

Registration Request for TI Vendor Extension RTCP XR Block Type – 5/30/07

1 Introduction

RFC 3611 (RTP Control Protocol Extended Reports) defines a new RTCP block type (XR), as shown in Table 1.

Table 1 - RTCP XR General Block Format

0					1					2					3																
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
V=2 P reserved					PT=XR=207					length																					
										SSRC																					
:					report [sub] blocks																				:						

Each RTCP XR report [sub] block has the format shown in Table 2 below. The **BT** field identifies the XR [sub] block type. There are seven types defined in the specification (an 8th has been subsequently reserved for a British Telecom defined block). These are presented in Table 3.

Table 2 - XR report [sub] blocks

0					1					2					3																
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
					BT					type-specific					block length																
:					type-specific block contents																				:						

Table 3 - Currently Assigned Block Types

BT	name
--	----
1	Loss RLE Report Block
2	Duplicate RLE Report Block
3	Packet Receipt Times Report Block
4	Receiver Reference Time Report Block
5	DLRR Report Block
6	Statistics Summary Report Block
7	VoIP Metrics Report Block

Texas Instruments (TI) would like to define a TI vendor extension XR block type. This is discussed next.

2 Proposed TI Vendor Extension Block Format

The TI vendor extension XR block format is in Type-Length-Value (TLV) format, and given in Table 4.

Table 4 - TI Vendor Extension XR Block Format

0	1	2	3
0 1 2 3 4 5 6 7 8 9	0 1 2 3 4 5 6 7 8 9	0 1 2 3 4 5 6 7 8 9	0 1
+-----+-----+-----+-----+			
BT= TBD	version (=0)	block length	
+-----+-----+-----+-----+			
TI Subtype	subtype length	values ..	
values..	TI Subtype	Subtype length	values ..
..		0	0
+-----+-----+-----+-----+			

The TI vendor extension block will consist of the standard 32 bit RTCP XR [sub] block header (consisting of block type, a type specific field that we use to carry a version number, and the block length), and a series of 0 or more Type/Length/Value (TLV) sequences. Each TLV sequence consists of an 8-bit subtype field, an 8-bit length field that defines the number of octets to follow for the value field, and 0 or more octets holding the values. The last TLV entry in the block shall consist of the **type = 0** and **length = 0-3** such that the sub-block can be padded out to 32-bit alignment. The proposed TI TLVs are as summarized in Table 5.

Table 5 - TI TLVs

TI Subtypes	Value Length (octets)	Purpose	Value Contents		
0	0-3	Terminate TLV sequence and pad	Values, if present should be 0		
1	10	Echo Quality Index	Field	Length (octets)	Format
			eqiAvg	2	Q15
			eqiMin	2	Q15
			eqiMax	2	Q15
			eqiInst	2	Q15
eqiCount	2	Unsigned 16 bit integer			
2	12	Packet Loss Redundancy Performance Statistics	Field	Length (octets)	Format
			nReceived	4	Unsigned integer
			nLost	4	Unsigned integer
			nRecover	4	Unsigned Integer
3	2 (16 bit)	Exception Record	Field (MSbit->LSbit)	Length (bits)	Format
			Direction	1	Binary (see below)
			Exception code	4	Binary code (see below)
			Exception sub_code	3	Binary code (see below)
			Exception duration percentage	8	Binary fraction (see below)
255		Reserved for expansion			

The details of the proposed TI block fields are described below:

- **Block Type:** To be assigned
- **Version field:** Set to 0 and incremented for each major change
- **Block length:** As per RFC 3611, this counts the number of 32 bit words in the TI block minus 1. Note that a block length of 0 is allowed as per RFC 3611.
- **TLV subtype 0:** Termination/padding subtype. This MUST be present as the last subtype if any other subtype is present. The length of the subtype SHALL be sufficient to pad the TI sub block out to a 32 bit boundary (see padding considerations below).
- **TLV subtype 1:** Echo Cancellation Equalization Statistics. This statistic subtype gives a numerical score (between 0 and 1.0 in Q15 format) indicating the quality of the echo signal as seen by the line echo canceller. A low value indicates a potential for echo being heard by the far side of the call, while a high value indicates a high probability of echo-free conversation. The statistic subtypes has four values, each of 2 octets (or a total of 8 octets for the subtype) and includes the following:
 - eqiAvg: Running average echo quality index in Q15 format, -1 (0xffff) => undefined value
 - eqiMin: Minimum echo quality index in Q15 format, -1 => undefined value
 - eqiMax: Maximum echo quality index in Q15 format, -1 => undefined value
 - eqiInst: Instantaneous echo quality index in Q15 format, -1 => undefined value
 - eqiCount: Count of EQI "samples" present in the EQI statistics above. eqiCount will typically increment by 1 every 2 seconds of the call, but may freeze for various reasons, for example if the far side is not talking.
*Note: to convert to floating point from Q15, use the following:
float = (float)(EQI Q15)/32768.0.*
- **TLV subtype 2:** Packet Level Redundancy Statistics. This statistics subtype gives information on the performance of packet level redundancy (e.g., as defined by RFC 2198) as seen by this receiver. It includes the following:
 - nReceived: cumulative count of RTP packets that have been received, not counting duplicates (e.g., does not count duplicates received as part of the redundancy scheme).
 - nLost: cumulative count of RTP packets that were lost by network and could not be recovered by the redundancy scheme.
 - nRecover: cumulative count of RTP packets that have been recovered by the redundancy scheme. This is the count of RTP packets that would have been lost if the redundancy scheme was not in place.

Usage (assuming $nReceived+nLost>0$):

- $nLost/(nReceived+nLost)$ gives the cumulative fraction lost with redundancy. Transmitters can use this receiver feedback as an indicator of the effectiveness of the redundancy level/technique being used.
- $nRecover/(nReceived+nLost)$ gives the cumulative fraction that would have been lost if redundancy was not in place. Transmitters can use this receiver feed back as an indicator of whether redundancy is really needed.

- **TLV subtype 3: Exception Record.** This statistics subtype gives the percentage of time during the call that an exception condition is active. Zero or more exception subtypes may be present, depending on the number of exceptions conditions that have been active. An exception subtype should not be present if the duration percentage (defined as the percentage of the call time where the exception was present) is zero. This way the size of the TI extension block can be minimized. The details of the 16-bit Exception Record are as follows:
 - Bit 15 (MSB): direction indicator. 0= exception in RTP stream from this device, 1= exception in RTP stream to this device)
 - Bit 14-11: Exception Code (see Table below)

Table 6 - Exception Codes

Code	Meaning
0x00	Reserved
0x01	One-Way Voice Detected
0x02	Choppy Voice Detected
0x03	Signal/Noise Problem Detected
0x04-0x0F	Reserved

- Bit 10-8: Exception Sub Code. This gives additional information on the exception type, such as possible cause (see Table below – undefined sub-codes should be treated as reserved)

(Code)	(SubCode)	Meaning
0xXX	0x00	Unknown cause
0x01	0x01	No packets received
0x01	0x02	No signal (packets present but no signal power)
0x01	0x03	Receive error (packets seen but rejected)
0x01	0x04	Remote end sees packets w/o signal power, but our transmit signal level is within range
0x01	0x05	No packets received by us but Remote end sender report indicates they are sending
0x01	0x06	Our TX signal level is out of range
0x02	0x01	Packet loss
0x02	0x02	Packet Jitter
0x03	0x01	Signal Level Hot

0x03	0x02	Noise Level Hot
0x03	0x04	Signal Level Low

- Bits 7-0: Exception Duration percentage information expressed as a fixed point, unsigned number with binary point at left edge of field (like RTCP RR fraction lost). This gives the percentage of the call that the exception condition has been present. To convert **dur** to a decimal percentage, use the following formula: $float\ per = [(float)(dur)/256.0] * 100.0$.
- **TLV subtype 255:** reserved for future expansion
- **TLV subtype length.** This counts the number of octets in the subtype, not including the subtype or this field itself. A subtype length of 0 is permitted.
- **Subtype values:** 0 or more octets as discussed above. TI subtype values SHOULD be 16-bit or 32-bit aligned (i.e., the TLV subtype length SHOULD be always even). Sixteen and 32-bit values are transmitted in network byte order.

Padding Considerations:

The individual TI subtype TLVs do not need to be padded. However the TI [sub] block MUST be padded to a 32-bit boundary (so as to allow subsequent XR [sub] blocks that might be present to be properly aligned). This is accomplished through the inclusion of TLV subtype 0 and the end of the [sub] block. The following are valid combinations of subtype 0 TLV for this purpose:

Table 7 – Padding Combinations

		Last TLV word				
(i)	..	x	x	x	x	0 2 0 0
(ii)	..	x	x	x	0	3 0 0 0 //SHOULD NOT occur
(iii)	..	x	x	0	0	
(iv)	..	x	0	1	0	//SHOULD NOT occur

In addition, the entire XR block containing the TI vendor extension [sub] type MAY need to be padded as per RFC 3611.

Example:

The following is a hex dump of an XR block containing the proposed TI vendor extension [sub] block, with TI subtypes 1 and 2. The TI [sub] block type is assumed to be **20 hex** in the example.

Table 8 - Example TI vendor extension [sub] block

0x80	0xCF	0x00	0x09	// block header (length = 9 +1 32 bit words)
0x01	0x02	0x03	0x04	// ssrc 01020304
0x20	0x00	0x00	0x07	//start of TI vendor extension [sub] block (len = 7+1 32 bit words)
0x01	0x0a	0x7f	0xfa	// start of TI subtype 1 (EQI)
0x7f	0xfa	0x7f	0xfa	
0x7f	0xfa	0x00	0x50	// end of TI subtype 1
0x02	0xc0	0x00	0x00	//sstart of TI subtype 2 (RED stats)
0x00	0x00	0x00	0x00	
0x00	0x00	0x00	0x00	
0x00	0x00	0x00	0x00	//end & padding

SDP Considerations:

RFC 3611 defines an SDP attribute, **a=rtcp-xr: params**, that is used to signal that XR can be sent and received and pass information on each XR block type that can be supported. We define a new parameter for this attribute:

- **ti-piqua**

An example of this SDP usage in the case where an XR block will contain the voip metrics and the TI vendor extension sub-blocks would be:

a=rtcp-xr: voip-metrics ti-piqua

Devices SHOULD be configurable as to whether they send this TI vendor extension block if the remote end does not indicate support. There is benefit in transmitting the TI block even if the remote device does not support it, because network management devices that snoop the RTCP packet flow may still be able to process this block. RTCP XR compliant devices should be able to ignore unknown XR block types, so there is no real harm in sending the TI vendor extension block other than potentially wasted bandwidth (~ 40 bytes).