

A Recommended Amendment of the Joint Committee on the NTCIP

NTCIP 1203 Amendment 1 v07

National Transportation
Communications for ITS Protocol

Object Definitions for Dynamic
Message Signs (DMS) Amendment 1

Amendment 1 v07 July 03, 2001

This is a draft pre-standard document, which is distributed for review and ballot purposes only. You may reproduce and distribute this document within your organization, but only for the purposes of and only to the extent necessary to facilitate review and ballot to AASHTO, ITE, or NEMA. Please ensure that all copies reproduced or distributed bear this legend. This pre-standard contains recommended information which is subject to approval.

Published by

American Association of State Highway and Transportation Officials (AASHTO)

444 North Capitol St., N.W., Suite 249
Washington, D.C. 20001

Institute of Transportation Engineers (ITE)

1099 14th Street, N.W., Suite 300 West
Washington, D.C. 20005-3438

National Electrical Manufacturers Association (NEMA)

1300 North 17th Street, Suite 1847
Rosslyn, Virginia 22209-3801

Revision History

1203 Amendment 1 v01	November 2000	
1203 Amendment 1 v02	November 2000	
1203 Amendment 1 v03	January 2001	
1203 Amendment 1 v04	February 2001	
1203 Amendment 1 v05	February 2001	Former filename 1203 Amendment 1-14Feb01
1203 Amendment 1 v06	February 2001	Former filename 1203 Amendment 1-20Feb01
1203 Amendment 1 v07	July 2001	Former filename 1203 Amendment 1d-010703

This document is NTCIP 1203 Amendment 1. This document, when approved by AASHTO, ITE, and NEMA, will revise NTCIP Standards Publication NTCIP 1203:1997, version 01.15, December 1999.

When this amendment is approved, notice will be provided to previous purchasers of NTCIP 1203:1997. The approved revisions will be incorporated into NTCIP 1203:1997 as version 01.16, for a revised printed version.

The referenced clauses in NTCIP 1203:1997 are recommended for revision according to the instructions in the “dash-italic text” below.

1.2.1 Normative References

-Add the following statement after the listing:

NTCIP 1102 *National Transportation Communications for ITS Protocol – Octet Encoding Rules (OER)
– Ballot Draft*

1.3 GENERAL STATEMENTS

-Second paragraph shall be modified to read

This document including OID nodes and official extension is managed by the DMS Working Group of the NTCIP Joint Committee. Proprietary features should be defined through vendor-specific nodes or vendor-specific extensions to this MIB.

1.4 GLOSSARY OF TERMS

-Change definitions to the following

Bulb Matrix	A type of display technology where an incandescent light source is used for each pixel.
Central Control Mode	Control of the sign from the central computer. Preferred term for remote control mode.
Lamp	A light source used to illuminate the message other than on a pixel-by-pixel basis.
Pixel Service	A generic term for a periodic activation to exercise mechanical pixels. The service may or may not be enabled during the display of a particular message.
Pixel Test	A test to determine the operational status of individual pixels (i.e. stuck on, stuck off).
Remaining Message Display Time	The amount of time before the message currently being displayed is to end.

-Delete the following definition

Lamp Matrix	A type of display technology where an incandescent light source is used for each pixel.
--------------------	---

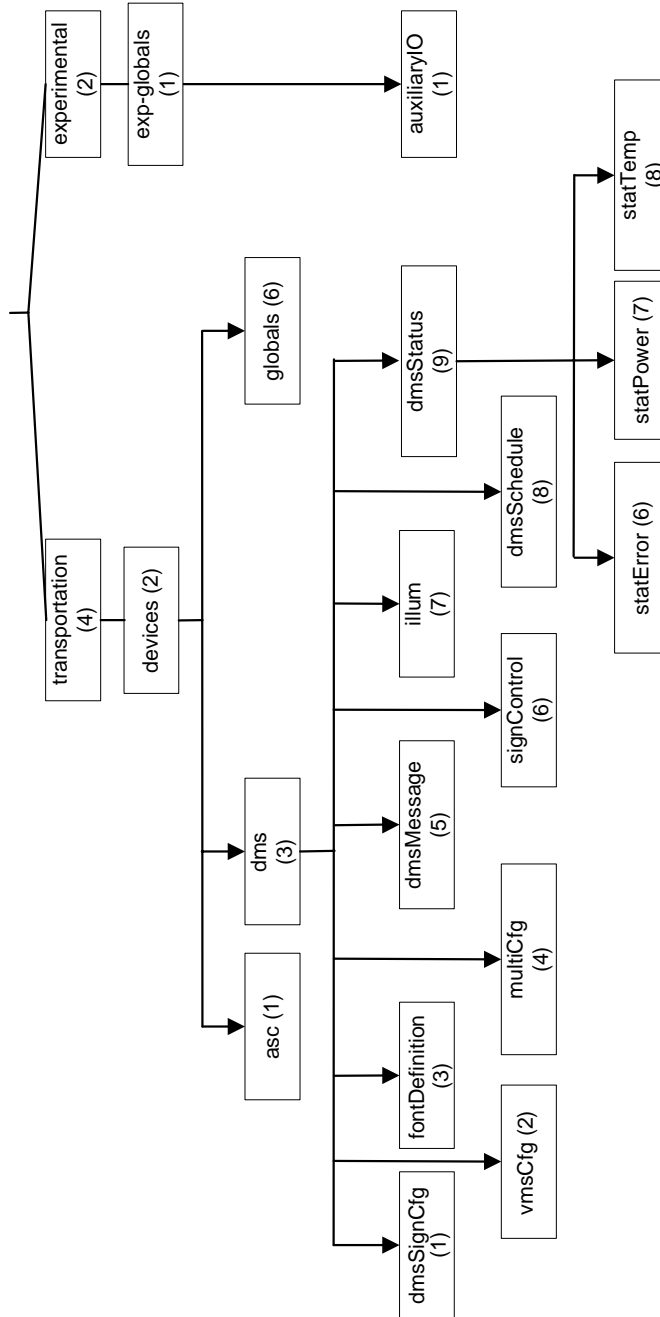
-Add the following definition

Watchdog

A circuit or device that monitors the operational status of software and/or hardware. This circuit or device resets the controller if an error is detected.

1.5 DMS OBJECT TREE

-Correct the figure to indicate that fontDefinition is (3) under dms and vmsCfg is (2) under dms in order to be consistent with the MIB definition.



2.1 DYNAMIC MESSAGE SIGNS (DMS) OBJECTS

-Change section 2.1 to the following

```
DMS-MIB DEFINITIONS ::= BEGIN
```

```
IMPORTS
```

```
IpAddress, Counter
```

```
FROM RFC1155-SMI
```

```
DisplayString
```

```
FROM RFC1213-MIB
```

```
OBJECT-TYPE
```

```
FROM RFC-1212
```

```
experimental
```

```
FROM NEMA_SMI
```

```
OwnerString, devices
```

```
FROM TMIB;
```

```
--For the purpose of this section, the following OBJECT IDENTIFIERS are used:
```

```
--the node location is: nema / transportation
```

```
    dms OBJECT IDENTIFIER ::= {devices 3}
```

```
-- Additionally, OBJECT IDENTIFIERS for the Auxiliary objects (see section 2.7.3) which are
```

```
-- located under:
```

```
    exp-global OBJECT IDENTIFIER ::= {experimental 1}
```

```
MessageIDCode ::= OCTET STRING (SIZE(5))
```

```
-- The MessageIDCode consists of those parameters required to define a
```

```
-- message within a dmsMessageTable. It is defined as an OCTET STRING containing the OER-encoding
```

```
-- of the following ASN.1 structure.
```

```
--MessageIDCodeStructure ::= SEQUENCE
```

```
-- {
```

```
--     messageMemoryType      INTEGER (0..255),
```

```
--     messageNumber          INTEGER (0..65535),
```

```
--     messageCRC              OCTET STRING (SIZE (2))
```

```
-- }
```

```
-- The fields are defined below.
```

```
MessageActivationCode ::= OCTET STRING (SIZE(12))
```

```
-- The MessageActivationCode consists of those parameters required to activate a
```

```
-- message on a DMS. It is defined as an OCTET STRING containing the OER-encoding
```

```
-- of the following ASN.1 structure.
```

```
--MessageActivationCodeStructure ::= SEQUENCE
```

```
-- {
```

```
--     duration                INTEGER (0..65535),
```

```
--     activatePriority         INTEGER (0..255),
```

```
--     messageMemoryType       INTEGER (0..255),
```

```
--     messageNumber           INTEGER (0..65535),
```

```
--     messageCRC              OCTET STRING (SIZE (2)),
```

```
--     sourceAddress           OCTET STRING (SIZE (4))
```

```
-- }
```

-- Duration (16 bits) shall indicate the maximum amount of time, in minutes, the message may be displayed prior to activating the `dmsDefaultEndDurationMessage`. `dmsMsgTimeRemaining` shall be set to this value upon successful display of the indicated message. A Value of 65535 shall indicate an infinite duration.

--

-- `activatePriority` (8 bits) shall indicate the Activation Priority of the message. If this value is greater than or equal to the `dmsMsgRunTimePriority` of the currently displayed message, the new message shall be displayed unless errors are detected.

--

-- `messageMemoryType` (8 bits) shall indicate the `dmsMsgMemoryType` of the desired message.

--

-- `messageNumber` (16 bits) shall indicate the `dmsMsgNumber` of the desired message.

--

-- `messageCRC` (16 bits) shall indicate the `dmsMsgMessageCRC` of the desired message.

--

-- `sourceAddress` (32 bits) shall indicate the 4-byte IP address of the device which requested the activation.

--

-- The `dmsActivateMsgError` object shall be used to indicate the success or failure of activating any message requested by an object with a SYNTAX of `MessageActivationCode`.

--

-- Three special conditions exist with the `MessageActivationCode` and `MessageIDCode` structures. The first condition is related to blanking the sign. A blank sign is activated by setting the `MsgMemoryType` to 'blank' and the `MessageNumber` to the desired run time priority. Note that these are actual entries into the message table, but there are only 255 blank messages (because there are only 255 priority levels) and therefore the high-order byte of the `MessageNumber` field shall always be 0x00. Further, to minimize errors in attempting to blank the sign, the CRC for all blank messages shall be 0x0000.

--

-- The second condition is related to operating the scheduler. The scheduler is activated in a fashion similar to other messages. The `dmsMessageMemoryType` is set to 'schedule' (value of 6), the `MessageNumber` is set to '1', and the `MessageCRC` is set to 0x0000. The schedule has a run-time priority, as defined by `dmsRunTimePriority.6.1`, that overrides the run-time priority of the called message. Thus, the run-time priority is constant for all scheduled messages and the central system can set this priority by modifying the value of `dmsRunTimePriority.6.1`. If an invalid message code is received, the sign will continue operations as if the code was not received, after the correct response is transmitted.

--

-- The third special condition pertains to selecting the currently displayed message. This condition is only valid for the 'MessageIDCode' SYNTAX, not for the 'MessageActivationCode' SYNTAX. Specifying the `currentBuffer 5.1` within the `MessageMemoryType` and `MessageNumber` fields of the 'MessageIDCode' SYNTAX will be used within default messages such as `dmsShortPowerRecoveryMessage`. This allows a message that was running prior to a power loss to run after the power loss without changing the contents of `dmsShortPowerRecoveryMessage` every time the `activateMessage` is changed. The `messageCRC` field of the default messages (such as `dmsShortPowerRecoveryMessage`) shall be 0x0000, when the `messageMemoryType` field has a value of 'currentBuffer'.

2.2.1.1.1.1 Sign Access Parameter

-Add the following to the end of the DESCRIPTION field

If a bit is set to one (1), then the associated feature exists; if the bit is set to zero (0), then the associated feature does not exist.

2.2.1.1.9 Sign Technology Parameter

*-Change bit 5 definition to
Bit 5- Bulb*

-Add the following to the end of the DESCRIPTION field

If a bit is set to one (1), then the associated feature exists. If the bit is set to zero (0), then the associated feature does not exist.

2.4.1.1.1.2 Font Table Parameter

-Add the following to the end of the DESCRIPTION field

Changing an object in a font or character table while the font is used by a displayed message will yield unpredictable results.

2.4.1.1.1.2.5 Font Character Spacing Parameter

-Add the following to the end of the DESCRIPTION field

Value of spacing is rounded up to nearest whole pixel.

2.4.1.1.1.2.6 Font Line Spacing Parameter

-Change section 2.4.1.1.1.2.6 to

2.4.1.1.1.2.6 Font Line Spacing Parameter

fontLineSpacing OBJECT-TYPE
SYNTAX INTEGER (0..255)
ACCESS read-write
STATUS mandatory

DESCRIPTION "Indicates the default vertical spacing (in pixels) between each of the lines within the font. This object only applies to Full Matrix. The line spacing for a line is the largest font line spacing of all fonts used on that line. The number of pixels between adjacent lines is the average of the line spacing of each line. Value of spacing is rounded up to nearest whole pixel. Character Matrix VMS and Line Matrix VMS shall either set this object to zero (0), or not support this object."
 ::= {fontEntry 6}

2.4.1.1.1.2.7 Font Version ID Parameter

-A duplicate section of comment text was found. Delete the following from comment.

-- The following 4 definitions are used to define the above referenced FontVersionByteStream.

--CharacterInformation ::= SEQUENCE {
-- characterNumber INTEGER,
-- characterWidth INTEGER,
-- characterBitmap OCTET STRING }

-- Complete definitions for these fields, including size information, is contained below.

--

--CharacterInfoList ::= SEQUENCE OF CharacterInformation

-- This list only includes entries for a single fontIndex and where the associated

-- characterWidth value is non-zero

```
--  
--FontInformation ::= SEQUENCE {  
--    fontNumber          INTEGER,  
--    fontHeight          INTEGER,  
--    fontCharSpacing     INTEGER,  
--    fontLineSpacing     INTEGER }  
-- Complete definitions for these fields, including size information, is contained  
-- elsewhere in this document.  
--  
--FontVersionByteStream ::= SEQUENCE {  
--    fontInformation      FontInformation,  
--    characterInfoList   CharacterInfoList }  
-- The characterInfoList shall be for the fontNumber indicated within the fontInformation  
-- field.
```

2.6.1.1.1.8.1 Message Memory Type Parameter

-Add the memory type blank (7)

```
dmsMessageMemoryType OBJECT-TYPE  
SYNTAX      INTEGER {  
                other (1),  
                permanent (2),  
                changeable (3),  
                volatile (4),  
                currentBuffer (5),  
                schedule (6),  
                blank (7)  
            }
```

-Add the following to the end of the DESCRIPTION field

-- blank – there shall be 255 (message numbers 1 through 255) pre-defined, static rows with this
-- message type. These rows are defined so that message codes (i.e., objects with SYNTAX of either
-- MessageIDCode or MessageActivationCode) can blank the sign at a stated run-time priority. The run-
-- time priority of the blank message is equal to the message number (i.e., blank message number 1 has -
-- a run time priority of 1 and so on). The dmsMessageCRC for all messages of this type shall be
-- 0x0000 and the dmsMultiString shall be an OCTET STRING with a length of zero (0). The activation
-- priority shall be determined from the activation priority of the MessageActivationCode.

2.6.1.1.1.8.2 Message Number Parameter

-Add the following to the end of the DESCRIPTION field

When the primary index is 'blank', this value shall be from 1 through 255 and all compliant devices must support all 255 of these 'blank' rows."

2.6.1.1.1.8.3 Message MULTI String Parameter

-Add the following to the end of the DESCRIPTION field

When the primary index is 'schedule', 'blank', 'currentBuffer' or 'permanent', this object shall return a genErr to any SET-request. When the primary index is 'schedule', the object shall return the MULTI string of the currently scheduled message in response to a GET-request. The value of the MULTI string is not allowed to have any null character.

2.6.1.1.1.8.5 Message CRC Parameter

-Add the following to the end of the DESCRIPTION field

Note that the calculation shall assume a value of zero (0) for the `dmsMessageBeacon` object and/or for the `dmsMessagePixelService` object if they are not supported. For messages of the 'blank' message type, the above algorithm shall be ignored and the `dmsMessageCRC` value shall always be zero (0). For messages of the 'schedule' message type, the CRC value of the currently scheduled message shall always be returned (regardless whether this message is actually being displayed)."

2.6.1.1.1.8.6 Message Beacon Parameter

-Add the following to the end of the DESCRIPTION field

When the primary index is 'schedule', 'blank', 'currentBuffer', or 'permanent', this object shall return a `genErr` to any SET-request. When the primary index is 'schedule', the object shall return the `dmsMessageBeacon` setting of the currently scheduled message in response to a GET-request (regardless whether this message is actually being displayed)."

DEFVAL {0}

2.6.1.1.1.8.7 Message Pixel Service Parameter

-Add the following to the end of the DESCRIPTION field

When the primary index is 'schedule', 'blank', 'currentBuffer', or 'permanent', this object shall return a `genErr` to any SET-request. When the primary index is 'schedule', the object shall return the `dmsMessagePixelService` setting of the currently scheduled message in response to a GET-request (regardless whether this message is actually being displayed)."

DEFVAL {0}

2.6.1.1.1.8.8 Message Run Time Priority Parameter

-Add the following to the end of the DESCRIPTION field

When the `dmsMessageType` is 'schedule,' the value set in this object (i.e. `dmsMessageRunTimePriority.6.1`) shall override the run-time priority of the scheduled message. When the `dmsMessageType` is 'currentBuffer' the value returned shall be determined from the `dmsMessageRunTimePriority` of the message referenced in the `dmsMessageTableSource` object. . When the `dmsMessageType` is 'blank', the value returned shall be equal to the `dmsMessageNumber` of that particular message.

2.6.1.1.2 Validate Message Error Parameter

-Add the following to the end of the DESCRIPTION field

The use of values 3 and 4 for this object is strongly discouraged.

2.7.1.1.1.3 Activate Message Parameter

-Add the following to the end of the DESCRIPTION field

If a GET is performed on this object, the sign shall respond with the value for the last message that was successfully implemented. If this message was activated by a message ID code (such as EndDurationMessage), the duration will indicate 65535 (infinite), the activate priority will indicate 255, and the source address will indicate the source address of the sign.

2.7.1.1.1.4 Message Display Time Remaining Parameter

-Add the following to the end of the DESCRIPTION field

A set operation on this object shall allow a management station to extend or shorten the duration of the message. Setting this object to zero (0) shall result in the immediate display of the dmsEndDurationMessage.

2.7.1.1.1.5 Message Table Source Parameter

-Add the following to the end of the DESCRIPTION field

When the scheduler has control of the sign, the value of this object shall point to the 'schedule' row of the table (see definition of the MessageIDCode SYNTAX for clarification on the values of the fields).

2.7.1.1.1.17 Activate Message Error Parameter

*-Change underValidation (4), to:
messageStatus (4),*

2.7.1.1.1.18 MULTI Syntax Error Parameter

*-Change the description of textTooBig to:
-- textTooBig: Too many characters on a line, too many lines for a page, or font is too large for display.*

2.7.1.1.1.21 Pixel Service Duration Parameter

-Delete the following from the DESCRIPTION field

Indicates the pixel service duration in seconds.

-Add the following to the end of the DESCRIPTION field

Indicates the number of seconds to perform pixel service on an entire sign. If the vmsPixelServiceDuration expires during a pixel service routine, that routine shall be completed before stopping or restarting a new pixel service routine due to vmsPixelServiceFrequency. A value of zero disables pixel service.

2.7.1.1.1.22 Pixel Service Frequency Parameter

-Add the following to the end of the DESCRIPTION field

Value of zero indicates continuous pixel service from vmsPixelServiceTime to the epoch of midnight. Value of 1440 indicates one pixel service in a 24-hour period.

2.8.1.1.1.3 Status of Illumination Photocell Level Parameter

-Change the DESCRIPTION field to the following:

DESCRIPTION "Indicates the level of Ambient Light as a value ranging from 0 (darkest) to the value of *dmsIllumMaxPhotocellLevel* object (brightest), based on the photocell detection. The *dmsIllumPhotocellLevelStatus* object is considered a virtual photocell level in that it may be algorithmically determined from one or more photocells and is the value used for calculations dealing with the brightness table. The algorithm used to determine the virtual level from the actual photocell readings is manufacturer specific to accommodate various hardware needs."

2.8.1.1.1.8 Brightness Values Error Parameter

-Change the DESCRIPTION field to

DESCRIPTION "Indicates the error encountered when the brightness table was SET.
other(1) is for a manufacturer specific indication when none of the other possible values can be used.

none(2) indicates that no error was encountered.

photocellGap(3) indicates that certain photocell levels do not have an associated brightness level.

negativeSlope(4) indicates that the photocell range used to select a brighter brightness level is lower or overlaps the photocell range used to select a dimmer brightness level. Note that some signs may allow a negative slope for special conditions without generating an error; e.g., external illumination for a reflective sign may be allowed to turn off during daylight conditions rather than getting brighter.

tooManyLevels(5) indicates that more brightness levels are defined than are reported by *dmsIllumNumBrightLevels*.

invalidData(6) indicates a manufacturer defined condition of invalid data not described by the other options."

2.9.1.1.1.2.2 Action Message Code Parameter

-Add the following after the DESCRIPTION field

DEFVAL {0x0000000000}

2.11.1.1.1.5 Watchdog Failure Count Parameter

-Change the DESCRIPTION field to

DESCRIPTION "A counter indicating the number of watchdog failures that have been detected."

2.11.1.1.1.6 Open Door Status Parameter

-Add the following to the end of the DESCRIPTION field

Each door is associated with a bit (bit-door correlation order specified by manufacturer) allowing for up to 8 doors.

2.11.2.1.1.1 Short Error Status Parameter

-Add the following sentence to the end of the description:

To track a history of transient error conditions utilize the event logging table located in the Global Objects Definitions (NTCIP 1201).

2.11.2.1.1.3.1 Pixel Failure Detection Type Parameter

-Add the following to the end of the DESCRIPTION field

Once a pixel is detected as failed, it is entered in the table with a type of either *pixelTest* or *messageDisplay*. In either case the failed pixel stays in the table until *pixelTestActivation* is set to either *test* or *clearTable*. Detection type *pixelTest* and *messageDisplay* are considered different methods of testing for failed pixels. The *pixelTest* method is considered a foreground processing method of failed pixel detection. Failed pixels detected during a foreground pixel test are entered in the *pixelTest* pixel failure type. During a foreground pixel test, the message on the display may or may not stay present on the display.

The *messageDisplay* method is considered a background processing method of failed pixel detection. During a background test, the readability/legibility of the message should not be affected by the test. If the manufacturer supports background pixel test, failed pixels detected during a background pixel test are entered in the *messageDisplay* pixel failure type

2.11.2.1.1.3.2 Pixel Failure Index Parameter

-Add the following to the end of the DESCRIPTION field

Within each *pixelFailureType*, entries shall start with 1 and be sequential.

2.11.2.1.1.4 Pixel Test Activation Parameter

-Change the DESCRIPTION field to the following

DESCRIPTION "Indicates the state of the pixel testing. The actual test routine can vary among different manufacturers. The results of the pixel failure test shall be stored in the pixel failure table. The pixel failure table will be cleared (both *messageDisplay* and *pixelTest* types) when a pixel test is started (*test*) or table is cleared (*clearTable*). Setting the value to *test* will start the test, meaning this test will be executed once. Pixel failures identified by setting this object to *test* are entered into the *pixelTest* type of the *pixelFailureDetectionType*. The sign controller will automatically set the value of this object back to *noTest* after completion."

2.11.2.1.1.8 Fan Failure Parameter

-Add the following to the end of the DESCRIPTION field

Each fan system is associated with a bit (bit-fan correlation order specified by manufacturer) allowing for up to 32 fan systems to report failure status. A fan system is defined as a single fan, group of fans, sensors, or filter systems. Whether each bit specifies a fan or fan system is dependent on the manufacturer.

2.11.3.1.1.2 Low Fuel Threshold Parameter

-Change the SYNTAX to:

SYNTAX INTEGER (0..100)

2.11.4.1.1.6 Maximum Temperature of Sign Housing Parameter

-Change the SYNTAX to the following:

SYNTAX INTEGER (-128..127)

3.2 MULTI - SETUP AND DEFINITION

3.2.1 Definition

-Change the second paragraph to the following

MULTI currently uses 8-bit characters, but there is consideration and planning to allow the selection of either 8-bit or 16-bit characters. The null character (0x00) is not allowed within MULTI strings. All of the MULTI tags are defined in ASCII, 8-bit characters with the most significant bit set to 0.

-Change the fourth paragraph to the following

Some tags may operate in pairs, in which case the standard tag notation is defined as the opening tag. The opening tag defines where the tag's functionality begins. The closing tag defines where the functionality of the tag ends, and is defined as an opening tag with a forward slash preceding the tag ID i.e., the opening flash tag is "[fl]", and the closing flash tag is "[/fl]".

-Change the seventh paragraph to the following

MULTI allows tags within the text field of another tag (i.e. flashing within moving text), however there are limitations as to the number of tags, use of tags, and complexity of a message due to the display technology of the sign and the sign manufacturer.

3.3 RULES TO APPLY ATTRIBUTE TAGS

-Add the following rule

7) Any tag may be placed between the opening and the closing tag of any attribute.

**Table 3-1
MULTI Tags**

-Change the following table entries

Attribute Tag (opening)	Closing Tag (if existing)	Description
scx	/sc	Spacing Character Specify the spacing between characters

3.4.1 Color Background

-Change the last paragraph to the following

This standard does not require the sign to be able to change the background color, however the Controller must recognize the tag and take appropriate action by implementing functionality or by generating a dmsMultiSyntaxError with a value of unsupportedTag.

3.4.2 Color Foreground

-Change the last sentence to:

This standard does not require the sign to be able to change the foreground color, however the Controller must recognize the tag and take appropriate action by implementing functionality or by generating a dmsMultiSyntaxError with a value of unsupportedTag.

3.4.3 Fields

-Change the table to the following

**Table 3-2
Field Descriptions**

ID	Default Field Width	Allowable Widths	Fill Character	Justification	Overflow Fill	Example	Description
1	5	5	space	right	n/a	'_9:00'	Time, 12 hour format (no AM/PM indicator present)
2	5	5	0	right	n/a	'09:00'	Time, 24 hour format
3	3	2, 3	space	right	space (?)	'-10' or '_10'	Ambient (Outside) Temperature, degrees Celsius (no plus sign)
4	3	2, 3	space	right	space (?)	'-10' or '_10'	Ambient (Outside) Temperature, degrees Fahrenheit (no plus sign)
5	3	2, 3	space	right	'9'	' 90'	Speed, km/h
6	2	2, 3	space	right	'9'	' 55'	Speed, mph
7	3	3	n/a	n/a	n/a	'MON'	Day of week Shall be one of (SUN, MON, TUE, WED, THU, FRI, SAT)
		4-9	manufacturer specific				
8	2	2	0	right	n/a	'05'	Date of month (number)
9	2	2	0	right	n/a	'04'	Month of year (number)
10	2	2	0	right	n/a	'00'	Year, 2 digits
11	4	4	0	right	n/a	'2000'	Year, 4 digits
12 - 49							Reserved for future assignment
50 - 99							User-definable

-Change the two sentences following table to:

The second parameter, y, is optional and if included, must be in the range of 'Allowable Widths'. This defines the width, or number of characters used to display the data.

If the width parameter is supplied, and the data requires fewer characters than specified, the data will be right justified and the indicated fill character will be used to make up the missing characters. If this parameter is supplied, and the data requires more characters than specified, the indicated overflow character will be used for all characters.

If the width parameter is not supplied and the Default Width is variable, only the characters actually present in the data will be used. Overflow characters will still be used if the data is larger than the variable range, for example -100 degrees for field tag 3 or 4.

Specifying widths other than the default width should only be used when you wish to force the fill character to be used, or when you wish to limit the range of displayed data, i.e. to display detected speeds of only 99 mile or kilometers per hour or less.

-Change the last sentence to:

This standard does not require the sign to be able to implement all field ID's, however the Controller must recognize the tag and take appropriate action by implementing functionality or by generating a `dmsMultiSyntaxError` with a value of `unsupportedTag`.

-Add the following sentence before the examples

If a character immediately precedes or follows a field tag, the current character spacing shall be inserted between the character and the field.

-Change the examples to:

To display the message "YOUR SPEED IS aa MPH", where aa is filled with the real-time speed (limited to a maximum of 99 mph) from a local device, the MULTI string could read:
"YOUR SPEED IS [f6,2] MPH"

To display the message "TIME IS aa:aa TEMPERATURE IS bbb F", where aa:aa is filled with the current time and bbb is filled with the current temperature, (using no fill characters), the MULTI string could read:
"TIME IS [f1] TEMPERATURE IS [f4] F"

3.4.4 Flash Time

-Change the paragraph before the examples to:

This standard does not require what, if anything, a sign can or cannot flash, a specific character, word, line, or page. However, the Controller must recognize the tag and take appropriate action by implementing functionality or by generating a `dmsMultiSyntaxError` with a value of `unsupportedTag`.

3.4.5 Font

-Change the text to the following

This tag controls the selection of the font used to display a Message. The font is selected using the *fontNumber*, not the *fontIndex* object from the *fontTable*. The default font is indicated in the *defaultFont* object. When fonts of different heights are displayed on the same line, the bottom-most pixel of each font shall be aligned.

This standard does not require how many fonts are to be supported, or what happens when an undefined font is selected. However, the Controller must recognize the tag and take appropriate action by implementing functionality or by generating a `dmsMultiSyntaxError` with a value of `unsupportedTag`.

3.4.6 Hexidecimal Character

-Change all of the text to the following:

3.4.6 Hexadecimal Character

Tag format: [hcx]

where x is an octet string up to four characters in length, and indicates the character from the current font using the hexadecimal value of the character code to be displayed. "X" shall be a hexadecimal (0-9, A-F) value between 1 and FFFF.

This tag is intended as a method to select a character from a font that cannot be typed on a keyboard (characters 0 through 31 and 128 through 65535).

This standard does not require what happens when an undefined character is selected, however, the Controller must recognize the tag and take appropriate action by implementing functionality or by generating a `dmsMultiSyntaxError` with a value of `unsupportedTag`.

EXAMPLES:

To display the Message "THIS IS A TEST", where " " is the hexadecimal code 8A to have all pixels of the character open, the MULTI string could read:

"THIS IS [hc8A] A TEST"

3.4.7 Justification - Line

-Change the paragraph after table 3.3 to the following:

The centering of text shall be positioned to have the extra space AFTER the text, when exact centering is not possible because of an odd number of remaining spaces. For example, to center NEMA on a seven (7) character sign, the result would be ". NEMA . .", one space before the word NEMA and two spaces after the word NEMA.

-Add the following paragraph before the last paragraph

The line justification tag must be used in logical order (from left, center, right), otherwise `dmsMultiSyntaxError` will be set to "tagConflict". Overlapping of text results in a "textTooBig" value for `dmsMultiSyntaxError`. No other justification tag may be used in conjunction with full justification on the same line.

-Modify the last paragraph to the following:

If an unsupported justification code is selected the Controller must recognize the tag and take appropriate action by generating a `dmsMultiSyntaxError` with a value of `unsupportedTag`.

3.4.8 Justification - Page

-Change the last paragraph to the following

This standard does not require what happens when an unsupported justification code is selected or when combination of justification causes overwriting of characters, however the Controller must recognize the tag and take appropriate action by implementing functionality or by generating a `dmsMultiSyntaxError` with a value of `unsupportedTag`.

3.4.10 Moving Text Tag

-Change this section to the following

Tag format: [mvt dw,s,r,text]
where t

is a character(s) indicating the type of the moving tag. Two types are available:
c = circular,
lx = linear with "x" optionally indicating the delay in tenths of a second between the end of linear motion and the restarting of the linear motion from the initial state. If x is not present, there shall be no delay.

- where d is a character indicating the direction in which the text is moving with the following possibilities:
l = left moving text
r = right moving text
- where w is a number indicating the width, in pixels, of the window in which the 'text' is to be moved/scrolling
- where s is a number indicating the number of pixels that the text shall move at the defined rate 'r'.
- where r is a number indicating the time, in tenths of a second, between two steps 's'.
- where text is the array of characters that is to be moved/scrolling. The text shall be case-sensitive.

This tag allows the moving (or scrolling) of the text indicated within the brackets. The different parameters indicate different functions that can be associated with the moving/scrolling of text.

For left moving/scrolling, the window shall be initialized with the first character of the text aligned with the left edge of the window.

For right moving/scrolling, the window shall be initialized with the last character of the text aligned with the right edge of the window.

Circular moving/scrolling is the continuous display of the indicated text, including all spaces shown within the text. In this case, the text will appear moving across the window as though multiple copies of the text were appended to itself. The character spacing is applied between the apparent multiple copies of text.

Linear moving/scrolling is the intermittent display of the indicated text, including all spaces shown within the text. In this case, the window initialized with beginning of the text appearing in the window, then moving across the window until all characters have been displayed. The process will repeat again after the indicated delay time defined by the value x when the t-parameter is lx.

The text can only be moved over one line. If text is supposed to be moved over more than one line, then this tag needs to be indicated for each line.

If this tag is unsupported, or if the display would appear incorrect for the selected parameters (e.g., using a value of 's' equals one (1) on a character matrix sign), the sign should report an unsupportedTagValue error.

If a character immediately precedes or follows a moving text tag, the current character spacing shall be inserted between the character and the moving text window.

If necessary, the number of pixel columns in the final shift of a linear move (before repeating) will be reduced such that the last column of the moving text will appear in the rightmost column of the window for left moves or in the leftmost column of the window for right moves.

EXAMPLES:

Although the printed examples show the text moving by whole character positions, the text displayed on the sign will shift by the amount in the MULTI tag (in the following examples, 1 pixel at a time).

To display the moving text "THIS IS A TEST" which moves circularly to the right within a window of 50 pixels and a rate of 1 pixel per 3 tenths of a second, the MULTI string could read:

"[mvcr50,1,3,THIS IS A TEST]"

Circular right scrolling:

```
[ IS A TEST]
[S IS A TES]
[IS IS A TE]
[HIS IS A T]
[THIS IS A ]
[TTHIS IS A]
[STTHIS IS ]
[ESTTHIS IS]
[TESTTHIS I]
[ TESTTHIS ]
[A TESTTHIS]
```

and so on..

(Note that the text string does not include any spaces (character 0x20) between the words TEST and THIS, however, the display will show inter-character spacing between TEST and THIS.)

To display the moving text "THIS IS A TEST" (no spaces before and after the text) which moves linearly (with a delay of 0.5 seconds) to the left within a window of 50 pixels and a rate of 1 pixel per 3 tenths of a second, the MULTI string could read:

```
"[mvl5l50,1,3,THIS IS A TEST]"
```

Linear left scrolling:

```
[THIS IS A ]
[HIS IS A T]
[IS IS A TE]
[S IS A TES]
[ IS A TEST]
```

<0.5 sec delay occurs here>

```
[THIS IS A ]
[HIS IS A T]
```

and so on..

The following is an example of what occurs when the text field is smaller than the specified window for a left moving linear effect. Because all of the text in the MULTI string has been displayed, the text does not move.

```
"[mvl5l50,1,3,TEST]"
```

Linear left scrolling:

```
[TEST      ]
```

<0.5 sec delay occurs here>

```
[TEST      ]
```

The following is an example of what occurs when the text field is smaller than the specified window for a right moving linear effect.

```
"[mvl5r50,1,3,TEST]"
```

Linear right scrolling:

```
[      TEST]
```

<0.5 sec delay occurs here>

```
[      TEST]
```

The following is an example of how to move text where a small string of characters is to be scrolled through a larger window. For readability in this example an asterisk represents the space character.

“[mvl5r50,1,3,*****TEST****]”

Linear right scrolling:

[TEST****]

[*TEST***]

[**TEST**]

[***TEST*]

[****TEST]

[*****TES]

[*****TE]

[*****T]

[*****]

<0.5 sec delay occurs here>

[TEST****]

[*TEST***]

and so on...

3.4.13 Page Time

-Change the sentence before the examples to:

This standard does not require minimum page times values, however, the Controller must recognize the tag and take appropriate action by implementing functionality or by generating a `dmsMultiSyntaxError` with a value of `unsupportedTag`.

3.4.14 Spacing - Character

-Insert the following sentence before the examples to:

This standard does not require support of spacing character values, however, the Controller must recognize the tag and take appropriate action by implementing functionality or by generating a `dmsMultiSyntaxError` with a value of `unsupportedTag`.

4.1 to 4.20 and Section 5

-Change the content of 'REFERENCE' column within each of the tables to read:

NTCIP 1203

4.6 MESSAGE TABLE CONFORMANCE GROUP

-Change the table entry from

	Object or Table Name	Reference
	dmsMessageMsgStatus	TS3.6
<i>to</i>		
	Object or Table Name	Reference
	dmsMessageStatus	NTCIP 1203

4.11 ILLUMINATION/BRIGHTNESS CONFORMANCE GROUP

-Change the table entry from

	Object or Table Name	Reference
	dmsIllumBrightnessValuesStatus	TS3.6
<i>to</i>		
	Object or Table Name	Reference
	dmsIllumBrightnessValuesErrors	NTCIP 1203

4.13 AUXILIARY I/O CONFORMANCE GROUP

-Change the table to:

	Object or Table Name	Reference
	maxAuxIODigital	NTCIP 1203
	maxAuxIOAnalog	NTCIP 1203
	auxIOTable	NTCIP 1203
	auxIOPortType	NTCIP 1203
	auxIOPortNumber	NTCIP 1203
	auxIODescription	NTCIP 1203
	auxIOResolution	NTCIP 1203
	auxIOValue	NTCIP 1203
	auxIOPortDirection	NTCIP 1203