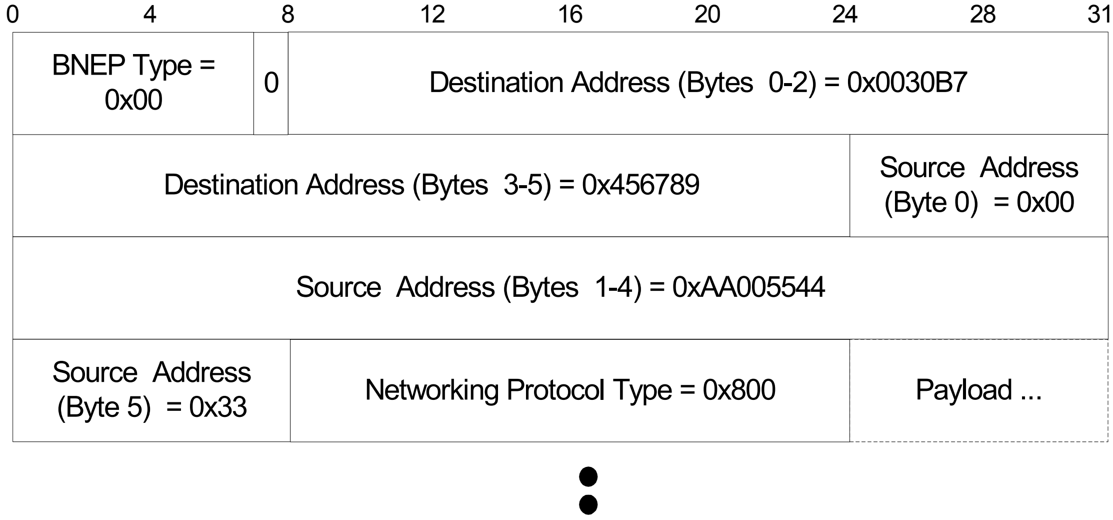

0
4
8
12
16
20
24
28
31
BNEP Type = 0x00
0
Destination Address (Bytes 0-2) = 0x0030B7
Destination Address (Bytes 3-5) = 0x456789
Source Address (Byte 0) = 0x00
Source Address (Bytes 1-4) = 0xAA005544
Source Address (Byte 5) = 0x33
Networking Protocol Type = 0x800
Payload ...

Figure 21: Sending an IP Packet Example

5.4 Sending an IP Packet between Bluetooth Master and Slave Example

The following is a simple example in which an IPv4 packet is sent using BNEP. In this example, the BNEP packet is sent from Device A, which is the master, to Device B, which is a slave of Device A. Device A has a 48 bit Bluetooth address of 00:AA:00:55:44:33 and Device B has a 48 bit Bluetooth Address of 00:30:B7:45:67:89.

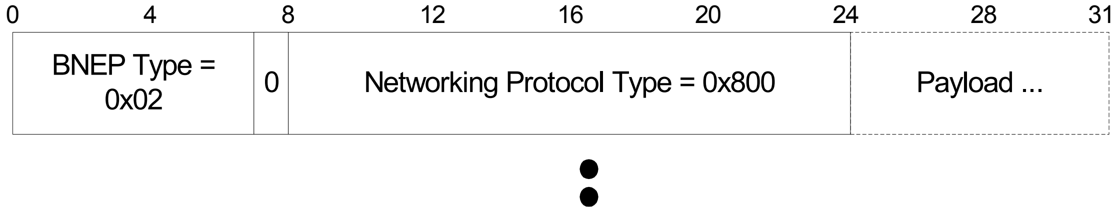


Figure 22: Sending an IP Packet between a Bluetooth Master and Slave Example

5.5 Setting Network Type Filter Examples

5.5.1 Enabling only IPv6

The following is a simple example for setting a filter to enable only IPv6.

0 4 8 12 16 20 24 28 31

<table>
<tr><td>BNEP Type =
0x01</td><td>0</td><td>BNEP Control Type =
0x03</td><td>List Length = 0x0004</td></tr>
<tr><td colspan="3">Network Protocol Type Range Start =
0x86DD</td><td>Network Protocol Type Range End =
0x86DD</td></tr>
</table>

Figure 23: Setting Filter to Enable only IPv6 Example

5.5.2 Enabling only IPv4

The following is a simple example for setting a filter to enable only IPv4 (including ARP).

0 4 8 12 16 20 24 28 31

<table>
<tr><td>BNEP Type =
0x01</td><td>0</td><td>BNEP Control Type =
0x03</td><td>List Length = 0x0008</td></tr>
<tr><td colspan="3">Network Protocol Type Range Start = 0x800</td><td>Network Protocol Type Range End =
0x0800</td></tr>
<tr><td colspan="3">Network Protocol Type Range Start = 0x0806</td><td>Network Protocol Type Range End =
0x0806</td></tr>
</table>

Figure 24: Setting Filter to enable only IPv4 and ARP

5.6 Setting Multicast Address Filter Examples

5.6.1 Enabling only IPv4 Multicast

The following is a simple example for setting a filter to enable only IPv4 IEEE multicast address (01-00-5E-20-00-00).

0 4 8 12 16 20 24 28 31

<table>
<tr><td>BNEP Type =
0x01</td><td>0</td><td>BNEP Control Type =
0x05</td><td>List Length = 0x000C</td></tr>
<tr><td colspan="4">Multicast Address Start #1 (Bytes 0-3) = 0x01005E20</td></tr>
<tr><td colspan="3">Multicast Address Start #1 (Bytes 4-5) =
0x0000</td><td>Multicast Address End #1 (Bytes 0-1) =
0x0100</td></tr>
<tr><td colspan="4">Multicast Address Start #1 (Bytes 2-5) = 0x5E200000</td></tr>
</table>

Figure 25: Setting Filter to Enable only IPv4 Multicast Example

5.6.2 Enabling only IPv6 Neighbor Discovery Multicast Address Range

The following is a simple example for setting a filter to enable only IPv6 multicast address (33-33-00-00-00-00 to 33-33-FF-FF-FF-FF)

0 4 8 12 16 20 24 28 31

<table>
<tr><td>BNEP Type =
0x01</td><td>0</td><td>BNEP Control Type =
0x05</td><td>List Length = 0x000C</td></tr>
<tr><td colspan="4">Multicast Address Start #1 (Bytes 0-3) = 0x33330000</td></tr>
<tr><td colspan="3">Multicast Address Start #1 (Bytes 4-5) =
0x0000</td><td>Multicast Address End #1 (Bytes 0-1) =
0x3333</td></tr>
<tr><td colspan="4">Multicast Address Start #1 (Bytes 2-5) = 0xFFFFFFFF</td></tr>
</table>

Figure 26: Setting Filter to enable only IPv6 Multicast Address Range

5.7 Sending an IP Packet with one Extension Header Example

The following is an example extension in which an IPv6 packet is sent using BNEP and a BNEP filter extension header is also included in the packet. The example illustrates an IPv6 packet sent from a device with 48 bit IEEE address of 00:AA:00:55:44:33 to a 48 bit Bluetooth address of 00:30:B7:45:67:89. The filter control message in the extension header is set to enable only IPv6.

<table>
<tr><th colspan="4">0</th><th colspan="4">4</th><th colspan="4">8</th><th colspan="4">12</th><th colspan="4">16</th><th colspan="4">20</th><th colspan="4">24</th><th colspan="3">28</th><th colspan="1">31</th></tr>
<tr><td colspan="7">BNEP Type = 0x00</td><td colspan="1">1</td><td colspan="24">Destination Address (Bytes 0-2) = 0x0030B7</td></tr>
<tr><td colspan="24">Destination Address (Bytes 3-5) = 0x456789</td><td colspan="8">Source Address (Btye 0) = 0x00</td></tr>
<tr><td colspan="32">Source Address (Bytes 1-4) = 0xAA005544</td></tr>
<tr><td colspan="8">Source Address (Byte 5) = 0x33</td><td colspan="16">Networking Protocol Type = 0x86DD</td><td colspan="7">Extension Type = 0x00</td><td colspan="1">0</td></tr>
<tr><td colspan="8">Extension Length = 0X07</td><td colspan="8">BNEP Control Type = 0x03</td><td colspan="16">List Length = 0x0004</td></tr>
<tr><td colspan="16">Network Protocol Type Range Start = 0x86DD</td><td colspan="16">Network Protocol Type Range End = 0x86DD</td></tr>
<tr><td colspan="32">Payload ...</td></tr>
</table>

Figure 27: Sending an IP Packet with one extension Example

5.8 Sending an IP Packet between Bluetooth Master and Slave with one Extension Header Example

The following is a simple example in which an IP packet is sent using BNEP using the BNEP_COMPRESSED_ETHERNET Packet Type Header format. In this example, the IPv4 packet is sent from Device A, which is the piconet master, to Device B, which is a slave of Device A. Device A has a 48 bit Bluetooth address of 00:AA:00:55:44:33 and Device B has a 48 bit Bluetooth Address of 00:30:B7:45:67:89. The filter control message in the extension header is set to enable only IPv4 (including ARP).

<table>
<tr><td colspan="4">0</td><td colspan="4">4</td><td colspan="4">8</td><td colspan="4">12</td><td colspan="4">16</td><td colspan="4">20</td><td colspan="4">24</td><td colspan="3">28</td><td colspan="1">31</td></tr>
<tr><td colspan="7">BNEP Type = 0x02</td><td colspan="1">1</td><td colspan="16">Networking Protocol Type = 0x800</td><td colspan="7">Extension Type = 0x00</td><td colspan="1">0</td></tr>
<tr><td colspan="8">Extension Length = 0X0B</td><td colspan="8">BNEP Control Type = 0x03</td><td colspan="16">List Length = 0x0008</td></tr>
<tr><td colspan="16">Network Protocol Type Range Start = 0x800</td><td colspan="16">Network Protocol Type Range End = 0x0800</td></tr>
<tr><td colspan="16">Network Protocol Type Range Start = 0x0806</td><td colspan="16">Network Protocol Type Range End = 0x0806</td></tr>
<tr><td colspan="32">Payload ...</td></tr>
</table>

Figure 28: Sending an IP Packet between a Bluetooth Master and Slave with Filter Extension Example

5.9 BNEP Control packet with one Extension Header Example

The following is an example in which a BNEP control packet containing a network protocol type filter set message has an extension header containing a multicast address filter set message. The example illustrates a control packet with a filter extension header.

<table>
<tr><th>0</th><th>4</th><th></th><th>8</th><th>12</th><th>16</th><th>20</th><th>24</th><th>28 31</th></tr>
<tr><td colspan="2">BNEP Type = 0x01</td><td>1</td><td colspan="2">BNEP Contol Type = 0x03</td><td colspan="4">List Length = 0x0004</td></tr>
<tr><td colspan="5">Network Protocol Type Range Start = 0x86DD</td><td colspan="4">Network Protocol Type Range End = 0x86DD</td></tr>
<tr><td colspan="2">Extension Type = 0x00</td><td>0</td><td colspan="2">Extension Length = 0X0F</td><td colspan="2">BNEP Control Type = 0x05</td><td colspan="2">List Length (Byte 0) = 0x00</td></tr>
<tr><td colspan="3">List Length (Byte 1) = 0x0C</td><td colspan="6">Multicast Address Start #1 (Bytes 0-2) = 0x333300</td></tr>
<tr><td colspan="7">Multicast Address Start #1 (Bytes 3-5) = 0x000000</td><td colspan="2">Multicast Address End #1 (Byte 0) = 0x33</td></tr>
<tr><td colspan="9">Multicast Address Start #1 (Bytes 1-4) = 0x33FFFFFF</td></tr>
<tr><td colspan="3">Multicast Address End #1 (Byte 5) = 0xFF</td></tr>
</table>

Figure 29: Sending an IP Packet with one extension Example

5.10 Sending an IP packet with .1Q tag header and one extension header Example

The example illustrates an IPv4 packet sent from a device with 48 bit IEEE address of 00:AA:00:55:44:33 to a 48 bit Bluetooth address of 00:30:B7:45:67:89. This BNEP packet has an IEEE 802.1Q tag header and has one extension header resetting the network filter list.

<table>
<tr><td colspan="4">0</td><td colspan="4">4</td><td colspan="4">8</td><td colspan="4">12</td><td colspan="4">16</td><td colspan="4">20</td><td colspan="4">24</td><td colspan="3">28</td><td colspan="1">31</td></tr>
<tr><td colspan="7">BNEP Type = 0x00</td><td colspan="1">1</td><td colspan="24">Destination Address (Bytes 0-2) = 0x0030B7</td></tr>
<tr><td colspan="24">Destination Address (Bytes 3-5) = 0x456789</td><td colspan="8">Source Address (Byte 0) = 0x00</td></tr>
<tr><td colspan="32">Source Address (Bytes 1-4) = 0xAA005544</td></tr>
<tr><td colspan="8">Source Address (Byte 5) = 0x33</td><td colspan="16">Networking Protocol Type = 0x8100 = IEEE 802.1Q Tag header ETPID</td><td colspan="7">Extension Type = 0x00</td><td colspan="1">0</td></tr>
<tr><td colspan="8">Extension Length = 0X03</td><td colspan="8">BNEP Control Type = 0x03</td><td colspan="16">List Length = 0x0000</td></tr>
<tr><td colspan="16">IEEE 802.1Q Tag Header TCI</td><td colspan="16">Payload -- Network Protocol Type = 0x0800</td></tr>
<tr><td colspan="32">Reset of the data payload ...</td></tr>
</table>

●
●

Figure 30: Sending an IP Packet with an IEEE 802.1Q Tag Header and one BNEP Control Extension Header Example

6 References

[1] Bluetooth Special Interest Group, “Bluetooth Personal Area Networking Profiles”, Specification of the Bluetooth System, Version 1.0, February 14, 2003

[2] Bluetooth Special Interest Group, “Bluetooth Core”, Specification of the Bluetooth System, Version 1.1, February 22, 2001

[3] http://www.iana.org/assignments/ethernet-numbers

[4] “The Ethernet – A Local Area Network”, Version 1.0 Digital Equipment Corporation, Intel Corporation, Xerox Corporation. September 1980.

[5] Internet Engineering Task Force, “A Standard for the Transmission of IP Datagrams over Ethernet Networks”, RFC 894.

[6] Internet Engineering Task Force, “Classical IP and ARP over ATM”, RFC 2225, April 1998

[7] Internet Engineering Task Force, “IPv4 over IEEE 1394”, RFC2734, December 1999.

[8] Bluetooth Special Interest Group, “Bluetooth Assigned Number”, http://www.Bluetooth.org/assigned-numbers.htm

[9] IEEE Std 802.1Q-1998, IEEE Standards for Local and Metropolitan Area Networks: Virtual Bridged Local Area Networks

7 Acronyms and Abbreviations

List of abbreviations necessary for the understanding BNEP.

Abbreviation or Acronym	Meaning
BNEP	Bluetooth Network Encapsulation Protocol
IP	Internet Protocol
L2CAP	Logical Link Control and Adaptation Protocol
MTU	Maximum Transmission Unit
OSI	Open Systems Interconnect (model)
PAN	Personal Area Network

Table 7: Acronyms and Abbreviation Table

8 List of Figures

9 List of Tables