IEEE P802.11
Wireless LANs

|  |
| --- |
| EDMG Flow control extension |
| Date: 2017-07-02 |
| Author(s): |
| Name | Affiliation | Address | Phone | email |
| Solomon Trainin | Qualcomm |  |  | strainin@qti.qualcomm.com |
| Alecsander Eitan | Qualcomm |  |  | aeitan@qti.qualcomm.com |
| Assaf Kasher | Qualcomm |  |  | akasher@qti.qualcomm.vom |
| Oren Kedem | Intel |  |  | oren.kedem@intel.com |
| Carlos Cordeiro | Intel |  |  | carlos.cordeiro@intel.com |
| Ran Mor | Intel |  |  | ran.mor@intel.com |
| Rob Sun | Huawei |  |  | Rob.Sun@huawei.com |
| Joe Andonieh  | Peraso |  |  | joe@perasotech.com |
| Yuichi Morioka | Sony |  |  | Yuichi.Morioka@sony.com |

EDMG Flow control text

**9.3.1.9 BlockAck frame format**

**9.3.1.9.1 Overview**

P12L1 [1]

Modify Figure 9-33 as follows:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|   | B0 | B1 B4 | B5 B9 | B9 | B10 | B11 | B12 B15 |
|   | BA AckPolicy | BA Type | Reserved | No\_MemKept | Memory configuration tag | ManagementACK | TID INFO |
| Bits: | 1 | 4 | 5 | **1** | **1** | 1 | 4 |

*Insert new text after paragraph that starts with: “The GCR field indicates whether the BlockAck frame…”*

The No\_Mem\_kept field indicates that Free Memory space provided in last RBUFCAP delivery in frame exchange sequence is not kept at start of next sequence if the field is set to 1 and it is kept otherwise. Value of the No\_Mem\_kept is set to 0 at successive Block Ack agreement establishment.

The Memory\_configuratiuon\_tag field indicates one of two Recipient memory configurations as indicated in Memory config tag field in Recipient memory configuration subelement (9.4.2.258)

Value of the Memory\_configuratiuon\_tag is set to 0 at successive Block Ack agreement establishment.

**9.3.1.9.7 EDMG Compressed BlockAck variant**

P13L8 in [1]

The RBUFCAP field is defined ~~in 9.3.1.9.5~~.**Table 9-y1**

**Table 9-y1 RBUFCAP encoding for EDMG Compressed BlockAck variant**

|  |  |  |
| --- | --- | --- |
| **RBUFCAP Value** | **RBUFCAP Value name** | **Definition** |
| 0 | Unlimited\_space | Space of the Recipient memory to receive A-MPDUs with length that is not less than indicated by Maximum A-MPDU Length Exponent (Table 1 —A-MPDU Parameters field definition) |
| 0xFF | Zero\_space | No place in the Recipient memory |
| 1-0xFe | RBUFCAP(RBUF\_Unit\_Size)  | Size of Recipient memory that Originator may use to transmit MPDUs to the Recipient is measured in RBUFCAP units of RBUF\_Unit\_Size  |

RBUF\_Unit\_Size is defined in 9.4.2.258

**9.4.2 Elements**

**9.4.2.1 General**

P15L13 [1]

**Table 9-77—Element IDs *(continued)***

|  |  |  |  |
| --- | --- | --- | --- |
| **Element** | **Element ID** | **Element ID extension** | **Extensible** |
| EDMG Flow Control extension configuration  | 255 | ANA | Yes |

**9.4.2.258 EDMG Flow control extension configuration element**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Element ID | Length | Element ID extension | RBUFCAP | Advanced Recipient Memory length exponent | Optional subelements  |
| 1 | 1 | 1 | 1 | 1 | Variable  |

**Figure 9-xyz EDMG Flow control extension configuration**

The Element ID, Length, and Element ID fields are defined in 9.4.2.1

The RBUFCAP field is defined in 9.3.1.9.7

The Advanced Recipient Memory length exponent indicates free space of Recipient memory at start of TXOP or SP. This subfield is an integer in range 0 to 9. The length defined by this subfield is equal to

2(13 + Advanced Recipient Memory Length Exponent) – 1 octets. This field belongs to all successfully established block ack agreements identified by the same Address 1, and the same Address 2.

The Optional subelements of the EDMG Flow control extension configuration element is defined in Table 9-xyz. The EDMG Flow control extension configuration element may contain no more than two Recipient memory configuration subelements.

**Table 9-xyz Optional subelement IDs for EDMG Flow control extension configuration element**

|  |  |  |
| --- | --- | --- |
| **Subelement ID** | **Name** | **Extensible** |
| 0 | EDMG Flow control capabilities | Yes |
| 1 | Recipient memory configuration | Yes |
| 2-220 | Reserved |  |
| 221 | Vendor specific |  |
| 222-225 | Reserved |  |

|  |  |  |  |
| --- | --- | --- | --- |
|  | Element ID | Length | Recipient memory capabilities |
| Octets | 1 | 1 | 1 |

**Figure 9-xyz1 EDMG Flow control capabilities subelement**

The Element ID is defined in Table 9-xyz

The Length field is defined in sub clause 9.4.2.1

The Recipient memory capabilities are presented in Figure 9-xyz2

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | B0 | B1 | B2 | B3 | B4 | B5 B7 |
|  | RBUFCAP quantity capable | Advanced Recipient Memory length capable | Recipient Memory Multiple Buffer Units capable  | TID Grouping capable  | Memory config tagcapable | Reserved |
| Bits | 1 | 1 | 1 | 1 | 1 | 3 |

**Figure 9-xyz2 Recipient memory capabilities field**

The RBUFCAP quantity capable subfield is set to 1 to indicate support of RBUFCAP values in range 1-0xFE and set to 0 otherwise (Table 9-y1). The RBUF\_Unit\_Size field value is greater than 0 in the Recipient memory configuration subelement provided by the Recipient if it sets to 1 the RBUFCAP quantity capable subfield.

The Advanced Recipient Memory length capable subfield is set to 1 to indicate support of Advanced Recipient Memory length exponent and is set to 0 otherwise (Figure 9-xyz and Table 1 —A-MPDU Parameters field definition).

The Recipient Memory Multiple Buffer Units capable subfield is set to 1 to indicate support of Mem\_Unit\_Size, MaxMPDU\_per\_MemUnit, and Mult\_Buff\_MPDU values and is set to 0 otherwise (Figure 9-xyz3).

The TID Grouping capable subfield is set to 1 to indicate support of TID Grouping values and set to 0 otherwise (Figure 9-xyz3).

The Memory config tag capable subfield is set to 1 to indicate capability to support two Memory\_Config\_tag values and is set to 0 otherwise (Figure 9-xyz3).

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Element ID | Length | Memory\_config\_tag | RBUF\_Unit\_Size | Recipient Memory Multiple Buffer Units parameters | TID Grouping |
|  | Mem\_Unit\_ Size  | Max MPDU\_per\_ MemUnit | Mult\_Buff\_MPDU |
| TID/TSID |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|  Octets 1 | 1 | 1 | 2 | 2 | 1 | 1 | 2 |

**Figure 9-xyz3 Recipient memory configuration subelement**

The Recipeint memory configuration subelement is presented in Figure 9-xyz3

The Element ID is defined in Table 9-xyz

The Length field is defined in sub clause 9.4.2.1

The Memory\_config\_tag indicates one of two memory structures applicable for the TID/TSID indicated in ADDBA Response frame there the EDMG Flow control extension configuration element is presented. Allowed values are 0 and 1.

The RBUF\_Unit\_Size is used as a measure unit in RBUFCAP to deliver information of the Recipient free space available for Originator MPDU delivery. The recipient memory free space = RBUFCAP x RBUF\_Unit\_Size (byte)

The Mem\_Unit\_Size field (byte) indicates size of each buffer unit in the Responder memory. Minimal allowed value is 32.

The MaxMPDU\_per\_ MemUnit is a maximal number of MPDUs that can be collected in a single buffer. Valid values are 1-0xFE, value equal to 0xFF indicates unlimited number of MPDUs in the single buffer. This field is NA if the Mem\_Unit\_Size field = 0

The Mult\_Buff\_MPDU if set to 1 to indicate that single MPDU may be split between memory buffer units in Recipient memory, and is set to 0 otherwise. This field is NA if the Mem\_Unit\_Size field = 0

The TID grouping field indicates TID/TSIDs that correspond to a TID of a ADDBA Response frame a Recipient memory configuration subelement(s) is delivered with. The Recipient memory configuration becomes applicable to the corresponding TID/TSIDs. The RBUFCAP field delivered in a EDMG Flow control extension configuration element of ADDBA Response frame and a RBUCAP field delivered in a BlockAck frame is applicable for all TID/TSID that correspond to the TID of the ADDBA Response frame the TID grouping is delivered with. The Advanced Recipient Memory length exponent field delivered in a EDMG Flow Control Extension Configuration element of ADDBA Response frame is applicable for all TID/TSID that correspond to the TID/TSID of the ADDBA Response frame the TID grouping is delivered with.

**9.6.5.2 ADDBA Request frame format**

*Editor, modify Table 9-303—ADDBA Request frame Action field format as presented below and add text after last paragraph of the sub clause*



|  |  |
| --- | --- |
| 11 | EDMG Flow control extension configuration (optional) |

**9.6.5.3 ADDBA Response frame format**

*Editor, modify Table 9-304—ADDBA Response frame Action field format as presented below and add text after last paragraph of the sub clause*

**

|  |  |
| --- | --- |
| 11 | EDMG Flow control extension configuration (optional) |

The EDMG Flow control extension configurationelement is defined in 9.4.2.258.

**10.24 Block acknowledgment (block ack)**

**10.24.1 Introduction**

*Editor, append at end of the sub clause:*

A EDMG Flow control is an amendment to Block Ack mechanism to prevent overloading of Recipient memory space that may happen in case that link speed is higher than system capability to deliver, process or consume incoming data stream. The EDMG Flow control solves Recipient limitations and optimizes Network and the Recipient performance.

**10.24.2 Setup and modification of the block ack parameters**

*Editor, append at end of the sub clause:*

A EDMG Originator STA may insert EDMG Flow Control Extension Configuration element (9.4.2.258) in a ADDBA Request frame. A EDMG Flow control capabilities subelement shall be present in the EDMG Flow control extension configuration element.

A EDMG Recipient STA that responds to ADDBA Request frame that contains EDMG Flow control extension configuration element should insert a EDMG Flow control extension configuration element in the response. A EDMG Flow control capabilities subelement shall be present in the EDMG Flow control extension configuration element. A Recipient memory configuration subelement shall be included in the EDMG Flow control extension configuration element if at least one subfield in Recipient memory capabilities field is not equal to 0.

EDMG STAs that established block ack agreement with or without exchange of EDMG Flow control extension configuration element shall follow RBUFCAP operation rules defined in (10.24.3, 10.24.4, 10.24.7.5, 10.24.7.7)

Following negotiation rules apply to EDMG STAs that exchange ADDBA Request and ADDBA Response frames.

All recipient memory capabilies are supported if relevant subfileds in the Recipient memory capabilities field are set to one in ADDBA Request and ADDBA Response frames of the block ack agreement established between two EDMG STAs, and not supported otherwise.

A recipient shall not respond with Status Code = SUCCESS in a ADDBA Response frame it sends if the recipient sets to one at least one of subfields in the Recipient memory capabilities field and

* the subfiled is set to 0 in the Recipient memory capabilities field in a ADDBA Request frame received from originaror or
* no EDMG Flow control extension configuration element is present in the ADDBA Request frame.

Note: Status Code values REFUSED, REFUSED\_REASON\_UNSPECIFIED, or REQUEST\_DECLINED, or INVALID\_PARAMETERS can be used in the mentioned case

If a Recipient Memory Multiple Buffer Units capable subfield is set to one and a RBUFCAP quantity capable subfield is set to zero then the Recipient Memory Multiple Buffer Units capability is not supported at Block Ack agreement.

An Advanced Recipient Memory length capable is supported in the successfully established Block Ack agreement if at least one of the conditions is met and is not supported otherwise

* the Advanced Recipient Memory length capable is set to one and the RBUFCAP quantity capability is set to one,
* the Advanced Recipient Memory length capable is set to one and

Advanced Recipient Memory Length Exponent >= Maximum A-MPDU Length Exponent

**10.24.3 Data and acknowledgment transfer using immediate block ack policy and delayed**

**block ack policy**

*Editor, add new text in the indicated place (P1418)*

After setting up either an immediate block ack agreement or a delayed block ack agreement following the procedure in 10.24.2, and having gained access to the medium and established protection, if necessary, the originator may transmit a block of QoS Data frames separated by SIFS, with the total number of frames not exceeding the Buffer Size subfield value in the associated ADDBA Response frame and subject to any additional duration limitations based on the channel access mechanism.

Total number of frames to be transmitted by a EDMG originator is limited by following rules:

- the total number of frames shall not exceed the Buffer Size subfield value in the associated ADDBA Response frame and

- memory occupied by the frames shall not exceed maximum between value indicated in Advanced Recipient Memory length exponent if the Advanced Recipient Memory length capability is supported and value indicated in RBUFCAP field in the associated ADDBA Response frame. Actual RBUFCAP value is either delivered by EDMG Flow control extension configuration element of the ADDBA Response frame or it is last update of RBUFCAP of other TIDs as indicated in TID grouping field of the Recipient memory configuration sublement. If the ADDBA Response frame does not contain the EDMG Flow control extension configuration element the relevant originator parameter shall become value of getting RBUFCAP of Unlimited\_space (9.3.1.9.7)

*Editor, add new text in the indicated places (P1418)*

The originator requests acknowledgment of outstanding QoS Data frames by sending a Basic

BlockAckReq frame. The recipient shall maintain a block ack record for the block. A EDMG recipient shall maintain free memory space for the established Block Ack agreement(s) (10.24.7.5).

*Editor, add new text at end of paragraph that starts with*

“Subject to any constraints in this sub clause about permitted use of TXOP or SP according to the channel access mechanism used, the originator may…”

A EDMG recipient that is TXOP holder or source of SP may transmit ADDBA response frame to the originator as defined in (10.24.4).

**10.24.4 Receive buffer operation**

*Editor, add new text after last paragraph in the sub clause*

A STA that has successfully negotiated a block ack agreement shall obey the following rules as a block ack recipient in addition to rules specified in 10.24.7.

The recipient may transmit ADDBA Response frame that is not response to ADDBA Request frame to update RBUFCAP and Advanced Recipient Memory length values. The unsolicited ADDBA Response frame shall be sent by the recipient which is an TXOP holder. A dialog token of the frame shall be set to 0. Status code shall be set to SUCCESS in the frame. A EDMG Flow control extension configuration element shall be included in the frame. The EDMG Flow control extension configuration element shall not contain EDMG Flow control capabilities and Recipeint memory configuration subelements. RBUFCAP and Advanced Recipient Memory length exponent fields shall comply with RBUFCAP quantity and Advanced Recipient Memory length capabilities respectively established for the block ack agreement. After delivering of the ADDBA Response frame the recipinet may transmit a Grant frame to the TXOP responder in compliance with (10.36.7.3) to relinquish the remainder of the TXOP to the TXOP responder that is the originator of the block ack agreement. After delivering of the ADDBA Response frame the recipient may grant reverse direction in compliance with (10.28) to the TXOP responder that is the originator of the block ack agreement.

A STA that has successfully negotiated a block ack agreement shall obey the following rule as a block ack originator in addition to rules specified in 10.24.7.

The originator that receives ADDBA Response frame with dialog token of the frame set to 0 shall ignore Block Ack Parameter set and Block Ack timeout Value fields of the frame. At receive of the ADDBA Response frame the originator shall update values of RBUFCAP and Advanced Recipient Memory length exponent to values delivered in the frame in compliancy with RBUFCAP quantity and Advanced Recipient Memory length capabilities respectively.

**10.24.7 HT-immediate block ack extensions**

**10.24.7.1 Introduction to HT-immediate block ack extensions**

*Editor, add new text in the indicated place (P1422)*

A DMG STA shall support HT-immediate block ack.

A EDMG STA shall support HT-immediate block ack with EDMG Flow control extension.

**10.24.7.2 HT-immediate block ack architecture**

*Editor, add new text in the indicated place (P1422)*

WinStartO is the starting sequence number of the transmit window, and WinSizeO is the number of buffers negotiated in the block ack agreement.

The EDMG originator contains an EDMG recipient free memory control that uses RBUFCAP to limit number of transmitted MPDUs to not exceed applicable memory space of the recipient.

*Editor, add figure (P1423)*



**Figure 10-35x—HT-immediate block ack architecture with EDMG Flow control amendment**

*Editor, add new text in the indicated place (P1423)*

This entity provides the bitmap and the value for the Starting Sequence Number subfield to be sent in BlockAck frame responses to the originator.

The EDMG recipient contains an EDMG free memory space maintenance entity. The entity is responsible to compute RBUFCAP value. This entity provides the RBUFCAP value subfield to be sent in BlockAck Response frame and in ADDBA Response frame to the originator.

**10.24.7.5 Generation and transmission of BlockAck frames by an HT STA or DMG STA**

*Editor, add new text after last paragraph in the sub clause*

In EDMG STA when responding with a BlockAck frame to either a received BlockAckReq frame or a received A-MPDU with Ack Policy equal to Normal Ack (i.e., implicit block ack request) during either full-state or partial-state operation, adjustment to the free space memory value shall be performed at the generation and transmission of the response BlockAck frame. The RBUFCAP subfield of the BlockAck frame shall be computed as defined in Table 10-xyz. The Free memory space is estimation of amount of recipient memory that is available to collect MPDUs at a time and during receiving of a forthcoming A-MPDU.

**Table 10-xyz RBUFCAP value computing**

|  |  |  |
| --- | --- | --- |
| Free memory space comparison | RBUFCAP quantity support (10.24.2) | RBUFCAP field value |
| Free\_memory\_space>= (2(13 + Maximum A-MPDU Length Exponent) – 1)  | Supported  | Unlimited\_space |
| Free\_memory\_space>=(2(13 + Maximum A-MPDU Length Exponent) – 1) | Not supported | Unlimited\_space |
| Free\_memory\_space<(2(13 + Maximum A-MPDU Length Exponent) – 1) | Supported  | Int [Free\_memory\_space/ RBUF\_Unit\_Size] |
| Free\_memory\_space<(2(13 + Maximum A-MPDU Length Exponent) – 1) | Not supported | Zero\_space |

**10.24.7.7 Originator’s behavior**

*Editor, add new text after last paragraph in the sub clause*

An originator that is an EDMG STA shall obey the following rules:

The Originator shall identify applicable QoS data transfer parameters of established Block Ack agreement by Address 1, Address 2, TID, and Memory\_configuratiuon\_tag if the latter is supported.

The Originator shall not transmit QoS data to the Recipient at start of data transfer sequence if one of the following conditions is met:

* Advanced Recipient Memory length capable =0, and RBUFCAP= Zero\_space
* Advanced Recipient Memory length capable =0, and No\_Mem\_kept = 1

The Originator shall not transmit QoS data to the Recipient when getting Block Ack frame or ADDBA Response frame in middle of sequence if RBUFCAP= Zero\_space in the received frame.

The originator may transmit QoS Data frames with a TID matching an established block ack agreement that their sequence numbers lie within the current transmission window (10.24.7.7). Number of the frames shall be less or equal to A-MPDU\_Data\_frames\_Limit. The A-MPDU\_Data\_frames\_Limit is computed using A-MPDU Byte Count Limit and Recipient Memory Multiple Buffer Units parameters if Recipient Memory Multiple Buffer Units capable =1 (Table 10-xyz3). The A-MPDU Byte Count Limit computing is defined in the Table 10-xyz1 and in the Table 10-xyz2. The A-MPDU Byte Count Limit does not include MPDU Delimiter and A-MPDU padding.

**Table 10-xyz1 A-MPDU\_Byte\_Count\_Limit computing at start of data transfer sequence**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Advanced Recipient Memory length capable | Advanced Recipient Memory length = 2(13 + Advanced Recipient Memory Length Exponent) – 1 | RBUFCAP quantity support | RBUFCAP | No\_Mem\_kept | A-MPDU Byte Count Limit |
| 0 | NA | NA | Unlimited\_space | 0 | 2(13 + Maximum A-MPDU Length Exponent) – 1 |
| 0 | NA | 1 | RBUFCAP  | 0 | RBUFCAP \* RBUF\_Unit\_Size |
| 1 | Advanced Recipient Memory Length Exponent >= Maximum A-MPDU Length Exponent | NA | NA | 1 | 2(13 + Maximum A-MPDU Length Exponent) – 1 |
| 1 | Advanced Recipient Memory Length Exponent < Maximum A-MPDU Length Exponent | 1 | NA | 1 | 2(13 + Advanced Recipient Memory Length Exponent) – 1 |
| 1 | COND1 and COND2 | 1 | RBUFCAP/Zero\_space | 0 | 2(13 + Advanced Recipient Memory Length Exponent) – 1 |
| 1 | COND1 and COND3 | NA | Unlimited\_space | 0 | 2(13 + Maximum A-MPDU Length Exponent) – 1 |
| 1 | COND1 and COND3 | 1 | RBUFCAP | 0 | RBUFCAP \* RBUF\_Unit\_Size |

NOTE:

COND1 indicates Advanced Recipient Memory Length Exponent < Maximum A-MPDU Length Exponent

COND2 indicates RBUFCAP \* RBUF\_Unit\_Size= <2(13 + Advanced Recipient Memory Length Exponent) – 1

COND3 indicates RBUFCAP \* RBUF\_Unit\_Size >2(13 + Advanced Recipient Memory Length Exponent) – 1

**Table 10-xyz2 A-MPDU\_Byte\_Count\_Limit computing in middle of data transfer sequence**

|  |  |  |
| --- | --- | --- |
| RBUFCAP quantity support | RBUFCAP | A-MPDU Byte Count Limit |
| NA | Unlimited\_space | 2(13 + Maximum A-MPDU Length Exponent) – 1 |
| 1 | RBUFCAP  | RBUFCAP \* RBUF\_Unit\_Size |

**Table 10-xyz3 A-MPDU\_Data\_frames\_Limit computing**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Recipient Memory Multiple Buffer Units capable | Mem\_Unit\_ Size(byte)  | Max\_MPDU\_per\_MemUnit | Mult\_Buff\_MPDU | A­-MPDU\_Data\_frames\_Limit |
| 0 | NA | NA | NA |  Σ(Ceiling[BCk/4] \*4+4) <= A-MPDU\_Byte\_Count\_Limitk=1: A­-MPDU\_Data\_frames\_Limit |
| 1 | M | 1 | No | (A-MPDU\_Byte\_Count\_Limit)/M, M>=Ceiling[BCk/4] \*4+4 |
| 1 | M | 1 | Yes |  Σ(Ceiling[(Ceiling[BCk/4] \*4+4)/M] \*M) <= A-MPDU Byte Count Limitk=1: A­-MPDU\_Data\_frames\_Limit |
| 1 | M | N | No | See pseudocode in Fig. 10-36  |

NOTE: BC is byte count of MPDU

|  |
| --- |
| k=1;M /\* Mem\_Unit\_Size - memory buffer size in bytes \*/N /\* Max MPDU\_per\_MemUnit - maximum number of MPDUs per Buffer \*/k /\* running MPDU index, in TX queue \*/;While /\* fill another buffer \*/While  End End |

**Fig. 10-36 A-MPDU\_Data\_frames\_Limit computing when**

**Recipient Memory Multiple Buffer Units capable**

**References:**

1. Draft IEEE P802.11ay/D0.3, March 2017
2. IEEE Std 802.11-2016