IEEE P802.11
Wireless LANs

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|

|  |
| --- |
| Fragment Flushing BlockAckReq |
| Date: 2017-08-09 |
| Author(s): |
| Name | Affiliation | Address | Phone | email |
| Matthew Fischer | Broadcom |  |  | Matthew.fischer@broadcom.com |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

 |

Abstract

Proposed language to create a mechanism for a transmitter to command a flush of incomplete MSDUs within the RX Buffer of a receiver using a new variant of the BAR frame. The motivation for this feature is to allow simplified implementation of transmitter side fragmentation.

Changes are referenced to TGax D6.0.

**REVISION NOTES:**

**R0**:

initial

**R1**:

Add CID

Update text changes from D1.4 to D2.2 reference, and importantly, updating the baseline TGmd Draft text that is used to the latest TGmd draft, which is D1.0

**R2**:

Changed to LB233 from LB230, i.e. removed LB230 CID, added CIDs from LB233

Update text changes to D3.2

Update doc references

**R3**:

Update text changes to D3.3

Update doc references

**R4**:

Update text changes to D4.0

Update CIDs to LB238 CID 21587, 21050

Update doc references

**R5**:

Add CID 21588

Update doc references

**R6**:

Change to LB244 CID 24267

Update to D5.1

Update doc references

**R7**:

Change to LB247 CID 24267

Update to D6.0

Update doc references

**R8**:

Fix clause v page number in CID table

Update doc references

**R9**:

Add a capability bit – the transmitter of the Fragment Flushing BAR needs to know if the recipient will perform the desired action

Update doc references

**END OF REVISION NOTES**

Interpretation of a Motion to Adopt

A motion to approve this submission means that the editing instructions and any changed or added material are actioned in the TGax Draft. This introduction is not part of the adopted material.

***Editing instructions formatted like this are intended to be copied into the TGax Draft (i.e. they are instructions to the 802.11 editor on how to merge the text with the baseline documents).***

***TGax Editor: Editing instructions preceded by “TGax Editor” are instructions to the TGax editor to modify existing material in the TGax draft. As a result of adopting the changes, the TGax editor will execute the instructions rather than copy them to the TGax Draft.***

**CIDs**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| CID | Commenter | Clause | Page | Comment | Proposed Change | Resolution (Proposed) |
| 24267 | Matthew Fischer | 326.06 | 26.3.1 | Need a mechanism to allow the transmitter of fragments to re-partition an MSDU for which some fragments have been transmitted. This requires a fragment flush command | Add a mechanism to allow the transmitter of fragments to re-partition an MSDU by creating the ability to signal a fragment flush command to its recipient STA. See 11-18-0218 | Revise - TGax editor to make changes as shown in 11-18/0218r9 that are marked with CID 24267 which generally agree with the commenter’s suggestion. |

**Discussion:**

This document proposes a mechanism for transmitter-commanded RX BUFFER Flush of incomplete MSDUs. It is selective in that the receipient only flushes incomplete MSDUs up to and including the indicated end sequence number. No receiver window move occurs. No BlockAck is transmitted in response. MSDUs and fragments of MSDUs that are not covered by the SEQ number range or which are completely assembled in the buffer are unaltered.

**Proposed Changes to TGax D6.0:**

**9.4.2.26 Extended Capabilities element**

***TGax editor: within TGax D6.0, add another row to Table 9-153 – Extended Capabilities field as shown:***

**Table 9-153—Extended Capabilities field**

|  |  |  |
| --- | --- | --- |
| **Bit** | **Information** | **Notes** |
| <ANA> | Fragment Flushing Support | A STA sets the Fragment Flushing Support field to 1 when dot11FragmentFlushingOptionActivated is true and sets it to 0 otherwise. |

**9.3.1.7 BlockAckReq frame format**

**9.3.1.7.1 Overview**

***TGax editor: within TGax D6.0, modify the rows of Table 9-28 – BlockAckReq frame variant encoding that are shown below, noting that the header row is shown for orientation purposes and noting that one row is a newly inserted row:***

**Table 9-28 – BlockAckReq frame variant encoding**

|  |  |
| --- | --- |
| **BAR Type** | **BlockAckReq frame variant** |
| 7 | Fragment Flushing BlockAckReq **(#24267)** |
| 8-9 | Reserved **(#24267)** |

***TGax editor: within TGax D6.0, insert a new subclause and editing instruction, as shown:***

***Insert a new subclause as follows:***

**9.3.1.7.6a Fragment Flushing BlockAckReq frame format (#24267)**

The TID\_INFO subfield of the BAR Control field of the Fragment Flushing BlockAckReq frame is reserved.

The BAR Information field of the Fragment Flushing BlockAckReq frame comprises a single Fragment Flushing TID Bitmap and one or more Fragment Flushing End Sequence Control subfields, as shown in Figure 9-31aa (BAR Information field format (Fragment Flushing BlockAckReq)), where the number of Fragment Flushing End Sequence Control subfields present is equal to the number of bits in the Fragment Flushing TID Bitmap that are equal to 1.

|  |  |  |
| --- | --- | --- |
| Octets: | 2 | 2 |
|  | Fragment Flushing TID Bitmap | Fragment Flushing End Sequence Control |

Repeat for each TID

**Figure 9-31aa – BAR Information field format (Fragment Flushing BlockAckReq)**

The Fragment Flushing TID Bitmap subfield is 16 bits in length and is a bitmap that indicates for which TID values a Fragment Flushing End Sequence Control subfield is present in the BlockAckReq frame. The lowest numbered bit of the Fragment Flushing TID Bitmap corresponds to the TID value of 0, the second lowest numbered bit of the Fragment Flushing TID Bitmap corresponds to the TID value of 1, and each successively higher-numbered bit in the bitmap corresponds to the next higher TID value. If a bit in the bitmap corresponding to a TID is equal to 1, then a Fragment Flushing End Sequence Control subfield is present for that TID; Otherwise, no Fragment Flushing End Sequence Control subfield is present for that TID.The Fragment Flushing End Sequence Control subfields that are present are ordered in monotonically increasing order per their corresponding TID values.

The Fragment Flushing End Sequence Control subfield indicates which contents are to be flushed by the recipient from the Rx Buffer for the TID associated with the Fragment Flushing End Sequence Control subfield value. The format of the Fragment Flushing End Sequence Control subfield is shown in Figure 9-31bb – Fragment Flushing End Sequence Control subfield.

|  |  |  |  |
| --- | --- | --- | --- |
|  | B0 | B1 B3 | B4 B15 |
|  | Flush All Fragments | Reserved | Fragment Flushing End Sequence Number |
| Bits: | 1 | 3 | 12 |

**Figure 9-31bb – Fragment Flushing Ending Sequence Control subfield**

The Flush All Fragments subfield indicates if the recipient of the Fragment Flushing BlockAckReq is instructed to flush all incomplete MSDU and A-MSDUs in the Rx Buffer. If the Flush All Fragments subfield is 1, the recipient is instructed to flush all incomplete MSDUs and A-MSDUs in the Rx Buffer. Otherwise, the recipient uses the Fragment Flushing End Sequence Number subfield to determine which incomplete MSDUs and A-MSDUs to flush.

The Fragment Flushing End Sequence Number subfield is used to indicate the last incomplete MSDU Rx Buffer entry that is to be flushed at the recipient. If the Flush All Fragments subfield has the value 1, then the Fragment Flushing End Sequence Number subfield is reserved, otherwise, the Fragment Flushing End Sequence Number subfield contains the value of the last incomplete MSDU Rx Buffer entry that is to be flushed at the recipient.

***TGax editor: in the appropriate location within TGax Draft D6.0, insert the following text and editing instruction:***

**10.25.6.3 Scoreboard context control during full-state operation**

***Change the 4th paragraph as shown:***

c) For each received BlockAckReq frame that is not a Fragment Flushing BlockAckReq frame and that is related with a specific full state operation HT-immediate block ack agreement that is not a protected block ack agreement, the block acknowledgement record for that agreement is modified as follows, where SSN is the value from the Starting Sequence Number subfield of the received BlockAckReq frame: **(#24267)**

***TGax editor: in the appropriate location within TGax Draft D6.0, insert the following text and editing instructions:***

**10.25.6.4 Scoreboard context control during partial-state operation**

***Change the 7th paragraph as shown:***

d) For each received BlockAckReq frame that is not a Fragment Flushing BlockAckReq frame and that is related with a specific partial-state operation HTimmediate block ack agreement that is not a protected block ack agreement, when no temporary record for the agreement related with the received frame exists at the time of the receipt of the frame, a temprory block acknowledgement record is created as follows, where *SSN* is the starting value of the Sequence Number subfield of the received BlockAckReq frame: **(#24267)**

***Change the 8th paragraph as shown:***

e) For each received BlockAckReq frame that is not a Fragment Flushing BlockAckReq frame and that is related with a specific partial-state operation HTimmediate block ack agreement that is not a protected block ack agreement, when a temporary record for the agreement related with the received frame exists at the time of the receipt of the frame, the temprory block acknowledgement record for that agreement is modified in the same manner as the acknowledgement record for a full-state agreement described in 10.24.7.3 (Scorebored context control during full-state operation). **(#24267)**

***TGax editor: in the appropriate location within TGax Draft D6.0, insert the following text and editing instructions:***

**10.25.6.5 Generation and transmission of BlockAck frames by an HT STA, DMG STA or S1G STA**

***Change the 1st paragraph as shown:***

Except when operating within a PSMP exchange, a STA that receives a PPDU that contains a BlockAckReq frame that is not a Fragment Flushing BlockAckReq frame and in which the Address 1 field matches its MAC address during either full-state operation or partial-state operation shall transmit a PPDU containing a BlockAck frame that is separated on the WM by a SIFS from the PPDU that elicited the BlockAck frame as a response. A STA that receives an A-MPDU that contains one or more QoS Data frames with Implicit BAR ack policy during either full-state operation or partial-state operation shall transmit a PPDU containing a BlockAck frame that is separated on the WM by a SIFS from the PPDU that elicited the BlockAck frame as a response. **(#24267)**

***Change the last paragraph as shown:***

If a BlockAckReq frame that is not a Fragment Flushing BlockAckReq frame is received and no matching partial state is available, the recipient shall send a null BlockAck frame in which the bitmap is set to all 0s. **(#24267)**

If a BlockAckReq frame that is a Fragment Flushing BlockAckReq frame is received, the recipient shall send an ACK frame in response. **(#24267)**

***TGax editor: in the appropriate location within TGax Draft D6.0, insert the following text and editing instruction which modify the text of 10.25.6.6.3 Operation for each received BlockAckReq as shown, noting that the bullet items 2) and 3) are not replacaing the existing 2) and 3) but are new 2) and 3) that should appear between the existing 1) and 2), and therefore, the baseline existing bullets starting with 2) need to be renumbered:***

**10.25.6.6.3 Operation for each received BlockAckReq**

***Change the third paragraph as shown:***

1. In a block ack agreement that is not a protected block ack agreement, if the received BlockAckReq is not a Fragment Flushing BlockAckReq frame, set *WinStartB* = *SSN.* See 10.25.8 (Protected block ack agreement) for a protected block ack agreement.
2. If the received BlockAckReq is a Fragment Flushing BlockAckReq frame and the Flush All Fragments subfield corresponding to this specific HT-immediate block ack agreement is equal to 0, then discard all incomplete MSDUs and incomplete A-MSDUs with sequence numbers that are equal to or lower than the Fragment Flushing End Sequence Number corresponding to this specific HT-immediate block ack agreement.
3. If the received BlockAckReq is a Fragment Flushing BlockAckReq frame and the Flush All Fragments subfield corresponding to this specific HT-immediate block ack agreement is equal to 1, then discard all incomplete MSDUs and incomplete A-MSDUs corresponding to this specific HT-immediate block ack agreement **(#24267)**

***TGax editor: in the appropriate location within the TGax Draft, insert the following text and editing instruction:***

**10.25.6.7 Originator’s behavior**

***Insert a new paragraph after the 7th paragraph as shown:***

The originator may send a Fragment Flushing BlockAckReq frame to force the recipient to discard incomplete MSDUs if it has received an Extended Capability element with the Fragment Flushing Support bit equal to 1 from the recipient. The originator may restart the fragmentation of an MSDU for a block ack greement after receiving successful acknowledgement of a Fragment Flushing BlockAckReq frame for that agreement that commanded the recipient to flush that MSDU. **(#24267)**

**TGax Editor: *Add a new MIB variable in C.3 MIB detail within the dot11StationConfigEntry group as shown:***

**C.3 MIB Detail**

dot11FragmentFlushingOptionActivated OBJECT-TYPE **(#24267)**

SYNTAX TruthValue

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This is a capability variable. Its value is determined by device capabilities.

This attribute, when true, indicates that the STA implementation acting as a recipient for a BA agreement flushes incomplete MSDUs and A-MSDUs from its receive buffer if it receives a Fragment Flushing variant BlockAckReq from its originator and indicates that when acting as an originator of a BA agreement, is capable of transmitting a Fragment Flushing variant BlockAckReq. The capability is disabled, otherwise."

DEFVAL { false }

::= { dot11StationConfigEntry <XX>}

**End of proposed changes.**