Pat Thaler
4 Feb 98

Usage of maxFrameSize in 8802-3 and it's supplements

Based upon search of 8802-3, 802.3u, 802.3xy and 802.3z with deprecated clauses excluded. This means that several instances of maxFrameSize in clause 19 are not cited.

Places where the value must be 1518:

802.3xy, 3.2.7 Data and PAD Þelds: The maximum possible size of the data field is maxFrameSize - (2 x addressSize + 48)/8 octets.

802.3z, 4.2.7.1 Common Constants, and Types and Variables: maxValidFrame = maxFrameSize – (2 x addressSize + lengthOrTypeSize + crcSize) / 8;
{in octets, the maximum length of the MAC client data field. This constant is defined for editorial convenience, as a function of other constants}

Places where implementations may use either 1518 or 1522 and are not required to base that on the Type Field value:

802.3z, 4.2.4.2.1 Framing: a) Maximum Frame Size. The receiving CSMA/CD sublayer is not required to enforce the frame size limit, but it is allowed to truncate frames longer than maxFrameSize octets and report this event as an (implementation-dependent) error.

802.3z, 4.2.9 Frame Reception: exceedsMaxLength := ...; {check to determine if receive frame size exceeds the maximum permitted frame size (maxFrameSize)}

802.3xy, 4.4.2 Allowable implementations: NOTE-Current approved projects that are in development in IEEE 802 may result in an increase in maxFrameSize of several octets. This increase would be accompanied by a change to the Data Link layer frame format. The number of octets available for upper layer protocols or user data would not increase.

802.3z, 5.2.4.1 Common Constants and Types: maxDeferTime = ...; {2 x (maxFrameSize x 8) for operating speeds of 100 Mb/s and below, and 2 x (burstLimit + maxFrameSize x 8 + headerSize) for operating speeds greater than 100 Mb/s, in bits, error timer limit for maxDeferTime} [Note: this was not addressed in 802.3ac, but I put it in this category since the precision of this value is not that important. This value is twice maxFrameSize and will not detect false errors with Tagged frames even if 1518 is used.]

8802-3, 12.4.3.2.5 Retiming (jitter removal): Excessive differences in clock rates (caused by clocks not meeting 12.3.2.4.1) and excessively long packets (caused by exceeding maxFrameSize) may each cause the capacity of the retiming function to be exceeded.

802.3z, 30.3.1.1.25 aFrameTooLongErrors: NOTE— The parameter maxFrameSize is being considered for revision in P802.3ac to accommodate the requirements of two

bridging projects under development, P802.1p and P802.1Q.; [Interestingly, the actual behavior just vaguely says "frames that exceeded the maximum permitted frame size" rather than refer to maxFrameSize parameter or exceedsMaxLength variable, though many other objects use maxFrameSize.]

802.3z, 30.4.3.1.4 aReadableFrames: A representation of the total frames of valid frame length. Increment counter by one for each frame whose OctetCount is greater than or equal to minFrameSize and less than or equal to maxFrameSize (see 4.4.2.1) and for which the FCSError and CollisionEvent signals are not asserted.

802.3z, 30.4.3.1.6 aFrameCheckSequenceErrors: Increment counter by one for each frame with the FCSError signal asserted and the FramingError and CollisionEvent signals deasserted and whose OctetCount is greater than or equal to minFrameSize and less than or equal to maxFrameSize (see 4.4.2.1)…..
NOTE— The parameter maxFrameSize is being considered for revision in P802.3ac to accommodate requirements of two bridging projects under development, P802.1p and P802.1Q.;

802.3z, 30.4.3.1.7 aAlignmentErrors: Increment counter by one for each frame with the FCSError and FramingError signals asserted and CollisionEvent signal deasserted and whose OctetCount is greater than or equal to minFrameSize and less than or equal to maxFrameSize (see 4.4.2.1).

802.3z, 30.4.3.1.8 aFramesTooLong: Increment counter by one for each frame whose OctetCount is greater than maxFrameSize (see 4.4.2.1)….
NOTE—The parameter maxFrameSize is being considered for revision in P802.3ac to accommodate the requirements of two bridging projects under development, P802.1p and P802.1Q.;

8802-3, B.2.1 Delay budget:
Hub Delay Stretch/Shrink (see 12.9.5) 3
((preamble + <sfd> + maxFrameSize) · 0.01% · 2)

Variable declaration and value assignment:

802.3z, 4.2.7.1 Common Constants, and Types and Variables: maxFrameSize = ... ; {in octets, implementation-dependent, see 4.4}

802.3z, 4.4.2.1 Parameterized values: maxFrameSize 1518 octets

802.3z, 4.4.2.2 Parameterized values: maxFrameSize 1518 octets

802.3z, 4.4.2.3 Parameterized values: maxFrameSize 1518 octets

802.3z, 4.4.2.4 Parameterized values: maxFrameSize 1518 octets

802.3z, 5.2.4.1 Common Constants and Types: maxFrameSize = ...; {in octets, implementation-dependent, see 4.4}

Other references to maximum frame size:

802.3xy, 4.3.2 Services provided by the MAC sublayer: The frameTooLong error indicates that a frame was received whose frameSize was beyond the maximum allowable frame size.

802.3u, 25.3 General exceptions: maximum stream size = 3054 code-groups.

802.3u, 26.3 General exceptions: maximum stream size = 3054 code-groups.

Places where the value used must be dependent on Type Field value:

Absolutely NONE!

A proposal for adapting 802.3 for tagged frame lengths

In no place is maxFrameSize actually used where it has a different value for tagged and untagged packets because the requirements for acceptable transmit frames control length in terms of MAC client data size which is the same regardless of tag. Therefore, a general approach which does not require any pseudo-constant which changes value depending on frame type field can be applied.

Leave maxFrameSize as a true constant with a value of 1518 and rename it maxUntagFrameSize.

Create qTagPrefixSize := `... ; {4 octets}`

For receive packet handling by MACs, change the action for setting exceedsMaxLength to allow a choice between maxUntagFrameSize or maxUntagFrameSize + tagSize and implementations may either always use one value in which case the latter is recommended or may choose the value based on the type field value.

In MAC entity managed objects, for 30.3.1.1.25 aFrameTooLongErrors no change is necessary since the exceedsMaxLength change in the Pascal takes care of it. (exceedsMaxLength is used to determine whether frame status is FrameTooLong.)

In repeater entity managed objects for 802.3z, 30.4.3.1.8 aFramesTooLong: Increment counter by one for each frame whose OctetCount is greater than maxFrameSize (see 4.4.2.1)….

Change to " Increment counter by one for each frame whose OctetCount is greater than maxUntagFrameSize or maxUntagFrameSize +tagSize (see 4.4.2.1). A repeater may use either value, in which case the latter value is recommended. Alternatively, a repeater may use the value maxUntagFrameSize + tagSize for tagged frames and maxUntagFrameSize for untagged frames…."

For aReadableFrames, aAlignmentErrors, and aFrameCheckSequenceErrors, condition them incrementing on aFramesTooLong not being incremented rather than on length less than either value as that makes sure that only one increments for each frame.

For 4.2.4.2.1, Wording is open enough that no change except updating to "maxUntagFrameSize" is necessary. "is allowed to truncate". Do we want to add a recommendation here that the receiving sublayer be capable of receiving frames maxUntagFrameSize + tagSize?

For 5.2.4.1, 12.4.3.2.5 and B.2.1, update variable name. No other change is needed. Clock skew is less than 3 bits for either frame size and deference already has a 100% safety margin. Should we deprecate 12?

4.3.2 would be a good text place to explain the use of maxUntagFrameSize and maxUntagFrameSize + tagSize.

In 25.3 and 26.3 increase the maximum stream by 8 code groups or change to a calculation based on maxUntagFrameSize + tagSize.