**IEEE P802.15**

**Wireless Personal Area Networks**

|  |  |
| --- | --- |
| Project | IEEE P802.15 Working Group for Wireless Personal Area Networks (WPANs) |
| Title | Comments and Resolution Work Items re 802.15.4e Draft D04 |
| Date | May 10, 2011 |
| Source | René StruikStruik Security Consultancy723 Carlaw AvenueToronto ON Canada M4K 3K8 | E-mail: rstruik.ext@gmail.comPhone US: +1 (415) 690-7363Phone Toronto: +1 (416) 466-8084Cell: +1 (647) 867-5658 |
| Re: | IEEE 802.15.4 TGe/Draft D04  |
| Abstract | This document provides suggested resolutions of (mostly) security-related comments that were submitted with letter ballot on 802.15.4e/D04. |
| Purpose | Assist with fixing/improving the (mostly) security-relevant portions of the draft 802.15.4e standard.  |
| Notice | This document has been prepared to assist the IEEE P802.15. It is offered as a basis for discussion and is not binding on the contributing individual(s) or organization(s). The material in this document is subject to change in form and content after further study. The contributor(s) reserve(s) the right to add, amend or withdraw material contained herein. |
| Release | The contributor acknowledges and accepts that this contribution becomes the property of IEEE and may be made publicly available by P802.15. |

IMPORTANT NOTE:

The suggested resolutions in this submission are relative to security-related comments on IEEE 802.15.4e/D04. Some of these are address comments that were already discussed in the context of comment resolution on 802.15.4e/D02 and 802.15.4e/D03, but whose resolution did not make it to the draft D03 or the current draft (D04). This document is an update of 11/201r0, which was relative to 802.15.4e/D03, so as to make suggested resolutions relative to baseline documents 802.15.4e/D04 and 802.15.4i/D07. Note that 802.15.4e/D03 was still defined as amendment of 802.15.4-2006, whereas 802.15.4e/D04 is relative to 802.15.4i/D07. Our update consists mostly of a shift of references to TG4i/D07 (since changes suggested in 11/201r0 were mostly not incorporated along the transition path from 802.15.4e Draft D03 to Draft D04).

**Editorial note RS:**

**Unless stated otherwise, changes are relative to 802.15.4i/D07. Changes relative to 802.15.4e/D04 are indicated as such.**

**6.3.1 MCS-DATA.request**

***Change the MCPS-DATA.request primitive as follows:***

MCPS-DATA.request (

SrcAddrMode,

DstAddrMode,

DstPANId,

DstAddr,

msduLength,

msdu,

msduHandle,

TxOptions,

SecurityLevel,

FrameCounterMode,

KeyIdMode,

KeySource,

KeyIndex

)

***Insert the following element in Table 46, directly below the SecurityLevel element***

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Range | Description |
| *FrameCounterMode* | Integer | 0x00-0x07 | Representation of the frame counter to be used (see 7.6.2.2.3). This parameter is ignored if theSecurityLevel parameter is set to 0x00. |

***Editorial note*** – The FrameCounterMode parameter should be added to all outgoing primitives with security-related attributes SecurityLevel, KeyIdMode, KeySource, KeyIndex and text modified accordingly (first occurrence with 6.3.1 noted above). With the MLME-Start.request (6.2.12.1), one should add both the CoordRealignFrameCounterMode, and the BeaconFrameCounterMode.

Impacted clauses of 802.15.4i/D07 are enumerated below (similar changes to TG4e-only primitives):

MCPS-DATA.indication (6.3.3), MLME-Associate.request (6.2.2.1), MLME-Associate.indication (6.2.2.2), MLME-Associate.response (6.2.2.3), MLME-Associate.confirm (6.2.2.4), MLME-Disassociate.request (6.2.3.1), MLME-Dissociate.indication (6.2.3.2), MLME-GTS.request (6.2.6.1), MLME-GTS.indication (6.2.6.3), MLME-Orphan.indication (6.2.7.1), MLME-Orphan.response (6.2.7.2), MLME-Scan.request (6.2.10.1), MLME-Comm-Status.indication (6.2.4.2), MLME-Start.request (6.2.12.1), MLME-Sync-Loss.indication (6.2.13.2), MLME-Poll.request (6.2.14.1).

**7.5.8.1.4 Frame counter**

***Change 7.5.8.1.4 as follows:***

The ~~4-octet~~ 6-octet frame counter is used to provide replay protection and semantic security of the cryptographic building block used for securing outgoing frames. The frame counter is included in each secured frame and is one of the elements required for the unsecuring operation at the recipient(s). The frame counter is incremented each time an outgoing frame is secured, as described in the outgoing frame security procedure (see 7.5.8.2.1). When the frame counter is used, it is scaled-down to a 4 ½ octet value for usage with a particular key. When ~~the~~ this scaled frame counter reaches its maximum value of 0xfffffffff, the associated keying material can no longer be used, thus requiring ~~all~~ this particular key~~s~~ ~~associated with the device~~ to be updated. This provides a mechanism for ensuring that the keying material for every frame is unique and, thereby, provides for sequential freshness. The frame counter may be included with the secured frame in a compressed format, thus allowing bandwidth savings in scenarios where the full frame counter value can be faithfully reconstructed from the compressed frame counter value, as contained in an incoming frame, and locally maintained status information.

**7.2.1**

***Change the first sentence of 7.2.1 as follows:***

The inputs to this procedure are the frame to be secured and the SecurityLevel, FrameCounterMode, KeyIdMode, KeySource, and KeyIndex parameters from the originating primitive or automatic request PIB attributes.

***Change the following steps in 7.2.1:***

***Insert the following step right before Step a):***

a0) The procedure shall set *macCurrentTime* to the current absolute device time (measured in 16 kHz granularity). If

this procedure fails, the procedure shall set this parameter to the integer value zero.

***Replace Step d) by the following two steps, to be inserted right after Step e):***

***Editorial note – the order has to be changed, since Step f1 below depends on availability of the KeyDescriptor.***

f1) The procedure shall set the frame counter to the maximum value of the *macFrameCounter*and the

*macCurrentTime* attributes and shall set the scaled frame counter to the difference between the

frame counter and the KeyOffset element of the KeyDescriptor.

f2) If the scaled frame counter value is negative, the procedure shall return with a status of KEY\_ERROR. If the scaled frame counter is greater than or equal to 0xfffffffff, the procedure shall return with a status of COUNTER\_ERROR.

***Change Step h) as follows:***

***Insert the following step right after Step h)2):***

2a) The frame counter mode subfield of the security control field shall be set to the Frame-

CounterMode parameter.

***Change Step g)3) as follows:***

3) The Frame Counter field shall be set to the representation of the frame counter indicated by the Frame Counter mode parameter.

If the FrameCounterMode is set to 0x00, the FrameCounter subfield of the auxiliary security header shall be set to value of the frame counter (modulo 232).

If the FrameCounterMode is set to 0x01, the FrameCounter subfield of the auxiliary security header shall be set to value of the frame counter (modulo 256).

If the FrameCounterMode is set to 0x02, the FrameCounter subfield of the auxiliary security header shall be set to the empty string.

***Change the first sentence of Step g)1) as follows:***

The procedure shall then use *aExtendedAddress*, the scaled frame counter, the SecurityLevel parameter, and the ~~k~~Key Element of the KeyDescriptor to produce the secured frame according to the CCM\* transformation process defined in 7.2.4.

**7.2.3 Incoming frame security procedure**

***Change the following steps in 7.2.3:***

***Insert the following step right before Step a):***

a0): The procedure shall set *macCurrentTime* to the current absolute device time (measured in 16 kHz granularity). If this procedure fails, the procedure shall set this parameter to the integer value zero.

***Change the first sentence of Step c) as follows:***

c) If the Security Enabled subfield of the Frame Control field of the frame to be unsecured is set to one, the procedure shall set the security level, the frame counter mode, and the key identifier mode to the corresponding subfields of the Security Control field of the auxiliary security header of the frame to be unsecured and shall set the key source and key index to the corresponding subfields of the Key Identifier field of the auxiliary security header of the frame to be unsecured, if present.

***Replace Step l) by the following four steps (in order):***

l1) The procedure shall reconstruct the frame counter from the FrameCounter field of the auxiliary security header of the frame to be unsecured, according to the representation indicated by the FrameCounterMode of that field, as follows:

* If the FrameCounterMode is set to 0x00, the frame counter shall be set to the smallest integer N such that N = FrameCounter (modulo 232) and such that N is greater than or equal to the FrameCounter element of the DeviceDescriptor.
* If the FrameCounterMode is set to 0x01, the frame counter shall be set to the smallest integer N such that N = FrameCounter (modulo 256) and such that N is greater than or equal to the FrameCounter element of the DeviceDescriptor.
* If the FrameCounterMode is set to 0x02, the frame counter shall be set to the smallest integer N such that N = DSN (modulo 256) and such that N is greater than or equal to the FrameCounter element of the DeviceDescriptor.

Editorial Note RS – add language on detection of stale frames! (and interval boundaries checks)

l2) shall set the scaled frame counter to the difference between the resulting frame counter and the KeyOffset element of the KeyDescriptor.

l3) If the scaled frame counter value is negative, the procedure shall set the unsecured frame to be the frame to be unsecured and return with a status of KEY\_ERROR.

l4) If the scaled frame counter is greater than or equal to 0xfffffffff, the procedure shall set the unsecured frame to be the frame to be unsecured and return with a status of COUNTER\_ERROR.

***Change the first sentence of Step o) as follows:***

o) The procedure shall then use the ExtAddress element of the DeviceDescriptor, the scaled frame counter, the security level, and the Key element of the KeyDescriptor to produce the unsecured frame according to the CCM\* inverse transformation process described in the security operations, as described in 7.3.5).

**7.5 Security-related MAC PIB Attributes**

Change the macFrameCounter element of Table 59 as follows:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Attribute | Identifier | Type | Range | Description | Default |
| *macFrameCounter* | 0x77 | Integer | 0x000000000000-0xffffffffffff | The outgoing frame counter for this device | 0x000000000000 |

Insert the following elements to Table 59 as follows:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Attribute | Identifier | Type | Range | Description | Default |
| *macAutoRequestFrameCounterMode* | TBD | Integer | 0x00-0x07 | The frame counter mode used for automatic data requests. This attribute is invalid if the *macAutoRequestSecurityLevel* attribute is set to 0x00. | 0x00 |
| *macCurrentTime* | TBD | Integer | 0x000000000000-0xffffffffffff | The most recent absolute device time, as measured just prior to performing an incoming or outgoing frame security procedure, in 16 kHz accuracy.  | 0x000000000000 |

*Editorial note –* The *macAutoRequestFrameCounterMode* attribute should be placed right below the *macAutoRequestSecurityLevel* attribute; the other attribute should be inserted at the end of the table.

Insert the KeyOffset attribute in Table 60 as follows:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Range | Description |
| KeyOffSet | Integer | 0x000000000000-0xffffffffffff | The start time counter value for this key, prior to which this key shall not be used. A value of 0 indicates that there are no time restrictions as to when to start using this key. |

*Editorial note –* The KeyOffset attribute should be placed right before the KeyDeviceList attribute.

Insert the AcknowledgementFrameIdentifier attribute in Table 61 as follows:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Range | Description |
| AcknowledgementFrameIdentifier | Integer | 0x00-TBD | See Table {TBD}. |

*Editorial note –* The AcknowledgementFrameIdentifier attribute should be placed right after the CommandFrameIdentifier attribute.

Insert the AcknowledgementFrameIdentifier attribute in Table 62 as follows:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Range | Description |
| AcknowledgementFrameIdentifier | Integer | 0x00-TBD | See Table {TBD}. |

*Editorial note –* The AcknowledgementFrameIdentifier attribute should be placed right after the CommandFrameIdentifier attribute.

Change the FrameCounter element of Table 63 as follows:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Range | Description |
| *macFrameCounter* | Integer | 0x000000000000-0xffffffffffff | The incoming frame counter of the device in this DeviceDescriptor. This value is used to ensure sequential freshness of frames. |

**7.4 Auxiliary security header**

Change Figure 69 as follows:

|  |  |  |
| --- | --- | --- |
| **Octets: 1** | **0/1/4/6** | **0/1/5/9** |
| Security Control | Frame Counter | Key Identifier |

**7.4.1 Security control field**

Change Figure 70 as follows:

|  |  |  |
| --- | --- | --- |
| **Bits: 0–2** | **3–4** | **5-7** |
| Security Level | Key Identifier Mode | Frame Counter Mode |

Insert after 7.4.1.2 (Key Identifier Mode subfield) the following subclause:

{This replaces 802.15.4e/Draft D4, Clause 7.4.1.3, 7.4.1.4, 7.4.1.5}

**7.6.2.2.3 Frame Counter Mode subfield**

The Frame Counter Mode subfield is 3 bits in length and is used to indicate the particular representation of the frame counter field (see 7.6.2.3). The frame counter mode subfield shall be set to one of the nonreserved values listed in Table {98a}.

Table{58a}– Values of the frame counter mode

|  |  |  |  |
| --- | --- | --- | --- |
| Frame counter mode | Frame Counter Mode subfield b2 b1 b0 | Description | Frame Counter subfield length |
| 0x00 | ‘000’ | Frame counter is determined from the 4-octet Frame Counter subfield of the auxiliary security header in conjunction with *macCurrentTime*. This mode corresponds to the frame counter as used in 802.15.4-2006, where the *macCurrentTime* attribute is not present (“stuck at zero”). | 4 |
| 0x01 | ‘001’ | Frame counter is determined from the 1-octet Frame Counter subfield of the auxiliary security header in conjunction with *macCurrentTime*. | 1 |
| 0x02 | ‘010’ | Frame Counter is determined from the DSN subfield of the frame header in conjunction with *macCurrentTime*. | 0 |
| 0x03-0x07 | − | Reserved | − |

**7.4.2.1.1 Frame counter field**

***Change 7.4.2.1.1 as follows:***

**7.4.2.1.1** The frame counter field is 0, 1, 4, or 6 octets in length, according to the representation specified by the frame counter mode subfield of the auxiliary security header (see 7.4), and ~~represents~~ indicates the *macFrameCounter* attribute of the originator of a protected frame. It is used to provide semantic security of the cryptographic mechanism used to protect a frame and to offer replay protection.

**7.3.2 CCM\* Nonce**

 ***Change Figure 68 as follows:***

|  |  |  |
| --- | --- | --- |
| **Octets: 8** | **4** | **1** |
| Source address | Frame counter | Security field |

***Insert the following figure right after Figure 68:***

|  |  |  |
| --- | --- | --- |
| **Bits: 0-3** | **4** | **5-7** |
| Ext. frame counter | Set to 0 | Security level |

Figure {68a} – Security field

***Change the paragraph right after Figure 68 as follows:***

The source address field shall be set to the extended address *aExtendedAddress* of the device originating the frame. ~~, the frame counter to the value of the respective field in the auxiliary security header (see 7.6.2),~~The frame counter field and the ext. frame counter field shall be set so that the right-concatenation of the extended frame counter field and the frame counter field, as bit-strings, has the same integer value as the scaled frame counter, as reconstructed from the frame counter subfield of the auxiliary security header (see 7.4), taking into account the frame counter mode subfield of the security control field hereof (see 7.4.1.3)~~, and the~~ The security level field shall be set to the security level.

**APPENDIX A – Comments René Struik on IEEE 802.15.4e, Draft D4**

Note RS: cross-reference to comment CID in database 11/368r01 added.

1. (CID#46)(E) General: most of my comments on Draft D3 (CID#6-CID#18) were rejected (witness 11/205r9), since \*claimed\* to be carry-overs from prior comments and already incorporated with Draft D3. Problem is: they have \*not\* been implemented with Draft D3 at all. While I do appreciate little errors in comment resolution spreadsheets, the massive scale of rejecting almost all my comments, is remarkable and misrepresents the maturity of the TG4e draft to the wider 802.15 community. I was not consulted on any of these verdicts, but most comments have dragged on from draft to draft, but now with incorrect representation of fact during comment resolution. **Suggested remedy:** I suggest people familiar with 802.15 process should look into this. Right now, it seems that the motion to move forward with recirc (Slide 13 and Slide 15 of 11/293) was supported by information with lots of lapses of correctness.
2. (CID#47)(TR) 802.15.4i/D7, §5.1.6.1, p. 39: Timing information: cryptographic processing assumes that timing information related to time of actual sending of frame is available. This information needs to be propagated ‘down the stack’. Add language to the effect that the current time *macCurrentTime* will be stored.Add corresponding parameters to all MAC and PHY primitives. **Suggested remedy:**  Implement accordingly. Please also cf. 11/201 and update 11/362. Note RS: this was previously included in 08/828 {technique #1}, 08/848, 08/849 and supposed to have made it to Draft D03 of TG4e, but did not yet, despite of 11/205r9/CID#6 saying so.
3. (CID#48)(TR) 802.15.4i/D07, §5.1.6.2, p. 40: Timing information: cryptographic processing assumes that timing information related to time of actual receipt of frame is available. This information needs to be propagated ‘up the stack’. Add language to the effect that the current time *macCurrentTime* will be stored. Add corresponding parameters to all MAC and PHY primitives. **Suggested remedy:**  Implement accordingly. Please also cf. 11/201 and update 11/362. Note RS: this was previously included in 08/828 {technique #1}, 08/848, 08/849 and supposed to have made it to Draft D03 of TG4e, but did not yet, despite 15-11-025-09/CID#7 saying so.
4. (CID#49)(TR) 802.15.4i/D07, §6.4.2, p. 125 ff, Table 52: Timing information: cryptographic processing assumes that timing information related to time of actual sending and receipt of frame is available. This information needs to be propagated ‘up and down the stack’. Add language to the effect that the current time *macCurrentTime* will be stored, suggested as a 48-bit integer, with 14-bit granularity per second (as an alternative, one may store this information in the PIB table for security-related attributes – cf. 802.15.4i/D07, §7.5, p. 141, Table 59, ff). Add corresponding parameters to all MAC and PHY primitives. This parameter is required for timeliness security property to work, so is in scope. **Suggested remedy:**  Implement accordingly. Please also cf. 11/201 and update 11/362. Note RS: this was previously included in 08/828 {technique #1}, 08/848, 08/849 and supposed to have made it to Draft D03 of TG4e, but did not yet, despite 15-11-025-09/CID#8 saying so.
5. (CID#50)(TR) §6.3.1, p. 148, l. 16-20, 802.15.4i/D07, §7.2.1, p. 130, l. 29-31: Add FrameCounterMode parameter; add *macCurrentTime* parameter. Adapt MAC sublayer service primitives (§6.2-6.3) accordingly. **Suggested remedy:**  Implement accordingly. Please also cf. 11/201 and update 11/362. Note RS: this was previously included in 08/828 {technique #1}, 08/848, 08/849 and supposed to have made it to Draft D03 of TG4e, but did not yet, despite 15-11-025-09/CID#9 saying so.(It is ignored for everything except TSCH.)
6. (CID#51)(TR) 802.15.4i/D07, §7.2.1, p. 130, Step d): Obtain *macFrameCounter* attribute from *macCurrentTime* parameter and locally maintained info (so-called frame counter conversion), such that value never decreases. This correlates frame counter to time parameter (while still allowing devices without time on board to function properly. **Suggested remedy:**  Implement accordingly. Please also cf. 11/201 and update 11/362. Note RS: this was previously included in 08/828 {technique #1}, 08/848, 08/849 and supposed to have made it to Draft D03 of TG4e, but did not yet, despite 15-11-025-09/CID#10 saying so.
7. (CID#52)(TR) 802.15.4i/D07, §7.2.1, p. 131, Step f), iii): Set frame counter to representation of *macFrameCounter* compliant with FrameCounterMode parameter. **Suggested remedy:**  Implement accordingly. Please also cf. 11/201 and update 11/362. Note RS: this was previously included in 08/828 {technique #1}, 08/848, 08/849 and supposed to have made it to Draft D03 of TG4e, but did not yet, despite 15-11-025-09/CID#11 saying so.
8. (CID#53)(TR) 802.15.4/i/D07, §7.2.1, p. 131, Step h, l. 32: Mute frame header fields in the protected frame compliant with representation mode parameter settings. **Suggested remedy:**  Implement accordingly. Please also cf. 11/201 and update 11/362. Note RS: this was previously included in 08/828 {technique #1}, 08/848, 08/849 and supposed to have made it to Draft D03 of TG4e, but did not yet, despite 15-11-025-09/CID#12 saying so.
9. (CID#54)(TR) 802.15.4i/D07, §7.2.3, p. 132, l. 25-26: Add FrameCounterMode parameter; add *macCurrentTime* parameter. Adapt MAC sublayer service primitives (§6.2-6.3) accordingly. **Suggested remedy:**  Implement accordingly. Please also cf. 11/201 and update 11/362. Note RS: this was previously included in 08/828 {technique #1}, 08/848, 08/849 and supposed to have made it to Draft D03 of TG4e, but did not yet, despite 15-11-025-09/CID#13 saying so.
10. (CID#55)(TR) 802.15.4i/D07, §7.2.3, p. 133, Step l), l. 43-45: Reconstruct frame counter from representation hereof in auxiliary security header, taken into account the frame counter mode and locally maintained info as to the current time (so-called frame counter conversion). **Suggested remedy:**  Implement accordingly. Please also cf. 11/201 and update 11/362. Note RS: this was previously included in 08/828 {technique #1}, 08/848, 08/849 and supposed to have made it to Draft D03 of TG4e, but did not yet, despite 15-11-025-09/CID#14 saying so.
11. (CID#56)(TR) 802.15.4i/D07, §7.2.3, p. 133, Step m), l. 46-48: Correlate the frame counter and the current time *macCurrentTime*; reject frame if these differ by more than a set amount (presumably, because the frame was stale).This provides timeliness and facilitates significant reduction of the storage cost for frame counters on recipient’s device (since one does not need to store these for stale communications). **Suggested remedy:**  Implement accordingly. Please also cf. 11/201 and update 11/362. Note RS: this was previously included in 08/828 {technique #1}, 08/848, 08/849 and supposed to have made it to Draft D03 of TG4e, but did not yet, despite 15-11-025-09/CID#15 saying so.
12. (CID#57)(TR) 802.15.4i/D07, §7.2.3, p. 133, Step n), l. 49-52: After this step, reconstruct muted frame header fields from information in the received frame and locally maintained status information. **Suggested remedy:**  Implement accordingly. Please also cf. 11/201 and update 11/362. Note RS: this was previously included in 08/828 {technique #1}, 08/848, 08/849 and supposed to have made it to Draft D03 of TG4e, but did not yet, despite 15-11-025-09/CID#16 saying so.
13. (CID#58)(TR) §7.5, Table 59, p. 166; 802.15.4i/D07, §7.5, Table 59, p. 142, l. 5-7: represent frame counter as 6-octet integers, rather than 4- or 5-octet integers. Adapt overflow checks with security processing of outgoing frames (§7.2.1) and of incoming frames (§7.2.3) accordingly (i.e., replace 0xffffffff by 0xffffffffffff). This implements time-correlated frame counters. **Suggested remedy:**  Implement accordingly. Please also cf. 11/201 and update 11/362. Note RS: this was previously included in 08/828 {technique #1}, 08/848, 08/849 and supposed to have made it to Draft D03 of TG4e, but did not yet, despite 15-11-025-09/CID#17 saying so. (With Draft D04, there is some lip service on p. 166, but not complete, nor at right place at all.)
14. (CID#59)(TR) 802.15.4i/D07, §7.2.1: Proper implementation of overhead reduction techniques assumes that one can reconstruct muted fields upon receipt, thereby requiring shared conventions between sender and recipient(s) as to which representation of data elements to choose (if there is more than one option). This is currently still insufficiently specified (also with §7.2.3 and with §5.1.6.1 and §5.1.6.2). **Suggested remedy:**  Implement accordingly. Please also cf. 11/201 and update 11/362. Note RS: this was previously included in 08/828 {technique #1}, 08/848, 08/849 and supposed to have made it to Draft D03 of TG4e, but did not yet, despite 15-11-025-09/CID#18 saying so.
15. (CID#60)(TR) 802.15.4i/D07, §7.2.1: Processing of secured acknowledgement frames is still not properly supported. As an example, security policy checks are not defined for secured ACK frames. Add language on how to compress auxiliary security header and other header fields. **Suggested remedy:**  Implement accordingly. Please also cf. 11/201 and update 11/362. Note RS: this was previously included in 08/828 {technique #1}, 08/848, 08/849 and supposed to have made it to Draft D03 of TG4e, but did not yet, despite 15-11-025-09/CID#21 saying so.