Network Working Group
Request for Comments: 5285
Category: Standards Track

D. Singer
Apple, Inc.
H. Desineni
Qualcomm

July 2008

A General Mechanism for RTP Header Extensions

Status of This Memo

Abstract

This document provides a general mechanism to use the header extension feature of RTP (the Real-Time Transport Protocol). It provides the option to use a small number of small extensions in each RTP packet, where the universe of possible extensions is large and registration is de-centralized. The actual extensions in use in a session are signaled in the setup information for that session.

Table of Contents

1. Introduction 2
2. Requirements Notation 2
3. Design Goals 2
4. Packet Design 3
4.1. General 3
4.2. One-Byte Header 5
4.3. Two-Byte Header 6
5. SDP Signaling Design 7
6. Offer/Answer 9
7. BNF Syntax 12
8. Security Considerations 12
9. IANA Considerations 13
9.1. Identifier Space for IANA to Manage 13
9.2. Registration of the SDP extmap Attribute 14
10. Acknowledgments 15
11. Normative References 15

1. Introduction

The RTP specification [RFC3550] provides a capability to extend the RTP header. It defines the header extension format and rules for its use in Section 5.3.1. The existing header extension method permits at most one extension per RTP packet, identified by a 16-bit identifier and a 16-bit length field specifying the length of the header extension in 32-bit words.

This mechanism has two conspicuous drawbacks. First, it permits only one header extension in a single RTP packet. Second, the specification gives no guidance as to how the 16-bit header extension identifiers are allocated to avoid collisions.

This specification removes the first drawback by defining a backward-compatible and extensible means to carry multiple header extension elements in a single RTP packet. It removes the second drawback by defining that these extension elements are named by URIs, defining an IANA registry for extension elements defined in IETF specifications, and a Session Description Protocol (SDP) method for mapping between the naming URIs and the identifier values carried in the RTP packets.

This header extension applies to RTP/AVP (the Audio/Visual Profile) and its extensions.

2. Requirements Notation

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

3. Design Goals

The goal of this design is to provide a simple mechanism whereby multiple identified extensions can be used in RTP packets, without the need for formal registration of those extensions but nonetheless avoiding collision.

This mechanism provides an alternative to the practice of burying associated metadata into the media format bit stream. This has often been done in media data sent over fixed-bandwidth channels. Once this is done, a decoder for the specific media format is required to extract the metadata. Also, depending on the media format, the metadata may need to be added at the time of encoding the media so that the bit-rate required for the metadata is taken into account. But the metadata may not be known at that time. Inserting metadata at a later time can require a decode and re-encode to meet bit-rate requirements.

In some cases, a more appropriate, higher-level mechanism may be available, and if so, it should be used. For cases where a higher-level mechanism is not available, it is better to provide a mechanism at the RTP level than have the metadata be tied to a specific form of media data.

4. Packet Design

4.1. General

The following design is fit into the "header extension" of the RTP extension, as described above.

The presence and format of this header extension and its contents are negotiated or defined out-of-band, such as through signaling (see below for SDP signaling). The value defined for an RTP extension (defined below for the one-byte and two-byte header forms) is only an architectural constant (e.g., for use by network analyzers); it is the negotiation/definition (e.g., in SDP) that is the definitive indication that this header extension is present.

This specification inherits the requirement from the RTP specification that the header extension "is designed so that the header extension may be ignored". To be specific, header extensions using this specification MUST only be used for data that can safely be ignored by the recipient without affecting interoperability, and MUST NOT be used when the presence of the extension has changed the form or nature of the rest of the packet in a way that is not compatible with the way the stream is signaled (e.g., as defined by the payload type). Valid examples might include metadata that is additional to the usual RTP information.

The RTP header extension is formed as a sequence of extension elements, with possible padding. Each extension element has a local identifier and a length. The local identifiers may be mapped to a larger namespace in the negotiation (e.g., session signaling).

As is good network practice, data should only be transmitted when needed. The RTP header extension should only be present in a packet if that packet also contains one or more extension elements, as defined here. An extension element should only be present in a packet when needed; the signaling setup of extension elements indicates only that those elements may be present in some packets, not that they are in fact present in all (or indeed, any) packets.

Each extension element in a packet has a local identifier (ID) and a length. The local identifiers present in the stream MUST have been negotiated or defined out-of-band. There are no static allocations

of local identifiers. Each distinct extension MUST have a unique ID. The value 0 is reserved for padding and MUST NOT be used as a local identifier.

There are two variants of the extension: one-byte and two-byte headers. Since it is expected that (a) the number of extensions in any given RTP session is small and (b) the extensions themselves are small, the one-byte header form is preferred and MUST be supported by all receivers. A stream MUST contain only one-byte or two-byte headers: they MUST NOT be mixed within a stream. Transmitters SHOULD NOT use the two-byte form when all extensions are small enough for the one-byte header form.

A sequence of extension elements, possibly with padding, forms the header extension defined in the RTP specification. There are as many extension elements as fit into the length as indicated in the RTP header extension length. Since this length is signaled in full 32-bit words, padding bytes are used to pad to a 32-bit boundary. The entire extension is parsed byte-by-byte to find each extension element (no alignment is required), and parsing stops at the earlier of the end of the entire header extension, or, in one-byte headers, on encountering an identifier with the reserved value of 15.

In both forms, padding bytes have the value of 0 (zero). They may be placed between extension elements, if desired for alignment, or after the last extension element, if needed for padding. A padding byte does not supply the ID of an element, nor the length field. When a padding byte is found, it is ignored and the parser moves on to interpreting the next byte.

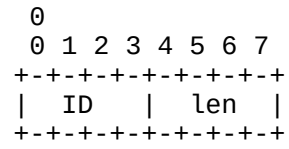
Note carefully that the one-byte header form allows for data lengths between 1 and 16 bytes, by adding 1 to the signaled length value (thus, 0 in the length field indicates 1 byte of data follows). This allows for the important case of 16-byte payloads. This addition is not performed for the two-byte headers, where the length field signals data lengths between 0 and 255 bytes.

Use of RTP header extensions will reduce the efficiency of RTP header compression, since the header extension will be sent uncompressed unless the RTP header compression module is updated to recognize the extension header. If header extensions are present in some packets, but not in others, this can also reduce compression efficiency by requiring an update to the fixed header to be conveyed when header extensions start or stop being sent. The interactions of the RTP header extension and header compression is explored further in [RFC2508] and [RFC3095].

4.2. One-Byte Header

In the one-byte header form of extensions, the 16-bit value required by the RTP specification for a header extension, labeled in the RTP specification as "defined by profile", takes the fixed bit pattern 0xBEDE (the first version of this specification was written on the feast day of the Venerable Bede).

Each extension element starts with a byte containing an ID and a length:

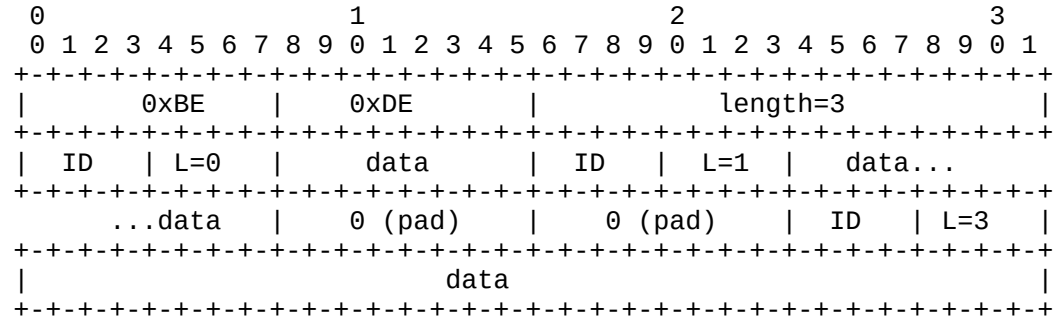


The 4-bit ID is the local identifier of this element in the range 1-14 inclusive. In the signaling section, this is referred to as the valid range.

The local identifier value 15 is reserved for future extension and MUST NOT be used as an identifier. If the ID value 15 is encountered, its length field should be ignored, processing of the entire extension should terminate at that point, and only the extension elements present prior to the element with ID 15 considered.

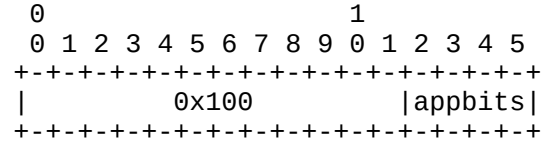
The 4-bit length is the number minus one of data bytes of this header extension element following the one-byte header. Therefore, the value zero in this field indicates that one byte of data follows, and a value of 15 (the maximum) indicates element data of 16 bytes. (This permits carriage of 16-byte values, which is a common length of labels and identifiers, while losing the possibility of zero-length values -- which would often be padded anyway.)

An example header extension, with three extension elements, some padding, and including the required RTP fields, follows:



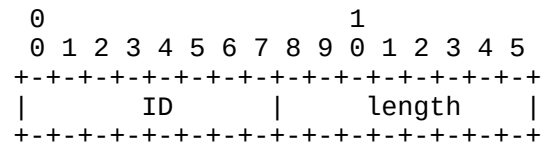
4.3. Two-Byte Header

In the two-byte header form, the 16-bit value required by the RTP specification for a header extension, labeled in the RTP specification as "defined by profile", is defined as shown below.



The appbits field is 4 bits that are application-dependent and may be defined to be any value or meaning, and are outside the scope of this specification. For the purposes of signaling, this field is treated as a special extension value assigned to the local identifier 256. If no extension has been specified through configuration or signaling for this local identifier value 256, the appbits field SHOULD be set to all 0s by the sender and MUST be ignored by the receiver.

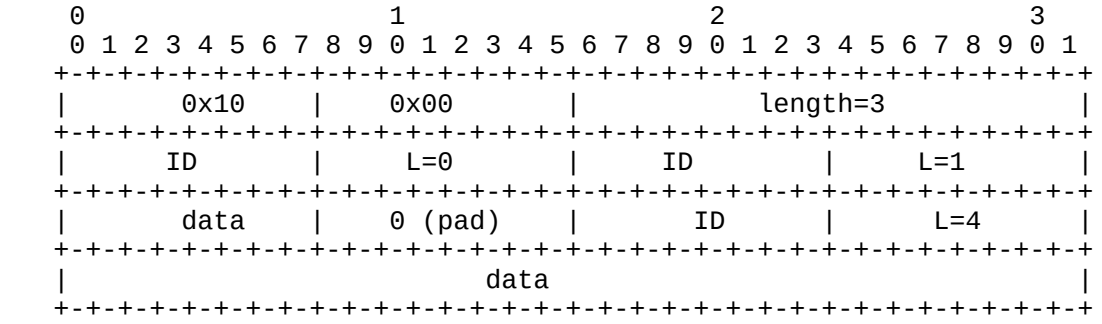
Each extension element starts with a byte containing an ID and a byte containing a length:



The 8-bit ID is the local identifier of this element in the range 1-255 inclusive. In the signaling section, the range 1-256 is referred to as the valid range, with the values 1-255 referring to extension elements, and the value 256 referring to the 4-bit field 'appbits' (above).

The 8-bit length field is the length of extension data in bytes not including the ID and length fields. The value zero indicates there is no data following.

An example header extension, with three extension elements, some padding, and including the required RTP fields, follows:



5. SDP Signaling Design

The indication of the presence of this extension, and the mapping of local identifiers used in the header extension to a larger namespace, MUST be performed out-of-band, for example, as part of a SIP offer/answer exchange using SDP. This section defines such signaling in SDP.

A usable mapping MUST use IDs in the valid range, and each ID in this range MUST be used only once for each media (or only once if the mappings are session level). Mappings that do not conform to these rules MAY be presented, for instance, during offer/answer negotiation as described in the next section, but remapping to conformant values is necessary before they can be applied.

Each extension is named by a URI. That URI MUST be absolute, and precisely identifies the format and meaning of the extension. URIs that contain a domain name SHOULD also contain a month-date in the form mmyyyy. The definition of the element and assignment of the URI MUST have been authorized by the owner of the domain name on or very

close to that date. (This avoids problems when domain names change ownership.) If the resource or document defines several extensions, then the URI MUST identify the actual extension in use, e.g., using a fragment or query identifier (characters after a '#' or '?' in the URI).

Rationale: the use of URIs provides for a large, unallocated space,

and gives documentation on the extension. The URIs are not required to be de-referencable, in order to permit confidential or experimental use, and to cover the case when extensions continue to be used after the organization that defined them ceases to exist.

An extension URI with the same attributes MUST NOT appear more than once applying to the same stream, i.e., at session level or in the declarations for a single stream at media level. (The same extension may, of course, be used for several streams, and may appear differently parameterized for the same stream.)

For extensions defined in RFCs, the URI used SHOULD be a URN starting "urn:ietf:params:rtp-hdrext:" and followed by a registered, descriptive name.

The registration requirements are detailed in the IANA Considerations section, below.

An example (this is only an example), where 'avt-example-metadata' is the hypothetical name of a header extension, might be:

urn:ietf:params:rtp-hdrext:avt-example-metadata

An example name not from the IETF (this is only an example) might be:

http://example.com/082005/ext.htm#example-metadata

The mapping may be provided per media stream (in the media-level section(s) of SDP, i.e., after an "m=" line) or globally for all streams (i.e., before the first "m=" line, at session level). The definitions MUST be either all session level or all media level; it is not permitted to mix the two styles. In addition, as noted above, the IDs used MUST be unique for each stream type for a given media, or for the session for session-level declarations.

Each local identifier potentially used in the stream is mapped to a string using an attribute of the form:

a=extmap:<value>["/"<direction>] <URI> <extensionattributes>

where <URI> is a URI, as above, <value> is the local identifier (ID) of this extension and is an integer in the valid range inclusive (0 is reserved for padding in both forms, and 15 is reserved in the one-byte header form, as noted above), and <direction> is one of "sendonly", "recvonly", "sendrecv", or "inactive" (without the quotes).

The formal BNF syntax is presented in a later section of this specification.

Example:

```
a=extmap:1 http://example.com/082005/ext.htm#ttime

a=extmap:2/sendrecv http://example.com/082005/ext.htm#xmeta short
```

When SDP signaling is used for the RTP session, it is the presence of the 'extmap' attribute(s) that is diagnostic that this style of header extensions is used, not the magic number indicated above.

6. Offer/Answer

The simple signaling described above may be enhanced in an offer/answer context, to permit:

- asymmetric behavior (extensions sent in only one direction),
- the offer of mutually exclusive alternatives, or
- the offer of more extensions than can be sent in a single session.

A direction attribute MAY be included in an extmap; without it, the direction implicitly inherits, of course, from the stream direction, or is "sendrecv" for session-level attributes or extensions of "inactive" streams. The direction MUST be one of "sendonly", "recvonly", "sendrecv", or "inactive". A "sendonly" direction indicates an ability to send; a "recvonly" direction indicates a desire to receive; a "sendrecv" direction indicates both. An "inactive" direction indicates neither, but later re-negotiation may make an extension active.

Extensions, with their directions, may be signaled for an "inactive" stream. It is an error to use an extension direction incompatible with the stream direction (e.g., a "sendonly" attribute for a "recvonly" stream).

If an offer or answer contains session-level mappings (and hence no media-level mappings), and different behavior is desired for each stream, then the entire set of extension map declarations may be moved into the media-level section(s) of the SDP. (Note that this specification does not permit mixing global and local declarations, to make identifier management easier.)

If an extension map is offered as "sendrecv", explicitly or implicitly, and asymmetric behavior is desired, the SDP may be modified to modify or add direction qualifiers for that extension. If an extension is marked as "sendonly" and the answerer desires to receive it, the extension MUST be marked as "recvonly" in the SDP answer. An answerer that has no desire to receive the extension or does not understand the extension SHOULD remove it from the SDP answer.

If an extension is marked as "recvonly" and the answerer desires to send it, the extension MUST be marked as "sendonly" in the SDP answer. An answerer that has no desire to, or is unable to, send the extension SHOULD remove it from the SDP answer.

Local identifiers in the valid range inclusive in an offer or answer must not be used more than once per media section (including the session-level section). A session update MAY change the direction qualifiers of extensions under use. A session update MAY add or remove extension(s). Identifiers values in the valid range MUST NOT be altered (remapped).

Note that, under this rule, the same local identifier cannot be used for two extensions for the same media, even when one is "sendonly" and the other "recvonly", as it would then be impossible to make either of them sendrecv (since re-numbering is not permitted either).

If a party wishes to offer mutually exclusive alternatives, then multiple extensions with the same identifier in the (unusable) range 4096-4351 may be offered; the answerer should select at most one of the offered extensions with the same identifier, and remap it to a free identifier in the valid range, for that extension to be usable.

Similarly, if more extensions are offered than can be fit in the valid range, identifiers in the range 4096-4351 may be offered; the answerer should choose those that are desired, and remap them to a free identifier in the valid range.

It is always allowed to place the offered identifier value "as is" in the SDP answer (for example, due to lack of a free identifier value in the valid range). Extensions with an identifier outside the valid range cannot, of course, be used. If required, the offerer or answerer can update the session to make space for such an extension.

Rationale: the range 4096-4351 for these negotiation identifiers is deliberately restricted to allow expansion of the range of valid identifiers in future.

Either party MAY include extensions in the stream other than those negotiated, or those negotiated as "inactive", for example, for the benefit of intermediate nodes. Only extensions that appeared with an identifier in the valid range in SDP originated by the sender can be sent.

Example (port numbers, RTP profiles, payload IDs and rtpmaps, etc. all omitted for brevity):

The offer:

```
a=extmap:1 URI-toffset
a=extmap:14 URI-obscure
a=extmap:4096 URI-gps-string
a=extmap:4096 URI-gps-binary
a=extmap:4097 URI-frametype
m=video
a=sendrecv
m=audio
a=sendrecv
```

The answerer is interested in receiving GPS in string format only on video, but cannot send GPS at all. It is not interested in transmission offsets on audio, and does not understand the URI-obscure extension. It therefore moves the extensions from session level to media level, and adjusts the declarations:

```
m=video
a=sendrecv
a=extmap:1 URI-toffset
a=extmap:2/recvonly URI-gps-string
a=extmap:3 URI-frametype
m=audio
a=sendrecv
a=extmap:1/sendonly URI-toffset
```

7. BNF Syntax

The syntax definition below uses ABNF according to [RFC5234]. The syntax element 'URI' is defined in [RFC3986] (only absolute URIs are permitted here). The syntax element 'extmap' is an attribute as defined in [RFC4566], i.e., "a=" precedes the extmap definition. Specific extensionattributes are defined by the specification that defines a specific extension name; there may be several.

```
extmap = mapentry SP extensionname [SP extensionattributes]

extensionname = URI

direction = "sendonly" / "recvonly" / "sendrecv" / "inactive"

mapentry = "extmap:" 1*5DIGIT ["/" direction]

extensionattributes = byte-string

URI = <Defined in RFC 3986>

byte-string = <Defined in RFC 4566>

SP = <Defined in RFC 5234>

DIGIT = <Defined in RFC 5234>
```

8. Security Considerations

This defines only a place to transmit information; the security implications of the extensions must be discussed with those extensions.

Care should be taken when defining extensions. Clearly, they should be solely informative, but even when the information is extracted, should not cause security concerns.

Header extensions have the same security coverage as the RTP header itself. When Secure Real-time Transport Protocol (SRTP) [RFC3711] is used to protect RTP sessions, the RTP payload may be both encrypted and integrity protected, while the RTP header is either unprotected or integrity protected. Therefore, it is inappropriate to place information in header extensions that cause security problems if disclosed, unless the entire RTP packet is protected by a lower-layer security protocol providing both confidentiality and integrity capability.

9. IANA Considerations

9.1. Identifier Space for IANA to Manage

The mapping from the naming URI form to a reference to a specification is managed by IANA. Insertion into this registry is under the requirements of "Expert Review" as defined in [RFC5226].

The IANA will also maintain a server that contains all of the registered elements in a publicly accessible space.

Here is the formal declaration required by the IETF URN Sub-namespace specification [RFC3553].

- Registry name: RTP Compact Header Extensions
- Specification: RFC 5285 and RFCs updating RFC 5285.
- Information required:

  A. The desired extension naming URI

  B. A formal reference to the publicly available specification

  C. A short phrase describing the function of the extension

  D. Contact information for the organization or person making the registration

  For extensions defined in RFCs, the URI is recommended to be of the form urn:ietf:params:rtp-hdrext:, and the formal reference is the RFC number of the RFC documenting the extension.

- Review process: Expert review is required. The expert review should check the following requirements:

  1. that the specification is publicly available;

  2. that the extension complies with the requirements of RTP and this specification, for extensions (notably, that the stream is still decodable if the extension is ignored or not recognized);

  3. that the extension specification is technically consistent (in itself and with RTP), complete, and comprehensible;

4. that the extension does not duplicate functionality in existing IETF specifications (including RTP itself), or other extensions already registered;

5. that the specification contains a security analysis regarding the content of the header extension;

6. that the extension is generally applicable, for example point-to-multipoint safe, and the specification correctly describes limitations if they exist; and

7. that the suggested naming URI form is appropriately chosen and unique.

- Size and format of entries: a mapping from a naming URI string to a formal reference to a publicly available specification, with a descriptive phrase and contact information.

- Initial assignments: none.

9.2. Registration of the SDP extmap Attribute

This section contains the information required by [RFC4566] for an SDP attribute.

- contact name, email address, and telephone number:

  D. Singer
  singer@apple.com
  +1 408-974-3162

- attribute name (as it will appear in SDP): extmap

- long-form attribute name in English: generic header extension map definition

- type of attribute (session level, media level, or both): both

- whether the attribute value is subject to the charset attribute: not subject to the charset attribute

- a one-paragraph explanation of the purpose of the attribute: This attribute defines the mapping from the extension numbers used in packet headers into extension names as documented in specifications and appropriately registered.

- a specification of appropriate attribute values for this attribute: see RFC 5285.

10. Acknowledgments

Both Brian Link and John Lazzaro provided helpful comments on an initial draft of this document. Colin Perkins was helpful in reviewing and dealing with the details. The use of URNs for IETF-defined extensions was suggested by Jonathan Lennox, and Pete Cordell was instrumental in improving the padding wording. Dave Oran provided feedback and text in the review. Mike Dolan contributed the two-byte header form. Magnus Westerlund and Tom Taylor were instrumental in managing the registration text.

11. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2508] Casner, S. and V. Jacobson, "Compressing IP/UDP/RTP Headers for Low-Speed Serial Links", RFC 2508, February 1999.

[RFC3095] Bormann, C., Burmeister, C., Degermark, M., Fukushima, H., Hannu, H., Jonsson, L-E., Hakenberg, R., Koren, T., Le, K., Liu, Z., Martensson, A., Miyazaki, A., Svanbro, K., Wiebke, T., Yoshimura, T., and H. Zheng, "RObust Header Compression (ROHC): Framework and four profiles: RTP, UDP, ESP, and uncompressed", RFC 3095, July 2001.

[RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, July 2003.

[RFC3553] Mealling, M., Masinter, L., Hardie, T., and G. Klyne, "An IETF URN Sub-namespace for Registered Protocol Parameters", BCP 73, RFC 3553, June 2003.

[RFC3711] Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)", RFC 3711, March 2004.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005.

[RFC4566] Handley, M., Jacobson, V., and C. Perkins, "SDP: Session Description Protocol", RFC 4566, July 2006.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

[RFC5234] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008.

Authors' Addresses

David Singer
Apple, Inc.
1 Infinite Loop
Cupertino, CA 95014
USA

Phone: +1 408 996 1010
EMail: singer@apple.com
URI: http://www.apple.com/quicktime

Harikishan Desineni
Qualcomm
5775 Morehouse Drive
San Diego, CA 92126
USA

Phone: +1 858 845 8996
EMail: hd@qualcomm.com
URI: http://www.qualcomm.com

Full Copyright Statement

Intellectual Property